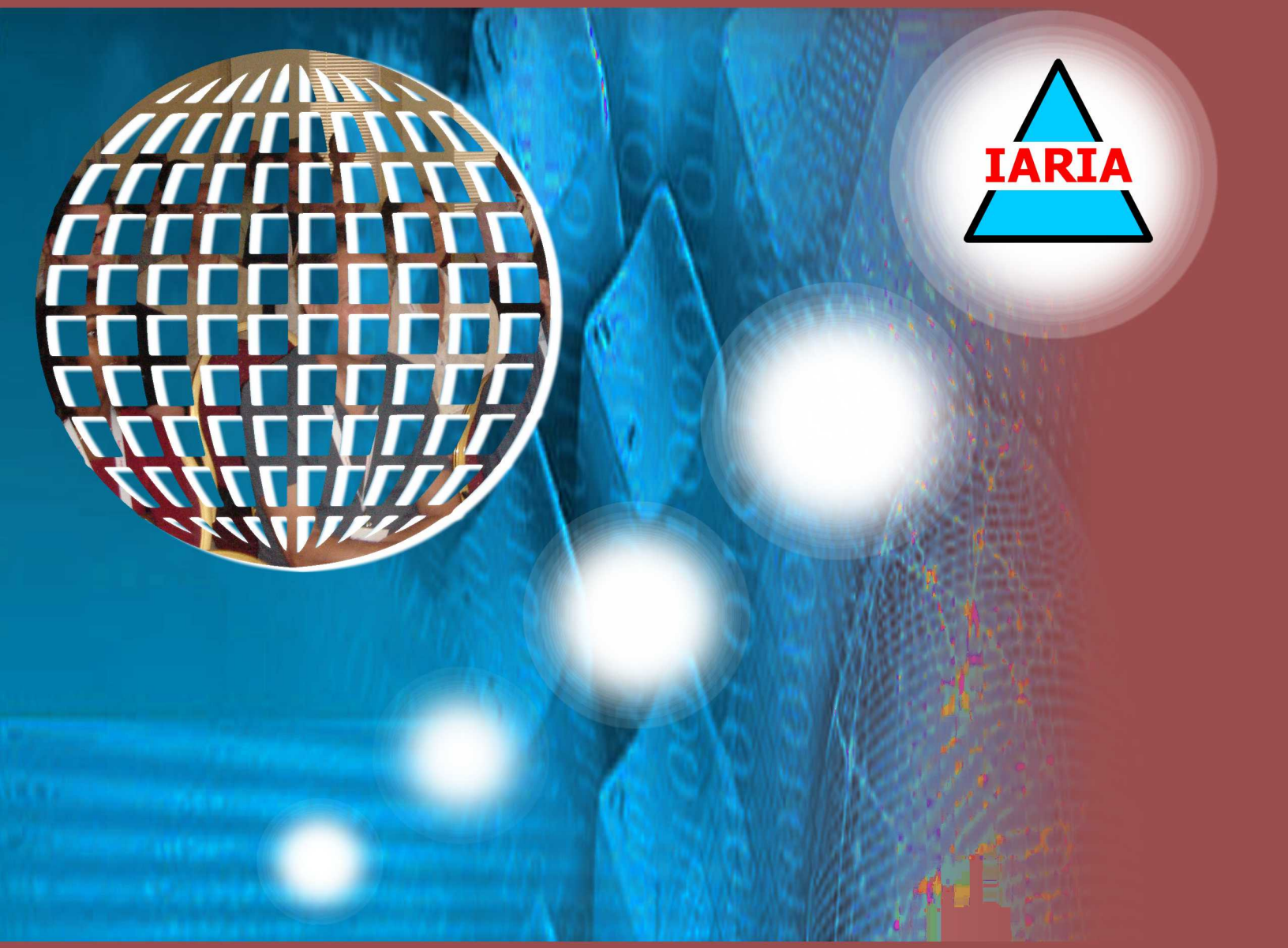
International Journal on Advances in Security



2024 vol. 17 nr. 1&2

The *International Journal on Advances in Security* is published by IARIA.
ISSN: 1942-2636
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Responsibility for the contents rests upon the authors and not upon IARIA, nor on IARIA volunteers, staff, or contractors.

IARIA is the owner of the publication and of editorial aspects. IARIA reserves the right to update the content for quality improvements.

Abstracting is permitted with credit to the source. Libraries are permitted to photocopy or print, providing the reference is mentioned and that the resulting material is made available at no cost.

Reference should mention:

International Journal on Advances in Security, issn 1942-2636
vol. 17, no. 1 & 2, year 2024, http://www.iariajournals.org/security/

The copyright for each included paper belongs to the authors. Republishing of same material, by authors or persons or organizations, is not allowed. Reprint rights can be granted by IARIA or by the authors, and must include proper reference.

Reference to an article in the journal is as follows:

<Author list>, "<Article title>"
International Journal on Advances in Security, issn 1942-2636
vol. 17, no. 1 & 2, year 2024, <start page>:<end page> , http://www.iariajournals.org/security/

IARIA journals are made available for free, proving the appropriate references are made when their content is used.

Sponsored by IARIA
www.iaria.org

Copyright © 2024 IARIA

International Journal on Advances in Security
Volume 17, Number 1 & 2, 2024

Editors-in-Chief

Hans-Joachim Hof,
- Full Professor at Technische Hochschule Ingolstadt, Germany
- Lecturer at Munich University of Applied Sciences
- Group leader MuSe - Munich IT Security Research Group
- Group leader INSicherheit - Ingolstädter Forschungsgruppe angewandte IT-Sicherheit
- Chairman German Chapter of the ACM

Editorial Board

Oum-El-Kheir Aktouf, Univ. Grenoble Alpes | Grenoble INP, France
Eric Amankwa, Presbyterian University, Ghana
Ilija Basicevic, University of Novi Sad, Serbia
Cătălin Bîrjoveanu, "Al.I.Cuza" University of Iasi, Romania
Steve Chan, Decision Engineering Analysis Laboratory, USA
Abdullah S. Al-Alaj, Virginia Wesleyan University, USA
El-Sayed M. El-Alfy, King Fahd University of Petroleum and Minerals, Saudi Arabia
Rainer Falk, Siemens Technology, Germany
Steffen Fries, Siemens AG, Germany
Damjan Fujs, University of Ljubljana, Slovenia
Hans-Joachim Hof, Technische Hochschule Ingolstadt, Germany
Gahangir Hossain, University of North Texas, Denton, USA
Fu-Hau Hsu, National Central University, Taiwan
Sokratis Katsikas, Norwegian University of Science and Technology - NTNU, Norway
Hyunsung Kim, Kyungil University, Korea
Dragana Krstic, University of Nis, Serbia
Yosra Lakhdhar, Digital Research Center of Sfax (CRNS) / CN&S Research Lab at SUP'COM, Tunisia
Petra Leimich, Edinburgh Napier University, UK
Shimin Li, Winona State University, USA
Yi Liu, University of Massachusetts Dartmouth, USA
Giuseppe Loseto, LUM "Giuseppe Degennaro" University, Italy
Mohammadreza Mehrabian, South Dakota School of Mines and Technology, USA
Carla Merkle Westphall, Federal University of Santa Catarina (UFSC), Brazil
Aleksandra Mileva, Goce Delcev University, Republic of N. Macedonia
Vasudevan Nagendra, Sekyurity AI, USA
Brajendra Panda, University of Arkansas, USA
Paweł Rajba, University of Wroclaw, Poland
Danda B. Rawat, Howard University, USA
Claus-Peter Rückemann, Universität Münster / DIMF / Leibniz Universität Hannover, Germany
Antonio Ruiz Martínez, University of Murcia, Spain
Rocky Slavin, University of Texas at San Antonio, USA

Pedro Sousa, University of Minho, Braga, Portugal
Miroslav Velev, Aries Design Automation, USA
Cong-Cong Xing, Nicholls State University, USA

International Journal on Advances in Security
Volume 17, Numbers 1 & 2, 2024

CONTENTS

pages: 1 - 10
Pixels versus Privacy: Leveraging Vision-Language Models for Sensitive Information Extraction
Sergej Schultenkämper, Bielefeld University of Applied Sciences and Arts, Germany
Frederik Simon Bäumer, Bielefeld University of Applied Sciences and Arts, Germany

pages: 11 - 28
Securing Confidential Data for Distributed Software Development Teams: Encrypted Container File
Tobias J. Bauer, Fraunhofer Institute for Applied and Integrated Security, Germany
Andreas Aßmuth, Ostbayerische Technische Hochschule Amberg-Weiden, Germany

pages: 29 - 43
Joining of Data-driven Forensics and Multimedia Forensics - Practical Application on DeepFake Image and Video Data
Dennis Siegel, Otto-von-Guericke-University Magdeburg, Germany
Christian Kraetzer, Otto-von-Guericke-University Magdeburg, Germany
Stefan Seidlitz, Otto-von-Guericke-University Magdeburg, Germany
Jana Dittmann, Otto-von-Guericke-University Magdeburg, Germany

pages: 44 - 52
Monitoring Physical-World Access of Virtual Automation Functions
Rainer Falk, Siemens AG, Germany
Steffen Fries, Siemens AG, Germany

pages: 53 - 71
Can Secure Software be Developed in Rust? On Vulnerabilities and Secure Coding Guidelines
Tiago Espinha Gasiba, Siemens AG, Germany
Sathwik Amburi, Siemens AG, Germany
Andrei-Cristian Iosif, Siemens AG, Germany

pages: 72 - 81
ChatIDS: Advancing Explainable Cybersecurity Using Generative AI
Victor Jüttner, Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, Germany
Martin Grimmer, Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, Germany
Erik Buchmann, Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, Germany

pages: 82 - 91
A Set of Social Requirements for Self-adaptive Privacy Management Based on Social Groups' Belonging
Angeliki Kitsiou, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece
Maria Sideri, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece
Aikaterini – Georgia Mavroeidi, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece
Katerina Vgena, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece

Eleni Tzortzaki, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece
Michail Pantelelis, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece
Stavros Simou, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece
Christos Kalloniatis, Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Greece

pages: 92 - 98
Supporting Cryptographic Algorithm Agility with Attribute Certificates
Steffen Fries, Siemens, Germany
Rainer Falk, Siemens, Germany

pages: 99 - 114
Redesign and Feasibility Verification of Access Control System Based on Correlation Among Files
Yuki Kodaka, The Graduate University for Advanced Studies, Japan
Hirokazu Hasegawa, National Institute of Informatics, Japan
Hiroki Takakura, National Institute of Informatics, Japan

Pixels versus Privacy: Leveraging Vision-Language Models for Sensitive Information Extraction

Sergej Schultenkämper
Bielefeld University of Applied Sciences and Arts
Bielefeld, Germany
sergej.schultenkaemper@hsbi.de

Frederik S. Bäumer
Bielefeld University of Applied Sciences and Arts
Bielefeld, Germany
frederik.baeumer@hsbi.de

***Abstract*—Threats to user privacy in Web 2.0 are abundant and can arise from various sources, including texts, geoinformation, images, videos, or combinations of these. To alert users of potential threats, it is crucial to gather all relevant information. However, aggregating user-specific information from various web platforms, including social networks, can be challenging due to the vast amount of data available, as well as issues with data quality and the numerous possible variants. This paper examines the capability of current Vision-Language Models to accurately identify relevant image data and extract sensitive information. To accomplish this, we developed our own dataset with diverse expressions for privacy attributes, based on the VISPR dataset. Furthermore, we address the challenge of synthetic images of people and its impact on our approach. Our findings suggest that these models are effective in pre-selecting relevant images, but there are limitations in information extraction.**

Keywords*—*Computer Vision*; *Privacy*; *Social Networks.

I. Introduction

In our previous study [1], we introduced a new dataset and proposed the use of Vision-Language Models (VLMs) to extract sensitive information from images. This current study expands on our previous work in two ways. Firstly, we evaluate two other State-Of-The-Art (SOTA) VLMs, BLIP-2 [2] and InstructBLIP [3]. Secondly, we analyze the models' ability to follow prompts and generate constrained answers.

Users leave active and passive footprints through nearly every activity on the Web [4]. This includes quite obvious information, such as images, texts, and videos that are knowingly uploaded by the users, as well as information that is passed on without the user's intervention, such as the IP addresses of the end devices or the user agent string. Furthermore, inherent information *hidden* in texts and images that are unknowingly published is difficult for users to keep track of.

In the past, this has been demonstrated several times in an impressive and media-effective manner, such as by the automatic identification of vacation announcements and the extraction of hidden Global Positioning System (GPS) image data on Twitter, which could, for example, be used to scout vacant properties for burglaries [5] or to reveal the running routes of soldiers on secret army bases, whose publication on sports portals revealed the exact location of the military installations [6]. It turns out that even small amounts of information can be dangerous in combination with other information [7].

In this paper, we focus on images with human attributes and documents that are shared on the Web by users on different platforms and due to different motivations. Some of these images are meant to highlight a tweet, others are vacation or profile photos, and some are simply memes or photos of animals. From this fact comes the first challenge: Every day, millions of images are uploaded that pose no risk to users' privacy. Finding relevant images that display human attributes and personal identification documents, revealing the complex dynamics of privacy and data exposure on the internet, is a challenging task in this vast and ever-growing dataset. Since we want to relate all the knowledge we get from an image to each other in order to extract reliable information, most classical image classification and segmentation methods fall short (e.g., limited domain, no extraction of class instances). We need an efficient, technical approach that enables sequential information extraction from images. For example, obviously, it is not sufficient to determine that a person has eyes and an eye color; rather, the specific eye color must also be reliably extracted (s. Figure 1).

We analyze existing datasets to expand and explore techniques for understanding vision and language. Recent developments in this field can assist in extracting information from images, including sensitive information. Our focus is on techniques that enable Visual Question Answering (VQA) and chat-based functionality, allowing the user to engage in a back-and-forth conversation with an image while maintaining context. Textual responses enable sequential questions and validations, including in-depth and control questions, to generate structured data for further processing. The primary goal is to create a comprehensive dataset that is suitable for privacy analysis with VQA. The dataset is used to evaluate the current SOTA VLM models, employing various specifically crafted prompts, in order to get optimal outcomes for extracting privacy-related variables.

All of these considerations are taking place as part of the Authority-Dependent Risk Identification and Analysis in online Networks (ADRIAN) research project, which is dedicated to exploring and developing machine-learning-based methods for detecting potential threats to individuals based on online datasets and Digital Twins (DT). For this purpose, we discuss related work in Section II and describe the research concept (Section III and Section IV) and results of our privacy VQA approach in Section V. Finally, we discuss our findings in Section VI and draw our conclusions in Section VII.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

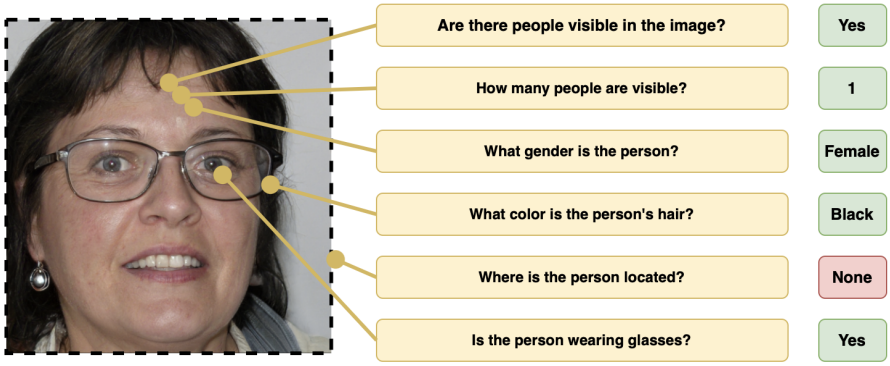


Fig. 1. Attribute extraction approach using VLMs.

II. Related Work

Here, we discuss the notion of DTs (s. Section II-A) in the context of cyber threats and present related privacy research and image datasets (s. Section II-B). Furthermore, we give an insight into VLMs (s. Section II-C).

A. Digital Twins in the context of cyber threats

The term DT is ambiguous and is used in a variety of areas in research and practice. It can be found in mechanical engineering, medicine, and computer science [8]. Developments in the field of Artificial Intelligence (AI) have given the term a wider usage. More generally, "DTs can be defined as (physical and/or virtual) machines or computer-based models that are simulating, emulating, mirroring, or 'twinning' the life of a physical entity, which may be an object, a process, a human, or a human-related feature" [8]. There are three levels of integration for DTs [8]: (a) *Digital Model*, (b) *Digital Shadow* and (c) *Digital Twin*. A Digital Model is the basic representation of a physical object or system in the virtual world, without any automatic information flow between the virtual and physical worlds. Changes in the physical object must be manually updated in the digital model. A Digital Shadow takes this further and involves a unidirectional automatic information flow from the physical world to the virtual world. Sensors measure information from the physical model and transmit signals to the virtual model. A complete DT exists when the virtual and physical environments communicate bidirectionally, with information flowing automatically between both environments. This allows the DT to accurately reflect the current state and development of its physical counterpart.

In the ADRIAN research project, we understand the term to mean the digital representation of a real person instantiated by information available on the Web [6]. In this context, the DT can never reflect the entire complexity of a real person but reproduces features that, alone or in combination with other attributes, can pose a threat to the real person. In this way, the DT makes it possible to model the vulnerability of a person and make it measurable. The modeling of DTs is based on freely available standards of the semantic web, such as Schema.org [9] and Friend Of A Friend (FOAF) [10]. This allows us to connect and extend DTs. At the same time, the sheer number of possible sources of information, the quality of the data, and a multitude of contradictory data make modeling challenging. However, studies show that a large amount of relevant information is knowingly and, to a large extent, unknowingly revealed by users themselves [7], [11]. It is precisely this fact that knowingly and unknowingly shared information on the Web can be merged and thus pose a threat to users, which we aim to highlight [6].

B. Privacy research and image datasets

According to DataReportal [12], the average number of social media accounts per Internet user worldwide was 7.5 in 2022. The various Online Social Networks (OSNs) use mechanisms to protect the privacy of users. For user-generated content, such as user profiles (e.g., on Facebook), or geo-information (e.g., on Twitter), there are settings that can help protect this data. With regard to images, there are so far barely any options for protecting private visual information [13].

That said, DeHart et al. [14] processed Twitter data by analyzing texts and images in a privacy context. Their study examines how users perceive privacy, how often privacy violations occur, and what threats exist on Twitter. As for image analysis, the images were classified into three risk categories: "*severe*", "*moderate*", and "*no risk*". As a result, images in the high-risk category were found to contain primarily license plates, job offers, and car keys. Moderate-risk images are mainly images of job references, school information, and promotion letters. The study confirms that, depending on age, users are differently concerned about explicit websites, financial theft, identity theft, and stalking. It also confirms that female and male participants are differently concerned about burglaries, explicit websites, and identity theft.

Work already exists here that aims to help users preserve their own privacy. For example, Orekondy et al. [13] proposed a so-called Visual Privacy Advisor. This tool aims to assist users in enforcing their privacy preferences and preventing the disclosure of private information. They first create a dataset by annotating 68 personal information in images based on the EU Data Protection Directive 95/46/EC [15] and the US Privacy Act of 1974. Next, they conduct a user study to understand

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the privacy preferences of different users with respect to these attributes. They publish the Visual Privacy (VISPR) Dataset, which contains 22,167 images with a total of 115,742 labels. Finally, they extract visual features using CaffeNet [16] and GoogleNet [17], and train a linear Support Vector Machine (SVM) model [18]. A final comparison between human and machine predictions of privacy risks on images shows an improvement in their model over human estimation.

In later work, Orekondy et al. [19] selected a subset of images from their VISPR dataset for pixel-level annotation. This time, they focus on attributes that can be used for redaction, so that the image is still useful. Reduction of a large building, such as a church, can make the image unusable. They propose the Visual Redactions Dataset, with 8,473 images annotated with 47,600 instances for 24 attributes. The attributes are divided into three categories: textual, visual, and multi-modal, which are then annotated. They also apply Optical Character Recognition (OCR) [20] from the Google Cloud Vision API to locate the text-based attributes. Furthermore, they apply Named Entity Recognition (NER) [21] to recognize entity classes from the texts. As for visual attributes, they apply models such as the Fully Convolutional Instance-Aware Semantic Segmentation Method (FCIS) [22] and OpenALPR to localize objects such as faces, persons, and license plates. Multi-modal attributes are a combination of visual and textual information. Due to the limited number of training examples and the large range of these attributes, they treat this as a classification problem. As a result, they propose a first model for automatic redaction of different private information.

Another system is presented by Spyromitros-Xioufis et al. [23]. This system performs privacy-aware classification of images. They created a dataset called YourAlert by asking users to provide privacy annotations for photos of their personal collections. The authors applied Latent Dirichlet Allocation (LDA) [24] to their corpus to identify the themes within annotations. In total, there were six topics related to privacy: "*Children*", "*Drinking*", "*Eroticism*", "*Relatives*", "*Vacation*", and "*Wedding*". They make the dataset publicly available, with a total of 1,511 images, covering 444 private and 1,067 public images. Finally, the VGG-16 model is applied to extract features, and then they compute a modified version of the semfeat descriptor. The trained semi-personalized models lead to performance improvements over a generic model trained on a random subset of the PicAlert dataset.

Another relevant dataset is VizWiz-Priv [25]. The dataset consists of images taken by people who are blind to better understand the disclosure of their data. This dataset is used to develop algorithms that can decide first whether an image contains private information and second whether a question about an image requires information about the private content of the image. A total of 8,862 regions, including private content, were tagged in the 5,537 images. When annotating the images, a distinction was made between private objects and objects that usually show private text. Images that show private objects consist of five categories, while images that contain private text consist of 14 categories.

C. VLM and LLMs

In recent years, several VLMs, such as Vision Transformer (ViT) [26], Contrastive Language-Image Pre-Training (CLIP) [27], and Bootstrapping Language-Image Pre-Training (BLIP) [28], have been published for multi-modal deep learning. These models can be used to address various challenges in Computer Vision (CV) and Natural Language Processing (NLP). ViT is a type of neural network architecture designed specifically for image classification tasks [26]. It is based on the transformer architecture used in NLP models and uses self-attention mechanisms to process the image pixels in a parallel manner, allowing it to learn a rich representation of the relationships between different regions of the image [26]. ViTs have shown promising results in a variety of image classification tasks and have also been applied to other computer vision tasks, such as object detection and segmentation.

CLIP is a deep learning model for cross-modal representation learning. It learns a representation between natural language text and visual input (e.g., images) by comparing the similarity of the different image-text pairs [27]. The model has been trained on a dataset of 400 million image-text pairs collected from publicly available sources on the Internet [27].

The goal of CLIP is to create a representation that can be used for a variety of tasks, such as image captioning, VQA, and text-to-image synthesis. CLIP is pre-trained on large amounts of text-image data and then fine-tuned on smaller task-specific datasets. This pre-training step helps the model learn a robust representation of the relationship between text and image, which can lead to improved performance on downstream tasks.

CLIP consists of two encoders: a text encoder and an image encoder. The text encoder takes in a natural language text and produces a high-dimensional representation of the text. The text representation is generated by passing the text through a pre-trained language model. In CLIP, the text encoder is initialized with the pre-trained Bidirectional Encoder Representations from Transformers (BERT) weights [29]. The image encoder takes in an image and produces a high-dimensional representation of the image. The image representation is generated by passing the image through a pre-trained Convolutional Neural Network (CNN) [30]. Here, CLIP uses a ViT or ResNet, depending on the task. The contrastive loss is used to train the encoders to generate similar representations for semantically related image-text pairs and dissimilar representations for semantically unrelated image-text pairs.

The authors of BLIP propose a new method to process noisy web data by bootstrapping the captions. It is called Captioning and Filtering (CapFilt) and improves the quality of the training data. Furthermore, they propose a multi-modal Mixture of Encoder-Decoders (MED), a multi-task model that can operate in one of three functionalities: unimodal encoder, image-grounded text encoder, and image-grounded text decoder [28]. The unimodel encoder for text and image is trained with an Image-Text Contrastive (ITC) loss. This functionality is the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

same as for the CLIP model pre-training. The image-grounded text encoder uses additional cross-attention layers to describe the interactions between image and speech and is trained with an Image-Text Matching (ITM) loss to distinguish between positive and negative image-text pairs [28]. Image-grounded text decoders replace bidirectional self-attention layers with causal self-attention layers and use the same cross-attention layers and feed-forward networks as encoders. For those given images, the decoder is trained with a Language Modeling (LM) loss to generate labels [28].

BLIP-2 and InstructBLIP are able to combine current VLMs with current Large Language Models (LLM). BLIP-2 integrates frozen image encoders with LLMs for pre-training purposes. BLIP-2's architecture is built around the Querying Transformer (Q-Former), which effectively bridges the modality gap between the visual and linguistic components. Q-Former enables the leveraging of pre-trained, powerful vision and language models for downstream tasks like visual question answering and image-text generation without the need to update their weights. Its carefully designed two-stage pre-training procedure results in unparalleled effectiveness across various vision-language tasks, including visual question answering, image captioning, and image-text retrieval. The model's ability to perform zero-shot image-to-text generation with natural language instructions highlights its usefulness in situations requiring adaptive, multi-modal interaction. In addition, it has a significantly smaller number of trainable parameters than its predecessors. This functionality allows the model to engage in a back-and-forth conversation, generating responses to textual prompts in a context-aware manner. When using language models such as OPT and T5, the context length limitation in BLIP-2 is restricted to 512 tokens. It is important to take this limitation into account when developing detailed prompts and their possible responses. In addition, it is crucial to optimize responses for conciseness and relevance to prevent information truncation, as this can quickly affect the results of the model.

InstructBLIP enhances the capabilities of the pre-trained BLIP-2 model through a technique known as instruction tuning. This process leverages instruction-aware feature extraction facilitated by the Q-Former, effectively transforming data from 26 datasets into an instruction-based format. Additionally, InstructBLIP employs a strategy of balanced training dataset sampling to optimize learning. This approach improves zero-shot performance and, when fine-tuned for specific tasks, leads to SOTA results. InstructBLIP is compatible with models like Vicuna [31], which has been fine-tuned using the Llama base model [32].

Llama focuses on advancing pretraining and fine-tuning methods to boost both performance and safety in language models. It introduces innovative training strategies such as Supervised Fine-Tuning (SFT) and Reinforcement Learning with Human Feedback (RLHF), aimed at aligning model outputs more closely with human preferences [33]. Llama's development focuses on safety and ethical use, achieving top performance in open-source LLMs while ensuring responsible deployment practices.

III. Methodology

Our privacy VQA approach (s. Figure 2) follows a structured approach starting with the VISPR dataset, leading through: (1) Data Preparation, (2) Modeling, and (3) Evaluation.

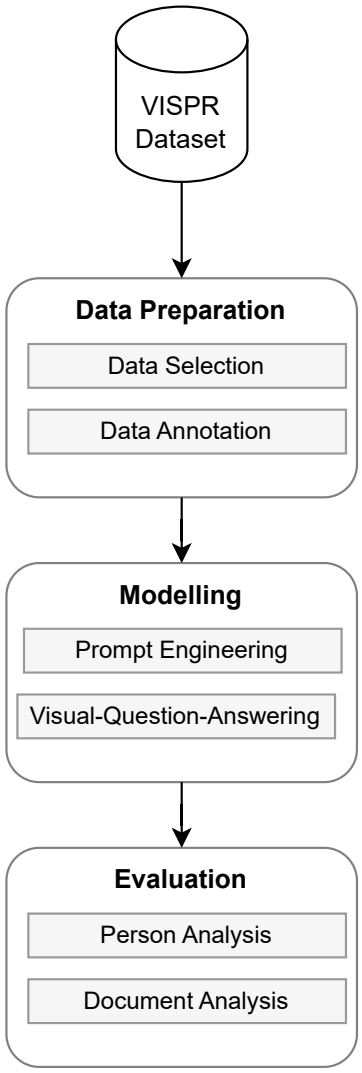


Fig. 2. Privacy analysis approach for images.

In (1), we initiate our study with the VISPR dataset, which encompasses a diverse range of privacy-sensitive attributes across 67 labels. This dataset serves as the foundation for our exploration into privacy-preserving VQA techniques [19]. We begin by selecting a subset of labels, as not all labels are suitable for VQA processing. For instance, textual information such as full names or places of birth is excluded. Our primary focus is on directly visible personal attributes. Additionally, we aim to evaluate how different types of documents can be identified using VQA. The selected list of attributes and documents is presented in Table I.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Data annotation follows data selection, where we use LabelStudio to manually annotate all chosen personal attributes and documents. We also define possible values for each attribute, as shown in Table I. In an initial experiment, we tested several alternative answer candidates and realized that if too many predictions are differentiated, such as "middle-aged" and "old-aged" adults, the annotation becomes very difficult because age can be very subjective. This phase also involves augmenting each category with an equal number of non-relevant images to enhance model robustness against unrelated prompts. For the documents, we grouped all images and used only one prompt and all documents available in the dataset as answer candidates. To analyze the attributes of a person in the context of yes-or-no answers, we added to each category the same number of images that did not belong to that label. To do this, we used images from the VISPR dataset with "*a0_safe*" labels, which indicate images that do not belong to any of the existing labels. It is equally important to see how well the model performs on images that are unrelated to the prompt. The final step is to evaluate the VQA performance.

In (2), the focus is on prompt engineering, which is integral to the VQA part [34]. In the context of VQA, we utilize BLIP, BLIP-2, and InstructBLIP to detect and analyze personal attributes and documents. For BLIP, we maintain the ranking-style Question-Answering (QA) approach used in our previous study [1]. This approach utilizes a set of predefined answers and measures the degree of matching between visual information and candidate answers to output the final answer that is relevant to the image contents [35]. For BLIP-2 and InstructBLIP, we use various prompts to evaluate their impact on the zero-shot performance of VQA models. According to Jin et al. [28], prompts significantly affect zero-shot performance. We test different prompts, from simple ones like "*Identify the hair color: black, red, gray, blond, or brown?*" to more detailed ones like "*Examine the person's hair in the image and determine the color. Options are black, red, gray, blond, or brown. Choose the one that accurately describes the hair color.*" This involves testing variations in language to understand their influence on model accuracy for extracting personal attributes and identifying documents. We use the prompts listed in Table I and their corresponding answer candidates are presented in Table II.

In (3), for person analysis, it is crucial to determine how many people are present in the image. The selected attributes can only be reliably extracted from images containing only one person. To do this, we use the following prompts: (I) "*Are there people in the picture?*", (II) "*How many people are in the picture?*", and (III) "*Is the face of the person visible?*" By identifying the images we are able to process with further analysis, we combine our annotated dataset with the results from the models. Person analysis involves evaluating our VQA performance through meticulous person analysis, assessing the model's accuracy in identifying personal attributes and the presence of individuals in images. Document analysis involves an examination of our model's capability to accurately identify and classify various document types.

IV. Dataset

For our privacy analysis, we need to define the categories for the analyzed attributes. Our annotation process for images is defined with a focus on simplicity and clarity to ensure consistency and reliability. We categorize the attribute "age" based on the guidelines proposed by Geifman et al. [36], using three broad categories: "child" for individuals up to about 16 years of age, "adult" for individuals up to about 55 years of age, and "elderly" for individuals aged 55 years and over. We deliberately exclude more detailed age descriptors such as "middle-aged" and "old-aged" adults because the perception of age can be highly subjective and such granularity could complicate the annotation process.

TABLE I
Selected VISPR dataset attributes.

Attribute	Category	# of Img.
a1_age_approx	[*child*, *adult*, *elderly*]	1,711
a4_gender	[*male*, *female*]	1,863
a5_eye_color	[*blue*, *green*, *gray*, *brown*]	1,348
a6_hair_color	[*black*, *blond*, *brown*, *gray*, *red*]	1,759
a11_tattoo	[*yes*, *no*]	45
a12_semi_nudity	[*yes*, *no*]	247
a13_full_nudity	[*yes*, *no*]	11
a17_color	[*black*, *brown*, *white*]	1,914
a29_ausweis,	[*national identification card*,	47
a30_credit_card,	*credit card*,	97
a31_passport,	*passport*,	263
a32_drivers_license,	*driver's licence*,	70
a33_student_id	*student ID*]	70
a39_disability_physical	[*yes*, *no*]	41

Regarding skin and hair color, our classifications follow the categorizations described by Jablonski et al. [37]. We aim to maintain simplicity in our annotations; hence, we do not include excessively detailed or specific color values that could lead to ambiguity or inconsistency.

For the annotation of eye color, we rely on the classifications provided by Frost [38], ensuring that our categorizations are both accurate and straightforward. In summary, for all attributes, including age, skin color, hair color, and eye color, our approach is to use broad, well-defined categories. This method helps to avoid the potential complexity and subjectivity involved in more detailed classifications, thereby enhancing the consistency and reliability of our image annotation process. Also, for the color values, we keep it simple and leave out all values that are very uncommon.

As for the distribution of the images, the age group, gender, eye color, hair color, and skin color have a large number of images, with an average of 1,719 images. Following the five documents consisting of a national identification card, credit card, passport, driver's license, and student ID, there are much fewer images, with an average of 109. Further attributes of a person, such as tattoos, nudity, and physical disabilities, are the least covered, with an average of 86 images. To determine the categories, we create prompts for the VLMs (s. Table II).

Our approach involves identifying prompts and evaluating whether detailed prompts lead to better results. Additionally, we aim to determine how a model like Llama, with a focus

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II
ILLUSTRATIONS OF THE SPECIFIED PROMPTS.

Attributes and Prompts
a1_age_approx **P1**: How old is the person? **P2**: What is the approximate age category of the person in the image?
a4_gender **P1**: What is the gender of the person? **P2**: What gender does the person in the image appear to be?
a5_eye_color **P1**: Which color are the eyes of the person? **P2**: What is the predominant color of the person's skin in the image?
a6_hair_color **P1**: Which color is the hair of the person? **P2**: What is the hair color of the person in the image?
a11_tattoo **P1**: Does the person have a tattoo? **P2**: Is there a tattoo visible on the person in the image?
a12_semi_nudity **P1**: Is the person partially nude? **P2**: Does the image depict the person as being semi-nude?
a13_full_nudity **P1**: Is the person fully nude? **P2**: Does the image depict the person as being semi-nude?
a17_color **P1**: What is the skin color of the person? **P2**: What is the predominant color of the person's skin in the image?
a29_ausweis, a30_credit_card, a31_passport, a32_drivers_license, a33_student_id **P1**: Which document is in this picture? **P2**: Can you identify the type of document or card shown in the image?
a39_disability_physical **P1**: Does the person have a physical disability? **P2**: Can you identify any physical disability in the person depicted in the image?

on security, ensures this through evaluations and mitigation strategies for responsible interactions. This work also includes an assessment of our ability to use a model like Llama for processing privacy-related information. Prompt engineering is essential here, as it involves crafting queries that improve the VLMs' ability to accurately extract and classify information from images. We experiment with different prompt formulations to assess their efficacy in eliciting detailed and precise responses, thus enhancing the model's performance. In addition, we experimented with detailed prompts, which are not included in Table II for the sake of brevity. For each prompt, we included defined categories as potential answers to ensure clear and specific responses.

V. RESULTS AND EVALUATION

To evaluate the privacy VQA performance of BLIP, BLIP-2, and InstructBLIP, we used the precision, recall, and F_1 scores. As hardware, we used an A6000 graphics card. BLIP is the smallest model with a size of 1.54 GB, followed by BLIP-2 (flan-t5-xxl version) with 49.44 GB and the InstructBLIP model (Vicuna-13b version) with 49.49 GB. In terms of processing speed, BLIP was the fastest model, processing each attribute with three different prompts in 1:06 hours. BLIP-2 came in second at 2:26 hours, and InstructBLIP came in third at 3:15 hours. Regarding prompt evaluation, we found that BLIP-2 performed better with simple and concise prompts, while InstructBLIP showed better results with more detailed prompts. The following results are based on the prompt that achieved the highest F_1-score. The person detection results are shown in Table III. Our dataset for person detection consisted of 1,000 images, of which 46 were excluded due to ambiguity, such as not being visible in specific scenarios like driving a racing car. The performance for detecting persons was highly reliable, with an F_1-score of 0.9658, as shown by the InstructBLIP model. However, detecting a single individual was the least effective, with an F_1-score of around 0.9021. For person detection, the models exhibit high precision and recall scores, indicating effective person identification. However, slight differences in performance emphasize the need for careful model selection based on specific requirements. InstructBLIP's superior performance in this category highlights its enhanced capability for accurately identifying and classifying people in varied imaging conditions.

TABLE III
PERSON DETECTION RESULTS.

	Precision	Recall	F_1-score	Support
Person Detection				
BLIP	0.9602	0.9602	0.9602	954
BLIP-2	0.9503	0.9599	0.9551	954
InstructBLIP	**0.9608**	**0.9707**	**0.9658**	954

TABLE IV
PERSON ATTRIBUTE RESULTS.

	Precision	Recall	F_1-score	Support
Age				
BLIP	**0.9137**	**0.9345**	**0.9240**	1,666
BLIP-2	0.9079	0.9286	0.9181	1,666
InstructBLIP	0.8838	0.9040	0.8937	1,666
Gender				
BLIP	**0.9725**	**0.9824**	**0.9774**	1,766
BLIP-2	0.9719	0.9807	0.9763	1,766
InstructBLIP	0.9697	0.9796	0.9746	1,766
Eye Color				
BLIP	**0.8132**	**0.8391**	**0.8260**	628
BLIP-2	0.7708	0.7879	0.7792	628
InstructBLIP	0.7404	0.7608	0.7504	628
Hair Color				
BLIP	**0.8798**	**0.8865**	**0.8831**	1,577
BLIP-2	0.7202	0.7231	0.7216	1,577
InstructBLIP	0.7988	0.8032	0.8010	1,577
Skin Color				
BLIP	**0.9501**	**0.9645**	**0.9573**	1,858
BLIP-2	0.8787	0.8889	0.8838	1,858
InstructBLIP	0.7637	0.7692	0.7665	1,858
Tattoo				
BLIP	0.8222	0.8222	0.8222	90
BLIP-2	0.8222	0.8222	0.8222	90
InstructBLIP	**0.8555**	**0.8555**	**0.8555**	90
Semi Nudity				
BLIP	0.7974	0.8009	0.7991	462
BLIP-2	**0.8297**	**0.8333**	**0.8315**	462
InstructBLIP	0.7780	0.7814	0.7797	462
Full Nudity				
BLIP	**0.9545**	**0.9545**	**0.9545**	22
BLIP-2	0.9090	0.9090	0.9090	22
InstructBLIP	**0.9545**	**0.9545**	**0.9545**	22
Disability Physical				
BLIP	0.7439	0.7439	0.7439	82
BLIP-2	0.8048	0.8048	0.8048	82
InstructBLIP	**0.8293**	**0.8293**	**0.8293**	82

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

For attribute classification, gender recognition showed remarkably high F_1 scores, especially for the BLIP model. This suggests that gender attributes are clearly represented and easier to recognize by these models. Conversely, attributes such as eye color and hair color presented more challenges, yet the scores were reasonably high, pointing towards the efficacy of these models in extracting and classifying detailed features. The tasks of detecting tattoos, semi-nudity, and full nudity showed varied results, with certain models like InstructBLIP demonstrating higher accuracy in tattoo recognition. This variability may stem from the inherently diverse nature of these attributes in real-world images, which can significantly affect model performance. Physical disability detection had the lowest F_1-score among the attributes, which could indicate the need for more specialized training or more representative data to improve model performance in this area. The relatively lower scores in this category highlight the challenges and the necessity for advanced model training techniques and more comprehensive datasets.

For document analysis, we utilized a dataset comprising 536 images of various documents. The InstructBLIP model struggled to generate structured answers, which were crucial for computing our evaluation metrics. This limitation appears to be related to the Llama model's inability to process personal data due to privacy restrictions. This issue arises because the images, such as driver's licenses or passports, contain sensitive information. The BLIP model performed worse than both BLIP-2. The results from the BLIP model, as shown in Table V, illustrate a varied performance across different types of documents. The model demonstrates reasonable accuracy with passports and credit cards but shows limitations when processing driver's licenses and national identification cards.

TABLE V
DETAILED RESULTS FOR DOCUMENTS BY BLIP MODEL.

	Precision	Recall	F_1-score	Support
Documents				
Credit Card	0.8557	0.8384	0.8468	99
Driver's License	0.5000	0.3723	0.4268	94
Nat. Ident. Card	0.2979	0.3043	0.3011	46
Passport	0.7719	0.9531	0.8529	213
Student ID	0.8000	0.8595	0.6788	95

A detailed breakdown of the performance metrics for the BLIP-2 model is provided in Table VI, further illustrating the advancements in document analysis technology.

TABLE VI
DETAILED RESULTS FOR DOCUMENTS BY BLIP-2 MODEL.

	Precision	Recall	F_1-score	Support
Documents				
Credit Card	0.9773	0.9053	0.9399	99
Driver's License	1.0000	0.6418	0.7818	94
Nat. Ident. Card	0.2866	0.9574	0.4412	46
Passport	0.9951	0.7739	0.8707	213
Student ID	1.0000	0.6818	0.8108	95

Regarding document analysis, the highest performance was achieved for credit cards, passports, student IDs, and driver's licenses, with F_1-scores of 0.9399, 0.8708, 0.8108, and 0.7818, respectively. However, the analysis of national identification cards resulted in a significantly lower F_1-score of 0.4412. The BLIP-2 model demonstrates superior performance across most document types when compared to the BLIP model. This is particularly notable in the precision and F_1-scores for driver's licenses.

Overall, the BLIP model, using the ranking QA style, achieved the highest scores for 6 out of 11 evaluated attributes. This demonstrates its efficiency and effectiveness, as it allows for predefined answers to be provided and ranked. However, it cannot perform detailed analysis in a chat-based setting, like the other two models. InstructBLIP excelled at identifying complex attributes such as tattoos, nudity, and disabilities, highlighting the strength of its advanced architecture and larger model size. BLIP-2 performed on the "semi nudity" attribute.

VI. DISCUSSION

All in all, the results show that our naive approach already leads to useful results, which can accelerate and improve the selection of relevant images. In particular, the important step of person detection has yielded good results. In the following, we discuss positive and negative examples (s. Figure 3, a–d). As can be noted, there are some positive hits where it could be difficult for an AI model to identify the exact number of people that are present in the image. Examples are Figure 3 (a), which shows a woman standing in front of a large mural of Michael Jackson, and Figure 3 (b), in which a little girl is standing in front of a mirror. In both cases, the image was classified as "*1 person*". As for the negative examples, there are many images of statues or emblems that, for example, were classified as images with one (s. Figure 3 , c) or more persons (s. Figure 3 , d). While this can be considered a *not completely wrong* classification, further experiments are necessary to find out how well real people can be distinguished from statues, for example.

For the personal attributes, all cases achieved very good and usable results. It should be noted here that the attributes "*age*" and "*hair color*" are very difficult to annotate. For "*age*", for example, it is very difficult to distinguish between an older adult and an elderly person without further knowledge. For "*eye color*", the annotators had to skip almost half of the images, despite the zoom function and high resolution of the images, because it was not possible to reliably determine the person's eye color. For the attributes with yes/no answers, "*nudity*" gave very good results, and "*tattoos*" gave decent results. Both of these attributes are fairly easy to annotate. In the case of "*semi-nudity*", it is difficult to determine where semi-nudity starts and where it ends. For example, according to the VISPR annotations, a man with a naked torso is semi-nudity; the applied BLIP model mostly did not detect these cases.

For document identification task, "*passport*" and "*credit card*" are well detected as they do not differ much between countries. "*Driver's licences*" and "*national identification*

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

(a) Positive Example #1

(b) Positive Example #2

(c) Negative Example #1

(d) Negative Example #2

Fig. 3. Positive and negative examples.

cards" were very poorly identified by the model. Here, a detailed observation reveals a high variance in the representation of these documents across countries. We are currently working on an approach that currently only takes German documents into account in order to be able to develop country-specific approaches in further work, if necessary. However, we assume that in these cases a fine tuning of the models is necessary.

The overall recognition precision is an important indicator of the success of the approach described here. However, there is one limitation that has not yet been sufficiently considered: AI-generated fake images of people, or, to put it another way, synthetic data [39]. Synthetic images are artificial images that are generated with the help of algorithms. There are various methods for generating synthetic images, which offer different advantages and challenges depending on the application and objective. Some common methods are GANs, diffusion models, VAEs, and neural style transfer [40]. These synthetic images must be detected [41] and ignored before the approach presented in this paper is applied, as this would significantly corrupt the resulting DTs. Next to that, the recognition of artificially generated privacy-relevant images is of great importance to ensure security on social media and to detect and prevent criminal activities that arise from these new technologies more quickly. We are giving high priority to the topic of synthetic data in further research work, as the influence of synthetic images on the quality of DTs is immense.

VII. Conclusion

This study presents valuable findings in the field of VLMs, demonstrating the efficiency of BLIP-based models in capturing and extracting predefined privacy attributes from images using a newly created dataset for privacy analysis. The evaluation shows that the model can extract attributes from images with high accuracy, achieving high micro-average values for person detection, attribute classification, and document analysis. Additionally, this study aimed to investigate whether an exemplar-based method for visual question answering (VQA) can assist in pre-selecting relevant images from a given dataset and extracting specific human attributes. This could be a crucial pre-processing step in our research project ADRIAN, which seeks to extract pertinent attributes for various OSN users and initiate a DT.

IntructBLIP and BLIP-2 were able to identify complex identifications of nudity, tattoos, and physical disability. We were able to show that the BLIP-based models in their original form, i.e., without further fine tuning, can already demonstrate a very good detection rate for the number of people in an image and also shine in the recognition of human attributes. However, in terms of documents, the model is only suitable for identifying specific documents, such as credit cards, and fails to detect country-specific types of documents.

For the future, there are already new models to analyze, such as CogVLM [42]. The promise of models like CogVLM lies in the integration of visual and linguistic data. While traditional VLMs often struggle with the challenge of deeply fusing these two types of data, CogVLM represents a promising advance. It demonstrates how deep fusion of these data can be achieved without compromising the performance of a pre-trained large-scale language model. By being able to identify and label objects in images and accurately extract their coordinates, CogVLM opens up new perspectives for processing and analyzing visual-linguistic information. This could not only improve accuracy and efficiency in existing use cases but also open the door to new applications in artificial intelligence.

Building on these initial findings, we plan to develop richer datasets to enhance the analysis of privacy vulnerabilities. For instance, the VISPR dataset can detect images containing sensitive elements, such as signatures, personal phone numbers, identifiable landmarks, or street signs. After detection, the next step involves extracting this sensitive information, which is crucial for assessing privacy risks. Accurately identifying specific data points, such as residential addresses, workplace locations, and direct contact phone numbers, allows for the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

assessment of potential privacy threats. This detailed information is particularly valuable in understanding the scope and scale of social engineering attacks. The goal is to leverage this enriched data to develop advanced predictive models that can foresee and neutralize such threats before they materialize. By taking this approach, we aim to proactively protect individuals from privacy breaches and reduce the risks associated with unauthorized data exploitation. This proactive approach is crucial in the digital age, where data privacy and security are of the utmost importance.

ACKNOWLEDGMENT

This research is funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr. dtec.bw is funded by the European Union – NextGenerationEU.

REFERENCES

[1] S. Schultenkämper and F. S. Bäumer, "Looking for a needle in a haystack: How can vision-language understanding help to identify privacy-threatening images on the web," in *Proceedings of the 18th International Conference on Internet and Web Applications and Services, ICIW 2023*. IARIA, 2023, pp. 1–6.

[2] J. Li, D. Li, S. Savarese, and S. Hoi, "BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models," in *Proceedings of the 40th International Conference on Machine Learning*, ser. ICML'23. Journal of Machine Learning Research, 2023.

[3] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. N. Fung, and S. Hoi, "InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[4] C. Iordanou, G. Smaragdakis, I. Poese, and N. Laoutaris, "Tracing Cross Border Web Tracking," in *Proceedings of the Internet Measurement Conference 2018*. New York, NY, USA: Association for Computing Machinery, 2018, pp. 329–342.

[5] M. B. Flinn, C. J. Teodorski, and K. L. Paullet, "Raising Awareness: An examination of embedded GPS data in images posted to the social networking site twitter," *Issues in Information Systems*, vol. 11, no. 1, pp. 432–438, 2010.

[6] F. S. Bäumer, S. Denisov, Y. Su Lee, and M. Geierhos, "Towards Authority-Dependent Risk Identification and Analysis in Online Networks," in *Proceedings of the IST-190 Research Symposium (RSY) on AI, ML and BD for Hybrid Military Operations (AI4HMO)*, A. Halimi and E. Ayday, Eds., 10 2021.

[7] F. S. Bäumer, N. Grote, J. Kersting, and M. Geierhos, "Privacy Matters: Detecting Nocuous Patient Data Exposure in Online Physician Reviews," in *International Conference on Information and Software Technologies*. Springer, 2017, pp. 77–89.

[8] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications," *IEEE Access*, vol. 7, pp. 167 653–167 671, 2019.

[9] R. V. Guha, D. Brickley, and S. Macbeth, "Schema.org: Evolution of Structured Data on the Web," *Communications of the ACM*, vol. 59, no. 2, pp. 44–51, 01 2016.

[10] N. Pankong, S. Prakancharoen, and M. Buranarach, "A Combined Semantic Social Network Analysis Framework to Integrate Social Media Data," in *Knowledge and Smart Technology (KST)*, 2012, pp. 37–42.

[11] F. S. Bäumer, J. Kersting, M. Orlikowski, and M. Geierhos, "Towards a Multi-Stage Approach to Detect Privacy Breaches in Physician Reviews." in *SEMANTICS Posters&Demos*, 2018.

[12] Data Reportal, 01 2022, https://datareportal.com/reports/digital-2022-global-overview-report, retrieved 2024/03/11.

[13] T. Orekondy, B. Schiele, and M. Fritz, "Towards a Visual Privacy Advisor: Understanding and Predicting Privacy Risks in Images," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 3706–3715.

[14] J. DeHart, M. Stell, and C. Grant, "Social Media and the Scourge of Visual Privacy," *Information*, vol. 11, no. 2, 2020.

[15] E. Directive, "95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data," *Official Journal of the EC, L 281*, vol. 38, pp. 31–50, 11 1995.

[16] Y. Jia *et al.*, "Caffe: Convolutional Architecture for Fast Feature Embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia*. New York, NY, USA: Association for Computing Machinery, 2014, pp. 675–678.

[17] C. Szegedy *et al.*, "Going Deeper with Convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[18] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[19] T. Orekondy, M. Fritz, and B. Schiele, "Connecting Pixels to Privacy and Utility: Automatic Redaction of Private Information in Images," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8466–8475.

[20] S. N. Srihari, A. Shekhawat, and S. W. Lam, *Optical character recognition (OCR)*. GBR: John Wiley and Sons Ltd., 2003, p. 1326–1333.

[21] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition," in *Proc. of the Seventh Conf. on Natural Language Learning at HLT-NAACL 2003*, 2003, pp. 142–147.

[22] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully Convolutional Instance-aware Semantic Segmentation," in *Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2359–2367.

[23] E. Spyromitros-Xioufis, S. Papadopoulos, A. Popescu, and Y. Kompatsiaris, "Personalized privacy-aware image classification," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. New York, United States: Association for Computing Machinery, Inc, 06 2016, pp. 71–78.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[25] D. Gurari *et al.*, "VizWiz-Priv: A Dataset for Recognizing the Presence and Purpose of Private Visual Information in Images Taken by Blind People," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 939–948.

[26] A. Dosovitskiy *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *International Conference on Learning Representations*, 2021.

[27] A. Radford *et al.*, "Learning Transferable Visual Models from Natural Language Supervision," in *Proceedings of the 38th International Conference on Machine Learning*, vol. 139. PMLR, 07 2021, pp. 8748–8763.

[28] W. Jin, Y. Cheng, Y. Shen, W. Chen, and X. Ren, "A Good Prompt Is Worth Millions of Parameters? Low-resource Prompt-based Learning for Vision-Language Models," in *Proceedings of the 60th Annual Meeting of the ACL (Volume 1: Long Papers)*. Dublin, Ireland: ACL, 05 2022, pp. 2763–2775.

[29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the ACL: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: ACL, 06 2019, pp. 4171–4186.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, 05 2015.

[31] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica, "Judging llm-as-a-judge with mt-bench and chatbot arena," 2023.

[32] H. Touvron *et al.*, "Llama 2: Open Foundation and Fine-Tuned Chat Models," 2023.

[33] L. Ouyang *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 730–27 744, 2022.

[34] Y. Lan, X. Li, X. Liu, Y. Li, W. Qin, and W. Qian, "Improving Zero-shot Visual Question Answering via Large Language Models with Reasoning Question Prompts," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 4389–4400.

[35] Y. Qiao, Z. Yu, and J. Liu, "Rankvqa: Answer Re-Ranking For Visual Question Answering," in *2020 IEEE International Conference on Multimedia and Expo (ICME)*, 2020, pp. 1–6.

[36] N. Geifman and E. Rubin, "Towards an age-phenome knowledge-base," *BMC bioinformatics*, vol. 12, no. 1, pp. 1–9, 2011.

[37] N. G. Jablonski, "The Evolution of Human Skin and Skin Color," *Annual Review of Anthropology*, vol. 33, no. 1, pp. 585–623, 2004.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[38] P. Frost, “European hair and eye color: A case of frequency-dependent sexual selection?” *Evolution and Human Behavior*, vol. 27, no. 2, pp. 85–103, 2006.

[39] L. Papa, L. Faiella, L. Corvitto, L. Maiano, and I. Amerini, “On the use of Stable Diffusion for creating realistic faces: from generation to detection,” in *2023 11th International Workshop on Biometrics and Forensics (IWBF)*, 2023, pp. 1–6.

[40] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image Style Transfer Using Convolutional Neural Networks,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2414–2423.

[41] H. Agarwal, A. Singh, and R. D, “Deepfake Detection Using SVM,” in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2021, pp. 1245–1249.

[42] W. Wang *et al.*, “Cogvlm: Visual expert for pretrained language models,” *arXiv preprint arXiv:2311.03079*, 2024.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Securing Confidential Data For Distributed Software Development Teams: Encrypted Container File

Tobias J. Bauer
Fraunhofer Institute for Applied and Integrated Security
Weiden, Germany
e-mail: tobias.bauer@aisec.fraunhofer.de

Andreas Aßmuth
Ostbayerische Technische Hochschule Amberg-Weiden
Amberg, Germany
e-mail: a.assmuth@oth-aw.de

***Abstract*—In the context of modern software engineering, there is a trend towards Cloud-native software development involving international teams with members from all over the world. Cloud-based version management services like GitHub are commonly used for source code and other files. However, a challenge arises when developers from different companies or organizations share the platform, as sensitive data should be encrypted to restrict access to certain developers only. This paper discusses existing tools addressing this issue, highlighting their shortcomings. The authors propose their own solution, Encrypted Container Files, designed to overcome the deficiencies observed in other tools.**

Keywords*—*Cloud-based software development; hybrid encryption; agile software engineering.

I. Introduction

Modern software development processes are agile, span organizational boundaries, and are carried out by teams whose members are not confined to one location. They produce Cloud-native software components that communicate with each other. However, there is strong interest in ensuring the confidentiality of the transmitted data. Usually, encryption algorithms fulfill this task, which is also true for storing confidential data, e.g., in a database. Often during the development of Cloud-native software components, passwords, private certificate keys, and symmetric encryption keys arise. On the one hand, they must be kept secret, but on the other hand, they must be managed in a sensible way and some authorized team members have to have access to these secrets. In the following, we refer to these types of data as *confidential data*. To address the emerging conflict of these two opposing objectives, this paper aims to extend our solution we presented in [1].

The practice of Continuous Integration (cf. [2]) is a long-lasting trend in software development [3][4]. The developed applications are then distributed and deployed automatically via Continuous Delivery and Continuous Deployment [3]. However, designing a secure deployment process or pipeline is difficult and requires special tooling as hinted in [5][6]. As a positive aspect, the awareness to build these automated processes in a secure way has grown over the past two years (cf. [4][7]). This leads to incorporating the security aspect into the development and operation or *DevOps* cycle and stressing this new component by naming the new approach *DevSecOps* [4]. Although best practice guides stress the need for securing confidential data, e.g., in [6] and [8], they lack providing a pipeline vendor-independent solution (cf. [6][9]) or advertise third party (cloud) services (cf. [8]). Some implementations have been found to expose secrets [10].

Software developing teams often use version control systems, e.g., *git* [11], to manage both the source code and other artifacts. The aforementioned practice of Continuous Integration requires the version control system to contain *all* artifacts, which – by definition – includes secrets we described earlier. However, storing confidential data without protective measures against unauthorized access would be grossly negligent and poses a serious hazard. There are tools available that offer the encryption of such confidential data before storing it within the version control system. However, we observed multiple drawbacks and shortcomings when using these tools. We present Encrypted Container File (ECF), our solution for a Cloud-based, hybrid-encrypted data storage aimed at software development teams. With ECF we extend the functionality of currently available tools and simultaneously overcome the found deficiencies.

The remainder of this paper is structured as follows: in Section II, we take a look at three existing tools and discuss their benefits and shortcomings. Next, we provide a high-level workflow description of the usage of ECF in Section III. This leads to the requirements and the file format description presented in Section IV. In Section V, we describe the operations of an ECF in detail. Since our initial presentation of ECF, we extended our work in various ways, which are described in Section VI. Finally, we conclude this paper in Section VII.

II. Related Work

Whilst researching available solutions for the problem described above, we came across two projects: *jak* and *git-crypt*. In our opinion, however, both have some disadvantages, which we will briefly describe below. Lastly, we argue that Pretty Good Privacy (PGP) or GNU Privacy Guard (GPG) can solve the problem but the usage is inconvenient and error prone.

A. jak

jak [12], a tool written in Python, allows the encryption of files with symmetric keys using the Advanced Encryption Standard (AES). With *jak*, keys can be generated and stored in a key file, which itself is not encrypted. If a special text file containing a list of the file names of the corresponding files is added to the repository, *jak* can perform automatic encryption and decryption with a single command. As is well known, the so-called key exchange problem cannot be solved using encryption with symmetric keys only. This problem also exists with *jak*, which relies exclusively on AES as stated before. The tool therefore cannot offer a solution to the key exchange problem, but leaves it up to the developers/users to take care of it themselves. The distribution of confidential data (keys) becomes increasingly challenging, especially as

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the size of the development team increases, which is why we believe that the practical use of *jak* is limited. Another issue with *jak* is that the content of confidential files is stored unencrypted on the developers' computers. The reason for this is that *jak* decrypts these files during checkout and re-encrypts them before committing. A prerequisite for the secure use of *jak* is therefore that external parties may only have read-only access to the repository. In addition, third parties must be prevented from accessing the developers' computers. If an attacker manages to gain access to such a computer, they can read the confidential information.

For further information on *jak*, please refer to the project documentation [12].

B. git-crypt

git-crypt [13] allows the encryption of files within a *git* repository with AES as well. This results in the same restrictions as with *jak* with regard to the necessary access restrictions for third parties. However, *git-crypt* offers a solution to the key exchange problem by using GPG. For this purpose, the public GPG keys of the recipients are added to the repository. When encrypting confidential files within the repository, *git-crypt* generates an asymmetrically encrypted key file for each recipient. Each authorized recipient can use this to access the symmetric key, which enables each of these recipients to decrypt the confidential files in this repository. *git-crypt* is implemented in a way that all confidential files are encrypted with the same symmetric key and this key must therefore be communicated to all recipients that are added to the repository. Unfortunately, this means that it is not possible to differentiate between different authorized recipients when accessing different confidential files. In our opinion, such a feature would be absolutely desirable. For example, confidential information about the production environment should only be accessible by the production team. However, there may also be other confidential files to which not only (some members of) the production team can have access. But this cannot be realized with *git-crypt*. As with *jak*, the contents of confidential files that are stored locally on the developer's computer are not secured. The reason for this is that *git-crypt* decrypts the confidential data during checkout (same as with *jak*). The integration of *git-crypt* into the mechanisms of *git* is optional, but recommended. With *git-crypt*, we also miss an option of being able to remove authorized users from the confidential files at a later date, for example if one of these developers leaves the project. In such a case, he or she should no longer have access to the confidential files. [13] justifies this by arguing that when using a version control system, a remote recipient can still access old versions of the repository and thus the confidential data stored in it. This is true, of course. However, it should also be borne in mind that a former developer should not have access to new certificates, passwords, etc. once the old ones have expired. Against the background of such changes of personnel, we consider this feature to be desirable. In addition, we recommend also changing the symmetric keys in this scenario.

C. Pretty Good Privacy

Another way to solve the problem would be to use PGP or GPG directly. To do this, one of the developers would generate a random session key to encrypt the confidential data. The random session key would then be encrypted with the public key of another developer. The file consisting of the encrypted session key and the symmetrically encrypted ciphertext could then be stored in the online repository. Only the aforementioned second developer would be able to access the confidential data.

As this brief description of PGP/GPG suggests, this solution does not scale very well. If the confidential data is required to be shared with multiple recipients, one such file per recipient must be created and stored in the repository. Of course, this is feasible in principle. However, it is important to note that each recipient must know which of these files is intended for them. In addition, it is absolutely impure that the same confidential data has to be stored in encrypted form in several instances. Changes to the content of the confidential data naturally require all these recipient-specific files to be recreated and stored. This makes this solution very challenging to use, and there is also the problem of keeping the content of all the confidential data's ciphertext files consistent. Just imagine that the confidential data of a project is spread over multiple files and each for different groups of recipients. Although handling with this solution is feasible in principle, it demands a great deal from everyone involved and is anything but convenient.

III. Application

In this section, we present a high-level description of multiple parties using an ECF. In order to illustrate the basic operations, we follow a common workflow involving various participants representing the different parties. The generic users Alice, Bob, and Charlie use a command line-based tool to interact with an ECF. This tool is also available as a part of this paper's accompanying source code[1]. Furthermore, all basic operations are depicted in Figure 1. Each subfigure corresponds to its analogously named subsection, e.g., Figure 1a visualizes Subsection III-A.

This use case-oriented view serves as a foundation for the distillation of requirements of the ECF format presented in Section IV. We describe the technical details in Section V, which follows the workflow presented in this section.

A. Creating an ECF

Alice is the head of an operations team working for a company. Her task is to securely store secrets, such as server certificates for the company's website. Furthermore, a subset of her coworkers as well as an automated deployment job should get access to these secrets. Third parties, i.e., all other coworkers as well as any other person, must not get access to the sensitive data. Alice therefore chooses the ECF format to protect and store the secret data.

[1] https://github.com/Hirnmoder/ECF

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

As a first step, Alice wants to create an ECF using the aforementioned command line tool (hereinafter referred to as the tool). To do so, Alice needs to generate a key pair consisting of a private and a public key. She may use this key pair for as many different ECFs as she likes, but she can also generate as many different key pairs as she likes. In order to keep this workflow description basic, we assume that Alice generates a single key pair using the tool. Since the private key must not be accessible to any person other than Alice, storing it unencrypted is not an option. This is why the tool asks for a self-chosen password which is then used to encrypt Alice's private key before storing it to disk.

In Figure 1a and all other subfigures in Figure 1, we depict a key pair as a keyring with two keys attached. Additionally, we always use the same color for a user as well as their keys and attributes, which is red for Alice.

After this initial step, Alice uses the tool to create an empty ECF. The tool then asks for her self-chosen password to get access to her private and public key. This is needed in order to add Alice as a recipient to the newly created ECF and to store its initial content, i.e., the secret server certificate, inside. To indicate that Alice is a recipient of that newly created ECF, we add an appropriately colored rectangle to it in Figure 1a.

When using a version control system, such as *git* [11], Alice may now add the ECF to the repository, commit the changes, and push it to a remote server. Since the sensitive data is encrypted within the ECF and no recipient but Alice was added to the ECF, only Alice is able to decrypt and access the secrets stored in that ECF.

B. Decrypting and Using an ECF

In order to access the data stored inside an ECF, Alice needs to be a recipient of the ECF in question. According to our workflow, Alice was added as a recipient to the ECF during its creation. She, thus, can decrypt and view the content of the ECF. In addition to that, Alice is also able to alter the recipient set as well as the sensitive data. Both of these actions are described later in Subsections III-C to III-E. This subsection is focused solely on the decryption and usage of an ECF.

Alice uses the tool to access the encrypted content. The tool first asks for Alice's self-chosen password and then decrypts the ECF using her private key. It then displays the decrypted content of the file for Alice to view and use. This operation is visualized in Figure 1b.

Usually, manual inspection of the content of an ECF is rare. It is much more likely that other processes, e.g., an automated deployment job, needs to use the secrets and therefore needs to access them. Such a process should run in a trusted environment, e.g., on a self-hosted deployment server. For the ECF, the process looks like any other recipient and, thus, Alice has to first add the process as a recipient to the ECF in order for it to be able to access the stored secret data. The procedure for adding a recipient is described in Subsection III-C. The process then can read (and possibly modify) the encrypted secrets inside the ECF and perform its task, e.g., deploying the company's website with the server certificate.

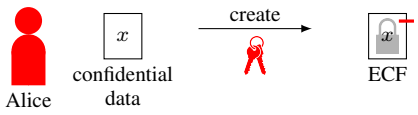
C. Adding Recipients

We now assume the following situation: Alice has two coworkers that also need access to the secrets, e.g., as a substitute when she is ill. Furthermore, the aforementioned automated deployment job also needs access to the secrets in order to fulfill its task.

As a preparatory step, her coworkers Bob and Charlie need to generate a key pair each, analogous to what Alice did in Subsection III-A. Then, Bob and Charlie export their public keys using the tool and sending those to Alice, e.g., via email. Bob and Charlie can perform these steps completely independently of each other and may use different computers and operating systems. Since Bob and Charlie keep their private keys private and only share their public keys, we use a single-key symbol for them in Figure 1c.

Since the automated deployment job is a process and not a person, Alice (or someone else she trusts for that matter) can generate a key pair for the process. This key pair can now be used to export the public key using the tool. The private key must be made available to the process while still keeping it safe from unauthorized access. This can be done by restricting access to the server the process is running on. This is also why the process should run within a trusted environment as stated in Subsection III-B.

Still, nobody but Alice has currently access to the content stored in the ECF Alice created in Subsection III-A. To change this, Alice now receives and checks the exported public keys of the recipients-to-be. She has to make sure that the received public keys were not tampered with during the transmission. A way to ensure authenticity is to compare, e.g., fingerprints of these public keys via a second channel. For example, we assume that Alice and Bob work in the same facility. They now can verify that the transmission of Bob's public key did not alter it by displaying the fingerprints on their laptop screens and comparing it visually.



(a) Alice creates an Encrypted Container File (cf. Subsection III-A).

(b) Alice decrypts and uses an Encrypted Container File (cf. Subsection III-B).

(c) Alice adds recipients Bob, Charlie and the automated deployment job (cf. Subsection III-C).

Figure 1. Overview of ECF operations according to Subsections III-A to III-C.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

After verifying the received public keys, Alice can now add Bob, Charlie, and the automated deployment job as recipients to the ECF. She uses the tool for each recipient addition. Since modifying the ECF also involves decrypting it, Alice is prompted for her self-chosen password by the tool in order to access her private key. The order in which Alice adds the recipients does not matter. Afterwards, she may delete the exported public keys since they are no longer of use.

As a last step Alice must make the updated ECF available to the other users. She may commit the changes and push them to a remote server using *git*. Bob, Charlie, and the automated deployment job now gained access to the ECF using their respective private keys and, thus, can perform the same actions Alice can, i.e., decrypting and using the ECF, add or remove recipients, and change the encrypted content. In Figure 1c, we indicate that Bob, Charlie and the automated deployment job are now recipients by appropriately colored rectangles.

D. Removing Recipients

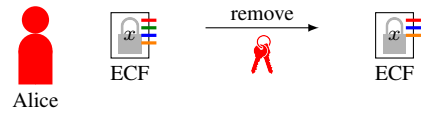
So far, we have tackled the recipient addition procedure. We now assume that Bob leaves the company and should, therefore, no longer have access to the content of the ECF. In this case, both, Alice and Charlie, can remove Bob from the recipient set – we opted for Alice to perform this step as she is the head of the team. This operation is also visualized in Figure 1d. Alice uses the tool to identify and remove Bob from the recipient set of the ECF. As with any operation, the tool asks for her self-chosen password beforehand. Alice then commits the changes and pushes them to a remote server.

Bob now looses access to the current and future versions of the ECF. However, Bob still has access to older versions. It is futile to withdraw access to older versions because (a malicious) Bob could have copied the secret content anyways when he still had access to. An ECF is designed to securely store secrets, such as access keys, private certificate keys, passwords, asymmetric and symmetric encryption keys, etc. As it is a good practice to rotate keys as described in [14], all secrets Bob once had access to will eventually expire. This is especially true for short-lived TLS certificates.

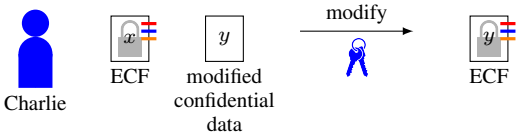
Nevertheless, it is advisable to only add recipients to an ECF if it is really necessary in the first place [15]. In fact, since each ECF is independent of other ECFs, the format allows for different recipient sets per file. Let us assume that Alice has two different secrets to share with different groups of people. Instead of creating a single ECF and adding all recipients to it, Alice should rather create two ECFs, one for each secret and only add the respective intended recipients to each.

E. Modifying the Content

The original ECF Alice created contains the server certificates for the company's website. Since these certificates are usually short-lived, they have to be renewed regularly. Alice's coworker Charlie – who got access to the ECF as described in Subsection III-C – now has the task of modifying the content of the ECF, which is depicted in Figure 1e.



(d) Alice removes recipient Bob (cf. Subsection III-D).



(e) Charlie modifies the content (cf. Subsection III-E).

Figure 1. *(Continued.)* Overview of ECF operations according to Subsections III-D and III-E.

Charlie uses the tool to modify the ECF and therefore is prompted to enter her self-chosen password. Then she can view and edit the content of the ECF. In order to fulfill her task, she removes the expired server certificate from the ECF and inserts the renewed one. In Figure 1e, we label the original content as x and the modified one as y. Last, she commits her changes using *git* and pushes them to a remote server to make the new version of that ECF available to all other recipients.

Now Alice, Charlie, and the automated deployment job can access the renewed certificate stored inside the ECF. Bob, on the other hand, cannot decrypt this ECF anymore as he was removed from the recipient set in Subsection III-D.

IV. Requirements Engineering and Internal File Structure

Based on the workflow described in Section III, we define requirements for the ECF format. Then, we use these requirements to formulate design goals for the definition of the file format. In the following subsections, we describe the file structure and used data types in detail. A technical description of the operations on the ECF format is given in Section V.

A. Use Case Requirements

We derive some requirements from the high-level workflow description given earlier. Furthermore, we take the features and weaknesses of the *jak* and *git-crypt* tools as well as the plain-PGP use into consideration. We have distilled the following requirements for the ECF format, cf. [1]:

1) The encryption of confidential data is mandatory,
2) possibility to modify the confidential data, i.e., the content is writable,
3) key distribution is no prerequisite,
4) decryption occurs not during checkout but on demand,
5) support for many recipients,
6) addition and removal of recipients,
7) minimal information gain for external parties,
8) a customizable set of recipients per file, and
9) be reasonably fast to be used in production.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

B. Design Goals

The requirements in mind, we have decided to define the following design goals for our ECF format, cf. [1]. The numbers in brackets refer to the enumeration in Subsection IV-A.

- Use of hybrid encryption (Items 1, 3 and 5),
- inclusion of recipient information to allow re-encryption on changes (Items 2, 5, 6 and 8),
- minimal information about recipients publicly available (Item 7),
- obfuscation of recipient information for respective external parties (Items 7 and 8),
- delivery of the associated software as a library for embedding into existing applications (Item 4), and
- optimization of the encryption and decryption procedure to take less than 100 ms on modern hardware for common confidential data types (Item 9), while also
- be flexible enough to support different cipher suites for future extensions (Items 1, 3 and 9).

C. Structure, Data Types, Storage Format, and Notation

Each ECF comprises a public header, a private body, and a public footer. The general structure of the ECF format is depicted in Figure 2. The public fields are unencrypted and therefore readable by anyone. The private body, however, is fully encrypted using a symmetric encryption scheme and can only be decrypted by the recipients of the ECF. Therefore, external parties do not get access to the confidential data stored within the body of the ECF. We describe the public and the private components of an ECF in more detail in Subsections IV-D and IV-E, respectively.

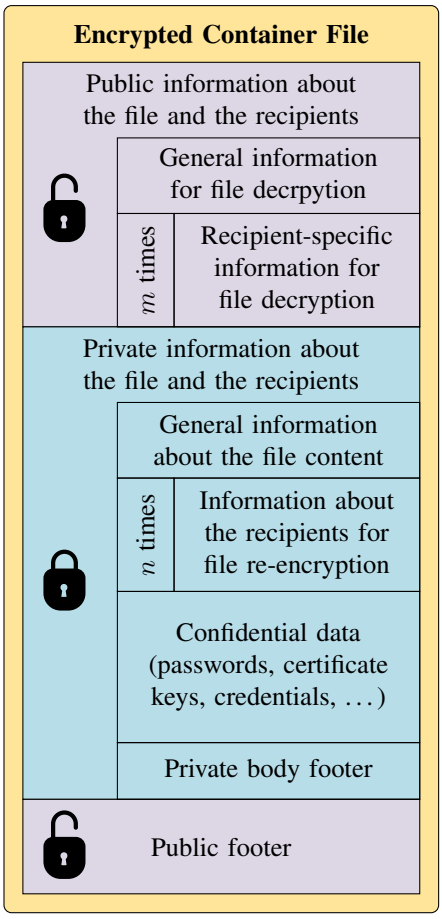


Figure 2. General structure of an Encrypted Container File, cf. [1].

Throughout the following descriptions of the public and private fields of an ECF, we use this notation to denote the data types of each field:

- `u32le` Unsigned 32 bit Integer, Little Endian
- `u8` Unsigned 8 bit Integer (byte)
- `s` String: length in bytes as `u32le`, then UTF-8 bytes without byte order mark (BOM)
- `x [n]` Array of type `x` with n entries, e.g., `u8 [16]` denotes an array of 16 bytes.

All variables referenced in this subsection are listed in Table I in Appendix B. Our solution employs four types of cryptographic algorithms to achieve the aforementioned goals:

- Diffie-Hellman-like Key Agreement/Exchange Algorithm
- Matching Signature Scheme (cf. Appendix A)
- Symmetric Encryption Scheme
- Suitable Hash Function

Each set of these algorithms is called a cipher suite. In our Proof of Concept (PoC) implementation, we define four cipher suites listed below. We designed the ECF format to be flexible with regards to the used cipher suite and in order to determine the cipher suite of a specific ECF, we assigned a unique identifier to each. In comparison to our previous work in [1], Cipher Suites III and IV were added to the PoC implementation with little effort, proving the flexibility of the ECF format. The four cipher suites are:

I)	Key Exchange:	X25519 [16]
	Signature:	Ed25519 [17]
	Symmetric Encryption:	AES-256-GCM [18][19]
	Hash Function:	SHA-256 [20]
	Identifier:	`0x01010101`
II)	Key Exchange:	X25519 [16]
	Signature:	Ed25519 [17]
	Symmetric Encryption:	AES-256-GCM [18][19]
	Hash Function:	SHA-512 [20]
	Identifier:	`0x01010102`
III)	Key Exchange:	X25519 [16]
	Signature:	Ed25519 [17]
	Symmetric Encryption:	AEGIS-256 [21]
	Hash Function:	SHA-256 [20]
	Identifier:	`0x01010201`
IV)	Key Exchange:	X25519 [16]
	Signature:	Ed25519 [17]
	Symmetric Encryption:	AEGIS-256 [21]
	Hash Function:	SHA-512 [20]
	Identifier:	`0x01010202`

The provided PoC contains a command line tool, which uses Cipher Suite II by default. When creating an ECF using the command line tool, the user can override the cipher suite to use. There is currently no built-in option to change the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

cipher suite afterwards. This is because some combinations of cipher suites – in particular the keys for key exchange and signing – may be incompatible. This problem does not affect the implemented ones, but may arise due to future extensions.

D. Unencrypted, Public Header and Footer Fields

All authorized recipients must be able to decrypt the file and, thus, must be able to acquire the symmetric encryption key used to encrypt the private body. Simultaneously, in order to achieve the goal of minimal public information about the recipients, the public header fields contain as little identifying information as possible. Our solution allows us to separate the information needed for decrypting and the information needed for encrypting. Whereas the former must be made public in order for the whole scheme to work, the latter can be included in the encrypted body and therefore be protected against unauthorized access.

The public header of an ECF comprises a general, recipient-independent part followed by m identically constructed recipient-specific blocks. Figure 3 shows all public header fields in their respective order. The recipient-independent fields contain general information about the ECF:

- CONTAINER VERSION of type `u32le`
  ECF format version; intended for future extensions; currently set to `0x00010000` for "Version 1.0"
- CIPHER SUITE of type `u32le`
  Identifier for chosen cipher suite; cf. enumeration in Subsection IV-C
- PUBLIC LENGTH of type `u32le`
  Length h of the public header in bytes
- PRIVATE LENGTH of type `u32le`
  Length b of the encrypted private body in bytes
- RECIPIENT COUNT of type `u32le`
  Number of recipients m in the public header
- SALT of type `u8 [16]`
  Salt value; cf. Subsections V-A and V-B
- SYMMETRIC NONCE of type `u8 [c]`
  Symmetric nonce value; cf. Subsections V-A and V-B; Length dependent on chosen cipher suite, usually $c = 12$

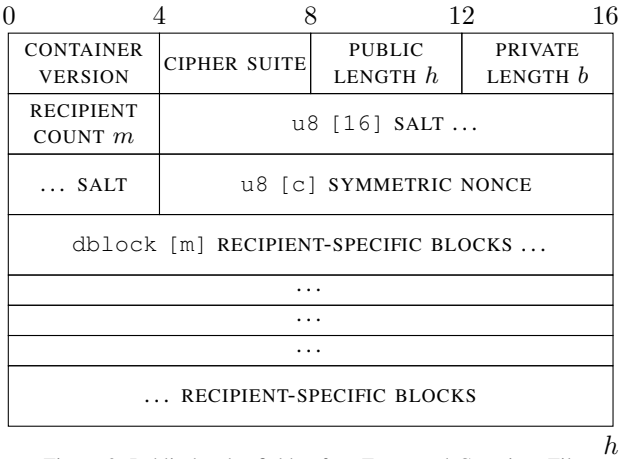


Figure 3. Public header fields of an Encrypted Container File.

All 4 byte-long fields are of type `u32le`. While the length of SALT is fixed to 16 bytes, the length of SYMMETRIC NONCE is dependent on the chosen cipher suite. However, since all implemented cipher suites use an Authenticated Encryption with Associated Data (AEAD) symmetric encryption scheme, we follow the recommendations in RFC 5116 [22] and set it to $c = 12$ bytes as shown in Figure 3 for the Galois/Counter Mode (GCM) [18] of AES. The AEGIS specification [21], however, requires $c = y$.

There are exactly m recipient-specific decryption blocks or `dblock`s following the recipient-independent part. The structure and length of these blocks is also highly dependent on the cipher suite. Figure 4 shows the general, three-part structure of these blocks. During the decryption procedure, recipients have to find their respective block by matching a unique identification number. This is why IDENTIFICATION TAG is the first entry in each block and has a fixed length of 16 bytes. It allows for up to $(2^8)^{16} = 2^{128}$, i.e., practically unlimited number of recipients while saving space compared to a full-length hash value. For illustration purposes, we decided to use the array lengths of the default Cipher Suite II. In this case, KEY AGREEMENT INFORMATION contains an ephemeral X25519 public key of length $a = 32$ bytes and SYMMETRIC PRE KEY 1 has, because of AES-256, a length of $y = 32$ bytes.

In Figure 4, we use o_i to denote the offset of a recipient-specific block $0 \leq i < m$ in bytes measured from the start of the file. It can be calculated as $o_i = i \cdot (16 + a + y) + o_0$. Since the blocks follow the recipient-independent part, $o_0 = 36 + c$. Furthermore, $o_m = h$ must hold in order for an ECF to be considered valid.

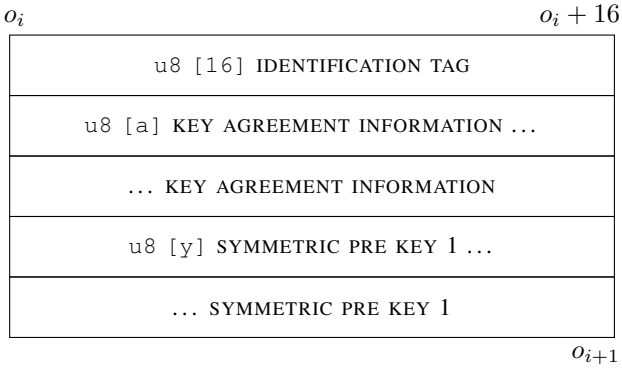


Figure 4. Recipient-specific block structure within the public header of an Encrypted Container File.

As shown in Figure 2, an ECF also contains a public footer. It consists of a single field: FILE HASH of type `u8 [d]`. Its length is dependent on the hash function of the chosen cipher suite. For example, $d = 64$ bytes applies to SHA-512. The field FILE HASH serves primarily to check the integrity of the file, e.g., after a copy or transfer action, and, thus, helps to detect accidental alterations. It does not increase the resilience against a malicious attacker as it can be recomputed easily after a deliberate alteration of other parts of the file.

The encryption procedure fills or generates all public header and footer fields and the decryption procedure uses them. Both operations are described in detail in Section V.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

E. Encrypted, Private Body Fields

After retrieving the symmetric encryption key using the public header information, an authorized recipient is able to decrypt the private body of the ECF. External parties, who are not recipients of that ECF, are unable to obtain the encryption key and, thus, unable to access any information stored within the body of that ECF. The body of an ECF consists of four parts as shown in Figure 2. The fields within the ECF body, which are depicted in Figure 5, are composed as follows:

- CONTENT TYPE of type `u32le`
  Describes the type of the confidential data
- PUBLIC HEADER HASH of type `u8 [d]`
  Hash value of the public header
- RECIPIENT COUNT of type `u32le`
  Number of true recipients $n \leq m$
- RECIPIENT INFORMATION BLOCKS of type `rblock [n]`
  Information about the recipients of that ECF
- CONTENT LENGTH of type `u32le`
  Length q of the confidential data in bytes
- CONTENT of type `u8 [q]`
  Confidential data
- PRIVATE HASH of type `u8 [d]`
  Hash value of the private body so far

The field CONTENT TYPE characterizes the type of confidential data stored within CONTENT. Our PoC implementation provides a built-in type of content, Binary Large Object (BLOB), using the identifier `0x00000001`. Since our implementation is flexible and extensible, future applications can and should define and handle their own types of content.

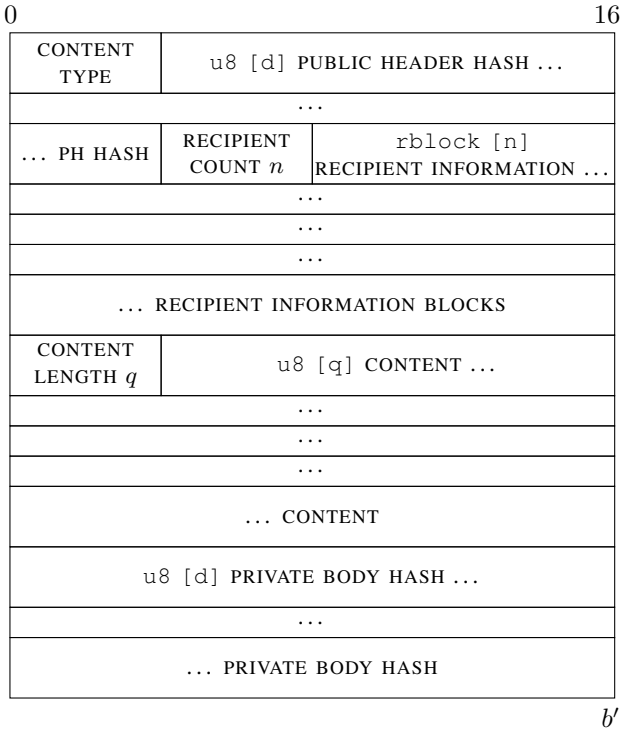


Figure 5. Private body fields of an Encrypted Container File.

Similar to the public footer described in Subsection IV-D, the lengths of the two fields PUBLIC HEADER HASH and PRIVATE BODY HASH is dependent on the chosen cipher suite. For Cipher Suite II, $d = 64$ bytes. Both fields aim to ensure integrity of the ECF and to detect unauthorized or unintended modifications of both the public header and the private body. For example, if one of the m public, recipient-specific blocks is altered preventing the corresponding recipient from obtaining the symmetric encryption key, all other recipients can detect this alteration. We opted to use fields for the hash values in order to allow for non-AEAD-based symmetric encryption schemes. This is also why we do not make use of the "associated data" option during encryption and decryption (cf. Section V).

Next, the private body contains information about the recipients in variable-sized recipient information blocks or `rblock`s. There are exactly n of these blocks present as denoted in Figure 5 and they contain the necessary information for file re-encryption. Allowing for $m \geq n$ is a design choice we made to hide the true number of recipients n from external parties, who can only read the public header field RECIPIENT COUNT and, thus, only obtain m. Furthermore, by storing the recipient-identifying information within the encrypted part of an ECF, we ensure that this information is only accessible by the recipients themselves and hidden from external parties, thus, pursuing the goal of minimal publicly available information about the recipients.

Each `rblock` contains three fields as shown in Figure 6:

- PUBLIC KEY of type `u8 [u]`
  Public key of the recipient;
  Length dependent on chosen cipher suite, e.g., $u = 32$
- NAME of type `s`
  Self-chosen name of the recipient, usually email address or another organization-specific identifier
- NAME SIGNATURE of type `u8 [g]`
  Signature over NAME;
  Length dependent on chosen cipher suite, e.g., $g = 64$

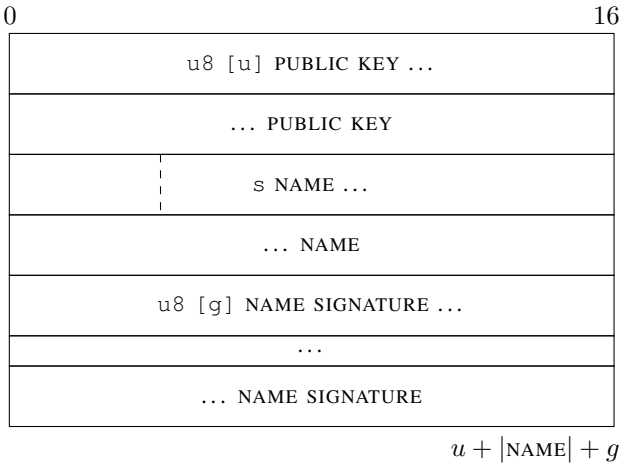


Figure 6. Recipient information block structure within the private body of an Encrypted Container File.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Since these blocks do not have a fixed size due to the variable-length self-chosen NAME, the offset measured from the start of the private body must be calculated by iterating through all previous blocks. The signature over the self-chosen name serves to assure that the person owning the associated private key has chosen the name and that no changes have been made to the name afterwards. The name is for information purposes only, e.g., when displaying the recipients or removing an existing recipient by name.

The main motivation for designing the ECF format was to securely store confidential data in a multi-recipient setting. The secrets, which may have any format the users want them to, are stored within the field CONTENT. Its theoretical maximum size $\max(q)$ in bytes is ultimately dependent on the number of true recipients n and the lengths of their self-chosen names as well as the chosen cipher suite. This is because the length of the encrypted private is limited to $\max(b) = 2^{32} - 1$ bytes $\approx 4\,\mathrm{GiB}$. We can calculate the theoretical maximum size of the confidential data as follows:

$$\begin{aligned}\max(q) &= \max(b) - \bar{q} - (b - b') \\ \bar{q} &= b' - q \\ &= 4 + d + 4 + \sum_{i=1}^{n} \left(u + |\text{NAME}_i| + g\right) + 4 + d \\ &= 12 + 2d + n\,(u + g) + \sum_{i=1}^{n} |\text{NAME}_i|\end{aligned}$$

The total length of all private fields is denoted by $b' \leq b$, whereas b denotes the length of the encrypted private body. The public header field PRIVATE LENGTH holds the value b, which may be greater than b' depending on the used symmetric encryption scheme. This is definitely true for AEAD-based algorithms. Therefore, $b - b' \geq 0$ describes the number of bytes the encrypted private body exceeds the unencrypted one. The helper variable $\bar{q} = b' - q > 0$ denotes the length of all private body fields except CONTENT.

We can show that $\bar{q} + (b - b')$ is negligible compared to $\max(b)$, and therefore the amount of data that can be stored within an ECF is sufficient for passwords, certificate keys, and similar confidential data. We assume $n = 1000$ recipients with an average name length of $|\text{NAME}_i| = 500$ bytes and using the default Cipher Suite II:

$$\begin{aligned}\bar{q} + (b - b') &= 12 + 2 \cdot 64 + 1000 \cdot (32 + 64 + 500) + 16 \\ &= 596157 \approx 2^{19} \ll 2^{32} - 1 = \max(b) \\ \implies \max(q) &\approx 2^{32} - 2^{19} \approx 2^{32}\end{aligned}$$

In practice, both n and the average name length will be significantly less, supporting our argument even more. Our strong assumption is that, in practice, the number of recipients of a single ECF does typically not exceed 20. Furthermore, using email addresses as recipient identification in the field NAME as we suggested earlier yields far shorter names than in the assumption above.

V. ECF OPERATIONS

This section describes the basic operations on an ECF. We follow roughly the workflow presented in Section III and describe the encryption procedure in Subsection V-A, the decryption procedure in Subsection V-B, and the modification of the recipient set and confidential data in Subsections V-C and V-D, respectively.

Nomenclature. We use a common notation for the following subsections. Since the ECF format is flexible with regards to the used cipher suite, we opted to use variable and function names that are not tied to a specific algorithm. Due to readability concerns, we moved the notation to Appendix B, and specifically to Tables II and III.

A. Creating and Encrypting an ECF

As described in Subsection III-A, Alice first needs to generate a key pair consisting of a private and a public key. All cipher suites in our PoC implementation allow to convert an Ed25519 key pair to a X25519 key pair as described in [23][24]. Therefore, we generate the key pair for signing and then convert it to a key pair for key exchange:

$$\begin{aligned}\left(\mathrm{sk}_\mathrm{A}^\mathrm{S}, \mathrm{pk}_\mathrm{A}^\mathrm{S}\right) &\leftarrow \mathrm{Gen}^\mathrm{S} \\ \left(\mathrm{sk}_\mathrm{A}^\mathrm{X}, \mathrm{pk}_\mathrm{A}^\mathrm{X}\right) &= \left(\mathrm{Convert}^{\mathrm{S}\to\mathrm{X}}\left(\mathrm{sk}_\mathrm{A}^\mathrm{S}\right), \mathrm{Convert}^{\mathrm{S}\to\mathrm{X}}\left(\mathrm{pk}_\mathrm{A}^\mathrm{S}\right)\right)\end{aligned}$$

Two components are essential to an ECF: the confidential data $\Omega \in \{0,1\}^{8q}$ and the set of recipients $\Psi = \{\psi_0, \ldots, \psi_{n-1}\}$. When creating a new ECF, both components are empty. As a next step, Alice adds herself to the recipient set and, therefore, must generate a recipient entry $\psi_a = \mathrm{pk}_\mathrm{A}^\mathrm{S} \| \text{NAME}_a \| t_a$. She may choose a name or use, for example, her email address as an identifier. Then she signs it:

$$t_a = \mathrm{Sign}\left(\mathrm{sk}_\mathrm{A}^\mathrm{S}, \mathrm{UTF8}\left(\text{NAME}_a\right)\right)$$

Alice now adds herself to the recipient set, i.e., $\Psi = \{\psi_a\}$. Following the basic workflow described earlier, Alice takes the byte representation of the confidential data in any format she likes and assigns it to Ω. As a final step, Alice encrypts the two components, thus, obtaining a valid ECF:

$$\mathcal{E} \leftarrow \mathrm{Enc}^\mathrm{ECF}\left(\Psi, \Omega\right)$$

The encryption procedure takes two inputs, the set of recipients Ψ and the confidential data Ω. It is important to note that parts of the cipher suite are selected implicitly by the construction of ψ_a. In the following, we assume that all cipher suite algorithms and their respective properties, e.g., the symmetric key length y and the nonce length c, are well defined according to the cipher suite Alice chose.

Additionally, the encryption procedure does not require any private key directly. However, the generation of ψ_a did require Alice's private key for signing, which is why our PoC implementation requests access to it. Yet, e.g., a pre-created recipient set Ψ, allows anyone to create an ECF without first generating a key pair. This operation, however, is not supported by our PoC implementation as the creator would not be a recipient and, thus, be unable to access the content of that ECF.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

The encryption procedure is defined as follows:

(1) Generate a symmetric key, a nonce, and a salt $\mathrm{k}_{\text{final}} \leftarrow \mathrm{Gen}^{\text{SYM}}, \alpha \leftarrow \mathrm{Random}(c), \theta \leftarrow \mathrm{Random}(16)$.

(2) Choose a random integer $m \geq n = |\Psi|$.

(3) For all $\psi_i \in \Psi, 0 \leq i < n$ generate r_i: ($R = \emptyset$ initially)

(a) Compute IDENTIFICATION TAG $\text{id_tag} = \mathrm{H}\left(\mathrm{pk}^{\mathrm{S}}_{\psi_i} \| \theta\right)[0, \ldots, 15]$.

(b) Generate an ephemeral key pair for key exchange $\left(\mathrm{sk}^{\mathrm{X}}_{\mathrm{e}}, \mathrm{pk}^{\mathrm{X}}_{\mathrm{e}}\right) \leftarrow \mathrm{Gen}^{\mathrm{X}}$.

(c) Convert the recipient's public key for signing to a public key for key exchange $\mathrm{pk}^{\mathrm{X}}_{\psi_i} = \mathrm{Convert}^{\mathrm{S}\rightarrow\mathrm{X}}\left(\mathrm{pk}^{\mathrm{S}}_{\psi_i}\right)$.

(d) Execute the key exchange algorithm: $\mathrm{ss} = \mathrm{Kex}\left(\mathrm{sk}^{\mathrm{X}}_{\mathrm{e}}, \mathrm{pk}^{\mathrm{X}}_{\psi_i}\right)$.

(e) Calculate $\mathrm{k}_{\text{pre2}} = \mathrm{H}\left(\mathrm{ss} \| \mathrm{pk}^{\mathrm{X}}_{\psi_i} \| \mathrm{pk}^{\mathrm{X}}_{\mathrm{e}}\right)[0, \ldots, y-1]$. Using this hash function construction instead of the shared secret directly is recommended in [25].

(f) Calculate $\mathrm{k}_{\text{pre1}} = \mathrm{k}_{\text{final}} \oplus \mathrm{k}_{\text{pre2}}$.

(g) Add $r_i = \text{id_tag} \| \mathrm{pk}^{\mathrm{X}}_{\mathrm{e}} \| \mathrm{k}_{\text{pre1}}$ to R.

(4) Generate $m - n$ deception entries as described in Subsection V-E, add them to R and shuffle or sort the elements $r_i \in R, 0 \leq i < m$ by id_tag.

(5) Write all public header fields into a temporary buffer $\mathcal{H}^*$, except for PRIVATE LENGTH as it is not known yet. Set its value b to `0xECFFC0DE` (ECF format "code") and compute the hash value of the public header:

$$\begin{aligned}\mathcal{H}^* &= \text{CONTAINER VERSION} \| \text{CIPHER SUITE} \| h \| \\ &\quad \texttt{0xECFFCODE} \| m \| \theta \| \alpha \| r_1 \| r_2 \| \ldots \| r_m \\ \beta_{\text{public}} &= \mathrm{H}\left(\mathcal{H}^*\right).\end{aligned}$$

(6) Write all private body fields unencrypted into a temporary buffer $\mathcal{B}'$ with $q = |\Omega|$ in bytes and compute the hash value of the private body:

$$\begin{aligned}\psi_i &= \mathrm{pk}^{\mathrm{S}}_{\psi_i} \| \text{NAME}_i \| t_i \\ \mathcal{B}' &= \text{CONTENT TYPE} \| \beta_{\text{public}} \| n \| \\ &\quad \psi_1 \| \psi_2 \| \ldots \| \psi_n \| q \| \Omega \\ \beta_{\text{private}} &= \mathrm{H}\left(\mathcal{B}'\right).\end{aligned}$$

(7) Append β_{private} to $\mathcal{B}'$ and then encrypt the buffer:

$$\mathcal{B} \leftarrow \mathrm{Enc}^{\text{SYM}}\left(\mathrm{k}_{\text{final}}, \alpha, \mathcal{B}' \| \beta_{\text{private}}\right).$$

(8) Update the previously written field PRIVATE LENGTH in $\mathcal{H}^*$ with its actual value $b = |\mathcal{B}|$ in bytes to obtain the final header $\mathcal{H}$ and compute the hash value over everything:

$$\begin{aligned}\mathcal{H} &= \text{CONTAINER VERSION} \| \text{CIPHER SUITE} \| h \| \\ &\quad b \| m \| \theta \| \alpha \| r_1 \| r_2 \| \ldots \| r_m \\ \beta_{\text{all}} &= \mathrm{H}\left(\mathcal{H} \| \mathcal{B}\right).\end{aligned}$$

(9) The output $\mathcal{E}$ of the ECF encryption procedure is

$$\mathcal{E} = \mathcal{H} \| \mathcal{B} \| \beta_{\text{all}}.$$

B. Decrypting an ECF

Only recipients of an ECF are able to decrypt it. In particular, having access to the private key of a recipient's key pair implies access to the encrypted content stored in the private body of an ECF. Alice has to use her private key for signing in order to decrypt the ECF and obtain the recipient set Ψ and the confidential data Ω:

$$(\Psi, \Omega) = \mathrm{Dec}^{\text{ECF}}\left(\mathrm{sk}^{\mathrm{S}}_{\mathrm{A}}, \mathcal{E}\right)$$

The decryption procedure is defined as follows:

(1) Load the public header fields and determine the container version as well as the cipher suite. Now h, b, m, θ, and α have values. Let $\mathcal{H}$ be the public header, $\mathcal{B}$ be the encrypted private body, and β_{all} the public footer.

(2) Check file integrity. If $\mathrm{Vrfy}^{\mathrm{H}}\left(\mathcal{H} \| \mathcal{B}, \beta_{\text{all}}\right) \stackrel{?}{=} 0$, *Exit*.

(3) Compute all necessary keys: $\mathrm{sk}^{\mathrm{X}}_{\mathrm{A}} = \mathrm{Convert}^{\mathrm{S}\rightarrow\mathrm{X}}\left(\mathrm{sk}^{\mathrm{S}}_{\mathrm{A}}\right)$, $\mathrm{pk}^{\mathrm{S}}_{\mathrm{A}} = \mathrm{Convert}^{\text{sk}\rightarrow\text{pk}}\left(\mathrm{sk}^{\mathrm{S}}_{\mathrm{A}}\right)$, $\mathrm{pk}^{\mathrm{X}}_{\mathrm{A}} = \mathrm{Convert}^{\text{sk}\rightarrow\text{pk}}\left(\mathrm{sk}^{\mathrm{X}}_{\mathrm{A}}\right)$.

(4) Compute $\text{id_tag} = \mathrm{H}\left(\mathrm{pk}^{\mathrm{S}}_{\mathrm{A}} \| \theta\right)[0, \ldots, 15]$.

(5) Search recipient-specific block $r_a = \text{id_tag} \| \mathrm{pk}^{\mathrm{X}}_{\mathrm{e}} \| \mathrm{k}_{\text{pre1}}$ with matching id_tag.

If not found: Alice is not a recipient of this ECF. *Exit*.

(6) Execute the key exchange algorithm: $\mathrm{ss} = \mathrm{Kex}\left(\mathrm{sk}^{\mathrm{X}}_{\mathrm{A}}, \mathrm{pk}^{\mathrm{X}}_{\mathrm{e}}\right)$.

(7) Calculate $\mathrm{k}_{\text{pre2}} = \mathrm{H}\left(\mathrm{ss} \| \mathrm{pk}^{\mathrm{X}}_{\mathrm{A}} \| \mathrm{pk}^{\mathrm{X}}_{\mathrm{e}}\right)[0, \ldots, y-1]$.

(8) Calculate $\mathrm{k}_{\text{final}} = \mathrm{k}_{\text{pre1}} \oplus \mathrm{k}_{\text{pre2}}$.

(9) Decrypt the private body $\mathcal{B}$ using the computed symmetric key and the nonce α into a buffer $\mathcal{B}^*$:

$$\mathcal{B}^* = \mathrm{Dec}^{\text{SYM}}\left(\mathrm{k}_{\text{final}}, \alpha, \mathcal{B}\right).$$

(10) Deconstruct $\mathcal{B}^* = \mathcal{B}' \| \beta_{\text{private}}$ into the private body fields.

(11) Compute the hash over the public header fields with b set to `0xECFFCODE` and verify it:

$$\begin{aligned}\mathcal{H}^* &= \text{CONTAINER VERSION} \| \text{CIPHER SUITE} \| h \| \\ &\quad \texttt{0xECFFCODE} \| m \| \theta \| \alpha \| r_1 \| r_2 \| \ldots \| r_m\end{aligned}$$

If $\mathrm{Vrfy}^{\mathrm{H}}\left(\mathcal{H}^*, \beta_{\text{public}}\right) \stackrel{?}{=} 0$, *Exit*.

(12) For each recipient $\psi_i \in \Psi$:

(a) Load and deconstruct ψ_i into $\mathrm{pk}^{\mathrm{S}}_{\psi_i}$, NAME_i, and t_i.

(b) Verify the signature:

If $\mathrm{Vrfy}^{\mathrm{S}}\left(\mathrm{pk}^{\mathrm{S}}_{\psi_i}, \mathrm{UTF8}\left(\text{NAME}_i\right), t_i\right) \stackrel{?}{=} 0$, *Exit*.

(13) Compute the hash over the unencrypted private body excluding the field PRIVATE BODY HASH and verify it:

$$\begin{aligned}\mathcal{B}' &= \mathcal{B}^*\left[0, \ldots, b' - d\right] \\ &= \text{CONTENT TYPE} \| \beta_{\text{public}} \| n \\ &\quad \psi_1 \| \psi_2 \| \ldots \| \psi_n \| q \| \Omega\end{aligned}$$

If $\mathrm{Vrfy}^{\mathrm{H}}\left(\mathcal{B}', \beta_{\text{private}}\right) \stackrel{?}{=} 0$, *Exit*.

(14) The outputs of the ECF decryption procedure are the recipient set Ψ and the confidential data Ω.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Performing Steps 2, 11, 12 and 13 is optional but strongly recommended in order to detect unintended and/or malicious modifications to the ECF. However, Step 12 takes a considerable amount of time and therefore may be omitted in certain cases. For example, a Continuous Integration (CI) pipeline usually contains a build stage and a test stage [2]. We assume that the whole pipeline runs in a trusted environment, e.g., a self-hosted server. Then, it is advisable to check the recipient's signatures as a prerequisite to the build stage, while this step may be skipped during testing in order to speed up the pipeline execution. Nevertheless, the execution times in question are in the range of milliseconds, cf. Subsection VI-C.

C. Adding and Removing Recipients

Adding recipients to an ECF or removing recipients from an ECF implies modifying the recipient set Ψ. We follow the basic workflow described in Subsections III-C and III-D. In order to obtain Ψ (and the confidential data Ω), Alice is required to be a recipient of the ECF $\mathcal{E}$, i.e. $\psi_a \in \Psi$.

In the following, we describe the recipient addition procedure for a single recipient: Bob. Adding more recipients, such as Charlie and the automated deployment job, is performed analogously.

Before Alice can add Bob to $\mathcal{E}$, her coworker has to generate his recipient entry ψ_b and, thus, has to generate key pairs analogous to what Alice did in Subsection V-A:

$$\begin{aligned}\left(\mathrm{sk}_\mathrm{B}^\mathrm{S}, \mathrm{pk}_\mathrm{B}^\mathrm{S}\right) &\leftarrow \mathrm{Gen}^\mathrm{S}\\ \left(\mathrm{sk}_\mathrm{B}^\mathrm{X}, \mathrm{pk}_\mathrm{B}^\mathrm{X}\right) &= \left(\mathrm{Convert}^{\mathrm{S}\to\mathrm{X}}\left(\mathrm{sk}_\mathrm{B}^\mathrm{S}\right), \mathrm{Convert}^{\mathrm{S}\to\mathrm{X}}\left(\mathrm{pk}_\mathrm{B}^\mathrm{S}\right)\right)\\ t_b &= \mathrm{Sign}\left(\mathrm{sk}_\mathrm{B}^\mathrm{S}, \mathrm{UTF8}\left(\mathrm{NAME}_b\right)\right)\\ \psi_b &= \mathrm{pk}_\mathrm{B}^\mathrm{S} \| \mathrm{NAME}_b \| t_b\end{aligned}$$

Bob now sends ψ_b to Alice. Alice performs the following steps to add him as a recipient to $\mathcal{E}$:

(1) Verify signature:
$$\text{If } \mathrm{Vrfy}^\mathrm{S}\left(\mathrm{pk}_{\psi_b}^\mathrm{S}, \mathrm{UTF8}\left(\mathrm{NAME}_b\right), t_b\right) \stackrel{?}{=} 0, \textit{ Exit.}$$
(2) Decrypt the ECF: $(\Psi, \Omega) = \mathrm{Dec}^\mathrm{ECF}\left(\mathrm{sk}_\mathrm{A}^\mathrm{S}, \mathcal{E}\right)$.
(3) Check, if Bob is already a recipient by comparing the public keys for signing: If $\exists\, \psi_i \in \Psi : \mathrm{pk}_{\psi_i}^\mathrm{S} = \mathrm{pk}_\mathrm{B}^\mathrm{S}$, *Exit*.
(4) Optionally check, if Bob's name is already present in the recipient set: If $\exists\, \psi_i \in \Psi : \mathrm{NAME}_i = \mathrm{NAME}_b$, *May exit*.
(5) Add Bob: $\Psi' = \Psi \cup \{\psi_b\} = \{\psi_0, \psi_1, \ldots, \psi_{n-1}, \psi_b\}$.
(6) Encrypt the modified ECF: $\mathcal{E}' \leftarrow \mathrm{Enc}^\mathrm{ECF}\left(\Psi', \Omega\right)$.

The name of a recipient is for information purposes only, which results in Step 4 to be optional. However, in order to avoid (human) confusion about the recipients of an ECF, it is advisable to enforce the constraint that all recipient names must be unique. Our PoC implementation does not allow duplicate names by default but can be configured otherwise.

We now assume that Alice added her coworkers Bob and Charlie as well as the automated deployment job as recipients to the ECF, thus, following the basic workflow described in Subsection III-C.

Next, Alice wants to remove Bob from the recipient set as motivated in Subsection III-D. She has to identify Bob in the recipient set either by his public key or his name. If Alice uses Bob's name to identify him, his name must be unique within the ECF. Alice performs the following steps to remove Bob from the recipient set:

(1) Decrypt the ECF: $(\Psi, \Omega) = \mathrm{Dec}^\mathrm{ECF}\left(\mathrm{sk}_\mathrm{A}^\mathrm{S}, \mathcal{E}\right)$.
(2) Find $\psi_b \in \Psi$ based on $\mathrm{pk}_\mathrm{B}^\mathrm{S}$ and/or NAME_b.
If no $\psi_i \in \Psi$ matches, Bob is not a recipient of $\mathcal{E}$, *Exit*.
(3) Optionally check, if Alice tries to remove herself from the ECF, i.e., Alice and Bob are identical.
If $\mathrm{pk}_\mathrm{B}^\mathrm{S} \stackrel{?}{=} \mathrm{pk}_\mathrm{A}^\mathrm{S} = \mathrm{Convert}^{\mathrm{sk}\to\mathrm{pk}}\left(\mathrm{sk}_\mathrm{A}^\mathrm{S}\right)$, *May exit*.
(4) Remove Bob: $\Psi' = \Psi \setminus \{\psi_b\}$.
(5) Encrypt the modified ECF: $\mathcal{E}' \leftarrow \mathrm{Enc}^\mathrm{ECF}\left(\Psi', \Omega\right)$.

As noted in Subsection V-A, the final Step 5 does not require any private keys for the encryption procedure. This implies that recipients of an ECF are able to remove themselves. However, this is generally an undesired feature and, therefore, the implementation should prevent such action by executing the optional Step 3. Our PoC implementation of the command line tool does not allow any recipient self-removal.

Finally, it must be noted that the restriction explained in Subsection III-D still holds: All former recipients are always able to access old versions of an ECF. As stated earlier, this is unpreventable and should be mitigated by using short-lived secrets and by applying the principle of minimal privilege.

D. Changing the Content

Modifying the confidential data stored within an ECF is similar to modifying the recipient set. Following the workflow described in Subsection III-E, we describe how the recipient Charlie can change the content of an ECF. We denote the resulting bit string after modification as $\Omega' = \mathrm{Modify}\left(\Omega\right)$ and Charlie's private key for signing as $\mathrm{sk}_\mathrm{C}^\mathrm{S}$:

(1) Decrypt the ECF: $(\Psi, \Omega) = \mathrm{Dec}^\mathrm{ECF}\left(\mathrm{sk}_\mathrm{C}^\mathrm{S}, \mathcal{E}\right)$.
(2) Modify the confidential data: $\Omega' = \mathrm{Modify}\left(\Omega\right)$.
(3) Encrypt the modified ECF: $\mathcal{E}' \leftarrow \mathrm{Enc}^\mathrm{ECF}\left(\Psi, \Omega'\right)$.

Although possible, it is uncommon to change both the recipient set Ψ and the content Ω in a single operation. This is why our PoC command line tool offers both actions only separately.

E. Generating $m - n$ Deception Blocks

In order to achieve the goal of obfuscating the information about recipients in the public header of an ECF, we take two measures. First, the ECF format stores the names of recipients in the encrypted private body, thus, hiding them from external parties. Furthermore, if the public keys of users are not published to the open public, i.e., only visible to members of the same institution or company, external parties cannot determine if a given user is a recipient of an ECF. This is because without knowledge of the public key, one cannot compute the id_tag (cf. Step 4 in Subsection V-B) and therefore not search for a recipient-specific block within the public header of a given ECF.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Second, we opted to obfuscate the true number of recipients n in the public header. This is why $m \geq n$ gets chosen randomly in Step 2 in Subsection V-A. Our PoC implementation chooses m randomly depending on n, such that $\max\{8, 2n\} \geq m \geq n$. After generating n valid recipient-specific blocks in Step 3 in the encryption procedure, $m - n$ deception blocks must be generated to fill the public header and obtain a valid ECF.

The main purpose of these deception blocks is to obfuscate n. Therefore, these blocks must be generated in a way, such that an external party is not able to distinguish between real blocks and deception blocks. This is why the deception blocks should not be random bit strings because there is a possibility that the outputs of the used cryptographic algorithms used to generate the real blocks suffer from statistical biases.

To avoid distinguishability, we suggest a generation procedure as follows:

(1) Generate a new deception key pair:
$\left(\mathrm{sk}_\phi^{\mathrm{S}}, \mathrm{pk}_\phi^{\mathrm{S}}\right) \leftarrow \mathrm{Gen}^{\mathrm{S}}, \quad \mathrm{pk}_\phi^{\mathrm{X}} = \mathrm{Convert}^{\mathrm{S}\to\mathrm{X}}\left(\mathrm{pk}_\phi^{\mathrm{S}}\right)$.

(2) Compute IDENTIFICATION TAG
$\mathrm{id_tag} = \mathrm{H}\left(\mathrm{pk}_\phi^{\mathrm{S}} \| \theta\right)[0, \ldots, 15]$.

(3) Generate an ephemeral key pair for key exchange
$\left(\mathrm{sk}_{\mathrm{e}}^{\mathrm{X}}, \mathrm{pk}_{\mathrm{e}}^{\mathrm{X}}\right) \leftarrow \mathrm{Gen}^{\mathrm{X}}$.

(4) Generate a random symmetric key $\mathrm{k}_\phi \leftarrow \mathrm{Gen}^{\mathrm{SYM}}$.

(5) Execute the key exchange algorithm:
$\mathrm{ss} = \mathrm{Kex}\left(\mathrm{sk}_{\mathrm{e}}^{\mathrm{X}}, \mathrm{pk}_\phi^{\mathrm{X}}\right)$.

(6) Calculate $\mathrm{k}_{\mathrm{pre2}} = \mathrm{H}\left(\mathrm{ss} \| \mathrm{pk}_\phi^{\mathrm{X}} \| \mathrm{pk}_{\mathrm{e}}^{\mathrm{X}}\right)[0, \ldots, y-1]$.

(7) Calculate $\mathrm{k}_{\mathrm{pre1}} = \mathrm{k}_\phi \oplus \mathrm{k}_{\mathrm{pre2}}$.

(8) Add $r_\phi = \mathrm{id_tag} \| \mathrm{pk}_{\mathrm{e}}^{\mathrm{X}} \| \mathrm{k}_{\mathrm{pre1}}$ to R.

However, generating deception blocks using this procedure is computationally more expensive than generating real blocks. In order to simplify the procedure, decrease the computational load, and therefore increase runtime performance, our PoC implementation uses a different one. This is only possible if the used cryptographic hash function generates truly random looking bit strings, i.e., the outputs of Steps 2 and 6 could have been generated at random instead. This implies that the output of Step 7 also could have been generated at random, thus, eliminating most of the computationally expensive parts of the procedure. The assumption regarding the hash function is justified considering the lengths of the inputs, cf. [26]. The simplified version of the deception block generation procedure reads as follows:

(1) Generate an ephemeral key pair for key exchange
$\left(\mathrm{sk}_{\mathrm{e}}^{\mathrm{X}}, \mathrm{pk}_{\mathrm{e}}^{\mathrm{X}}\right) \leftarrow \mathrm{Gen}^{\mathrm{X}}$.

(2) Generate a random IDENTIFICATION TAG:
$\mathrm{id_tag} \leftarrow \mathrm{Random}\,(16)$.

(3) Generate a random $\mathrm{k}_{\mathrm{pre1}} \leftarrow \mathrm{Random}\,(y)$.

(4) Add $r_\phi = \mathrm{id_tag} \| \mathrm{pk}_{\mathrm{e}}^{\mathrm{X}} \| \mathrm{k}_{\mathrm{pre1}}$ to R.

Of course, the generation of deception blocks does take some time, but the overhead is in the range of milliseconds as shown in Subsection VI-C.

VI. Recent Work and Improvements

Since the original release of the ECF format and the PoC implementation in [1], we extended and improved our work in different regards. The changes, enhancements, and new experiments are presented in the following subsections.

A. Building on Linux and Dockerfile

The original version of our PoC implementation was targeted towards the Windows operating system. It could be cross-compiled for Linux, but building on Linux was not possible. However, enabling compilation on Linux was always a goal. The changes we made since the original release allow our PoC implementation to be compiled on both operating systems and cross-compiled to the respective other operating system. Furthermore, using the *single-file deployment* [27] feature of recent .NET [28] versions makes our PoC implementation portable and easy to distribute without the need of installing further dependencies. Additionally to the build on Linux, we provide a *Dockerfile* for compiling in a *Docker* [29] container.

Software and its dependencies need to be kept up-to-date, which is why we updated the targeted runtimes. This means that our PoC implementation now can be compiled for *.NET 7* and *.NET 8*. We dropped support for *.NET 6* in order to be able to use newer language features. The aforementioned *Dockerfile* uses the most recent runtime. Next, we updated all libraries used for our PoC implementation. This includes the *Sodium* library [30], which received new features, such as the support for the symmetric encryption scheme family AEGIS [21][31]. The wrapper library *NSec* [32] was also updated in order to be able to use the new algorithms in *Sodium*.

B. New Cipher Suite

As mentioned in Subsection IV-C, we introduced two new cipher suites to the PoC implementation. Namely, the symmetric encryption scheme AEGIS-256 [21][31] was added to the set of cipher suites. Although the specifications for this new algorithm family are very recent and still work-in-progress, the authors of *Sodium* recommend using AEGIS-256 over AES-256-GCM [33]. It is important to note that our use of AES-256-GCM never reaches the theoretical limit stated in [33], because the encryption procedure generates the symmetric encryption key every time anew (cf. Subsection V-A).

Because of the changing nature of the AEGIS specification, we opted to keep the default cipher suite in our PoC implementation. However, users of our command line tool may select the cipher suite that best suit their needs. Furthermore, our performance experiments in the following Subsection VI-C hint that Cipher Suite IV using AEGIS-256 as symmetric encryption scheme offers currently the best performance.

The flexibility of the ECF format allows for different cipher suites to be implemented and the source code of our PoC implementation is structured in a way to support the easy integration of more cryptographic algorithms. Therefore, implementing new cipher suites is straight-forward. This makes the ECF format suitable in the long term since algorithms that are considered weak or are cracked in the future can be easily replaced by secure ones.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

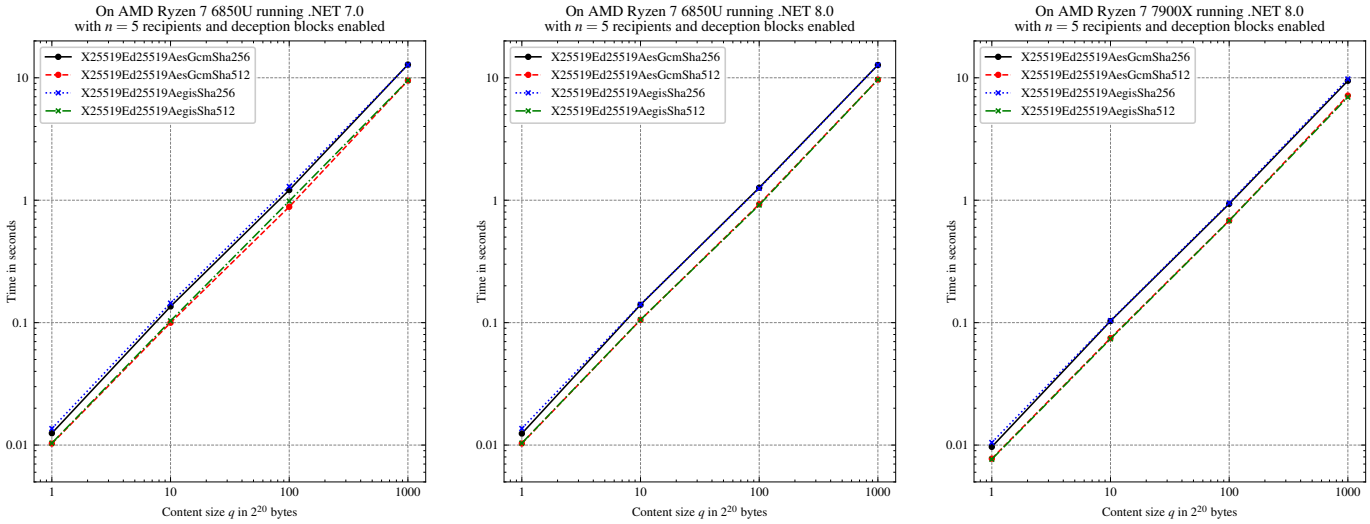


Figure 7. Average encryption time for different content sizes, cipher suites, and runs.

C. Performance Analysis

Extending our previous work [1], we took performance measurements of our PoC implementation on two different machines and in total using two different runtimes. The raw data of all runs is published alongside the code repository on GitHub. For the charts and tables in this paper, we aggregate the data by using average values. We provide the numeric values used to generate the performance charts in Appendix C.

The first experiment encrypts ECFs with $n = 5$ recipients and different content sizes ranging from 1 MiB to 1000 MiB. The results are depicted in Figure 7. We chose a log-log scale plot as both the content sizes and the amount of time it takes to encrypt the ECF increase in orders of magnitudes. Although not directly visible in the plot, there is a linear relationship between the content size and the processing time, which can be validated using the numerical values in Table IV in Appendix C. The differences between the runtimes *.NET 7.0* and *.NET 8.0* (left vs. middle chart) are negligible while a stronger processor (right chart) results in better absolute values.

Similarly, the chosen cipher suite makes a difference in performance. While the two symmetric encryption algorithms implemented in our PoC perform almost identical, the hash function SHA-512 (Cipher Suites II and IV) outperforms its sibling SHA-256 (Cipher Suites I and III) regardless of content size, number of recipients, runtime, and processor in all of our experiments. This is to be expected as the throughput of SHA-512 is up to 1.6 times [34] as much as the throughput of SHA-256. Since our PoC implementation uses the cryptographic library *Sodium* [30], which focuses on portability and therefore does not use any specialized processor instructions for SHA-256, our observed performance difference of about a factor of 1.5 seems reasonable.

In a second experiment, we focus on the decryption speed and re-use the same parameters of $n = 5$ and the content sizes of our previous experiment. Figure 8 shows the results of this second experiment. We can see the same linear relationship between the content size and the processing time. The absolute decryption times are slightly lower than those for

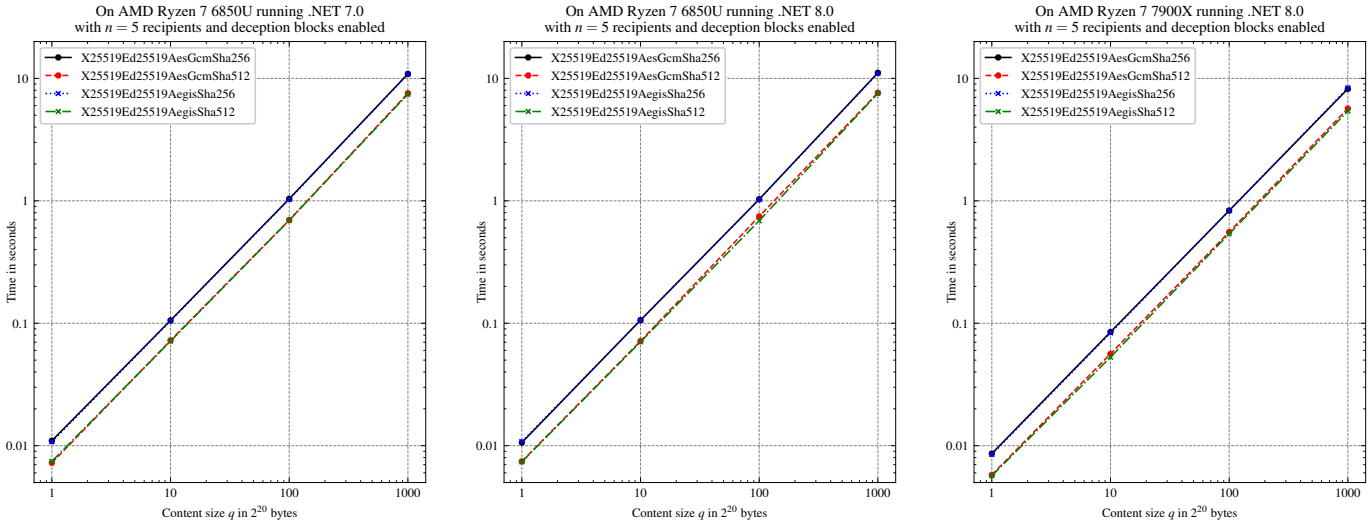


Figure 8. Average decryption time for different content sizes, cipher suites, and runs.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

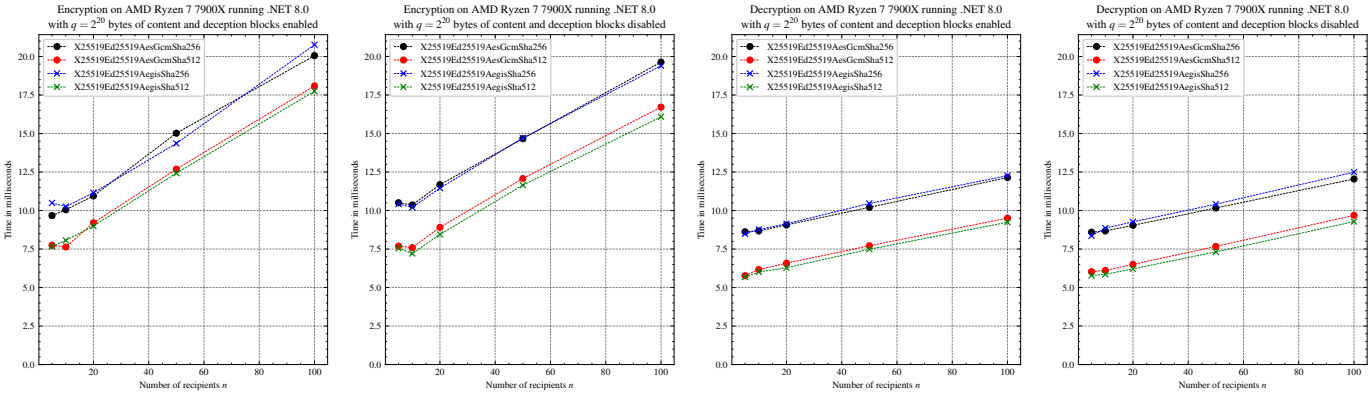


Figure 9. Average encryption and decryption time for different number of recipients, cipher suites, and whether deception blocks are enabled.

encryption, which must be due to the symmetric encryption and decryption algorithms. We can see the same speedup when using SHA-512 over SHA-256 as well as using a stronger processor and we observe little to no difference between the two symmetric encryption algorithms.

Our third experiment varies the number of recipients n from 5 to 1000 in order to make statements about the performance impact of generating the recipient-specific blocks in the public header. Furthermore, we can show that generating deception blocks increases the encryption time only slightly. We show the results in Figure 9, which are linear-linear plots and omit the samples for $n = 1000$ for clarity. The numerical values are listed in Table VI in Appendix C. Deception blocks have no impact on the decryption, which is why the two decryption charts differ only in run-to-run variance. All four charts show a linear relationship between the number of recipients and the average processing time.

In a fourth experiment, we focused on the performance impact of validating all recipient signatures during decryption. Similar to the previous experiment, the number of recipients was varied while the content size was fixed to 1 MiB. Activating the validation – essentially executing Step 12 in Subsection V-E – results in a linear performance hit as depicted in Figure 10. Deactivating the validation step does not remove the linear dependency completely since the recipients still have to be parsed as stated in Subsection IV-E. It is, however, drastically decreased, offering a performance benefit to certain applicable situations described earlier. In general, we recommend keeping the recipient signature validation enabled.

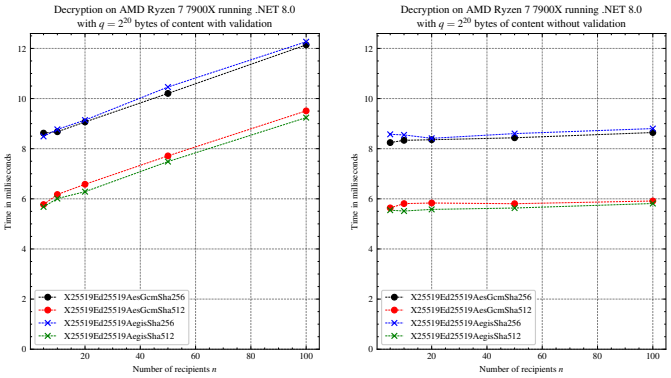


Figure 10. Average decryption time for different number of recipients, cipher suites, and whether recipient signature validation is enabled.

Accomplishing our goal of performing the encryption and decryption procedure in less than 100 ms (cf. Subsection IV-B) is therefore dependent on the amount of confidential data stored within an ECF as well as the number of recipients. Our experiments show that using a modern processor and choosing a cipher suite with SHA-512 as the hash function can yield to execution times below 100 ms for content sizes smaller than 10 MiB. As mentioned in Subsection III-D, an ECF is designed to store passwords, certificate keys, and other similar credentials for a small set of recipients. The size of these types of content is typically in the range of 2^5 to 2^{15} bytes. Therefore, even the smallest content size in our experiments, 1 MiB $= 2^{20}$ bytes, is more than sufficient for typical ECF usage. Additionally, even though the ECF format supports far more than 1000 recipients, typically there are only a few recipients per file. Our experiments show that both the encryption and decryption procedure take less than 15 ms each with 1 MiB-sized content and 50 recipients. Therefore, we can safely state that the PoC implementation fulfills the objective.

D. Private Key Management

A major change was made to the management and storing of the private keys. While our original solution featured basic password-based encryption using a Key Derivation Function (KDF), we improved the implementation to support user-chosen parameters. Furthermore, the former static and predefined private key storage format was replaced by a more flexible one. This enables our PoC implementation to be extended in the future with cipher suites using different Diffie-Hellman-like key exchange and matching signature algorithms. We discuss the option of choosing the key exchange algorithm independently from the signature algorithm in Appendix A.

In order for the ECF format to work, three components must be selected to match: the cipher suite, a private key, and the recipient information in the private body of an ECF. This is why our PoC implementation provides three base types `CipherSuite`, `ECFKey`, and `Recipient`. Then, the three subtypes `CSX25519Ed25519Base`, `EKX25519Ed25519`, and `RX25519Ed25519` are interlocked to form a group of matching components. Our implementation ensures this way that the correct type of key is used and the parsing of the recipient information blocks (cf. Figure 6) is done correctly.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

The newly added, flexible format for private key storage is depicted in Figure 11. It starts similar to a public header with a VERSION to ensure future extensibility. The next three fields, KEY TYPE, SYMMETRIC ENCRYPTION TYPE, and KDF TYPE define which type of private key is stored, which symmetric encryption scheme was used to encrypt it, and which KDF was used to derive the symmetric encryption key from the user-chosen password. Next, a SALT that is used for key derivation is stored, followed by a SYMMETRIC NONCE used for encrypting and decrypting the private key.

Depending on the KDF, its configuration is stored next in the field KDF CONFIG. For the default algorithm Argon2id [35], three values are stored: the number of iterations, the memory size, and the degree of parallelism. These parameters can be chosen freely by the user as long as they meet the specification. We provide sensible defaults in our PoC command line tool.

Finally, the encrypted private key is appended last. Its length is dependent on the cipher suite and the chosen symmetric encryption algorithm. All implemented cipher suites use an Ed25519 private key as their basis, since it can be converted into an Ed25519 public key and furthermore into both a X25519 private key and a X25519 public key. Therefore, it is only necessary to store the Ed25519 private key. However, when using different cipher suites (cf. Appendix A), it might be necessary to store multiple keys, which is why the storage format is designed as flexible as possible.

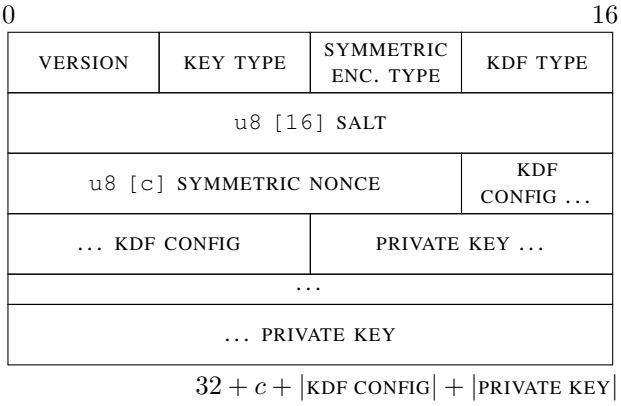


Figure 11. Private key storage format.

To keep the private key storage format simple, we opted to not include an encrypted hash value as a field. Instead, we utilize the "associated data" option of the AEAD-based symmetric encryption algorithm. All unencrypted fields are used as input to the "associated data" option, which means the PoC can detect modifications to any of the fields. This approach limits the choice of symmetric encryption schemes to AEAD-based ones. However, since both symmetric encryption schemes used within our PoC implementation are indeed AEAD-based, there is actually no restriction. The user can choose which symmetric encryption scheme and which KDF to use in order to protect the private key. Our PoC implementation uses AES-256-GCM [18][19] and Argon2id [35] by default.

VII. Conclusion and Future Work

In this paper, we extended our previous work of Encrypted Container File in [1] and described the ECF format in more detail. Additionally, we presented an example use case workflow in Section III, which motivated the requirements, the design goals, and ultimately the ECF format presented in Section IV. To demonstrate the ECF format flexibility we claimed in our previous work, we implemented two new cipher suites in our Proof of Concept implementation and focused on a cipher suite-independent presentation of the ECF format and operations in Section V. Although all discussed cipher suites use a key exchange and a signature algorithm based on Curve25519 [16] [17], we provide guidance for implementing ECF with different algorithms in Appendix A.

Section VI describes more changes and improvements made to our PoC implementation, which is now native to Linux and sports a new and improved private key management. We also took performance measurements in order to validate our claim of providing a fast-enough implementation to be used in production. We are confident that our work is ready to be used in production.

The full code of the updated and extended PoC implementation as well as unit tests, performance tests, and performance analysis data for that code are available at:

https://github.com/Hirnmoder/ECF

The modular structure of our PoC implementation allows for an easy extension and the implementation of more features and functionalities. We are looking forward to the feedback from the community to further improve the ECF format and its implementation.

Future work may focus on finding further use cases and applications of ECF. For example, the subject of applying ECF or ECF-based solutions in diverse contexts, such as within public authorities, raises interesting research questions and may lead to fruitful improvements of the ECF format as well as our PoC implementation.

Another area of work could be the implementation of a Public Key Infrastructure (PKI)-based trust model amongst the recipients of an ECF. Such extension could help establishing common trust anchors in multi-organization settings. In addition to that, changes to the ECF format could be made to prove content modification ownership by employing digital signatures.

Furthermore, it may be possible to design a two-staged access control framework which allows a set of recipients to read the confidential data, but only a subset of those recipients to alter it. Such amendment could help distinguishing between different roles of entities accessing an ECF.

Finally, the authors would like to thank the International Academy, Research, and Industry Association for the opportunity to present the idea of Encrypted Container Files in more detail, supplemented by current developments of the project.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

References

[1] Tobias J. Bauer and Andreas Aßmuth, "Encrypted Container File: Design and Implementation of a Hybrid-Encrypted Multi-Recipient File Structure," in *Proc of the 14th International Conference on Cloud Computing, GRIDs, and Virtualization (Cloud Computing 2023)*, Nice, France, Jun. 2023.

[2] M. Fowler, "Continuous Integration," Jan. 18, 2024, [Online]. Available: https://www.martinfowler.com/articles/continuousIntegration.html (visited on 2024.05.27).

[3] Intellectsoft, "Best Software Development Trends to Follow in 2024," Intellectsoft, Apr. 19, 2024, [Online]. Available: https://www.intellectsoft.net/blog/software-development-trends/ (visited on 2024.05.27).

[4] A. Lomas, "The Future of Software Development in 2024 and Beyond," Net Solutions, Mar. 10, 2023, [Online]. Available: https://www.netsolutions.com/insights/the-future-of-software-development/ (visited on 2024.05.27).

[5] P. Bajpai and A. Lewis, "Secure Development Workflows in CI/CD Pipelines," in *2022 IEEE Secure Development Conference (SecDev)*, 2022, pp. 65–66. DOI: 10.1109/SecDev53368.2022.00024.

[6] J. Walsh, "Secure CI/CD Pipelines: Practices for Managing CI/CD Secrets," CyberArk, Feb. 15, 2023, [Online]. Available: https://developer.cyberark.com/blog/secure-ci-cd-pipelines-best-practices-for-managing-ci-cd-secrets/ (visited on 2024.05.27).

[7] N. Sakovich, "Top 10 Software Development Trends of 2024," SaM Solutions, Apr. 8, 2024, [Online]. Available: https://www.sam-solutions.com/blog/software-development-trends/ (visited on 2024.05.27).

[8] P. Belagatti, "Managing Secrets in Your DevOps Pipeline," The New Stack, Aug. 4, 2022, [Online]. Available: https://thenewstack.io/managing-secrets-in-your-devops-pipeline/ (visited on 2024.05.27).

[9] The DevSecOps Guides Authors, "Secure Pipeline," DevSecOps Guides, Jun. 2, 2023, [Online]. Available: https://devsecopsguides.com/docs/code/secure-pipeline/ (visited on 2024.05.27).

[10] I. Koishybayev *et al.*, "Characterizing the Security of Github CI Workflows," in *31st USENIX Security Symposium (USENIX Security 22)*, Boston, MA: USENIX Association, 2022, pp. 2747–2763, ISBN: 978-1-939133-31-1.

[11] Git, "Git," 2024, [Online]. Available: https://git-scm.com/ (visited on 2024.05.27).

[12] Dispel LLC, "jak – simple git encryption," Dispel LLC, 2017, [Online]. Available: https://jak.readthedocs.io/en/latest/ (visited on 2024.05.27).

[13] A. Ayer, "git-crypt – transparent file encryption in git," 2023, [Online]. Available: https://www.agwa.name/projects/git-crypt/ (visited on 2024.05.27).

[14] E. Barker, *Recommendation for Key Management: Part 1 – General*. May 2020. DOI: 10.6028/nist.sp.800-57pt1r5.

[15] J. Saltzer and M. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE*, vol. 63, no. 9, pp. 1278–1308, 1975. DOI: 10.1109/PROC.1975.9939.

[16] D. J. Bernstein, "Curve25519: New diffie-hellman speed records," in *Public Key Cryptography - PKC 2006*, Springer Berlin Heidelberg, 2006, pp. 207–228.

[17] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of Cryptographic Engineering*, vol. 2, no. 2, pp. 77–89, Aug. 2012.

[18] D. A. McGrew and J. Viega, "The galois/counter mode of operation (GCM)," National Institute of Standards and Technology, May 31, 2005, [Online]. Available: https://csrc.nist.rip/groups/ST/toolkit/BCM/documents/proposedmodes/gcm/gcm-revised-spec.pdf (visited on 2024.05.27).

[19] M. J. Dworkin, "Recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac," Tech. Rep., Nov. 2007. DOI: 10.6028/nist.sp.800-38d.

[20] Q. H. Dang, "Secure Hash Standard," Tech. Rep., Aug. 2015. DOI: 10.6028/nist.fips.180-4.

[21] F. Denis and S. Lucas, "The AEGIS Family of Authenticated Encryption Algorithms," Internet Engineering Task Force, Internet-Draft draft-irtf-cfrg-aegis-aead-10, Jan. 20, 2024, Work in Progress.

[22] D. McGrew, *An Interface and Algorithms for Authenticated Encryption*, RFC 5116, Jan. 2008. DOI: 10.17487/RFC5116.

[23] "Squeamish Ossifrage", "Curve25519 over Ed25519 for key exchange? Why?" Crypto Stack Exchange, Mar. 19, 2019, [Online]. Available: https://crypto.stackexchange.com/a/68129 (visited on 2024.05.27).

[24] The Sodium Authors, "Ed25519 to curve25519 – libsodium," Oct. 27, 2023, [Online]. Available: https://doc.libsodium.org/advanced/ed25519-curve25519 (visited on 2024.05.27).

[25] The Sodium Authors, "Point*scalar multiplication – libsodium," Sep. 12, 2023, [Online]. Available: https://doc.libsodium.org/advanced/scalar_multiplication (visited on 2024.05.27).

[26] "fgrieu", "Using SHA2 as random number generator?" Crypto Stack Exchange, Jun. 20, 2020, [Online]. Available: https://crypto.stackexchange.com/a/81459 (visited on 2024.05.27).

[27] Microsoft Learn, "Single-file deployment," Microsoft, Mar. 11, 2023, [Online]. Available: https://learn.microsoft.com/en-us/dotnet/core/deploying/single-file/overview (visited on 2024.05.27).

[28] Microsoft, ".net – build. test. deploy.," Microsoft, 2024, [Online]. Available: https://dotnet.microsoft.com/en-us/ (visited on 2024.05.27).

[29] Docker Inc., "Docker: Accelerated Container Application Development," Docker Inc., 2024, [Online]. Available: https://www.docker.com/ (visited on 2024.05.27).

[30] The Sodium Authors, "Introduction – libsodium," May 25, 2024, [Online]. Available: https://doc.libsodium.org/ (visited on 2024.05.27).

[31] The Sodium Authors, "AEGIS-256 – libsodium," Nov. 1, 2023, [Online]. Available: https://doc.libsodium.org/secret-key_cryptography/aead/aegis-256 (visited on 2024.05.27).

[32] K. Hartke, "NSec – Modern cryptography for .NET Core," 2024, [Online]. Available: https://nsec.rocks/ (visited on 2024.05.27).

[33] The Sodium Authors, "AES256-GCM – libsodium," Sep. 12, 2023, [Online]. Available: https://doc.libsodium.org/secret-key_cryptography/aead/aes-256-gcm (visited on 2024.05.27).

[34] R. Frame, "SHA-512 faster than SHA-256?" Crypto Stack Exchange, Jun. 17, 2015, [Online]. Available: https://crypto.stackexchange.com/a/26351 (visited on 2024.05.27).

[35] A. Biryukov, D. Dinu, and D. Khovratovich, "Argon2: The memory-hard function for password hashing and other applications," version 1.3, Mar. 24, 2017, [Online]. Available: https://raw.githubusercontent.com/P-H-C/phc-winner-argon2/master/argon2-specs.pdf (visited on 2024.05.27).

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

APPENDIX

A. Choosing the Key Exchange Algorithm Independently from the Signature Algorithm

The ECF format as well as our PoC implementation use a well-defined conversion method to convert an Ed25519 private or public key to its corresponding X25519 private or public key [23][24], respectively. This implies that storing, using, and having access to the Ed25519-version of a private or public key is enough to carry out all signing and key agreement operations described in this paper. In order to save memory and remove unnecessary redundancy, we opted to employ this conversion method throughout the ECF format and operations.

However, different key exchange and signature algorithms may not support such key conversion. In this case, the recipient information block stored inside the private body of an ECF (cf. Figure 6) must include not one but two public keys: one for the signature algorithm and one for the key exchange algorithm. We depict the altered structure in Figure 12 and assume for simplicity reasons that both public keys have the same length of u bytes. This is, however, not a requirement.

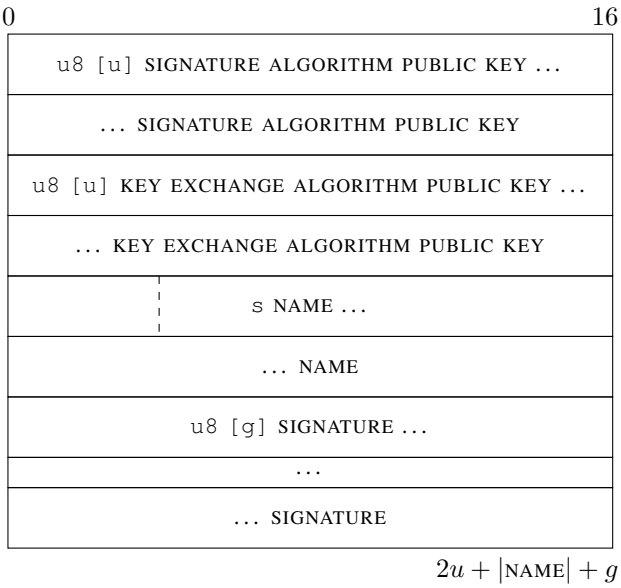


Figure 12. Recipient information block structure within the private body of an ECF when no key conversion method is available to convert between the keys of the key exchange and the signature algorithm.

The field SIGNATURE now contains a signature not only over the NAME, but over the combination of KEY EXCHANGE ALGORITHM PUBLIC KEY and NAME. This is compulsory in order to create a strong connection between the public key for key exchange and the human-readable name of a recipient.

The presented changes in data structure also need to be reflected in the ECF operations. The recipient entry is now constructed as follows:

$$\psi = \text{pk}^{\text{S}}_{\psi} \| \text{pk}^{\text{X}}_{\psi} \| \text{NAME} \| t$$
$$t = \text{Sign}\left(\text{sk}^{\text{S}}_{\psi}, \text{pk}^{\text{X}}_{\psi} \| \,\text{UTF8}\left(\text{NAME}\right)\right)$$

During encryption, $\text{pk}^{\text{X}}_{\psi_i}$ is already present and, therefore, Step 3c in Subsection V-A is omitted. In Step 6, the extended recipient information block structure is used.

The decryption procedure (cf. Subsection V-B) is extended and now needs three input parameters:

$$(\Psi, \Omega) = \text{Dec}^{\text{ECF}}\left(\text{sk}^{\text{S}}, \text{sk}^{\text{X}}, \mathcal{E}\right)$$

Similarly to the encryption procedure, the first conversion in Step 3 in Subsection V-B is omitted. Next, the deconstruction in Step 12a produces four values $\text{pk}^{\text{S}}_{\psi_i}$, $\text{pk}^{\text{X}}_{\psi_i}$, NAME_i, and t_i and the verification in Step 12b needs to be adjusted, too:

$$\text{If } \text{Vrfy}^{\text{S}}\left(\text{pk}^{\text{S}}_{\psi_i}, \text{pk}^{\text{X}}_{\psi_i} \| \,\text{UTF8}\left(\text{NAME}_i\right), t_i\right) \stackrel{?}{=} 0, \textit{ Exit.}$$

Adding new recipients to an ECF requires them to generate their recipient entries as shown in Subsection V-C. The computations for t_b and ψ_b need to be adjusted accordingly and include the public key for key exchange as described above. This is also true for the verification in Step 1 in Subsection V-C. The mandatory check in Step 3 now needs to check for both $\text{pk}^{\text{S}}_{\text{B}}$ and $\text{pk}^{\text{X}}_{\text{B}}$.

The optimized procedure for generating deception blocks, which is described in Subsection V-E, does not require any modifications. This is because the IDENTIFICATION TAG is generated randomly instead of being derived from a key pair and, thus, no key conversion is necessary.

B. Variables and Functions

Table I contains all simple variables used primarily to describe the ECF format. The encryption and decryption procedures make use of various cryptographic algorithms and their respective input and output parameters. These are defined and described in Table II. All used functions are listed in Table III. A randomized or probabilistic algorithm is indicated by the left arrow "$\leftarrow$", whereas an equal sign "$=$" indicates deterministic execution.

TABLE I. Variables.

Variable	Description
a	length of KEY AGREEMENT INFORMATION in bytes
b	length of the encrypted private body in bytes
b'	length of the unencrypted private body in bytes
c	length of SYMMETRIC NONCE in bytes
d	length of the output of the hash function in bytes
g	length of a signature in bytes
h	length of the public header in bytes
m	number of recipients in the public header
n	number of true recipients in the private body
o_i	offset of recipient-specific block i within the file in bytes
q	length of CONTENT in bytes
$\bar{q}$	length of all private body fields except CONTENT in bytes
u	length of a public key in bytes
v	length of a private key in bytes
y	length of the symmetric key in bytes

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II. Component Variables.

Variable	Description
$\mathrm{pk}_{\mathrm{A}}^{\mathrm{X}} \in \{0,1\}^{8u}$	public key of Alice for key exchange
$\mathrm{sk}_{\mathrm{B}}^{\mathrm{X}} \in \{0,1\}^{8v}$	private/secret key of Bob for key exchange
$\mathrm{k}_{\mathrm{final}} \in \{0,1\}^{8y}$	final symmetric encryption key
$\alpha \in \{0,1\}^{8c}$	symmetric nonce
$\theta \in \{0,1\}^{8 \cdot 16}$	salt
$t \in \{0,1\}^{8g}$	a signature t is g bytes long
$\beta \in \{0,1\}^{8d}$	hash function's output is always d bytes
$r = \mathrm{id_tag} \Vert \mathrm{pk}_{\mathrm{e}}^{\mathrm{X}} \Vert \mathrm{k}_{\mathrm{pre1}}$	recipient in public header
$R = \{r_0, \ldots, r_{m-1}\}$	recipient set in public header
$\psi = \mathrm{pk}_{\psi}^{\mathrm{S}} \Vert \mathrm{NAME} \Vert t$	recipient in private body
$\Psi = \{\psi_0, \ldots, \psi_{n-1}\}$	recipient set in private body
$\Omega \in \{0,1\}^{8q}$	confidential data stored in an ECF, q bytes long
$\mathcal{E} = \mathcal{H} \Vert \mathcal{B} \Vert \beta_{\mathrm{all}}$	an ECF consists of a public header $\mathcal{H}$, a private body $\mathcal{B}$, and a public footer β_{all}

TABLE III. Functions.

Function Definition	Description
$x = x_1 \Vert x_2$	concatenation of two bit strings
$x = x_1 \oplus x_2$	bitwise exclusive OR (XOR) operation on two same-length bit strings
$x' = x\,[0, \ldots, j-1]$	truncation of the bit string $x \in \{0,1\}^{l \geq 8j}$ to the first j bytes
$\left(\mathrm{sk}^{\mathrm{S}}, \mathrm{pk}^{\mathrm{S}}\right) \leftarrow \mathrm{Gen}^{\mathrm{S}}$	generates a key pair for signing
$\left(\mathrm{sk}^{\mathrm{X}}, \mathrm{pk}^{\mathrm{X}}\right) \leftarrow \mathrm{Gen}^{\mathrm{X}}$	generates a key pair for key exchange
$\mathrm{sk}^{\mathrm{X}} = \mathrm{Convert}^{\mathrm{S} \to \mathrm{X}}\left(\mathrm{sk}^{\mathrm{S}}\right)$ $\mathrm{pk}^{\mathrm{X}} = \mathrm{Convert}^{\mathrm{S} \to \mathrm{X}}\left(\mathrm{pk}^{\mathrm{S}}\right)$	converts a private/public key for signing to a private/public key for key exchange (*)
$\mathrm{pk}^{\mathrm{S}} = \mathrm{Convert}^{\mathrm{sk} \to \mathrm{pk}}\left(\mathrm{sk}^{\mathrm{S}}\right)$ $\mathrm{pk}^{\mathrm{X}} = \mathrm{Convert}^{\mathrm{sk} \to \mathrm{pk}}\left(\mathrm{sk}^{\mathrm{X}}\right)$	converts a private key to a public key
$\mathrm{k} \leftarrow \mathrm{Gen}^{\mathrm{SYM}}$	generates a symmetric encryption key
$x = \mathrm{UTF8}\,(s)$	converts the character string s without BOM into the UTF-8 bit string x
$x = \mathrm{Random}\,(j)$	generates a random j-byte long bit string x
$\mathrm{ss} = \mathrm{Kex}\left(\mathrm{sk}_{\mathrm{A}}^{\mathrm{X}}, \mathrm{pk}_{\mathrm{B}}^{\mathrm{X}}\right)$ $= \mathrm{Kex}\left(\mathrm{sk}_{\mathrm{B}}^{\mathrm{X}}, \mathrm{pk}_{\mathrm{A}}^{\mathrm{X}}\right)$	obtain a shared secret by performing the key exchange with a secret and a public key
$t \leftarrow \mathrm{Sign}\left(\mathrm{sk}_{\mathrm{A}}^{\mathrm{S}}, x\right)$	signing the bit string x with Alice's private key for signing yields a signature t
$\mathrm{Vrfy}^{\mathrm{S}}\left(\mathrm{pk}_{\mathrm{A}}^{\mathrm{S}}, x, t\right) \in \{0,1\}$	returns 1, if and only if the signature t is valid for the given public key and the bit string x, otherwise 0
$\gamma \leftarrow \mathrm{Enc}^{\mathrm{SYM}}\,(\mathrm{k}, \alpha, x)$	encrypting the bit string x with a symmetric encryption key k and a nonce α produces a ciphertext bit string γ
$x' = \mathrm{Dec}^{\mathrm{SYM}}\,(\mathrm{k}, \alpha, \gamma)$	decrypting a ciphertext bit string γ with the correct symmetric encryption key k and the correct nonce α produces the bit string x'
$\beta = \mathrm{H}\,(x)$	hash functions take any-length bit strings x and produce a fixed-sized output β
$\mathrm{Vrfy}^{\mathrm{H}}\,(x, \beta) \in \{0,1\}$	returns 1, if and only if the hash value β matches the hash value of the bit string x, otherwise 0

(*) This behavior is specific for Ed25519 [17] and X25519 [16] using the conversion method described in [23][24]. However, if a cipher suite does not allow such conversion, cf. Appendix A.

C. Performance Analysis Value Tables

This subsection provides the numerical values of the performance charts in Subsection VI-C. All tables contain the aggregated data used to draw the performance charts. However, Tables VI and VII contain additional data points for $n = 1000$ recipients, showing that our PoC implementation is capable of handling even such unrealistic parameters efficiently. The raw data and the code to aggregate and plot the performance measurements is published on GitHub.

TABLE IV. Average encryption time (in seconds) for different content sizes and cipher suites. Visualized in Figure 7.

Content Size	Run	Cipher Suite I	II	III	IV
1 MiB	R7 6850U (.NET 7.0)	0.012	0.010	0.014	0.010
	R7 6850U (.NET 8.0)	0.012	0.010	0.014	0.010
	R7 7900X (.NET 8.0)	0.010	0.008	0.010	0.008
10 MiB	R7 6850U (.NET 7.0)	0.136	0.100	0.144	0.103
	R7 6850U (.NET 8.0)	0.140	0.105	0.141	0.106
	R7 7900X (.NET 8.0)	0.103	0.075	0.104	0.074
100 MiB	R7 6850U (.NET 7.0)	1.210	0.883	1.295	0.983
	R7 6850U (.NET 8.0)	1.265	0.928	1.250	0.913
	R7 7900X (.NET 8.0)	0.934	0.683	0.950	0.680
1000 MiB	R7 6850U (.NET 7.0)	12.822	9.490	12.740	9.508
	R7 6850U (.NET 8.0)	12.711	9.665	12.726	9.650
	R7 7900X (.NET 8.0)	9.449	7.153	9.782	6.948

TABLE V. Average decryption time (in seconds) for different content sizes and cipher suites. Visualized in Figure 8.

Content Size q	Run	Cipher Suite I	II	III	IV
1 MiB	R7 6850U (.NET 7.0)	0.011	0.007	0.011	0.007
	R7 6850U (.NET 8.0)	0.011	0.007	0.011	0.007
	R7 7900X (.NET 8.0)	0.009	0.006	0.008	0.006
10 MiB	R7 6850U (.NET 7.0)	0.106	0.072	0.105	0.071
	R7 6850U (.NET 8.0)	0.106	0.071	0.106	0.070
	R7 7900X (.NET 8.0)	0.085	0.056	0.083	0.053
100 MiB	R7 6850U (.NET 7.0)	1.038	0.696	1.028	0.693
	R7 6850U (.NET 8.0)	1.030	0.746	1.028	0.687
	R7 7900X (.NET 8.0)	0.832	0.555	0.831	0.538
1000 MiB	R7 6850U (.NET 7.0)	10.882	7.557	10.836	7.436
	R7 6850U (.NET 8.0)	11.113	7.626	10.992	7.559
	R7 7900X (.NET 8.0)	8.212	5.677	8.366	5.415

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE VI. Average encryption and decryption times (in milliseconds) for different number of recipients and cipher suites. Visualized in Figure 9.

n	Operation	Deception blocks	Cipher Suite I	II	III	IV
5	Enc	✓	9.671	7.745	10.493	7.659
		✗	10.506	7.688	10.424	7.552
	Dec	✓	8.622	5.771	8.490	5.679
		✗	8.594	6.032	8.364	5.784
10	Enc	✓	10.050	7.630	10.253	8.061
		✗	10.363	7.593	10.205	7.211
	Dec	✓	8.676	6.172	8.771	6.011
		✗	8.675	6.111	8.859	5.858
20	Enc	✓	10.948	9.204	11.154	8.992
		✗	11.680	8.909	11.457	8.451
	Dec	✓	9.065	6.580	9.141	6.285
		✗	9.038	6.507	9.264	6.216
50	Enc	✓	15.023	12.688	14.358	12.424
		✗	14.664	12.076	14.706	11.646
	Dec	✓	10.204	7.713	10.459	7.485
		✗	10.164	7.670	10.414	7.315
100	Enc	✓	20.055	18.090	20.763	17.747
		✗	19.636	16.715	19.402	16.085
	Dec	✓	12.141	9.509	12.266	9.240
		✗	12.049	9.689	12.491	9.284
1000	Enc	✓	119.864	115.551	119.596	115.086
		✗	104.953	101.514	106.809	99.056
	Dec	✓	46.000	43.972	47.385	42.321
		✗	46.109	43.981	47.173	42.594

TABLE VII. Average decryption time (in milliseconds) for different number of recipients and whether recipient validation is enabled. Visualized in Figure 10.

n	Validation	Cipher Suite I	II	III	IV
5	✓	8.622	5.771	8.490	5.679
	✗	8.244	5.639	8.573	5.547
10	✓	8.676	6.172	8.771	6.011
	✗	8.334	5.809	8.549	5.513
20	✓	9.065	6.580	9.141	6.285
	✗	8.357	5.833	8.414	5.578
50	✓	10.204	7.713	10.459	7.485
	✗	8.435	5.805	8.602	5.633
100	✓	12.141	9.509	12.266	9.240
	✗	8.644	5.914	8.802	5.813
1000	✓	46.000	43.972	47.385	42.321
	✗	11.239	8.300	11.366	7.840

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Joining of Data-driven Forensics and Multimedia Forensics - Practical Application on DeepFake Image and Video Data

Dennis Siegel
Dept. of Computer Science
Otto-von-Guericke-University
Magdeburg, Germany
dennis.siegel@ovgu.de

Christian Kraetzer
Dept. of Computer Science
Otto-von-Guericke-University
Magdeburg, Germany
christian.kraetzer@ovgu.de

Stefan Seidlitz
Dept. of Computer Science
Otto-von-Guericke-University
Magdeburg, Germany
stefan.seidlitz@ovgu.de

Jana Dittmann
Dept. of Computer Science
Otto-von-Guericke-University
Magdeburg, Germany
jana.dittmann@ovgu.de

***Abstract*—DeepFake technology poses a new challenge to the validation of digital media integrity and authenticity. In contrast to 'traditional' forensic sub-disciplines (for example dactyloscopy), there are no standardized process models for DeepFake detection yet that would enable its usage in court in most countries. In this work, two existing best-practice methodologies (a data-centric model and a set of image authentication procedures) are combined and extended for the application of DeepFake detection. The extension includes aspects required to expand the focus from digital images to videos and enhancements in the quality assurance for methods (here focusing on the peer review aspect). Particular emphasis is put on the different actors involved in the forensic examination process. The new methodology is applied to the example of DeepFake detection in two application scenarios, based on image and video respectively. The process itself is further separated in the initial assessment of the media followed by DeepFake detection. In total 36 features from nine existing and implemented tools are used as methods. In addition, the value types, ranges and their tendency for a DeepFake are determined for each feature. To further diversify the application field, the DeepFake detectors represent both handcrafted and deep learning based feature spaces for Media content analysis. The whole process is then manually evaluated, highlighting potential loss, error and uncertainties within the process and individual tools. With the discussed potential extensions towards video evidence and machine learning involved, we identified additional requirements. These requirements are addressed in this paper as a proposal for an extended methodology to serve as starting point for future research and discussion in this domain.**

Keywords-forensics; media forensics; DeepFake detection; machine learning.

I. Introduction and Motivation

Recent advances in computer vision and deep learning enabled a new digital media manipulation technology called DeepFakes, replacing identities in digital images, videos and audio material. They pose a challenge to the integrity and authenticity of digital media and the trust placed in media objects for forensic science. With the advances in technology and also DeepFake quality, they are no longer easily recognizable as such to the bare eye. For this reason, most existing protection approaches use machine learning algorithms for DeepFake detection. The use of machine learning makes it necessary to fulfil additional requirements for artificial intelligence (AI) systems (i.e., legal regulations). In consequence, DeepFake detectors are still not suitable for court room usage. This is due to aspects such as lack of maturity, including (besides precisely validated error rates) modeling and standardization efforts so that they can be integrated into established forensic procedures.

In this paper, as an extended work of [1], this gap (i.e., the lack of process modeling and investigation steps) is partially addressed by the following contributions:

- conceptional joining of IT and media forensic methodologies on the selected example of the existing *Data-Centric Examination Approach (DCEA)* [2], [3] and the *Best Practice Manual for Digital Image Authentication (BPM-DI)* from the *European Network of Forensic Science Institute (ENFSI)* [4].
- strengthening of the Human-in-the-Loop aspect in the forensic examination by highlighting the human operators involved and usage of algorithms in a decision support system.
- application of our concept to both an image and video DeepFake detection scenario, by utilizing a total of nine tools for general purpose media analysis, image processing and DeepFake detection.

With the focus on process modelling in the context of individual investigations, the prerequisites for the use of the individual tools are not considered in this paper in detail. This includes essential aspects such as initial model training, appropriate benchmarking and certification of the proposed tools. For these aspects the reader is referred to [5].

The paper is structured as follows. First, an overview of the state of the art on digital forensics, standards and regulations as well as the topic of DeepFake and its different types is presented in Section II. Following that, our concept of combining data-driven and media forensics can be found in Section III based on the DCEA [2] and BPM-DI [4]. Additional details on different human operators involved in the process are provided. In Section IV the proposed concept is being used in two application scenarios, both for image and video. This application is divided into two parts: first, an initial assessment is carried out, to validate the suitability of the material and then the DeepFake detection is performed. Finally, conclusions are drawn from the evaluation results presented and future directions are outlined in Section V.

It has to be noted, that this paper is an extended version of our work, presented at the SECURWARE 2023 conference [1]. This paper significantly expands on the aspect of human

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

operators involved in the forensic process as well as validating the practical applicability of the proposed methodology. To be more precise, it presents an expanded view of the fundamentals in the context of forensics by integrating additional Best Practice Manuals (BPM) of ENFSI (especially [6], presented in Section II-A) as well as DeepFake creation and detection (found in Section II-C). These fundamentals are used to conceptualize human operators involved in the forensic investigation in Section III-A. Furthermore, the practical application of the proposed methodology is expanded by testing on both an image and video DeepFake detection scenario. The Methods used in these scenarios are separated in Initial Assessment (Section IV-A), to validate the suitability of DeepFake detection Methods and the DeepFake detection itself (Section IV-B). To further support the separation, six additional tools are introduced.

II. Forensic Investigations in the Context of DeepFake Detection

With the potential of DeepFake manipulations in digital media it is even more important to validate integrity and authenticity of digital media especially for intended court room usage. The following sections address the current state and challenges in digital forensics, existing and upcoming regulations and the topic of DeepFake. These three aspects state fundamentals for the intended court room usage and while they are established in themselves, they are mostly considered in isolation.

A. Digital Forensics

Digital forensics is a subdomain of forensics, which is defined as "*the use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation, and presentation of digital evidence derived from digital sources [...]*" [7]. In [8] the domain of digital forensics is further divided into computer and multimedia forensics based on their link to the outside world. Computer forensics operates exclusively in the digital domain, whereas multimedia forensics uses sensors to capture and connect with the real world.

In general, the application of media forensics is governed by national legislation. For this reason, our focus will be on European documents and views on media forensics. Here, the European Network of Forensic Science Institutes (ENFSI) provides a broad list of BPM and guidelines in forensics.

All recent BPM share a common structure, governed by a common template: the scope of the BPM, definitions and terms, resources (including personnel), methods, validation and estimation of uncertainty of measurement, quality assurance, handling of items, initial assessment, prioritization and sequence of examinations, evaluation and interpretation as well as presentation of results.

The discussions on the personnel usually include discussions on the separation of duties between different roles as well as aspects of training and proficiency testing. Another item of relevance here is the validation and estimation of uncertainty of measurement. One main goal of the validation considerations is defined in [9] as precisely described, tool driven and repeatable processes: "*For software tools that can be configured in a variety of ways and/or uses a number of different parameters, it is particularly important to document the set-up and individual parameter values in order to produce a process that can be repeated*". These reproducibility requirements are the same for 'manual analysis software' as well as 'automated' (i.e., pattern recognition driven) software solutions.

In [6] extensive considerations are put into the validation and estimation of uncertainty. An important aspect of these discussions lies in the distinguishing between verified or non-verified functions and tools, 'validated processes' and 'trustworthy processes'.

Not only automated processes are within the scope of the verification and re-verification work to be performed. In [6] specifically the human-based methods are also included: "*Human-based functions are the pivotal elements within technical forensic processes, all forensic processes are likely to require user interaction, therefore an evaluation of user capability must be made as part of validated process within the laboratory. Even if an instrument-based function returns a valid result, it may still be reliant on the correct interpretation by the user associating the result.* [...] *Verification of human-based (user) functions are covered within proficiency testing* [...]."

The availability of the required forensic practitioners with sufficient training and currently valid certification (if required) is an important factor in every forensic investigation. In this paper, this is accompanied by the need to ascertain that other relevant types of personnel (e.g., data scientists) are also available to perform tasks that need forms or specific know-how (e.g., the creation/curation/update of trained models for AI-based investigation methods).

The whole issue on the validation of tools and processes is a necessity in the risk assessment required for case handling. In [6] it is stated on that issue: "*For the interpretation of evidential significance in the context of the case, a laboratory should always consider the use of techniques and equipment whose risks have been formally assessed; as part of the required functional verification, in preference to those, which have not. This does not mean that a method or process that has not been formally evaluated cannot be used to aid the analysis; rather it means that if there is a wish to use such a solution, a formal justification as to why it has been chosen in preference to one that is part of a validated process must be made. When designing a validation process, five key elements of a successful validation policy are:*

1) An understanding of known errors and uncertainty
2) The Statement of Requirements;
3) Risk Analysis and Assessments;
4) Effective validation test sets; and
5) Routine verification.*"*

According to this list, the fourth and fifth elements are more or less self-explanatory. Elements 1, 2 and 3 needs additional

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

explanations, which are given directly in the following for the 'An understanding of known errors and uncertainty', 'Statement of Requirements' as well as the 'Risk Analysis and Assessments': The 'understanding of known errors and uncertainty' needs a closer specification of the term 'uncertainty', which in [6] is specified as: "*is the unknown (random) difference (delta) between the measurement taken and its true value. It can never be completely defined, or eliminated, and is represented as a bounded region, in which the true value exists within its given confidence level.*" In complex systems, uncertainty is aggregating: "*Uncertainty within a system is additive in nature, and generally increases with the number of functions deployed within a process. The decision as to whether the uncertainty should be calculated at the function level or abstracted to the process level is at the discretion of each laboratory.* [...] *Software solutions will also contain additional uncertainty on top of the uncertainty associated with the physical systems, including the operating system, they are running on. This is especially true for software, which relies on functions with no formal specification and/or calibrated standard. As a result, software uncertainty properties will also need to be acknowledged and accounted for.*"

Regarding the considerations on "*uncertainty within image authentication*", BPM-DI [4] identifies three domain specific potential factors as: "*tool inaccuracies*", "*operator inaccuracies*" and "*data inconsistencies*". Also acknowledging that those factors are interlinked, the BPM-DI elaborates: "*Given the intricate dependencies which could exist between uncertainties that arise at various points during the image authentication analysis procedures, the uncertainty attached to a specific measurement cannot always be quantified.*"

The 'Statement of Requirements' is defined in [6] as: "*The statement of requirements defines the problem to be solved by a technical process. It should provide explanatory text to set the scene for a lay reader, summarising the problem, noting the scope and acceptable risks or limits of any solution and acknowledging the relevant stakeholders. It should be created independently of and without regard to any particular implementation or solution.*" Furthermore, the statement of requirements "*provides the interface (or formal bridge) between what the customer believes is achievable (customer requirements), and so desires, and what the laboratory can realistically achieve (laboratory capability) with the available staff, tools and the incurred time costs.*"

Optimally, this statement of requirements is not only a list, which expresses a set of needs and corresponding associated constraints and conditions but also includes a "*list of well-formed, testable requirements*." [6] In the ENFSI BPM FIT, an exemplary list of types for such requirements are presented, including functional and performance requirements as well as requirements focusing on the interfaces for the solution, its compliance with local laws and processes, etc. In addition it is stated that "*If the risks are considered too great then either the statement of requirements will need to be amended, or alternate solutions sought, to reduce the risks to acceptable levels.*". It basically determines, which methods are to be used within a forensic examination to be conducted, based on customer requirements. For the 'Risk Analysis and Assessments' [6] states: "*risk analysis and verification stages are paramount in creating a reliable validation method*", with the BPM providing a very general description how to perform such an risk analysis and how to record/document the risk in a formal assessment process. Different examples for corresponding evaluation questions to be used within such an assessment process are provided, including method-specific questions, implementation specific question as well as questions regarding the labs organizational procedures regarding the usage within a process. Summarising the discussion on risk analysis, [6] states: "*Risk analysis can not only be used to explain why a verified function has been used within a validated process, but also why in certain circumstances a formally unverified function has been chosen in preference.*"

The ENFSI BPM FIT [6] explicitly integrates the competence of the forensic practitioner(s) available to handle a case into the risk analysis: "*The lower the level of knowledge [of the analyst], the greater will be the potential errors and risks.*" But also experienced analysts might encounter challenges when interpreting the output of verified functions. In this case, the escalation procedure recommended is: "*If a new, unknown, discrepancy is detected then the evaluation will need to be highlighted for the peer review, and one or more of the verified tools may need to be reassessed, along with the existing validated process.*"

With regard to the usage of non-verified functions, which is a very likely scenario for certain media forensics investigation that still lack maturity and for which only lower technology readiness level solutions exist so far, the recommendation of [6] with regard to the corresponding risk assessment would be: "*When using a non-verified function during analysis it is important that the analyst is competent enough to research the characteristics of the returned results, and can qualify them against standard validation methods employed within the laboratory* [...]."

In the field of digital imaging, there are currently three Best Practice Manuals existing. The first document addresses the aspect of forensic facial image comparison [10] and formulates the respective investigation steps. At the beginning, competing hypotheses are made, which need to be examined. In the context of comparing facial images, these hypotheses could be whether a subject in an image is a specific person or some other person. The comparison is performed based on the ACE-V methodology, which stands for *Analysis*, *Comparison*, *Evaluation* and *Verification*. ACE-V is a common practice in forensic comparison tasks, such as fingerprint [11] and facial image comparison [10].

In [11] the *Analysis* is described as: "*The examiner makes a determination, based upon previous training, experience, understanding, and judgments, whether the print is sufficient for comparison with another print. If one of the prints is determined to be insufficient, the examination is concluded with a determination that the print is insufficient for comparison purposes*." This highlights the importance of validating

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the suitability of material for a forensic investigation. In the domain of facial images, ENFSI provides a list with potential factors influencing the facial appearance [10]. This potentially non-exhaustive list contains the aspects of "*image resolution/distance from camera*", "*image compression*", "*aspect ratio*", "*lighting*", "*occlusion*", "*camera angle*", "*image/lens distortion*", "*number of available images*" and "*date an image was captured*" [10]. In addition, the National Institute of Standards and Technology (NIST) and German Federal Office for Information Security (BSI; the German national cyber security authority) are collaborating to create the so called Open Source Face Image Quality (OFIQ) metric [12], to estimate the facial image quality. This metric is intended to be derived from the Face Image Quality Assessment (FIQA) discussed by Schlett et al. [13]. If the quality of the media file is not sufficient, it has to be either discarded or enhanced. Procedures on image and video enhancement can be found in [14]. However, it has to be noted that "*Image enhancement processes alter the appearance and content of an image and may distort facial features or introduce artefacts that mislead the comparison examination*." [14]. The *Comparison* component of ACE-V for the domain of facial images is discussed in detail in [10]. This comparison is performed by an examiner on the basis of a so-called facial feature list, including a total of 19 facial components, such as eyes, nose and mouth. In *Evaluation*, the results of the comparison are used to confirm or refute the competing hypotheses. In the end, the examination process has to be repeated independently by another examiner for *Verification* purposes.

The most recent document on image forensics and also the closest to the topic of DeepFake detection, is the *Best Practice Manual for Digital Image Authentication (BPM-DI)* [4]. In its own words it "*aims to provide a framework for procedures, quality principles, training processes and approaches to the forensic examination*" in the context of image authentication. For this purpose it describes a total of four aspects to categorize and structure investigation steps. These aspects consist of two different analysis methods, namely **Auxiliary data analysis** and **Image content analysis**, which are used based on different **Strategies** fulfilling different purposes. The last method class is **Peer review**, enabling the validation, interpretation and evaluation of the individual methods and their outcomes by a forensic human examiners.

At the national level, the German situation is relevant for the authors. Here, the guidelines for IT forensic by BSI [55] are currently relevant. The DCEA is an extension of these guidelines, which has three main components: a model of the *phases* of a forensic process, a classification scheme for *forensic method classes* and *forensically relevant data types*.

The six DCEA *phases* are briefly summarized as: *Strategic preparation (SP), Operational preparation (OP), Data gathering (DG), Data investigation (DI), Data analysis (DA) and Documentation (DO)*. While the first two (*SP and OP*) contain generic (*SP*) and case-specific (*OP*) preparation steps, the three phases *DG*, *DI* and *DA* represent the core of any forensic investigation. At this point it is necessary to emphasize the importance of the *SP*, because it is the phase that also includes all standardization, benchmarking, certification and training activities considered. For details on the phase model the reader is referred, e.g., to [2] or [15].

In terms of data types, the DCEA proposes a total of six for digital forensics and ten for digitized forensics. In [3], the data types are specified in the context of media forensics and are referred to as *media forensic data types (MFDT)*. The resulting eight can be summarized as: digital input data *MFDT1* (the initial media data considered for the investigation), processed media data *MFDT2* (results of transformations to media data), contextual data *MFDT3* (case specific information, e.g., for fairness evaluation), parameter data *MFDT4* (contain settings and other parameter used for acquisition, investigation and analysis), examination data *MFDT5* (including the traces, patterns, anomalies, etc that lead to an examination result), model data *MFDT6* (describe trained model data, e.g., face detection and model classification data), log data *MFDT7* (data, which is relevant for the administration of the system, e.g., system logs), and chain of custody & report data *MFDT8* (describe data used to ensure integrity and authenticity, e.g., hashes and time stamps as well as the accompanying documentation for the final report).

An additional extension is made in the process modeling, in which individual processing steps are represented as atomic black box components. These components are accompanied by a description of the process performed. The individual components have four connectors input, output, parameters and log data. In addition, with the increasing use of machine learning, a fifth connection required for knowledge representation is defined. The labeled model can be found in [3].

B. Standards and Regulations in the Context of Media Forensics

With the intended court room usage of forensic methods, standardization is required in investigation and analysis procedures. One of the more established standards is the United States Federal Rules of Evidence (FRE; especially FRE 702, see [16]) and the Daubert standard in the US. Although these standards only apply in the US, its usage, e.g., in Europe has been discussed in [17]. In this work, the focus is on modelling media forensic methods within an investigation, whereby the following two (of five) Daubert criteria are particularly relevant [17]:

- "*whether the technique or theory has been subject to peer review and publication*";
- "*the existence and maintenance of standards and controls*".

In the context of standards and controls, the European Commission proposed the Artificial Intelligence Act (AIA), addressing the usage of Artificial Intelligence (AI) systems [18]. At the current time, the proposal has been adjusted and approved by the European Parliament [19]. This upcoming regulation places particular emphasis on the human in control aspects (Art. 14). The decisive factor is therefore not only

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the decision of the AI system, but the process of decision-making, which must be comprehensible for the human operator and thus enable the decision to be questioned and challenged. In addition, the International Criminal Police Organization (INTERPOL) recently published a document, addressing the usage of AI systems for law enforcement purposes [20]. Furthermore, the National Institute of Standards and Technology (NIST) currently develops a dataset for DeepFake detection for validation of methods [21]. All documents have in common that a human operator should comprehend and oversee the processing and decision-making of the AI system.

C. DeepFakes

With the advances in machine learning and computer vision DeepFake are a recent form of digital media manipulation and generation. In contrast to previous manipulation techniques, DeepFake utilizes deep learning to artificially generate or manipulate existing digital media, such as image, video and audio data. The application of DeepFakes is very versatile and can also be used for positive aspects, as described in [22]. Independently of their intended purpose, DeepFakes have to be identifiable both for integrity and authenticity of digital media and is further enforced by the recently adopted AIA [19].

Just as the created media of the DeepFake manipulations differ (i.e., image, video and audio), so do the creation methods. Mirsky et al. [23] divided the DeepFake generation methods into the four main categories, which are reenactment, replacement, editing and synthesis. Reenactment refers to the controlling of expressions of one person by another person without changing the identity. In contrast, replacement (e.g., face swap) is an attempt of impersonation by replacing the identity. Editing does not require a second identity, instead specific facial traits of the given face are adjusted and changed. Common examples of such forgeries include changing ethnicity, facial hair or age of a face. Akhtar [24] further states the possibility to add injury, effects of drugs or other health-related issues to the image of a person. The last category of face synthesis does not require a particular identity, as it creates new, non-existent persons. In addition, the different approaches of generation have various different generation methods, e.g., encoder-decoder networks or generative adversarial networks (GAN), potential traces of manipulation may vary depending on the generation method.

Li et al. [25] detect DeepFakes by analyzing warping artefacts, which are a common trace in face swap approaches. The DeepFake algorithm generates face images with a fixed size, which are afterwards adjusted by using affine transformation to get the resolution of the face in the target image. This process results in warping artefacts. For synthetic face images mostly GANs, such as StyleGAN [26] or its extensions [27]–[29] are used [24]. Each StyleGAN version introduces its own individual artefacts and fixes issues of its predecessors. In consequence, these architectures leave individual forensic traces, which are comparable to fingerprints. This direction is further explored by Yu et al. [30] and Marra et al. [31]. A more detailed overview of the specific artefacts originating from different generation methods is given in the survey of Akhtar [24]. In terms of DeepFake detection, methods can be divided into spatial and temporal feature spaces [23]. Initially, the focus of detection was solely on the proposal of suitable deep learning based detectors without any form of explanations. More recently publications further prioritise forensic aspects in detection. In [32] DeepFake detection with the consideration of compliance with existing and upcoming regulations are shown.

III. Conceptional Extension and Joining of Data-driven and Media Forensic

For the conceptual connection of data-driven and media forensics, the BPM-DI [4] is considered as a basis and extended for the case of DeepFake detection for video. To classify this further, it should be noted that [4] proposes the application in practice on a specific investigation. According to the phase modeling, this includes the phases *OP*, *DG*, *DI* and *DA*, with *SP* being omitted. In consequence, the tools used for the forensic investigation are assumed to be tested and verified (i.e., its error rates are known by means of benchmarking and also limits of applicability have been identified).

A. Human Operators Involved in the Forensic Investigation

Based on the findings in [1], more attention must be paid to the people involved in the forensic investigation. Table I provides a non-exhaustive list of typical roles of human operators in forensic processes. Note: Here a homogenized version of the ENFSI terminology is used since it slightly varies in terms and definitions between different ENFSI BPM (e.g., 'Case lead' vs. 'Case Leader' vs. 'Section Heads/Operations Managers' in the technical departments of a lab). In this set of typical roles involved, different subsets can be identified: For investigations, the minimal subset involved would be {Customer, Case Leader, System Administrator}. While the roles of the Customer and the Case Leader in a forensic investigation are obvious, the System Administrator is responsible for the availability and technical reliability of the used case management system(s) and the corresponding resources. Since this usually involves a need for elevated access privileges, special care has to be taken to prevent unauthorised access to investigation procedures and results by the System Administrator. In other typical working contexts, e.g., proficiency testing, other roles are involved (in the example of the proficiency testing the minimal subset would contain {Standardization Body, Examiner}).

B. Methods of the Forensic Investigation

In this paper, the considerations on methods used in forensic investigations are based on the categorization provided in [4]. The aspect of **Auxiliary data analysis** (see **Methods** in Figure 1) focuses on all traces of a media file. This includes the **Analysis of external digital context data**, which takes meta data of the file system into account. It can be used to identify potential traces of editing, for example by investigating the modify, access and change (MAC) times. The **File structure**

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

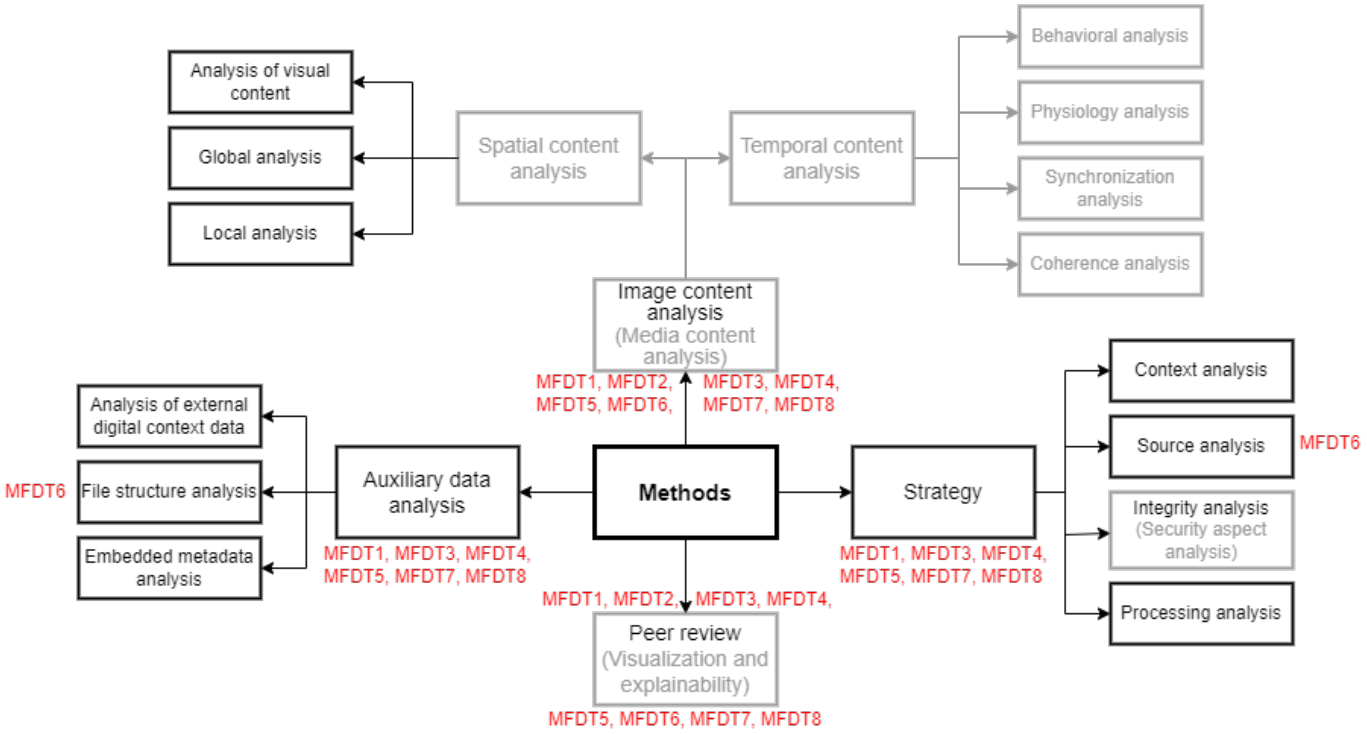


Figure 1. Categorization of forensic methods proposed in [4], extended on the case of media forensics, especially DeepFake detection. Extensions are marked in gray. Integration of media forensic data types (MFDT) can be found in red. Figure redrawn from [1].

TABLE I
TYPICAL ROLES OF HUMAN OPERATORS INVOLVED IN THE PROCESS OF A FORENSIC INVESTIGATION (NON-EXHAUSTIVE LIST). FURTHER SEPARATION BASED ON THE POINT OF TIME THE INVOLVEMENT OCCURS. THE PHASE DO IS OMITTED AS IT IS RELEVANT FOR ALL OPERATIONS PERFORMED.

Role	Phases	Description
System Administrator	SP, OP, DG, DI, DA	Entity responsible for the administration and maintenance of the system. Does not require any case specific information.
Data Scientist	SP	Human operator involved in the training of a machine learning model. The Data Scientist is responsible for managing and curating the datasets.
AI Expert	SP	Performs quality assurance on the feature space. The AI Expert applies explainable AI techniques to the feature space.
Standardization Body	SP	Entity verifying functions and tools. In the context of DeepFake detection this includes performing a benchmark and certifying the trained model.
Customer	OP	Entity requesting the examination of a digital media.
Third Party	OP	Acts as intermediary between Customer and Examiner. Formulates competing hypothesis for the investigation to mitigate potential bias of the Examiner.
Case Leader	OP	Examiner who prioritises the sequence of examination steps to be performed and assigns each to appropriate Examiners.
Examiner	OP, DG, DI, DA	Person(s) performing the forensic investigation of digital media.

analysis covers the examination of the file format. The format found for the examined file is compared with common formats including the specific version number. This can be a clue to the tools used to store the file. For videos, this is also useful to determine the potential origin based on the codec and its version used. **Embedded metadata analysis** takes into account all embedded metadata that can be found in the specific media. These can be used for the two main purposes of identifying the capturing device and gathering more details on the capturing process. For the identification of the capturing device the resolution and corresponding pixel format of images and videos can be used as a first indicator. For audio devices the sampling rate can be used as an equivalent. It is also possible for the device information to be specified in the metadata, but this is optional. For details on the capturing, there are optional metadata regarding the date and time of the recording and the GPS (Global Positioning System) location. In comparison to the BPM-DI [4], no extensions are required so far.

As discussed in Section II-C DeepFakes can occur in image, video as well as audio files. To address this aspect the BPM-DI [4] needs to extend the **Methods** to include spatial and temporal feature spaces in particular. This extension is suggested by a change in two steps, first the **Image content analysis** (see **Methods** Figure 1) has to become broader to also address video files by introducing **Media content analysis**. Second, a further separation of methods is presented, according to the categorization of DeepFake detection methods proposed

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

in [23] dividing into **Spatial** and **Temporal content analysis**. **Methods** of **Spatial content analysis** correspond to BPM-DI [4] **Image content analysis**, which are **Analysis of visual content**, **Global analysis** (i.e., analysis of the entire image) and **Local analysis** (i.e., analysis of a particular image region). These Methods can be found to the left of **Spatial content analysis** in Figure 1.

In contrast, **Temporal content analysis** is another required modality of DeepFake detection. There the first **Method** utilizes the **Behavioral analysis** shown in video or audio. For example in [33] facial movement is analyzed using facial action units to detect DeepFakes of Barack Obama, which is further enforced by the availability of reference data for this person. **Physiology analysis** relies on the assumption, that DeepFake creation lack physiological signals, e.g., in heart rate [34] or eye blinking behavior [22]. **Methods** for **Synchronization analysis** utilize different types of media to validate their correlation. In most cases this is done by extracting features from both audio and video and comparing them against each other. Previous research has been done for example on emotions [35] or lip synchronization [42]. **Coherence analysis** focuses on the aspect, that DeepFakes are created on a frame by frame basis, which might result in flickers and jitters in the video.

The general purpose of the category **Strategy** (see **Methods** in Figure 1) is to categorize previously mentioned **Methods**, both **Auxiliary data analysis** and **Media content analysis**, based on the specific investigation goal. In this work, we consider three of the investigation goals of BPM-DI [4] as they stand and extend the other. These address the correctness of the context the media is put into (**Context analysis**), identification of the device used to capture the media (**Source analysis**) and which processing steps applied to the media (**Processing analysis**). Extensions are made to the **Integrity analysis**, which initially identifies whether the questioned media was altered after acquisition. The extension aims to take into account all security aspects and additionally leave room for future requirements, (e.g., compliance with the AIA [18]). The existing method of **Integrity analysis** can be seen as method within the category of **Security aspect analysis**.

The **Peer review** (see **Methods** in Figure 1) of the BPM-DI [4] is the integration of a human examiner to analyze and interpret results during the whole process. With the introduction of machine learning techniques, especially for DeepFake detection, an extension of this aspect is proposed by introducing techniques to improve **Visualization and explainability**. Its purpose is therefore to support the human examiner in the process of investigation and decision making. With the introduction of machine learning algorithms, special attention has to be paid to the reproduceability of individual methods, their visualization and the entire examination process.

The application of data types is based on the existing 8 media forensic data types (MFDT) [3] mentioned in Section II-A and can also be seen in Figure 1 in red. Since the individual analysis **Methods** are kept generic our assignment of the data types is based on the higher level categories and is the same for the corresponding subcategories. In general, all **Methods** given require a process-accompanying documentation, which are specified to log data (*MFDT7*) and chain of custody & report data (*MFDT8*). Both **Auxilary data analysis** and **Strategy** work on the initial media representations (*MFDT1*), utilizing case specific information (*MFDT3*) and parameters (*MFDT4*) to yield examination data (*MFDT5*). In addition, model data (*MFDT6*) is required for both **File structure analysis** of and **Source analysis** to have a reference model of file structures or camera models respectively. The same can be said for **Media content analysis**, with the addition of various additional representations of the media (*MFDT2*) specific to the method of analysis and the potential usage of machine learning to introduce model data (*MFDT6*). One difference can be found in **Peer review**, in the initial proposal it suggests the analysis and interpretation of media representations (*MFDT2*) and examination data (*MFDT5*). By extending this category to **Visualization and explainability** and the identification of different human operators [5] it further introduces additional data types to be explained. These human operators include, but are not limited to, the forensic investigator, who requires *MFDT2*, *MFDT3*, and *MFDT5*, and the data scientist, who requires *MFDT3*, *MFDT4*, and *MFDT6*. Independent of the human operator, the data types *MFDT1*, *MFDT7* and *MFDT8* are required. In consequence, all MFDTs must be addressed in the method of **Visualization and explainability**.

To enable a more specific and descriptive assignment of the occurring data types, the individual processing steps have to be known, which is specific to the application used for the analysis. This is shown in more detail in the practical example given in Section IV.

IV. Application of DeepFake Detection on the Extended Modelling

To validate the applicability of the proposed extended **Methods** (see Figure 1), a practical application on the example of DeepFake detection is performed on two scenarios. The first scenario describes the forensic examination of an image. Here, an DeepFake image originating from the OpenForensics [36] dataset is selected, which can be found in Figure 3. The image contains two persons, of which only one (the person on the left) contains DeepFake manipulation. In the second scenario, the forensic examination of a video is performed. For this purpose, the DeepFake video 'id0_id1_0000' of Celeb-DF [37] is selected.

In accordance with the methodology discussed above, a total of *9* existing and implemented tools are considered in order to cover a broad spectrum of methods. This is an extension of [1] by a further 6 tools. Initial steps of the forensic investigation begin with the Customer requesting a forensic investigation for specific media data. This request should be made to an independent Third Party, to minimise potential biases of the Examiner. In this paper, the term Third Party is used as specified in [4] as the intermediary between the Customer and the actual investigation. Other ENFSI BPM use the same term with different meanings (e.g., the more traditional meaning

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

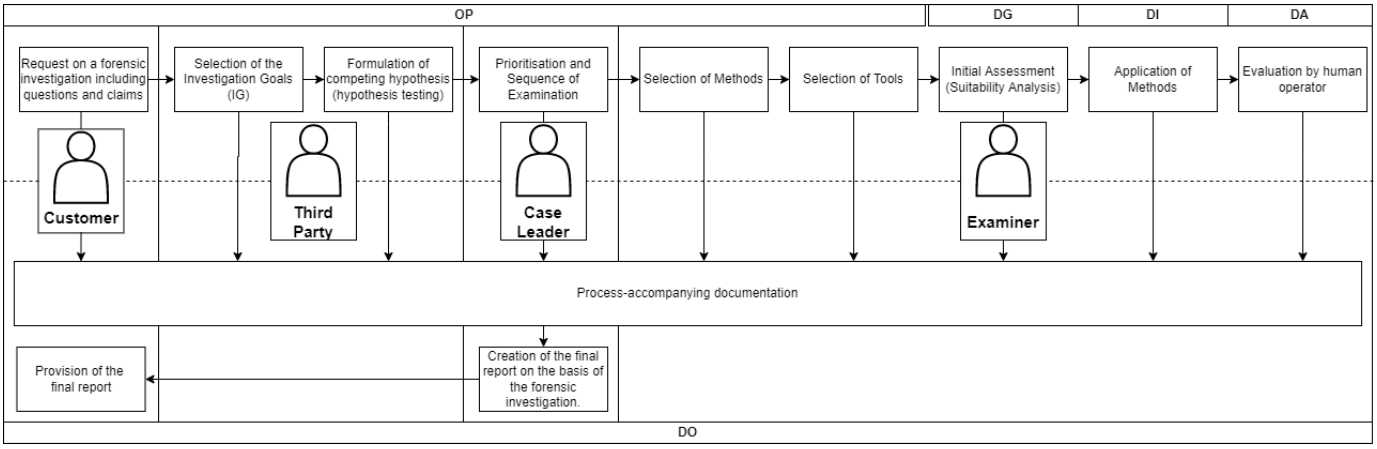


Figure 2. Workflow of an forensic investigation, separation according to the responsible human operator. Corresponding phases of the DCEA [2] are shown at the top and bottom respectively. The dashed line indicates the distinction between phases. For simplicity, log data (MFDT7) and chain of custody & report data (MFDT8) are combined to process-accompanying documentation.

Figure 3. Image selected for application scenario 1. The image shows two persons, of which the left person contains DeepFake manipulation. Image taken from the training set of the OpenForensics [36] dataset (id: 0b02353c85).

of an independent third party subcontractor or an independent lab with similar capabilities for result verification). However, as the focus of this paper is the topic of DeepFake detection, the initial request of a Customer and the request assessment of Third Party are omitted. It further assumes, that the competing hypothesis for the investigation derived are:

- The material under investigation appears to be tampered in a way that indicates a DeepFake manipulation
- There are no traces of post-processing or the identified post-processing seems plausible

It should be noted that the hypotheses considered here are specified for the case of DeepFake detection, as the detectors used are only intended for this purpose. In general, the hypotheses are derived from the request of the Customer and are closely connected to the **Strategies** discussed in Section III-B.

At this point the Case Leader determines and prioritises the investigation steps to be performed and assigns them to the corresponding forensic Examiner. Following the ACE-V methodology, the suitability of the material for the forensic examination has to be validated first. An initial assessment, including exemplary collection of tools for this purpose is discussed in Section IV-A. If none of the material (i.e., individual frames of a video) is found sufficient, no further investigation is performed. As described in [14], it would be possible to use enhancing techniques, but in the image domain these may distort features or introduce artefacts that mislead the examination. It has to be further noted, that the enhancement could possibly also remove traces of DeepFake manipulation. Once the digital media appears suitable for a forensic investigation, the selected Methods are applied. An exemplary application can be found in Section IV-B. The forensic Examiner then evaluates the results gathered for each Method. The entire process is supported by process-accompanying documentation. This documentation consists of log data (MFDT7), which is relevant for the administration of the system, and chain of custody & report data (MFDT8), which is used for forensic investigations reporting. Finally, all reportings are combined in a final report by the Case Leader, which is made available to the Customer. This workflow is further illustrated in Figure 2.

In the following, the individual processing steps and groups of features (hereinafter referred to as PS) as well as individual features (hereinafter referred to as ID) will be labeled and categorized in the extended BPM-DI [4] for **Auxiliary data analysis** (shown in Figure 4), **Media content analysis** (shown in Figure 5) and **Strategies** (shown in Figure 6).

A. Initial Assessment of the Media Under Investigation

To further extend the findings of [1] the initial assessment of the digital media is carried out using the methods of **Auxiliary data analysis** and **Spatial content analysis**. To analyse the metadata of the media both ExifTool [38] (PS-exif) and FFmpeg [39] (PS-ffmpeg) are used. While a variety of entries are available in the metadata, a total of eight features (ID-exif$_n$) are selected from ExifTool for this exemplary

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

approach and categorized according to the Ext. BPM-DI. Three features are selected from FFmpeg (ID-ffmpeg$_n$), two of which correspond to ExifTool features, so that a direct comparison between these two tools is possible. The first set of three Exiftool features address **Analysis of external digital context data** with the aim of **Processing analysis**. These can give first indications of possible manipulations, for example by validating timestamps for modification, access and creation (ID-exif$_1$), file size (ID-exif$_2$) or system feature flags such as user permissions (ID-exif$_3$). Furthermore, three additional features can be used for **File structure analysis**, by extracting the file format (ID-exif$_4$), its format version (ID-exif$_5$) and in case of a video file the used codec (ID-exif$_6$). The extracted information of **File structure** can then be compared to **Standard formats**, unveiling potential traces for **Processing analysis**. In addition, file formats and codecs can give an indication of the software or device to enable **Source analysis** as well. The third set, consisting of two features, which address **Embedded metadata analysis**, with the aim of **Context analysis**, by extracting the media files width and height (ID-exif$_7$, ID-ffmpeg$_1$) and frame rate if it is a video (ID-exif$_8$, ID-ffmpeg$_2$). The features ID-exif$_4$-ID-exif$_8$ can further be used to validate the suitability of subsequent DeepFake detectors. This refers in particular to media properties such as width and height of an image or frame (ID-exif$_7$), frame rate for videos (ID-exif$_8$) and format (ID-exif$_4$) or codec specific compression (ID-exif$_6$) and frame compression (ID-ffmpeg$_3$).

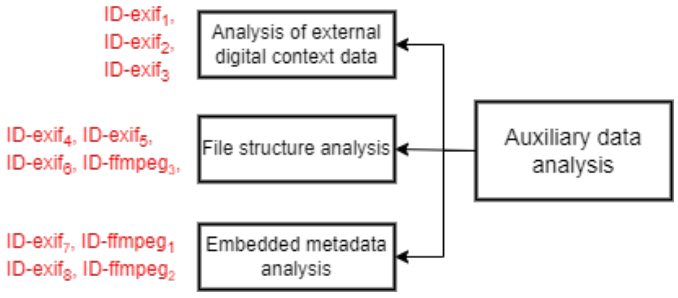


Figure 4. Individual features extracted using ExifTool [38] and FFmpeg [39] (in red) categorized in the extended BPM-DI [4] for the category Auxiliary data analysis. This is an extended version of [1], with the introduction of additional features (ID-ffmpeg).

The second set of Tools utilizes Methods of **Spatial content analysis** to estimate the quality of images, either globally (**Global Analysis**) or in the context of individual faces visible in the images (**Analysis of visual content**). For this exemplary approach, the image quality is estimated using blur (ID-global$_1$) and JPEG compression (ID-global$_2$) detection, which are derived from the frequency domain of the image. Face detection is performed using both MTCNN [40] (ID-face$_1$ and ID-face$_2$) and dlibs 68 landmarks [41] (ID-face$_3$). In addition, the head position (ID-face$_4$) is determined using the roll, yaw and pitch angle on the basis of the landmark positions. As the aforementioned tools and methods are used for the initial assessment, they provide context-specific information (MFDT3). The categorization of Tools in the extended BPM-DI can be found in Table II.

When using the Tools and Methods, it should first be noted, that not every media file contains all the information, as some features are video specific. In consequence, there are no information available regarding video codec (ID-exif$_6$), frame rate (ID-exif$_8$) and frame types (ID-ffmpeg$_3$) in scenario 1. However, FFmpeg provides a frame rate (ID-ffmpeg$_2$), which is due to the fact that it works primarily on video data and thus, incorrectly assumes a motion image. More information can be collected in scenario 2. The frame types in the video show a repeating pattern of an I-frame, followed by 11 P-frames (ID-ffmpeg$_3$) and shows no inconsistencies. Considering the results of both PS-global and PS-face, facial movement can be derived from the value range in ID-face$_4$, which could be the reason for motion blur identified in ID-global$_1$. In the context of the face detection algorithms considered, it should first be noted that only MTCNN provides a confidence measure (ID-face$_2$) for the detected faces. There are also small differences between the MTCNN (ID-face$_1$) and dlib (ID-face$_3$) face detector, as MTCNN did not find a face in two frames. This is a limiting factor for detection approaches based on Methods of Temporal content analysis. However, as the DeepFake detectors considered in Section IV-B utilize the dlib face detector, their suitability is given. In addition, it has to be noted that the *'C:'* in MAC timestamps (ID-exif$_1$) states the time of file change provided by the file system, as the meta data field for creation is empty in both scenarios. A summary of the results collected in this step of initial assessment can be found in Table III.

B. Practical Application of the Extended Methods to DeepFake detection

One of the more promising feature spaces for DeepFake detection utilizes the mouth region, addressing two flaws in DeepFake synthesis. First, the synthesis occurs on a frame-by-frame basis, which results in inconsistencies in the temporal domain, enabling aspects of lip movement analysis. In [42] the detection is performed based on lip synchronization, by considering both audio and video and detecting inconsistencies between phonemes in audio and visemes in video. A similar approach has been taken for the LipForensics detector [43] by identifying unnatural mouth movement. The second aspect utilizes the post processing, especially blurring, performed in DeepFake synthesis. In [44] and [45] texture analysis is performed on the mouth region to identify manipulations. A combination of both approaches is given in [46], where hand-crafted features are used to detect DeepFakes based on mouth movement and teeth texture analysis described as DF_{mouth}.

To evaluate the suitability of the proposed Ext. BPM-DI modeling for DeepFake detection the two detectors DF_{mouth} [46] and LipForensics [43] are selected, representing a hand-crafted as well as deep learning based detector. Both address **Media content analysis**, **Strategies** and **Peer review**. In addition, DF$_{mouth}$ utilizes the features ID-exif$_7$ and ID-exif$_8$ of **Auxiliary data analysis** for internal feature

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II
COLLECTION OF TOOLS AND FEATURES USED FOR THE INITIAL ASSESSMENT OF DIGITAL IMAGE AND VIDEO DATA. CATEGORIZATION BASED ON THE PROPOSED EXTENDED BPM-DI [1]. THE SUITABILITY FOR A FORENSIC EXAMINATION IS HIGHLIGHTED IN BOLD AND ITALIC, WHERE BOLD VALUES INDICATE HIGHER SUITABILITY FOR VALUES CLOSER TO THE UPPER BOUNDARY. IN CONTRAST, ITALIC VALUES INDICATE A HIGHER SUITABILITY CLOSE TO THE LOWER BOUNDARY.

Ext. BPM-DI		feature	description	value	processing step	analysis	strategy	data type
Auxiliary data analysis	Analysis of external digital context data	ID-exif$_1$	MACtime	timestamp	PS-exif	File system metadata	Processing analysis	MFDT3
		ID-exif$_2$	file size	string				
		ID-exif$_3$	system feature flags					
	File structure analysis	ID-exif$_4$	file format	string		File structures	Source & Processing analysis	
		ID-exif$_5$	file format version	version number				
		ID-exif$_6$	video codec	string				
	Embedded meta-data analysis	ID-exif$_7$	file resolution	int [0, ∞]		Additional metadata	Context analysis	
		ID-exif$_8$	file frame rate	real [0, ∞]				
	Embedded meta-data analysis	ID-ffmpeg$_1$	file resolution	int [0, ∞]	PS-ffmpeg	Additional metadata	Context analysis	
		ID-ffmpeg$_2$	file frame rate	int [0, ∞]				
	File structure analysis	ID-ffmpeg$_3$	frame types	string		File structures	Processing analysis	
Media content analysis	Spatial content analysis	ID-global$_1$	image blur estimation	**real [0, 1]**	PS-global	Global analysis	Processing analysis	MFDT3
		ID-global$_2$	JPEG compression estimation	*real [0, 1]*				
		ID-face$_1$	Face detector MTCNN (ROI)	5 landmarks	PS-face	Analysis of visual content	Context analysis	
		ID-face$_2$	Face detector MTCNN (confidence)	**real [0, 1]**				
		ID-face$_3$	face detector dlib (ROI)	68 landmarks				
		ID-face$_4$	face orientation	real [-90, 90]				

TABLE III
RESULTS OF THE INITIAL ASSESSMENT FOR SCENARIO 1 (IMAGE) AND SCENARIO 2 (VIDEO). VALUES IDENTIFIED FOR SCENARIO 2 ARE GIVEN AS RANGES, TO PROVIDE AN OVERVIEW OF ALL FRAMES.

feature	scenario 1	scenario 2
ID-exif$_1$	M: 2021:02:22 22:19:00+01:00 A: 2024:03:14 15:21:16+01:00 C: 2024:03:14 15:21:05+01:00	M: 2019:11:13 14:17:36+01:00 A: 2024:03:14 01:00:00+01:00 C: 2024:03:14 17:02:07+01:00
ID-exif$_2$	385 kB	2.1 mB
ID-exif$_3$	r w - r - - r - -	r w - r - - r - -
ID-exif$_4$	image/jpeg	video/mp4
ID-exif$_5$	1.01	0.2.0
ID-exif$_6$	-	-
ID-exif$_7$	1024x683	944x500
ID-exif$_8$	-	30
ID-ffmpeg$_1$	1024x683	944x500
ID-ffmpeg$_2$	25/1	30/1
ID-ffmpeg$_3$	-	repeating pattern of 11 P-frames between single I-frames
ID-global$_1$	no blur: 0.1160	blur: [0.0163, 0.0232]
ID-global$_2$	not compressed: 0.0699	not compressed: [0, 0.0004]
ID-face$_1$	2 faces found	1 face found in 467 of 469 frames
ID-face$_2$	left face: 1 right face: 0.9999	[0.9838, 0.9999]
ID-face$_3$	2 faces found	1 face found in 469 of 469 frames
ID-face$_4$	left face: {-4.64, 8.34, -24.99} right face: {-2.73, 1.32, -26.94}	Roll: [-21.25, 15.75] Yaw: [-10.84, 40.94] Pitch: [-35.32, 10.00]

normalization. With their intention of identifying DeepFakes the general **Strategy** of application is **Integrity analysis**. Starting with the SP phase for DF$_{mouth}$, the detector is introduced in [46] and trained using the WEKA machine learning toolkit [47]. For the classification the decision tree classifier J48 [48] is used on the datasets DeepfakeTIMIT [49], [50], Celeb-DF [37] and DFD [51]. Detection performance peaks at 96.3% accuracy on a distinct training and test split of DFD. Considering distinct datasets for training and testing, detection performance peaks at 76.4% accuracy trained on DeepfakeTIMIT and tested on DFD. In a later benchmark approach given in [5] DF$_{mouth}$ is applied on a larger variety of DeepFake synthesis methods, including FaceForensics++ [51], DFD [51], Celeb-DF [37] and HiFiFace [52]. With an achieved detection performance of 69.9% accuracy the approaches suitability is identified only for certain DeepFake synthesis methods. With the limitations of DF$_{mouth}$ in mind, it is first split into five processing steps and categorized according the extended model. The individual features are then used for decision support by human operator, using the thresholds provided by the classifier in [46].

1) The video under investigation is first split into individual frames (PS-mouth$_1$) to first focus on **Spatial content analysis**.
2) For each frame a face detection algorithm is applied, in [46] using dlib's 68 landmark detection model [41] to extract the corresponding region for the mouth region (PS-mouth$_2$), which shows a dependency on the underlying model for face detection.
3) Then in PS-mouth$_3$, based on the keypoint geometry, it is determined whether the mouth is open (referred to as "state 1") or closed ("state 0"). Furthermore, the occurrence of teeth (referred to as "state 2") are examined based on texture analysis.
4) Based on the extracted mouth region and the information gathered, a total of 16 features are extracted. The first set of features, ID-mouth$_1$-ID-mouth$_7$ and ID-mouth$_{12}$ refer to **Physiological analysis** by describing mouth movements and the presence of teeth, by embedding individual frame features back into the temporal context of the video (PS-mouth$_4$). With the idea of DeepFakes having fewer mouth movements, values closer to 0 indicate a DeepFake for the features ID-mouth$_1$-ID-mouth$_6$. Features ID-mouth$_7$ and ID-mouth$_{12}$ aim to identify potential post-processing of the media, where lower values in ID-mouth$_{12}$ and higher values in ID-mouth$_7$ indicate a DeepFake. These are used for **Context analysis** to identify temporal inconsistencies. The normalization of features is done based on the frame rate (ID-exif$_8$) identified in **Auxiliary data analysis**.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

5) The second group of features (PS-mouth$_5$), which consist of ID-mouth$_8$-ID-mouth$_{11}$ and ID-mouth$_{13}$-ID-mouth$_{16}$, refers to **Local analysis** to describe the sharpness of objects (here mouth and teeth region). In general, higher values for the features addressing state 1 (ID-mouth$_8$-ID-mouth$_{11}$) and lower values for the features addressing state 2 (ID-mouth$_{13}$-ID-mouth$_{16}$) indicate a potential DeepFake. The underlying **Strategy** is **Processing analysis**. The normalization of features is done based on the video frame resolution (ID-exif$_7$) identified in **Auxiliary data analysis**.

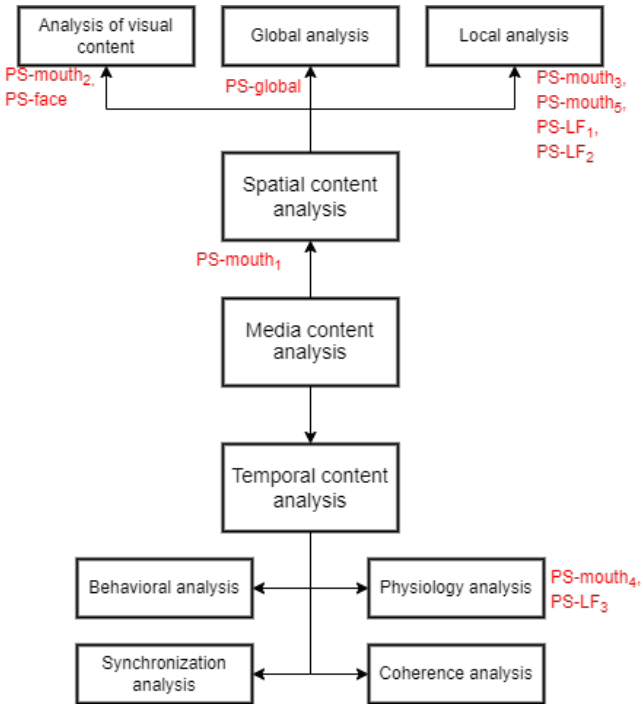


Figure 5. Processing steps (PS, in red) for ExifTool [38] and the DeepFake detectors DF$_{mouth}$ [46] and LipForensics [43] categorized in the extended BPM-DI [4] for the category Media content analysis. This is an extended version of [1], with the introduction of additional features (PS-face and PS-global).

More details on the individual features, their description as well as the categorization in the forensic methods can be found in the upper part of Table IV. Although all features can be categorized as *MFDT5*, the individual processing steps are more complex, containing multiple data types. For a more detailed description, the reader is referred to [22].

The second detector LipForensics [43] (herinafter referred to as LF) is included on a theoretical basis. For LF a total of three PS can be identified.

1) In the first step (PS-LF$_1$) the preprocessing occurs. First, a total of 25 frames are extracted from the video. These frames are converted to grayscale images, cropped to the mouth region and scaled to a resolution of 88x88. The resulting image representation can be categorized as *MFDT2*. With the intend of using only the mouth region, the corresponding method is **Local analysis** and the underlying strategy **Context analysis**.
2) In PS-LF$_2$ the feature extraction is done using a pre-trained ResNet-18 architecture trained on lip reading (*MFDT6*). As the result a feature vector of size 512 is generated (*MFDT3*). Again, the corresponding method is **Local analysis** and the underlying strategy **Context analysis**.
3) The resulting feature vector is used for classification purposes (PS-LF$_3$) using a multiscale temporal convolutional network (MS-TCN). The classification result *MFDT5* contains a classification label and the corresponding probability. With the aim of identifying unnatural behavior in mouth movement the corresponding method is **Physiology analysis** and the strategy of **Processing analysis**.

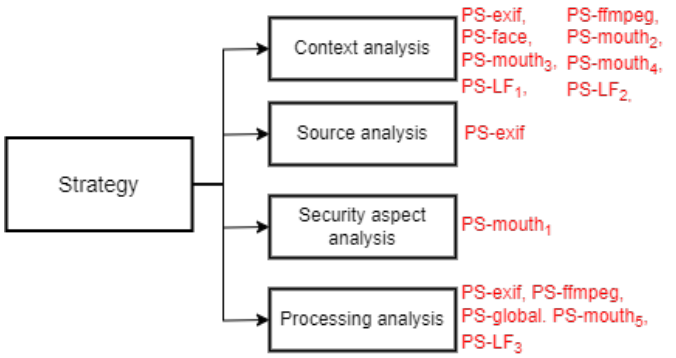


Figure 6. Processing steps (PS, in red) for ExifTool [38] and the DeepFake detectors DF$_{mouth}$ [46] and LipForensics [43] categorized in the extended BPM-DI [4] for the category Strategy. This is an extended version of [1], with the introduction of additional features.

For the practical implementation, starting with scenario 1, it has to be noted that LipForensics is not suitable for image data. In contrast, DF$_{mouth}$ also uses spatial features (PS-mouth$_5$), providing an indication for potential DeepFake manipulation. Only four of the sixteen features can be extracted with a single image, as only one state of the mouth can be given. In addition, when looking at the picture, it can be seen that both people show a closed mouth (state 0), which does not contribute to the decision for DF$_{mouth}$. However, applying the algorithm to the image classifies the left person's mouth as open with visible teeth (state 2) and the right person's mouth as open without visible teeth (state 1). The reason for this is the inaccurate position of the landmarks, particularly in the area of the lower lip margin. The assignment of the left face to mouth state 2, on the other hand, is less plausible, especially as the texture details are very small as shown by ID-mouth$_{15}$ = 0, for example. These findings, are a indication of possible manipulation, but since DF$_{mouth}$ operates outside the boundaries of its intended use, they cannot be directly related to DeepFake manipulation. A more detailed analysis of the features is possible in scenario 2 as it is a video and both LF and DF$_{mouth}$ are applicable. However, as LF requires all videos of the Celeb-DF dataset to evaluate, only the pre-

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE IV
CATEGORIZATION OF DF_{mouth} [46] (TOP SECTION) AND LIPFORENSICS [43] (BOTTOM SECTION) IN THE FORENSIC CONTEXT, BASED ON THE PROPOSED EXTENDED BPM-DI. FOR FEATURE VALUES HIGHLIGHTED IN BOLD HIGHER VALUES INDICATE A DEEPFAKE AND FOR ITALIC LOWER VALUES INDICATE A DEEPFAKE, PRESENTED IN [1].

Ext. BPM-DI		feature	description	value	processing step	analysis	strategy	data type
Media content analysis	Temporal content analysis	ID-mouth$_1$	abs max change Y	*real [0, ∞]*	PS-mouth$_4$	Physiology analysis	Context analysis	MFDT5
		ID-mouth$_2$	max change Y	*real [0, ∞]*				
		ID-mouth$_3$	min change Y	**real [-∞, 0]**				
		ID-mouth$_4$	abs max change X	*real [0, ∞]*				
		ID-mouth$_5$	max change X	*real [0, ∞]*				
		ID-mouth$_6$	min change X	**real [-∞, 0]**				
		ID-mouth$_7$	percentage time state 1	**real [0, 1]**				
		ID-mouth$_{12}$	percentage time state 2	*real [0, 1]*				
	Spatial content analysis	ID-mouth$_8$	max regions state 1	*real [0, ∞]*	PS-mouth$_5$	Local analysis	Processing analysis	
		ID-mouth$_9$	max FAST keypoints state 1	*real [0, ∞]*				
		ID-mouth$_{10}$	max SIFT keypoints state 1	*real [0, ∞]*				
		ID-mouth$_{11}$	max sobel pixel state 1	*real [0, ∞]*				
		ID-mouth$_{13}$	min regions state 2	**real [0, ∞]**				
		ID-mouth$_{14}$	min FAST keypoints state 2	**real [0, ∞]**				
		ID-mouth$_{15}$	min SIFT keypoints state 2	**real [0, ∞]**				
		ID-mouth$_{16}$	max sobel pixel state 2	**real [0, ∞]**				
Media content analysis	Spatial content analysis	ID-LF$_1$	extraction of 25 frames, grayscale, crop and align	int [0, 255]	PS-LF$_1$	Local analysis	Context analysis	MFDT2
		ID-LF$_2$	feature extraction utilizing ResNet-18	feature vector of size 512	PS-LF$_2$	Local analysis	Context analysis	MFDT3
	Temporal content analysis	ID-LF$_3$	classification of mouth movement based on MS-TCN	label: {real, fake} probability: real [0, 1]	PS-LF$_3$	Physiology analysis	Processing analysis	MFDT5

processing part of the detector (LF_1) is considered. DF_{mouth} provides more details on discussion and interpretation. Considering the contents of the video in conjunction with the video, the percentage of time the person is showing an open mouth (ID-mouth$_7$ and ID-mouth$_{12}$, adding up to 0.3091) appears to be relatively low, since the person is talking in the video. The texture analysis (ID-mouth$_8$-ID-mouth$_{11}$ and ID-mouth$_{13}$-ID-mouth$_{16}$) does not provide any indications on DeepFake manipulation as the individual features are not that high or low respectively. Also, the DeepFake detector classifies this Video as no DeepFake in this instance. In consequence, there are indications of post-processing in the mouth region, but they cannot be identified as DeepFake manipulation. The full set of features extracted for both scenarios can be found in Table V.

TABLE V
RESULTS OF THE DEEPFAKE DETECTION FOR SCENARIO 1 (IMAGE) AND SCENARIO 2 (VIDEO).

feature	scenario 1	scenario 2
ID-mouth$_1$	-	3.8333
ID-mouth$_2$	-	3.8333
ID-mouth$_3$	-	-3.0804
ID-mouth$_4$	-	1.4904
ID-mouth$_5$	-	1.4904
ID-mouth$_6$	-	-1.2923
ID-mouth$_7$	-	0.2878
ID-mouth$_{12}$	-	0.0213
ID-mouth$_8$	right face: 0.0469	0.1148
ID-mouth$_9$	right face: 0.0781	0.0714
ID-mouth$_{10}$	right face: 0.0313	0.0492
ID-mouth$_{11}$	right face: 6.3438	7.7049
ID-mouth$_{13}$	left face: 0.0556	0.0441
ID-mouth$_{14}$	left face: 0.0741	0.0526
ID-mouth$_{15}$	left face: 0	0.0441
ID-mouth$_{16}$	left face: 6.6852	7.3659
ID-LF$_1$	2 images of mouth	469 images of mouth
ID-LF$_2$	-	-
ID-LF$_3$	-	-

With the introduction of machine learning algorithms in combination with previously discussed aspects of human in control and human oversight, the **Peer review** component becomes even more important. Its aim should be to enable the human operator to validate the results of each machine learning step to reduce the potential for error. Figure 7 demonstrates a potential direction to enhance the Method of **Peer review** on the basis of DF_{mouth} and ExifTool [54]. In general, the aim of this visualization is to remove the decision-making from the detector. Instead, the individual features are displayed and evaluated by the human operator. To enable the advanced methodology and the human operator to make a decision, this first conceptual example consists of four segments.

1) A filter for the forensic Methods of analysis (i.e., Auxiliary data analysis and Media content analysis), Strategy, detector and data type (see the top left box of Figure 7). Based on the selected features only suitable features are shown and selectable for further investigation.
2) The second block (see the top right box of Figure 7) acts as media player. It has different views to either visualize the video, individual frames (including potential

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

visualizations for explainability) and the metadata.
3) Based on the selected feature, this element shows its categorization in the forensic Methods and visualizes its value for each frame (see the bottom left box of Figure 7).
4) The last block (see the bottom right box of Figure 7) integrates the human operator in the decision-making process. The operator is provided with questions based on specific features and values to identify potential errors of the algorithm. In addition, the detectors thresholds for classification are provided without the decision itself. This is done to reduce the risk of bias by the Examiner based on the decision being provided.

In addition, it should be noted that each step in the pipeline discussed involving machine learning for DF_{mouth} could also have been performed by manually labeling the data to reduce the error susceptibility. However, this would come at the expense of the required review time, especially for long videos with high frame rates.

This potential usage of machine learning indicates the necessity of the *SP* phase within the investigation process. Models have to be benchmarked properly to identify both error rates and potential limitations in their usage, to comply with the Daubert criteria discussed previously [17]. Furthermore, in the context of forensic investigations they have to be certified, so that these are approved for the investigation. These required steps must be performed before the actual investigation in the *SP* phase, which is not considered in the BPM-DI, in contrast to our extended BPM-DI.

V. Conclusion and Future Work

In this work an extension to the ENFSI BPM for digital image authentication is proposed, utilizing data-driven forensics by adding the eight media forensic data types (MFDT) from DCEA [2], [3] in **Methods** of BPM-DI [4]. This makes it possible to establish a connection between existing practices in forensic science and DeepFake detection. On the one hand, the forensic investigataion steps, that are necessary for DeepFake detection are shown. On the other hand, the necessary requirements are outlined, particularly with regard to the provision of information generated by the detector. In addition, extensions are proposed in the **Media content analysis Methods** using **Spatial** and **Temporal content analysis** to reflect the typical analysis domain of DeepFake detection (and other video authentication methods). Furthermore, the extension of the **Peer review** component to address also **Visualization and explainability** was touched upon by introducing a graphical interface that provides further information about the internal processing of the DeepFake detector. Here, the aspects 'human in the loop' and 'human in control' as well as the topic 'explainable AI' represent important foundations for this component as there are different operators involved in the forensic examination process.

The extended BPM-DI model is applied the forensic investigation process of image and video data. By using a total of nine existing and implemented tools as methods the applicability can be shown. Potential limitations and errors have been shown, as the selected DeepFake detectors of DF_{mouth} and LipForensics are intended for the analysis of video data. In addition, it was found that the deep learning based features are too complex to achieve the same granularity as the detector DF_{mouth}. Another limitation resulted from the structuring according to the phases, as suggested in DCEA. By omitting the Strategic Preparation (SP) phase, the detection approaches introduced for investigation have to be trained, benchmarked and certified beforehand. On this basis, the suitability of the individual detectors for the respective investigation must be determined, but this is not possible without prior knowledge of SP. In order to compensate for these disadvantages, the preliminary work of the detectors under consideration was taken as the findings of the SP phase and further verified in the application within this paper. Moreover, the interplay of individual **Methods** have been identified. Starting with the initial assessment of the media provides further insights for the suitability of individual detection approaches.

The evaluation of the proposed methodology on the two examples shows the need to manually verify particular aspects of the algorithms used. In the first scenario, relating to the analysis of image media, both DF_{mouth} and LipForensics act outside their intended use. While Lipforensics cannot be used, DF_{mouth} would only analyse one face in its default configuration. By adding the **Analysis of visual content** in the Initial Assessment the investigation can be extended to both faces shown in the image. This also provides the insight that both faces show a closed mouth, which differs from the results of DF_{mouth}. Consequently, the joining of multiple methods leads to inconsistencies being unveiled, which remain hidden when viewed in isolation. The same applies to the second scenario, in which a video is the subject of the investigation. In particular, information from the **Auxiliary data analysis** as a result of the Initial Assessment can be applied. It includes the usage for feature engineering and normalization as shown in PS-$mouth_4$ and PS-$mouth_5$. Furthermore, PS-$mouth_4$ states (see Table IV and Figure 5), that spatial traces can be utilized in the temporal context as well. In addition, the **Auxiliary data analysis** can be further used as a selection strategy for single-image approaches. This applies in particular to the individual frame types in the video (ID-$ffmpeg_3$). However, the DeepFake could not be identified with certainty in either scenario. Instead, the manual review of the results revealed inconsistencies that indicate changes to the media. This highlights the importance of the transparency and interpretability of the algorithms used, which is required for the forensic examiner to comprehend the results.

In contrast to our previous work in [1], the applicability of the proposed extended BPM-DI has been further improved by adding six tools to a total of nine. However, not all methods of the proposed model could be covered with the selected detectors. This shows that individual tools cannot and should not cover all methods. This is further supported by the findings in [46], where DF_{mouth} is only one of three modalities used for DeepFake detection. In relation to these modalities, ENFSI also provides a list of facial features in their "Best Practice

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

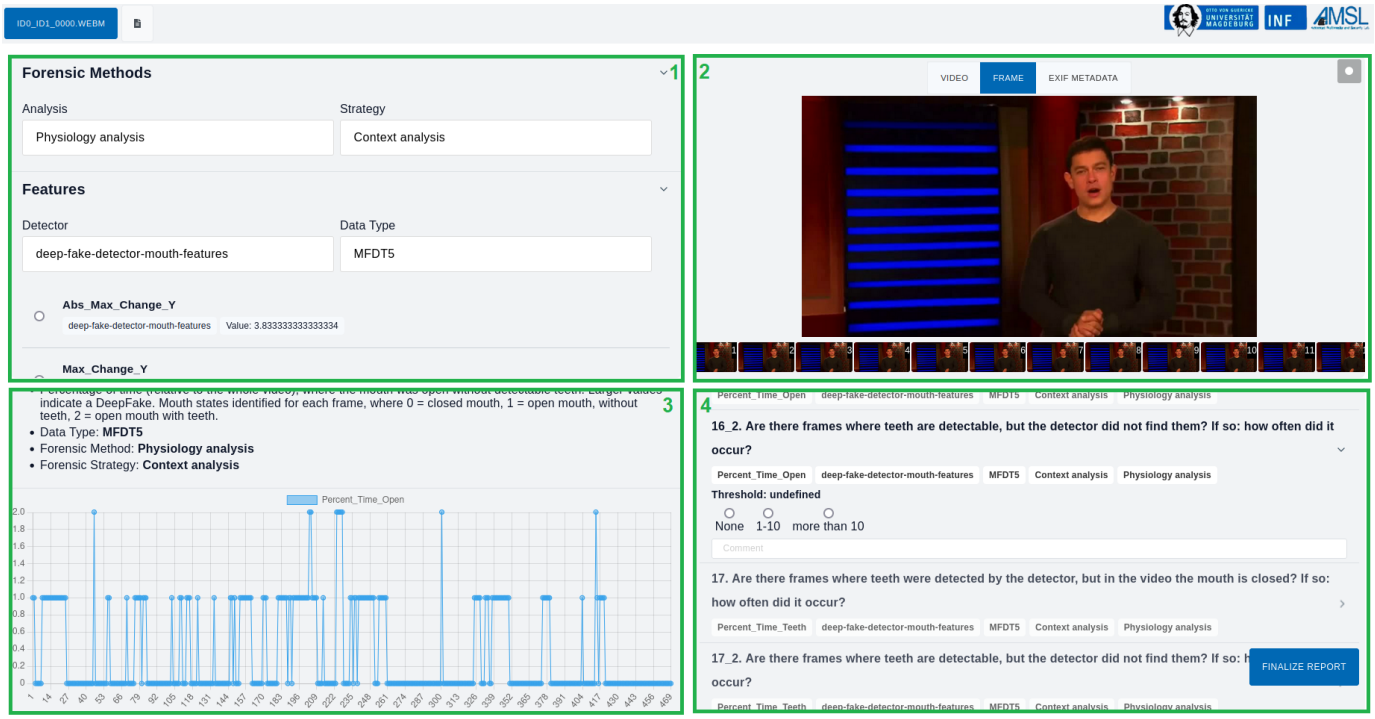


Figure 7. Demonstration of the extended Methods, exemplified on DF_{mouth} for video id0_id1_0000 of the Celeb-DF dataset - from a student project in the context of the lecture "Multimedia and Security", 2023 Department of Computer Science, Otto-von-Guericke-University of Magdeburg, expanded version of [1], further highlighting the individual areas of the interface.

Manual for Facial Image Comparison" [10] that can be used as a reference. A further extension can be derived from the findings in the SP phase, as the detectors can generally only detect certain types of DeepFakes, which is related to the specific traces of manipulation in the media. Lastly, it was also discussed that DeepFakes can occur in audio data, which is not specifically included in the extended model. For this purpose, there is the "Best Practice Manual for Digital Audio Authenticity Analysis" [53], which has to be addressed in the future.

Author Contributions and Acknowledgments

The work in this paper is funded in part by the German Federal Ministry of Education and Research (BMBF) under grant number FKZ: 13N15736 (project "Fake-ID"). Special thanks to team TraceMap, consisting of Stephan Haussmann, Hannes Hinniger, Tjark Homann and Malte Rathjens (a student project in the context of the lecture "Multimedia and Security", 2023 Department of Computer Science, Otto-von-Guericke-University of Magdeburg) for providing an initial demonstrator used as a basis for Figure 7.

Author Contributions: Initial idea & methodology: Jana Dittmann (JD), Christian Kraetzer (CK); Conceptualization: Dennis Siegel (DS); Modeling & application in the context of DeepFake: DS; Writing – original draft: DS; Writing – review & editing: CK, Stefan Seidlitz (StS), JD and DS.

References

[1] D. Siegel, C. Kraetzer, and J. Dittmann. "Joining of Data-driven Forensics and Multimedia Forensics for Deepfake Detection on the Example of Image and Video Data," *Proceedings of the SECURWARE 2023, The Seventeenth International Conference on Emerging Security Information, Systems and Technologies*, IARIA, 2023, pp. 43–51.

[2] S. Kiltz. "Data-centric examination approach (DCEA) for a qualitative determination of error, loss and uncertainty in digital and digitised forensics," *Ph. D. Thesis. Otto-von-Guericke-University Magdeburg, Fakultät für Informatik*, 2020.

[3] D. Siegel, C. Kraetzer, S. Seidlitz, and J. Dittmann. "Forensic data model for artificial intelligence based media forensics - illustrated on the example of DeepFake detection," *Electronic Imaging 34*, 2022, pp. 1–6.

[4] European Network of Forensic Science Institutes (ENFSI). "Best practice manual for digital image authentication," *ENFSI-BPM-DI-03*, 2021.

[5] C. Kraetzer, D. Siegel, S. Seidlitz and J. Dittmann. "Human-in-control and quality assurance aspects for a benchmarking framework for DeepFake detection models," in Electronic Imaging, 2023, pp. 379–1 - 379-6, https://doi.org/10.2352/EI.2023.35.4.MWSF-379.

[6] European Network of Forensic Science Institutes. "Best practice manual for the forensic examination of digital technology," *ENFSI-BPM-FIT-01*, 2015.

[7] M. Reith, C. Carr, and G. H. Gunsch. "An examination of digital forensic models," *Int. J. Digit. EVid. 1, 3*, 2002.

[8] R. Böhme, F. C. Freiling, T. Gloe, and M. Kirchner. "Multimedia forensics is not computer forensics," *Computational Forensics*, Springer, 2009, pp. 90–103.

[9] European Network of Forensic Science Institutes (ENFSI). "Best practice guidelines for ENF analysis in forensic authentication of digital evidence (BPM-ENF-001)," Technical Report, ENFSI, Wiesbaden, Germany, 2009.

[10] European Network of Forensic Science Institutes (ENFSI). "Best practice manual for facial image comparison," *ENFSI-BPM-DI-01*, 2018.

[11] National Institute of Justice. "Fingerprint Sourcebook," 2011.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[12] German Federal Office for Information Security (BSI). "Open source face image quality (OFIQ)," *online available at https://www.bsi.bund.de/dok/OFIQ-e*, last access: 2024-05-31.

[13] T. Schlett, C. Rathgeb, O. Henniger, J. Galbally, J. Fierrez, and C. Busch. "Face image quality assessment: a literature survey," *ACM Computing Surveys, Volume 54, Issue 10s*, 2022, pp. 1-49.

[14] European Network of Forensic Science Institutes (ENFSI). "Best practice manual for forensic image and video enhancement," *ENFSI-BPM-DI-02*, 2018.

[15] R. Altschaffel. "Computer forensics in cyber-physical systems : applying existing forensic knowledge and procedures from classical IT to automation and automotive," *Ph. D. Thesis. Otto-von-Guericke-University Magdeburg, Fakultät für Informatik*, 2020.

[16] Legal Information Institute. "Rule 702. testimony by expert witnesses," 2019 *online available at https://www.law.cornell.edu/rules/fre/rule_702*, last access: 2024-05-31.

[17] C. Champod and J. Vuille. "Scientific evidence in europe - admissibility, evaluation and equality of arms," *International Commentary on Evidence 9, 1*, 2011.

[18] European Commission. "Proposal for a regulation of the european parliament and of the council laying down harmonisd rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts," *COM/2021/206 final*, April,21 2021.

[19] European Parliament. "Amendments adopted by the european parliament on 14 june 2023 on the proposal for a regulation of the european parliament and of the council on laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts," *(COM(2021)0206 – C9-0146/2021 – 2021/0106(COD))*, June, 14 2023.

[20] United Nations Interregional Crime and Justice Research Institute (UNICRI) and International Criminal Police Organization (INTERPOL). "Toolkit for responsible AI innovation in law enforcement: principles for responsible AI innovation," *online available at https://unicri.it/Publication/Toolkit-for-Responsible-AI-Innovation-in-Law-Enforcement-UNICRI-INTERPOL*, last access: 2024-05-31.

[21] National Institute of Standards and Technology (NIST). "Digital and multimedia evidence," *online available at https://www.nist.gov/spo/forensic-science-program/digital-and-multimedia-evidence*, last access: 2024-05-31.

[22] C. Kraetzer, D. Siegel, S. Seidlitz, and J. Dittmann. "Process-driven modelling of media forensic investigations - considerations on the example of deepfake detection," *Sensors 22, 9*, 2022.

[23] Y. Mirsky and W. Lee. "The creation and detection of deepfakes: a survey," *ACM Comput. Surv. 54, 1, Article 7*, 2021.

[24] Z. Akhtar. "Deepfakes generation and detection: a short survey," *Journal of Imaging 9, 1*, 2023

[25] Y. Li and S. Lyu. "Exposing deepfake videos by detecting face warping artifacts," In Proceedings of the *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1-7.

[26] T. Karras, S. Laine, and T. Aila. "Training generative adversarial networks with limited data," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[27] T. Karras, S. Laine, T. Aila, J. Hellsten, J. Lethinen, and T. Aila. "Analyzing and improving the image quality of styleGAN," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[28] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lethinen, and T. Aila. "Training generative adversarial networks with limited data," *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020.

[29] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lethinen, and T. Aila. "A style-based generator architecture for generative adversarial networks," *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, 2021.

[30] N. Yu, L. Davis, and M. Fritz. "Attributing fake images to GANs: learning and analyzing GAN fingerprints," Proceedings of the *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[31] F. Marra, D. Gragnaniello, L. Verdoliva, and G. Poggi. "Do GANs leave artificial fingerprints?," *IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2019, pp. 506-511.

[32] B. Lorch, N. Scheler, and C. Riess. "Compliance challenges in forensic image analysis under the artificial intelligence act," *30th European Signal Processing Conference (EUSIPCO)*, 2022, pp. 613–617.

[33] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li. "Protecting world leaders against deep fakes," *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019 pp. 38–45.

[34] V. Conotter, E. Bodnari, G. Boato, and H. Farid. "Physiologically-based detection of computer generated faces in video," *IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 248–252.

[35] B. Hosler, D. Salvi, A. Murray, F. Antonacci, P. Bestagini, S. Tubaro, and M. C. Stamm. "Do deepfakes feel emotions? a semantic approach to detecting deepfakes via emotional inconsistencies," *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021, pp. 1013–1022.

[36] T.-N. Le, H.H. Nguyen, J. Yamagishi, and I. Echizen. "OpenForensics: large-scale challenging dataset for multi-face forgery detection and segmentation in-the-wild," *International Conference on Computer Vision (ICCV)*, 2021.

[37] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu. "Celeb-df: a large-scage challenging dataset for deepfake forensics," *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 3204–3213, doi:10.1109/CVPR42600.2020.00327.

[38] P. Harvey. "Exiftool," *online available at https://exiftool.org/*, last access: 2024-05-31.

[39] FFmpeg developers. "FFmpeg," *online available at https://ffmpeg.org/*, last access: 2024-05-31.

[40] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters 23(10)*, 2016, pp. 1499-1503.

[41] D. E. King. "Dlib-ml: a machine learning toolkit," *J. Mach. Learn. Res. 10*, 2009, pp. 1755–1758.

[42] S. Agarwal, H. Farid, O. Fried, and M. Agrawala. "Detecting deep-fake videos from phoneme-viseme mismatches," *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2814–2822.

[43] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic. "Lips don't lie: a generalisable and robust approach to face forgery detection," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 5039-5049.

[44] F. Matern, C. Riess, and M. Stamminger. "Exploiting visual artifacts to expose deepfakes and face manipulations," *IEEE Winter Applications of Computer Vision Workshops (WACVW)*, 2021, pp. 83–92.

[45] A. Elhassan, M. Al-Fawa'reh, M. T. Jafar, M. Ababneh, and S. T. Jafar. "DFT-MF: enhanced deepfake detection using mouth movement and transfer learning," *SoftwareX 19*, 2022.

[46] D. Siegel, C. Kraetzer, S. Seidlitz, and J. Dittmann. "Media forensics considerations on deepfake detection with hand-crafted features," *Journal of Imaging 7, 7*, 2021.

[47] M. A. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. "The weka data mining software: an update," SIGKDD Explor., 11(1):10–18, 2009.

[48] J. R. Quinlan. "C4.5: programs for machine learning," *Morgan Kaufmann Publishers Inc.*, San Francisco, CA, USA, 1993.

[49] P. Korshunov and S. Marcel. "Deepfakes: a new threat to face recognition? Assessment and detection," *arXiv preprint arXiv:1812.08685*, 2018.

[50] C. Sanderson and B. Lovell. "Multi-region probabilistic histograms for robust and scalable identity inference," *Lecture Notes in Computer Science (LNCS)*, 2009, pp. 199–208.

[51] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. "Faceforensics++: learning to detect manipulated facial images," *International Conference on Computer Vision (ICCV)*, 2019.

[52] Y. Wang, X. Chen, J. Zhu, W. Chu, Y. Tai, C. Wang, J. Li, Y. Wu, F. Huang, and R. Ji. "Hififace: 3d shape and semantic prior guided high fidelity face swapping," *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021, pp. 1136-1142.

[53] European Network of Forensic Science Institutes. "Best practice manual for digital audio authenticity analysis". *ENFSI-FSA-BPM-002*, 2018.

[54] D. Siegel and J. Dittmann. "TraceMap," Student project within the lecture of Multimedia and Security [MMSEC], Otto-von-Guericke-University Magdeburg, 2023, unpublished.

[55] German Federal Office for Information Security (BSI). "Leitfaden IT-forensik," *online available at https://www.bsi.bund.de/dok/6620610*, last access: 2024-05-31, available in German only, 2011.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Monitoring Physical-World Access of Virtual Automation Functions

Rainer Falk and Steffen Fries
Siemens AG
Technology
Munich, Germany
e-mail: {rainer.falk|steffen.fries}@siemens.com

***Abstract*—Virtualized automation functions can be used in cyber-physical systems to influence the real, physical world using sensors and actuators connected via input-output modules. At the same time, other virtualized automation functions may be used for planning, testing, or for optimization. A reliable method for determining whether a certain virtualized automation function has access to the real, physical world is proposed, based on a cryptographically protected physical-world access attestation issued by an input/output module. It confirms which virtualized automation function has in fact access to the real-physical world via this input-output module. This allows monitoring which automation functions interact in fact with the real, physical world, and which ones are used for other, less critical purposes.**

Keywords–cyber physical system; virtual automation system; attestation; industrial security; cybersecurity; security monitoring.

I. INTRODUCTION

A Cyber Physical System (CPS) contains control devices that interact with the real, physical world using sensors and actuators. Which automation and control devices are connected via sensors and actuators to the real, physical world has implicitly been clear from the structure of physical control devices, sensors, actuators. their cabling, and the overall system engineering. When control devices are virtualized, e.g., as container or virtual machine, executed on a common compute platform, they interact with the real, physical world using remote Input-Output (IO) modules. However, it is no longer clear implicitly which virtualized control device in fact interacts with the real, physical world, and which ones are used for simulation or optimization. A Physical World Access Attestation (PWAA) can confirm reliably, which automation function accesses a specific IO module [1].

Digital twins, supporting the simulation of the CPS and its control devices, provide the possibility to perform plausibility checks of the measured real-world behavior and the expected, simulated behavior in parallel. This eases the detection of unexpected system behavior, which may indicate a failure situation or even an attack. In addition, virtualization of control devices is increasing, allowing to deploy multiple instances of virtualized control devices that look and behave identically [2]. A virtualized control device can be realized as virtual machine or container hosted on an app-enabled edge device or on a cloud infrastructure by a virtualized Automation Function (vAF). In such a deployment, it has to be distinguished which vAF instances in fact interact with the real, physical world, and which ones are used for other purposes as, e.g., training, optimization, planning, virtual commissioning, simulation, or for testing. The vAF instance that in fact has access to the real physical world is the one that is the most critical, as its operation directly affects the real world.

In the past, CPS have been often rather static. After being put into operation, changes to the configuration happen only rarely, e.g., to replace a defect component, or to install smaller upgrades during a planned maintenance window. To cope with increasing demands for flexible production and increased productivity, CPS will also increasingly become more dynamic, allowing for reconfiguration during regular operation. Such scenarios for highly adaptive production system that can be adjusted flexibly to changing production needs have been described in the context of Industry 4.0 [3]. Virtualization of control functions by vAFs also simplifies flexible reconfiguration, as changes can be performed with less effort for software-based automation functions than for changing hardware components and cabling.

In this paper, we propose a reliable method for determining which vAF instance accesses the real, physical world. A cryptographically protected Physical-World Access Attestation (PWAA) issued by an IO module confirms which vAF instance accesses that IO module. The IO module itself provides the connectivity to the real, physical world via the connected sensors and actuators. This allows determining which vAFs are the critical instances that in fact monitor and control the real, physical world.

The remainder of the paper is structured as follows: Section II gives an overview on related work, and Section III on industrial security. Section IV describes the concept of physical world access attestations, and Section V presents a usage scenario in an industrial Operation Technology (OT) environment. Section VI provides an evaluation of the presented approach. Section VII concludes the paper and gives an outlook towards future work.

II. RELATED WORK

Cybersecurity for Industrial Automation and Control Systems (IACS) is specified in the standard series IEC62443 [4]. This series provides a security framework as a set of security standards defining security requirements for the development process and the operation of IACS as well as

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

technical cybersecurity requirements on automation systems and the used components. An overview on industrial security and IEC62443 is given in section III.

The Trusted Computing Group (TCG) defined attestation as the process of vouching for the accuracy of information [5]. An attestation is a cryptographically protected data structure that asserts the accuracy of the attested information. The Remote Attestation procedureS (RATS) working group of the Internet Engineering Task Force (IETF) described various attestation use cases [6]. Examples are the attestation of platform integrity and the attestation of the implementation approach for a cryptographic key store. An attestation allows a communication peer to reliably determine information about the (remote) platform besides the authenticated identity.

Virtualized automation functions have been described, e.g., by Gundall, Reti, and Schotten [7] investigating opportunities and challenges of hardware and operating system-level virtualization, and by Givehchi, Imtiaz, Trsek, and Jasperneite [8] presenting a performance evaluation of a cloud-based virtualized programmable logic controller.

III. INDUSTRIAL SECURITY

A CPS, e.g., an Industrial Automation and Control System (IACS), monitors and controls a technical system. Examples are process automation, machine control, energy automation, and cloud robotics. The impact of a vulnerability in the OT system may not only affect data and data processing as in classical Information Technology (IT), but it may have an effect also on the physical world. For example, production equipment could be damaged, or the physical process may operate outside the designed physical boundaries, so that the produced goods may not have the expected quality, or even safety-related requirements could be affected. Protecting IACSs against intentional attacks is increasingly demanded by operators to ensure a reliable operation, and also by regulation [9]. This section gives an overview on industrial security, and on the main relevant industrial security standard IEC 62443 [4] detailing security requirements for development, integration, and operation of IACS.

Cybersecurity mechanisms have been known for many years and are applied in smart devices (Internet of Things, Cyber Physical Systems, industrial and energy automation systems, operation technology). Such mechanisms target source authentication, system and communication integrity, and confidentiality of data in transit or at rest. Authentication, communication security, and authorization are also the basis for a Zero Trust (ZT) security approach. A ZT core principle is to assume that breaches may happen, and to verify explicitly security properties to improve the security posture before allowing access to resources and to avoid lateral threat movement. A ZT approach depends on security controls to assess, detect, and report attacks, and to act correspondingly. A resilience management function, as supported by the described monitoring functionality in this paper, can be used to keep an attacked CPS operational, and to recover quickly from attacks [10].

Industrial security is called also OT security, to distinguish it from general IT security. Industrial systems have not only different security requirements compared to general IT systems but come also with specific side conditions preventing the direct application of security concepts established in the IT domain in an OT environment. For example, availability and integrity of an automation system often have a higher priority than confidentiality. As an example, high availability requirements, different organization processes (e.g., yearly maintenance windows), and required component or system certifications may prevent the immediate installation of software or firmware updates.

The three basic security requirements in IT environments are confidentiality, integrity, and availability ("CIA" requirements). This CIA order corresponds to the classical priority of these basic security requirements. However, in OT systems, e.g., industrial automation systems or industrial IT, the priorities are often just the other way around: Availability of the IACS has typically the highest priority, followed by integrity. Confidentiality is often no strong requirement for control communications, but it may be needed to protect critical business know-how.

The international industrial security framework IEC 62443 [4] is a security requirements framework defined by the International Electrotechnical Commission (IEC). It addresses the need to design cybersecurity robustness and resilience into industrial automation and control systems, covering both organizational and technical aspects of security over the life cycle. Specific parts of this framework are applied successfully in different automation domains, including factory and process automation, railway automation, energy automation, and building automation. The standard specifies security for IACS along the lifecycle of industrial systems. Specifically addressed for the industrial domain is the setup of a security organization and the definition of security processes as part of an Information Security Management System (ISMS) based on already existing standards like ISO 27001 [11] or the NIST cyber security framework [12]. Furthermore, technical security requirements are specified distinguishing different security levels for industrial automation and control systems, and also for the used components. The standard has been created to address the specific requirements of IACS. Zones of an IACS having different security demands can be distinguished.

The parts of the IEC62443 standard are grouped into four clusters, covering:

– common definitions and metrics,
– requirements on setup of a security organization (ISMS related, similar to ISO 27001), as well as solution supplier and service provider processes,
– technical requirements and methodology for security on system-wide level, and
– requirements on the secure development lifecycle of system components, and security requirements to such components at a technical level.

The framework parts address different roles (actors) over different phases of the system lifecycle: The operator of an IACS operates the IACS that has been integrated by the system integrator, using components of product suppliers. In the set of corresponding documents, security requirements are

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

defined, which target the solution operator and the integrator but also the product manufacturer.

Part IEC62443-3-3 [13] defines technical security requirements for IACS, grouped into seven so-called foundational requirements. The foundational requirement FR6 "Timely Response to Events" defines security requirements for audit logs and continuous security monitoring. The requirements are specified in a way that they can be implemented in different ways. The approach described in this paper supports monitoring requirements for virtualized IACS components connected to the physical world.

IV. Physical World Access Attestation

A cryptographically protected PWAA is issued by an input/output (IO) module confirming in a reliable way that a certain vAF instance has in fact access to that IO module, i.e., that it has access to the physical world. This information can be used for monitoring the CPS operations as well as for adapting access permissions of the vAF. It can be reliably determined whether the intended vAFs have in fact access to the physical world. Furthermore, only those vAFs having the privilege of accessing the physical world can be granted access to perform security-critical operations during production, e.g., providing production data to a product database or executing control commands on attached actuators. Similarly, a corresponding attestation can be provided by virtual, simulated IO modules, confirming explicitly that this virtual IO module is *not* providing access to the real, physical world.

A. CPS System Model

Figure 1 shows an example of a CPS where multiple vAFs monitor and control the physical world via sensors (S) and actuators (A) connected to IO Modules (IOM). The vAFs exchange messages with IOMs over a data communication network (control network).

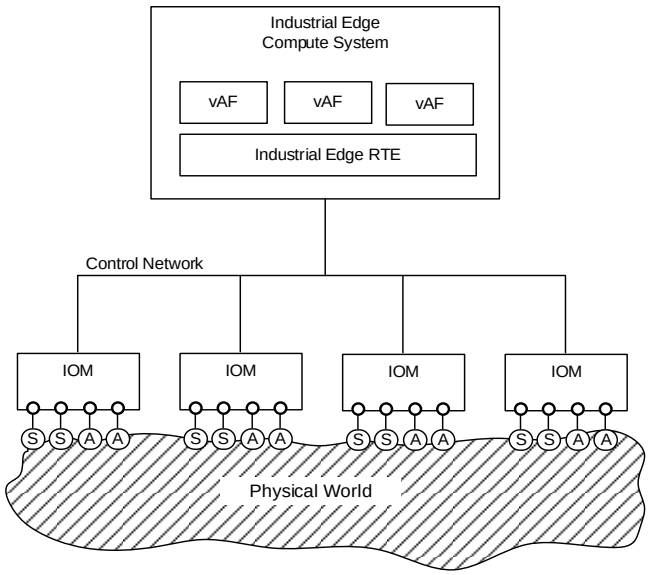


Figure 1. CPS system model

The vAFs are executed on an industrial edge compute system by an industrial edge RunTime Environment (RTE). It would also be possible that vAFs are executed on different edge compute systems or on a backend compute system (cloud-based control). A vAF can be realized, e.g., as container, as virtual machine, or also as native application executed by an operating system.

As depicted in Figure 1, an IOM is directly connected to physical sensors and actuators that in turn provide the interaction with the real, physical world. Thus, these IO modules are crucial as they control on one hand the actions to be performed in the physical world, but also provide monitoring data received from the physical world via the sensors.

B. Physical-World Access Attestation

An IOM authenticates the vAF that is accessing the IOM, e.g., by using a mutual certificate-based network authentication, e.g., Transport Layer Security (TLS) [14], Datagram TLS (DTLS) [15], QUIC [16], IKEv2 [17], or MAC security [18]. The IOM creates a cryptographically protected attestation, i.e., the PWAA, that confirms reliably which vAF is accessing this IOM, thereby confirming that the identified vAF has access to the sensors/actuators connected to the IOM, and thereby consequently having access to the physical world.

The PWAA confirms, based on the authenticated communication session between a vAF and the IOM, that the authenticated vAF has currently access to the physical world via this IOM. In addition, the PWAA may also provide additional information like information about the sensors and actuators connected to the IOM (e.g., identifier, type, calibration data), or about its location. The IOM may include a fixed, configurable location information, or may determine its location dynamically using a localization system.

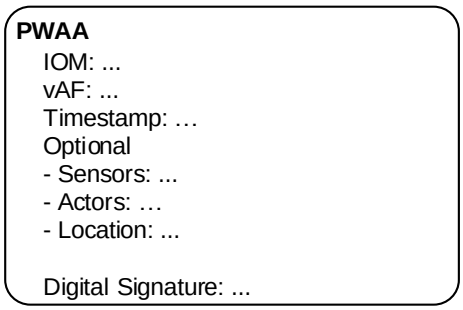


Figure 2. Physical world access attestation

Figure 2 visualizes the main elements of a PWAA. It indicates the IOM, the vAF, and it includes furthermore a timestamp to ensure freshness, and a digital signature of the IOM issuing the PWAA. The identification of the IOM and also the vAF may be done based on the credentials used for the mutual authentication between both. Optionally, the PWAA can comprise also an information on the sensors and actuators to which the indicated vAF has access, or on its location. The digital signature ensures that any manipulation of the PWAA can be detected. The PWAA can be encoded, e.g., as JSON Web Token (JWT) [19], as Concise Binary

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Object Representation (CBOR) [20], or as encoded Abstract Syntax Notation One (ASN.1) [21]. The digital signature can be realized using common cryptographic signature schemes, e.g., RSA signature, ECDSA, EdDSA [22], or a post-quantum safe digital signature scheme as CRYSTALS-Dilithium [23] or FALCON [24].

C. IO Module with Real-world Access Attestation

The PWAA is issued by an IOM depending on the authenticated entity that is accessing the IOM. Consequently, the IOM includes an attestation unit that determines the content to be attested depending on the authenticated vAF that is connected to the IOM, and that creates and provides the cryptographically protected PWAA.

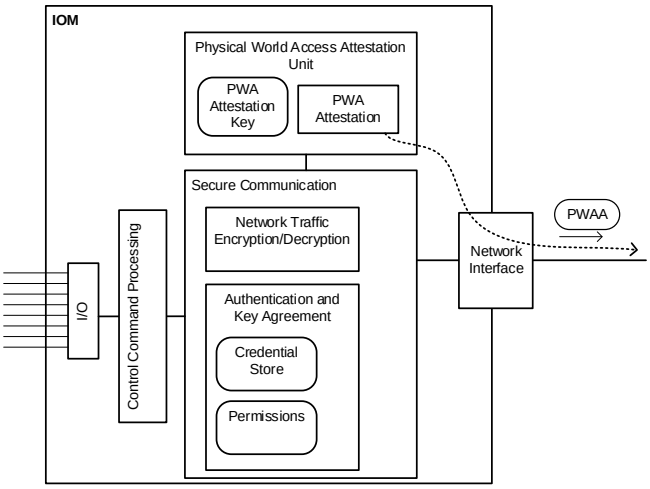


Figure 3. IO module with physical world access attestation

Figure 3 shows an IOM that includes an attestation unit that determines and provides the PWAA to a relying party, e.g., a CPS management system, a device management server, or a dedicated physical world access monitoring system. The IOM comprises an input-output interface (I/O) to which sensors and actuators can be connected. The IOM can be accessed via its network interface using a mutually authenticated secure communication session. The physical world access attestation unit determines which vAF has been authenticated by the IOM to establish a secure communication session, and builds a cryptographically protected PWAA. The digital signature of the PWAA may be build using the same credentials as used for mutual authentication or by distinct ones.

D. Adapting Access Permissions

The PWAA provided by an IOM is verified by a relying party, e.g., a production management system. Depending on the PWAA, it can adapt access control information related to the vAF that is indicated by the PWAA. Thereby, the PWAA can be seen as a context information that is used for access control decisions. This approach is related to a zero-trust security concept, where context information of both the requester and the responder is taken into account for making access control decisions.

E. Integrating with System Integrity Monitoring

The PWAAs provided by IOMs can also be used by a CPS integrity monitoring systems as described in [25]. It allows to determine reliably which vAF instances are the "real" ones that in fact have access to the physical world. Those vAFs are the ones that are subject to the operative CPS integrity monitoring. Other vAF instances may be used for simulations, tests, or as redundant backup functions.

Monitoring of PWAAs allows to detect if a vAF that is not intended to be used for operational control of real-world systems is connect to an IOM giving access to the physical world.

V. USAGE EXAMPLE

This section describes the usage of PWAA for CPS in an exemplary way. Figure 4 shows a CPS usage scenario comprising two control networks for two production networks (zone1, zone2) and a plant network. It can realize, e.g., a discrete production process on a manufacturing shop floor or a process automation system. The automation system is virtualized, i.e., it is realized by virtual automation functions (vAF) that are executed on an on-premise compute infrastructure (Industrial Edge Compute System) or in a backend computing infrastructure, e.g., a hyperscaler cloud or a multiaccess edge computing infrastructure of a mobile communication network. An industrial edge Run-Time Environment (RTE) executes the vAFs.

The vAFs interact with the real-world using sensors (S) and actuators (A) that are connected directly to IOMs. Sensors can measure, e.g., temperature, pressure, movement speed, power consumption, or detect physical objects. Actuators can cause a movement of a tool or the produced good, influence a motor speed, open or close a valve. The vAFs and the IOMs communicate via a communication network, e.g., Ethernet, WLAN, or using a 5G mobile shopfloor communication system.

In addition to the IOMs connected to the control network, also remote IO modules (rIOM) connected to the IOMs can be used. The IO modules (IOM, rIOM) provide a PWAA to a physical world access monitoring system. Optionally, also the RTEs executing the vAFs can provide attestations confirming to which IOMs a vAF is connected.

The physical world access monitoring system determines which vAFs have access to the physical world. Depending on the monitoring results, an authorization token, e.g., an OAUTH token [26], a verifiable credential [27], or an attribute certificate [28], can be provided to the vAF, or it can be granted the permission to perform a startup procedure of a technical system, e.g., a production machine. It is also possible to adapt access permissions of a vAF, e.g., to access a production management system or a Supervisory Control And Data Acquisition (SCADA) system.

Moreover, based on the context information contained in the PWAA, a pwAccess monitoring system as shown in Figure 4 can use this information to derive a system state based on specific sensor and actuator information. This system state can characterize if the system is operating in normal mode, in alert mode, or even in emergency mode,

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

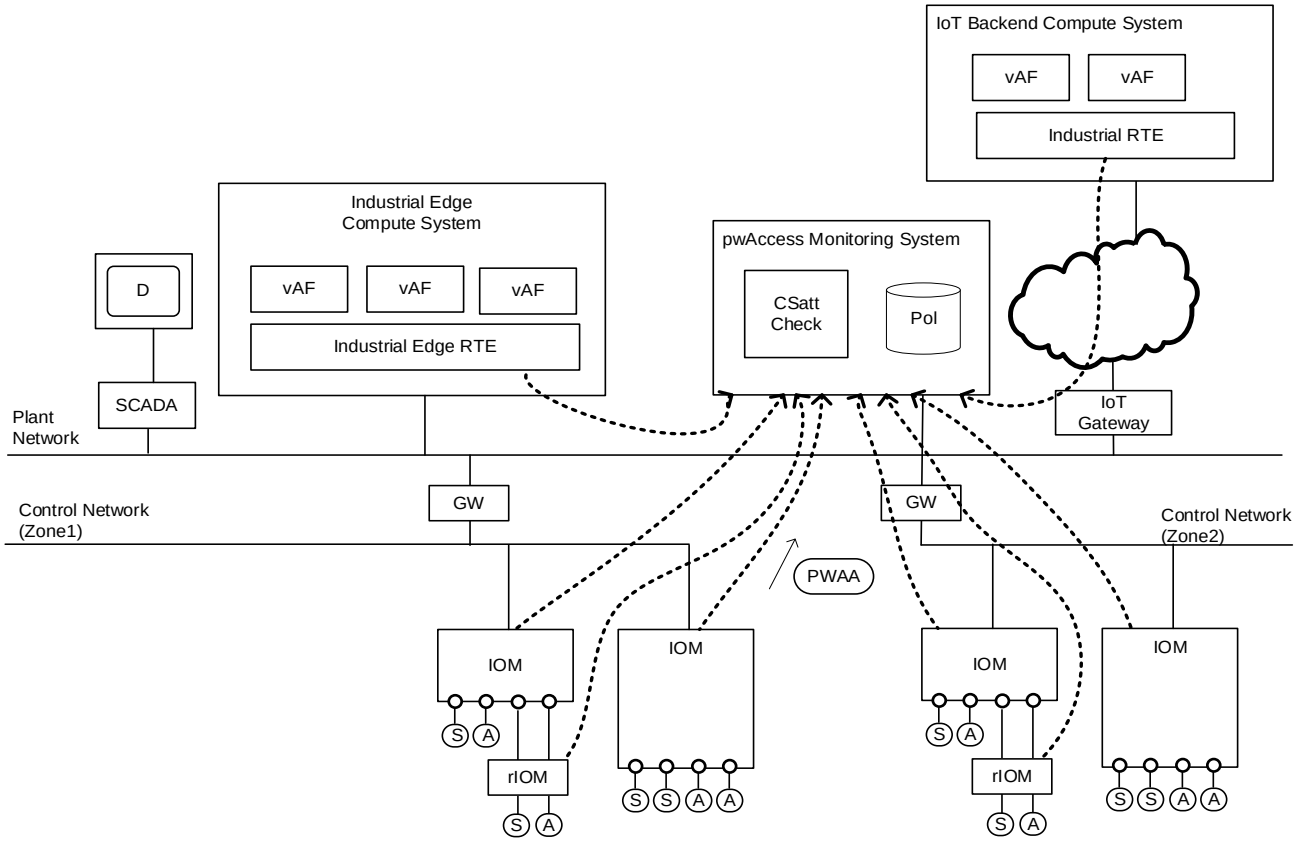


Figure 4. Example PWAA usage scenario

based on the evaluation of the actual measured values with potentially simulated and thus expected values. This derived system state in turn may influence further access decisions. This may be specifically important for systems in a critical infrastructure, like a power generation or distribution facility.

Here, it may be important to bind access decisions on the overall system state to ensure reliable operation of the system.

Furthermore, external provided system state information may also influence the access decision. An example may be the information about a maintenance period, to ensure that certain operation of a system is not possible during this time. Likewise, information about the operational environment can be considered, e.g., if a fire is detected in the production facilities.

The physical world access monitoring system is shown as dedicated component. However, it is also possible to realize it as virtualized function, e.g., as virtual machine or as container executed on an edge computing platform.

VI. EVALUATION

This section gives an evaluation of the presented PWAA concept from the perspective of the operator of a CPS, and from the perspective of the IOM implementation. Furthermore, performance impact and provisioning aspects are discussed.

Operator perspective: Availability and the flexibility to adapt to changing production requirements are important requirements for OT operators [29]. The proposed approach allows to apply strict cybersecurity controls automatically only when really needed, i.e., for operational real-world systems. The information may be utilized to report a system overall health state, which in turn can be considered in further access decisions. Other installations can be handled more openly, providing more flexibility.

Implementation perspective: The IOMs have to provide cryptographic attestations. This required support for basic cryptographic operations (cryptographic algorithms, key store, key management) is already available on IO modules that allow authenticated network access. So, only the additional functionality to create and provide attestations has to be implemented.

Performance perspective: The creation of an attestation is expected to have a negligible impact on the real-time performance of the IOM. For example, the signature can be generated during the authentication and key agreement phase of the secure communication protocol between IOM and vAF. Certain parts of the PWAA may also be prepared based on the locally available sensor information to require only minor lookup and completing of the information structure during the actual authentication and authorization phase.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Provisioning perspective: Additional key material has to be provisioned for protecting attestations, as the attestation key should be different to the device authentication key of IO module to have separate key material for different cybersecurity usages. Here, it may be assumed that for certificate management an automated interaction based on typical certificate management protocols like the Certificate Management Protocol CMP [30], Enrollment over Secure Transport EST [31], or the Simple Certificate Enrolment Protocol SCEP [32] is applied to overcome the burden of manual administration. In this context, a separate attestation key pair may be managed in addition to device authentication keys.

The risk reduction that can be achieved by the proposed PWAA can be evaluated using a Threat and Risk Analysis (TRA). A TRA is typically conducted at the beginning of the product design or system development, and updated after major design changes, or to address a changed threat landscape. In a TRA, possible attacks (threats) on the system are identified. The impact that would be caused by a successful attack and the probability that the attack happens are evaluated to determine the risk of the identified threats. The risk evaluation allows to prioritize the threats, focusing on the most relevant risks and to define corresponding security measures. Security measures can target to reduce the probability of an attack by preventing it, or by reducing the impact of a successful attack.

As long as the technology proposed in the paper has not been proven in a real-world operational setting, it can be evaluated conceptually by analyzing the impact that the additional security measure would have on the identified residual risks as determined by a TRA. However, TRAs for real-world CPS are typically not available publicly. Nevertheless, an illustrative example may be given by an automation system as depicted in Figure 4. It can be detected if a vAF that is not intended to be used for operational control is connected inadmissibly to an IOM that is connected to the real, physical world. Based on this detection, an alarm can be triggered to inform security administrators, or the connection could be blocked automatically by the IOM. Thereby, if a vAF that would be used rather for uncritical purposes as tests, simulation, or optimizations would by connected inadvertently or by ignorance to a real-world IOM where it would impact the real, physical word, could be detected in short time, so that corresponding countermeasures can take place.

Threat	Likelihood	Impact	Risk
Device communication intercepted	unlikely	moderate	minor
Device communication manipulated	unlikely	critical	moderate
Wrong vAF connected to physical-world IOM	likely	critical	major
⋮	⋮	⋮	⋮

Figure 5. Example Threats of a Threat and Risk Analysis

Figure 5 shows a simplified table as used typically in a TRA to collect and evaluate relevant threats to a technical system or component. Some selected threats are shown as example entries. Realistic TRAs for real-world systems and components usually contain a much longer list of threats. The likelihood and the impact of the threat is determined by judgement of competent personal, usually in a team including technical experts and people responsible for the product or system, and preferably also people involved in operation. To ensure consistency, typically criteria are defined that specify the conditions to assign a certain category. It has shown to be useful to define and document explicitly the criteria leading to the categorization of likelihood and impact, including also the made assumptions on the operational environment to ensure consistency and to allow for review.

The corresponding risk is determined based on the determined likelihood and impact, see Figure 6. The TRA with prioritized risks is the basis for security design decisions, focusing on the most critical risks. It is the basis to define a security concept that includes suitable protection measures. Protection measures may not be technical measures only, but include as well organizational and personal security measures (e.g., performing regularly security audits and security trainings). Likewise, for certain security measures that cannot be realized directly using installed system components, an operator may define compensating counter measures. An example is the introduction of additional security components for network traffic protection to avoid the replacement of a larger number of system components that are not capable of protecting exchanged data on their own. It is both possible that a security measure reduces the likelihood or the impact of relevant threats. The residual risk has to be accepted by management decision.

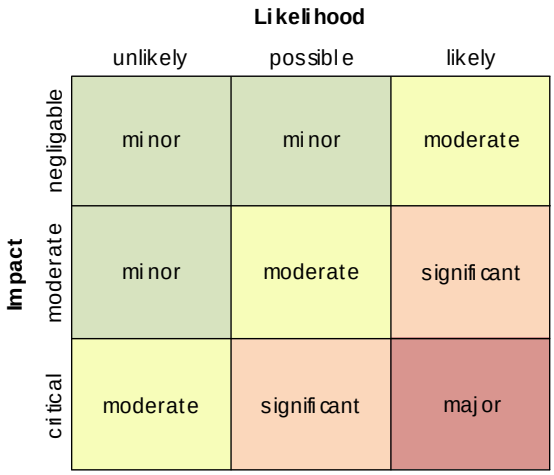


Figure 6. Risk Mapping

Figure 6 shows how the determined likelihood and the impact categories can be mapped to the corresponding risk value. In the example, the three categories unlikely, possible, and likely are used to describe the likelihood. For the impact, the three categories negligible, moderate, and critical are used. In practice, also more fine-granular rankings can be used, distinguishing, e.g., four or five different categories. Also, the risk evaluation can in general include further categories, e.g., disastrous.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

For the example threats shown in Figure 5, the risk that the device communication is intercepted is evaluated as minor, as the assumption in the example is that the device communication is protected cryptographically (e.g., by the Transport Layer Security protocol TLS [14] encrypting user data), and that the data would anyhow not reveal highly sensitive information. This results in a "minor" risk. In control communications within industrial automation systems, the confidentiality of control commands and of sensor measurements is often not very critical – but it may be different in specific operator environments when they would reveal sensitive operational parameters of the technical process.

The risk that the communication is manipulated, leading to a manipulated control operation, is unlikely as well, as the communication is assumed to be protected cryptographically in transit (e.g., by the Transport Layer Security protocol TLS [14] using mutual authentication with authenticated encryption, ensuring that user data cannot be manipulated without being detected). However, the impact is evaluated as critical, as, without any further protection, this threat could lead to arbitrary effects on the device operation and therefore also on the CPS. This results in a "moderate" risk.

The risk that a wrong, illegitimate vAF connects to a physical-world IOM is ranked here as major, as the assumption is that authorized operational personnel can flexibly setup and use vAFs, e.g., for simulation, optimization, as well as for control operations. Therefore, it is likely that unintentionally or carelessly a vAF that is not intended and released for operational control operations is connected to a real-world IOM. The impact would be critical as the correct, reliable control operation of the technical system could be affected.

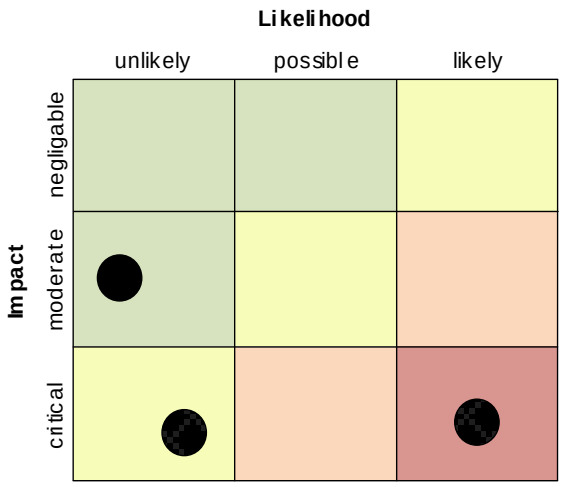


Figure 7. Risk Reporting for the Example Threats without Resilience-Under-Attack Protection

An overview on the determined risks can be shown in a graphical risk reporting as shown in Figure 7. It gives an easily understandable representation on the distribution of identified risks. This representation can be useful to depict the overall risk exposure of a CPS if many risks have been identified. In particular, the example shows the identified "major" risk (red field).

As the impact of the threat cannot easily be reduced in the assumed deployment scenario, the focus is to reduce the likelihood. Besides security training of operational personnel, a further approach to improve the identified "major" risk is to include in CPS integrity monitoring a detection function that identifies with short delay if a vAF that is not intended and released for operational control operations is connected to a real-world IOM. For this purpose, a positive list of vAFs that are approved for operational control of the technical system can be defined. The PWAAs of IOMs are collected regularly and analyzed to determine if a vAF that is not on the positive list is included in a PWAA. An alarm can be triggered to inform operational personnel and security responsibles about the security event and to trigger suitable reaction.

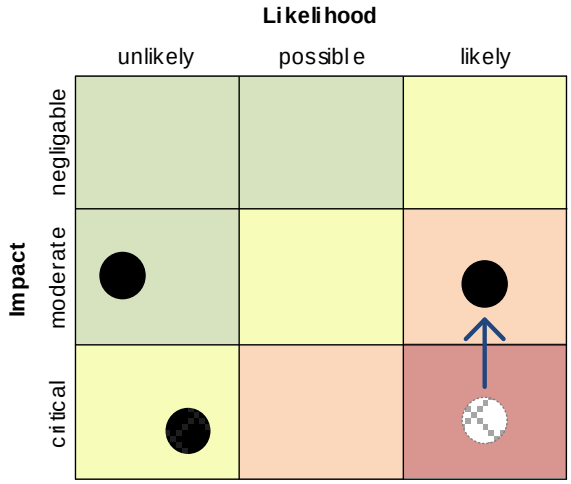


Figure 8. Risk Reporting for the Example Threats with Resilience-Under-Attack Protection

Monitoring PWAAs during CPS operation can reduce the impact of the identified threat, thereby improving the overall risk exposure. This effect is illustrated in Figure 8. In the example shown, the impact of the major risk reduces from critical to moderate, the risk is reduced correspondingly to moderate. Thereby, also the overall risk situation of the overall CPS is improved.

As the evaluation in a real-world CPS requires significant effort, and as attack scenarios cannot be tested that could really have a (severe) impact on the physical world, a simulation-based approach or using specific testbeds are possible approaches, allowing to simulate the effect on the physical world of certain attack scenarios with compromised components in a simulation model of the CPS, or to evaluate it in a protected testbed, e.g., a CPS test system. The simulation would have to include not only the IT-based control function, but also the physical world impact of an attack. Using physical-world simulation and test beds to evaluate the impact of attacks have been described by Urbina, Giraldo et al. [33]. They allow to analyze the impact of successful attacks on the physical world in a safe evaluation environment.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

VII. Conclusion and Future Work

The physical-world access attestation and the corresponding evaluation in a process monitoring system proposed in this paper allows to determine reliably which vAFs have in fact access to the real, physical world, i.e., to operational real-world technical systems. This information allows to apply stricter cybersecurity controls automatically specifically to those vAFs and their hosting platforms that are determined to be critical for the real-world CPS operation. It may also improve the operational reliability.

The exact implementation size and performance overhead of a technical realization has still to be evaluated. Cryptographic building blocks are needed to build a physical-world access attestation. As cryptographic building blocks available already within an IOM, e.g., for secure communication with vAFs, can be reused, the overhead in terms of implementation size is expected to be minor. As physical-world access attestations have to be created only rarely compared to protecting real-time control communications, also the overall performance overhead is estimated to be minor.

From a practical perspective, however, it is considered to be more important to determine the usefulness in practical use in operational automation systems, i.e., to what degree monitoring the physical-world access attestations allows to enhance flexibility in CPS planning and operation, and to increase operational efficiency by reducing the time needed for reconfiguring real-world technical systems while still being compliant with the required cybersecurity level. Furthermore, it can be investigated how the monitoring the physical-world access attestations can be best combined with monitoring further attestations, e.g., confirming the integrity of compute platforms and the runtime environment on which vAFs are executed.

References

[1] R. Falk and S. Fries, “Physical World Access Attestation”, CYBER 2023, The Eighth International Conference on Cyber-Technologies and Cyber-Systems, pp 8-11, 2023 [Online]. Available from: https://www.thinkmind.org/index.php?view=article&articleid=cyber_2023_1_20_80021 [retrieved: May, 2024]

[2] M. Gundall, D. Reti, and H. D. Schotten, “Application of Virtualization Technologies in Novel Industrial Automation: Catalyst or Show-Stopper?”, arXiv:2011.07804v1, Nov. 2020, [Online]. Available from: https://arxiv.org/abs/2011.07804 [retrieved: May, 2024]

[3] Plattform Industrie 4.0, “Industrie 4.0 Plug-and-produce for adaptable factories: example use case definition, models, and implementation”, Plattform Industrie 4.0 working paper, June 2017, [Online]. Available from: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Industrie-40-Plug-and-Produce.pdf [retrieved: May, 2024]

[4] IEC 62443, “Industrial Automation and Control System Security” (formerly ISA99), [Online]. Available from: http://isa99.isa.org/Documents/Forms/AllItems.aspx [retrieved: May, 2024]

[5] Trusted Computing Group, “Glossary”, 2012, [Online]. Available from https://trustedcomputinggroup.org/wp-content/uploads/TCG_Glossary_Board-Approved_12.13.2012.pdf [retrieved: May, 2024]

[6] H. Birkholz, D. Thaler, M. Richardson, N. Smith, and W. Pan, “Remote ATtestation procedureS (RATS) Architecture”, Internet Request for Comments RFC9334, 2023, [Online]. Available from: https://datatracker.ietf.org/doc/rfc9334/ [retrieved: May, 2024]

[7] M. Gundall, D. Reti, and H. D. Schotten, “Application of Virtualization Technologies in Novel Industrial Automation: Catalyst or Show-Stopper?”, arXiv:2011.07804v1, Nov. 2020, [Online]. Available from: https://arxiv.org/pdf/2011.07804 [retrieved: May, 2024]

[8] O. Givehchi, J. Imtiaz, H. Trsek, and J. Jasperneite, “Control-as-a-service from the cloud: A case study for using virtualized PLCs”, 10th IEEE Workshop on Factory Communication Systems (WFCS 2014), Toulouse, France, 2014, pp. 1-4. Available from: https://ieeexplore.ieee.org/document/6837587 [retrieved: May, 2024]

[9] EU Commission, “Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on horizontal cybersecurity requirements for products with digital elements and amending Regulation (EU) 2019/1020”, COM/2022/454 final, Sep., 2022, [Online]. Available from: https://digital-strategy.ec.europa.eu/en/library/cyber-resilience-act [retrieved: May, 2024]

[10] R. Falk and S. Fries, “Enhanced Attack Resilience within Cyber Physical Systems”, International Journal On Advances in Security, vol 16 no 1 & 2, pp. 1-11, June 2023, [Online]. Available from https://www.thinkmind.org/articles/sec_v16_n12_2023_1.pdf [retrieved: May, 2024]

[11] ISO/IEC 27001, “Information technology – security techniques – Information security management systems – requirements”, October 2013, [Online]. Available from: https://www.iso.org/standard/54534.html [retrieved: May, 2024]

[12] NIST, “Framework for Improving Critical Infrastructure Cybersecurity”, Version 1.1, National Institute of Standards and Technology, April, 2018, [Online]. Available from: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf [retrieved: May, 2024]

[13] IEC 62443-3-3:2013, “Industrial communication networks – Network and system security – Part 3-3: System security requirements and security levels”, Edition 1.0, August 2013. Available from: https://webstore.iec.ch/publication/7033 [retrieved: May, 2024]

[14] E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3”, Internet RFC8446, August 2018, [Online]. Available from: https://datatracker.ietf.org/doc/rfc8446 [retrieved: May, 2024]

[15] E. Rescorla, H. Tschofenig, and N. Modadugu, “The Datagram Transport Layer Security (DTLS) Protocol Version 1.3”, Internet RFC9147, April 2022, [Online]. Available from: https://datatracker.ietf.org/doc/rfc9147 [retrieved: May, 2024]

[16] M. Duke, “QUIC Version 2”, Internet RFC9369, December 2023, [Online]. Available from: https://datatracker.ietf.org/doc/rfc9369 [retrieved: May, 2024]

[17] C. Kaufmann, P. Hoffman, Y. Nir, and P. Eronen, “Internet Key Exchange Protocol Version 2 (IKEv2)”, Internet RFC7296, October 2014, [Online]. Available from: https://datatracker.ietf.org/doc/rfc7296 [retrieved: May, 2024]

[18] “IEEE Standard for Local and Metropolitan Area Networks--Port-Based Network Access Control”, IEEE Std 802.1X-2020, February 2020, [Online]. Available from: https://ieeexplore.ieee.org/document/9018454 [retrieved: May, 2024]

[19] M. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT)”, Internet RFC7519, May 2015, [Online]. Available from: https://datatracker.ietf.org/doc/rfc7519 [retrieved: May, 2024]

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[20] C. Bormann and P. Hoffman, "Concise Binary Object Representation (CBOR)", Internet RFC8949, December 2020, [Online]. Available from: https://datatracker.ietf.org/doc/rfc8949 [retrieved: May, 2024]

[21] "Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU X.690, February 2021, [Online]. Available from: https://www.itu.int/rec/T-REC-X.690 [retrieved: May, 2024]

[22] "Digital Signature Standard (DSS)", FIPS 186-5, February 2023, [Online]. Available from: https://doi.org/10.6028/NIST.FIPS.186-5 [retrieved: May, 2024]

[23] "CRYSTALS-Dilithium", [Online]. Available from: https://pq-crystals.org/dilithium/index.shtml [retrieved: May, 2024]

[24] "FALCON" [Online]. Available from: https://falcon-sign.info/ [retrieved: May, 2024]

[25] R. Falk and S. Fries, "Dynamic Trust Evaluation of Evolving Cyber Physical Systems", CYBER 2022, The Seventh International Conference on Cyber-Technologies and Cyber-Systems, pp. 19-24, 2022, [Online]. Available from: http://thinkmind.org/index.php?view=article&articleid=cyber_2022_1_30_80022 [retrieved: May, 2024]

[26] D. Hardt (Editor), " The OAuth 2.0 Authorization Framework", Internet Request for Comments RFC6749, 2012, [Online]. Available from: https://datatracker.ietf.org/doc/rfc6749/ [retrieved: May, 2024]

[27] D. Longley and M. Sporny, "Verifiable Credential Data Integrity 1.0 – Securing the Integrity of Verifiable Credential Data", W3C Working Draft 15 May 2023, [Online]. Available from: https://www.w3.org/TR/vc-data-integrity/ [retrieved: May, 2024]

[28] "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks", ITU-T X.509, October 2019, [Online]. Available from: https://www.itu.int/rec/T-REC-X.509-201910-I/en [retrieved: May, 2024]

[29] R. Falk and S. Fries, "System Integrity Monitoring for Industrial Cyber Physical Systems", Journal on Advances in Security, vol 11, no 1&2, July 2018, pp. 170-179, [Online]. Available from: www.iariajournals.org/security/sec_v11_n12_2018_paged.pdf [retrieved: May, 2024]

[30] C. Adams, S. Farrell, T. Kause, and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", Internet Request for Comments RFC4210, 2005, [Online]. Available from: https://datatracker.ietf.org/doc/rfc4210 [retrieved: May, 2024]

[31] M. Pritikin, P. Yee, and D. Harkins, "Enrollment over Secure Transport", Internet Request for Comments RFC7030, 2013, [Online]. Available from: https://datatracker.ietf.org/doc/rfc7030 [retrieved: May, 2024]

[32] P. Gutmann, "Simple Certificate Enrolment Protocol", Internet Request for Comments RFC8894, 2020, [Online]. Available from: https://datatracker.ietf.org/doc/rfc8894 [retrieved: May, 2024]

[33] D. Urbina, J. Giraldo, N. O. Tippenhauer, and A. Cardenas, "Attacking fieldbus communications in ICS: applications to the SWaT testbed", Singapore Cyber-Security Conference (SG-CRC), IOS press, pp. 75–89, 2016, [Online]. Available from: http://ebooks.iospress.nl/volumearticle/42054 [retrieved: May, 2024]

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Can Secure Software be Developed in Rust? On Vulnerabilities and Secure Coding Guidelines

Tiago Espinha Gasiba
T CST SEL-DE
Siemens AG
Munich, Germany
tiago.gasiba@siemens.com

Sathwik Amburi
T CST SEL-DE, Siemens AG
Technical University of Munich
Munich, Germany
sathwik.amburi@{siemens.com, tum.de}

Andrei-Cristian Iosif
T CST SEL-DE
Siemens AG
Munich, Germany
andrei-cristian.iosif@siemens.com

***Abstract*—Since the Rust programming language was accepted into the Linux Kernel, it has gained significant attention from the software developer community and the industry. Rust has been developed to address many traditional software problems, such as memory safety and concurrency. Consequently, software written in Rust is expected to have fewer vulnerabilities and be more secure. However, a systematic analysis of the security of software developed in Rust is still missing. The present work aims to close this gap by analyzing how Rust deals with typical software vulnerabilities. We compare Rust to C, C++, and Java, three widely used programming languages in the industry, regarding potential software vulnerabilities. We also highlight ten common security pitfalls in Rust programming that we think software developers and stakeholders alike should be wary of. Our results are based on a literature review and interviews with industrial cybersecurity experts. We conclude that, while Rust improves the status quo compared to the other programming languages, writing vulnerable software in Rust is still possible. Our research contributes to academia by enhancing the existing knowledge of software vulnerabilities. Furthermore, industrial practitioners can benefit from this study when evaluating the use of different programming languages in their projects.**

Index Terms*—*Cybersecurity*; *Software development*; *Industry*; *Software*; *Vulnerabilities*; *Rust Programming Language.

I. INTRODUCTION

Rust, a systems programming language that originated in 2010, has significantly increased in popularity over the past decade. Rust distinguishes itself from other programming languages through several key features. Firstly, it ensures memory safety without needing a garbage collector, utilizing an ownership system with rules about borrowing and lifetimes. This feature helps prevent common bugs such as null pointer dereferencing and dangling pointers prevalent in languages like C and C++. Secondly, Rust's concurrency model is designed to make concurrent programming safer and more straightforward, with the ownership and type systems enforcing thread safety and preventing data races at compile time. Additionally, Rust provides performance comparable to C and C++ due to its low-level control over system resources and zero-cost abstractions. Unlike higher-level languages such as Java, which rely on a virtual machine, Rust compiles directly to machine code, offering predictable performance and minimal runtime overhead. Rust's expressive type system supports advanced features like algebraic data types, pattern matching, and traits (Rust's version of interfaces), enabling developers to write robust and maintainable code. Moreover, Rust's tooling and ecosystem, including the Cargo package manager, facilitate dependency management and project building, making the development process smoother and more efficient. The Rust community is also known for its welcoming nature and comprehensive documentation, providing extensive learning materials for developers.

This paper extends the authors' previous work presented at the CYBER 2023 [1], which covered the initial findings and methodologies. The current study includes some additional insights into the Rust programming language and common security pitfalls.

According to a market overview survey by Yalantis [2], which conducted more than 9,300 interviews, 89% of developers prefer Rust over other widespread programming languages like C and C++ due to its robust security properties. Despite its steep learning curve, industry professionals argue that the time invested in learning Rust yields added benefits and fosters better programming skills [3]. Stack Overflow notes that developers appreciate Rust's focus on system-level details, which helps prevent null and dangling pointers, and its memory safety without the need for a garbage collector [4]. These factors contribute to its growing adoption in the industry. This sentiment is echoed by the industry's push toward adopting the Rust programming language. Furthermore, according to Stack Overflow Developer Surveys, Rust has been the most loved and admired language since 2016. In the most recent Stack Overflow 2023 Developer Survey [5], Rust secured the position of the most admired language, with over 80% of the 87,510 responses favoring it.

Due to its focus on memory safety and concurrency, Rust has become the language of choice for many tools developed for Linux, FreeBSD, and other operating systems. Notably, Rust's adoption in Linux Kernel development [6], [7] underscores its growing significance in an industrial context, including space systems [8]. Rust meets the critical requirements for such applications in several key aspects. First, its design enables code to operate close to the kernel, facilitating tight interactions with both software and hardware. Second, it supports determinism, ensuring consistent timing of outputs. Third, Rust does not use a garbage collector, which is crucial

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

for manual memory management and reclaiming memory without automatic interference. These characteristics ensure reliable and predictable outcomes, vital for space systems [8] and other critical industrial systems. Despite these advantages, Rust's proven effectiveness in space systems remains to be fully demonstrated. Developing toolchains, workforce education, and case studies in field applications are vital to validating the utility of memory-safe languages in such demanding environments [8].

Major platforms, such as Google, have started including Rust in their systems, including Android [9], demonstrating its broad applicability. Additionally, forums like RustSec [10] offer real-time updates and insights into the current state of Rust security, reflecting its critical role in secure computing. Recognizing these advantages, governments, including the US, have begun recommending memory-safe languages like Rust [11], further emphasizing its importance in terrestrial and extraterrestrial computing environments.

Rust promotes itself as being safer than traditional languages, such as C and C++, which are widely used in an industrial context, by borrowing many aspects from functional languages like Haskell. However, in the realm of industry, particularly in critical infrastructures, safety is not synonymous with security. As the industry is obliged to follow secure development standards, such as IEC 62443 [12], [13], the notion of safety in Rust must be understood not only from a memory management perspective but also from a security standpoint [14]. Rust was developed to address memory-related vulnerabilities, which constitute only 19.5% of the most exploited vulnerabilities in 2023, according to The Cybersecurity & Infrastructure Security Agency (CISA) [15]. Exploits related to routing and path abuse tied for second place with memory vulnerabilities, followed by default secrets (4.9%), request smuggling (4.9%), and weak encryption (2.4%). The most prevalent exploit was insecure exposed functions (IEF), accounting for 48.8% of incidents. This has led to the saying, "Rust won't save us, but its ideas will," emphasizing the importance of adopting Rust's safety and security principles beyond its direct application [16].

Developing industrial products and services follows strict guidelines, especially for those products and services aimed at critical infrastructures. In these cases, cybersecurity incidents can severely negatively impact companies and society in general. Therefore, the security of industrial products must be tightly controlled. Consequently, Rust is considered a good candidate for industrial software development.

While Rust has been celebrated for its safety features [17], [18], less research has been conducted on its security aspects. This lack of research is primarily because this programming language is still relatively young compared to longstanding players in the industry, such as C, C++, and Java. Furthermore, developers and users often conflate safety with security, potentially leading to software vulnerabilities. Therefore, this paper aims to *understand to what extent vulnerable software can be written in Rust*. We approach this topic in two ways:

1) Evaluating the difficulty of writing vulnerable software based on industry-recognized security standards like the SysAdmin, Audit, Network, Security (SANS) Institute TOP 25 [19], the Open Web Application Security Project (OWASP) TOP 10 [20], and the 19 Deadly Sins [21], and
2) Identifying ten common pitfalls in Rust that we feel developers should be aware of.

This study's contributions are as follows: firstly, through the present work, the authors aim to raise awareness, as defined by Gasiba et al. [22], about Rust security and its pitfalls within the industry (for both industrial practitioners and academia); secondly, our work provides expert opinions from industry security experts on how to mitigate such issues when developing software with Rust; furthermore, our work contributes to academic research and the body of knowledge on Rust security by adding new insights and fostering a deeper understanding of Rust security; finally, our work serves as motivation for further studies in this area.

The rest of this paper is organized as follows: Section II discusses previous work that is either related to or served as inspiration for our study. Section III briefly discusses the methodology followed in this work to address the research questions. In Section IV, we provide a summary of our results, and in Section V, we conduct a critical discussion of these results. Finally, in Section VI, we conclude our work and outline future research.

II. RELATED WORK

A significant contribution to understanding Rust's security model comes from Sible et al. [23]. Their work offers a thorough analysis of Rust's security model, focusing on its memory and concurrency safety features. However, they also highlight Rust's limitations, such as handling memory leaks. While Rust offers robust protections, the authors emphasize that these protections represent only a subset of the broader software security requirements. Their insights are invaluable for understanding both the strengths and limitations of Rust's security model. Wassermann et al. [24] presented a detailed exploration of Rust's security features and potential vulnerabilities. They highlighted issues when design assumptions do not align with real-world data. The authors stress the importance of understanding vulnerabilities from the perspective of Rust program users. They advocate for tools that can analyze these vulnerabilities, even without access to the source code. Discussions also touched upon the maturity of the Rust software ecosystem and its potential impact on future security responses. They suggest that the Rust community could benefit from the Rust Foundation either acting as or establishing a related CVE Numbering Authority (CNA). Their study further enriches the understanding of Rust's security model.

Qin et al. [25] conducted a comprehensive study revealing that unsafe code is widely used in the Rust software they examined. This usage is often motivated by performance optimization and code reuse. They observed that while developers aim to minimize the use of unsafe code, all memory-safety bugs involve it. Most of these bugs also involve safe code,

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

suggesting that errors can arise when safe code does not account for the implications of associated unsafe code. The researchers identified Rust's 'lifetime' concept, especially when combined with unsafe code, as a frequent source of confusion. This misunderstanding often leads to memory-safety issues. Their findings underscore the importance of fully grasping and correctly implementing Rust's safety mechanisms.

Zheng et al. [26] surveyed the Rust ecosystem for security risks by analyzing a dataset of 433 vulnerabilities, 300 vulnerable code repositories, and 218 vulnerability fix commits over 7 years. They investigated the characteristics of vulnerabilities, vulnerable packages, and the methods used for vulnerability fixes. Key findings reveal that memory safety and concurrency issues constitute two-thirds of the vulnerabilities, with a notable delay (averaging over two years) before vulnerabilities are publicly disclosed. Additionally, they observed an increasing trend in package-level security risks over time despite a decrease in code-level risks since August 2020. Moreover, popular Rust packages tend to have a higher number of vulnerabilities, and vulnerability fixes often involve localized changes. This study contributes to the understanding of security risks in the Rust ecosystem by providing a dataset for future research, summarizing patterns in vulnerability fixes, and discussing implications for securing Rust packages.

Balasubramanian et al.'s research [27] delves deeper into Rust's security aspects by leveraging its linear type system for enhanced safety in system programming. They demonstrate how Rust's safety mechanisms, which incur no runtime cost and eschew garbage collection, are applied to bolster software fault isolation, enforce static information flow control, and facilitate automatic checkpointing. These areas, crucial for security, are shown to benefit from Rust's design, which simplifies the implementation of complex security features. The paper underscores Rust's potential to significantly impact system programming by making high-level security features more attainable without compromising performance. The discussion also acknowledges the learning curve and paradigm shift required to fully utilize Rust's advantages, positioning these as necessary investments for achieving superior safety and security in system programming applications.

A. Security Standards and Guidelines

Various security standards and guidelines can be applied to Rust programming. The International Electrotechnical Commission Technical Report (IEC TR) 24772 [28] standard, "Secure Coding Guidelines Language Independent," provides guidelines suitable for multiple programming languages, including Rust. ISO/IEC 62443 [12], especially sections 4-1 and 4-2, sets the industry standard for secure software development [13]. The Common Weakness Enumeration (CWE) by MITRE [29] offers a unified set of software weaknesses.

The French Government's National Agency for the Security of Information Systems (ANSSI) has published a guide titled "Programming Rules to Develop Secure Applications with Rust" [30], which is a valuable resource for developers.

B. Security Documentation and Tools

Rust's safety guarantees and performance have led to its growing adoption across various domains. Notably, Google has integrated Rust into the Android Open Source Project (AOSP) to mitigate memory safety bugs, a significant source of Android's security vulnerabilities [9]. Updates and discussions about Rust security are frequently shared on blogs, forums, and other platforms.

Several Static Application Security Testing (SAST) tools are available for Rust, such as those listed on the Analysis Tools platform [31]. These tools play a crucial role in the secure software development lifecycle.

Community-driven initiatives like RustSec [10] offer advisories on vulnerabilities in Rust crates (A crate is the smallest amount of code that the Rust compiler considers at a time). Real-time updates from RustSec and other platforms are invaluable for developers to stay updated on potential security issues in Rust packages.

C. Secure Coding Guidelines

The paper "Secure Coding Guidelines - (un)decidability" by Bagnara et al. [32] delves into the challenges of secure coding. It mainly focuses on the undecidability of specific rules, such as "Improper Input Validation". The authors argue that determining adherence to specific secure coding guidelines can be complex due to factors like context.

D. Secure Code Awareness

Secure code awareness is crucial, especially in critical infrastructures. A study by Gasiba et al. [33] explored the factors influencing developers' adherence to secure coding guidelines. While developers showed intent to follow these guidelines, there was a noticeable gap in their practical knowledge. This highlights the need for targeted, secure coding awareness campaigns. The authors introduced a game, the CyberSecurity Challenges, inspired by the Capture The Flag (CTF) genre, as an effective method to raise awareness.

The Sifu platform [34] was developed in line with these challenges. The platform promotes secure coding awareness among developers by combining serious gaming techniques with cybersecurity and secure coding guidelines. It also uses artificial intelligence to offer solution-guiding hints. Sifu's successful deployment in industrial settings showcases its efficacy in enhancing secure coding awareness.

III. METHODOLOGY

Our research methodology, aimed at examining the security in the Rust programming language compared to Java, C, and C++, and its interaction with security assessment tools, was composed as shown in Figure 1.

A. Literature Exploration

Due to the scarcity of academic resources, we commenced with an integrated literature review, primarily focusing on gray literature, such as reports and blog posts. We also conducted an academic literature review using the ACM, IEEE Xplore,

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

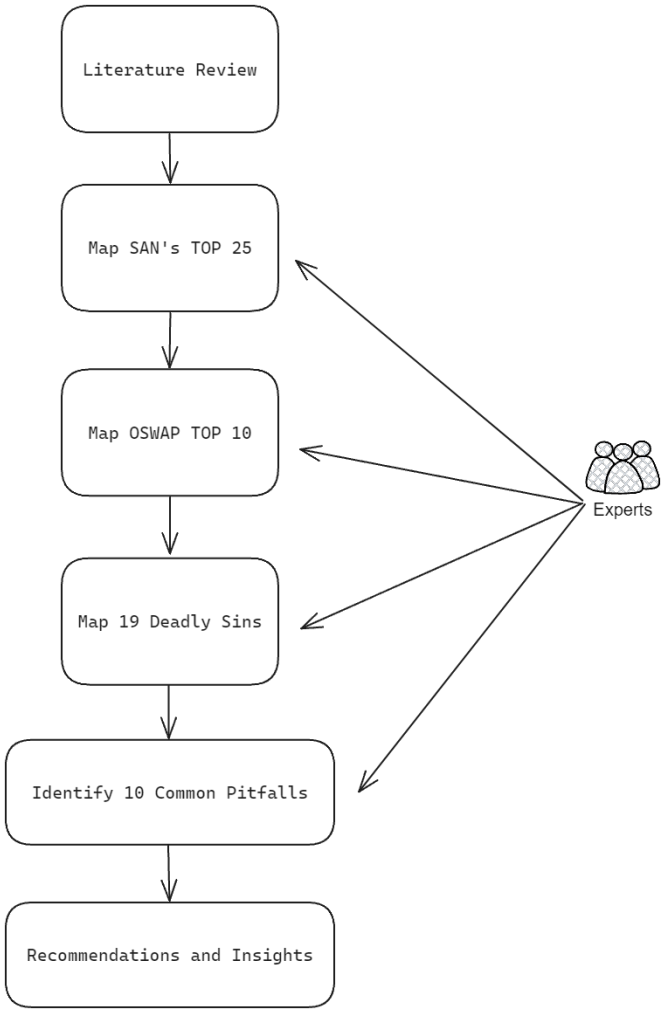


Fig. 1. Methodology

and Google Scholar databases, with search terms including ”Rust Security”, ”Java Security”, ”C Security”, and ”Static Application Security Testing”. The time frame was set from 2010 to early 2024.

B. Interviews with Security Experts

We held discussions with five industry security experts with experience with Rust, Java, C, and security assessment tools. The experts from the industry are consultants with more than ten years of experience and work on the topic of secure software development. Their insights contributed significantly to our understanding and interpretation of the literature. Additionally, we conducted informal surveys with two students who regularly use Rust and contribute to open-source projects developed in the same programming language. The student's background is a master's in computer science with five years of programming experience with Rust. The informal interviews with industry experts and computer science students commenced in August 2023 and lasted about thirty minutes.

C. Mapping to CWE/SANS, OWASP, and 19 Deadly Sins

In this phase, we categorized Rust security issues according to the Common Weakness Enumeration (CWE), SANS Top 25, and OWASP 10 and 19 deadly sins. This step helped in classifying and understanding the security threats relevant to Rust.

D. Analysis with Rust/SAST Tools

A comparative study was undertaken with Rust and Static Application Security Testing (SAST) tools to assess the effectiveness of these tools in identifying Rust's security vulnerabilities.

E. Identifying 10 Common Pitfalls

After our mapping and analysis, we have identified 10 common security pitfalls that developers and stakeholders should be aware of. These pitfalls are not exhaustive but serve as a starting point for stakeholders to consider security as an application-specific issue rather than merely a language-specific concern.

F. Definitions

In our research, we employed three categories to assess the level of security protection against specific issues in Rust: Rare and Difficult (RD), Safeguarded (SG), and Unprotected (UP).

- **Rare and Difficult (RD)**: This category refers to security issues Rust's inbuilt features or mechanisms can fully mitigate or prevent. The language itself provides robust protection against such issues. Security vulnerabilities falling into this category are rare and difficult to spot. They occur infrequently, making it challenging to encounter them. Rust's inherent protections are usually effective in addressing these issues, **unless unsafe blocks are used**. These issues are often not commonly observed and may require specific circumstances or careful analysis, often associated with a Common Vulnerabilities and Exposures (CVE) identifier.
- **Safeguarded (SG)**: Issues falling under this category benefit from protective measures provided by Rust. The programming language offers safeguards to mitigate these issues, reducing their likelihood or impact. However, additional precautions or interventions may be necessary in specific scenarios.
- **Unprotected (UP)**: This category encompasses security issues that the language does not inherently guard against or if the CWE does not apply to the language. The language lacks built-in mechanisms to protect against these issues. Addressing them requires utilizing external libraries or tools or a comprehensive understanding of the language and underlying systems. In cases where a particular CWE is irrelevant to the language, it is also categorized as UP.

We utilized this methodology to evaluate the SANS Top 25, OWASP Top 10, and 19 Deadly Sins of Software Security within the context of Rust. Additionally, we created Proof-of-Concept (PoC) Rust code [35] to validate its feasibility,

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

containing vulnerabilities for the following weaknesses: Command Injection, Integer Overflow, Resource Leakage, SQL Injection, and Time-of-Check-Time-of-Use (TOCTOU).

IV. RESULTS

A. SANS 25 (2022)

This section presents the findings of our analysis concerning vulnerabilities in Rust, with a particular focus on evaluating vulnerable software based on the SANS Top 25 list. Table I summarizes the protection levels for different CWE vulnerabilities in Rust. These are categorized into three groups: Rare and Difficult (RD), Safeguarded (SG), and Unprotected (UP). It is crucial to note that complete protection is extended to all code that does not use 'unsafe' blocks.

Among the analyzed CWE vulnerabilities, the following are identified as having Full Protection in Rust: CWE-787, CWE-125, CWE-416, CWE-476, CWE-362, and CWE-119. This finding suggests that Rust offers robust protection against these vulnerabilities, thereby minimizing the likelihood of their occurrence in Rust-based software, provided the code does not employ 'unsafe' blocks.

Conversely, several vulnerabilities, including CWE-79, CWE-22, CWE-352, CWE-434, CWE-502, CWE-287, CWE-798, CWE-862, CWE-306, CWE-276, CWE-918, and CWE-611, exhibit No Protection in Rust. This finding implies that Rust lacks built-in mechanisms to prevent or mitigate these vulnerabilities, even when 'unsafe' blocks are not in use. It is vital for developers working with Rust to be cognizant of these vulnerabilities and implement additional security measures to counteract them.

For certain vulnerabilities, such as CWE-79, CWE-20, CWE-78, CWE-190, CWE-77, CWE-400, and CWE-94, Rust provides some protection and safeguards. This result indicates that Rust incorporates certain features or constructs that can help diminish the likelihood of these vulnerabilities. However, additional precautions may still be necessary to mitigate the associated risks fully.

These findings underscore the importance of understanding the vulnerabilities inherent in Rust and implementing suitable security measures. While Rust provides strong protection against specific CWE vulnerabilities, there are areas where additional precautions are necessary. Developers should exercise caution when dealing with vulnerabilities categorized as UnProtected, as these require meticulous attention and specialized security practices.

In addition to analyzing the vulnerabilities in Rust, it is insightful to contrast the protection levels Rust offers with those provided by other prominent programming languages, such as C, C++, and Java. Table II facilitates a side-by-side comparison across these languages. In this table, the protection levels are denoted as follows: Rare and Difficult (RD), Safeguarded (SG), and Unprotected (UP) for C, C++, and Java.

Upon examining Table II, it is evident that C, being an older language, demonstrates fewer protections compared to C++ and Java, especially regarding memory-related vulnerabilities like CWE-787. For instance, C does not provide safeguards for CWE-787 [36], while C++ and Java offer robust protections.

Java, owing to its managed memory model and sandboxed execution environment, shows strong defenses against some vulnerabilities that are particularly problematic in C and C++, such as CWE-416.

Interestingly, for some vulnerabilities like CWE-79 and CWE-22, all three languages - C, C++, and Java - display limited or no protection. This observation accentuates the importance of following secure coding practices irrespective of the language used.

Furthermore, C++ seems to find a middle ground between C and Java regarding protection levels, which could be attributed to its evolution from C and its incorporation of modern language features.

Developers must be cognizant of these variations in protection levels across languages and carefully weigh the security aspects alongside other factors, such as performance and ecosystem, when choosing a language for their projects.

TABLE I
SANS TOP 25 CWE VS. PROTECTION LEVELS IN RUST

CWE ID	Short Description	RD	SG	UP
CWE-787	Out-of-bounds Write	•		
CWE-79	Cross-site Scripting			•
CWE-89	SQL Injection		•	
CWE-20	Improper Input Validation		•	
CWE-125	Out-of-bounds Read	•		
CWE-78	OS Command Injection		•	
CWE-416	Use After Free	•		
CWE-22	Path Traversal			•
CWE-352	Cross-Site Request Forgery			•
CWE-434	Unrestricted Dangerous File Upload			•
CWE-476	NULL Pointer Dereference	•		
CWE-502	Deserialization of Untrusted Data			•
CWE-190	Integer Overflow or Wraparound		•	
CWE-287	Improper Authentication			•
CWE-798	Use of Hard-coded Credentials			•
CWE-862	Missing Authorization			•
CWE-77	Command Injection		•	
CWE-306	Missing Critical Function Authentication			•
CWE-119	Buffer Overflow	•		
CWE-276	Incorrect Default Permissions			•
CWE-918	Server-Side Request Forgery			•
CWE-362	Race Condition	•		
CWE-400	Uncontrolled Resource Consumption		•	
CWE-611	Improper Restriction of XXE			•
CWE-94	Code Injection		•	
		24%	28%	48%

B. OWASP 10

The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about web applications' most critical security risks. The following is an assessment (summarized in Table III) of how the Rust language can offer protection against these vulnerabilities, according to the OWASP standard from 2021:

- **A01-Broken Access Control (SG):** While Rust does not inherently provide web application access control, its strong type system and ownership model can help prevent logical errors that might lead to such vulnerabilities.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II
SANS TOP 25 CWE VS. PROTECTION LEVELS IN C, C++, AND JAVA

CWE	C			C++			Java		
	RD	SG	UP	RD	SG	UP	RD	SG	UP
CWE-787			•		•		•		
CWE-79			•			•			•
CWE-89			•		•			•	
CWE-20			•			•		•	
CWE-125			•			•	•		
CWE-78			•			•		•	
CWE-416			•		•		•		
CWE-22			•			•			•
CWE-352			•			•			•
CWE-434			•			•			•
CWE-476			•		•		•		
CWE-502			•			•			•
CWE-190			•			•			•
CWE-287			•			•			•
CWE-798			•			•			•
CWE-862			•			•			•
CWE-77			•			•		•	
CWE-306			•			•			•
CWE-119			•		•		•		
CWE-276			•			•			•
CWE-918			•			•			•
CWE-362			•			•		•	
CWE-400			•		•			•	
CWE-611			•			•			•
CWE-94			•			•		•	
	0%	0%	100%	0%	24%	76%	20%	28%	52%

- **A02-Cryptographic Failures (SG):** Although Rust does not provide built-in cryptographic features, it has high-quality cryptographic libraries that can help mitigate these failures to some extent.
- **A03-Injection (SG):** Rust's strong type system and approach to handling strings can help prevent injection attacks. However, poor programming practices may still result in these attacks; see PoC code in [35].
- **A04-Insecure Design (UP):** This vulnerability is more about the design of the application rather than the language itself. While Rust offers memory safety [37], it does not inherently protect against insecure design, which encompasses many issues.
- **A05-Security Misconfiguration (UP):** This vulnerability is more about the application and environment configuration than the language itself.
- **A06-Vulnerable and Outdated Components (SG):** Rust's package manager, Cargo, and its ecosystem can help manage dependencies and their updates.
- **A07-Identification and Authentication Failures (UP):** Rust does not inherently provide user authentication and session management features.
- **A08-Software and Data Integrity Failures (UP):** Rust's ownership model and type system can help ensure data integrity, but it is up to the programmer to leverage these features effectively.
- **A09-Security Logging and Monitoring Failures (UP):** This vulnerability is more about the application's logging and monitoring capabilities than the language itself.
- **A10-Server-Side Request Forgery (SSRF) (UP):** Rust does not inherently protect against SSRF attacks. Programmers must validate and sanitize all URLs and restrict the server's ability to interact only with whitelisted endpoints [38].

We note that in literature, the numbering of the OWASP vulnerabilities can also appear together with the date of the OWASP standard, e.g., A01:2021.

TABLE III
MAPPING OF OWASP TOP 10 FROM 2021 TO RUST PROTECTION LEVELS

OWASP Vulnerability	RD	SG	UP
A01-Broken Access Control		•	
A02-Cryptographic Failures		•	
A03-Injection		•	
A04-Insecure Design			•
A05-Security Misconfiguration			•
A06-Vulnerable and Outdated Components		•	
A07-Identification and Authentication Failures			•
A08-Software and Data Integrity Failures		•	
A09-Security Logging and Monitoring Failures			•
A10-Server-Side Request Forgery			•
	0%	50%	50%

C. 19 Deadly Sins of Software Security

The book "19 Deadly Sins of Software Security: Programming Flaws and How to Fix Them" identifies and guides how to fix 19 common security flaws in software programming (see Table IV for a summary). Rust, a programming language, is designed to prevent some of the most common security vulnerabilities. Below is a brief analysis of how Rust addresses the 19 sins:

TABLE IV
MAPPING OF NINETEEN DEADLY SINS OF SOFTWARE SECURITY TO RUST PROTECTION LEVELS

Security Flaw	RD	SG	UP
Buffer Overflows	•		
Format String Problems		•	
Integer Overflows		•	
SQL Injection		•	
Command Injection		•	
Cross-Site Scripting (XSS)			•
Race Conditions	•		
Error Handling	•		
Poor Logging		•	
Insecure Configuration			•
Weak Cryptography		•	
Weak Random Numbers	•		
Using Known Vulnerable Components		•	
Unvalidated Redirects and Forwards			•
Injection		•	
Insecure Storage			•
Denial of Service		•	
Insecure Third-Party Interfaces			•
Cross-Site Request Forgery (CSRF)			•
	21%	47%	32%

- **Buffer Overflows (RD)**: Rust has built-in protection against buffer overflow errors. It enforces strict bounds

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

checking, preventing programs from accessing memory they should not.

- **Format String Problems (SG)**: Rust does not support format strings in the same way as languages like C, thereby reducing the risk of this issue. It provides strong protection against format string problems through its type-safe formatting mechanism. The std::fmt module in Rust offers a rich set of formatting capabilities while enforcing compile-time safety.
- **Integer Overflows (SG)**: In Rust, integer overflow is considered a "fail-fast" error. By default, when an integer overflow occurs during an operation, Rust will panic and terminate the program. This behavior helps catch bugs early in development and prevents potential security vulnerabilities. It also offers ways to handle integer overflows gracefully.
- **SQL Injection (SG)**: Rust itself doesn't inherently protect against SQL injection. This protection is usually provided by libraries that parameterize SQL queries, such as rusqlite; see PoC code in [35].
- **Command Injection (SG)**: Rust offers strong protections against command injection vulnerabilities through its string handling and execution mechanisms. The language's emphasis on memory safety and control over system resources helps mitigate the risk of command injection; see PoC code in [35].
- **Cross-Site Scripting (XSS) (UP)**: Rust does not provide inherent protection against XSS. However, web frameworks in Rust, such as Rocket and Actix, have features to mitigate XSS.
- **Race Conditions (RD)**: Rust's ownership model and type system are designed to prevent data races at compile time.
- **Error Handling (RD)**: Rust encourages using the Result type for error handling, which requires explicit handling of errors.
- **Poor Logging (SG)**: Poor logging is more of a design problem than a language issue. Rust offers powerful logging libraries, such as log and env_logger.
- **Insecure Configuration (UP)**: Although Rust's strong typing can catch some configuration errors at compile time, it does not offer direct protections against insecure configurations.
- **Weak Cryptography (SG)**: Rust has libraries that support strong, modern cryptography. However, the correct implementation depends on the developer.
- **Weak Random Numbers (RD)**: Rust's standard library includes a secure random number generator.
- **Using Components with Known Vulnerabilities (SG)**: This is more related to the ecosystem than the language itself. Rust's package manager, Cargo, simplifies updating dependencies.
- **Unvalidated Redirects and Forwards (UP)**: Protection against this is usually provided by web frameworks.
- **Injection (SG)**: Rust's strong typing and absence of eval-like functions lower the risk of code injection.
- **Insecure Storage (UP)**: Not directly related to the language itself.
- **Denial of Service (SG)**: Rust's memory safety and control over low-level details can help build resilient systems, but it does not inherently protect against all types of DoS attacks.
- **Insecure Third-Party Interfaces (UP)**: This issue is usually independent of the programming language.
- **Cross-Site Request Forgery (CSRF) (UP)**: Typically, it is handled by web frameworks rather than the language itself.

In summary, Rust provides strong protections against several of the "19 deadly sins", particularly those related to memory safety and data races. However, some issues, particularly those related to web development or design decisions, are not directly addressed.

D. 10 Common Security Pitfalls in Rust Programming

In our exploration of the security aspects of Rust programming, we have identified several vulnerabilities and potential issues. Beyond the vulnerabilities discussed earlier, it is crucial to highlight general security pitfalls that Rust programmers often encounter. While Rust's safety features significantly reduce certain security risks, awareness and avoidance of common pitfalls are essential for secure coding practices. We outline below 10 common security pitfalls in Rust programming (in Figure 2):

- **Injection Attacks:** Rust's type system can mitigate some risks, but vulnerabilities can arise from improperly handled user input in commands or queries.
- **Integer Overflow and Underflow:** Although Rust provides some level of protection, developers must be vigilant against integer-related vulnerabilities.
- **Request Forgery Attacks:** Rust does not inherently protect against request forgery attacks; developers are responsible for ensuring proper safeguards.
- **Cross-Site Scripting (XSS) Attacks:** In web applications, handling user input safely is vital to preventing XSS vulnerabilities, especially in rendering web pages.
- **Insecure Design:** Despite Rust's advanced features, security vulnerabilities can still result from poor design decisions. It's essential to integrate security considerations into the architectural design. However, these issues are generally independent of Rust and are related to overall security-aware design principles, applicable to any software development.
- **Faulty Access Control:** Inadequate or improperly implemented access control mechanisms can lead to unauthorized access to resources in Rust applications.
- **Logging and Monitoring Failures:** Adequate logging and monitoring are crucial for security, yet often overlooked or poorly implemented in Rust applications, leading to challenges in detecting and responding to security incidents.
- **Security Misconfiguration:** Configuring security settings inadequately in both Rust applications and deployment environments can expose them to risks.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

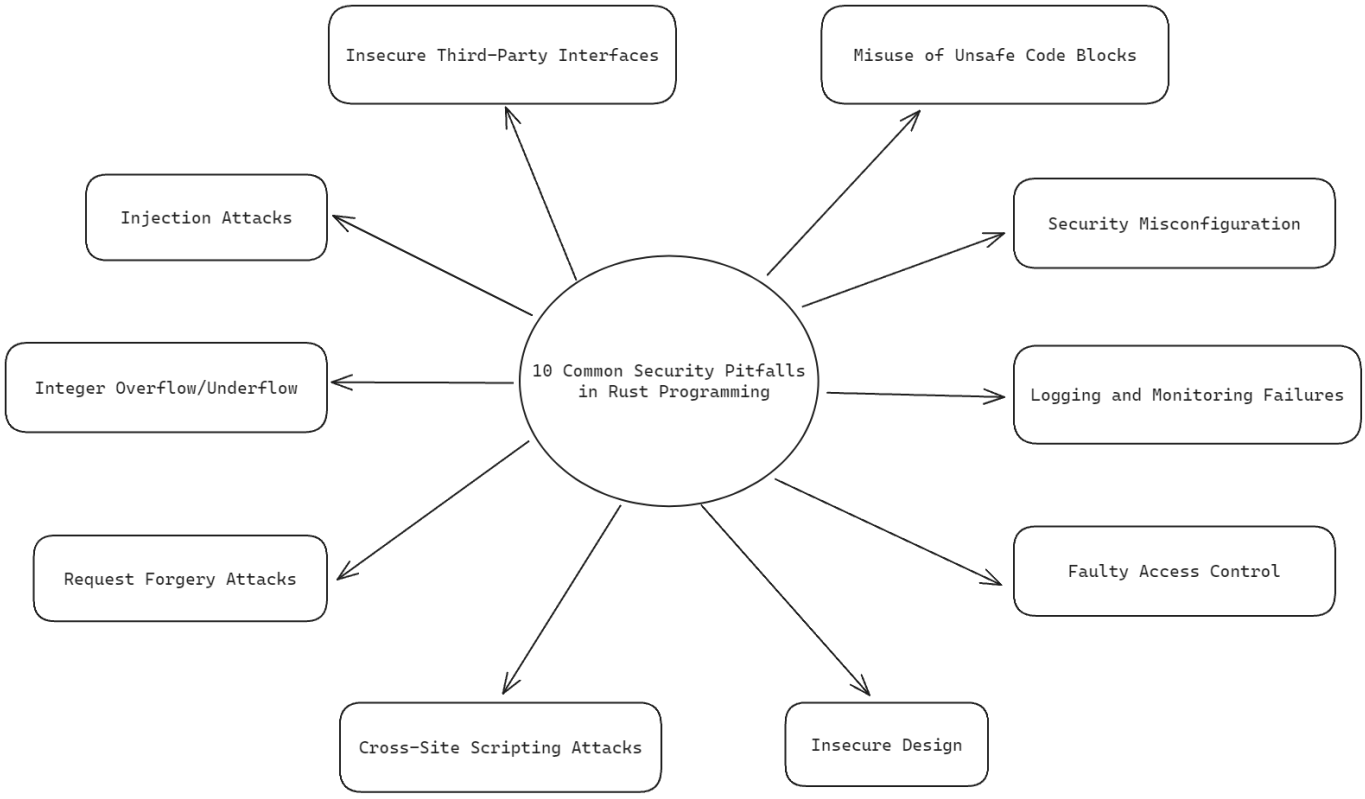


Fig. 2. Ten Common Security Pitfalls in Rust Programming, without Ranking

- **Insecure Third-Party Interfaces:** Relying on third-party libraries or interfaces without proper security vetting can introduce vulnerabilities in Rust applications.
- **Misuse of Unsafe Code Blocks:** Improper use of 'unsafe' code blocks in Rust can compromise the language's inherent safety features, leading to security risks.

This section serves as a guide for common pitfalls and reinforces the importance of comprehensive security practices in Rust development. These insights are intended to support programmers in recognizing and addressing these challenges, thereby enhancing the overall security of software developed in Rust.

1) Injection Attacks: Injection attacks are a prevalent threat in software development, and Rust is not immune to them despite its strong type system and memory safety features. These attacks typically occur when an application unsafely integrates user input into a command or query. While Rust's type system aids in mitigating some risks, vulnerabilities can still arise from improperly validated or sanitized user inputs. This is especially relevant in scenarios involving database queries, command-line arguments, or URL construction, where attackers can exploit unfiltered inputs. Let's examine a specific case of injection attacks: SQL injection.

a) Vulnerable SQL Query in Rust: The following Rust code snippet (Figure 3) demonstrates a vulnerable approach to constructing SQL queries by directly incorporating user input without sanitization:

```
pub fn get_user_by_id(conn: &Connection,
    user_id: &str) -> Result<Vec<User>> {
    // Vulnerable SQL query due to direct
    concatenation of user input
    let query = format!("SELECT id, name, age
    FROM users WHERE id = '{}'", user_id);
    let mut stmt = conn.prepare(&query)?;
    let user_iter = stmt.query_map([], |row| {
        Ok(User {
            id: row.get(0)?,
            name: row.get(1)?,
            age: row.get(2)?,
        })
    })?;
    let mut users = vec![];
    for user in user_iter {
        users.push(user?);
    }
    Ok(users)
}
```

Fig. 3. Vulnerable Code: SQL Injection

This code is prone to SQL injection because it creates a query string by concatenating a user-provided string (‘user_id‘) directly into the SQL command. An attacker could manipulate ‘user_id‘ to alter the structure of the SQL command and execute unintended database operations.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

b) Secured SQL Query in Rust: In contrast, the secure version (Figure 4) uses parameterized queries to safely handle user input, effectively preventing SQL injection:

```
fn get_user_by_id_safe(conn: &Connection,
    user_id: &str) -> Result<Vec<User>> {
    let query = "SELECT id, name, age FROM
    users WHERE id = ?";
    let mut stmt = conn.prepare(query)?;
    let user_iter = stmt.query_map([user_id],
    |row| {
        Ok(User {
            id: row.get(0)?,
            name: row.get(1)?,
            age: row.get(2)?,
        })
    })?;

    let mut users = vec![];
    for user in user_iter {
        users.push(user?);
    }

    Ok(users)
}
```

Fig. 4. Safe Code: Prevents SQL Injection

In the secure example, '?' placeholders are used in the SQL query. These placeholders are then filled with the actual 'user_id' in a way that the database engine understands as data, not as part of the SQL command. This approach ensures that even if 'user_id' contains malicious content, it will not be executed as SQL code.

Similarly, for command injections, it is crucial to avoid using user inputs to construct command strings dynamically [39]. Instead, Rust's standard library offers functions to pass arguments to commands in a way that prevents injection, ensuring that inputs are treated as literal text and not as executable code.

This example underscores the importance of carefully handling user inputs in Rust applications. Despite Rust's memory safety features, developers must remain vigilant against vulnerabilities like injection attacks, emphasizing the need for proper input validation and secure coding practices, such as parameterized queries, to maintain application security.

2) Integer Overflow/Underflow: Integer overflow and underflow represent a class of critical vulnerabilities in software applications where arithmetic operations exceed the maximum or minimum limits of the data type being used. Rust provides some level of protection against these vulnerabilities by including checks in the debug mode that cause a panic when an overflow or underflow occurs. However, in release builds, these checks are not enforced by default for performance, which can lead to silent wrapping and potential security risks.

a) Vulnerable Integer Arithmetic in Rust: The following Rust code snippet (in Figure 5) demonstrates an unsafe approach to integer arithmetic, which can lead to overflow or underflow without any warnings or errors in release builds:

```
pub fn withdrawal(balance: u32, amount: u32)
    -> u32 {
    let mut balance = balance;
    balance -= amount;
    balance
}
```

Fig. 5. Vulnerable Code: Integer Overflow/Underflow

In this code, subtracting a larger 'amount' from 'balance' can cause an underflow, which in release mode would wrap around to a very large number due to the unsigned integer type. This could lead to logical errors in the program and potentially severe security vulnerabilities, such as incorrectly authorizing a financial transaction.

b) Secured Integer Arithmetic in Rust: To prevent such issues, Rust offers built-in methods for safe arithmetic that return a 'Result' or an 'Option' type, which can be explicitly handled. Here is a secure version (in Figure 6) of the function that correctly handles underflow using 'checked_sub':

```
pub fn withdrawal(balance: u32, amount: u32)
    -> Result<u32, &'static str> {
    match balance.checked_sub(amount) {
        Some(new_balance) => Ok(new_balance),
        None => Err("Withdrawal not possible:
    insufficient funds"),
    }
}
```

Fig. 6. Safe Code: Prevents Integer Overflow/Underflow

In this secure example, 'checked_sub' is used for subtraction, which returns 'None' if underflow occurs. Using a 'match' statement, the code can handle the underflow case safely, either by returning an error message indicating that the withdrawal is impossible due to insufficient funds or by implementing alternative logic as needed.

This example highlights the need for developers to be aware of the integer arithmetic behavior in their chosen programming language, particularly in a systems language like Rust, often used for low-level tasks. By utilizing Rust's safe arithmetic functions and adequately handling their results, applications can be more robust and secure against integer overflow and underflow vulnerabilities.

3) Request Forgery Attacks: Request Forgery Attacks, such as Cross-Site Request Forgery (CSRF) and Server-Side Request Forgery (SSRF), represent significant security threats to web applications. These attacks exploit a service's trust in the user or the server itself. While Rust's memory safety and concurrency features are commendable, they offer no inherent protection against these web-specific attack vectors.

a) Cross-Site Request Forgery (CSRF): CSRF attacks deceive a web browser into executing an unwanted action on a trusted application where the user is authenticated. In the context of Rust web applications using frameworks like Rocket, such vulnerabilities emerge when state-changing

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

operations do not require explicit user consent beyond the initial authentication [40].

Consider the example of a Rust application (in Figure 7) with an endpoint to change a user's password, which is vulnerable to CSRF attacks:

```
use rocket::form::Form;

#[derive(FromForm)]
struct PasswordChangeForm {
    new_password: String,
}

#[post("/change_password", data =
    "<password_form>")]
fn change_password(password_form:
    Form<PasswordChangeForm>, user: User) ->
    String {
    // Implement code to change the user's
    password in the database
    // Ensure 'user' is the currently
    authenticated user
    // This code is still CSRF vulnerable
    "Password changed
    successfully".to_string()
}
```

Fig. 7. Vulnerable Code: CSRF

This endpoint is vulnerable because it processes the password change request without verifying the origin of the request, making it susceptible to CSRF attacks. An attacker could craft a malicious website with a form that, when submitted, sends a POST request to this endpoint. If the victim is logged into the Rust application in the same browser, the browser automatically includes the session cookies with the request, leading to an unauthorized password change.

To mitigate CSRF, a token-based strategy is typically employed. As of Rocket version 0.5, automatic CSRF protection was still under discussion, with improvements expected in future versions (See the GitHub issue for the discussion on CSRF protection in Rocket v0.6 [41]).

b) Server-Side Request Forgery (SSRF): SSRF attacks occur when an attacker can induce the server-side application to make requests to arbitrary domains, leading to unauthorized actions or data exposure. Unlike CSRF, SSRF exploits the trust a server has within its own system or between internal services.

In Rust web applications, SSRF can typically occur when user-supplied URLs are used without proper validation to make server-side requests. For instance, fetching a user-specified URL without checking if it points to an internal service or sensitive resource can be exploited.

To prevent SSRF, developers must validate and sanitize all URLs and restrict the server's ability to interact only with whitelisted endpoints. Additionally, following the principle of least privilege when granting network capabilities to the server can mitigate the impact of SSRF attacks.

Request forgery attacks, both CSRF and SSRF, require developers to be proactive in their defense strategies. Employing robust validation, leveraging security features provided by web frameworks, and adhering to security best practices are crucial in protecting Rust web applications from these types of vulnerabilities.

4) Cross-Site Scripting (XSS): Cross-Site Scripting (XSS) attacks are a significant concern in web applications, including those developed in Rust. XSS attacks involve the injection of malicious scripts into webpages viewed by other users, exploiting a user's trust in a particular site.

a) Rust and XSS Vulnerability: Although Rust is known for its robust safety features, it does not inherently protect against XSS attacks [42], [43]. These attacks are primarily concerned with the layer where HTML is generated or manipulated. Rust applications using web frameworks for frontend development, like Rocket, are susceptible to the same XSS vulnerabilities as applications written in other languages.

b) Example of XSS Vulnerability in Rust: Consider a Rust web application that allows users to input data directly displayed on a webpage. An attacker could inject malicious JavaScript code if the application does not properly escape or sanitize the user input. This code could be executed in the browsers of other site users, leading to data theft, session hijacking, or other malicious activities.

c) Protecting Against XSS in Rust: To mitigate XSS risks in Rust applications, developers need to:

- **Escape User Input:** Ensure that user inputs are correctly escaped before rendering on web pages. This prevents malicious scripts from being executed [44].
- **Use Safe Frameworks and Libraries:** Employ frameworks and libraries that automatically handle escaping. For instance, the Ammonia crate in Rust sanitizes HTML to prevent XSS by filtering out harmful tags and attributes (Sometimes even crates like Ammonia have XSS vulnerabilities [43], so developers should always keep themselves informed about such vulnerabilities).
- **Content Security Policy (CSP):** Implement a strong CSP to reduce the severity of any XSS vulnerabilities by restricting where scripts can be loaded from and executed [44].
- **Validate and Sanitize Input:** Rigorously validate and sanitize all user input, especially data that will be included in HTML output.
- **Regular Security Audits:** Conduct regular security reviews and testing, including dynamic application security testing (DAST) and penetration testing, to identify and fix XSS vulnerabilities [44].

While Rust provides significant advantages regarding memory safety, developers must still be diligent in protecting against XSS and other web-based vulnerabilities. Implementing best practices for input handling and utilizing security-focused libraries and frameworks are essential steps in creating secure Rust web applications.

5) Insecure Design: Insecure design in software development refers to flaws that arise not from specific coding errors but from fundamental issues in the software's architecture and design decisions. Even in a language like Rust, known for

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

its emphasis on safety and security, an application's overall security is significantly influenced by its design. We explore common design pitfalls in Rust programming and recommend strategies to foster secure software design.

a) Impact of Insecure Design: While Rust provides strong guarantees against memory safety issues, it does not inherently address higher-level design vulnerabilities such as authentication flaws, inadequate data protection, or insecure communication protocols. These vulnerabilities are often the result of oversight during the design phase and can lead to significant security risks, including data breaches, unauthorized access, and system compromises.

b) Common Design Flaws in Rust Applications:

- **Insufficient Authentication and Authorization:** Overlooking robust authentication and authorization mechanisms can lead to unauthorized access. Rust applications, especially those interfacing with web services, must implement strong authentication protocols and ensure user privileges are correctly managed.
- **Lack of Data Encryption:** Failing to encrypt sensitive data at rest and in transit can expose it to interception and misuse. Rust applications handling confidential information should utilize strong encryption algorithms and libraries to secure data.
- **Ignoring Secure Communication Protocols:** Neglecting to use secure communication channels like HTTPS can make Rust applications vulnerable to man-in-the-middle attacks. Ensuring encrypted communication is critical, especially for web-based applications.

c) Best Practices for Secure Design: To mitigate the risks associated with insecure design, the following best practices are recommended:

- **Threat Modeling:** Early in the design process, conduct threat modeling to identify potential security threats. This proactive approach helps in designing systems that are resilient against identified risks [45], [46].
- **Principle of Least Privilege:** Design systems where components operate with the minimum privileges necessary. This reduces the impact of a potential compromise [47].
- **Secure Defaults:** Ensure that the application's default configuration is secure. This includes settings for user access, data processing, and communication protocols [46].
- **Regular Security Audits:** Conduct regular security reviews and audits of the design to identify and address new and evolving security threats [46].
- **Staying Informed:** Keep abreast of the latest security trends and best practices in software design. Applying up-to-date knowledge can significantly enhance the security posture of Rust applications.

In Rust development, as in any software development endeavor, secure design is just as crucial as secure coding. An application's architecture and design decisions lay the foundation for its overall security. By adhering to best practices in secure design and being mindful of common design pitfalls, Rust developers can significantly reduce the risk of security vulnerabilities in their applications.

6) Faulty Access Control: Faulty access control is a critical security issue that can lead to unauthorized access and data breaches. In the context of Rust programming, while the language offers strong memory safety features, access control largely depends on the application's design and the use of libraries. This section discusses the common challenges and best practices in implementing robust access control in Rust applications.

a) Challenges in Access Control: Access control mechanisms are essential for defining and enforcing who can access what resources in an application. In Rust, the challenges in implementing access control often stem from:

- **Complex User Permissions:** Managing complex user permissions and roles can be challenging, especially in applications with multiple user levels and diverse access needs.
- **Dependency on External Libraries:** Rust's standard library does not provide specific features for access control, leading developers to rely on external libraries, which might vary in their security robustness.
- **Inadequate Session Management:** Implementing secure session management is crucial for web applications. Faulty session management can lead to vulnerabilities like session hijacking and fixation.

b) Best Practices for Access Control: To ensure effective access control in Rust applications, consider the following best practices:

- **Role-Based Access Control (RBAC):** Implement RBAC to manage user permissions efficiently. RBAC allows for grouping permissions into roles, which can be assigned to users, simplifying the management of user privileges.
- **Use of Vetted Libraries:** When relying on external libraries for access control features, choose well-vetted libraries with a strong security track record. Regularly update these libraries to incorporate security patches.
- **Secure Authentication and Session Management:** Implement strong authentication mechanisms and ensure that session management is secure. This includes using secure tokens, implementing session timeouts, and protecting against common session attacks.
- **Regular Access Reviews:** Regularly review and update access control policies to ensure they align with the current organizational structure and user roles.
- **Audit and Logging:** Maintain comprehensive audit logs for access control events. Monitoring and analyzing these logs can help detect unauthorized access attempts and improve the overall security posture.

Faulty access control can have severe implications for the security of a Rust application. While Rust provides the tools for building safe and concurrent applications, access control relies more on the application's design and the secure implementation of libraries and frameworks. By adopting

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

robust access control practices and staying vigilant in their application, developers can significantly enhance the security of their Rust applications.

7) Logging and Monitoring Failures: Effective logging and monitoring are crucial for the security and stability of any software application, including those developed in Rust. While Rust's language features promote safety and concurrency, they do not inherently provide solutions for logging and monitoring. This section addresses common pitfalls in logging and monitoring Rust applications and suggests best practices to mitigate these issues.

a) Importance of Logging and Monitoring: Logging and monitoring are pivotal in detecting, diagnosing, and responding to security incidents and system failures. In Rust applications, ineffective logging and monitoring can lead to:

- **Inadequate Detection of Security Incidents:** Without proper logging, malicious activities or security breaches may go unnoticed, increasing the risk of damage.
- **Difficulty in Troubleshooting and Debugging:** Insufficient logging can hinder the identification and resolution of bugs or performance issues, affecting the reliability and efficiency of the application.
- **Compliance Issues:** Failure to maintain adequate logs can lead to non-compliance with regulatory requirements, especially in industries where logging is mandated for audit trails.

b) Best Practices for Logging and Monitoring: Implementing effective logging and monitoring in Rust applications involves several best practices:

- **Comprehensive Logging Strategy:** Develop a logging strategy that covers what to log, at what level, and how to securely store and manage logs. Ensure that logs capture essential information for security and operational insights.
- **Use of Robust Logging Frameworks:** Utilize mature and well-supported logging frameworks in Rust, such as `log` and `env_logger`, which provide flexibility and ease of integration.
- **Real-time Monitoring and Alerting:** Implement real-time monitoring tools to promptly detect and alert on abnormal activities or performance issues.
- **Log Analysis and Correlation:** Regularly analyze logs to identify patterns or anomalies. Correlate logs from different sources to gain a comprehensive understanding of events.
- **Secure and Compliant Log Management:** Ensure logs are stored securely, with access controls in place. Logs should be managed in compliance with data protection regulations.

Effective logging and monitoring are vital for maintaining the security and integrity of Rust applications. By implementing a robust logging and monitoring strategy and utilizing appropriate tools and practices, developers can significantly enhance their ability to detect, diagnose, and respond to Rust application issues, reinforcing their overall security and reliability.

8) Security Misconfiguration: Security misconfiguration is one of the most common vulnerabilities in software applications, arising from improper setup or default configurations that are not secure. In Rust applications, as with any other technology, attention to configuration details is crucial for ensuring system security. This section highlights the typical areas where security misconfiguration can occur in Rust applications and provides guidelines to prevent such vulnerabilities.

a) Typical Areas of Misconfiguration: Security misconfigurations in Rust applications can manifest in various ways:

- **Default Settings:** Leaving default settings unchanged, especially those related to security, can expose applications to known vulnerabilities [48].
- **Insecure Database Configurations:** Inadequately secured database connections or default credentials can lead to unauthorized access [48].
- **Improper File Permissions:** Incorrect file and directory permissions can give attackers access to sensitive data or system files [48].
- **Exposed Sensitive Information:** Exposing sensitive information like debug details, stack traces, or cryptographic keys through error messages or logs [48].
- **Lack of Security Features in External Libraries:** Using external libraries without properly configuring their security features.

b) Best Practices to Prevent Security Misconfiguration: Implementing the following best practices can significantly reduce the risk of security misconfiguration in Rust applications:

- **Regular Configuration Reviews:** Conduct regular reviews of application configurations, particularly after updates or changes, to ensure security settings are appropriate and up to date.
- **Minimal Necessary Permissions:** Apply the principle of least privilege to file, database, and network permissions. Only grant access levels necessary for the operation.
- **Secure Default Settings:** Customize default settings to enforce security, including turning off unnecessary features and services.
- **Manage Sensitive Information:** Ensure sensitive information like keys, credentials, and personal data is securely managed and not exposed in logs or error messages.
- **Update and Patch Libraries:** Regularly update external libraries to incorporate security patches and review their configurations for security implications.
- **Security Hardening Guides and Checklists:** Utilize security hardening guides and checklists to systematically address potential misconfigurations.

Security misconfiguration can lead to severe vulnerabilities in Rust applications. By being vigilant about configuration details, regularly reviewing and updating settings, and following best practices for security management, developers can significantly enhance the security posture of their applications, mitigating the risks associated with misconfiguration.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

9) Misuse of Unsafe Code: The Rust programming language is lauded for its emphasis on safety, particularly memory safety. However, Rust also provides an ‘unsafe‘ keyword, allowing developers to opt out of some of these safety guarantees for various reasons [49], such as interfacing with low-level system components or optimizing performance [50]. While powerful, the misuse of ‘unsafe‘ code can introduce significant security vulnerabilities. This section discusses the responsible use of ‘unsafe‘ code in Rust and strategies to minimize its risks.

a) Risks Associated with Unsafe Code in Rust: ‘Unsafe‘ code in Rust bypasses the compiler’s safety checks, which can lead to several risks:

- **Memory Safety Violations:** ‘Unsafe‘ code can lead to issues like dereferencing null or dangling pointers, leading to undefined behavior or security vulnerabilities such as buffer overflows [50], [51].
- **Concurrency Issues:** Incorrect handling of concurrent operations in ‘unsafe‘ code can result in data races and undefined behavior [51].
- **Violations of Rust’s Ownership Model:** ‘Unsafe‘ code can violate Rust’s ownership rules, potentially leading to memory leaks or double-free errors [50], [51].

Consider a scenario depicted in Figure 8, where an unsafe block executes low-level operations, such as opening a file through direct system calls. In these instances, the safety mechanisms of Rust, designed to prevent memory and resource leaks, are circumvented. Should the programmer fail to explicitly close the file descriptor acquired via these operations, it may result in resource leakage. This differs from memory leaks, which involve un-freed allocated memory. Here, resource leakage denotes the depletion of available file descriptors, a limited system resource. Such oversight can lead to the application’s inability to open new files or sockets upon reaching the open file descriptor limit, potentially causing broader system issues if not adequately addressed. Consequently, ensuring that every file descriptor opened within an unsafe block is properly closed is imperative, thereby maintaining system stability and averting resource leakage.

b) Best Practices for Using Unsafe Code in Rust: To mitigate the risks associated with ‘unsafe‘ code, consider the following best practices:

- **Minimize Use of Unsafe Code:** Limit the use of ‘unsafe‘ code to situations where it is necessary, such as interfacing with hardware or legacy C code [51].
- **Isolate Unsafe Code:** Encapsulate ‘unsafe‘ code in small, well-defined modules or functions. This isolation makes it easier to audit and test the unsafe portions of your codebase.
- **Document Unsafe Code:** Clearly document the reasoning behind the use of ‘unsafe‘ code and the precautions taken to uphold safety guarantees.
- **Peer Review:** Unsafe code should undergo rigorous peer review by experienced Rust developers, focusing on the necessity and safety of the ‘unsafe‘ operations.

```
use libc::{c_char, c_int, open, read,
    O_RDONLY};
use std::{ffi::CString, thread,
    time::Duration};

fn read_file(filename: &str) ->
    Result<String, String> {
    let c_filename =
    CString::new(filename).expect("CString::new
    failed");
    let fd: c_int = unsafe {
    open(c_filename.as_ptr() as *const
    c_char, O_RDONLY) };

    if fd < 0 {
        return Err("Unable to open
    file".to_string());
    }

    let mut buffer = vec![0u8; 1024];
    let bytes_read = unsafe { read(fd,
    buffer.as_mut_ptr() as *mut libc::c_void,
    buffer.len()) };

    if bytes_read < 0 {
        return Err("Unable to read
    file".to_string());
    }

    // Remove extra null bytes from the buffer
    buffer.resize(bytes_read as usize, 0);

    // Failing to close the file descriptor
    results in resource leakage.
    // unsafe { close(fd); }

Ok(String::from_utf8_lossy(&buffer).into_owned())
}

fn main() {
    let filename = "test.txt";
    loop {
        println!("Reading file...");
        match read_file(filename) {
            Ok(contents) => println!("File
    contents: {}", contents),
            Err(err) => println!("Error: {}",
    err),
        }

    thread::sleep(Duration::from_millis(50));
    }
}
```

Fig. 8. Vulnerable Code: Resource Leak with Unsafe Rust

- **Automated Testing:** Implement comprehensive automated testing, including fuzz testing, to uncover potential issues in ‘unsafe‘ code [14].
- **Stay Informed:** Keep up-to-date with best practices and guidelines from the Rust community regarding the use of ‘unsafe‘ code.

The use of ‘unsafe‘ code in Rust, while sometimes nec-

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

essary, should be approached with caution. By minimizing its use, isolating it, documenting it, and rigorously testing it, developers can mitigate the risks associated with bypassing the compiler's safety checks. Responsible use of 'unsafe' code is essential for maintaining the security and reliability of Rust applications.

10) Insecure Third-Party Interfaces: In today's software development ecosystem, reliance on third-party interfaces and libraries is commonplace. While this approach boosts efficiency and productivity, it also introduces potential security risks, especially if these third-party components are insecure or misconfigured. Rust applications are no exception to this, and developers must be vigilant about the third-party interfaces they integrate. This section explores the challenges and best practices related to using third-party interfaces in Rust applications.

a) Challenges with Third-Party Interfaces: Integrating third-party interfaces in Rust applications can present several challenges:

- **Dependency Vulnerabilities:** Libraries and interfaces may contain vulnerabilities that can be exploited, compromising the security of the entire application.
- **Lack of Control:** Using third-party components often means relinquishing control over certain aspects of the application, which might lead to security lapses if these components are not adequately maintained.
- **Incompatibilities:** Incompatibilities between different libraries or between libraries and the Rust runtime can lead to unexpected behavior and potential security issues.
- **Outdated Components:** Using outdated versions of libraries and interfaces can expose applications to known vulnerabilities that have been fixed in later versions.

b) Best Practices for Secure Use of Third-Party Interfaces: To safely integrate third-party interfaces in Rust applications, consider the following best practices:

- **Vet Third-Party Libraries:** Before integrating a third-party library, vet it for security. Check its reputation, maintenance history, and community feedback.
- **Keep Dependencies Updated:** Regularly update third-party libraries to the latest versions to ensure you have the most recent security patches and features.
- **Monitor for Vulnerabilities:** Use tools to monitor dependencies for known vulnerabilities. Automated vulnerability scanning tools can be integrated into the development workflow.
- **Understand the Code:** As much as possible, understand the code of third-party libraries, at least those parts you integrate into your application. This understanding can be crucial for identifying potential security risks [52].
- **Limit Dependencies:** Minimize the number of third-party dependencies. The more dependencies your project has, the larger the attack surface.
- **Isolate Critical Components:** Isolate critical components of your application from third-party code. This isolation can prevent cascading failures or security breaches from affecting core functionalities [52].

While third-party interfaces are invaluable in modern software development, their integration must be cautiously approached, especially in the context of Rust applications. By carefully selecting, monitoring, and managing these third-party components, developers can mitigate potential security risks and maintain the integrity and security of their Rust applications.

In the following sections, we will delve deeper into the analysis of past vulnerabilities in the Rust language and its ecosystem and shed light on the time taken to address these vulnerabilities and the current open issues in the Rust security landscape. This comprehensive analysis aims to provide a better understanding of the vulnerabilities in Rust and guide developers and researchers in effectively addressing security concerns in Rust-based software.

E. CVEs Addressed by Rust Security Advisory

A quick search on CVE Mitre with the keyword "Rust" returns over 400 vulnerabilities at the time of writing. Various researchers have analyzed the CVEs, and the Rust community actively fix them once discovered [10], [53]. However, Rust's security advisory only addresses some of these vulnerabilities: CVE-2021-42574 [54], CVE-2022-21658 [55], CVE-2022-24713 [56], CVE-2022-36113 [57], CVE-2022-36114 [57], CVE-2022-46176 [58], CVE-2023-38497 [59] and CVE-2024-24576 [60].

One of the CVEs acknowledged by the Rust security advisory on their blog is CVE-2022-46176 [58]. This vulnerability, found in Cargo's Rust package manager, could allow for man-in-the-middle (MITM) attacks due to a lack of SSH host key verification when cloning indexes and dependencies via SSH. All Rust versions containing Cargo before 1.66.1 are vulnerable. Rust version 1.66.1 was released to mitigate this, which checks the SSH host key and aborts the connection if the server's public key is not already trusted.

F. Comparison of Rust Static Analysis Tools with Python, Java, and C++

Rust has been gaining traction due to its focus on safety and performance. As a young language, Rust's ecosystem of static analysis tools is still in rapid development. The primary tool for static analysis in Rust is the Rust compiler, which includes a robust type system and borrow checker that prevents many bugs at compile time. Moreover, tools like Clippy [61] and Mirchecker [62] provide lints to catch common mistakes and improve Rust code.

In contrast, languages like Python, Java, and C++ have been around for a considerable time and have a mature set of static analysis tools. Python, a dynamically typed language, relies on tools like PyLint, PyFlakes, and Bandit for static analysis. With its static type system, Java uses tools like FindBugs, PMD, and Checkstyle. C++, known for its complexity and flexibility, employs tools like cppcheck and Clang Static Analyzer.

While each language has its unique set of static analysis tools, the effectiveness of these tools can vary based on the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

language's features and characteristics. The rapidly evolving Rust ecosystem is a testament to the language's growing popularity and commitment to safety and performance. On the other hand, the mature toolsets of Python, Java, and C++ provide robust support for detecting potential bugs and improving code quality, backed by years of development and refinement.

V. DISCUSSION

In this study, we have explored the security implications of using the Rust programming language, which is gaining traction in the software industry due to its claims of safety and security. Our findings indicate that while Rust offers certain security advantages, it is not immune to vulnerabilities, and there are areas where it falls short compared to other, more mature languages.

A. Rust Security Model and Its Attributes

The Rust programming language incorporates a comprehensive security model to facilitate safe systems programming. The attributes of this model are integral to its capability to minimize common security vulnerabilities. These attributes of Rust are also the reason why many people love the language so much [63]. Figure 9 provides a visual representation of these attributes, each contributing to the overall robustness of the language's approach to security.

1) Key Attributes of the Rust Security Model:

a) Ownership and Borrowing: Central to Rust's memory safety guarantees are the principles of ownership and borrowing. These mechanisms ensure that memory is properly allocated and deallocated, preventing common vulnerabilities such as buffer overflows and dangling pointers.

b) Lifetime Tracking: Rust's compiler enforces lifetime annotations, which specify the scope of validity for references. This prevents memory leaks and unauthorized memory access, which are typical in systems programming.

c) Public/Private Module Management: By allowing developers to define public or private modules, Rust ensures encapsulation and control over data exposure, mitigating the risks associated with unauthorized access (by default modules are private).

d) Safe and Unsafe Code: While Rust defaults to safe code, it provides the 'unsafe' keyword to perform certain low-level operations. This dichotomy allows for flexibility where necessary yet maintains strict safety checks by default.

e) Memory Safety Without Garbage Collection: Rust provides memory safety guarantees without a garbage collector, offering deterministic performance critical in systems programming.

f) Zero-Cost Abstractions: Rust's zero-cost abstractions allow developers to write high-level abstractions without performance penalties, often associated with high-level languages.

g) Type System and Concurrency: The language's type system and concurrency model prevent data races and ensure safe concurrent programming, a common source of vulnerabilities in multi-threaded applications.

h) Compile-Time Memory Management: Rust's ability to manage memory at compile time prevents a class of bugs that can be exploited at runtime, enhancing the security and performance of applications.

i) Error Handling: Rust uses the 'Result' and 'Option' types for error handling, which prevents crashes due to unhandled errors and exceptions, enhancing the reliability of the application.

j) Cargo Package Manager and Dependency Management: Rust's package manager, Cargo, assists in managing dependencies securely and facilitates easy maintenance and updates, ensuring that security updates are seamlessly integrated into the development process.

B. Mapping of SANS TOP 25, OWASP TOP 10 and 19 Deadly Sins

Our research has shown that writing vulnerable software in Rust is possible. This finding is essential, as it challenges the perception that Rust is inherently secure. While Rust's design does make some types of vulnerabilities harder to introduce, it is not a panacea. Other security aspects are as problematic in Rust as in any other language. This point underscores the fact that while language choice can influence the security of a software system, it is not the only factor. Good security practices are essential, regardless of the language used.

Some vulnerabilities are hard or impossible to solve through an improved programming language as these belong to a "non-decidable" category. Therefore, writing a compiler or defining a programming language that identifies and eliminates such problems is impossible. However, we have observed that Rust does offer improvements over other languages in handling these issues, which is a positive sign.

One of the challenges we encountered in our research is the relative immaturity of Rust compared to other languages. There are fewer studies on Rust security, and the tools and support for secure development are not as robust. For example, SonarQube [64], a popular tool for static analysis of code to detect bugs, code smells, and security vulnerabilities, does not currently support Rust. This lack of tooling can significantly impede Rust's adoption in an industrial context, where such tools are critical for finding vulnerabilities and passing cybersecurity certifications.

Our discussions with industry experts found that Rust's high learning curve is another potential barrier to its adoption. More investigation is needed to understand the security consequences of this compared to other languages that might be easier to learn. The lack of a "competent" workforce skilled in Rust is another challenge that needs to be addressed.

In our analysis of the SANS Top 25, Rust provides inherent protections against 24% of the vulnerabilities, some safeguards against 28% of vulnerabilities, and does not offer protection or does not apply to 48% of the vulnerabilities. We made notable observations when comparing Rust with other programming languages like C, C++, and Java. C does not offer any inherent protections against the vulnerabilities listed in the SANS Top 25, as it was designed to be minimal and efficient.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

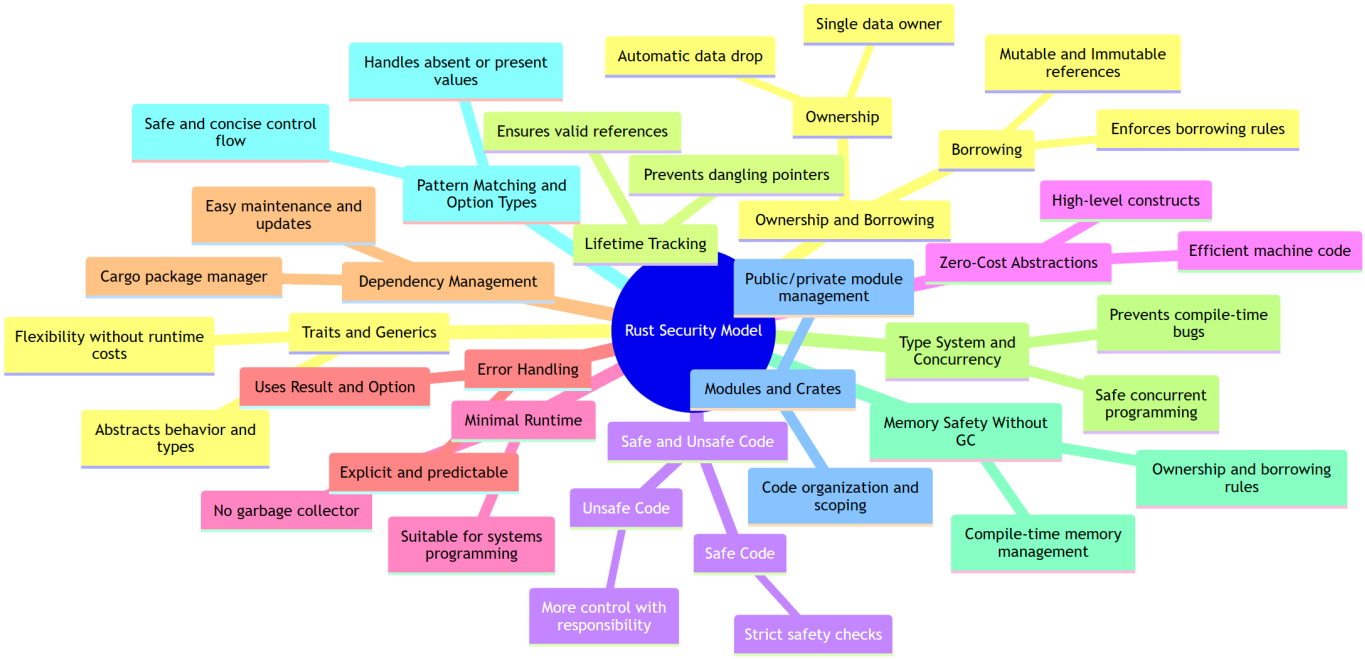


Fig. 9. Rust Security Model

C++, on the other hand, provides safeguards against particular vulnerabilities, such as CWE-787 and CWE-15. Examples of language features that can protect against these vulnerabilities include the C++ Standard Template Library (STL) and other features. Nevertheless, the C++ programming language does not inherently protect against them. In our study, we observe that C++ safeguards against only 24% of the vulnerabilities in the SANS Top 25. However, Java utilizes a garbage collector that inherently protects against memory-related issues. This feature puts Java closer to Rust in terms of protection [65].

Our analysis of the OWASP findings revealed that not a single finding is of the type RD, which is to be expected, as Rust is more a system-level programming language rather than a programming language for web technologies. Compared to C, C++, and Java, which are widely used in the industry, Rust shows promise but has limitations.

Our analysis of the 19 Deadly Sins showed that Rust provides inherent protections against 21% of these sins, offers safeguards for 47% of them, and leaves 32% of the sins unprotected.

We do not expect any current or future programming language to be able to cover 100% of the vulnerabilities, as many coding guidelines in CWE are non-decidable. However, our work shows that Rust does a commendable job addressing many CWE guidelines.

Our inspiration to use a three-point scale (RD, SG, and UP) in our analysis is based on the work by Jacoby (1971) [66], who argued that ”Three-point Likert scales are good enough.”

In addition to our Rust security vulnerabilities analysis, we have also delineated 10 common security pitfalls that developers should be vigilant about. This compilation is not exhaustive but serves as a crucial starting point for Rust developers to cultivate a security-first mindset. By highlighting these pitfalls—ranging from Injection Attacks to the Misuse of Unsafe Code—we aim to underscore the multifaceted nature of security in Rust programming.

These pitfalls were selected based on their prevalence and potential impact on the security of Rust applications. Pitfalls are accompanied by examples and best practices, offering developers concrete strategies to avoid or mitigate these risks. This proactive approach is essential in a landscape where security threats are continually evolving, and developer awareness can significantly influence the resilience of software systems.

Furthermore, discussing these pitfalls complements our broader research findings, providing a comprehensive overview of the security considerations specific to Rust. It acknowledges that while Rust’s design offers significant safety features, security is a broader concern that extends beyond the language’s inherent mechanisms. Developers must remain cognizant of the various ways vulnerabilities can manifest, whether through logical errors, misconfigurations, or the integration of insecure third-party components.

Our exploration of common security pitfalls in Rust programming serves as both a cautionary tale and a roadmap for secure development practices. As the Rust ecosystem continues to grow and evolve, so will the challenges and strategies for ensuring the security of Rust applications. We hope this guide will be a valuable resource for developers, encouraging a holistic approach to security that integrates seamlessly with Rust’s safety features.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

The authors consider the present work essential as Rust's usage for software development continues to grow. Without awareness of potential vulnerabilities, we risk replacing one problem with another. It is crucial to emphasize the security limitations of Rust early on rather than treating security as an add-on feature. Security should be prioritized from the inception of every project. Furthermore, due to Rust being a relatively new language, standardized testing tools for assessing compliance with ISO/IEC security standards are not yet available, or very few. This lack of tools makes it challenging to introduce Rust into the industry.

The present work does not focus on finding novel software weaknesses specific to the Rust programming language but rather on comparing well-known vulnerabilities, e.g., as present in secure programming standards, and their relation to the Rust programming language. Additional investigation is needed to understand potential vulnerabilities when developing software in Rust which are caused by the language itself.

In conclusion, our work contributes to scientific knowledge and industry practice by shedding light on the security implications of using Rust. While Rust is rising in significance and the industry is starting to adopt it, there is a lack of studies on its security aspects. Our work closes this gap and shows that while it is still possible to write vulnerabilities in Rust, some problems are well-considered. As Rust continues to grow in popularity, we hope our findings will help guide its development in a direction that prioritizes security and that our work will serve as a foundation for further research in this area.

While the interviews carried out in the present work include a limited number of participants, the results of the present work are validated. The authors did not only confirm some vulnerabilities with proof-of-concept code but also conducted interviews with highly experienced security experts. Nevertheless, the mapping to protection levels, while dependent on the authors' and interviewees' experience, can also change in future releases of the Rust programming language.

VI. CONCLUSION AND FUTURE WORK

Our research provided valuable insights into the security implications of the Rust programming language. While Rust has significantly enhanced software security, we have demonstrated that it is not immune to vulnerabilities. Our findings challenge the notion that Rust is inherently secure and highlight the need for robust security practices, regardless of the language used.

Our study has also shed light on the challenges associated with Rust's relative immaturity compared to other, more established languages. The lack of comprehensive studies on Rust security, the absence of robust tooling for secure development, and the high learning curve associated with Rust are all areas that require attention. Furthermore, the shortage of a skilled workforce in Rust is a significant barrier that needs to be addressed to facilitate its broader adoption in the industry.

Despite these challenges, Rust shows promise. Its design makes specific vulnerabilities harder to introduce and offers improvements over other languages in handling "non-decidable" problems. As Rust continues gaining traction in the software industry, it is crucial to investigate its security implications and develop tools and practices to mitigate potential vulnerabilities.

As the following steps, there are several avenues for future work. One of the critical areas is the development of tools to support secure development in Rust. These tools include static application security testing tools like SonarQube, which are critical for finding vulnerabilities and passing cybersecurity certifications. Another area of focus is the development of comprehensive training programs to lower Rust's learning curve and build a competent workforce skilled in Rust. Further research should also focus on studying real-world examples of security concerns in Rust applications to strengthen the points made by the authors. Additionally, providing suggestions on what Rust needs to improve, specifically in terms of the 10 common security pitfalls identified, will be essential. Comparing Rust with the Go programming language, particularly regarding inherent security considerations, will also provide valuable insights.

As more software is developed in Rust, it is crucial to maintain a sense of urgency in highlighting its security shortcomings. Security should not be an afterthought but should be integrated from the beginning of every project. We hope our work will contribute to developing safer and more secure software systems.

REFERENCES

[1] T. E. Gasiba and S. Amburi, "I Think This is the Beginning of a Beautiful Friendship - On the Rust Programming Language and Secure Software Development in the Industry," in *Proceedings of the Eighth International Conference on Cyber-Technologies and Cyber-Systems (CYBER 2023)*, IARIA. ThinkMind, September 2023, pp. 19–26. [Online]. Available: https://www.thinkmind.org/index.php?view=article&articleid=cyber_2023_1_40_80031

[2] Yalantis, "Rust Market Overview," 2023, accessed: July 16, 2023. [Online]. Available: https://yalantis.com/blog/rust-market-overview/

[3] E. D. C. Garcia, "Rust Makes Us Better Programmers," 2023, accessed: July 16, 2023. [Online]. Available: https://thenewstack.io/rust-makes-us-better-programmers/

[4] S. O. Ryan Donovan, "Why the developers who use Rust love it so much," Jun 2020, accessed: July 16, 2023. [Online]. Available: https://stackoverflow.blog/2020/06/05/why-the-developers-who-use-rust-love-it-so-much/

[5] S. Overflow, "Stack Overflow Developer Survey 2023," https://survey.stackoverflow.co/2023/#section-admired-and-desired-programming-scripting-and-markup-languages, 2023, accessed: July 16, 2023.

[6] J. Barron, "Rust's Addition to the Linux Kernel Seen as 'Enormous Vote of Confidence' in the Language," *SD Times*, Nov. 2022, accessed: July 16, 2023.

[7] Writing Linux Kernel Modules in Rust. [Online]. Available: https://www.linuxfoundation.org/webinars/writing-linux-kernel-modules-in-rust

[8] Office of the National Cyber Director, "Title of the report," White House, Washington, D.C., Tech. Rep., 02 2024. [Online]. Available: https://www.whitehouse.gov/wp-content/uploads/2024/02/Final-ONCD-Technical-Report.pdf

[9] G. S. Team, "Memory-safe languages in Android 13," 2022, accessed: July 16, 2023. [Online]. Available: https://security.googleblog.com/2022/12/memory-safe-languages-in-android-13.html

[10] "RustSec Advisory Database," 2023, accessed: July 16, 2023. [Online]. Available: https://rustsec.org/advisories/

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[11] National Security Agency, "U.s. and international partners issue recommendations to secure software products," Press Release, 12 2023. [Online]. Available: https://www.nsa.gov/Press-Room/Press-Releases-Statements/Press-Release-View/Article/3608324/us-and-international-partners-issue-recommendations-to-secure-software-products/

[12] "IEC 62443," international Electrotechnical Commission (IEC) Standards.

[13] International Electrotechnical Commission, "Understanding IEC 62443," https://www.iec.ch/blog/understanding-iec-62443, accessed: July 16, 2023.

[14] S. Hu, B. Hua, and Y. Wang, "Comprehensiveness, automation and lifecycle: A new perspective for rust security," in *2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS)*, 2022, pp. 982–991.

[15] Z. Hanley, "Rust Won't Save Us: An Analysis of 2023's Known Exploited Vulnerabilities," Feb. 2024. [Online]. Available: https://www.horizon3.ai/attack-research/red-team/analysis-of-2023s-known-exploited-vulnerabilities/

[16] Glitchbyte, "Rust wont save us, but its ideas will," Feb. 2024. [Online]. Available: https://glitchbyte.io/posts/rust-wont-save-us/

[17] J. Getreu, "Embedded System Security with Rust - Case Study of Heartbleed."

[18] "How Rust Prevents Out of Bound Reads/Writes." [Online]. Available: https://ancat.github.io//rust/2017/01/21/rust-out-of-bounds.html

[19] SANS Institute, "Top 25 Software Errors," https://www.sans.org/top25-software-errors/, accessed: July 16, 2023.

[20] OWASP Foundation, "OWASP Top Ten," https://owasp.org/www-project-top-ten/, accessed: July 16, 2023.

[21] M. Howard, D. LeBlanc, and J. Viega, *19 Deadly Sins of Software Security: Programming Flaws and How to Fix Them*. New York: McGraw-Hill, 2005, accessed: July 16, 2023.

[22] T. Espinha Gasiba, U. Lechner, M. Pinto-Albuquerque, and D. Méndez, "Is Secure Coding Education in the Industry Needed? An Investigation Through a Large Scale Survey," in *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 2021, pp. 241–252, accessed: July 16, 2023.

[23] J. Sible and D. Svoboda, "Rust Software Security: A Current State Assessment," Carnegie Mellon University, Software Engineering Institute's Insights (blog), Dec 2022, accessed: July 16, 2023. [Online]. Available: https://doi.org/10.58012/0px4-9n81

[24] G. Wassermann and D. Svoboda, "Rust Vulnerability Analysis and Maturity Challenges," Carnegie Mellon University, Software Engineering Institute's Insights (blog), Jan 2023, accessed: July 16, 2023. [Online]. Available: https://doi.org/10.58012/t0m3-vb66

[25] B. Qin, Y. Chen, Z. Yu, L. Song, and Y. Zhang, "Understanding Memory and Thread Safety Practices and Issues in Real-World Rust Programs," in *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 763–779, accessed: July 16, 2023. [Online]. Available: https://doi.org/10.1145/3385412.3386036

[26] X. Zheng, Z. Wan, Y. Zhang, R. Chang, and D. Lo, "A Closer Look at the Security Risks in the Rust Ecosystem," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 2, pp. 34:1–34:30, Dec. 2023. [Online]. Available: https://doi.org/10.1145/3624738

[27] A. Balasubramanian, M. S. Baranowski, A. Burtsev, A. Panda, Z. Rakamarić, and L. Ryzhyk, "System programming in rust: Beyond safety," in *Proceedings of the 16th Workshop on Hot Topics in Operating Systems*, ser. HotOS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 156–161. [Online]. Available: https://doi.org/10.1145/3102980.3103006

[28] I. J. S. 22, "ISO/IEC TR 24772-1:2019 - Programming languages — Guidance to avoiding vulnerabilities in programming languages — Part 1: Language-independent guidance," Online, 12 2019, accessed: July 16, 2023. [Online]. Available: https://www.iso.org/standard/71091.html

[29] T. M. Corporation, "Common Weakness Enumeration (CWE)," Online, 2023, accessed: July 16, 2023. [Online]. Available: https://cwe.mitre.org/

[30] ANSSI, "Publication: Programming Rules to Develop Secure Applications With Rust," https://www.ssi.gouv.fr/guide/programming-rules-to-develop-secure-applications-with-rust/, 2023, (accessed July 16, 2023).

[31] "Rust - Analysis Tools," 2023, accessed: July 16, 2023. [Online]. Available: https://analysis-tools.dev/tag/rust

[32] R. Bagnara, A. Bagnara, and P. M. Hill, "Coding Guidelines and Undecidability," *arXiv*, Dec 2022, accessed: July 16, 2023. [Online]. Available: http://arxiv.org/abs/2212.13933

[33] T. Espinha Gasiba, U. Lechner, M. Pinto-Albuquerque, and D. Mendez Fernandez, "Awareness of Secure Coding Guidelines in the Industry - A First Data Analysis," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020, pp. 345–352, accessed: July 16, 2023.

[34] T. Espinha Gasiba, U. Lechner, and M. Pinto-Albuquerque, "Sifu - a Cybersecurity Awareness Platform with Challenge Assessment and Intelligent Coach," *Cybersecurity*, vol. 3, no. 1, p. 24, 12 2020, accessed: July 16, 2023.

[35] S. Amburi, "Sathwik-Amburi/secure-software-development-with-rust: Secure Software Development with Rust," https://github.com/Sathwik-Amburi/secure-software-development-with-rust, Aug. 2023, last accessed: 2023-08-14. [Online]. Available: https://doi.org/10.5281/zenodo.8247155

[36] M. Noseda, F. Frei, A. Rüst, and S. Künzli, "Rust for secure iot applications: why c is getting rusty," in *Embedded World Conference*. Nuremberg, Germany: WEKA, June 2022. [Online]. Available: https://digitalcollection.zhaw.ch/handle/11475/25046

[37] H. Xu, Z. Chen, M. Sun, Y. Zhou, and M. R. Lyu, "Memory-safety challenge considered solved? an in-depth study with all rust cves," *ACM Trans. Softw. Eng. Methodol.*, vol. 31, no. 1, sep 2021. [Online]. Available: https://doi.org/10.1145/3466642

[38] "National vulnerability database: Cve-2021-29922," https://nvd.nist.gov/vuln/detail/cve-2021-29922, May 2021, accessed: 2024-06-10.

[39] "Rust Command Injection: Examples and Prevention." [Online]. Available: https://www.stackhawk.com/blog/rust-command-injection-examples-and-prevention/

[40] "Rust CSRF Protection Guide: Examples and How to Enable It." [Online]. Available: https://www.stackhawk.com/blog/rust-csrf-protection-guide-examples-and-how-to-enable-it/

[41] SergioBenitez, "Add (more) csrf protection," https://github.com/rwf2/Rocket/issues/14, 2016, accessed: 24th Feb.

[42] "Rust XSS Guide: Examples and Prevention." [Online]. Available: https://www.stackhawk.com/blog/rust-xss-guide-examples-and-prevention/

[43] "Cross-site scripting (xss) affecting ammonia package." [Online]. Available: https://security.snyk.io/vuln/SNYK-RUST-AMMONIA-2929007

[44] OWASP Foundation, "Cross Site Scripting (XSS) Prevention Cheat Sheet," https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html, Access Year.

[45] D. K. Sohr, "Architectural Aspects of Software Security." [Online]. Available: https://user.informatik.uni-bremen.de/sohr/papers/HabilSyn.pdf

[46] L. Futcher and R. von Solms, "Guidelines for secure software development," *ACM*, p. 56–65, 2008. [Online]. Available: https://doi.org/10.1145/1456659.1456667

[47] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," *Communications of the ACM*, vol. 17, no. 7, 07 1974, presented at the Fourth ACM Symposium on Operating System Principles (October 1973). [Online]. Available: https://www.cs.virginia.edu/~evans/cs551/saltzer/

[48] R. A. Khan, S. U. Khan, H. U. Khan, and M. Ilyas, "Systematic literature review on security risks and its practices in secure software development," *IEEE Access*, vol. 10, pp. 5456–5481, 2022.

[49] V. Astrauskas, C. Matheja, F. Poli, P. Müller, and A. J. Summers, "How do programmers use unsafe rust?" *Proc. ACM Program. Lang.*, vol. 4, no. OOPSLA, nov 2020. [Online]. Available: https://doi.org/10.1145/3428204

[50] K. Martin, I.-Y. Bang, J.-S. You, J.-W. Seo, and Y.-H. Paek, "A Study on Security Issues Due to Foreign Function Interface in Rust," *Proceedings of the Korea Information Processing Society Conference*, pp. 151–154, 2021, publisher: Korea Information Processing Society. [Online]. Available: https://koreascience.kr/article/CFKO202125036382349.page

[51] A. N. Evans, B. Campbell, and M. L. Soffa, "Is rust used safely by software developers?" in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ser. ICSE '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 246–257. [Online]. Available: https://doi.org/10.1145/3377811.3380413

[52] H. Nina, J. A. Pow-Sang, and M. Villavicencio, "Systematic mapping of the literature on secure software development," *IEEE Access*, vol. 9, pp. 36 852–36 867, 2021.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[53] H. Xu, Z. Chen, M. Sun, Y. Zhou, and M. R. Lyu, "Memory-Safety Challenge Considered Solved? An In-Depth Study with All Rust CVEs," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 31, no. 1, sep 2021, accessed: July 16, 2023. [Online]. Available: https://doi.org/10.1145/3466642

[54] The Rust Security Response WG, "Security advisory for rustc (CVE-2021-42574)," November 2021, accessed: July 16, 2023. [Online]. Available: https://blog.rust-lang.org/2022/01/20/cve-2022-21658.html

[55] ——, "Security advisory for the standard library (CVE-2022-21658)," January 2022, accessed: July 16, 2023. [Online]. Available: https://blog.rust-lang.org/2022/01/20/cve-2022-21658.html

[56] ——, "Security advisory for the regex crate (CVE-2022-24713)," March 2022, accessed: July 16, 2023. [Online]. Available: https://blog.rust-lang.org/2022/03/08/cve-2022-24713.html

[57] ——, "Security advisories for Cargo (CVE-2022-36113, CVE-2022-36114)," September 2022, accessed: July 16, 2023. [Online]. Available: https://blog.rust-lang.org/2022/09/14/cargo-cves.html

[58] ——, "Security advisory for Cargo (CVE-2022-46176)," January 2023, accessed: July 16, 2023. [Online]. Available: https://blog.rust-lang.org/2023/01/10/cve-2022-46176.html

[59] ——, "Security advisory for Cargo (CVE-2023-38497)," August 2023, accessed: June 10, 2024. [Online]. Available: https://blog.rust-lang.org/2023/08/03/cve-2023-38497.html

[60] ——, "Security advisory for Cargo (CVE-2024-24576)," April 2024, accessed: June 10, 2024. [Online]. Available: https://blog.rust-lang.org/2024/04/09/cve-2024-24576.html

[61] The Rust Project Developers, "rust-clippy: A bunch of lints to catch common mistakes and improve your rust code," https://github.com/rust-lang/rust-clippy, 2024, rust language. [Online]. Available: https://github.com/rust-lang/rust-clippy

[62] Z. Li, J. Wang, M. Sun, and J. C. Lui, "Mirchecker: Detecting bugs in rust programs via static analysis," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2183–2196. [Online]. Available: https://doi.org/10.1145/3460120.3484541

[63] K. R. Fulton, A. Chan, D. Votipka, M. Hicks, and M. L. Mazurek, "Benefits and drawbacks of adopting a secure programming language: Rust as a case study," in *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, Aug. 2021, pp. 597–616. [Online]. Available: https://www.usenix.org/conference/soups2021/presentation/fulton

[64] SonarSource, "SonarQube," https://www.sonarqube.org, [retrieved July 16, 2023]. [Online]. Available: https://www.sonarqube.org

[65] P. C. van Oorschot, "Memory errors and memory safety: A look at java and rust," *IEEE Security & Privacy*, vol. 21, no. 3, pp. 62–68, 2023.

[66] J. Jacoby and M. S. Matell, "Three-Point Likert Scales Are Good Enough," *Journal of Marketing Research*, vol. 8, no. 4, pp. 495–500, 1971, accessed: July 16, 2023. [Online]. Available: https://doi.org/10.1177/002224377100800414

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

ChatIDS: Advancing Explainable Cybersecurity Using Generative AI

Victor Jüttner*, Martin Grimmer†, Erik Buchmann‡
Dept. of Computer Science, Leipzig University, Germany
Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, Germany
Email: *juettner@informatik.uni-leipzig.de,
†grimmer@informatik.uni-leipzig.de,
‡buchmann@informatik.uni-leipzig.de

***Abstract*—An intrusion detection system (IDS) is a proven approach to securing networks. Network-based IDS solutions are typically installed on routers or Internet gateways. They can inspect all incoming and outgoing network traffic, compare network packet signatures against a database of suspicious signatures, or use artificial intelligence. If the IDS identifies a network connection as suspicious, it sends an alert to the user. However, on a home network, it is difficult for users without cybersecurity expertise to understand IDS alerts, distinguish cyberattacks from false alarms, and take appropriate action in a timely manner. This puts the security of home networks, smart home installations, home office workers, etc. at risk, even if an IDS is properly installed and configured. In this work, we propose ChatIDS, our approach to explain IDS alerts to non-experts using large language models. We evaluate the feasibility of ChatIDS using ChatGPT and identify open research questions with the help of interdisciplinary experts in artificial intelligence. Potential issues in areas such as trust, privacy, ethics, etc. need to be addressed before ChatIDS can be put into practice. Our results show that ChatIDS has the potential to improve network security by suggesting meaningful security measures from IDS alerts in an intuitive language.**

***Keywords*—Intrusion Detection; ChatGPT; Smart Home.**

I. Introduction

This paper is an extended version of our previous short paper [1]; the current paper extends that work by providing additional use cases, experiments, and analysis.

In recent years, private networks have come into the focus of cyberattacks. Reasons for this include the increased use of home-office work models [2], a shift to private areas during pandemics [3], or the proliferation of smart home devices [4]. IDSs are a well-established approach to detecting and mitigating cyberattacks [5], [6]. An IDS scans the network and/or network devices and sends alerts about suspicious network activity. This allows its users to detect cyberattacks at an early stage, possibly before any damage is done. On the other hand, an IDS might generate numerous false alarms.

In industry, business, and government, IDSs are a critical line of defense in the cybersecurity infrastructure. To this end, these sectors employ well-trained cybersecurity experts to configure, manage, and maintain IDSs, continuously improve the IDS rule set, distinguish false positives from real attacks, and design, prioritize, and implement appropriate countermeasures. It is possible to pre-configure a network-based IDS for home networks [7]. However, without a solid background in cybersecurity, it is difficult for a home user to interpret IDS alerts such as `MALWARE-CNC Harakit botnet traffic`, distinguish false alerts from real attacks, and develop appropriate and timely countermeasures. As cyberattacks, IDS configurations, and network traffic evolve rapidly, static explanations of known cyberattacks [8] cannot easily replace cybersecurity expertise.

In this paper, we outline ChatIDS, our approach to having a large language model (LLM)—a generative artificial intelligence approach—explain IDS alerts and suggest countermeasures in an intuitive, non-technical way to users without cybersecurity knowledge. ChatIDS sends anonymized IDS alerts to an LLM and allows the user to ask questions if the generated texts are not yet understandable. We make four contributions:

- We specify the requirements for an approach that increases network security in private networks by explaining IDS alerts to a non-expert.
- We describe ChatIDS, our approach to having ChatGPT [9] explain alerts from Snort [10], Suricata [11], and Zeek [12]. The explanations include cybersecurity actions and guidance on why/when to take the actions.
- We evaluate the feasibility of this approach through a small series of experiments with typical IDS alerts.
- To explore the design space of ChatIDS, we had interdisciplinary AI experts identify questions that need to be researched before ChatIDS can be put into practice.

Our experiments show that ChatIDS is easy to implement, although more work needs to be done on prompt engineering to ensure intuitive explanations on the first try. It is difficult to measure whether ChatIDS actually increases network security because it depends on the user. Our interdisciplinary experts have provided valuable insights. For example, from an ethical point of view, it is important to prevent the user from becoming too dependent on the technology, e.g., if ChatIDS allows security incidents to be repaired without the user having to acquire knowledge.

Paper Structure: Section II introduces related work. Section III outlines ChatIDS, our approach to explaining IDS messages to non-experts. Section IV describes a series of experiments to prove feasibility, and Section V identifies open research questions. Finally, Section VI concludes.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

(a) Smart bulb (b) Smartphone application

Figure 1. Smart bulb and control application

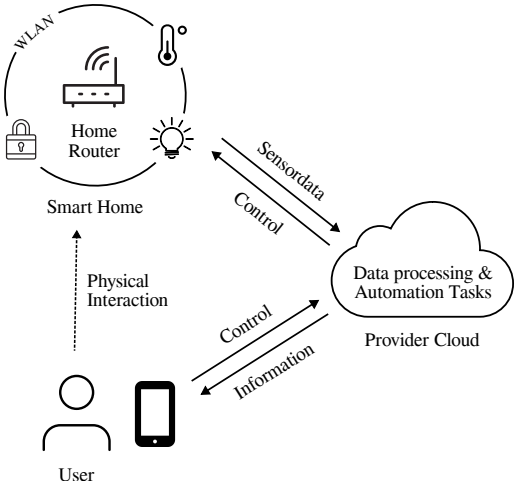


Figure 2. Smart Device Ecosystem

II. Related Work

In this section, we present related work on the smart home, network security, general warning message design, and generative AI models.

A. Smart Home

Home Automation transforms living spaces into dynamic, responsive environments for comfort and efficiency. Popular examples include climate control, home security, lighting or entertainment systems [13]. Typically, the installation of home automation systems is expensive and requires experts [14]. The **smart home** simplifies the process of home automation [13]. For example, let's consider a smart lighting system that can be controlled remotely, adjusting brightness and color tone, and automatically turning on in the morning or simulating a sunset at night. To install this system, all you need to do is replace the regular light bulbs with smart bulbs (Figure 1a), connect them to the Internet router, and install an app on the homeowner's smartphone. The smartphone becomes an remote control (fig. 1b) that can be used to control and configure all desired lighting. Typically, a smart home consists of several such smart devices [15].

Most smart devices depend on a sophisticated **IT ecosystem** that communicates with various external parties over the Internet, as shown in Figure 2. In particular, a smart device establishes Internet-based communication to enable functionalities such as remote control, automation tasks, multimedia services, interaction with other smart devices, cloud services, voice assistants, or software updates [15]. As a consequence, smart home devices are typically **accessible over the Internet**, perhaps through other devices. This distinguishes smart home devices from consumer devices such as laptops or smartphones, which use the Internet as a client, but do not listen for connections from the Internet. Because of this, smart home devices can be victims of unsolicited communication attempts that can be exploited for cyberattacks.

B. Network Security

IDSs monitor a system for unauthorized or suspicious activity and can be distinguished by system type and detection type. The **system type** can be *host-based*, to monitor a single device, or *network-based* to monitor a network of devices. Host-based IDSs work by analyzing information in the local host's logs, which can be a combination of audit, system, and application and system call logs. Network-based IDSs are often centrally located to monitor multiple devices at the network level, they analyze network traffic by inspecting features of the traffic flow and network protocols [16].

Detection types can be *anomaly-detection* or *misuse-detection*. *Anomaly-detection* models the regular behavior of a system and detects activity that is significantly different from this behavior [16]. Anomaly detection systems calculate a score that is compared to an expert defined threshold, and if the score exceeds the threshold, an alert is sent. By adjusting the threshold, the sensitivity of the system can be set. There are detection engines based on Machine Learning [17], Deep Learning [18], Genetic Algorithms [19] and many more [20].

Misuse-detection, which models anomalous behavior and detects malicious patterns by comparing them to a predefined set of rules. Misuse-detection systems send an alert when a known pattern is identified [16]. Popular examples of network-based misuse-detection IDSs are Snort [10], Suricata [11] and Zeek [12]. A ruleset is required to use these IDSs. Popular predefined rulesets for networks are snort3-community-rules [21], suricata-rules [22], Yara [23] and Sigma [24].

C. Warning Message Design

In general, **warning messages** are designed to warn people before a possible harm occurs. However, warning messages must be interpreted and understood. Thus, the effect of warning messages can be unreliable, and other measures should be exhausted before sending a warning message [25]. Non-experts do not always comply with the advice of warning messages. There are three key reasons for this behavior: (1) non-experts do not fully understand warning messages, (2)

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

they do not always trust warning messagess, and (3) they think that compliance will cost them [26].

To increase the **effect of warning messages** in the field of cybersecurity, a developer might be tempted to design warning messages that try to make users fear cyberattacks. This approach has proven to be ineffective [27], [28]. Current recommendations say, that **good warning messages** should be *brief* [29], use *nontechnical language* [29], [30], describe the *risk* [29], describe the *consequences* of notcompliance [29], describe how the cyberattack will *affect* the user personally [31], [32], provide *instructions* on how to avoid the risk [29] and do so in a way that *aligns with how the user thinks* about cyberattacks [31].

D. Generative AI

Generative modeling strives to create models that are capable of generating new data, such as sound, text, or images, that are similar to the data on which the model was trained [33]. Popular examples of generative models are WaveNet [34], which can generate speech and music, Pix2Pix, which can transform images into different styles [35], or GPT-3, an LLM that allows the generation of human-like text [36]. Another example of an LLM is ChatGPT [9]. Like a chatbot, ChatGPT is conversational and can generate detailed answers to questions. Bard [37] follows a similar approach. There are generative models trained for cybersecurity problems, such as Microsoft Security Copilot [38], but these are aimed at experts and therefore not suitable for our purpose.

ChatGPT's reliability varies across domains, it shows high levels of accuracy in recreation and technology domains but struggles with science and law. Problems that reduce the accuracy of ChatGPT are false information, bias, and hallucinations [39].

ChatGPT and LLMs in general are capable of generating text that appears natural and to be grounded in the real context, but is unfaithful and nonsensical. This is called *hallucinated text*, and much like psychological hallucinations, it can be difficult to distinguish from real perception [40].

Prompts are the input to a generative model, they can be a text or an image that gives the model instructions for the requested output. Prompts provide an intuitive way to interact with generative models [41]. For image generation, a prompt can be another image or a text description. For LLMs, a prompt is text that provides context for the desired output, such as a question or a command to summarize information.

Prompt Engineering deals with optimizing prompts to achieve better responses from LLMs. For recurring problems, design patterns can be used to construct prompts and optimize the output, analogous to software patterns [42]. For example, the *Persona Pattern* lets the LLM take on a specific role. This can be useful if the LLM should respond in a special way. If the output must follow a structure, a *Template* can be given in the prompt. The *Context Manager* pattern allows the user to provide or remove context from a prompt.

III. ChatIDS: Explainable Security

In this section we describe ChatIDS, our approach to explaining IDS messages to non-experts. Our goal is to integrate a network-based IDS into private networks to protect the network against cyberattacks from the Internet. Therefore, we want to replace the lack of cybersecurity expertise with an LLM. The LLM transforms and enhances the alerts of an IDS so that a private user can understand them and take appropriate action. For this purpose, we distinguish two roles:

An ***expert*** has the cybersecurity expertise necessary to operate and maintain an IDS, to understand its alarms, and respond to alarms with appropriate and timely actions.

A ***user*** lacks this kind of expertise. A user may follow manuals written without technical vocabulary. It is difficult for a user to determine whether an IDS alert is due to a real attack or a false positive by the IDS, and to act accordingly.

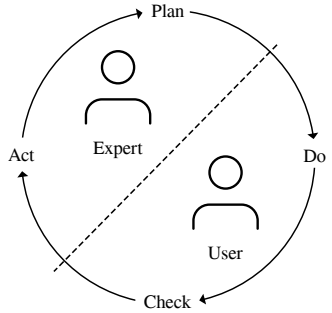


Figure 3. Adapted IT-Security Process

The IT-Security process follows a classic **Plan-Do-Check-Act** cycle [43]. A smart home IDS [7] can be integrated into such a process [44] as follows:

In the *Plan* phase, an expert preconfigures an IDS for typical smart homes. In the *Do* phase, the user installs the preconfigured IDS, which inspects network traffic for potential attacks in the *Check* phase. In the *Act* phase, an expert reviews logs to adapt the IDS for further attacks. Figure 3 illustrates this.

However, without knowledge of cybersecurity the user is left in the *Check* phase with only three possible actions: (a) do nothing, (b) turn off the device that may be under attack, or (c) ask an expert for help. Our ChatIDS approach strives to provide intuitive and understandable explanations of IDS alerts to give users a wider range of appropriate security measures.

A. Requirements

Therefore, ChatIDS must meet three requirements:

R1: (Errors) The user must assess the probability that the IDS has sent a false alert. For example, the IDS might have detected by mistake an attack that is impossible on the device.

R2: (Urgency) The user must assess the urgency of the alert, i.e., whether or not immediate action is required.

R3: (Actions) The user must identify appropriate actions, such as performing a factory reset and installing a security patch.

To explore the solution space for a generative AI approach that fulfills these requirements for IDS, we use a constructive

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

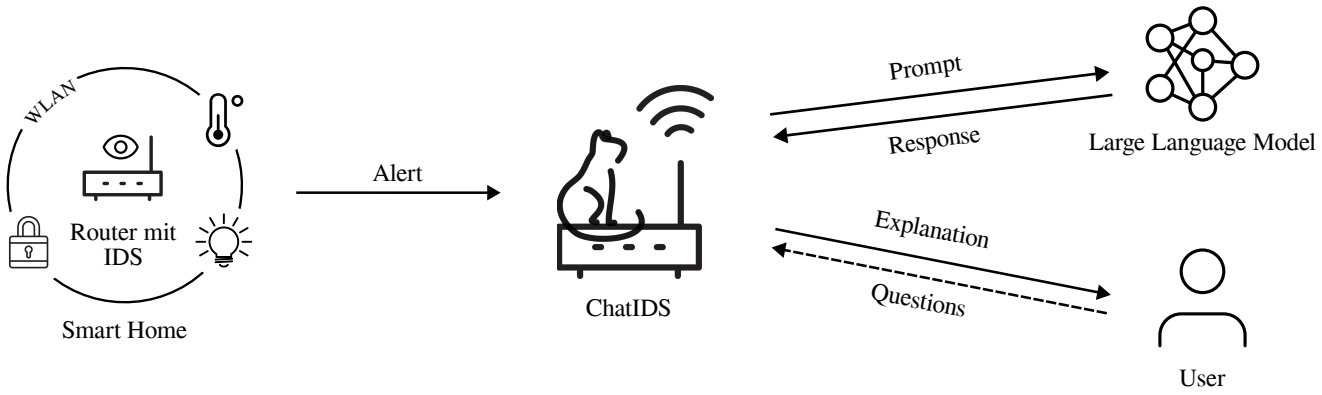


Figure 4. ChatIDS Workflow

research method. In particular, we (a) model ChatIDS, (b) use it to evaluate its technical feasibility, and (c) to discuss potential problems with interdisciplinary AI experts.

B. Our ChatIDS Approach

The ChatIDS workflow is illustrated in Figure 4. A **network-based IDS component** inspects network packets passing through a router for suspicious traffic and generates alerts. The IDS should be a signature-based IDS so that its alerts are specific enough for the LLM.

The **LLM component** is responsible for translating the alerts from the IDS into a language that a non-expert can understand. Furthermore, this component can be used in an interactive way: If users do not understand the explanation or the suggested actions, they can ask for details. Similar to the IDS, the LLM is an external component of ChatIDS.

The **ChatIDS component** is the core of our approach. ChatIDS accepts alerts from the IDS component, sends them to the LLM component for a translation into an intuitive explanation, and presents a user interface with the explanations to the user. If the user requires further support, they can use the interface to send follow-up questions to the LLM. To translate alerts into intuitive explanations, the ChatIDS component contains predefined templates for LLM prompts.

For privacy reasons, the alerts are anonymized in three ways before being sent to the LLM component: First, ChatIDS removes any device identifiers or network information from the alert. Second, ChatIDS sends the anonymized alert to the LLM component along with a set of dummy alerts, so that the LLM component cannot learn the real alert with certainty. The explanations from the LLM component are cached so that the same explanation does not have to be requested twice.

IV. Experimental Evaluation

We evaluate ChatIDS with 20 selected cyberattack alerts to find out how well it meets our requirements R1-R3.

A. Experimental Setup

In line with Figure 4, we assume a home network with several smart home devices. A router connects the network to the Internet and can monitor all network packets. We assume that a Philips Hue Bridge [45] is being attacked. To implement the network-based IDS it is installed on the router and runs either the Snort [21], Suricata [22], Yara [23], or Sigma [24] ruleset. From each IDS implementation, we experiment with 20 alerts, as shown in the first column of Table I. All alerts are classified as important, and a user intervention is required.

The alerts generated are influenced by the target device, the user, and the alert. In our evaluation, we only set the alert as a variable to demonstrate variation and allow for comparability of our proof of concept.

> The Intrusion Detection System in a home network has detected an intrusion and sent out the alert [ALERT MSG]. Your job is to inform [USER] about the alert in a warning message. You're in the role of a cybersecurity expert that interprets the alert and explains the alert in a warning message to [USER]. Your goal is to inform [USER] about the intrusion in a way he understands and motivate the user to take steps to stop the intrusion. [USER] has no cybersecurity expertise and won't understand technical instructions, you need to provide clear, easy and non-technical instructions to follow. Don't use technical terms like "two-factor-authentication", "Intrusion Detection System", "intrusion" or "unassigned message", use simple non-technical terms instead. Don't use the term "Intrusion Detection System". Your explanation and instructions have to align with how [USER] thinks about cyberattacks. The Smart Home consists of several Smart Home Devices, the Intrusion Detection System has detected the intrusion on [DEVICE]. The warning message has to follow this order: Explain the intrusion, explain the potential consequences for the user if he won't comply with the warning message and give instructions on how to stop the intrusion in an itemized list.

Figure 5. Template for a ChatGPT Prompt

We realized ChatIDS using ChatGPT (gpt-3.5-turbo) [9]. To generate an explanation, ChatIDS embeds each alert into a ChatGPT prompt, as shown in Figure 5. This prompt implements the "Template", "Persona", and "Context Manager" patterns, as explained in Section II. Since the training data for ChatGPT comes from the Internet and much of the text on the Internet deals with security issues, we do not expect ChatIDS to produce hallucinations or erroneous output.

B. Evaluation

Figure 6 shows an example of the output generated by ChatIDS. For better understanding, the figure shows the non-anonymized output, which includes names and devices. We

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE I. EVALUATION OF ALERTS AND RESPONSES

Alert	Description	Intuition	Consequences	Urgency	Countermeasures	Correctness
MALWARE-CNC Harakit botnet traffic	2	1	2	1	1	-1
SERVER-WEBAPP NetGear router default password login attempt admin/password	2	2	1	1	-2	1
PROTOCOL-ICMP TFN Probe2	2	2	2	2	1	0
PROTOCOL-FTP Bad login	2	2	2	1	-1	1
SERVER-OTHER SSH server banner overflow	1	2	1	2	0	-1
SURICATA MQTT unassigned message type (0 or >15)	0	2	2	2	0	1
SURICATA HTTP Response abnormal chunked for transfer-encoding	0	-1	2	0	0	-1
SURICATA SSH too long banner	2	2	2	2	2	0
SURICATA FTP Request command too long	1	2	2	2	1	1
SURICATA HTTP invalid content length field in request	0	-1	2	2	2	1
Mirai Botnet TR-069 Worm - Generic Architecture	1	0	2	2	-1	-1
Linux.IotReaper	2	2	2	2	-1	-1
BleedingLife2 Exploit Kit Detection	1	2	2	2	0	2
Weevely Webshell - Generic Rule - heavily scrambled tiny web shell	2	2	1	1	1	1
Mirage Identifying Strings	1	2	2	1	0	0
(Zeek) Identifies IPs performing DNS lookups associated with common Tor proxies.	0	1	0	1	-1	-1
Ensure that all account usernames and authentication credentials are transmitted across networks using encrypted channels.	-2	1	1	2	-1	1
Identifies clients that may be performing DNS lookups associated with common currency mining pools	1	1	2	1	-1	0
Detects URL pattern used by iOS Implant	2	-1	2	1	-2	-1
Detects a bash contecting to a remote IP address	2	-1	2	1	-2	-1

have tested the prompts from the first column of the table I with ChatIDS, one by one. To evaluate their output according to our three requirements **R1: (Errors)**, **R2: (Urgency)**, and **R3: (Actions)**, we used a qualitative approach. We rated the degree of fulfillment of each requirement on a 5-point Likert [46] scale from strongly agree (2) to strongly disagree (-2).

For each requirement, we consider two different features: For **R1**, we evaluate the quality of the generated *Description* and how *Intuitive* it is. The *Description* should provide an accurate description of the attack, while the *Intuition* is considered good if the message is written in accessible language without technical vocabulary. For **R2** we evaluate the description of the *Consequences* for ignoring the alert and the *Urgency* of the description. The *Consequences* of an attack should be explained objectively, taking into account the attacked device, the home network, and the user's privacy. To convey a sense of *Urgency* and motivate the user to act, the message should be clear and concise. For **R3**, we look for a *Correct* and meaningful description of *Countermeasures*. A user should be able to counter the attack with the countermeasures provided by the message and not need further instructions. We also evaluate the correctness of the explanation. The countermeasures provided by the message should resolve all problems on the compromised device and on the home network.

C. Results

Columns 2-7 in Table I list the results of the evaluation of these features. The table shows that ChatIDS fulfilled R1 (Errors) rather well. With only one exception, ChatIDS produced a good description of the security issue that caused the alert. The terminology used in the generated messages is almost always intuitive, with only a few exceptions. R2 (Urgency) was fulfilled even better. ChatIDS described the consequences of inaction and conveyed a sense of urgency almost always to our complete satisfaction. R3 (Errors) leaves room for improvement. We were surprised to find that the description of countermeasures was often too general to be followed by a user without further instruction. In addition, the recommended countermeasures wouldn't completely eliminate the threat described by the IDS alert and are therefore incorrect.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Subject: Important Alert - Your Philips Hue Bridge is at Risk

Dear Jon,

I hope you're well. I have some concerning news about your Smart Home, specifically the Philips Hue Bridge. Our system detected something unusual, and I want to make sure you understand the situation and how to address it.

1. What Happened?
The alert we received mentions "SURICATA SSH too long banner." This means someone is trying to access your Philips Hue Bridge in a way that's not typical or authorized. It's like someone trying to open a window to your house without your permission.

2. Potential Consequences:
If we don't address this intrusion, the person trying to access your Smart Home could potentially:
- Change your lighting settings or turn off lights when you don't want them to.
- Access personal data or settings related to your lights and Smart Home devices.
- Create a security risk for your home network, potentially leading to other issues.

Now, here's what you can do to stop this intrusion and secure your Smart Home:

Instructions:

A. Update Your Philips Hue Bridge:
- Go to the Philips Hue app on your smartphone.
- Check for any available updates for your Hue Bridge and follow the on-screen instructions to install them. Updates often contain important security fixes.

B. Change Your Password:
- Open the Philips Hue app.
- Find the settings or account section.
- Look for the password or security settings and change your password to something strong and unique.
- Avoid using common passwords, like "123456" or "password."

C. Check for Suspicious Devices:
- Open the Philips Hue app.
- Look for the "Connected Devices" or "Authorized Devices" section.
- Review the list and make sure you recognize all the devices connected to your Hue Bridge. If you see any unfamiliar devices, remove them.

D. Monitor Activity:
- Keep an eye on the activity of your Philips Hue Bridge in the app.
- If you notice any unexpected changes or activities, such as lights turning on or off by themselves, take it seriously.

These steps should help you secure your Philips Hue Bridge and prevent any further unauthorized access to your Smart Home. If you have any questions or need further assistance, please don't hesitate to reach out.

Your Smart Home security is important, so let's work together to keep it safe. Stay vigilant, and don't hesitate to take action. Thank you for your attention to this matter.

Best regards,

ChatIDS

Figure 6. Example for an Explanation

V. Open Research Issues

In this section, we outline promising directions for future research. We are interested in warnings that address different audiences, we are interested in practical implementation, and we want to compile open questions for interdisciplinary research.

A. Prompt Design for Different Target Groups

We know from related work (cf. Subsection. II-C) that warning messages are more effective when they are tailored to the recipient's information needs and mindset. For example, a suspicious person may ignore a message that emphasizes the potential damage, while a confident person needs such a warning to spur action. An experienced user may be frustrated by a message that oversimplifies technical details.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

One promising way to address this issue is to customize the LLM prompts for different cultural backgrounds, skill levels, and protection requirements. Our ChatIDS approach makes it easy to integrate multiple pre-defined templates for LLM prompts, so that warning messages can be targeted to different audiences. We ran a series of preliminary experiments with different templates for the same IDS alert. For illustration, Figures 8 and 9 in the Appendix contain two different warning messages. Both of them translate the IDS alert "SURICATA SSH too long banner". Figure 8 addresses users with some technical expertise, while Figure 9 is aimed at reducing anxiety.

B. Implementation and Design

As ChatIDS addresses a practical cybersecurity challenge, implementation aspects such as system architectures or user interfaces need to be considered. We have implemented an interface from ChatIDS to the open source home automation software Home Assistant [47]. Figure 7 shows what a warning message from Home Assistant looks like on a Raspberry Pi. To generate the warning message, we used a ChatIDS template for a very short and simple message.

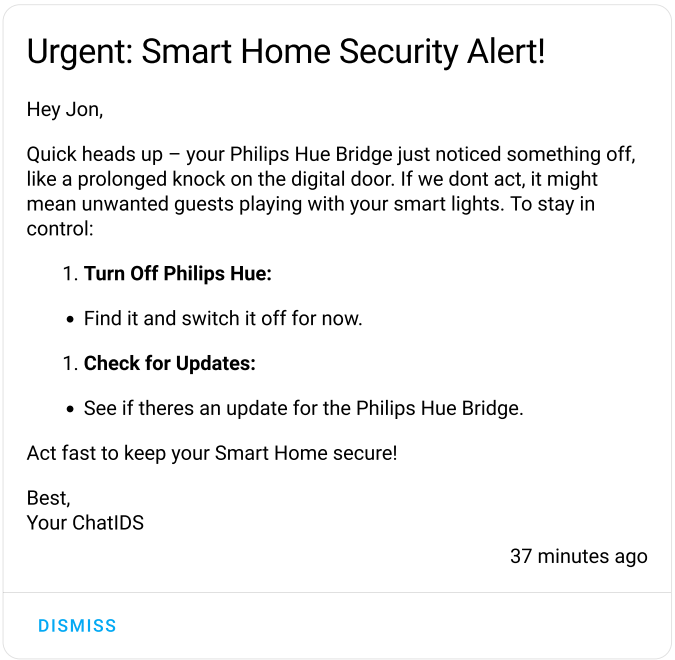


Figure 7. Short Warning Message displayed by Home Assistant

C. Interdisciplinary Research

We have conducted a pre-study with experts from the Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig to compile open issues for interdisciplinary research. Our AI experts cover the topics applications, cybersecurity, ethics, jurisprudence, and privacy. We presented our ChatIDS approach, asked for potential problems, and consolidated the answers. Furthermore, we demonstrate different showcases on how to vary warning messages and aim them at specific user groups. Our experts identified challenges from 6 areas:

Security: ChatIDS potentially increases network security, compared to a scenario where a non-expert is left alone with the alert. However, an external LLM can be a new attack surface, and incorrect or incomprehensible explanations might lead to inappropriate actions.

Privacy: With ChatIDS, the LLM learns that a cyberattack may have occurred on a particular network. Anonymizing device IDs and sending dummy alerts still allows the LLM to infer some information, e.g., if none of the (dummy) alerts sent to the LLM is possible for a particular type of device.

Compliance: ChatIDS has an impact on cybersecurity. However, it is unclear yet, how to conduct a risk analysis on LLMs and on components building upon these, how to evaluate and mitigate associated risks, and to integrate ChatIDS into security frameworks such as the Common Criteria [48].

Jurisprudence: If an alert is not explained well enough, the network could be successfully attacked. Conversely, ChatIDS could convince the user to take action upon false alerts. This creates legal issues. Do special liabilities exist, e.g., from user expectations into a superior AI? How to prove that a harm was caused by a misconducting or negligent AI engineer?

Trust: Users might have a non-rational view on AI approaches, and could fear that a persuasive, non-human intelligence plots against their interests. Conversely, if a user trusts ChatIDS too much, false alerts might result in false actions.

Ethics: ChatIDS could provide explanations that are not only convincing, but manipulative, even if this is in the interest of the user. This raises ethical and moral questions. How drastic can explanations be formulated to induce them to take action (which may even be harmful due to a false positive)? At what point does this limit the autonomy of the user?

VI. Conclusion

This paper describes our ongoing work on ChatIDS, our approach to explaining alerts from an IDS to non-experts. ChatIDS sends anonymized alerts to ChatGPT, an LLM, to intuitively explain the alert and suggest meaningful countermeasures to cyberattacks. Our experiments show that ChatIDS can be easily implemented, although more work is needed on prompt engineering to ensure intuitive explanations on the first try. Furthermore, it needs to be analyzed whether anonymizing the data could remove relevant context or influence the report. It is difficult to measure whether ChatIDS actually increases network security, as it depends on the user. Our interdisciplinary experts have provided valuable insights. In the future, we will continue to improve ChatIDS in terms of security and privacy, as well as interdisciplinary aspects such as compliance, ethics and trust.

Acknowledgement

We would like to thank Prof. Dr. Birte Platow, Dr. Hermann Diebel-Fischer, and Prof. Dr. Johannes Eichenhofer for their valuable contributions on ethical and legal questions.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

REFERENCES

[1] V. Jüttner, M. Grimmer, and E. Buchmann, "ChatIDS: Explainable cybersecurity using generative AI," in *Proceedings of the 17th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE'23)*, 2023.

[2] N. Vakakis, O. Nikolis, D. Ioannidis, K. Votis, and D. Tzovaras, "Cybersecurity in SMEs: The smart-home/office use case," in *2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, 2019, pp. 1–7.

[3] R. O. Andrade, I. Ortiz-Garcés, and M. Cazares, "Cybersecurity attacks on smart home during covid-19 pandemic," in *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2020, pp. 398–404.

[4] Y. Shah and S. Sengupta, "A survey on classification of cyber-attacks on IoT and IIoT devices," in *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 2020, pp. 0406–0413.

[5] I. Butun, S. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, pp. 266–282, 2013.

[6] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, pp. 1–29, 2014.

[7] C. Haar and E. Buchmann, "Securing smart homes using intrusion detection systems," in *Proceedings of the 14th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE'20)*, 2020.

[8] P. Datta *et al.*, "A user-centric threat model and repository for cyber attacks," in *37th ACM/SIGAPP Symposium on Applied Computing*. Association for Computing Machinery, 2022, pp. 1341–1346.

[9] Introducing ChatGPT. OpenAI. (Accessed on: 18.12.2023). [Online]. Available: https://openai.com/blog/chatgpt

[10] What is Snort? Cisco. (Accessed on: 18.12.2023). [Online]. Available: https://www.snort.org/

[11] Suricata. Open Information Security Foundation. (Accessed on: 18.12.2023). [Online]. Available: https://suricata.io/

[12] The Zeek Project. An open source network security monitoring tool. (Accessed on: 18.12.2023). [Online]. Available: https://zeek.org/

[13] W. A. Jabbar, T. K. Kian, R. M. Ramli, S. N. Zubir, N. S. M. Zamrizaman, M. Balfaqih, V. Shepelev, and S. Alharbi, "Design and fabrication of smart home with internet of things enabled automation system," *IEEE Access*, vol. 7, 2019.

[14] J. Greichen, "Value based home automation for todays' market," *IEEE Transactions on Consumer Electronics*, vol. 38, 1992.

[15] B. K. Sovacool and D. D. Furszyfer Del Rio, "Smart home technologies in Europe: A critical review of concepts, benefits, risks and policies," *Renewable and Sustainable Energy Reviews*, vol. 120, p. 109663, 2020.

[16] T. Verwoerd and R. Hunt, "Intrusion detection techniques and approaches," *Computer communications*, vol. 25, no. 15, pp. 1356–1365, 2002.

[17] H. Xu, Z. Sun, Y. Cao, and H. Bilal, "A data-driven approach for intrusion and anomaly detection using automated machine learning for the internet of things," *Soft Computing*, 2023.

[18] B. Sharma, L. Sharma, C. Lal, and S. Roy, "Anomaly based network intrusion detection for IoT attacks using deep learning technique," *Computers and Electrical Engineering*, vol. 107, 2023.

[19] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, and M. L. Proença, "Network anomaly detection system using genetic algorithm and fuzzy logic," *Expert Systems with Applications*, vol. 92, pp. 390–402, 2018.

[20] M. Alkasassbeh and S. Baddar, "Intrusion detection systems: A state-of-the-art taxonomy and survey," *Arabian Journal for Science and Engineering*, vol. 48, 2022.

[21] Snort community rules. Cisco. (Accessed on: 18.12.2023). [Online]. Available: https://www.snort.org/downloads/

[22] Open Information Security Foundation. (Accessed on: 18.12.2023). [Online]. Available: https://github.com/OISF/suricata/tree/master/rules

[23] Yara-rules. YaraRules Project. (Accessed on: 18.12.2023). [Online]. Available: https://github.com/Yara-Rules/rules

[24] Sigma rules. SigmaHQ. (Accessed on: 18.12.2023). [Online]. Available: https://github.com/SigmaHQ/sigma/tree/master/rules

[25] M. Wogalter, "Purposes and scope of warnings," *Handbook of Warnings*, pp. 3–9, 01 2006.

[26] K. S. Jones, N. R. Lodinger, B. P. Widlus, A. Siami Namin, E. Maw, and M. E. Armstrong, "How do non experts think about cyber attack consequences?" *Information & Computer Security*, vol. 30, no. 4, pp. 473–489, 2022.

[27] A. Sasse, "Scaring and bullying people into security won't work," *IEEE Security & Privacy*, vol. 13, no. 3, pp. 80–83, 2015.

[28] M. Dupuis, A. Jennings, and K. Renaud, "Scaring people is not enough: An examination of fear appeals within the context of promoting good password hygiene," in *Proceedings of the 22nd Annual Conference on Information Technology Education*. Association for Computing Machinery, 2021, pp. 35–40.

[29] C. Bravo-Lillo, L. F. Cranor, J. Downs, and S. Komanduri, "Bridging the gap in computer security warnings: A mental model approach," *IEEE Security & Privacy*, vol. 9, pp. 18–26, 2011.

[30] L. Bauer, C. Bravo-Lillo, L. Cranor, and E. Fragkaki, "Warning design guidelines," CyLab, Carnegie Mellon University, Tech. Rep., 2013.

[31] S. Bartsch, M. Volkamer, H. Theuerling, and F. Karayumak, "Contextualized web warnings, and how they cause distrust," in *Trust and Trustworthy Computing: 6th International Conference*. Springer, 2013, pp. 205–222.

[32] M. Kauer, T. Pfeiffer, M. Volkamer, H. Theuerling, and R. Bruder, "It is not about the design - it is about the content! making warnings more efficient by communicating risks appropriately," in *SICHERHEIT 2012 – Sicherheit, Schutz und Zuverlässigkeit*, 2012.

[33] A. Lamb, "A brief introduction to generative models," *Computing Research Repository*, vol. abs/2103.00265, 2021.

[34] Wavenet. Alphabet Inc. (Accessed on: 18.12.2023). [Online]. Available: https://www.deepmind.com/research/highlighted-research/wavenet

[35] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *Computing Research Repository*, vol. abs/1611.07004, 2016.

[36] GPT-3 powers the next generation of apps. OpenAI. (Accessed on: 18.12.2023). [Online]. Available: https://openai.com/blog/gpt-3-apps

[37] Meet Bard. Google Ireland Limited. (Accessed on: 18.12.2023). [Online]. Available: https://bard.google.com/

[38] Introducing Microsoft Security Copilot. Microsoft. (Accessed on: 18.12.2023). [Online]. Available: https://www.microsoft.com

[39] X. Shen, Z. J. Chen, M. Backes, and Y. Zhang, "In ChatGPT we trust? measuring and characterizing the reliability of ChatGPT," *Computing Research Repository*, vol. abs/2304.08979, 2023.

[40] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Comput. Surv.*, vol. 55, 2023.

[41] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba, "Large language models are human-level prompt engineers," *Computing Research Repository*, vol. abs/2211.01910, 2022.

[42] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with ChatGPT," 2023.

[43] J. Eloff and M. Eloff, "Information security architecture," *Computer Fraud & Security*, 2005.

[44] C. Haar and E. Buchmann, "It-security compliance for home offices," in *Proceedings of the 15th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE'21)*, 2021.

[45] Philips Hue Bridge. Signify Holding. (Accessed on: 18.12.2023). [Online]. Available: https://www.philips-hue.com

[46] R. Likert, "A technique for the measurement of attitudes," *Archives of psychology*, 1932.

[47] Home Assistant. Home Assistant Core Team and Community. (Accessed on: 18.12.2023). [Online]. Available: https://www.home-assistant.io/

[48] Management Committee of the CC Recognition Arrangement, "CC:2022 Release 1," https://www.commoncriteriaportal.org/cc/, 2022.

APPENDIX A

WARNING MESSAGE FOR EXPERTS OR ANXIOUS USERS

Figures 8 and 9 show warning messages that were generated with templates tailored for expert users or anxious users.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Subject: Urgent: Security Alert - Intrusion Detected on Philips Hue Bridge

Dear Jon,

I hope this message finds you well. I need to bring an urgent matter to your attention regarding your Smart Home network. Our Intrusion Detection System has flagged an alert with the message "SURICATA SSH too long banner," indicating a potential security breach on your Philips Hue Bridge.

Explanation of the Intrusion:

The alert suggests that an unauthorized attempt has been made to access your Philips Hue Bridge using an SSH connection with an excessively long banner. This could be a sign of a malicious actor attempting to gain unauthorized access to your Smart Home network.

Potential Consequences:

If this intrusion is not addressed promptly, it could lead to the following consequences:

1) Unauthorized access to your Smart Home devices, compromising your privacy.
2) Manipulation or control of your Philips Hue lights and other connected devices.
3) Potential exposure of sensitive information within your network.

Instructions to Stop the Intrusion:

To secure your Smart Home and mitigate the potential risks, please follow these steps:

1) **Isolate Philips Hue Bridge:**
   - Disconnect the Philips Hue Bridge from your network immediately.
   - Power it off to prevent further communication.

2) **Review Device Logs:**
   - Examine the logs on the Philips Hue Bridge for any suspicious activities or login attempts.

3) **Update Firmware:**
   - Check for and apply the latest firmware updates for your Philips Hue Bridge.
   - This ensures that known vulnerabilities are patched.

4) **Change SSH Credentials:**
   - If applicable, change the SSH credentials for your Philips Hue Bridge.
   - Use a strong, unique password to enhance security.

5) **Network Security Check:**
   - Conduct a thorough review of your home network's security settings.
   - Ensure that all devices are protected with strong passwords and up-to-date security measures.

6) **Monitor Activity:**
   - Keep a close eye on your Smart Home devices for any unusual behavior.
   - Report any further suspicious activity to our IT support team.

Please take these steps urgently to safeguard your Smart Home from potential threats. If you have any questions or need assistance, feel free to reach out to our IT support team immediately.

Thank you for your prompt attention to this matter.

Best regards,
ChatIDS

Figure 8. Warning message aimed at expert users

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Subject: Important: Action Required to Secure Your Smart Home

Dear Jon,

I hope this message finds you well. I wanted to bring to your attention a recent notification from your Smart Home setup, specifically concerning the ”Philips Hue Bridge.” No need to worry, but it’s essential to take a few simple steps to ensure everything stays secure.

What Happened:

Our Smart Home system noticed something unusual with the ”Philips Hue Bridge,” which helps control your smart lights. It’s like a little detective that caught a small hiccup.

Potential Consequences (But Nothing to Panic About!):

If we don’t address this now, it could affect the smooth functioning of your smart lights. Think of it as a minor hiccup that might cause some inconvenience, but nothing serious. We want to make sure everything runs seamlessly for you.

Here’s What You Can Do (Super Easy, I Promise!):

- **Power Cycle the Philips Hue Bridge:**
  1) Unplug it from the power source.
  2) Wait for 10 seconds.
  3) Plug it back in.

- **Check for Software Updates:**
  1) Open the app you use for your smart lights.
  2) Look for any updates available for the Philips Hue system.
  3) If there are any, go ahead and update.

- **Secure Your Wi-Fi:**
  1) Ensure your Wi-Fi password is strong and known only to you.
  2) If you haven’t changed it recently, now might be a good time to do so.

Remember, this is just a precaution to keep everything running smoothly. If you have any questions or need further assistance, feel free to reach out.

Best regards,

ChatIDS

Figure 9. Warning message aimed to reduce anxiety

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

A Set of Social Requirements for Self-adaptive Privacy Management Based on Social Groups' Belonging

Angeliki Kitsiou, Maria Sideri, Aikaterini – Georgia Mavroeidi, Katerina Vgena,
Eleni Tzortzaki, Michail Pantelelis, Stavros Simou and Christos Kalloniatis
Privacy Engineering and Social Informatics Laboratory, Department of Cultural Technology and Communication, University of the Aegean, Mytilene, Greece
{a.kitsiou, msid, kmav, kvgena, etzortzaki, mpantel, ssimou, chkallon}@aegean.gr

***Abstract*—This paper examines the privacy representations and privacy management practices of cloud services users that relate to the social group they belong to, through a quantitative survey addressed to the student population of three Universities in Greece, England, and Spain. Findings provide valuable insights regarding social identity-based users' privacy practices and indicate important information for the design of self-adaptive privacy schemes within cloud services, setting specific social requirements based on users' social groups belonging.**

Keywords-adaptive privacy; privacy management; social requirements.

I. Introduction

This paper examines critical issues about users' social groups within cloud services related to their privacy management practices, as an extension of our previous paper presented in IARIA CONGRESS in 13–17 November 2023 in Valencia, concentrating only on self-presentation and self-disclosure practices [1]. Cloud services have significantly expanded in current society, transforming the way individuals and organizations store, access, and manage their data and applications. They often offer integration and interoperability capabilities, allowing different applications and systems to communicate and work together seamlessly, indicating the new notion of the Internet of Cloud [2]. This facilitates the exchange of data and information across platforms, enabling real-time collaboration, sharing, and communication among several team members regardless of their physical locations. Thus, the potential challenges and concerns associated with the expansion of cloud services are immense, such as data privacy and security, vendor lock-in and regulatory compliance [3]. Organizations and individuals should carefully evaluate their specific requirements and consider the appropriate privacy measures and service-level agreements when adopting cloud services [4]. Towards these requirements and measures, the notion of social identity has been indicated as an important factor that influences individuals' privacy preferences and concerns [5]. Social identity refers to the way individuals perceive themselves in relation to various social groups they belong to. The forming of these groups can include factors, such as nationality, ethnicity, gender, religion, profession, or interests [6]. Cloud services provide individuals with opportunities to express and project their social identities to others through profiles, content sharing, and interactions. People often join groups or follow pages related to their social identities, fostering a sense of belonging and connection. In this regard, social identity plays a key role in how individuals present themselves and manage their online image within cloud services [7]. Different social groups may have varying attitudes towards self-disclosure and privacy management practices [8]. However, the nature of self-disclosure on cloud services raises privacy concerns, as individuals need to consider the potential risks associated with sharing personal information publicly [9]. Respectively, the variety of attitudes within cloud services concerns privacy as well, such as prioritizing the protection of personal information or embracing a more open approach. People may strategically disclose or withhold personal information in order to shape their online identity and project a desired image that aligns with their social identity and the desired/intended impression they want to create. They may share personal milestones, hobbies, achievements, opinions, or emotions, while choosing to keep other aspects of themselves and their lives private. Social identity can shape the norms and expectations around privacy within specific social groups. Group members may have shared understandings of what information is appropriate to share, the level of privacy they expect, and the consequences of privacy breaches. These group norms and the values associated with them can shape members' privacy preferences and may influence individuals' privacy management practices and decisions [10].

Privacy management, in this context, involves considering what information to disclose and how it aligns with individuals' social identity and desired impression. Users may employ privacy settings and controls to manage their self-disclosure and control who can access their shared content. Towards this, self-adaptive privacy measures and techniques have been indicated as an effective approach. Self-adaptive privacy in cloud computing refers to the ability of cloud systems to dynamically adjust privacy measures based on specific requirements and preferences of individual users or organizations. It involves tailoring privacy controls, mechanisms, and policies to meet the unique privacy needs of different users and data types [11]. In this regard, self-adaptive privacy aims at empowering users by giving them greater control over their privacy. It provides users with visibility into how their data is being handled within the cloud, offering transparency into privacy practices, and enabling informed decision-making [12]. Considering that privacy management is changing based on users' social groups, several social factors and attributes play a significant role in self-adaptive privacy approaches. These factors influence

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the design, implementation, and acceptance of self-adaptive privacy mechanisms and practices. Thus, as previous research indicates, these factors are usually hard to be identified or are neglected during systems' design [13]. Recent studies have focused on developing algorithmic implementations of such self-privacy adaptation methods that pay attention to users' individual attributes or context [14], [15] and not on groups' norms, while other work concentrates on the user interface mechanism to adopt such adaptations in order to be protected [16].

Therefore, individuals' social attributes should be examined in relation to their social group's belonging [1]. Thus, in this paper, we aim to identify more determinants, based on each social group, of privacy management practices within the cloud. To gather the required data, a survey was conducted among the students of three Universities in Greece, England, and Spain. The findings from this study contribute to valuable insights regarding users' privacy practices based on their belonging to a group and provide important information for the design of usable and self-adaptive privacy features within the cloud, since they promote specific privacy requirements based on users' social identity and groups, considering adaptation on a basis of group privacy management. Section II presents the research field, the methodology followed, and the implemented instrument. In Section III, the results of our survey are outlined, indicating users' privacy management practices. Section IV discusses and concludes the main findings, raising future research directions and practical implications.

II. Methodology

Supporting the arguments above suggesting that social identity pertains to how individuals shape their attitudes and behaviors within various domains of activity [6], the following foundational research question has been formulated to guide our study: RQ *"Is belonging in a social group affecting users' privacy management?"*. To address that, the research population selected for this study included the students of three Universities in Greece, England, and Spain: University of the Aegean, University of Bournemouth, and University of Malaga, respectively. The survey was administered to undergraduate, postgraduate, and doctoral students. Due to its diverse nature in terms of geographical location and demographics, the research population holds significant potential for providing respected insights regarding users' disclosure practices within cloud-based services. It focuses on the domain of social media as the aforementioned cloud environments have been pointed out in the study as the handiest in users' everyday online practices. To ensure access to a substantial portion of the research population and facilitate the generalizability of results [17], a quantitative approach was chosen, and a structured questionnaire was developed. The researchers opted for the Hellenic Statistical Authority's categorizations when determining the values for measuring users' socio-demographics across their survey in order to ensure reliability, representativeness, and transparency. The measurement instrument that was developed, adopted constructs and their respective metrics from both sociological and privacy literature, aiming at examining multiple information about users' social attributes and privacy management within Cloud Services. All items were compiled from previous literature and, in particular, participants were asked to identify the groups to which they belong within cloud services using a social identity taxonomy that aligns with the work of [18]. This taxonomy encompassed a range of group categories, including 15 types of groups, such as leisure groups, well-being groups, professional groups, and other user-indicated groups. Privacy literature was thoroughly investigated in order for the validated metrics of previous works regarding privacy perceptions and management to be adopted in our instrument. Since privacy, apart from the several definitions of its concept; it has specific and very often descriptive and measurable interactive functions within a society, such as privacy concerns, privacy risks, and privacy behaviors, it was important these measures to be incorporated in our instrument. Furthermore, the nature of self-disclosure on cloud services raises privacy concerns, as individuals need to consider the potential risks associated with sharing of personal information publicly. Respectively the variety of attitudes within cloud services concerns privacy as well, such as prioritizing the protection of personal information or embracing a more open approach. Therefore, the questionnaire that was developed for the data collection, included wider sections, concerning users' social identity, users' self-disclosure and privacy management, along with their respective items. For example, in order to ensure the reliability and validity of our instrument, a comprehensive review of the literature for self-presentation and self-disclosure practices was conducted. This review allowed us to incorporate validated metrics from previous studies [19]–[22] on self-presentation and information disclosure into our instrument. These concerned 15 items, as follows: *"I share personal information, I share photos of myself, I share information about my family, I share information about my friends, I share information about my job, I share information about my hobbies, I share information about my daily activities, I share information regarding my sexuality, I share religion-related views, I share information about my political views, I state my location, I update my status, I include contact information (e.g. email, links to other profiles, personal web pages, mobile number, postal address), I have included a short cv in my profile, I tag others in the photos I share"*.

Moreover, the instrument included a set of six questions aiming at capturing participants' socio-demographic characteristics based on previous work [23]. These questions encompassed gender, age, family structure, educational level, professional experience, and monthly income. By incorporating these questions in the final part of the instrument, participants had the time required to complete it more effectively. Prior to distributing the questionnaire to the research population, a pilot study was conducted with a sample of 60 students from the three universities. The purpose of this pilot study was to test the instrument for its form, language, clarity, difficulty level, and responsiveness to respondents' interests, leading to the necessary revisions to the questionnaire items. The survey was conducted using Google Forms, which allowed for direct distribution via email. In the introductory note of

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the survey, the purpose, procedure, and ethical considerations were clearly explained, adhering to established research ethics and standards [24]. The collected data was then recoded and processed using IBM SPSS Statistics 28 (SPSS28).

III. Results

Out of the 368 responses received, thorough checks for completeness were performed, resulting in 280 valid responses being included in the analysis. The survey involved more women than men, while a small percentage declared a different gender. Despite the distribution of ages, the majority was in the age group of 18–32. Regarding family structure, the nuclear form dominates, while it is quite interesting that some of the responders preferred not to provide an answer. Most of the participants held a Master's diploma, and 92% of the respondents have professional experience of at least 1–5 years. The majority declared a relatively low monthly income, ranging from 301 to 800€. Participants' individual attributes, presented in detail in Table I, are associated with their level of social capital [25], setting the standard for a better understanding of users' self-categorization procedure in order to formulate their social identity and define their perceptions and willingness to belong to a social group.

The findings of our survey indicate that participants declare belonging to various social groups when adopting cloud services, namely: Companionship group (33.9%), Professional group (11.3%), Political group (3.1%), Trade union group (2.4%), Voluntary group (8.1%), Sport group (7.7%), Leisure group (11.7%), Cultural group (5.9%), Human Support group (1.5%), Scientific group (2.9%), Environmental group (2.3%), Mutual Support group (1.1%), Religious group (2.0%), Technological Interest group (3.1%) and Gender equality group (3.2%). Previous research has already suggested that individuals who possess multiple social identities are shaping their behaviors, respectively, within specific contexts [26].

Moreover, in order to check if participation in a specific group is associated to participants' stated reasons for social media and cloud services usage, chi-square test for two nominal variables was used. Statistically significant results are shown in Table II. According to this table, there is an association between the variables of companionship, professional, voluntary, sport, leisure, cultural and scientific groups, and specific reasons of use. In all other cases of groups (political, trade union group, human support, environmental, mutual support, religious, technological interest and gender equality group) no statistically significant results came up. Considering that ϕ_c (Phi) takes values between 0 and +/-1, the strength of association of the nominal by nominal relationships is positive in all cases, although low (from 0.129 to 0.166).

The reasons for using social media and cloud services across different social groups are also differentiated. The practice of presenting oneself on platforms like Instagram, Messenger, and Facebook is significantly associated with companionship group. This indicates that individuals may use these platforms to connect with others and establish relationships.

Professionals are more likely to use social media and cloud services for professional activities, as indicated by the statistically significant associations. This suggests that platforms like Google services and WhatsApp may be used for work-related communication and collaboration. Similar to the professional group, individuals interested in sports and scientific activities also tend to use social media and cloud services for professional purposes. Members of voluntary groups show a significant association with using social media and cloud services for professional activities as well. People in leisure groups use social media to seek emotional relationships, partnerships, and job opportunities, serving as avenues for both personal and professional interactions within the leisure context. Individuals interested in cultural activities tend to use them, not only for professional reasons, but also for political activities.

In this regard and in order to check whether participation in a specific social group is associated with specific self-presentation and information disclosure practices, the chi-square test for two nominal dichotomous variables was used. Results are shown in Table III, as follows.

Results show that there are statistically significant associations between the nominal variables of "*group participation*"and "*self-presentation and information disclosure practices*", highlighting that the group in which one chooses to participate is related to the practices that she/he chooses or

TABLE I. Respondents' demographics.

	Sample Socio-Demographics	
	Value	***Percentage%***
Gender	Male	37.5%
	Female	61.8%
	Other	0.7%
Age	18–32	58.9%
	33–47	28.6%
	>48	12.1%
Family Form	Nuclear Family	61.8%
	Large Family	7.5%
	Single-Parent Family	11.8%
	Other Form	9.3%
	Prefer not answering	9.3%
Educational Level	ICD4	36.8%
	Bachelor	23.2%
	MSc	35.7%
	PhD	3.6%
Professional Experience	1 to 5	43.6%
	6 to 10	17.5%
	11 to 15	9.6%
	16 to 20	8.9%
	21 to 25	6.4%
	>26	5.7%
Monthly Income	301–800€	40.7%
	801–1000€	16.1%
	1001–1500€	20.7%
	1501–2000€	6.1%
	2001–3000€	3.2%

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II. SOCIAL GROUPS AND REASONS FOR SOCIAL MEDIA AND CLOUD SERVICES USAGE.

Groups	Practices	Media & Services *Instagram, Messenger, Facebook, Google services, What's up*
Companionship	Present myself	$\chi^2(1)$=4.869, p=0.027, ϕ_c=0.133
Professional	For professional activities	$\chi^2(1)$=6.936 p=0.008, ϕ_c=0.159
Sport	Look for friendships	$\chi^2(1)$=7.589 p=0.006, ϕ_c=0.166
Scientific	For professional activities	$\chi^2(1)$=6.235 p=0.013, ϕ_c=0.151
Voluntary	For professional activities	$\chi^2(1)$=4.580 p=0.032, ϕ_c=0.129
Leisure	Look for emotional relationship	$\chi^2(1)$=4.911 p=0.027, ϕ_c=0.134
	Look for partnerships	$\chi^2(1)$=6.565 p=0.010, ϕ_c=0.155
	Look for job	*$\chi^2(1)$=4.761 p=0.029, ϕ_c=0.132*
Cultural	For professional activities	$\chi^2(1)$=5.599 p=0.018, ϕ_c=0.143
	For professional activities	$\chi^2(1)$=5.377 p=0.020, ϕ_c=0.140

avoids for self-presentation. Most of the associations were revealed for users' self-presentation and information disclosure practices on Messenger (25 associations) and Instagram (22 associations), less on Facebook (15 associations) and few (1-2) on What's Up and Google services. These results are not surprising, considering that the cumulative percent of participants using "once daily"and "several times daily"Messenger, Instagram and Facebook are, according to the results of the research, high (78.3%, 70.2% and 61.9%, respectively).

The majority of associations were positive with the exception of fifteen (15) negative revealed in the case of participating in specific types of groups (mainly trade-union, professional, technological interest, scientific, voluntary, cultural, environmental) and for specific social media, mostly Instagram and less Messenger. Although the negative associations refer to nine (9) different practices, more negative associations were revealed for practices including photos sharing ("I share photos of myself"and "I tag others in the photos I share") and for practices referring to hobbies and daily activities information sharing. This finding implies that the aforementioned practices are considered rather inappropriate by people participating in professional groups or groups that serve specific interests. Moreover, results revealed that those participating in companionship groups use more self-disclosure practices compared to others participating in other type of groups, which is explicable considering the more open goal of participation and the expected benefits from self-disclosure. Results also revealed that the self-presentation practices more used (or avoided) by people according to the type of group they belong, and the media context, were that of sharing information about hobbies (12 associations, 3 of them negative) and photos sharing of oneself (9 associations, 3 of them negative).

TABLE III. SOCIAL GROUPS' SELF-PRESENTATION AND INFORMATION DISCLOSURE PRACTICES.

Groups	Disclosure Practices	Media & Services *Instagram, Messenger, Facebook, Google services, What's up*
Companionship	Personal information	***Messenger:*** $\chi^2(1)$=6.844, p=0.009, ϕ_c=0.157
	Photos of myself	***Instagram:*** $\chi^2(1)$=11.024, p=0.001, ϕ_c=0.200
		Messenger: $\chi^2(1)$=6.517, p=0.011, ϕ_c=0.154
	About my friends	***Messenger:*** $\chi^2(1)$=3.957, p=0.047, ϕ_c=0.120
	About my job	***Messenger:*** $\chi^2(1)$=5.227, p=0.022, ϕ_c=0.138
	About my hobbies	***Instagram:*** $\chi^2(1)$=10.663, p=0.001, ϕ_c=0.197
		Messenger: $\chi^2(1)$=5.632, p=0.018, ϕ_c=0.143
	About my daily activities	***Instagram:*** $\chi^2(1)$=10.115, p=0.001, ϕ_c=0.191
		Messenger: $\chi^2(1)$=6.479, p=0.011, ϕ_c=0.153
	My location	***Instagram:*** $\chi^2(1)$=4.082, p=0.043, ϕ_c=0.122
	I tag others in the photos I share	***Instagram:*** $\chi^2(1)$=5.520, p=0.019, ϕ_c=0.141
Professional	About my job	***Messenger:*** $\chi^2(1)$=7.917, p=0.005, ϕ_c=0.169
	Religious views	***Messenger:*** $\chi^2(1)$=5.553, p=0.018, ϕ_c=-0.142
	A short cv in my profile	***Instagram:*** $\chi^2(1)$=5.470, p=0.019, ϕ_c=-0.141
	I tag others in the photos I share	***Instagram:*** $\chi^2(1)$=5.549, p=.018, ϕ_c=-0.142
Political	About my family	***Messenger:*** $\chi^2(1)$=4.953, p=0.026, ϕ_c=0.134
	About my friends	***Facebook:*** $\chi^2(1)$=3.936, p=0.047, ϕ_c=0.119
	About my job	***Messenger:*** $\chi^2(1)$=6.415, p=0.011, ϕ_c=0.152
	About my hobbies	***Facebook:*** $\chi^2(1)$=8.561, p=0.003, ϕ_c=0.176
	I tag others in the photos I share	***Facebook:*** $\chi^2(1)$=7.527, p=0.006, ϕ_c=0.165
Technological Interest	Photos of myself	***Instagram:*** $\chi^2(1)$=8.102, p=0.004, ϕ_c=-0.171
	About my hobbies	***Instagram:*** $\chi^2(1)$=4.825, p=0.028, ϕ_c=-0.132
	About my daily activities	***Instagram:*** $\chi^2(1)$=5.751, p=0.016, ϕ_c=-0.144

Continues...

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Furthermore, in order to check if participating in a specific group relates to perceptions about beliefs in privacy rights, privacy concerns, comfortability with information collection, privacy control, attitude towards collaborative privacy management and self-disclosure cost-benefit evaluation, a Mann-Whitney test for to independent samples (those participating in a group vsthose not participating) was used. Kolmogorov-Smirnov test of normality firstly applied didn't show normal distribution for these variables, which is a prerequisite for using a T-test. To run the Mann-Whitney test the total score of the statements included in the variables above has been calculated. The results of Mann Whitney tests are shown in Table IV, revealing statistically significant differences (p <0.05) between those who declared their participation into some of the groups.

Results revealed significant differences in several aspects of privacy-related perceptions among participants in different groups compared to non-participants. Firstly, individuals who participated in the Companionship group exhibited statistically significant differences in their perceptions of privacy control compared to non-participants. Similarly, participants in the Political group showed significant differences in privacy control compared to those not in the group. Moreover, both the Companionship and Political groups displayed significant differences in collaborative privacy management compared to non-participants. This suggests that group participation influences individuals' attitudes towards managing privacy collaboratively. Additionally, individuals associated with the Trade Union group showed significantly different perceptions of privacy control compared to non-participants. In terms of specific interest groups, participants in the Sport group displayed significant differences in collaborative privacy management compared to non-participants. Similarly, individuals in the Cultural group exhibited significant differences in their approach to collaborative privacy management. Lastly, participants in the Gender Equality group had significantly different beliefs in privacy rights compared to non-participants. Furthermore, participants in sport groups overall had significantly different self-disclosure cost-benefit evaluations compared to non-ones.

In order to check also if participation in a group is related to self-protection strategies, chi-square test for two nominal variables was again used. Results are shown in Table V. As revealed there is an association between the variables of companionship, professional, voluntary, leisure, scientific, environmental, religious, technological interest and gender equality group, and specific self-protection strategies. In all other cases of groups (political, trade union group, sport, cultural, human support and mutual support) no statistically significant results came up. The strength of association of the nominal by nominal relationships is positive in 8 cases and negative in 7 (marked in Italics), but low in all cases.

Results indicate that individuals who often adjust their privacy settings are more likely to belong to social groups centered around companionship. This indicates a proactive approach to managing privacy concerns within this context. Participants who do not restrict access to the content they upload are associated with professional social groups. This suggests

TABLE III. SOCIAL GROUPS' SELF-PRESENTATION AND INFORMATION DISCLOSURE PRACTICES (CONT.).

Groups	**Disclosure Practices**	**Media & Services** *Instagram, Messenger, Facebook, Google services, What's up*
Trade Union	Photos of myself	***Instagram:*** $\chi^2(1)$=4.502, p=0.034, ϕ_c=-0.128
	About my hobbies	***Facebook:*** $\chi^2(1)$=6.686, p=0.010, ϕ_c=0.156
		Instagram: $\chi^2(1)$=5.633, p=0.018, ϕ_c=-0.143
	My location	***Instagram:*** $\chi^2(1)$=7.107, p=0.008, ϕ_c=-0.160
	I tag others in the photos I share	***Instagram:*** $\chi^2(1)$=8.209, p=0.004, ϕ_c=-0.172
Gender equality	Personal information	***Messenger:*** $\chi^2(1)$=4.871, p=0.027, ϕ_c=0.133
	About my family	***Messenger:*** $\chi^2(1)$=15.645, p=0.000, ϕ_c=0.238
	About my friends	***Messenger:*** $\chi^2(1)$=9.468, p=0.002, ϕ_c=0.185
	About my daily activities	***Messenger:*** $\chi^2(1)$=5.639, p=0.018, ϕ_c=0.143
	Contact information	***Facebook:*** $\chi^2(1)$=5.563, p=0.018, ϕ_c=0.142
Religious	Information about my hobbies	***Facebook:*** $\chi^2(1)$=5.076, p=0.024, ϕ_c=0.136
Voluntary	Photos of myself	***Instagram:*** $\chi^2(1)$=4.410, p=0.036, ϕ_c=-0.126
		What's up: $\chi^2(1)$=4.226, p=0.040, ϕ_c=0.124
	About my job	***Facebook:*** $\chi^2(1)$=8.503, p=0.004, ϕ_c=0.176
	About my hobbies	***Messenger:*** $\chi^2(1)$=4.735 p=0.030, ϕ_c=0.131
	My daily activities	***Facebook:*** $\chi^2(1)$=4.720, p=0.030, ϕ_c=0.131
	Contact information	***Google services:*** $\chi^2(1)$=3.878, p=0.049, ϕ_c=0.119
	I tag others in the photos I share	***Facebook:*** $\chi^2(1)$=4.268, p=0.039, ϕ_c=0.124
Scientific	About my job	***Facebook:*** $\chi^2(1)$=9.700, p=0.002, ϕ_c=0.187
	About my hobbies	***Instagram:*** $\chi^2(1)$=4.189, p=0.041, ϕ_c=-0.123
	About my daily activities	***Messenger:*** $\chi^2(1)$=4.597, p=0.032, ϕ_c=-0.129

Continues...

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE III. SOCIAL GROUPS' SELF-PRESENTATION AND INFORMATION DISCLOSURE PRACTICES (CONT.).

Groups	Disclosure Practices	Media & Services *Instagram, Messenger, Facebook, Google services, What's up*
Sport	Personal information	***Messenger:*** $\chi^2(1)$=4.467, p=0.035, ϕ_c=0.127
	About my friends	***Instagram:*** $\chi^2(1)$=4.484, p=0.034, ϕ_c=0.127
	About my hobbies	***Facebook:*** $\chi^2(1)$=5.774, p=0.016, ϕ_c=0.145
		Instagram: $\chi^2(1)$=8.501, p=0.004, ϕ_c=0.175
	My daily activities	***Messenger:*** $\chi^2(1)$=5.480, p=0.019, ϕ_c=0.141
	My location	***Instagram:*** $\chi^2(1)$=6.245, p=0.012, ϕ_c=0.150
	I tag others in the photos I share	***Instagram:*** $\chi^2(1)$=4.086, p=0.043, ϕ_c=0.122
Leisure	Personal information	***Google services:*** $\chi^2(1)$=3.972, p=0.046, ϕ_c=0.120
	Photos of myself	***Facebook:*** $\chi^2(1)$=4.667, p=0.031, ϕ_c=0.130
		Instagram: $\chi^2(1)$=4.730, p=0.030, ϕ_c=0.131
	About my hobbies	***Facebook:*** $\chi^2(1)$=7.015, p=0.008, ϕ_c=0.159
	I update my status	***Facebook:*** $\chi^2(1)$=4.634, p=0.031, ϕ_c=0.130
Cultural	About my family	***Messenger:*** $\chi^2(1)$=4.405, p=.0036, ϕ_c=0.126
	About my sexuality	***Messenger:*** $\chi^2(1)$=11.908, p=0.001, ϕ_c=0.208
	Religious views	***Messenger:*** $\chi^2(1)$=9.344, p=0.002, ϕ_c=0.184
	About my political views	***Messenger:*** $\chi^2(1)$=8.041, p=0.005, ϕ_c=0.171
	My location	***Messenger:*** $\chi^2(1)$=8.671, p=0.003, ϕ_c=0.177
	Contact information	***Instagram:*** $\chi^2(1)$=3.863, p=0.049, ϕ_c=-0.118
		Messenger: $\chi^2(1)$=3.888, p=0.049, ϕ_c=0.119
Environmental	Personal information	***Messenger:*** $\chi^2(1)$=4.182, p=0.041, ϕ_c=-0.123
Human Support	Photos of myself	***Facebook:*** $\chi^2(1)$=7.492, p=0.007, ϕ_c=0.164

a willingness to share professional information openly. Those who do not accept friendship requests from strangers are more likely to belong to groups advocating for gender equality. This behavior aligns with cautious online practices regarding social connections. Individuals familiar with platform mechanisms for self-protection tend to belong to religious groups. This indicates a sense of awareness and possibly guidance within religious communities regarding online safety. Members who have left privacy settings at default are linked to voluntary groups. This suggests a lack of awareness or concern about privacy implications within this group. Changing initial privacy settings and adjusting them frequently are common practices among individuals in leisure-oriented groups. Additionally, they tend to consider contextual factors when sharing information, reflecting a balanced approach to privacy management. Moreover, changing initial privacy settings and using limited profile options are prevalent among individuals in scientific groups. This indicates a proactive stance towards safeguarding privacy, possibly influenced by professional or research-related considerations. Usage of limited profile options is associated with environmental groups, suggesting a conscious effort to control the visibility of personal information. Finally, those who frequently adjust privacy settings often belong to groups interested in technology. This behavior may stem from a deeper understanding of online privacy risks and a proactive approach to mitigating them.

IV. DISCUSSION AND CONCLUSION

Our analysis highlights the diverse motivations driving the use of social media and cloud services across different social groups, ranging from personal connections and professional networking to cultural interests and political activities. These findings underscore the multifaceted nature of online engagement and the varying needs of different user demographics. The results suggest that group participation influences various aspects of individuals' perceptions and behaviors related to privacy. Depending on the specific group, individuals may exhibit different attitudes towards privacy control, collaborative privacy management, beliefs in privacy rights, and self-disclosure -cost-benefit evaluation. These findings highlight the importance of considering group dynamics when examining privacy-related behaviors in social contexts. The findings also underscore the influence of group participation on individuals' perceptions and behaviors related to privacy and self-disclosure. As the findings above indicate, social belonging in a group affects users' self-disclosure practices and, respectively, influences their privacy preferences. Self-disclosure on cloud services contributes to users' digital footprints, leaving a trace of their activities, interests, and interactions [27]. Thus, findings highlighted that users who share a similar social identity based on companionship, feel more comfortable disclosing personal information and photos within cloud services and particularly within social media. However, other users emphasizing certain aspects of their identity, mostly the professional based ones, and downplaying the others, declared to be mindful of their social identity presentation and self-disclosure on social media, considering

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE IV. SOCIAL GROUPS' PRIVACY ATTITUDES.

Variables/Groups	Companionship	Political	Trade union	Sport	Cultural	Environmental	Gender equality
Privacy Control	U=1715.000 p=0.006	U=2033.500 p=0.016	U=1491.000 p=0.009				
Collaborative Privacy Management	U=1542.000 p=0.001	U=1828.000 p=0.003			U=4111.000 p=0.039	U=1573.000 p=0.047	
Beliefs In Privacy Rights			U=1591.000 p=0.023				
Self Disclosure Cost Benefit				U=5190.000 p=0.034			

the potential consequences and impacts on their privacy, well-being, and relationships. Evidently, previous research has shown that this digital footprint can have implications for reputation management, online perception, and potential consequences in both personal and professional contexts [28]. What is more the analysis highlights how self-protection strategies vary across different social groups, reflecting varying levels of awareness, concern, and proactive behavior regarding online privacy and security.

In this regard, the identification of social groups' privacy management practices on the cloud can have a significant impact on the design and implementation of self-adaptive privacy schemes, in order for users to be aware of privacy settings, critically evaluate the information shared, and maintain a balance between online and offline identities which can contribute to a more positive and authentic online presence. Considering that social groups' norms serve as guidelines for users and societies to navigate privacy boundaries and expectations, contributing to the preservation of personal autonomy, dignity, and trust [29], the identification of the practices that lead to specific group-based needs is of great importance. Since self-adaptive privacy in cloud services seeks to strike a balance between data utility and privacy protection, by tailoring privacy measures to users' needs and dynamically adapting to changing circumstances [30], users' empowerment can be enhanced when self- adaptive privacy schemes from the beginning of the design take into account groups preferences and the balance between maintaining privacy and participating in social interactions within one's social identity networks. Furthermore, incorporating the understanding of social groups' privacy management practices into the concept of "privacy by design"methodologies, such as the extended PriS framework for cloud computing services [31] that should be used for designing self-adaptive privacy schemes, can help ensure that privacy considerations are embedded in the development process of cloud services.

Despite the limitations of our survey, concerning the weak strength of association of the nominal-by-nominal relationships (ϕ coefficient takes values between 0 and $+/-1$), our results indicate the diversity of privacy management practices across different social groups, providing a guide for specific social requirements that could be integrated from the initial design stages of self-adaptive privacy schemes. It is indicated that the users within Cloud exhibit several key characteristics. Firstly, they are heterogeneous, representing diverse social identities that reflect both their individual norms and their interactions within social networks. Secondly, they are socially interrelated, as they share personal information to gain symbolic benefits and resources from their online networks. Thirdly, they prioritize privacy, holding privacy rights in high regard and expressing significant concerns about privacy when using the services. Despite this, they have a limited level of trust regarding the use of their personal information. Lastly, they demonstrate collaborative behavior, co-managing their personal information with other users and sometimes sharing information about others without their explicit consent.

In this respect, the defining of the privacy management practices can influence the establishment of privacy defaults in cloud platforms. Therefore, the identification of specific social privacy related requirements are presented in the following tables and figure as follows:

Based on the significant associations between group participation and reasons for social media and cloud services usage presented, the social requirements for privacy protection can be identified, as presented in Table VI.

These requirements underscore the importance of context-specific privacy protections that accommodate the diverse reasons for social media and cloud services usage within different groups, ensuring that users can engage in various activities while maintaining control over their personal information. In Figure **??**, the self-disclosure practices are visualized by group and cloud service, aiming to aid the self-adaptive privacy schemes designed to be aligned with the preferences of social groups by setting initial privacy defaults that reflect their common practices and expectations.

Figure 1. Social Requirements for Self-Adaptive Privacy Schemes in Cloud based on Social Groups' self disclosure practices.

Furthermore, considering that privacy control, collabora-

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE V. SOCIAL GROUPS' SELF-PROTECTION STRATEGIES.

Groups	Practices	Media & Services *Instagram, Messenger, Facebook, Google services, What's up*
Companionship	Often adjust privacy settings	$\chi^2(1)$=6.498, p=0.011, ϕ_c=-0.155
Professional	Do not restrict access to the content I upload	$\chi^2(1)$=4.833, p=0.028, ϕ_c=-0.133
Gender equality	Do not accept friendship requests from strangers	$\chi^2(1)$=9.079, p=0.003, ϕ_c=-0.182
	Untag myself from others' photos	$\chi^2(1)$=3.921, p=0.048, ϕ_c=0.120
Religious	Familiar with the mechanisms provided by the platform to protect myself	$\chi^2(1)$=4.732, p=0.030, ϕ_c=0.132
Voluntary	Have let privacy settings at default	$\chi^2(1)$=4.166, p=0.041, ϕ_c=0.124
	Untag myself from others' photos	$\chi^2(1)$=6.121, p=0.013, ϕ_c=-0.150
Leisure	Have changed initial privacy settings	$\chi^2(1)$=5.876, p=0.015, ϕ_c=0.147
	Often adjust privacy settings	$\chi^2(1)$=4.881, p=0.027, ϕ_c=0.134
	Carefully consider the context (where am I) when I provide information	$\chi^2(1)$=5.940, p=0.015, ϕ_c=0.148
Scientific	Have changed initial privacy settings	$\chi^2(1)$=4.947, p=0.026, ϕ_c=-0.135
	Use a limited profile option	$\chi^2(1)$=5.420, p=0.020, ϕ_c=-0.141
	Have excluded contact information from my profile	$\chi^2(1)$=5.200, p=0.023, ϕ_c=-0.138
Environmental	Use a limited profile option	$\chi^2(1)$=3.865, p=0.049, ϕ_c=0.119
Technological Interest	Often adjust privacy settings	$\chi^2(1)$=4.212, p=0.040, ϕ_c=0.124

TABLE VI. SOCIAL REQUIREMENTS FOR SELF-ADAPTIVE PRIVACY SCHEMES IN CLOUD BASED ON SOCIAL GROUPS' REASONS FOR USING SOCIAL MEDIA AND CLOUD SERVICES.

SR	Description
SR 1	Tailor privacy settings and controls to accommodate the diverse reasons for social media and cloud services usage across different groups, ensuring that individuals can present themselves and seek various types of relationships without compromising their privacy.
SR 2	Implement privacy measures that support professional activities on social media platforms, acknowledging the need for privacy while engaging in career-related networking and interactions.
SR 3	Provide privacy features that align with the voluntary nature of group participation, respecting users' autonomy and preferences in sharing information within these contexts.
SR 4	Recognize the privacy needs of individuals engaging in sports-related groups, ensuring that privacy controls enable users to maintain their privacy while participating in sports-related discussions and activities.
SR 5	Develop privacy mechanisms that cater to leisure and cultural group participation, acknowledging the importance of privacy in recreational and cultural exchanges online.
SR 6	Implement privacy measures that support scientific activities and discussions on social media platforms, safeguarding the privacy of individuals engaging in scientific research and collaborations.

TABLE VII. SOCIAL REQUIREMENTS FOR SELF-ADAPTIVE PRIVACY SCHEMES IN CLOUD BASED ON SOCIAL GROUPS' PRIVACY ATTITUDES.

SR	Description
SR 1	Implement mechanisms for collaborative privacy management within online groups to empower users in controlling the information they share about others.
SR 2	Cultivate a culture of respect for privacy rights within online groups, emphasizing the importance of privacy and providing education on privacy-related issues.
SR 3	Enhance users' understanding of the importance of privacy control and of costs and benefits of self-disclosure within online communities, to enable informed decision-making regarding personal information sharing.

tive privacy management, beliefs in privacy rights and self-disclosure cost-benefit evaluation impact on users' disclosure behavior regarding the risks they uptake for themselves and other, as well as that the level of privacy control and collaborative privacy management should be high when participating in social groups in order users to protect themselves and others, we propose the requirements, as presented in Table VII.

Finally, as far as the self-protection strategies concerns, the following requirements are presented in the Table VIII.

Since the insights into social groups' self-disclosure practices can inform the design process, this knowledge can enable in particular the design of contextual privacy settings. These settings can dynamically adjust privacy levels based on the specific context or situation, taking into account groups' preferences in order, for example, to be more restrictive for the information of the professional groups, while more permissive for companionship or leisure groups. Finally, the provided insights into the self-disclosure practices can enhance the transparency and consent mechanisms in the self-adaptive privacy schemes. Users can be provided with clear and understandable information about how their data will be used, shared, and stored on the cloud, allowing them to make informed decisions and providing meaningful consent based on their social group norms. Therefore, users will be provided with control and agency over their information and with respect to their individual privacy preferences, reducing the risk of unintentional oversharing or undersharing.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE VIII. SOCIAL REQUIREMENTS FOR SELF-ADAPTIVE PRIVACY SCHEMES IN CLOUD BASED ON SOCIAL GROUPS' SELF PROTECTION STRATEGIES.

SR	Description
SR 1	Users participating in companionship groups should be able to easily change their initial privacy settings, ensuring control over their online information.
SR 2	Professionals engaging in online communities should have the ability to adjust privacy settings frequently, allowing them to tailor their online presence according to their professional needs and preferences.
SR 3	Volunteers involved in online platforms should be provided with a limited profile option, empowering them to manage their privacy settings effectively while participating in various activities.
SR 4	Individuals engaging in leisure groups should have the option to untag themselves from others' photos easily, granting them control over their online image and associations.
SR 5	Users interested in scientific communities should be familiar with the mechanisms provided by the platform to protect their privacy, enabling them to make informed decisions about their online activities.
SR 6	Environmental enthusiasts should have the capability to exclude contact information from their profile, safeguarding their privacy while actively participating in environmental initiatives.
SR 7	Participants in religious groups should be empowered to carefully consider the context when providing information online, ensuring that their actions align with their religious beliefs and values.
SR 8	Those with technological interests should be able to avoid accepting friendship requests from strangers, enhancing their online security and privacy.

ACKNOWLEDGMENT

The research project was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the '2nd Call for H.F.R.I. Research Projects to support Faculty Members & Researchers' (Project Number: 2550).

REFERENCES

[1] A. Kitsiou, M. Sideri, A.-G. Mavroeidi, K. Vgena, eleni Tzortzaki, M. Pantelelis, S. Simou, and C. Kalloniatis, "Social requirements for designing self-adaptive privacy schemes in cloud," in *The 2023 IARIA Annual Congress on Frontiers in Science, Technology, Services, and Applications*, vol. ISBN:978-1-68558-089-6, IARIA, 2023.

[2] A. Cook, M. Robinson, M. A. Ferrag, L. A. Maglaras, Y. He, K. Jones, and H. Janicke, *Internet of Cloud: Security and Privacy Issues*, pp. 271–301. Springer International Publishing, 2018.

[3] D. Peras and R. Mekovec, "A conceptualization of the privacy concerns of cloud users," *Information & Computer Security*, vol. 30, pp. 653–671, Apr. 2022.

[4] A. Tsouplaki, *Internet of Cloud (IoC): The Need of Raising Privacy and Security Awareness*, pp. 542–550. Springer Nature Switzerland, 2023.

[5] A. Kitsiou, E. Tzortzaki, C. Kalloniatis, and S. Gritzalis, "Chapter 2 - towards an integrated socio-technical approach for designing adaptive privacy aware services in cloud computing," in *Cyber Influence and Cognitive Threats* (V. Benson and J. Mcalaney, eds.), pp. 9–32, Academic Press, 2020.

[6] M. A. Hogg, D. Abrams, and M. B. Brewer, "Social identity: The role of self in group processes and intergroup relations," *Group Processes & Intergroup Relations*, vol. 20, pp. 570–581, Mar. 2017.

[7] H. Erin E., "Self-presentation in social media: Review and research opportunities," *Review of Communication Research*, vol. 9, pp. 80–98, 2021.

[8] K. Vgena, A. Kitsiou, and C. Kalloniatis, "Understanding the role of users' socio-location attributes and their privacy implications on social media," *Information & Computer Security*, vol. 30, pp. 705–729, May 2022.

[9] T. Dienlin, P. K. Masur, and S. Trepte, "A longitudinal analysis of the privacy paradox," *New Media & Society*, vol. 25, pp. 1043–1064, June 2021.

[10] A. Kitsiou, E. Tzortzaki, C. Kalloniatis, and S. Gritzalis, "Identifying privacy related requirements for the design of self-adaptive privacy protections schemes in social networks," *Future Internet*, vol. 13, p. 23, Jan. 2021.

[11] M. Belk, C. Fidas, E. Athanasopoulos, and A. Pitsillides, "Adaptive and personalized privacy and security (apps 2019): Workshop chairs' welcome and organization," in *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, UMAP '19, ACM, June 2019.

[12] B. P. Knijnenburg, "Privacy? I can't even! making a case for user-tailored privacy," *IEEE Security & Privacy*, vol. 15, no. 4, pp. 62–67, 2017.

[13] A. Kitsiou, E. Tzortzaki, C. Kalloniatis, and S. Gritzalis, *Self Adaptive Privacy in Cloud Computing Environments: Identifying the Major Socio-Technical Concepts*, pp. 117–132. Springer International Publishing, 2020.

[14] I. Saini, S. Saad, and A. Jaekel, "A context aware and traffic adaptive privacy scheme in vanets," in *2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS)*, IEEE, Nov. 2020.

[15] F. Schaub, B. Könings, and M. Weber, "Context-adaptive privacy: Leveraging context awareness to support privacy decision making," *IEEE Pervasive Computing*, vol. 14, pp. 34–43, Jan. 2015.

[16] M. Namara, H. Sloan, and B. P. Knijnenburg, "The effectiveness of adaptation methods in improving user engagement and privacy protection on social network sites," *Proceedings on Privacy Enhancing Technologies*, vol. 2022, pp. 629–648, Nov. 2021.

[17] M. Chalikias, P. Lalou, and A. Manolesou, "Research methodology and introduction to statistical data analysis via ibm spss statistics," 2016.

[18] S. V. Bentley, K. H. Greenaway, S. A. Haslam, T. Cruwys, N. K. Steffens, C. Haslam, and B. Cull, "Social identity mapping online.," *Journal of Personality and Social Psychology*, vol. 118, pp. 213–241, Feb. 2020.

[19] M. J. Hernandez-Serrano, P. Renés-Arellano, R. Campos Ortuño, and B. González-Larrea, "Privacidad en redes sociales: análisis de los riesgos de auto-representación digital de adolescentes españoles," *Revista Latina de Comunicación Social*, pp. 133–154, Nov. 2021.

[20] M. Aresta, L. Pedro, C. Santos, and A. Moreira, "Portraying the self in online contexts: context-driven and user-driven online identity profiles," *Contemporary Social Science*, vol. 10, pp. 70–85, Jan. 2015.

[21] K. Vgena, A. Kitsiou, C. Kalloniatis, and S. Gritzalis, "Determining the role of social identity attributes to the protection of users' privacy in social media," *Future Internet*, vol. 14, p. 249, Aug. 2022.

[22] Z. Jordán-Conde, B. Mennecke, and A. Townsend, "Late adolescent identity definition and intimate disclosure on facebook," *Computers in Human Behavior*, vol. 33, pp. 356–366, Apr. 2014.

[23] A. Kitsiou, E. Tzortzaki, C. Kalloniatis, and S. Gritzalis, *Measuring Users' Socio-contextual Attributes for Self-adaptive Privacy Within Cloud-Computing Environments*, pp. 140–155. Springer International Publishing, 2020.

[24] E. R. Babbie, *The Practice of Social Research*. Boston, MA: Cengage, fifteenth ed., 2021.

[25] P. Bourdieu, *The Forms of Capital*, ch. 15, pp. 241–258. New York, USA: Greenwood, 1986.

[26] R. Jenkins, *Social Identity*. Routledge/Taylor and Francis Group, May 2008.

[27] N. Ní Bhroin, T. Dinh, K. Thiel, C. Lampert, E. Staksrud, and K. Ólafsson, "The privacy paradox by proxy: Considering predictors of sharenting," *Media and Communication*, vol. 10, pp. 371–383, Mar. 2022.

[28] K. Feher, "Digital identity and the online self: Footprint strategies – an exploratory and comparative research study," *Journal of Information Science*, vol. 47, pp. 192–205, Oct. 2019.

[29] S. Gritzalis, M. Sideri, A. Kitsiou, E. Tzortzaki, and C. Kalloniatis, *Sustaining Social Cohesion in Information and Knowledge Society: The Priceless Value of Privacy*, pp. 177–198. Springer International Publishing, June 2020.

[30] A. Kitsiou, M. Pantelelis, A.-G. Mavroeidi, M. Sideri, S. Simou, A. Vgena, E. Tzortzaki, and C. Kalloniatis, "Self-adaptive privacy in cloud computing: An overview under an interdisciplinary spectrum," in

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Proceedings of the 26th Pan-Hellenic Conference on Informatics, PCI 2022, ACM, Nov. 2022.

[31] C. Kalloniatis, "Incorporating privacy in the design of cloud-based systems: a conceptual meta-model," *Information & Computer Security*, vol. 25, pp. 614–633, Nov. 2017.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Supporting Cryptographic Algorithm Agility with Attribute Certificates

Steffen Fries, Rainer Falk
Siemens AG
Technology
Munich, Germany
e-mail: {steffen.fries|rainer.falk}@siemens.com

***Abstract*—Asymmetric cryptography is broadly used to protect confidentiality, integrity, and authenticity of data during transfer, and potentially also at rest. Typical applications are authentication and key agreement in secure communication protocols, and digital signatures for authentication and integrity protection of documents and messages. These are used in daily life applications like online banking but are specifically used in critical infrastructures to protect against misuse and manipulation. Asymmetric cryptographic algorithms are most often used with digital certificates binding a user identity to a public key of the user. These certificates are used for authentication performed during the handshake by common cryptographic security protocols like Transport Layer Security, Datagram Transport Layer Security, or by authentication and key agreement protocols like the Internet Key Exchange or Group Domain of Interpretation. The cryptographic algorithm for public-key-based user authentication is fixed by the user's certificate. More flexibility to support multiple cryptographic algorithms for user authentication is needed, e.g., by the introduction of new, quantum-safe cryptographic algorithms. Attribute certificates can be used to support flexibly multiple cryptographic algorithms for user authentication, supporting a stepwise transition towards newer cryptographic algorithms.**

Keywords–communication security; cryptographic agility; post-quantum cryptography; attribute certificates; industrial automation and control system; Internet of Things; automation control systems.

I. INTRODUCTION

Asymmetric cryptography and digital signatures are a cornerstone in many security architectures. One important application of digital signatures is related to user (entity) authentication and integrity protection of data at rest and in transit. These cryptographic security mechanisms are increasingly used in Critical Infrastructures (CI) to ensure reliable operation. CIs are technical installations that provide essential services for the daily life within a society and the economy of a country. Examples are services in healthcare, telecommunication, transportation, water supply, and power systems. In all types of CIs, a clear trend and also demand towards increased connectivity can be seen. It ensures remote access, but also continuous monitoring to optimize operation and also to support resiliency in case of failures or attacks. This goes along with a tighter integration of systems from Information Technology (IT) in common enterprise environments with the Operation Technology (OT) part of the automation systems in industrial domains.

There are several differences between IT and OT in terms of security requirements, operational processes, and the lifetime of components used in the related environments. The integration of both domains has mutual influences on the overall security and availability and requires sound security design of interconnected cyber-physical systems.

As stated before, cryptography is one of functions supporting a secure, reliable operation. Cryptographic algorithms typically also underly a lifetime in which they are can be treated as secure. Symmetric algorithms are typically designed in a way that they utilize a specific mathematical construct, like a permutation, and depend on the secrecy and/or uniqueness of certain input parameters like a secret key and nonces. Asymmetric cryptographic algorithms are often designed leveraging a specific mathematical problem, in which the calculation in one direction is easy and in the reverse direction the problem solving is computationally hard. These algorithms use two keys, a private key and a public key. Good security design uses public review and does not depend on the secrecy of the underlying mathematical construct.

As outlined in [1], cryptographic algorithms "age", as the technology to solve certain mathematic problems gets better and better. This can be seen for instance in the availability of increased computational power, e.g., increasingly higher performance processors or available cloud services, to perform brute force attacks to symmetrically encrypted data. Contrary, developments in the area of quantum computers leverage certain physical properties and utilize long-known approaches, which specifically endanger asymmetric cryptographic algorithms [2]. They can solve the previously assumed computationally hard problems much more efficiently. To keep systems secure, also considering the aging of cryptographic algorithms cryptographic agility is required in the system design and operation. This requires support for a migration from currently used cryptographic algorithms to potential new stronger algorithms in the utilized protocols and applications. While this requires considerations in the design, it also requires the flexibility from underlying systems, specifically if cryptographic algorithms are realized in hardware.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

This paper focusses on two main points. It provides background information why cryptographic algorithms agility is important from a general requirements point of view, and it addresses specific aspects related to the migration of asymmetric cryptographic algorithms. These algorithms use the construct of public and private keys. Here, a user utilizes his private key for authentication. A peer (relying party) verifies the authentication using the corresponding public key. Digital certificates, e.g., according to the ITU-T X.509 standard [3], confirm the user identity associated with the user's public key.

Besides entity authentication, digital signatures provide integrity protection of the signed content, which may be a document or, in case of the initial phase of security protocols, protect the negotiation of security parameters for a communication session as used in common security protocols like Transport Layer Security (TLS) [4] and Datagram Transport Layer Security (DTLS) [5], or in "pure" authentication and key agreement protocols like the Internet Key Exchange (IKEv2) [6] or the Group Domain of Interpretation GDOI) [7] protocol.

Due to advances in quantum computing, currently used asymmetric cryptographic algorithms like RSA (Rivest, Shamir, Adleman) or ECDSA (Elliptic Curve Digital Signature Algorithm) are endangered, as there underlying mathematical problems, like factorization and discrete logarithm problems (see also [8]) can be solved efficiently using a cryptographically relevant quantum computer leveraging Shor's algorithm (see also [9]). Symmetric cryptographic algorithms can also be attacked using Grover's algorithm (see also [9]), but for them it is currently seen sufficient to double the key length without a change of the algorithms (see also [10]).

While the standardization and the journey to introduce new, post-quantum asymmetric algorithms that withstand such attacks is still ongoing, the discussion of transition approaches for currently used cryptographic algorithms to new algorithms has already started (see [11]). In this context, different strategies are being discussed, like the combined or hybrid use of classical and post-quantum algorithms. This also relates to the utilized credentials, which may come in different formats like hybrid certificates supporting alternative cryptographic algorithms in the same certificate (see [1]). However, only a single second public key of a single second cryptographic algorithm can be included. As multiple quantum-safe cryptographic algorithms are currently standardized, a more flexible approach to support multiple public keys for authentication of a single user is needed.

Note that the case of post-quantum cryptographic algorithms is taken here as example. Crypto agility as the ability to adopt to alternative cryptographic algorithms, is a general design objective for protocols and architectures to ensure that new algorithms with similar boundary conditions can be deployed easily.

Transition is specifically important for industrial use cases, as the component lifetime here is much longer compared to consumer electronics. Therefore, it is important to elaborate ways to allow an upgrade of systems already in the field not only with new algorithms, but also with new or enhanced credentials for entity authentication.

This paper is structured in the following way. Section II provides background on requirements from regulation and standardization to design systems in a way supporting the migration of cryptographic algorithms. Section III sheds light on the topic from a more technical perspective by investigating into related work on cryptographic challenges Section IV gives an overview on public key certificates and attribute certificates to show the general structure and approach as used in asymmetric cryptographic algorithms. Section V investigates a new approach utilizing attribute certificates to support migration towards stronger cryptographic algorithms. Section VI concludes the paper and provides an outlook to potential future work.

II. From Requirements to Solutions

Security in communication infrastructures is not a new topic. Specifically in office environments or information technology (IT), it is handled as state of the art, and depending on the operational environment certification requirements of specific security processes is mandatory, or at least may provide a competitive advantage.

Critical infrastructures or operational technology (OT) on the other hand also rely on communication and utilize increasingly standard communication protocols or standard components whenever possible. This provides some commonalities regarding the utilized technology for communication, but there are distinct differences in the management and operation of these infrastructures as seen in Figure 1.

	Critical Infrastructures, e.g., Power Systems	Office IT
Protection target for security	OT, e.g., generation, transmission	IT- Infrastructure
Component Lifetime	Up to 20 years	3-5 years
Availability requirement	Very high	Medium, delays accepted
Real time requirement	Can be critical	Delays accepted
Physical Security	Very much varying	High (for IT Service Centers)
Application of patches	Slow (in maintenance windows)	Regular / scheduled
Anti-virus	Hard to deploy, white listing	Common / widely used
Security testing / audit	Increasing, partially	Scheduled and mandated

Figure 1. Comparison IT/OT management and operation

These differences in management and operation of the IT systems consequently lead to different high level security requirements as outlined in Figure 2.

	Critical Infrastructures	Office IT
Security Awareness	Increasing	High
Security Standards	Under development, regulation	Existing
Confidentiality (Data)	Low – medium	High
Integrity (Data)	High	Medium
Availability / Reliability	24 x 365 x ...	Medium, delays accepted
Non-Repudiation	Medium to High	Medium

Figure 2. Comparison IT/OT high level security requirements

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

For critical infrastructures, the European Network and Information System (NIS2) Directive [12] requires security measures to be supported by system operators specifically of critical infrastructures. This directive must be ratified by the European member states.

Germany, for instance, has passed the Information technology (IT) Security Act already in 2021 [13], which requires the further definition of domain-specific security measures that have to be implemented by operators of critical infrastructures. For the power system infrastructure, for instance, the domain specific security standard is provided by ISO 27019 [14]. Both documents target communication security in terms of authentication of communicating entities in addition to integrity and confidentiality protection of the data exchange, but without specifying specific technical means in terms of security protocols or specific cryptographic algorithms. Recommendations for the usage of cryptographic algorithms and protocol features of selected security protocols are provided from the German BSI in TR-02102 [15] and maintained on a yearly base.

In addition, the European Cyber Resilience Act (EU-CRA) [16] is currently being finalized. In addition to the NIS2 Directive, the EU-CRA defines specific requirements for manufacturers of devices, which are to be used, beyond others, also in critical infrastructures. The defined requirements relate to different aspects like the product development process, the security provided by the products, based on their features as well as the handling of vulnerabilities, detected while the products are in operation.

These regulative requirements in turn require standards of holistic nature, covering the different aspects from development and production, integration up to the end of lifetime of products. Ideally, these standards will be harmonized across different application domains to ease certification of processes and features.

A standard framework defining specific requirements for operators, integrators, and manufacturers is provided by IEC 62443 [17]. It specifically describes in two distinct parts technical requirements on system and component level, targeting four different security levels, which relate to the strength of a considered attacker. Moreover, this framework also contains requirements regarding the use of cryptographic algorithms including their strength. While ISO 62443 has been written for industrial control systems, it is meanwhile applied in power systems, in the railway industry, but also in not directly related application domains like healthcare.

Security requirements for critical infrastructures are also defined outside Europe, for instance in requirements specified by NIST Cybersecurity Framework [18], which was recently revised to an edition 2. Specifically for the power system infrastructure requirements are posed by the North American Energy Reliability Council in the NERC Critical Infrastructure Protection (CIP) standards [19]. These documents pose similar requirements as the IEC 62443 series, which relate most often to the security processes of an operator and with this direct and indirect requirement to the products used in these environments.

Common to all of the previously stated requirement documents is that they describe requirements on a "what" level, stating the expected security measures, leaving the concrete realization open. Hence, additional standards/specifications are necessary to address the technical implementation of such requirements in components and systems, while ensuring interoperability between different vendor's products. For the power system infrastructure, this is provided by the IEC 62351 series [20].

The combination of both, procedural and technical security measures provide the necessary support for reliable operation of critical infrastructure systems addressing regulative requirements. This is depicted in Figure 3.

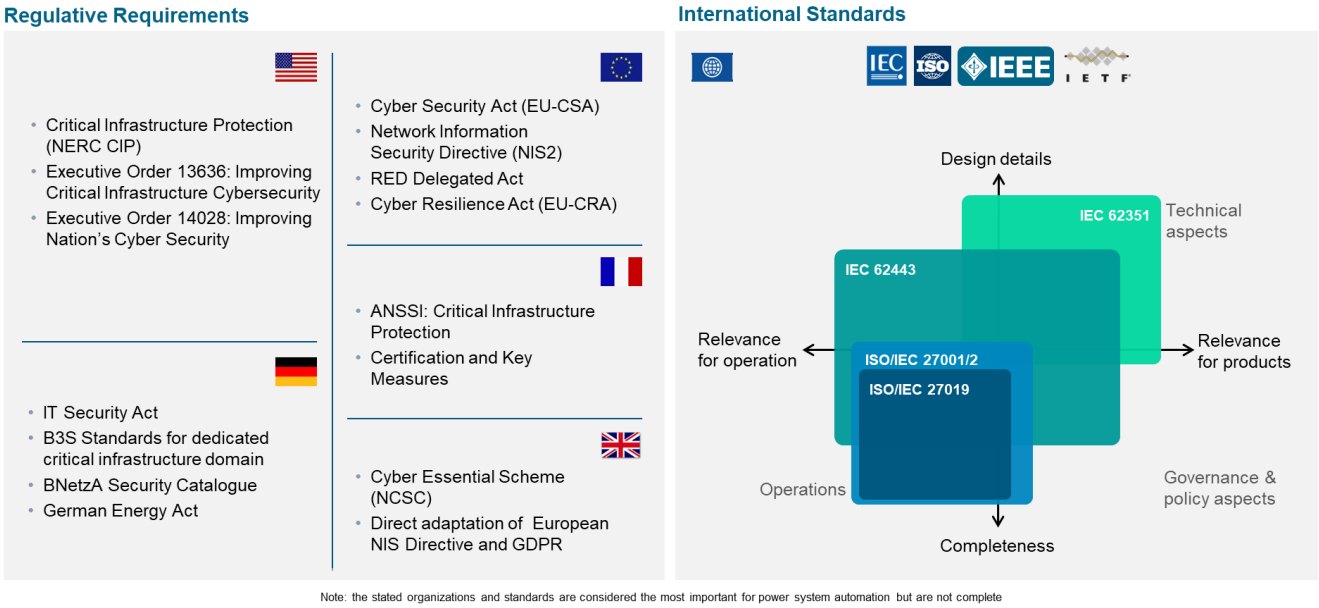


Figure 3. Relation of Regulative Requirements and Standards on the Example of Power Systems

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

III. Related Technical Work

As stated in Section II, there are several requirement sources that point to the ability to update utilized cryptographic algorithms. That this is necessary can be seen on the example of already deprecated cryptographic algorithms, which are no longer considered secure to protect security of sensitive information accordingly. Examples are, for instance, hash functions like MD5 and SHA-1 [21][22], or asymmetric cryptographic algorithms like RSA in key length with less than 2048 bit [15], or symmetric algorithms like DES [15][23].

Quantum computers are investigated since quite a while and advances in the number of supported quantum bits is increasing [24]. Cryptographically relevant quantum computers endanger algorithms like RSA or ECDSA, as their underlying mathematical problems, the factorization problem (for RSA) or the discrete logarithm problem (for ECDSA, see also [8]) can be solved efficiently leveraging Shor's algorithm (see also [9]). Symmetric cryptographic algorithms can also be attacked using Grover's algorithm (see also [9]), but for them, it is currently seen sufficient to double the key length without a change of the algorithms (see also [10]).

To find appropriate cryptographic algorithms that are considered quantum save, NIST initiated a challenge on replacement algorithms for digital signatures. This challenge is about to finish after six years. Three digital signature candidates have been selected for standardization (see [11]):

- CRYSTALS-Dilithium (ML-DSA, FIPS 204 [28])
- SPHINCS+ (SLH-DSA, FIPS 205 [29])
- FALCON

These algorithms have different parameters and different parameter sizes as the classical algorithms like RSA or ECDSA. The key size can be significantly larger compared to classical cryptographic algorithms. These parameters and key sizes need to be supported by implementations and most importantly also in the context of existing user authentication credentials like X.509 certificates.

The migration or transition to quantum-safe cryptographic algorithms is a complex undertaking. The National Institute for Standards and Technology NIST has published a draft guideline on the migration to post-quantum cryptography [27].

Transition of cryptographic algorithms has been worked on in the context of ITU-T X.509 [3] with the support of alternative cryptographic algorithms as investigated in the following Section IV.A.

With the IETF, a further standardization organization investigates the different options of migration towards post-quantum cryptographic algorithms. Here, the emphasis lies on utilizing hybrid approaches in protocols like TLS [4] or DTLS [5]. Besides integrating new algorithms in cipher suites, also approaches like Key Encapsulation (KEM, [26]) are being discussed to avoid generation of digital signatures on constraint devices.

Besides standardization of general usage protocols, also domain-specific standardization takes the migration to post-quantum cryptography into account. One example is the recent development in the power system related security standardization in the IEC, which currently works on a technical report on the Migration towards stronger cryptographic algorithms in IEC 62351-90-4.

IV. Public Key and Attribute Certificates

X.509 certificates are used for entity authentication and integrity protection. As shown in Figure 4, the concept of a public key certificate is the binding of an entity's identity to a public key, which has a corresponding private key. This private key is kept secret by the entity and can be used to authenticate the entity. The certificate itself is issued by a trusted third party, a certification authority, that digitally signs the certificate. This signature is verified by the relying party as part of certificate path validation to a root certificate.

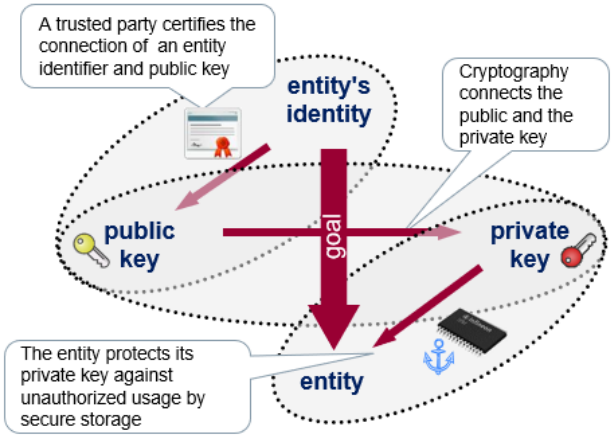


Figure 4. Concept of Binding Public Keys to Identities

These certificates are called public key certificates, as they bind the public key to an entity's identity. In addition, there attribute certificates are defined, which can be seen as temporary enhancement of public key certificates. They do not contain public keys but additional attributes that are connected to the holder of the public key certificate as shown in Figure 5. As visible in the figure, an attribute certificate has a validity period, which may vary based on the application use case. As the attribute certificate can be assumed as a temporary enhancement of a statements contained in a public key certificate, it may be short-lived, or it may have a similar validity as the public key certificate. Figure 5 also shows that the issuing authority may be different for the attribute certificate as for the public key certificate. This fact may be interesting in cases where a separation of duty is targeted.

The following subsections will provide more details on both certificate types.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

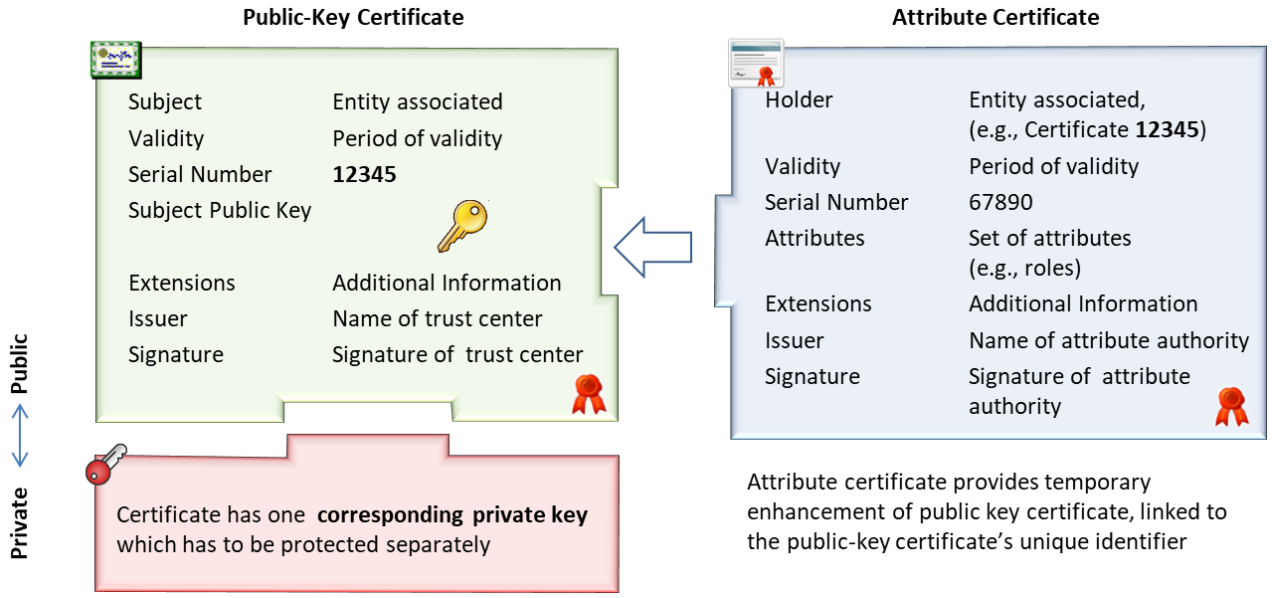


Figure 5. Concept of Public Key Certificates and Attribute Certificates

A. Public Key Certificates

ITU-T X.509 [3] is the public key certificate and attribute certificate framework widely applied in Information Technology (IT) solutions an increasingly being used in Operational Technology (OT) solutions. It defines the structure and content of public key certificates as well as the verification of the components.

```
Certificate ::= SIGNED{TBSCertificate}

TBSCertificate ::= SEQUENCE {
  version                 [0]  Version DEFAULT v1,
  serialNumber                 CertificateSerialNumber,
  signature                    AlgorithmIdentifier{{SupportedAlgorithms}},
  issuer                       Name,
  validity                     Validity,
  subject                      Name,
  subjectPublicKeyInfo         SubjectPublicKeyInfo,
  issuerUniqueIdentifier  [1] IMPLICIT UniqueIdentifier OPTIONAL,
  ...,
  [[2:  -- if present, version shall be v2 or v3
  subjectUniqueIdentifier  [2] IMPLICIT UniqueIdentifier OPTIONAL]],
  [[3:  -- if present, version shall be v2 or v3
  extensions               [3]  Extensions OPTIONAL ]]
  -- If present, version shall be v3]]
} (CONSTRAINED BY { -- shall be DER encoded -- } )
```

Figure 6. Public Key Certificate structure (see [1])

As shown in Figure 6, the certificate is a signed structure, containing the `subject` as the name of the entity and the `subjectPublicKeyInfo` structure with information about algorithm and the contained public key. The certificate is signed by an issuing certificate authority. Besides further components the certificate structure can also be extended using the `extensions` component.

To support alternative algorithms, X.509 defines three extensions to convey the:

- `subjectAltPublicKeyInfo` – alternative public key
- `altSignatureAlgorithm` – alternative signature algorithm (used to sign the public key certificate) and
- `altSignatureValue` – alternative signature value.

Using theses extensions allows a relying party depending on its capabilities to either utilize classical cryptographic algorithms or alternative (here post quantum) algorithms for the verification of the certificate (and potential digital signatures performed with the public key corresponding to the contained public key. Depending on the security policy of the relying party, both signatures of the certificate may need to be verified.

This approach is limited to a single alternative key for a public key in practical application, i.e., limited to a single alternative cryptographic algorithm. Simply adding multiple alternative keys to the authentication certificate would increase the certificate size significantly.

B. Attribute Certificates

Besides public key certificates, ITU-T X.509 [1] also defines the structure and content of attribute certificates, as well as the binding to public key certificates and the verification of contained components. Note that besides the binding to public key certificates, an attribute certificate may also be bound to a name of an entity or some fingerprint of information.

An attribute certificate may be seen as temporary enhancement of a public key certificate.

```
AttributeCertificate ::= SIGNED{TBSAttributeCertificate}

TBSAttributeCertificate ::= SEQUENCE {
  version                 AttCertVersion, -- version is v2
  holder                  Holder,
  issuer                  AttCertIssuer,
  signature               AlgorithmIdentifier{{SupportedAlgorithms}},
  serialNumber            CertificateSerialNumber,
  attrCertValidityPeriod  AttCertValidityPeriod,
  attributes              SEQUENCE OF Attribute{{SupportedAttributes}},
  issuerUniqueID          UniqueIdentifier OPTIONAL,
  ...,
  ...,
  extensions              OPTIONAL }
```

Figure 7. Attribute Certificate structure (see [1])

As shown in Figure 7, similar to public key certificates an attribute certificate is also a signed structure, containing the `holder` as the name of the entity, information about the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

issuer, including the signature algorithm and values as well as the possibility to define extensions of the attribute certificate. Like for public key certificates, to support alternative algorithms, X.509 defines two extensions to convey the:

- `altSignatureAlgorithm` –alternative signature algorithm (used to sign the attribute certificate) and
- `altSignatureValue` – alternative signature value.

The standard does not foresee the capability to contain an alternative public key of the holder as additional attribute. The next section discusses the merits of providing this information as well as further, policy related information in the context of an attribute certificate.

V. Proposed New Attributes

As discussed in Section IV, not all extensions defined for public key certificates are defined for inclusion in attribute certificates. This paper therefore proposes to use the `subjectAltPublicKeyInfo` extension also in attribute certificates to convey an alternative public key and information about the corresponding cryptographic algorithms, e.g., a public key for a post quantum asymmetric algorithm like FALCON, DILITHIUM, or SPHINCS+. This allows to associate and utilize alternative public keys to already existing certificates. As multiple attribute certificates can be issued for a single user certificate, implicitly various cryptographic algorithms can be supported in a flexible way by issuing multiple corresponding attribute certificates.

Attribute certificates contain attributes, and providing an alternative public key as attribute is proposed as novel approach. It is intended to support smooth transition to public-key certificates using solely alternative, in the case here, post quantum cryptographic algorithms. As they are intended as temporary enhancement of public key certificates, this approach is seen appropriate. It is even possible to issue attribute certificates for an entity's public key certificate at a later point in time.

For migration to post-quantum cryptography, it is necessary to also support a security policy which handles the transition from one cryptographic algorithm to an alternative cryptographic algorithm (in the case here for digital signatures). Such a policy may require verifying only one signature, both signatures (classic and alternative), and may also provide a weight on the verification result, e.g., by the order of operations. Such a security policy may be configured per relying party. In case of automation networks, it may be part of the engineering data for the Intelligent Electronic Devices (IED).

An alternative approach to the device configuration of security policies is the provisioning of the policy as part of the certificate, also in the form factor of an extensions. This paper proposes such an extension as shown in Figure 8 that may be applied in both certificate types, i.e., to public key certificates as well as to attribute certificates.

```
altCryptoPolicy ::= SEQUENCE {
  combAND     [0] boolean OPTIONAL,
  combOR      [1] boolean OPTIONAL,
  weightOnAlt [2] boolean OPTIONAL
}
```

Figure 8. Proposed Migration Policy Extension

The extension allows to specify the following security policies for the associated alternative public key:

- `combAND` requires the verification of the signature performed with the classic asymmetric algorithm as well as the alternative algorithm.
- `combOR` requires the verification signatures created with of either the classical or the alternative cryptographic algorithm,
- `weightOnAlt` indicates if the alternative algorithm has a higher weight in the evaluation. Note that this can be used in conjunction with `combOR` for the selection of classical or alternative signatures and also for the `combAND` case in cases, in which one signature verification may fail.

The extension may be included in the certificate as critical extension to ensure that it will be evaluated by the relying party. The inclusion into public key certificate can be done to associate a fixed security policy to the two contained public keys. There is also a benefit by placing the extension into an attribute certificate even in cases where the second public key is not contained in the attribute certificate but in the public key certificate. This approach allows to change the security without the need to issue a new public key certificate, enabling dynamic policy changes.

VI. Conclusion and Outlook

This paper provides an overview on the need for a transition from currently used classical cryptographic algorithms to new, alternative cryptographic algorithms from a requirements and standardization point of view, but also from a technical perspective. More specifically, the focus is placed on the use of digital signatures and credentials conveying the public key within X.509 certificates.

In that respect, a novel approach for using alternative asymmetric algorithms in the context of X.509 certificates has been described. It is proposed to support alternative public keys and associated information in attribute certificates, which enhances the application of already defined certificate extensions for public key certificates also for attribute certificates. By this approach, multiple cryptographic algorithms can be supported flexibly by issuing multiple attribute certificates corresponding to the different public keys of a user. Moreover, a further security policy extension is proposed that allows a dynamic adaptation of the security policy for the transition from classic cryptographic algorithms towards alternative, e.g., post quantum algorithms.

The discussed approach is currently in its infancy and needs to be implemented and tested to get practical experience. This is seen as the next consequent step. Due to the use of an already existing extension to transport the alternative public key, further investigation of the transport of algorithm specific parameters is not seen necessary as already considered in the originally defined extension.

Besides the necessity to perform a deeper investigation of the side conditions of this approach and also a proof-of-concept implementation, it is seen necessary to discuss this approach within standardization. This is because most interacting systems are built with products from different manufacturers. Therefore, standardization is necessary to

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

ensure interoperability of productions developed by different manufacturers.

REFERENCES

[1] S. Fries and R. Falk, "Using Attribute Certificates to Support Cryptographic Algorithm Flexibility", ICSNC 2023, The Eighteenth International Conference on Systems and Networks Communications, pp. 6-9, 2023 [Online]. Available from: https://www.thinkmind.org/index.php?view=article&articleid=icsnc_2023_1_20_20022 [retrieved: May, 2024]

[2] Federal Office for Information Security, "Quantum-safe cryptography – fundamentals, current developments and recommendations", 2022, [Online]. Available from: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Brochure/quantum-safe-cryptography.pdf, [retrieved: February, 2024]

[3] ITU-T X.509 ISO/IEC 9594-8:2020, Rec. ITU-T X.509 (2019), Information technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks, [Online]. Available from: https://www.itu.int/rec/T-REC-X.509-201910-I/en, [retrieved: May, 2024]

[4] E. Rescorla, IETF RFC 8446, "Transport Layer Security (TLS) Protocol v1.3", August 2018, [Online]. Available from: https://tools.ietf.org/html/rfc8446, [retrieved: August, 2023]

[5] E. Rescorla, H. Tschofenig, and N. Modadugu, IETF RFC 9147, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", April 2022 [Online]. Available from: https://datatracker.ietf.org/doc/html/rfc9147, [retrieved: May, 2024]

[6] C. Kaufman, P. Hoffman, Y. Nir, P.Eronen, and T. Kirvinen., IETF RFC 7296, „Internet Key Exchange Protocol Version 2 (IKEv2)", 2014, [Online]. Available from: https://datatracker.ietf.org/doc/html/rfc7296, [retrieved: May, 2024]

[7] B. Weis, S. Rowles, and T. Hardjono, IETF RFC 6407, "The Group Domain if Interpretation", October 2011, [Online]. Available from: https://datatracker.ietf.org/doc/html/rfc6407, [retrieved: May, 2024]

[8] A. J. Menezes, P. C. v. Oorschot, and S. A. Vanstone, "Handbook of Applied Cryptography", CRC-Press, October 1996, ISBN: 0-8493-8523-7

[9] D. J. Bernstein, J. Buchmann, and E. Dahmen, "Post-quantum cryptography", Springer, Berlin, 2009. ISBN 978-3-540-88701-0

[10] L. Cehen et al., NISTIR 8105, "Report on Post-Quantum Cryptography", April 2016, [Online]. Available from: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf, [retrieved: May, 2024]

[11] NIST "PQC Standardization Process: Announcing Four Candidates to be Standardized", July 2022, [Online]. Available from https://csrc.nist.gov/news/2022/pqc-candidates-to-be-standardized-and-round-4, [retrieved: May, 2024]

[12] NIS2 Directive, 2022, [Online]. Available from https://www.bsi.bund.de/EN/Das-BSI/Auftrag/Gesetze-und-Verordungen/IT-SiG/it_sig.html, [retrieved: February, 2024]

[13] The German IT Security Act, 2021, [Online]. Available from: https://eur-lex.europa.eu/eli/dir/2022/2555, [retrieved: May, 2024]

[14] ISO 27019: Information technology - Security techniques - Information security controls for the energy utility industry, 2017, [Online]. Available from: https://www.iso.org/standard/68091.html, [retrieved: May, 2024]

[15] BSI TR-02102, "Cryptographic Mechanisms", Online]. Available from: https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr02102/tr02102_node.html, [retrieved: May, 2024]

[16] EU Cyber Resilience Act, [online]. Available from: https://digital-strategy.ec.europa.eu/en/policies/cyber-resilience-act, [retrieved: May, 2024]

[17] IEC 62443, "Industrial Automation and Control System Security", [Online]. Available from: https://www.iec.ch/blog/understanding-iec-62443, [retrieved: May, 2024]

[18] NIST Cybersecurity Framework 2.0, 2024, [Online]. Available from: https://www.nist.gov/cyberframework, [retrieved: May, 2024]

[19] NERC CIP Set of Standards, [Online]. Available from: https://www.nerc.com/pa/Stand/Pages/Cyber-Security-Permanent.aspx, [retrieved: May, 2024]

[20] IEC 62351-x Power systems management and associated information exchange – Data and communication security, [Online]. Available from: http://iectc57.ucaiug.org/wg15public/default.aspx, [retrieved: May, 2024]

[21] L. Velvindron, K. Moriarty, and A. Ghedini, IETF RFC 9155, "Deprecating MD5 and SHA-1 Signature Hashes in TLS 1.2 and DTLS 1.2", 2021, [Online]. Available from: https://datatracker.ietf.org/doc/html/rfc9155, [retrieved: May, 2024]

[22] Announcement, "NIST retires SHA-1 Cryptographic Algorithm", 2022, [Online]. Available from: https://www.nist.gov/news-events/news/2022/12/nist-retires-sha-1-cryptographic-algorithm, [retrieved: May, 2024]

[23] NIST SP 800-131Arev.2, "Transitioning the Use of Cryptographic Algorithms and Key Lengths", 2019, [Online]. Available from: https://csrc.nist.gov/pubs/sp/800/131/a/r2/final, [retrieved: May, 2024]

[24] Federal Office for Information Security, "Status of quantum computer development", Version 2.0, 2023, [Online]. Available from: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Quantencomputer/Entwicklungstand_QC_V_2_0.pdf, [retrieved: May, 2024]

[25] NIST Announcement, "PQC Standardization Process: Announcing Four Candidates to be Standardized", July 2022, Online]. Available from: https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4, [retrieved: May, 2024]

[26] F. Giacon., F. Heuer, and B. Poettering, "KEM Combiners", 2018, [Online]. Available from: https://doi.org/10.1007/978-3-319-76578-5_7, [retrieved: May, 2024].

[27] W. Newhouse, M. Souppaya, W. Barker, and C. Brown, "Migration to Post-Quantum Cryptography: Preparation for Considering the Implementation and Adoption of Quantum Safe Cryptography", April 2023, [Online]. Available from: https://www.nccoe.nist.gov/crypto-agility-considerations-migrating-post-quantum-cryptographic-algorithms [retrieved: May, 2024]

[28] FIPS 204: "Module-Lattice-Based Digital Signature Standard", 2024, [Online]. Available from: https://csrc.nist.gov/pubs/fips/204/ipd, [retrieved: May, 2024].

[29] FIPS 205: "Stateless Hash-Based Digital Signature Standard", 2024, [Online]. Available from: https://csrc.nist.gov/pubs/fips/205/ipd, [retrieved: May, 2024].

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Redesign and Feasibility Verification of Access Control System Based on Correlation Among Files

Yuki Kodaka
Department of Informatics, The Graduate University for Advanced Studies
Tokyo, Japan
email: y_kodaka@nii.ac.jp

Hirokazu Hasegawa
Center for Strategic Cyber Resilience Research & Development, National Institute of Informatics
Tokyo, Japan
email: hasegawa@nii.ac.jp

Hiroki Takakura
Center for Strategic Cyber Resilience Research & Development, National Institute of Informatics
Tokyo, Japan
email: takakura@nii.ac.jp

***Abstract*—File access control is an effective method for protecting information from unauthorized access both inside and outside an organization. However, conventional methods based on organizational structure have some limitations. Modern business requires flexible access control that reflects the dynamic changes in workflow. Still, it is difficult to achieve the requirement at the same time the prevention of information leakage and destruction due to cyberattacks. Therefore, this paper proposes an access control system based on the correlation among files. The correlation is inferred from users' access histories within the same group, and access privilege is determined based on the strength of the correlation. This system adapts to changing access needs and prevents unauthorized access by automatically denying access with low file-to-file correlation in a series of accesses. The initial implementation of the system was carried out in a simplified environment, which raised issues about whether the system could be feasible and efficient in real-world, more complex scenarios. This work extends the findings of our previous paper by addressing identified issues with the proposed system through targeted modifications. To further validate the system's performance and feasibility in real-world scenarios, we conducted subsequent implementation and verification experiments under conditions that were not only more practical but also involved higher loads. These efforts aimed to rigorously test the system's scalability and efficiency in environments that closely mimic actual operational conditions. As a result of these modifications and experiments, the system demonstrated the capability to handle high-load conditions efficiently. This outcome suggests that the potential impact on file system processing due to the introduction of new features via the proposed system is not serious. Therefore, our extended research confirms the proposed system's robustness and suitability for real-world application, highlighting its ability to maintain efficiency even under significant stress. To ensure the feasibility of the proposed system, future work should address the effectiveness issue.**

***Index Terms*—File Access Control; Graph Theory; Bell-LaPadula Model.**

I. Introduction

This paper follows up our previous paper "Design and Implementation of Access Control Method Based on Correlation Among Files" already published in the proceedings of CENTRIC 2023 [1].

File access control has long been used as an effective method of protecting an organization's information assets. It prevents unauthorized accesses by users and minimizes information leakage due to cyber attacks. Various access control methods have been proposed and developed [2]–[4].

However, many of the current methods and operations are not flexible enough. Due to changes in the situations, access control loses accuracy over time [5]. In some cases, policymakers (high-level policy architects) and implementers of policy designed by others are separated. And policies are often managed by several persons rather than a single person [6]. These also make flexible operation difficult.

Strict access control is required, especially in environments where sensitive information is handled. For example, the Bell-LaPadula model [7] was proposed to prevent the leakage of information known only to the supervisor to subordinates. However, in many cases, supervisors can write to files that their subordinates can read and write to.

According to Proofpoint report [8], the cost of insider threats has surged from $8.30 million in 2018 to $15.38 million in 2022, an 85% increase. In order to mitigate insider threats, not only technical approaches like access control systems, but also non-technical approaches like user behavior analytics are needed [9].

To address these issues, we point out two challenges. First, the principle of least privileges frequently discussed, but there is no practical method of assigning access privileges based on it. As business progresses, the required access privileges are constantly changing. However, such changes are always incurred and unpredictable beforehand. Second, there is no method of implementing granular file access control in response to constantly changing needs. Obviously, the assignment of access privileges should be done with caution. Excessive access privileges increase the risk of leakage or destruction. On the other hand, insufficient access privilege affects the ability of users to perform their operations. As a result, it may undermine the efficiency and productivity of the organization.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

To solve these problems, we have proposed an access control system based on the correlation among files [10]. The correlation is inferred from user's access histories. The system automatically determines whether access is allowed or denied based on the degree of the correlation. It responds to access needs based on changing situations. The system automatically denies accesses with low correlation. It prevents excessive expansion of the access privilege. It is assumed that access by malware is an uncorrelated access. Or, even access by an insider is assumed to be uncorrelated if it is not related to the person's business. These accesses are different from legitimate users. This system can prevent such file accesses. We designed and implemented our system to include file-correlation-based access control as one of its elements. However, there were issues about this design, and the implementation was in a simplified environment. To verify the system's feasibility, it was necessary to address these issues and verify its scalability in a more practical environment. For this study, after making modifications to our system, we extended implementation and verification experiments under more practical and high-load conditions.

This paper is organized in the following sections. Section II refers to related work to this paper. Section III describes the assumptions of the proposed system. After that, we explain the design of the proposed system. Section IV describes the implementation of the proposed system. Section V describes the evaluation experiment of the proposed system. Section VI concludes this paper and presents future work.

II. Related work

Users are sometimes denied access to files they actually have the right to use, and administrators are required to modify the access control of the files. They might make a misconfiguration at the modification that gives more access privileges than necessary. Xu et al. investigated how and why such problems occur [11]. Although several reasons for misconfiguration are shown, administrators must solve such problems by themselves, and the possibility of misconfiguration and the burden on administrators remains.

Beckerle and Martucci proposed the metrics to evaluate and quantify access control rule sets in terms of security and usability [12]. The metrics helps users generate better rule sets. One of the evaluation indicators is the difference between the owner's intention and the rule set. However, the actual method of getting the intention is out of the scope of the paper.

Mazurek et al. proposed reactive policy creation in response to user's access request [13]. The experiment involves sharing files on digital devices at home with people, including supervisors and co-workers. If a user tries to access a resource but lacks sufficient privilege, they can use the proposed system to send a request to the resource owner, who can opt to update their policy and allow the access. This method requires the file owner to make determinations for all unauthorized access.

Shalev et al. proposed an improved method for containers that allows monitoring and logging of operations by the system administrator [14]. The operations used by system administrators include not only support by internal IT department employees, but also by third parties such as storage service providers and automated management tools used by the IT department. The system administrator is expected to operate based on user requests (tickets in this paper), but there is no mention of whether or not those requests are necessary.

Desmedt and Shaghaghi proposed an access control method that considers three dimensions: subject, object, and operation, rather than the conventional two dimensions of subject and object [15]. These mainly counter internal threats and provide granular access control by controlling operations. It shows how to implement granular access control, but does not mention how to update access privileges once they have been set.

Although not access control, research has been conducted to identify legitimate and malicious users based on user behavior. Mannila et al. introduced the Window-based Episode Discovery (WINEPI) and Minimal Occurrence-based Episode Discovery (MINEPI) algorithms, offering advanced methods for analyzing patterns in event sequences known as 'episodes' [16]. WINEPI calculates the frequency of episodes within time windows, effectively identifying common patterns. MINEPI complements this by focusing on the minimal occurrences of episodes, revealing the shortest intervals for episode beginnings. These approaches provide a comprehensive framework for pattern detection, crucial in fields such as telecommunications and user behavior analysis.

Camina et al. focused on masquerade detection using user navigation structure, employing the Naïve Bayes classifier [17]. The 'user navigation structure' refers to how a user interacts with their file system. In their 2014 research, they shifted focus to abstracting user behavior into tasks, utilizing both Naïve Bayes and Markov Chain classifiers [18]. Here, a 'task' is associated with a file system (FS) directory, encompassing a number of related file system objects. These studies illustrate the evolution in detecting masquerades, from analyzing direct navigation patterns to abstracting user actions into categorized tasks.

Huang et al. addressed the challenge of intrusion detection through the analysis of user file access patterns [19]. Their approach involves modeling user behavior using both Non-frequency-based and Frequency-based feature sets. The former includes metrics like the duration and diversity of file access paths, while the latter focuses on aspects like access frequency and file type variations. This comprehensive analysis of user behavior significantly contributes to the effectiveness of intrusion detection systems.

Mehnaz and Bertino explored anomaly detection in file system accesses by leveraging enhanced user profiles [20]. These profiles, incorporating frequency and access cluster data similar to Mannila et al.'s approach [16], are further enriched by block-level analysis derived from access size logs. The research involves comparing these comprehensive profiles against users' access patterns to effectively identify anomalies, providing an advanced framework for detecting irregularities in file system usage.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

In research that utilizes these access patterns for detecting unauthorized access, it is possible to distinguish between the main similar access patterns and others. However, there is a risk of misjudging the unique access patterns of individual users.

III. Design of Proposed System

In this section, we first discuss the file correlation-based access control system proposed in our previous works [1], [10]. We then address the issues identified with the system's design. Finally, we describe the modifications made to the design in response to these issues.

A. Assumption

The proposed system assumes an organization consisting of a hierarchical structure as shown in Fig. 1. Although agile development is practiced, a certain organizational structure still exists. This paper calls the largest segment of an organization, such as a department in a typical enterprise, a group. Divided units within the group are called subgroups, and further divided units within a subgroup are called subsubgroups. In the example shown in Fig. 1, each department is a group, and each section is a subgroup.

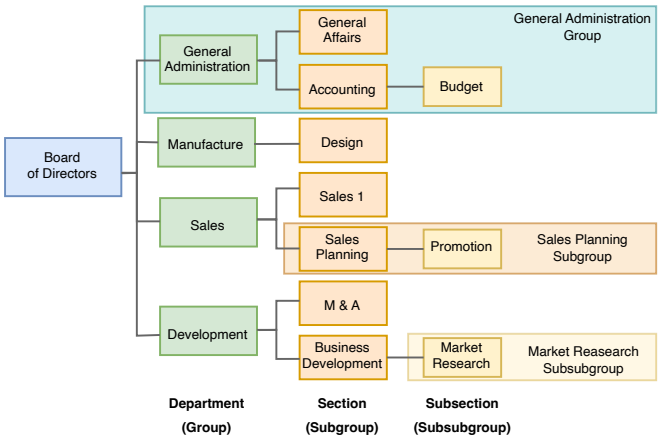


Fig. 1: Example of Organizational Structure

In this paper, we propose a file access control system based on the correlation among files. Our working hypothesis is that user access patterns are not random or irregular, but rather exhibit a certain level of consistency and commonality.

In collaborative environments, it is a recognized fact that users tend to be team-oriented [21]. This study operates under the hypothesis that typical users within a Collaborative Information System are likely to form and function as communities.

Therefore, we assume that the proposed system will be effective in enhancing the security of team-based workflows. This approach is particularly well-suited for business processes that are team-oriented.

Fig. 2 shows the assumed access control environment. Generally, an Access Control List (ACL) is implemented with coarse-grained, such as per folder, for groups or subgroups. The access privilege under a folder is determined using the information in a directory service, such as Active Directory (AD). If fine-grained access control is to be implemented, it is set by the file owner or the administrator, but their load becomes significant.

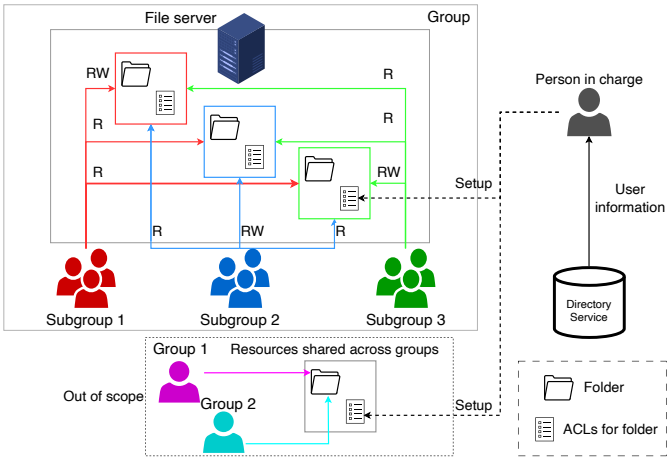


Fig. 2: Assumed Access Control Environment

In this paper, resources shared within each group are targeted, and resources shared across groups are out of scope.

B. Overview of Proposed System

With the dynamic changes in the environment, the required privileges are constantly changing. To respond to such changes, the proposed system has the capability to continuously evaluate and manage access privileges. It executes two main functions: the addition of necessary access privileges and the revocation of unnecessary ones.

1) Addition of necessary access privileges: Granting of access privileges is executed on the basis of two-step determination, i.e., automatic determination and manual determination. When a user tries to access a file without access privileges, such access is denied first. The proposed system triggered by the event of access denial. Then the proposed system performs an automatic access determination on the denied access. If the results of the determination show that there is a correlation between the files to which the user has access privilege and the denied files, the ACL is changed to "allowed" to access the file. Even if the access is denied as a result of the automatic access determination, the user can request a manual access control determination if the access is truly necessary. In this paper, it is assumed that manual determination is performed by the file owner.

2) Deletion of unnecessary access privileges: The revocation is executed when there has been no access for a certain period of time. The proposed system records the file access history of each user for a certain period in the past. If a user has not accessed a file for this period, the access privilege is considered no longer needed, and it is revoked. The longer the period of access history recorded, the longer access privileges can be retained, which offers advantages from a usability

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

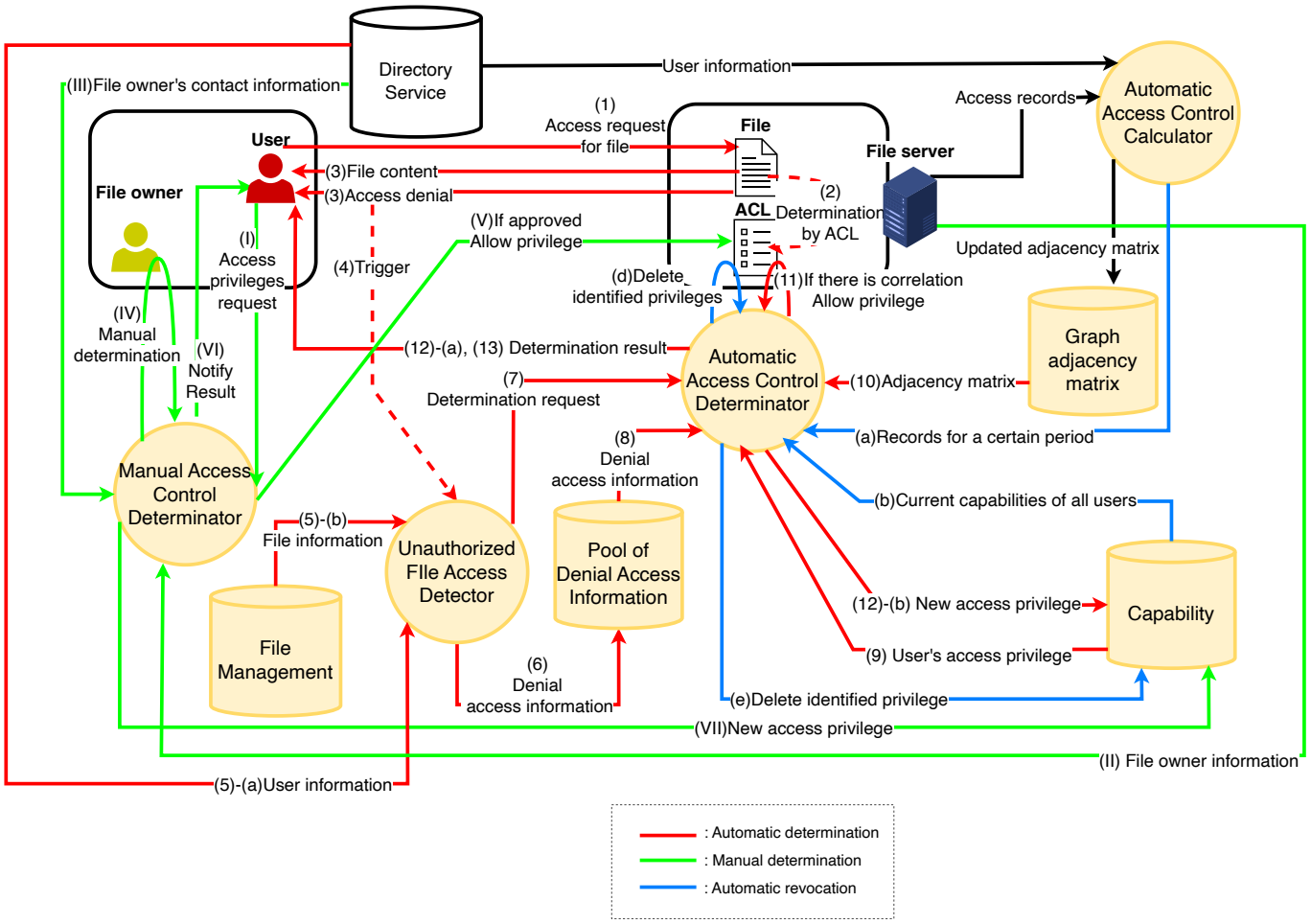


Fig. 3: Architecture of Proposed System (yellow)

perspective. On the other hand, the shorter the recording period, the quicker unused access privileges are removed, providing benefits from a security perspective. Given these trade-offs, the period of recorded access history should be flexibly set according to the organization's needs and the job operations. As an example, this paper records the file access history for the past month (the past 30 days).

C. Architecture of Original System

Fig. 3 shows the architecture of the proposed system. The compoment of the proposed system is yellow color. It is assumed that the other components, apart from the proposed system, already exist within the organization. It includes the assumed flow of access determination and the source of information necessary for the determination. The proposed system consists of an automatic access control calculator (AACC), unauthorized file access detector (UFAD), automatic access control determinator (AACD), manual access control determinator (MACD), and four databases store the target file information, the denied access information, the adjacency matrix of the graph, and the access privileges by each user.

The procedure of the proposed system process is as follows. The term "username" refers to a unique identifier for users, similar to a user ID, which is managed within a directory service. Similarly, "filename" refers to a unique identifier that can be assigned to a full path or an ID given to a file.

- Automatic determination of access privileges (red line)

(1) Users access files
(2) File server determines whether the access is allowed or denied based on the ACL set for each file
(3) (a) If allowed, the user gets the file content
(b) If denied, the user is notified of access denial, and the access denial is recorded in the access log
(4) The UFAD is triggered by the event of access denial
(5) (a) The UFAD fetches the target user information from the database for directory services
The user information is username, rank, affiliation
(b) The UFAD fetches the target file name from the database for file management database
(6) If the record shows denial access for the target files and the users,
the UFAD store it in the pool of denial access information database
The denial access information is Timestamp, Username, Filename, Accesstype (R or W)

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

(7) The UFAD makes a determination request to the AACD
(8) The AACD fetches the denial access information from the pool of denial access information database
(9) The AACD fetches the user's access privileges from the the capability database
(10) The AACD performs the access determination using graphs
(11) If there is a correlation, The AACD allows the access privilege to the user
(12) If allowed,
  (a) the AACD notifies the user of the result 'Allowed' The notification includes Timestamp, Username, Filename and Accesstype (R or W)
  (b) The AACD adds the new access privilege to the capability database
(13) If denied,
the AACD notifies the user of the result 'Denied'
The notification includes Timestamp, Username, Filename and Accesstype (R or W)
Additionally, the user has the option to submit a new access privilege request along with the reason for the request

- Manual determination of access privileges (green line)

(I) The MACD receives the new access privileges request from users
(II) The MACD fetches the file owner information from the File server
(III) The MACD fetches the file owner's contact information from the directory service
The contact information is like email address or chat-tool used in the organization
(IV) The MACD requests the file owner to determine whether the access is allowed or not
(V) If approved, the MACD allow the access privilege to the user
(VI) The MACD notifies the result to the user
(VII) The MACD adds the new access privilege to the capability database

- Deletion of unnecessary access privileges (blue line)

(a) The AACD fetches the records for certain period from the AACC
(b) The AACD fetches the capabilities, which detail the current access privileges of each user
(c) The AACD compares the records with the capabilities identifying user names, file names, and access privileges (R or W) that are present in the capability database but not in the records
(d) The AACD deletes the identified privileges from the ACL
(e) The AACD also deletes the identified privileges from the capability database

D. Function of each Component

The functions of each component are described below.

1) Automatic Access Control Calculator (AACC): The AACC calculates graphs utilizing graph theory for correlation determination. The graph infers the correlation among files based on the user's access histories, which are sourced as records from access logs on the file server. Each event of a user accessing a file is logged as a record.

The graph calculation is planned to schedule at two specific times: the change of the date and at arbitrary intervals during business hours. Firstly, at the stroke of midnight, the proposed system fetches access records from the past 30 days up to the day before from the file server, along with usernames, ranks, affiliations as the necessary user information from the directory service. This data is used to calculate the graph's adjacency matrix, as indicated by the black lines in Fig. 3. During business hours, the proposed system continues to update this information at set intervals, such as hourly. It fetches the current day's access records and user information to update the adjacency matrix. The final graph is a composite of the matrices generated at the date change and during the business hours. This composite graph is then stored in a database for adjacency matrices, as indicated by the black lines in Fig. 3.

For the graph calculation, the AACC maintains a dataset that combines the access records from the past 30 days (up to the day before) obtained at the moment the date changes, with the access records of the current day acquired during business hours. This ensures that the AACC possesses continuous records of access from 30 days prior up to the most recent data available at the predetermined intervals.

The calculation procedure is as follows. Data is access records for a certain period, which is the past month (the past 30 days) in this paper.

a) Extract specific information from access records: Specific information in the access records is fetched as the access history used in the calculation. The specific information is "Timestamp", "Accesstype" (R or R/W), "Username", "Filename". The extracted access histories are sorted by username and time.

b) Categorize access histories by user rank and access type: Extracted access histories are categorized by user rank and access type. Ranks are assumed to be hierarchical. For example, from the top, director, manager, section chief, member. For each user rank, two access histories are categorized. One is the access history of the "Read" access type for users below the same rank. The other is the access history of the "Write" access type for users in the same rank.

c) Create graphs from access histories: An example graph is shown in Fig. 4. The graph consists of nodes (V_1, V_2, V_3) and links between nodes (L_{1-2}, L_{2-3}, L_{1-3}). In the graph, nodes represent files. Links represent the correlations among files. The graph is assumed to be undirected. The order of accesses, A-B and B-A are counted as the same.

The graph is calculated using an adjacency matrix. Adjacency means that node i and node j are adjacent to link $i-j$ in the graph. An adjacency matrix is a square matrix used to represent a finite graph. The elements of this matrix indicate if a pair of nodes is adjacent or not in the graph. If so, it

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

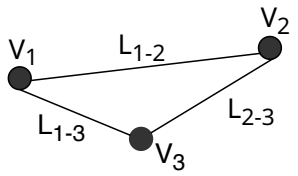


Fig. 4: Example of Graph

indicates the weights of the links between adjacent nodes. An example of an adjacency matrix is shown in Table I.

TABLE I: EXAMPLE OF ADJACENCY MATRIX

	FileA	FileB	FileC	FileD
FileA	0	3	0	1
FileB	3	0	1	5
FileC	0	1	0	1
FileD	1	5	1	0

The following procedure is used to calculate link weights.

a. Get a list of files by rank and access type from categorized access histories
b. Determine the size of the adjacency matrix from the list
c. Create adjacency matrix initialized to 0
d. Calculate weights of links from categorized access histories
   Add link weights between consecutive files in the categorized access histories, if the same user accesses different files within a certain period of time (one hour in this case). Furthermore, when calculating weights, we consider the time inclination shown in (1) based on the timestamp:
$$1 - \left(\frac{D}{D_{\max}}\right)^n \tag{1}$$
   where D is the number of days elapsed from the most recent day, $D_{\max}$ is the number of calculation days, and n is an adjustment parameter.
e. Normalize weights of links
   Let A be the adjacency matrix before normalization, and $S(n)$ be the total weight of the links connected to each node n. Normalization is performed as shown in (2):
$$B(i,j) = \frac{A(i,j)}{S(i)} + \frac{A(j,i)}{S(j)} \tag{2}$$
   where $B(i,j)$ is the element at the ith row and jth column of the adjacency matrix after normalization, $A(i,j)$ is the element at the ith row and jth column of the adjacency matrix before normalization, and $A(j,i)$ is the element at the jth row and ith column of the adjacency matrix before normalization. Also, round off to the second decimal place.

The algorithm, which is the pseudocode of the above procedure, is shown in Algorithm 1.

Table II shows the results of the above calculation steps using Table I as an example.

Algorithm 1
Calculating link weight based on access histories with time inclination

Require: Categorized access histories, list of target users, list of target files
Ensure: Adjacency matrices for each rank and access type

```
Define list of ranks and list of access types
for each rank in list of ranks do
    for each access type in list of access types do
        Load a access history data
         from categorized access histories
        Obtain list of unique filenames from data
         and sort them as specified
        Initialize adjacency matrix
         based on sorted filenames
        for each user in data do
            Group access histories by user
            for each history in user's group do
                Set time limit according to access type
                 (1 hour for read, 2 hours for write)
                if time difference to a next history
                 within limit and
                 a next history refers to different file then
                    Calculate elapsed time
                     since the log's timestamp
                    Calculate weight
                     using time decay formula:
                        weight = 1 − (elapsed days / D_max)^n
                    Add calculated weight
                     to corresponding elements
                     in adjacency matrix
                end if
            end for
        end for
        Output adjacency matrix
    end for
end for
```

TABLE II: EXAMPLE OF NOMALIZED ADJACENCY MATRIX

	FileA	FileB	FileC	FileD
FileA	0.00	1.08	0.00	0.39
FileB	1.08	0.00	0.61	1.27
FileC	0.00	0.61	0.00	0.64
FileD	0.39	1.27	0.64	0.00

2) Unauthorized File Access Detector (UFAD): When access is denial, a record is logged in the access log. The UFAD is triggered by the event of this record being logged. Once triggered, The UFAD compares the username and filename from the record with the target usernames obtained from directory service and the target filenames obtained from the file management database. If there is a match, the UFAD extracts the timestamp, username, filename and access type

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

(R or W) from the record and saves this information as denial access information in the pool of denial access information. Subsequently, a request for access determination is made to the AACD.

3) Automatic Access Control Determinator (AACD): This component plays a crucial role in managing access privileges. When it is necessary to add access privileges, this component is responsible for actually adding these privileges to the ACL. Additionally, this component also has the role of removing privileges from the ACL when they are deemed unnecessary.

a) Addition of necessary access privileges: The AACD is activated upon receiving a determination request from the UFAD. Its role involves making access privileges determinations using information on access denial and the correlation between files. Specifically, it determines based on the correlation between files that a user has already the access privilege to and those that have been denied access. If the determination results in granting, the privileges are added on the ACL. In case of denial, the user is advised to request a manual determination if truly necessary. The AACD continues this process as long as there is data in the pool of denial access information.

The determination method is as follows. If the elements of the adjacency matrix exceed a certain threshold, it is considered correlated. The formula is shown in (3). Here, as an example, the threshold is set at 0.8 or higher.

$$\mathrm{Matrix}(\mathrm{File}_{\mathrm{old}}, \mathrm{File}_{\mathrm{new}}) \geq 0.8 \tag{3}$$

where $\mathrm{Matrix}(\mathrm{File}_{\mathrm{old}}, \mathrm{File}_{\mathrm{new}})$ is the correlation between files, $\mathrm{File}_{\mathrm{new}}$ is the new file to be accessed by user u, and $\mathrm{File}_{\mathrm{old}}$ is the file already accessed by user u.

b) Revocation of unnecessary access privileges: When the date changes, the AACC fetches records of access logs for a certain period. The AACD then fetches these records from the AACC. Additionally, the AACD fetches the current access privileges for each user from the capability database. The AACD compares the records with the current access privileges, identifying user names, access types (R or W), and file names that are present in the access privileges but not in the records. It indicates that the user has not accessed those files with those privileges for a certain period. The AACD then revocates the identified user names, access types, and file names from the ACL. Furthermore, the AACD also revocates these access privileges for the identified user names, access types, and file names from the capability database.

4) Manual Access Control Determinator (MACD): If the result of the automatic determination is denial, manual determination is carried out by the file owner in response to the user request. The MACD serves as the receiver of the access-privilege request from the user and the sender of the manual access determination request to the owner of the file. The user's request includes a Timestamp, Username, Filename, Access type (R or W), and the reason for the access privilege request. Upon receiving a request, the MACD retrieves the owner information of the specified Filename from the file server. Additionally, the file owner's contact information is obtained from the directory service. This ensures that the MACD can effectively communicate the manual access determination request to the file owner, equipped with all necessary details about the user's request and the file in question.

5) Directory Service: This component store user information. This setup assumes the use of pre-existing components within the system environment prior to the implementation of the proposed system. For example, AD can be considered as such a component. The information stored in this component includes user names, ranks, affiliations, and contact information.

6) File Management Database: This component stores the names of files targeted by the proposed system. The proposed system uses this component to decide whether a file is to be determineed.

7) Pool of Denial Access Information: This component stores information about denial access. This includes the timestamp, username, filename, and accesstype (R or W) all of which are in the access record. The UFAD is responsible for saving this information in this component. After a determination is made, the AACD then deletes this information from this component.

8) Graph Adjacency Matrix: This component stores the adjacency matrices of the graph. For each rank, matrices for each access type (R or W) are saved. These adjacency matrices are calculated by the AACC.

9) Capability: This component stores the capabilities for each user. These lists include the access privileges owned by each user, specifying the filenames and the accesstypes (R or W).

E. Issues on Scalability and Efficiency

After the proposed system was designed, the performance of the proposed system was tested in a brief environment. As a result, it turned out that there were issues regarding system scalability and efficiency [1]. Therefore, we modified the proposed system to enhance its efficiency and confirmed its improved scalability.

The first issue is the method of record detection. When a large number of denials are incurred, all their records are stored in an event log. The previous system maintained detected denial records as history. In each detection, records that the history did not have were fetched as missed detection. When many denial records are incurred in the event log file, it causes a heavy system load that the proposed system might not identify all missed detections, and the situation cannot be handled correctly. To address this, we refined the detection method to improve efficiency and ensure scalability, significantly reducing system load and improving detection accuracy.

The second issue is the method of record filtering. The proposed system could have caused delays in system processing. Record filtering is done by the target user name and file name. In addition to access logs, a large number of other logs are generated in the system environment. Thus, it is necessary to filter the target records among them. In the previous study, the records were filtered by a component responsible for

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

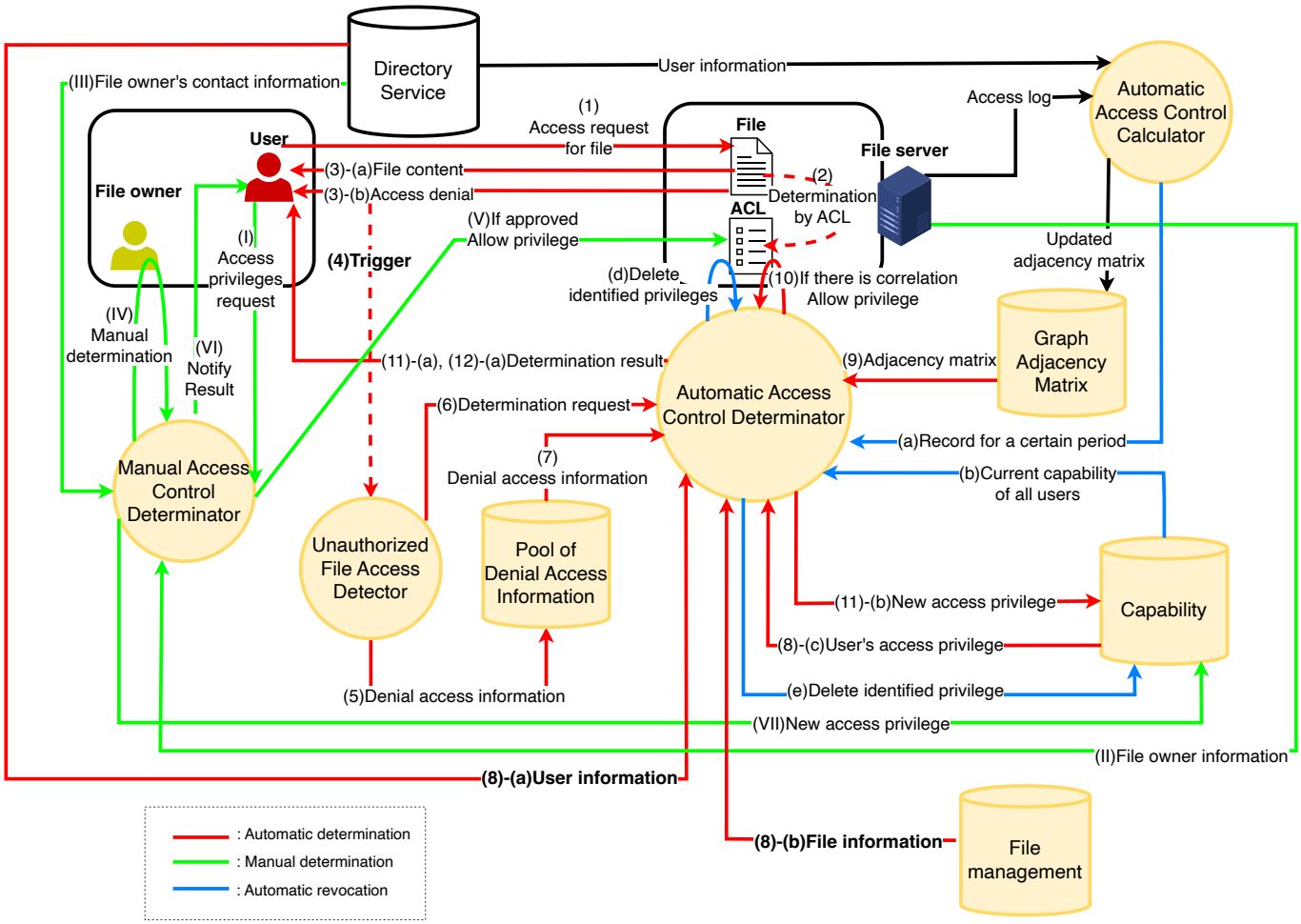


Fig. 5: Architecture of Modified System (yellow)

detection. If a large number of access denials occurred, the component was flooded with work. In the worst case, it could not be handled and could become fully dysfunctional. We have optimized the filtering process to handle high volumes more efficiently, preventing system delays and ensuring the component remains functional under increased load.

We modified the system's design regarding record management which traces the sequential number of the last record. The system thus retrieves the difference between the last and most recent records. Each record is assigned a sequential number by the operating system. By collecting not only the latest record but also the records representing these differences, the system prevents any missing denial records.

We also modified the system's design regarding record filtering for the detection component to specialize only in the detection of denial records. The detection component was detecting denial records and filtering whether they were target records or not. We gave that role to a component for determination. Thus, the load on the detection component is lightened.

However, this modification is expected to increase the load on the determination component. Since the above modification could increase the independence of the detection and determination components, When the load increases, a twist could be devised within each component. For example, in the detection component, if the number of detections is greater than the preset maximum number, only the maximum number of detections will be made and subsequent detections will be made after the process is completed. In the determination component, it is possible to record the determinations made during a certain period of time in the past and not determine the access types of the same user for the same file during that period of time. In this way, it could reduce the number of determinations.

These modifications have significantly improved the proposed system's efficiency and scalability, effectively addressing the initial concerns identified during the testing phase.

F. Architecture of Modified System

The architecture of modified system is shown in Fig. 5. The first change from the previous study is Step (4) in automatic determination of access privileges. After being triggered by denied events, the system collects attempts of denied access that have occurred since the last data collection, thereby

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

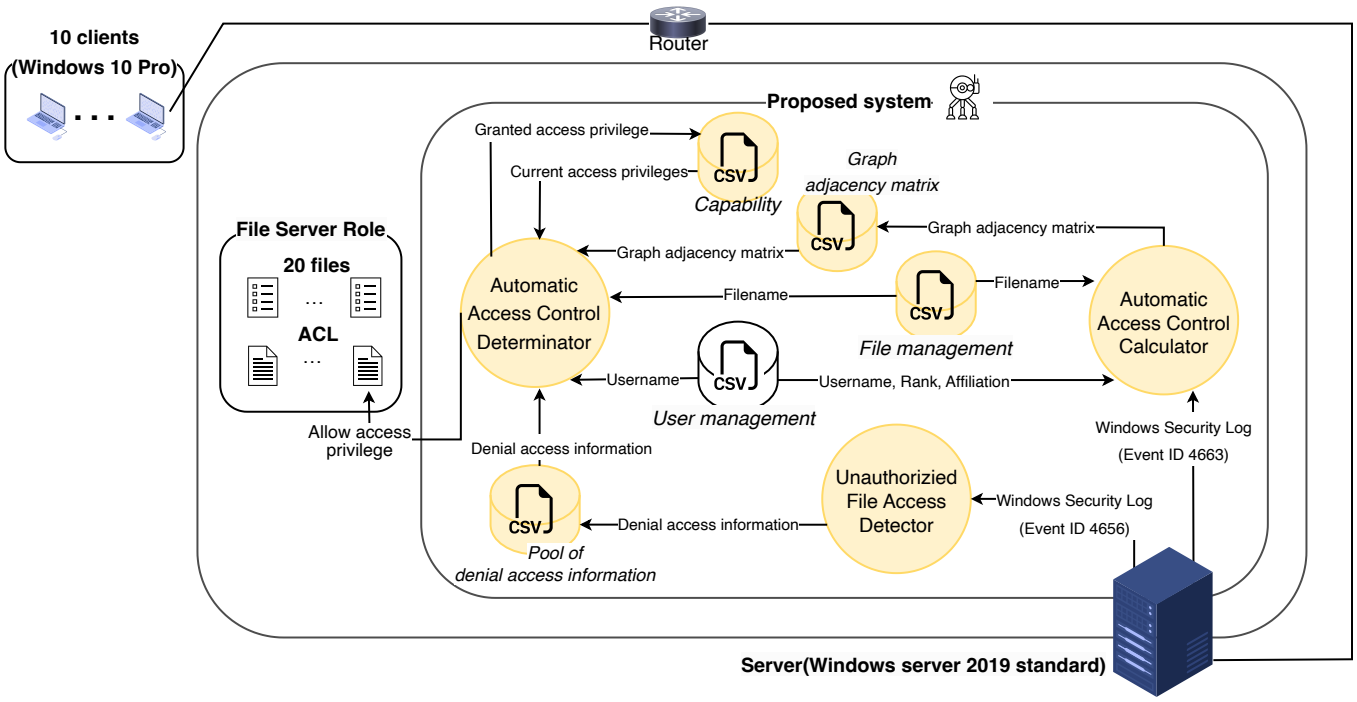


Fig. 6: Implementation of Proposed System (yellow)

ensuring that simultaneous access denials are also detected without omission. The second change is Step (8)-(a), (b) in automatic determination of access privileges. The role of log filtering is changed from the UFAD to the AACD, which lightens the workload of the UFAD and concentrates the tasks within the AACD, thereby ensuring scalability through efficient processing. The AACD executes record filtering on the basis of user and file information. If applicable, determinations are made; otherwise, the AACD deletes the access information without making a determination.

IV. Implementation of Proposed System

The proposed system was implemented in a more practical environment than previous work [1], because the previous environment was a simplified one, created on a single operating system on a single computer. We extended the system environment to address this limitation. The implemented environment is shown in Fig. 6. Ten machines were prepared to function as clients. Windows 10 Pro was installed as the operating system on each client machine. One user was assigned to each client machine. A single server was set up to serve both as a file server and to host the proposed system. Windows Server 2019 Standard was chosen as the operating system for the server. Twenty files, named from 00 to 19, were created on the file server. Each of the four DBs in the proposed system consists of CSV files.

The access log in Windows is “Security” in “Windows Log” (hereinafter referred to as Windows Security Log).

The following is a description of the setting for the system and the implementation of each component of the system.

A. Automatic Access Control Calculator (AACC)

This component has the role of graph calculation to represent the correlations among files and is implemented by Python in addition to the PowerShell command for fetching the Windows Security Log from the file server. It executes “Import-Csv” command to import Username from CSV File for User management and Filename from CSV File that stores target file information. It executes “Get-WinEvent” command to fetch the Windows Security Log from the file server. From the log, the “Where-Object” command retrieves 4663 event ID records of the target Username, Filename in the past certain period. The record with Event ID 4663 in the Windows Security Log indicates an attempt was made to access an object. Therefore, it extracts such records from the log. Then, it fetches information on Timestamp, Username, Filename, Accesstype (R or W) in the retrieved record. It executes “Sort-Object” command to sort the information based on Username and Timestamp.

For graph calculations, The “read_csv” function from the “pandas” library imports the user’s Rank from CSV File for User management. Considering the rank, the sorted information is classified for each rank for each operation (R or W) determination graph. Using Algorithm 1, the graphs of each operation in each rank is calculated from the classified information. The “to_csv” function exports the graphs to the CSV Files for Graph adjacency matrix.

The graph calculation was set up in two stages. The first stage is calculated at 1:00 a.m. daily using data from 30 days prior to the previous day. The second stage is calculated every hour during business hours (9:00-17:00), using data up to the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

present time of the day. After the second stage of calculation, the graphs from the first and second stages were combined and normalized. This is because graph calculation takes a lot of time.

B. Unauthorized File Access Detector (UFAD)

In the file server, "Task Scheduler" monitors the Windows Security Log and checks records with Event ID 4656. The record with this Event ID means the request to an object, and it includes the result of the request. Task Scheduler executes the UFAD component implemented by Windows PowerShell script when the request failure record is detected.

The script consists of four PowerShell commands. Firstly, it executes "Get-WinEvent" command similar to the AACC. From the log, the "Where-Object" command searches the request failure record. Timestamp, Username, Filename, and Access type (R or W) are extracted from the failure record by the "Select-Object" command. Finally, "Export-Csv" command saves extracted data to a CSV file that serves as Pool of Denial Access Information DB.

C. Automatic Access Control Determinator (AACD)

This component performs automatic access determination using denied access information and is implemented in Python. The "read_csv" function similar to the AACC imports Username and Filename from the oldest denied access information stored in the CSV file for pool of denied access information. The function imports Username from CSV File for User management and Filename from CSV File that stores target file information. If Usename and Filename of the denied access information match with the target Username and Filename, the code fetch Username, Filename, Accesstype from the oldest denied access information. If they do not match, delete them from CSV file for pool of denied access information without determination.

The "read_csv" function imports the user's Rank from CSV File for User management. The function imports the user's access privileges from CSV File for Capability. The function imports the rank's adjacency matrix from CSV File for graph adjacency matrix. The code determines if the maximum value in the adjacency matrix exceeds the threshold value, where the row is the filename stored in the CSV File for Capability and the column is the Filename of the denied access information.

If the result of the determination is to allow, in the Python code, the "os.system ()" function is used to execute the Windows "icacls" command, which allows the user the access privilege for the file. The "socket" function from the "socket" library create a new socket object. The "connect" method of the socket object s connects to the specified IP address (HOST) and port number (PORT). The "send" method transmits a message to the connected server. This message includes Timestamp, Username,Filename and Accesstype (R or W). The "csv.writer" function from the "csv" library write Username, Filename and Accesstype (R or W) into CSV file for Capability as new access privilege.

If the result of the determination is to deny, a message is sent using the 'socket' function as in the case of allowed privileges. This message includes Timestamp, Username, Filename and Accesstype (R or W). Additionally, the user has the option to submit a new access privilege request along with the reason for the request.

Regardless of the result of either determination, the oldest denied access information is deleted. As long as data exists in the CSV file for pool of denied access information, the determination is repeated.

On each user's client side, Python code utilizing the "socket" function from the "socket" library is constantly running to receive the determination results. The "listen" method of the socket object s place the socket in server mode and wait for incoming connections. If data is received, the content of the message is displayed in the GUI using the "messagebox" from the "tkinter" library. Each client user is distinguished by IP address, and port number 8000 is reconfigured to open for this implementation.

D. CSV File for User Management

This file stored the user information in the following three columns.

- Username
- Rank
- Affiliation

E. CSV File that stores Target File Information

This file stored target file information in the following one column.

- Filename

F. CSV File for Pool of Denied Access Information

This file stored the denied log information in the following four columns.

- Timestamp
- Username
- Filename
- Accesstype (R or W)

G. CSV Files for Graph Adjacency Matrix

These files stored the adjacency matrices of the graph calculated by AACC. There exist two types of adjacency matrices for each rank, corresponding to two access types (R or W).

H. CSV File for Capability

This file stores information about the accesstype (R, R/W) for which the user has privileges to access the file.

- Username
- Filename
- Accesstype (R or W)

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE III: PRELIMINARY EXPERIMENTAL RESULTS: RESPONSE TIME

	Response Time (s)						
Number of Users	1st Trial	2nd Trial	3rd Trial	4th Trial	5th Trial	5-Trial Average	Average per User
Original System							
1(A)	6.85	6.22	6.62	5.96	6.69	6.47	6.47
2(A/B)	7.96	7.92	6.51	6.71	7.82	7.38	3.69
3(A/B/C)	7.91	9.56	7.34	9.49	8.58	8.58	2.86
4(A/B/C/D)	8.61	8.26	8.35	8.79	8.16	8.43	2.11
5(A/B/C/D/E)	8.79	8.95	9.56	9.10	9.46	9.17	1.83
6(A/B/C/D/E/F)	11.20	12.77	9.91	12.54	11.84	11.65	1.94
Modified System							
1(A)	6.46	5.19	5.16	6.58	4.94	5.58	5.58
2(A/B)	5.46	5.50	5.64	4.76	4.83	5.24	2.61
3(A/B/C)	6.04	5.32	5.68	5.24	8.95	6.25	2.08
4(A/B/C/D)	10.41	5.22	5.62	5.39	5.82	6.49	1.62
5(A/B/C/D/E)	6.17	10.61	5.83	6.19	5.15	6.79	1.36
6(A/B/C/D/E/F)	6.00	10.62	6.11	5.78	6.44	6.99	1.16

V. Evaluation Experiment

The main objective of this paper was to verify the feasibility of the modified system in a real-world setting. The initial method of verification, which used a limited number of users and accesses, was found to be inadequate for a comprehensive evaluation. To address this, we implemented the improved system in a more practical environment and subjected it to high stress for feasibility testing. This was done to demonstrate that the proposed system, even when integrating processes from existing file systems, does not lose its feasibility.

A. Preliminary Experiment

We conducted preliminary experiments to compare the performance of both versions of our system (original and modified). Performance was compared by conducting the verification under the same conditions as those used in our previous study [1].

1) System Environment: Both version of the system were implemented in a experimental environment. We used Intel® NUC 8 Pro Kit (NUC8v7PNH) as the hardware and Windows 11 Pro as the operating system (OS). We set up six users (A$\cdots$F) on the OS and ten files to be accessed by the user.

2) Methodology: We measured the determination time of the both systems as a performance test. We measured the response time from when users access unauthorized files to when determination results are notified to the users.

3) Results: The results of the verification experiments are listed in Table III. Number of Users columns indicates the number and name of accessing users. The columns 1st Trial through 5th trial show the response times for each number of users. If the number of users is 2 or more, the latest response time among users is noted. The 5-trial Average column shows the average of five trials for the same number of users. The average per user is calculated by dividing the five-trial average by the number of users in the corresponding row.

The results indicate improved response times for both 5-trial average and average per user. Despite the preliminary experiment not supposing high-load conditions, the improvement in the response time was observed.

B. Experimental Environment

We built a more practical environment to verify the scalability of the modified system. The system environment is shown in Fig. 7. Ten machines were prepared as clients. Each client had Windows 10 as its operating system and segmented on the network into subgroups. Each client machine was assigned to one user, with each user being named from user_A to user_J. One server was also prepared with the active directory, file server, and the proposed system. Twenty files with file names 00 through 19 were created on the file server.

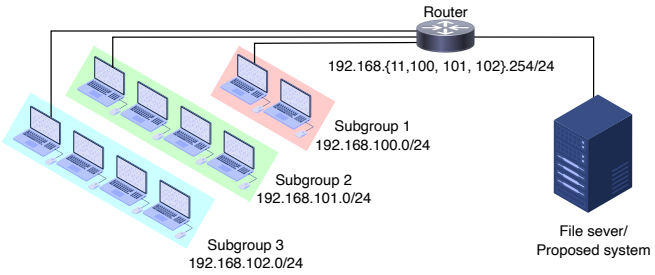


Fig. 7: Experimental Environment

Table IV shows the privileges allowed to each user for each file. Each user was assigned a job rank. As noted next to the user name in the user column of Table IV, we assigned rank 2 to users A, D, and H, and assigned rank 1 to the rest. It is assumed that rank 2 is higher than rank 1. Each user's rank is taken into account when calculating the graph.

C. Dataset

Graph calculation requires data on each user's access history. A simulated data set was generated for this experiment. The following steps were taken to generate the simulated data set.

1) Set the column items of the simulated data set to be generated
   - Timestamp
   - Username
   - Filename

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE IV: ACCESS PRIVILEGES FOR EACH USER IN A PRACTICAL ENVIRONMENT

UserName	File_00	File_01	File_02	File_03	File_04	File_05	File_06	File_07	File_08	File_09
User_A (2)			R	R	R		R	R/W	R/W	R/W
User_B (1)			R/W	R/W	R/W		R/W			R/W
User_C (1)			R/W	R/W	R/W		R/W			R/W
User_D (2)		R		R		R	R	R	R/W	
User_E (1)		R/W		R/W		R/W	R/W			
User_F (1)		R/W		R/W		R/W	R/W			
User_G (1)		R/W		R/W		R/W	R			
User_H (2)	R				R	R	R/W	R/W	R/W	R
User_I (1)	R/W				R/W	R/W	R/W			R
User_J (1)	R/W				R/W	R/W	R/W			R

UserName	File_10	File_11	File_12	File_13	File_14	File_15	File_16	File_17	File_18	File_19
User_A (2)	R			R			R			R
User_B (1)	R			R/W			R/W	R		R/W
User_C (1)	R		R/W			R/W		R	R/W	
User_D (2)	R/W		R/W		R	R		R	R	
User_E (1)	R/W				R/W	R/W				R/W
User_F (1)	R/W		R		R				R/W	
User_G (1)	R/W					R	R/W	R/W		
User_H (2)		R/W	R	R/W		R	R	R/W	R	
User_I (1)		R/W	R		R		R		R	R
User_J (1)		R/W		R	R/W		R	R	R/W	

TABLE V: DATA SET SUMMARY FOR GRAPH CALCULATION (read)

User	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19
A(2)	0	0	17	18	21	0	17	38	35	36	22	0	0	22	0	0	18	0	0	18
B(1)	0	0	19	27	32	0	35	0	0	14	20	0	0	27	0	0	29	17	0	30
C(1)	0	0	20	30	25	0	33	0	0	28	13	0	31	0	0	27	0	10	26	0
D(2)	0	26	0	17	0	35	21	21	30	0	37	0	31	0	29	15	0	22	16	0
E(1)	0	25	0	27	0	24	31	0	0	0	32	0	0	0	33	23	0	0	0	37
F(1)	0	28	0	56	0	37	34	0	0	0	36	0	19	0	23	0	0	0	29	0
G(1)	0	33	0	45	0	26	16	0	0	0	34	0	0	0	0	23	33	40	0	0
H(2)	15	0	0	0	18	10	29	20	24	14	0	24	19	20	0	11	15	23	15	0
I(1)	26	0	0	0	34	32	25	0	0	15	0	22	25	0	20	0	20	0	11	20
J(1)	33	0	0	0	24	24	27	0	0	21	0	19	0	13	27	0	14	19	23	0

TABLE VI: DATA SET SUMMARY FOR GRAPH CALCULATION (write)

User	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19
A(2)	0	0	0	0	0	0	0	17	14	13	0	0	0	0	0	0	0	0	0	0
B(1)	0	0	7	11	12	0	13	0	0	5	0	0	0	16	0	0	14	0	0	12
C(1)	0	0	9	16	13	0	11	0	0	11	0	0	14	0	0	13	0	0	12	0
D(2)	0	0	0	0	0	0	0	0	15	0	19	0	12	0	0	0	0	0	0	0
E(1)	0	10	0	10	0	7	12	0	0	0	14	0	0	0	12	5	0	0	0	14
F(1)	0	12	0	22	0	21	12	0	0	0	20	0	0	0	0	0	0	0	14	0
G(1)	0	11	0	24	0	12	0	0	0	0	13	0	0	0	0	0	11	18	0	0
H(2)	0	0	0	0	0	0	12	5	13	0	0	11	0	7	0	0	0	11	0	0
I(1)	12	0	0	0	15	19	3	0	0	0	0	13	0	0	0	0	0	0	0	0
J(1)	7	0	0	0	12	7	6	0	0	0	0	8	0	0	9	0	0	0	10	0

- Accesstype (R or W)

2) Set the date and time of the simulated data set to be generated. In this study, past month (past 30 days).
3) Generate simulated log data set
   a) Set the time as 9:00–17:00
   b) Set the probability of generating a log every 15 min
      - On the hour: 70%
      - 15 min past the hour: 50%
      - 30 min past the hour: 30%
      - 45 min past the hour: 20%

      If not generated, move to the next hour (e.g., if not generate at 9:15 → 10:00)
   c) Select file
      i) Set weight to each file
         - File the user does not have privileges to: 0
         - File the user has read privileges to: 0.3
         - File the user has read/write privileges to: 0.5
      ii) Calculate sum of these weights per user
      iii) Generate a random number from 0 to total sum
      iv) Select the file in which the accumulated weight exceeds the random number for the first time
   d) Select access type
      If read is generated and the user has read/write privileges for the file, there is a 40% chance of generating write in the next 15 min
      If not, generate read at a probability in (b)
   e) Repeat the above steps for each user

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE VII: ADJACENCY MATRIX (rank1_read)

	00	01	02	03	04	05	06	09	10	11	12	13	14	15	16	17	18	19
00	0.00	0.00	0.00	0.00	0.10	0.09	0.14	0.19	0.00	0.18	0.13	0.15	0.14	0.00	0.22	0.14	0.10	0.11
01	0.00	0.00	0.00	0.18	0.00	0.27	0.20	0.00	0.27	0.00	0.00	0.00	0.13	0.12	0.08	0.17	0.15	0.06
02	0.00	0.00	0.00	0.17	0.18	0.00	0.12	0.09	0.01	0.00	0.20	0.20	0.00	0.11	0.08	0.09	0.13	0.13
03	0.00	0.18	0.17	0.00	0.08	0.18	0.30	0.08	0.23	0.00	0.19	0.14	0.25	0.22	0.21	0.14	0.14	0.17
04	0.10	0.00	0.18	0.08	0.00	0.10	0.21	0.11	0.26	0.03	0.17	0.13	0.16	0.10	0.10	0.21	0.21	0.14
05	0.09	0.27	0.00	0.18	0.10	0.00	0.26	0.16	0.10	0.27	0.13	0.00	0.17	0.11	0.23	0.07	0.19	0.16
06	0.14	0.20	0.12	0.30	0.21	0.26	0.00	0.23	0.25	0.12	0.27	0.12	0.16	0.11	0.18	0.30	0.14	0.14
09	0.19	0.00	0.09	0.08	0.11	0.16	0.23	0.00	0.05	0.19	0.10	0.04	0.00	0.07	0.11	0.09	0.18	0.14
10	0.00	0.27	0.01	0.23	0.26	0.10	0.25	0.05	0.00	0.00	0.04	0.05	0.19	0.15	0.15	0.10	0.06	0.18
11	0.18	0.00	0.00	0.00	0.03	0.27	0.12	0.19	0.00	0.00	0.00	0.00	0.14	0.00	0.10	0.15	0.17	0.02
12	0.13	0.00	0.20	0.19	0.17	0.13	0.27	0.10	0.04	0.00	0.00	0.00	0.10	0.14	0.03	0.07	0.15	0.09
13	0.15	0.00	0.20	0.14	0.13	0.00	0.12	0.04	0.05	0.00	0.00	0.00	0.15	0.00	0.11	0.01	0.10	0.18
14	0.14	0.13	0.00	0.25	0.16	0.17	0.16	0.00	0.19	0.14	0.10	0.15	0.00	0.23	0.03	0.02	0.15	0.20
15	0.00	0.12	0.11	0.22	0.10	0.11	0.11	0.07	0.15	0.00	0.14	0.00	0.23	0.00	0.05	0.07	0.05	0.13
16	0.22	0.08	0.08	0.21	0.10	0.23	0.18	0.11	0.15	0.10	0.03	0.11	0.03	0.05	0.00	0.22	0.09	0.09
17	0.14	0.17	0.09	0.14	0.21	0.07	0.30	0.09	0.10	0.15	0.07	0.01	0.02	0.07	0.22	0.00	0.02	0.05
18	0.10	0.15	0.13	0.14	0.21	0.19	0.14	0.18	0.06	0.17	0.15	0.10	0.15	0.05	0.09	0.02	0.00	0.00
19	0.11	0.06	0.13	0.17	0.14	0.16	0.14	0.14	0.18	0.02	0.09	0.18	0.20	0.13	0.09	0.05	0.00	0.00

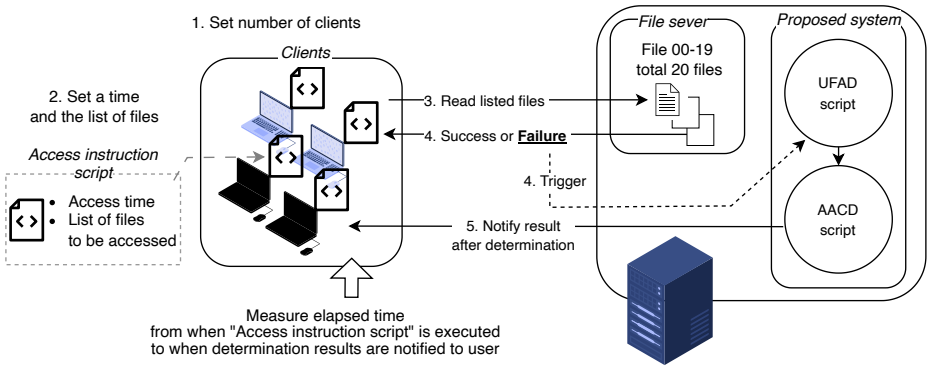


Fig. 8: Experimental Methodology

A summary of the dataset exists. The summary concerning read is shown in Table V, while the summary about write is shown in Table VI.

Graphs for access determination are calculated from the generated access logs. These graphs are shown across multiple tables. Graphs related to rank 1 are shown in Table VII for read determination and in Table VIII for write determination. Similarly, graphs related to rank 2 are displayed in Table IX for read determination and in Table X for write determination. The method for creating graphs is described in the function of the AACC. The graphs for read determination are calculated from the access histories of the same or lower rank users. The higher rank users tend to have more files represented on the graph compared to the lower rank users. The graphs for write determination are calculated from the access histories of the same rank users. As the number of the same rank users decreases for higher ranks, there is a higher likelihood of having fewer files represented on the graph.

D. Methodology

The experimental methodology is shown in Fig. 8. Each client has an "Access instruction script" for accessing files in the file server. We can set a time of access and a list of accessing files to the script. The verification experiment was conducted using the following steps.

1) Set the number of clients
   (Choosing from 1 to 10 clients)
2) Set the time and files to be accessed
   (Selecting 1 to 20 files) to the scripts of selected clients
3) Clients read all files at once when time is up by the script
4) The File server detects denial logs of accesses and executes the UFAD
5) The AACD makes determinations based on the logs and notifies the user

At each client, we measured the elapsed time as a response time from the Access instruction script starts to it receives the notification.

Total Access is the number of accessing users multiplied by the number of files listed. For example, a total of 100 accesses is 10 users accessing 10 files from 00 to 09. The number of accesses was increased from 1 to 10 in increments of 1. After 10, the number of accesses was increased in increments of 10 up to 200.

Accesses were done sequentially to the listed files. Although not strictly simultaneous accesses, the next files were accessed with 0.2-s delay. This condition simulated an intense load on our modified system. When the maximum number of access

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE VIII: ADJACENCY MATRIX (rank1_write)

	00	01	02	03	04	05	06	09	10	11	12	13	14	15	16	17	18	19
00	0.00	0.00	0.00	0.00	0.19	0.33	0.20	0.00	0.00	0.39	0.00	0.00	0.00	0.00	0.00	0.00	0.23	0.00
01	0.00	0.00	0.00	0.14	0.00	0.25	0.26	0.00	0.46	0.00	0.00	0.00	0.11	0.00	0.00	0.43	0.21	0.29
02	0.00	0.00	0.00	0.04	0.18	0.00	0.28	0.01	0.00	0.00	0.21	0.36	0.00	0.09	0.17	0.00	0.23	0.37
03	0.00	0.14	0.04	0.00	0.05	0.45	0.36	0.41	0.30	0.00	0.34	0.43	0.10	0.19	0.54	0.29	0.34	0.09
04	0.19	0.00	0.18	0.05	0.00	0.29	0.09	0.21	0.00	0.16	0.17	0.25	0.02	0.06	0.19	0.00	0.36	0.00
05	0.33	0.25	0.00	0.45	0.29	0.00	0.32	0.00	0.00	0.56	0.00	0.00	0.00	0.00	0.30	0.21	0.06	0.23
06	0.20	0.26	0.28	0.36	0.09	0.32	0.00	0.15	0.28	0.19	0.13	0.10	0.22	0.00	0.09	0.00	0.23	0.28
09	0.00	0.00	0.01	0.41	0.21	0.00	0.15	0.00	0.00	0.00	0.01	0.00	0.00	0.29	0.04	0.00	0.19	0.00
10	0.00	0.46	0.00	0.30	0.00	0.00	0.28	0.00	0.00	0.00	0.00	0.00	0.00	0.48	0.08	0.22	0.04	0.01
11	0.39	0.00	0.00	0.00	0.16	0.56	0.19	0.00	0.00	0.00	0.00	0.00	0.41	0.00	0.00	0.00	0.00	0.00
12	0.00	0.00	0.21	0.34	0.17	0.00	0.13	0.01	0.00	0.00	0.00	0.00	0.00	0.41	0.00	0.00	0.22	0.00
13	0.00	0.00	0.36	0.43	0.25	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.14	0.00	0.00	0.00
14	0.00	0.11	0.00	0.10	0.02	0.00	0.22	0.00	0.00	0.41	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.44
15	0.00	0.00	0.09	0.19	0.06	0.00	0.00	0.29	0.48	0.00	0.41	0.00	0.00	0.00	0.00	0.00	0.10	0.28
16	0.00	0.00	0.17	0.54	0.19	0.30	0.09	0.04	0.08	0.00	0.00	0.14	0.00	0.00	0.00	0.17	0.00	0.00
17	0.00	0.43	0.00	0.29	0.00	0.21	0.00	0.00	0.22	0.00	0.00	0.00	0.00	0.00	0.17	0.00	0.00	0.00
18	0.23	0.21	0.23	0.34	0.36	0.06	0.23	0.19	0.04	0.00	0.22	0.00	0.00	0.10	0.00	0.00	0.00	0.00
19	0.00	0.29	0.37	0.09	0.00	0.23	0.28	0.00	0.01	0.00	0.00	0.00	0.44	0.28	0.00	0.00	0.00	0.00

TABLE IX: ADJACENCY MATRIX (rank2_read)

	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19
00	0.00	0.00	0.00	0.00	0.13	0.07	0.16	0.03	0.08	0.14	0.00	0.14	0.11	0.10	0.11	0.02	0.19	0.11	0.09	0.09
01	0.00	0.00	0.00	0.19	0.00	0.21	0.16	0.07	0.10	0.00	0.24	0.00	0.03	0.00	0.10	0.13	0.06	0.16	0.17	0.05
02	0.00	0.00	0.00	0.16	0.18	0.00	0.09	0.04	0.15	0.12	0.04	0.00	0.13	0.16	0.00	0.07	0.09	0.06	0.09	0.09
03	0.00	0.19	0.16	0.00	0.06	0.14	0.26	0.07	0.06	0.11	0.20	0.00	0.17	0.07	0.21	0.16	0.20	0.13	0.13	0.17
04	0.13	0.00	0.18	0.06	0.00	0.08	0.22	0.10	0.06	0.11	0.24	0.09	0.12	0.15	0.12	0.07	0.13	0.17	0.17	0.19
05	0.07	0.21	0.00	0.14	0.08	0.00	0.21	0.08	0.06	0.12	0.16	0.19	0.15	0.01	0.25	0.13	0.18	0.09	0.20	0.12
06	0.16	0.16	0.09	0.26	0.22	0.21	0.00	0.14	0.15	0.21	0.22	0.15	0.22	0.14	0.13	0.14	0.15	0.26	0.11	0.12
07	0.03	0.07	0.04	0.07	0.10	0.08	0.14	0.00	0.17	0.17	0.06	0.06	0.09	0.12	0.08	0.04	0.08	0.03	0.09	0.03
08	0.08	0.10	0.15	0.06	0.06	0.06	0.15	0.17	0.00	0.09	0.16	0.07	0.10	0.16	0.05	0.05	0.08	0.06	0.08	0.05
09	0.14	0.00	0.12	0.11	0.11	0.12	0.21	0.17	0.09	0.00	0.06	0.14	0.07	0.12	0.00	0.05	0.13	0.08	0.12	0.16
10	0.00	0.24	0.04	0.20	0.24	0.16	0.22	0.06	0.16	0.06	0.00	0.00	0.10	0.06	0.21	0.14	0.13	0.17	0.07	0.16
11	0.14	0.00	0.00	0.00	0.09	0.19	0.15	0.06	0.07	0.14	0.00	0.00	0.05	0.05	0.09	0.04	0.15	0.12	0.14	0.01
12	0.11	0.03	0.13	0.17	0.12	0.15	0.22	0.09	0.10	0.07	0.10	0.05	0.00	0.02	0.13	0.17	0.02	0.09	0.19	0.06
13	0.10	0.00	0.16	0.07	0.15	0.01	0.14	0.12	0.16	0.12	0.06	0.05	0.02	0.00	0.08	0.06	0.11	0.05	0.11	0.12
14	0.11	0.10	0.00	0.21	0.12	0.25	0.13	0.08	0.05	0.00	0.21	0.09	0.13	0.08	0.00	0.17	0.02	0.02	0.11	0.17
15	0.02	0.13	0.07	0.16	0.07	0.13	0.14	0.04	0.05	0.05	0.14	0.04	0.17	0.06	0.17	0.00	0.03	0.13	0.05	0.09
16	0.19	0.06	0.09	0.20	0.13	0.18	0.15	0.08	0.08	0.13	0.13	0.15	0.02	0.11	0.02	0.03	0.00	0.21	0.08	0.09
17	0.11	0.16	0.06	0.13	0.17	0.09	0.26	0.03	0.06	0.08	0.17	0.12	0.09	0.05	0.02	0.13	0.21	0.00	0.05	0.04
18	0.09	0.17	0.09	0.13	0.17	0.20	0.11	0.09	0.08	0.12	0.07	0.14	0.19	0.11	0.11	0.05	0.08	0.05	0.00	0.00
19	0.09	0.05	0.09	0.17	0.19	0.12	0.12	0.03	0.05	0.16	0.16	0.01	0.06	0.12	0.17	0.09	0.09	0.04	0.00	0.00

TABLE X: ADJACENCY MATRIX (rank2_write)

	06	07	08	09	10	11	12	13	17
06	0.00	0.37	0.00	0.00	0.00	0.00	0.00	0.00	0.83
07	0.37	0.00	0.65	1.17	0.00	0.45	0.00	0.00	0.08
08	0.00	0.65	0.00	0.36	0.41	0.44	0.36	0.57	0.37
09	0.00	1.17	0.36	0.00	0.00	0.00	0.00	0.00	0.00
10	0.00	0.00	0.41	0.00	0.00	0.00	1.46	0.00	0.00
11	0.00	0.45	0.44	0.00	0.00	0.00	0.00	0.00	0.66
12	0.00	0.00	0.36	0.00	1.46	0.00	0.00	0.00	0.00
13	0.00	0.00	0.57	0.00	0.00	0.00	0.00	0.00	0.82
17	0.83	0.08	0.37	0.00	0.00	0.66	0.00	0.82	0.00

instances, which is 200, was reached, 200 accesses occurred within about 4 seconds.

The determination was made using graphs that were already calculated. In fact, the graphs could be recalculated at the same time as the determination. However, to align the experimental conditions, the graphs were generated under fixed conditions. Therefore, the determination time is not affected by the load due to recalculation.

E. Result

The results are listed in Table XI. Total Access is the total number of files listed in each user's script file. Unauthorized Access (UA) is access to files for which a user does not have read privileges among all accesses. Detected UAs are accesses detected with our system among UAs. Missed detection is the number of cases that our system failed to detect even though access logs were generated. Double detection is the number of cases in which our system detected double access logs for a single access log. The first response time refers to the response time of the proposed system for the first UA. Similarly, the

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE XI: EXPERIMENTAL RESULTS

Total Access (cases)	Unauthorized Access (cases)	Detected UA (cases)	Missed Detection (cases)	Double Detection (cases)	First Response Time (s)	Last Response Time (s)	Average Response Time (s)
1	1	1	0	0	2.97	2.97	2.97
10	7	7	0	0	3.22	3.27	3.24
50	27	27	0	0	4.05	5.14	4.50
100	48	48	0	0	5.03	8.32	7.06
150	74	74	0.67	0.67	4.73	10.24	8.87
200	97	97.33	0.33	0.67	5.97	13.56	12.05

last response time refers to the response time of the proposed system for the last UA. The average response time is calculated as the mean of the response times for each determination made with the system. The same trial was conducted three times for each total accesses. Each value in Table XI represents the average of three trials. Therefore, some values are decimal.

F. Discussion

From the results of our verification, we highlight three key points regarding our modified system. The first point concerns its scalability. The total number of accesses was varied from 1 to 200. The verification was conducted under conditions that placed a heavy burden on the system, with approximately half of the accesses being unauthorized. As a result, we achieved a detection rate of over 98% for up to 200 cases, despite some cases of missed or double detections.

The second point concerns the efficiency of the system. The average response time was 2.97s with a single access. In contrast, when the total number of accesses reached 200, the average response time increased to 12.05s. Even though the number of accesses went up by 200 times, the processing time only increased 4 times. This suggests that our system operates efficiently under heavy-load conditions.

The third point concerns missed and double detection. Missed and double detection began to occur after the total number of accesses exceeded 80. Although those numbers are less than 2%, they need to be corrected to improve the detection accuracy.

1) Limitation: We verified the efficiency and scalability of our modified access-control system, but not its effectiveness. It is necessary to show that our system could allow necessary access and deny unnecessary access. Additionally, the proposed method is considered to work well in team-oriented tasks, but its effectiveness and the optimal team size for its application have not been proven.

The modified system infers correlation among files only from user access patterns. There is still potential to investigate whether additional factors could be incorporated to infer more accurate correlations.

VI. Conclusion and Future Work

We implemented a modified version of our previously proposed access-control system in a practical environment to verify its scalability by applying a high load to it. This is because the previous implementation and verification were done in a simplified environment.

The verification results indicate that the system works under high loads. Comparing the rate of increase in the number of accesses to that in response time, the low rate of increase in response time indicates that the system is capable of efficient processing.

The evaluation results indicate that there are still some detection errors and double detections, so that the system needs further improvement to increase detection accuracy. The feasibility of this system has not yet been validated, and there is still potential for improvement to infer correlations. To address these issues, we will evaluate the effectiveness of the proposed method in teams of different sizes. Therefore, we plan to further improve this system from this perspective.

Acknowledgment

This work was partially supported by JSPS KAKENHI Grant Numbers JP19K20268, JP24K14959.

References

[1] Y. Kodaka, H. Hasegawa, and H. Takakura, “Design and implementation of access control method based on correlation among files,” in Proceedings of The 16th International Conference on Advances in Human oriented and Personalized Mechanisms, Technologies, and Services, pp. 44-51, 2023.

[2] P. Samarati and S. C. Vimercati, “Access control: policies, models, and mechanisms,” Foundations of Security Analysis and Design, R. Focardi, R. Gorrieri, ed., Springer, pp. 137-196, 2001.

[3] D. F. Ferraiolo and D. R. Kuhn, “Role-based access control,” 15th National Computer Security Conference, pp. 554-563, 1992.

[4] V. C. Hu et al., “Guide to Attribute Based Access Control (ABAC) definition and considerations,” U.S. Department of Commerce, 2014.

[5] H. Xia, M. Dawande, and V. Mookerjee, “Role refinement in access control: model and analysis,” INFORMS Journal on Computing vol. 26, no. 4, pp. 866-884, 2014.

[6] L. Bauer, L. F. Cranor, R. W. Reeder, M. K. Reiter, and K. Vaniea, “Real life challenges in access-control management,” in Proceedings of the CHI Conference on Human Factors in Computing Systems, Association for Computing Machinery, pp. 899-908, 2009.

[7] D. E. Bell and L. J. LaPadula, “Secure computer systems: mathematical foundation report ESD-TR-73-275,” MITRE Corp., 1973.

[8] Ponemon Institute, “2022 cost of insider threats global report,” Proofpoint, 2022.

[9] D. Tsiostas, and N. Chouliaras, and I. Kantzavelou, and L. Maglaras, and C. Douligeris, and V. Vlachos, “The insider threat: reasons, effects and mitigation techniques,” in 24th Pan-Hellennic Conference on Informatics, pp. 340-345, 2020.

[10] Y. Kodaka, H. Hasegawa, and H. Takakura, “A proposal for access control method based on file relation inference from users behavior (in Japanese),” IEICE Technical Report vol. 123, no. 86, pp. 40-47, 2023.

[11] T. Xu, H. M. Naing, L. Le and Y. Zho, “How do system system administrators resolve access-denied issues in the real world?,” in Proceedings of the CHI Conference on Human Factors in Computing Systems, pp. 348-361, 2017.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[12] M. Beckerle and L. A. Martucci, "Formal definitions for usable access control rule sets from goals to metrics," in Proceedings of the Ninth Symposium on Usable Privacy and Security, pp. 1-11, 2013.

[13] M. Mazurek, and P. Klemperer, and R. Shay, and H. Takabi, and L. Bauer, and L.Cranor, "Exploring reactive access control," in Proceedings of the CHI Conference on Human Factors in Computing Systems, Association for Computing Machinery, pp. 2085-2094, 2011.

[14] N. Shalev, I. Keidar, Y. Weinsberg. Y. Moatti, and E. B. Yehuda, "WatchIT: who watches your IT guy," in Proceedings 26th Symposium on Operating Systems Principles, pp. 515-530, 2017.

[15] Y. Desmedt and A. Shaghaghi, "Function-based access control (FBAC) from access control matrix to access control tensor," From Database to Cyber Security, vol 11170, pp. 143-165, 2018.

[16] H. Mannila, H. Toivonen, and A. Verkamo, "Discovery of frequent episodes in event sequence," Data Mining and Knowledge Discovery, vol. 1, pp. 259-289, 1997.

[17] B. Camina, R. Monroy, L. Trejo, and E. Sanchez, "Towards building a masquerade detection method based on user file system navigation," in Proceedings of the 10th Mexican International Conference on Artificial Intelligence, Lecture Notes in Computer Science, vol. 7094, pp. 174-186, 2011.

[18] B. Camina, J. Rodriguez, and R. Monroy, "Towards a masquerade detection system based on user's tasks," in Proceedings of the 17th International Symposium on Research in Attacks, Intrusions and Defenses, Lecture Notes in Computer Science, vol. 8688, pp. 447-465, 2014.

[19] S. Huang, Z. Cao, C. Raines, M. Yang, and C. Simon, "Detecting intruders by user fie access patterns," in Proceedings of the 13th International Conference on Network and System Security, Lecture Notes in Computer Science, vol. 11928, pp. 320-335, 2019.

[20] S. Mehnaz and E. Bertino, "A fine grained approach for anomaly detection in file system accesses with enhanced temporal user profiles," IEEE Transactions on Dependable and Secure Computing, vol. 18, Issue 6, pp. 2535-2550, 2021.

[21] Y. Chen and B. Malin, "Detection of anomalous insiders in collaborative environments wia relational analysis of access logs," in Proceedings of the 1st ACM conference on Data and application security and privacy, pp. 63-74, 2011.

2024, © Copyright by authors, Published under agreement with IARIA - www.iaria.org