TU WIEN Informatics

A QR-Code Optical Covert Channel in an Air-Gapped Secure Data Infrastructure

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Martin Weise, BSc
Matrikelnummer 01429167

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Andreas Rauber, Ao.Univ.Prof. Dipl.-Ing. Dr.techn.

Wien, 9. Dezember 2021

Martin Weise	Andreas Rauber

TU WIEN Informatics

A QR-Code Optical Covert Channel in an Air-Gapped Secure Data Infrastructure

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Martin Weise, BSc
Registration Number 01429167

to the Faculty of Informatics

at the TU Wien

Advisor: Andreas Rauber, Ao.Univ.Prof. Dipl.-Ing. Dr.techn.

Vienna, 9th December, 2021

Martin Weise	Andreas Rauber

Erklärung zur Verfassung der Arbeit

Martin Weise, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 9. Dezember 2021

Martin Weise

Acknowledgements

I want to thank my supervisor Ao.Univ.Prof. Dipl-Ing. Dr.techn. Andreas Rauber for the reliable supervision, the valuable input for writing this thesis and the open ear when stumbling across problems anytime. His guidance in challenging situations and valuable knowledge to solve any organizational- or technical problem made this thesis possible, as well as the related papers we have co-written.

Additional thanks goes to Peter Spieler for his valuable technical input and large professional experience in the set-up of the predecessor infrastructure of OSSDIP.

This thesis was made possible, thanks to the funding of the reference implementation as described from the EOSC-Secretariat program call H2020-INFRAEOSC-05-2018-2019 under grant agreement number 831644.

Further, the development and improvement of the reference implementation is made possible thanks to the EOSC-Life program call H2020-INFRAEOSC-2018-2 under grant agreement number 824087.

Kurzfassung

Kontext Die gegensätzlichen Ziele über Schutz und Erhalt der Kontrolle über sensitive Daten, bei gleichzeitigem Gewähren des Zugriffs auf die Daten für Dritte, ist eine Herausforderung. Sichere Dateninfrastrukturen unterstützen Datenbesuche in einer hoch kontrollierten und überwachten Umgebung die, sofern angemessen aufgesetzt und betrieben, hohe Sicherheitsgarantien durch Kombination von technischen, rechtlichen und prozeduralen Mechanismen bereitstellen kann.

Methoden In dieser Arbeit evaluieren wir zu welchen Grad ein identifizierter Analytiker in der Lage ist, Daten aus der sicheren Dateninfrastruktur zu schleusen mittels eines optischen, versteckten Kanals. Mithilfe der geteilt-verwendeten Ressourcen-Matrix-Methode erhalten wir mögliche Ressourcen die als optischer, verdeckter Kanal verwendbar sind und in einem weiteren Schritt im Hinblick auf Bandbreite und Fehlerrate untersucht werden. Der Kanal arbeitet sensitive Daten in unauffällige Bilder ein, die sich für das menschliche Auge nicht von den Originalbildern (ohne diese Bildermanipulation) unterscheiden.

Ergebnisse Wir schotten sensitive Daten in unserer Referenzimplementierung einer sicheren Dateninfrastruktur vom offenen Internet zu einem Grad ab, sodass ein Angreifer nur noch optische Möglichkeiten hat um sensitive Daten herauszuschleusen. Wir zeigen, dass trotz der technischen Einschränkungen ein solcher optischer Kanal eine Bandbreite von $3,729.53$ bit/s bei 0% Fehlerrate erreicht.

Schlussfolgerungen Wir präsentieren einen robusten, optischen, versteckten Kanal der eine große Menge von vermeintlich unauffälligen Bildern, die sensitive Information beinhalten, aus der Dateninfrastruktur schleusen kann, welche mit herkömmlichen Analysetools nicht offensichtlich erkannt werden.

Abstract

Background Protection and ongoing confinement of sensitive data while also allowing third parties to visit the data is a conflict and constitutes a significant challenge. A secure data infrastructure that enables visiting the data in a restricted and monitored environment which provides high guarantees to keep the sensitive data confidential, if properly set-up and operated and through the combination of technical, procedural and legal mechanisms can support data owners in solving this challenge.

Methods In this thesis, we take a look at the extent to which an authenticated, analyst is capable to transfer data out of the secure data infrastructure using an optical covert channel. Using the shared resource matrix approach, we derive possible resources that can be used as optical storage covert channel which are subject for later bandwidth and error rate calculation. The channel embeds sensitive data in ordinary images that are optically not deviating from the original image (without this image manipulation) for the human eye.

Findings In the secure data infrastructure reference implementation we virtually air-gap sensitive data from the open Internet, therefore leaving an adversary only with optical capabilities to exfiltrate the sensitive data. In this thesis we found that our covert channel can achieve a bandwidth of $3,729.53$ bit/s at 0% error rate.

Conclusions We present a robust, optical, covert channel that can exfiltrate a large amount of seemingly ordinary images that contain sensitive data from the data infrastructure that are not detected by common analysis tools with high likelihood.

Contents

CHAPTER 1

Introduction

We describe the overall idea of the underlying architecture and processes in [61] and a journal article currently in review. In this chapter, we outline our motivation to write this thesis in Sec. 1.1, state the main problem and the research questions in Sec. 1.2 and give an overview of the aim of the thesis in Sec. 1.3. In Sec. 1.4, we explain our methodology.

1.1 Motivation

In many modern settings, the need for safeguarding access to highly sensitive data arises e.g. due to privacy requirements or commercial sensitivity of data while still allowing third parties to access the data for research purposes or to assist in specific analytical tasks. This especially holds for the global outbreak of the novel SARS-CoV-2 virus, where evidence-based decision making has become acute and integration of highly sensitive information (e.g. health-, social science- or telecommunication movement data) was in high demand. Even outside of this unusual situation, the ongoing collaboration of academia with industry (as well as industry- industry collaboration) is oftentimes hindered by not being able to guarantee confidential access to the highly sensitive data, creating a deadlock where no party wants to share the data anymore.

But even outside this exceptional situation, academia–industry collaborations as well as industry-to-industry co-operations frequently are hindered by the conflicting needs to keep the data that the other party should process or analyze secret.

While data sharing is being proclaimed as the future in open science, many settings do not allow for such approaches. Data visiting, on the other hand, is an approach where data stays under the control of the owner and allows the consumers (e.g. analysts or machine learning algorithms) to come to the data to work with it. Closely monitoring the processes and interaction with data during these visits allows to put a certain level

of safe-guards in place to prevent accidental data leakage or intentional data breaches. However, most research infrastructures provide data visiting support only on a very rudimentary level, sometimes even failing to prevent the (even unintended) export of parts of data via simple file transfer mechanisms when analysis results containing parts of the data are downloaded.

In order to allow institutions to set-up and deploy secure data infrastructures that can be configured to specific requirements, allowing a specific trade-off between data and process utility and the specific security requirements, we define and document a system architecture and provide a modular pre-packaged configuration of components constituting the core of such a secure data infrastructure as reference implementation. The technical components are based entirely on open source software so that the system can be flexibly set-up on a minimal configuration of hardware, with individual components obviously being easily exchangeable by according commercial solutions and dedicated hardware components. This is complemented with process guidelines covering data delivery, access requests, as well as standard role definitions. Contractual obligations that need to be defined are discussed as part of the infrastructure set-up to complement the technical measures. It is initially configured to run as a trusted third-party environment, with a subset of this configuration being suitable for deployment within a data owner's environment [62]. The reference implementation, deployment scripts and documentation are available at the project's public repository.

1.2 Problem Statement

In the `OSSDIP` reference implementation an authenticated Analyst is not able to directly exfiltrate sensitive data through network connections, using copy-paste mechanisms or by physically being present at the server and to copy the sensitive data onto a portable flash drive. Although being designed to let authenticated, experts interact with sensitive data directly, the Analyst is by system design not able to exfiltrate the data beyond the data owner's control.

However, since the Analyst always has visual access to the data through the a remote desktop, systematic screen captures can be conducted and with the use of graphical tools decoded to receive the original sensitive data. We formulate an independent main research question (RQ) of this thesis that is divided into research questions subsequently as follows: "Using the secure data infrastructure setting, to what extent is an authenticated analyst capable to transfer data out of the secure data infrastructure using an optical channel?".

(RQ1) "Which performance with respect to robustness, undetectability and capacity can a QR-code based optical covert channel achieve?"

(RQ2) "Having an optical covert channel specification as input, how difficult is it to design a backdoor that enables a successful delivery of the covert channel?"

(RQ3) "Having an optical covert channel specification as input, to what extent can a mechanism inside the secure data infrastructure detect this covert channel?"

(RQ4) "Having an optical covert channel specification as input, to what extent can the performance be determined?"

1.3 Aim of the Work

The purpose of this thesis is to design an optical covert channel that fits into the setting of our secure data infrastructure [61]. The Analyst is an authenticated user with visual access (via remote desktop) to a virtual machine that has secure shell access to a virtual machine with sensitive data. Methods to design the optical covert channel that are subject to experimental evaluation in regards of robustness, undetectability and capacity include e.g. flicker code, matrix bar code. The main finding of the thesis (especially design and evaluation of a optical covert channel) can be applied in a multitude of other software applications or -infrastructures. We only focus on one optical covert channel in the thesis as to have a clear story. All proceedings of the thesis are subject to be implemented in the secure data infrastructure we develop at the Information and Software Engineering Group at TU Wien.

1.4 Methodological Approach

In this section we present the general methodology, argue why it is necessary to focus on optical covert channels rather than traditional network-based covert channels and outline how we measure the proposed optical covert channel.

1.4.1 General Methodology

To answer RQ1, we design the optical covert channel using e.g. bar codes that are subject to evaluation. To find resources capable to establish optical covert channels, we use the methodology of shared resource matrices[28] to enumerate resources available and select the most promising. The capacity of one (single) identified optical covert channel is derived based on the Markov method [5] which is also used in Qian et al [42]. The robustness is analyzed by artificially increasing noise by discarding channel information as proposed in the same paper. The methodologies to evaluate the covert channel are solid, as we only use select approaches that are published in peer-reviewed journals or top-conferences and often cited. The experimental evaluation of the optical covert channels will reveal the most promising in terms of robustness (message survives natural or maliciousness loss and is indispensable), undetectability (plausible having legitimate cover, message should be encoded to match channel statistically, open functionality) and capacity (maximum error-free transmission rate).

We answer RQ2 by again using the approach of Qian et al [42] via using their system model in order to be able to use their equations, then design a communication protocol

for our reference implementation setting and specify how we embed information in the communication. To improve covertness, we use the approach of Hadjuk et al [22] to hide QR-Codes in legitimate images (Steganography).

In order to answer RQ3 and find a suitable candidate, we use the methodology of Johnson et al [26] to obtain the Signal-to-Noise-Ratio methodology that quantifies the efficacy of Steganography embedding. This allows us to only consider potential covert channel candidates for the experimental evaluation.

We use the evaluation methods proposed in Qian et al [42] and Wu et al [66] and measure the bandwidth (in bits per second), error rate (in percent) and apply tuning to find the optimal data frame size and error correction strength. After comparison of the different tuning settings, the best one is selected to answer RQ4.

1.4.2 Why Optical Covert Channels?

The setting for the covert channel to be established in is our secure data infrastructure reference implementation. Since it controls all network aspects using draconian measures, it basically leaves the Analyst only with visual access to the sensitive data. Also, the infrastructure is supposed to operate remotely and being accessed with a device over the open Internet, increasing the utility value. We therefore further can neglect any electromagnetic, acoustic or any other covert channels that need physical access to the server. What the Analyst is left with, is the authenticated access to the Remote Desktop-VM and Analyst-VM. In our reference implementation of the secure data infrastructure, we are interested to implement measures to detect this covert channel. The metrics discovered in this step are of relevance since they enable a transparent comparison to other covert channels and therefore give a estimate of the channels likelihood to be used in a future deployment.

Although there exist many optical covert channels, to the best of our knowledge none involves generation of bar codes within an authenticated setting. In this thesis we construct a penetration plan on how to create the QR-Code Optical Covert Channel and elevate the capabilities such that the Analyst can exfiltrate sensitive data. We note that this approach is not trivial, given the long literature list necessary for the methodology to answer the research questions. Measuring the optical covert channel is non-trivial since it cannot be estimated how the channel will perform when certain network restrictions e.g. low bandwidth clients apply.

1.4.3 How to Measure the Optical Covert Channel

In order to evaluate the proposed QR-Code Optical Covert Channel in effectiveness, we use log inspection at the Monitoring-VM and measure the successful information transmissions that are received per time unit. The tools needed for this tasks are the standard shell, *R* using the *dplyr* and *tidyverse* packages and *RStudio*. The channel efficacy can be measured in successful information transmissions. Through these measurements,

we can determine the optical accuracy of this approach in achieved bandwidth (bit/s) and error rate (%).

1.5 Published Work

During writing of this thesis, we published a series of papers and one article that touch the secure data infrastructure and the processes within. We use them as peer reviewed published research work within this thesis too.

> M. Weise and A. Rauber: "A Data-Visiting Infrastructure for Providing Access to Preserved Databases that Cannot be Shared or Made Publicly Accessible", in *Proceedings of the 17th International Conference on Digital Preservation*, (Beijing, China), iPRES, 2021. DOI: 10.17605/OSF.IO/B7NX5.

> M. Weise, C. Michlitz, M. Staudinger, A. Rauber, K. Stytsenko, E. Gergely, R. Ganguly: "FDA-DBRepo: A Data Preservation Repository Supporting FAIR Principles, Data Versioning and Reproducible Queries", in *Proceedings of the 17th International Conference on Digital Preservation*, (Beijing, China), iPRES, 2021. DOI: 10.17605/OSF.IO/VKN4R.

A journal article describing the secure data infrastructure is currently in review, we use a pre-print in this thesis.

> M. Weise and A. Rauber: "Open Source Secure Data Infrastructure and Processes for Data Visiting", Zenodo, 2021. DOI: 10.5281/zenodo.4632903.

1.6 Thesis Outline

The rest of this thesis is structured as follows: in Chapter 2 we introduce the concept of virtualization technology, secure data infrastructures and (optical) covert channels. Chapter 3 describes our Open Source Secure Data Infrastructure and Processes in great detail. Chapter 4 uses the concepts presented earlier to describe possible covert channel candidates and select the best in terms of capacity, robustness and undetectablity. Chapter 5 defines how the covert channel communicates with the Analyst and encodes/decodes information and how it is established. Chapter 6 evaluates the performance of the optical covert channel for all possible configurations and selects the best through heuristics proposed also in this section. Additionally, it gives a starting point for detecting it. Chapter 7 discusses the main findings of the thesis and what the thesis adds to the knowledge and gives potential limitations of the approach and later concludes upon the thesis and propose future work.

CHAPTER 2

State of the Art

A secure data infrastructure enables governments, academia and businesses to provide a secure visiting point for people that visit the organization's infrastructure in a way such that control over sensitive data is never lost. Our reference implementation supports data visiting to a degree, where only pixels of sensitive data are transmitted to a authenticated entity using legally binding contracts. The rest of this chapter is organized as follows: Sec. 2.1 introduces the concept of virtualization and puts it into context with newer technologies, Sec. 2.2 presents related secure data infrastructures that are similar to our reference implementation. In Sec. 2.4, we introduce the concept of the subliminal channel and how to identify them and finally, Sec. 2.5 presents existing optical covert channels to put our novel covert channel into context.

2.1 Virtualization Technology

The concept of virtualization in the context of a computer system is well-established and existed even before virtual machines. In [18], Goldberg states that it was common to simulate a special purpose machine on another general purpose machine through writing simulators that create an environment (processor, memory, input and output devices) for the special-purpose machine. He defines a virtual machine as a simulator that can execute code directly on the underlying hardware without relying on software interpretation, orchestrated by a virtual machine monitor (hypervisor) software. It was motivated by the design of third-generation architectures (e.g. on IBM System 360) and the problem of being able to only run a single, privileged software nucleus at a time. The introduction of virtual machines (VMs) solved that problem by transforming the machine interface into arbitrary many where each of these interfaces is an "efficient replica of the original computer system, complete with all of the processor instructions [...] and system resources [...]" [18]. We explain the concept of a hypervisor in Sec. 2.1.1, traditional

virtual machines in Sec. 2.1.2 and modern approaches of lightweight virtualization in Sec. 2.1.3.

2.1.1 Role of the Hypervisor

To introduce the concept of a virtual machine hypervisor, we guide ourselves with the definition of Bressoud et al who define a hypervisor as "[...] a software layer that implements virtual machines having the same instruction-set architecture as the hardware on which the hyper-visor executes" [6]. One of the first successful commercial hypervisors, the IBM CP-67 [34] was released in IBM System 360 to provide the same instruction-set architecture as the underlying hardware with the goal of achieving nearly the same speed as it would have been executed on real hardware. A well-recognized and -established survey on early virtual machine research is presented by Goldberg [18] who explains the abstract concept of using a virtual machine monitor to increase parallel execution when only one privileged instruction-set interface is available.

In our reference implementation[1] we use Kernel-based Virtual Machine (KVM) in Rocky Linux which allows bare-metal virtualization[2]. Alternatively, a hypervisor can be a software that runs on top of an operating system. The involved components to provide virtualization and community-driven distributions can be obtained in Figure 2.1. The KVM module of the Linux Kernel along with the processor-vendor hardware acceleration modules allows the Kernel to act as a hypervisor and virtually running any operating system on top of another. The hypervisor (KVM) allows the emulator QEMU to use real hardware to accelerate computation through the Linux Kernel. The virtualization library *libvirt* is used by the hypervisor, QEMU and the Storage Pool Manager (SPM) role and is used by the virtualization host to coordinate the guest agents.

2.1.2 Traditional Virtualization

In modern cloud computing infrastructures, platform providers use virtualization technology that act as hypervisor on top of their physical machines. Scheepers [49] argues that using virtualization "[...] resources can be consumed more effectively than conventional bare-metal setups, which use physical machines for isolating different parts of application infrastructure". By nature, the hypervisor (see Sec. 2.1.1) manages multiple kernels, making isolation computationally expensive since it has to manage a large overhead. The traditional virtualization approach can be seen in Figure 2.1, it displays the hypervisor (Linux Kernel) that connects hardware resources with the virtualization module of the Linux Kernel along with libraries to manage emulated guest operating systems with accelerated performance (due to the hardware resources). It has major disadvantages compared to newer solutions: once the hypervisor is compromised, an Analyst would be

[1] M. Weise. "Open Source Secure Data Infrastructure and Processes". [Online]. URL: `https://gitlab.tuwien.ac.at/martin.weise/ossdip`, accessed 2021-08-04

[2] "What is KVM?" [Online]. URL: `https://www.redhat.com/en/topics/virtualization/what-is-KVM`, accessed 2021-08-04

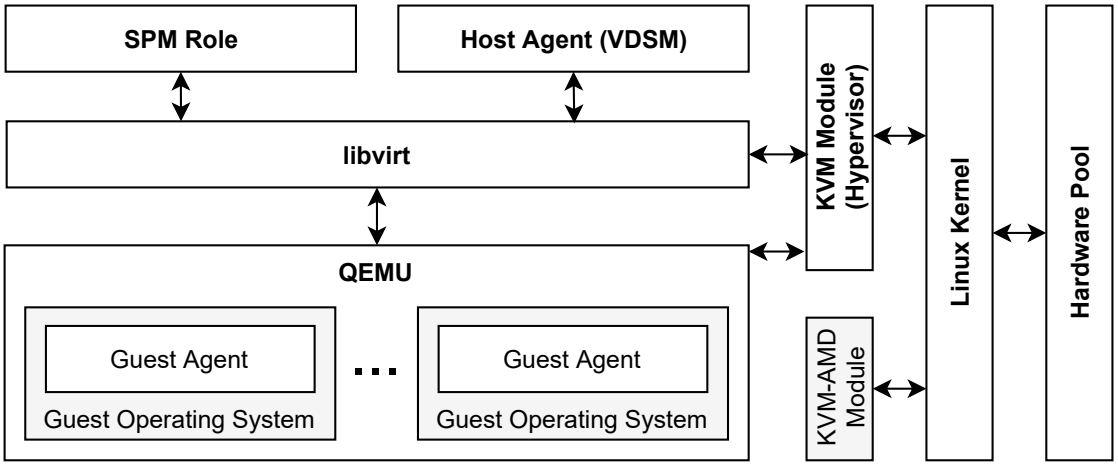

Figure 2.1: Traditional virtualization using KVM hypervisor in Red Hat Enterprise Linux (and distributions)

able to access sensitive data and escalate privileges from there on-wards. This needs to be prevented at all costs. In their work, Zhu et al [69] propose a lightweight hardware-assisted isolation approach for ARM systems that provides run-time protection even when the hypervisor is compromised.

With the growing size and features of commercial hypervisors, protecting data in cloud infrastructure settings becomes more complicated. Attacks that successfully escaped a KVM virtual machine have been conducted by Elhage [12] that identified device drivers as a weak spot. Additionally, Zhu et al state that malicious or curious cloud operators are hard to uncover since restricting the set of allowed operations must always be in balance with their maintenance tasks or even monitor them as it is hard to find suspicious activity among legitimate operations.

2.1.3 Lightweight Virtualization

According to Scheepers [49], traditional virtualization technology has a significant resource overhead for application infrastructures. It overcomes the problem of scheduling multiple kernels at once, by shifting the focus to application (or service) infrastructures. Modern container-based virtualization like Linux Containers[3] is a powerful technology since it can isolate processes and improve physical resource usage without the need of hardware emulation.

As seen in Figure 2.2, the high level principle of a container is to provide a minimal, abstracted environment for processes that can run a variety of different operating systems, but does not depend on hardware emulation or hardware requirements. These are managed by the underlying host operating system, we use a rpm-based distribution

[3] "Linux Containers". [Online]. URL: `https://www.linuxcontainers.org/`, accessed 2021-08-04

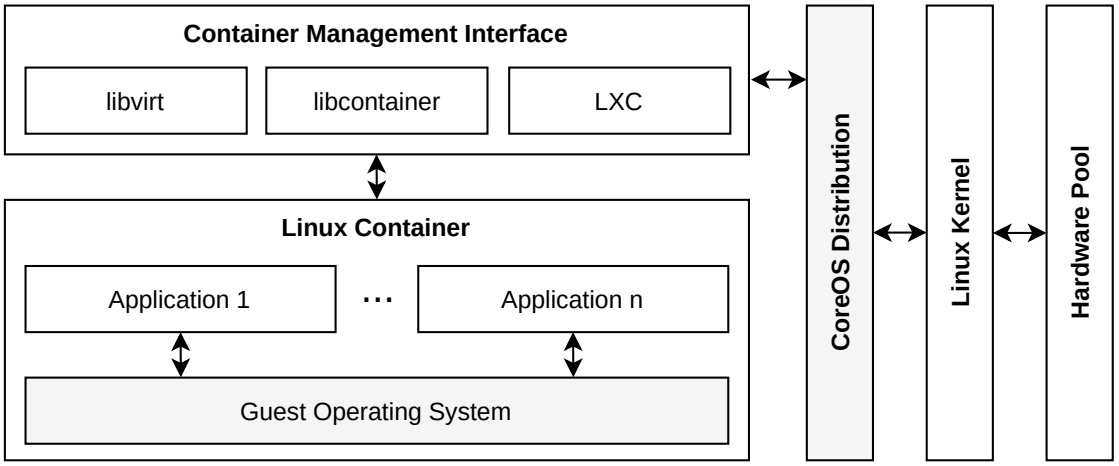

Figure 2.2: Virtualization using Linux Containers

Rocky Linux[4]. The system administrator interacts with the containers through the container management interface. Combined with Linux Kernel features, containers can access the underlying hardware directly without relying on emulators such as QEMU, only depending on required pre-packaged libraries or a runtime within the Linux Container operating system.

In this thesis we do not cover the more common Docker[5] ecosystem due to the strong dependency on encrypted storage. Since this field of engineering is rather new in the enterprise-setting, there does not exist audited best-practice guidance on how to use encrypted storage volumes with Docker to the best of our knowledge. Interested readers that want to know how Docker advances Linux Containers are referred to the excellent technical article[6].

2.2 Secure Data Infrastructures

Desai et al [10] propose the "five safes" dimensions to protect personal rights of people whose data is available to some organization. Its intent is to cover all aspects of handling sensitive data to protect the overall confidentiality of the data while at the same time give adopters enough flexibility to develop more mature dimensions in favor of others. We highlight on the dimensions and further describe them in Sec. 2.2.1: (i) *safe projects* address management decisions regarding appropriateness of the usage of the data through auditability and review processes, (ii) *safe people* identifies individuals that access the

[4]Rocky Enterprise Software Foundation. "Rocky Linux". [Online]. URL: `https://rockylinux.org/`, accessed 2021-08-09

[5]Docker Inc. "Empowering App Development for Developers". [Online]. URL: `https://www.docker.com/`, accessed 2021-08-04

[6]J. Fernandez. "Containers are Linux". [Online]. URL: `https://www.redhat.com/en/blog/containers-are-linux`, accessed 2021-08-04

sensitive data and require them to sign legally binding terms of use, (iii) *safe data* ensures appropriate data de-identification and access capabilities with respect to the research questions formulated, (iv) *safe settings* addresses the necessity of security and transparency to achieve trust with the public and data owners, (v) *safe outputs* ensures only approved, aggregated research results can be exported out of the system.. For the remainder of this Section, we color components that come in touch with sensitive data pink, firewall barriers purple and database-like components yellow.

2.2.1 Trusted Research Environments

The United Kingdom Health Data Research Alliance (UKHDRA) is an confederation of leading organizations in the healthcare field along with research institutions to establish best-practice and increase the ethical use of health data for research and innovation. Extending the "five safes" approach of Desai et al [10], the UKHDRA proposes in a recent green paper [24] to use the framework for the development of a system that enables access to sensitive data while keeping the risk of unauthorized re-identification of de-identified data low. It allows identified researchers to access and work with national health data (e.g. stroke-, prescription- or COVID-19 vaccine data). Their approach of a TRE+ environment consists of the "five safes" dimensions in Sec. 2.2 and (vi) *safe return* to allow de-identified research results to be re-identified and securely mapped back to the original data set. In the following we will explain the concept of the six safes that is implemented in the concept of TREs and further, indirectly, in our secure data infrastructure discussed in Chapter 3.

We want to discuss the motivation of separating the concerns of developing a secure data infrastructure into five controls. Hubbard et al [24] identified an additional control that needs to be addressed by a TRE to operate a secure environment that meets the necessary safety standards for healthcare data to protect the privacy of an individual. To ensure public trust, a organization should implement their secure data infrastructure based on these recommendations of them and have their solution independently accredited and audited. Depending on the context of the organization, the *safe setting* (e.g. healthcare) or *safe data* (e.g. research cohorts) control may be increased in favor of other controls. For some organizations that require high-performance computing (HPC), this approach is more complete than others as it explicitly deals with the dependencies of data ingress and -egress to these computing systems. Additionally, it also considers public clouds into the architecture model. Next, the five safes of Desai et al [10] are presented and discussed in respect to our approach [61].

(i) *safe people* is a control to ensure the interests of the individuals whose data is collected and used in the analysis. As they point out, the analysts that want to work with the data can have non-standard backgrounds (e.g. startups, practitioners) and need to sign a mandatory and legally-binding document to protect the data individual's privacy. Before actually accessing the data, an analyst is subject to mandatory information governance training. As this methodology may tremendously

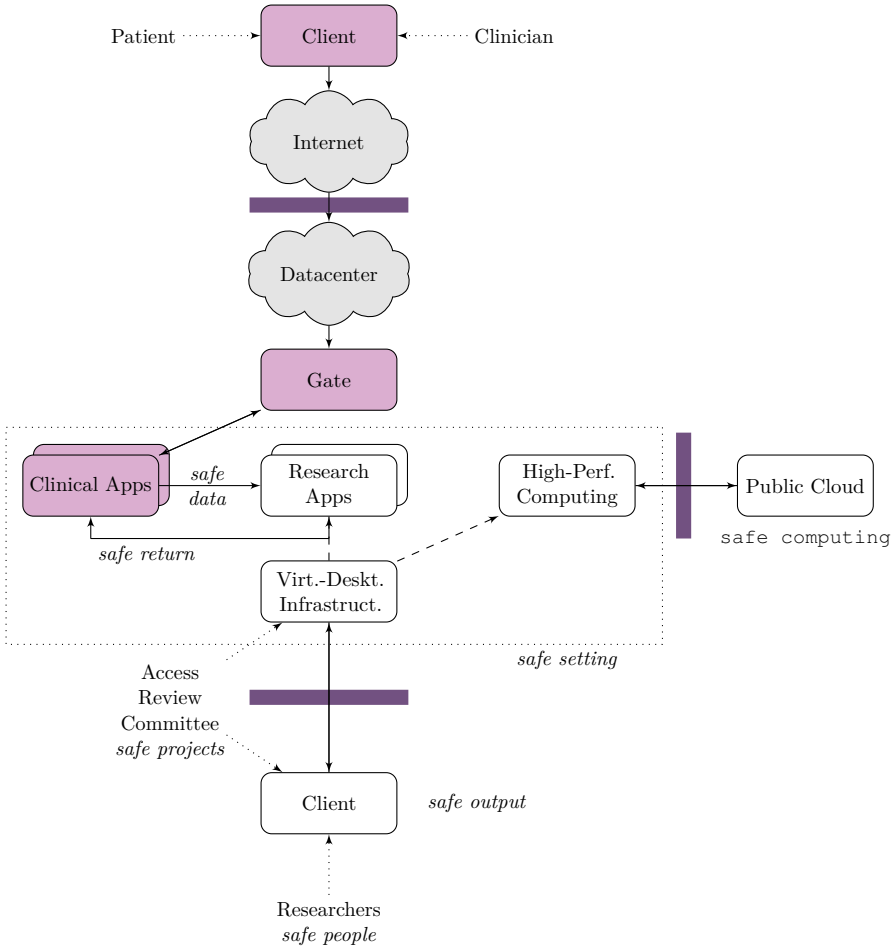

Figure 2.3: UKHDRA's approach of a TRE+ environment on the example of Genomics England. Connections that are not mentioned in the paper are marked with dashed lines.

increases the organizational overhead for a analyst that wants to use multiple secure data infrastructures, their approach considers a once-approved-access-all scheme.

(ii) *safe projects* deals with the appropriate use of sensitive data and mandatory possibility to externally audit this handling to ensure compliance. It requires from the organization to install a ethics board that reviews the project proposal and gives a clearance. Since this control deals with the transparency and involving of protected individuals into the processes, we argue that it targets the improvement of the maturity of the project management processes within an organization. Unfortunately, the scope of this control is not precise as of the time of writing, also no public response from members is available in the publication.

(iii) *safe setting* is the control to ensure a straightforward to use, secure environment for the sensitive data to reside in. Their work gives a minimum requirement catalog

that is compiled to the bare necessities for TREs, such as protected individual data cannot be exported and the actions of authorized analysts are monitored. To import individual data into and, after explicit grant, export data out of the TRE, an analyst must follow a defined and transparent process that involves trusted administrators that have permission to exchange data at the artificial air-gap boundaries.

(iv) Additionally, their work extends the control by *safe computing* which covers the issue of outsourcing infrastructure (e.g. content delivery networks) since this approach has become more common in some business models. To overcome the risk of exposing sensitive data to public cloud providers, additional safeguards that disallow any outsourced hardware or software to access the sensitive data at any time must be implemented. This approach is well-studied and supported from major public cloud providers.

(v) *safe data* is a control that describes the measurements necessary to minimize risk of (accidental) re-identification of protected individuals when importing data into the TRE. For this task usually de-identification software tools and encrypted (virtual) disks are used in an enterprise setting. Their description is not specific about the general minimal requirements, but it can be assumed that it is in good interest of the protected individual to provide as much data as necessary but not disclose any sensitive data that allows identification to infrastructure carriers. This may conflict with the analyst's interest that wants to have as many data to analyze available as possible. The TRE prioritizes the interests of the protected individual over the analyst through technical measures (e.g. data-shielding, homomorphic encryption, multi-party computation) as well as organizational measures (e.g. anonymization, pseudonymization).

(vi) *safe outputs* deals with the barrier ("air-gap") of the *safe setting* and the open internet. This control is not to be confused with the *safe setting*: while the *safe setting* deals with system-internal barriers while the safe outputs manages the communication to the outside world. This applies especially for the data ingress and data egress process that are monitored by the *safe projects* control. Since this often times will require manual intervention they recommend to at least partially automatize these processes to a degree where risk of disclosure can be controlled. Although not explicitly mentioned in the specification, we argue that a TRE has to allow (critical) security updates from the operating system (OS) manufacturer to not compromise the overall system's security. Similarly, the analyst must be provided with a standard set of tools appropriate to the set of tasks that he or she wants to accomplish, but not (trivially) risk the protected individual's privacy.

(vii) *safe return* is a control that enables de-identified research output to be returned into the TRE where the sensitive data comes from and the identity of the protected individual is known. This allows the TRE to gain information about the protected individual itself, therefore to profit from allowing analysts to work with the data. They argue the importance of this approach with the example case of Genomics

England[7] whose architecture is presented in Figure 2.3. It ensures that researchers can only access de-identified sensitive data while all their actions are overseen by a committee. Their research on genomes requires high performance compute which they obtain through contracting a public cloud provider that meets the *safe computing* requirements. Upon approval of both the patient and the ethics board, the research output can be re-identified and mapped back to the original data set to enrich value to it.

For analyst requirements and accreditation, interested readers are encouraged to read the respective sections in [24] to have a starting point for understanding the whole concept. For this thesis, we omit these sections as they do not add relevance to the topic of optical covert channels.

2.2.2 RemoteNEPS

In their work [54], Skopek et al propose a system that allows remote data access through web technology in a secure environment. The National Education Panel Study (NEPS) hosts a full-fledged secure data infrastructure capable of handling 50 users simultaneously at a machine cluster consisting of 72 physical cores (running at 2.0 GHz) and 1,344.0 TB random access memory. Due to data disclosure concerns, research data centers may enable researchers local data access only. In a connected, global community of researchers and practitioners this in almost all cases renders in unacceptable data visiting conditions for sensitive data. They argue, that this "[leads] to an under-utilization of research data, which was expensive to gather". Their technical approach is to use a web browser as client software to connect with the secure data infrastructure through a remote desktop server. This has a major advantage over conventional approaches like remote execution or job-submission systems that have input- and possibly output queues, as the output is immediately present on the screen of the user. During publication, their system was in production for four years and served more than 200 users. To prevent credential sharing, the system implements a bio-metric authentication through keystroke pattern recognition software KeyTrac[8] that is trained before providing the user credentials to the user. This training is executed in a supervised manner. Unfortunately, the paper does not reveal the nature of the supervised training, but out of context it can be assumed that the training is executed remotely.

Another way to access data in `RemoteNEPS` is through a download link, this requires the analyst to sign a *data use agreement* to retrieve strongly anonymized data sets. When accessing the data through the remote desktop, the data is moderately anonymized and the analyst has to sign an additional *supplementary agreement*. The least anonymization on the data is performed, when the analyst visits the data security rooms at Leibniz

[7] Genomics England Inc. "Genomics England". [Online]. URL: `https://www.genomicsengland.co.uk/`, accessed 2021-08-09

[8] TM3 Software GmbH. "Keyboard biometrics". [Online]. URL: `https://www.keytrac.net/en/`, accessed 2021-08-09

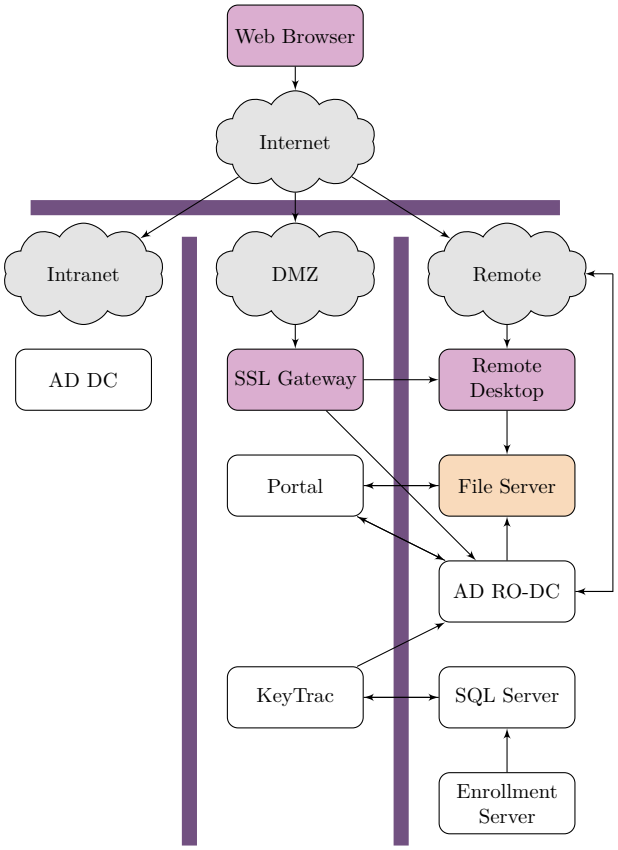

Figure 2.4: General architecture of *RemoteNEPS*

Institute for Educational Trajectories (Germany). Then, the analyst needs to sign the *on-site supplmentary agreement* together with the aforementioned agreements.

The architecture of `RemoteNEPS` [54] (in the remainder of this thesis we write secure infrastructure names in typewriter) is visualized in Figure 2.4 and consists of an Enrollment Server that stores the credentials and bio-metric information in a Active Directory Domain Controller (AD DC) and SQL Server respectively during the training session. Their architecture also uses a read-only mirror (AD RO-DC) for the publicly accessible portal and the File Server that holds the sensitive data to protect the authentication against unintended writes. To use the system, researchers and practitioners need to connect to the gateway that is placed behind a firewall through the web browser and the Java plugin. The system prompts for the credentials and bio-metric authentication and looks them up in the respective databases. It then enables a secure connection to the Remote Desktop Server which provides the workstation environment for the researcher or practitioner. From it, the researcher can access the imported and granted files from the Portal that were uploaded beforehand in a dedicated exchange folder. The Remote Desktop Server provides an isolated instance with the tools necessary to process the data, alternatively

the system administrator can install custom software in a manual process. Finally, when the researcher or practitioner is done with processing the sensitive data, there is a possibility to export artifacts (e.g. report) to the portal where it can be extracted out of the system after output permission is granted.

`RemoteNEPS` uses username, password and keystroke behavior for authentication to prevent credential sharing. Similarly our system enforces the usage of two-factor authentication which addresses the same issue. Their technical approach is to use a web browser with a Java plugin as client software to connect with the secure data infrastructure through a remote desktop server and allows no Internet access within the desktop session [1]. It extends TREs by providing metadata publicly in a web portal that can be accessed after creation of a user account, making data more findable. Also, their approach does not intend to be hosted on a public cloud, which simplifies agreements and processes of handling the sensitive data, but also reduces the value of their contribution for institutions that may want to adopt their approach. In contrast to TREs, their work [1], [54] describes the environment where the researcher works more detailed, where TREs only refer to a virtual desktop environment. Their infrastructure uses Active Directory Services[9] and biometric key-stroke authentication for each new login attempt.

Another downside already outlined by Skopek et al [54] is the tremendous cost of operating such a system since their environment consists of almost only commercial software (*SPSS*, *R*, *Microsoft Office*, *Windows 7*, etc.). Often times these are not purchased once, but running licenses or restrictions (e.g. only allowing a certain amount of VMs per on-premise hardware).

2.2.3 SAIL Gateway

Jones et al [27] describe a secure remote access system that allows sensitive data to be accessed with privacy preserving controls through *Secure Anonymized Information Leakage* (SAIL) Gateway. It allows two or more separated data sets to be linked together (e.g. by anonymous linkage field) to gain more information about a individual and making it as accessible as possible, without compromising the privacy assertions. Their central architecture is well-motivated and covers both technical and procedural controls that improved the situation for a data linkage for various projects [7], [8], [33] that would not be practical without a data linkage infrastructure. The `SAIL Gateway` runs in a high performance computing infrastructure[10], has a mean computing time of 30 seconds and is used by more than 100 approved projects. They also report a total of 75 remote desktops with possible support to 300.

The design of the architecture of the `SAIL Gateway` considers multiple risks for unauthorized data loss or -leakage, data integrity mechanisms and scenarios where the Analyst

[9] Microsoft. "Active Directory Services". [Online]. URL: `https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/dd578336(v=ws.10)`, accessed 2021-08-02

[10] Swansea University. "Secure eResearch Platform". [Online]. URL: `https://serp.ac.uk/`, accessed 2021-12-06

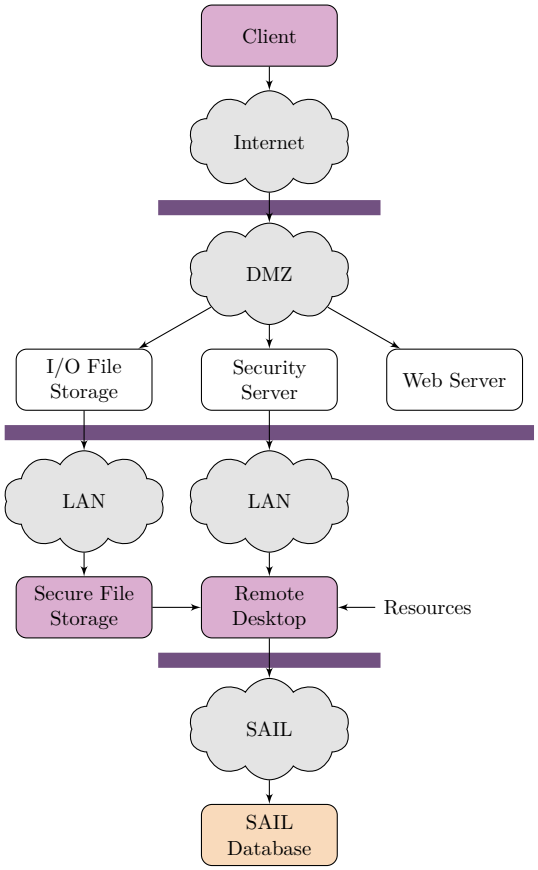

Figure 2.5: General architecture of *SAIL Gateway*

is a trusted insider to the organization that operates the Gateway (see Figure 2.5). Their approach is very straightforward and reduces the full data set to the needs of the analyst's proposal and an informal requirement to not trying to re-identify the protected individual. The Data-VM allows an analyst to access the data, but not modify it as only read access is permitted. Before the `SAIL Gateway` was put into operation, the standard procedure was to only grant approved researchers read access to the data through on-premise terminals.

Jones et al [27] further elaborate on their split-file approach which is a main contribution to the topic of secure data infrastructures. Their process is described on an abstract specification level. First, the original data set is split into two files: (1) identifier file (name, address, postal code, gender, date of birth), and (2) clinical data by the Data Provider. For not compromising the protected individual's privacy and still allow the researcher to use some individual context, file 1 is sent to an information service that replaces the identifiers with more general data (e.g. week of birth, gender code, lower super output area) to create file 3. This is sent together with file 2 to the SAIL Database to bring individual context to the clinical data. The trusted third party virtually has

only two responsibilities: (a) pseudonymization of the data using the split-file approach, and (b) matching of the pseudonymous linkage field (unique number generated using symmetrical encryption of the national heath service number). They argue that this anonymization is especially non-trivial for addresses that are in rural areas, where the lower super output area contains not enough inhabitants for sufficient anonymity. One robust method to ensure data privacy is to use residential anonymous linking fields [46] that is considerably useful to determine individuals who live in the same household if they pass certain heuristics without identification information.

Before a data user can access the anonymized data set at the `SAIL Gateway`, he or she has to submit a project request which can be reviewed by the data providers before granting access. This ensures that the `SAIL Gateway` is only used for approved researchers for disclosed purposes. Since nobody has access to identifiable data (recall the file-split), even the trained and trusted system administrators that have access to the raw data have no capabilities to see any identifiable data. Naïvely, one can now assume that the isolated analysis environment is not necessary, since no identification can happen. This is only partly true as Jones et al argue, the risk of potential disclosure of multi-variable data is still considered too high. Their system has a multitude of safeguards: (a) legal and technical mechanisms, (b) monitoring and logging of interactions, especially SQL commands issued, (c) provisioned data subsets and views, (d) secured hardware infrastructure. Their environment allows a Data User to use a Windows virtual workstation environment with scientific tools like *Microsoft Office*, *SPSS*, *R* and *STATA* available by default. The system allows to develop SQL scripts in an *IBM InfoSphere* environment and to request additional, possibly commercial, software.

2.2.4 DEXHELPP

Popper et al propose in [40], [68] the `DEXHELPP` infrastructure, that facilitates research in Austria for almost 10 years. The goal is to provide analysts with a secure and controlled environment without the need to exfiltrate data out of the system. Data owners on the other side deposit their data from heterogeneous sources (e.g. GAP-DRG 2006/2007[11]) in a encrypted vault (c.f. Data Endpoint in Figure 2.6) and specify fine-grained access rights to individuals or groups of analysts to entire data assets or just specific subsets through e.g. limiting the number of records or columns displayed (via the LDAP Node). The Monitoring Node continuously monitors the access to the Data Endpoint which allows for auditing and inspection of the usage of the data at any time. The Services Node provides a Docker environment where containers are provided (e.g. PostgreSQL, RStudio, Web Applications) to the Analysts working on the Remote Desktop Node. The Analyst connects through the VPN Client (*Cisco Anyconnect*) to the VPN Endpoint using two-factors (password, time-based one time password), the authentication uses the PAM Node that keeps track of the two-factor authentication for the VPN connection.

[11] Hauptverband der österreichischen Sozialversicherungsträger. "Grundlagenforschung für Ambulante, Personenbezogene Diagnoses Related Groups"

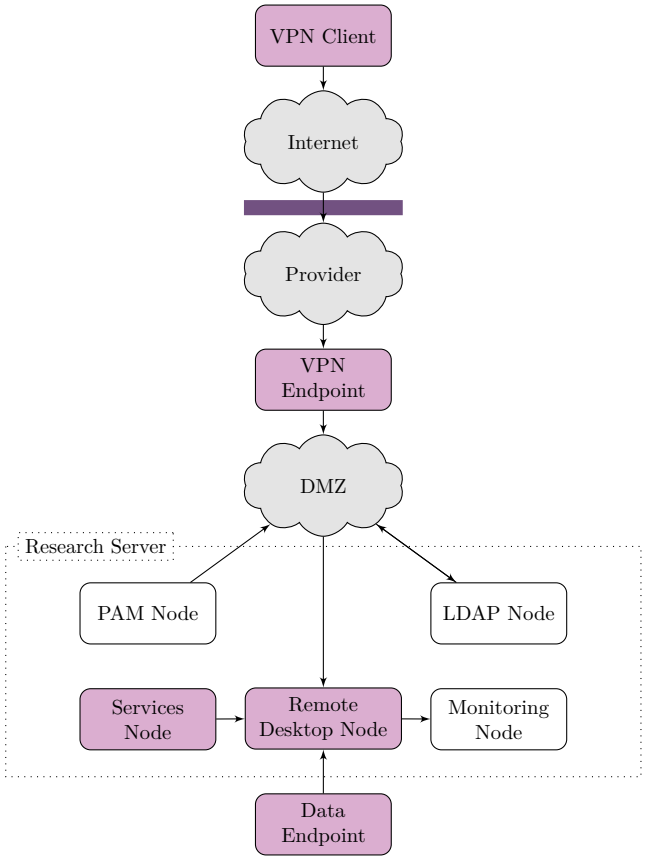

Figure 2.6: General Architecture of DEXHELPP [38]

The design of the system follows a trade-off between a restrictive environment to not expose the data to software vulnerabilities and a rich tool set to work with the data. Upon clearance of the system administrator, the system also offers a fast computing environment (through using *Xen*[12] instead of libvirt for GPU execution) with special hardware available on demand. In a recent study [3] Bicher et al evaluate the effectiveness of contact-tracing policies against the spread of SARS-CoV-2 [48] in Austria using the DEXHELPP infrastructure and an established population sampling routine GEPOC-ABM [4] that became a "powerful and versatile simulation tool for any kind of population-based research problem in Austria".

[12]"Xen Server". URL: https://xenserver.org/, accessed 2021-08-19. Note that this product was renamed to "Citrix Hypervisor" from 2019 onwards and acquired by Citrix Systems, Inc.

2.2.5 Comparison

The described infrastructures provide standard data science software installations running on proprietary Windows operating system[13]. Only the SAIL Gateway provides instant messenger software[14] and proprietary data mining software[15] by default. To provide a non-proprietary operating system in our reference implementation of OSSDIP, we use the enterprise operating system *Rocky Linux* that is available free of charge.

In regards to the remote desktop implementation, the presented systems are similar: DEXHELPP offers VNC and RDP (two common protocols). The SAIL Gateway does not disclose the protocol but uses a proprietary remote client software[16]. Unfortunately, the description of TREs does not specify how the remote desktop should be implemented. This influences the decision on to which protocol to use in the reference implementation, where we provide a remote desktop via VNC protocol similarly to DEXHELPP that is supported by common client software applications.

Other than the RemoteNEPS who uses a proprietary biometric mechanism, both DEXHELPP and the SAIL Gateway use second-factor authentication in the form of time-based one-time passwords or physical digital keys. In our reference implementation we incorporate the time-based one-time password second-factor authentication of DEXHELPP.

2.3 QR-Codes

Initially created for inventory tracking, QR-Codes ("Quick Response-Codes") are omnipresent in modern life. The recent pandemic of SARS-CoV-2 [48] especially reinforced the need for reliable information transfer while keeping a safe distance to an individual. Perez-Alba et al [37] for example use QR-Codes that patients (without an appointment) need to scan during waiting when arriving at a hospital in Mexico. In a recent article, Wilf-Miron et al [63] propose a "green pass" that is state-issued and uses QR-Code technology on paper to allow verification (pass forgery is a criminal act and can even lead to incarceration). This pass is checked when e.g. tickets for events are purchased. This system has also been implemented in the European Union[17] and the European Economic Area. In the rest of this thesis we only consider QR-Codes model 2 described in detail by Tiwari [58]. Other types are Micro QR-Codes (unidirectional, less space), iQR-Codes (product name, can use rectangular modules), SQRC-Codes (Secret-function-equipped QR-Codes, contain open public content and encrypted private content)

[13] "Windows-OS". [Online]. URL: https://www.microsoft.com/de-at/windows/, accessed 2021-12-06

[14] "Pidgin". [Online]. URL: https://www.pidgin.im/, accessed 2021-12-06

[15] "IBM InfoSphere Information Server". [Online]. URL: https://www.ibm.com/analytics/information-server, accessed 2021-12-06

[16] "VMware Horizon Agent". [Online]. URL: https://www.vmware.com/products/horizon.html, accessed 2021-12-06

[17] European Commission. "EU Digital COVID Certificate: EU Gateway goes live with seven countries one month ahead of deadline". June 1st 2021, Brussels

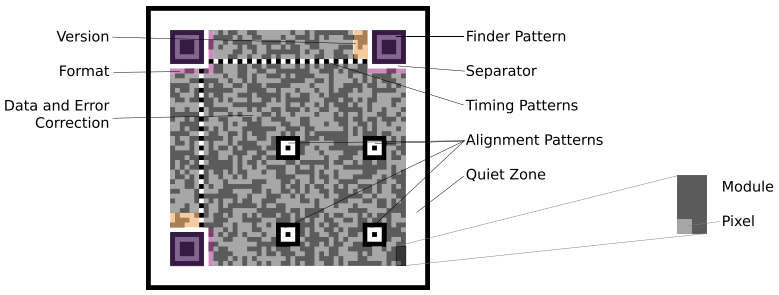

Figure 2.7: Anatomy of a QR-Code (adapted from [58])

A QR-Code is constructed also with metadata information (e.g. version, data format) in Figure 2.7. The finder patterns along with the alignment patterns allow the scanner to quickly find the data and error correction area, even when the QR-Code area is not flat. At the end of a successful decoding is the original information as encoded into the QR-Code (see Figure 2.8). We evaluate this process later in Section 4.1. A QR-Code has three free variables that need to be specified when generating a new one:

(1) *version* from 1 (smallest) to version 40 (highest), depending on the library used[18]. It determines the amount of modules which follows equation $m = 17 + 4 \cdot v$ which gradually increases the available information capacity.

(2) *error correction code* (ECC) level, one of L, M, Q, H determines the redundancy embedded in the information. The minimum is $L = 7\%$, $M = 15\%$, $Q = 25\%$ and the highest is $H = 30\%$. It is claimed by Tiwari [58] that ECC level M is used most frequently. Depending on the version and ECC level, a QR-Code can store up to $7,089$ digits and $4,296$ characters. The maximum capacity differs since the defined encoding allows a grouping of three digits per pixel while only allowing two characters per pixe[19].

(3) *size* determining the resulting QR-Code size. By default QR-Codes are created to be as small as possible while maintaining ability to scan them from the distance. The size of a QR-Code does not influence the information capacity.

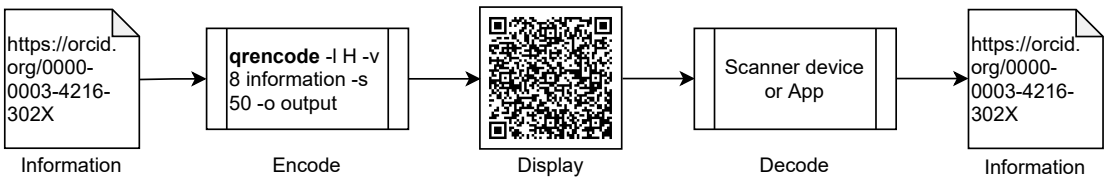

Figure 2.8: QR-Code encoding-decoding process

[18]K. Fukuchi. "libqrencode – A Fast and Compact QR Code Encoding Library". October 9th, 2021
[19]M. Nooner. "pyqrcode – Python 3 module to generate QR Code". December 5th, 2021

Their design when transmitting information through optical channels makes them a promising candidate for illegitimate data transfers out of a system, which is our main motivation to use them in the optical covert channel described in Chapter 5. For example, take a company-managed computer that disallows any data export through technical policies that prevent file uploads, printing, taking screenshots and send them via mail, etc. When network- and physical interfaces are not available anymore, an attacker still has access to the computer screen and can take pictures with another camera or a scanner. Using QR-Code technology [58], structured exfiltration through scanning a QR-Code presented on the screen is possible and fast.

2.4 Covert Channels

In order to comprehend the nature of covert channels, we first take a look at the historic concept of the subliminal channel and the prisoner's problem. It provides a general overview on plausible deniability and the need for covert communication in some situations. With this concept introduced, we explain a method to detect covert channels that still has impact and is widely accepted. Later, we give a general and more modern approach how to detect covert channels and provide a detection mechanism that is related to our secure data infrastructure.

2.4.1 Prisoner's Problem

Simmons [52] defines the prisoner's problem as "two accomplices in a crime have been arrested and are about to be locked in widely separated cells. Their only means of communication after they are locked up will be by way of messages conveyed for them by trustees – who are known to be agents of the warden". Since the prisoners have strong interest in keeping the communication flowing, they are willing to accept multiple ways of deception e.g. receiving forged or changed messages by the warden. Additionally they are not allowed to exchange hidden information, it allows the warden to possibly detect escape plans. Of course the prisoners will try to deceive the warden into believing that they only exchange innocuous information while in secret, they establish a subliminal channel in "full view of the warden, even though the messages themselves contain no secret [...] information". For a better understanding on how a communication model for this problem can look like, we visualized it in Figure 2.9.

Although this is part of the authentication without secrecy problem [53], the prisoner's problem adds encryption to secure the authentication against "stripping off" and appending it to a forged message. The warden is able to decrypt the cipher using a key provided to him and examine the contents (such nothing has been concealed from him or her) before relaying the cipher to the receiving prisoner. Upon receiving, the prisoner can authenticate the message through presence of the authentication information (Simmons uses a number of bits of redundant information to the content information) that is expected by the receiver to believe in receiving a genuine message.

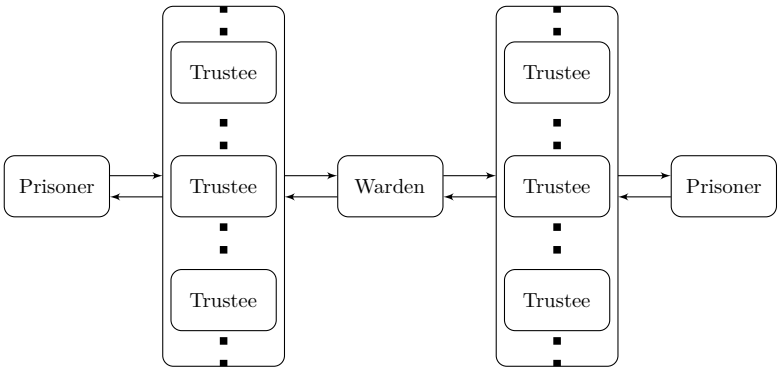

Figure 2.9: Communication model of the prisoner's problem

Using the concept of authentication without secrecy, Simmons describes the concept of subliminal channels in great detail. He explains that digital signature algorithms (e.g. found in the RSA-cryptosystem) who depend on "good" random large prime numbers can be used to establish a subliminal channel and use these parameters to send a message subliminally, but leaving an intact digital signature for the warden to verify. However, the warden can deceive each of the prisoners with almost certain probability into accepting a forged message as genuine. He argues that this risk can be reduced significantly through restricting the set of acceptable messages by e.g. using primes with same bit length.

2.4.2 Identification

In [28], Kemmerer parts covert channels in storage covert channels and timing covert channels. The first is described between a sender that alters a data item and a receiver that detects and interprets the changed value of the data item covertly. A timing covert channel uses time modulation of a value on a data item to covertly transfer information. He further argues, that discovering storage and timing channels can be exhaustive with the need of analyzing all possibilities. To this end, he proposes a shared resource matrix methodology to increase assurance that all channels have been found. The concept of shared resources is defined by Kemmerer as "[...] any object or collection of objects that may be referenced or modified by more than one process". This methodology has rather loose assumptions: (a) the actors are processes, and (b) the processes share are common processor, and (c) the processes are either local or distributed in a way that only one process is active at any one time.

The concept of overt channels is straightforward since they only consist of a system's data object that is used by one entity to write information into (sender) and by another entity to read information from (receiver). As the name suggests, a covert channel differs from an overt channel which resembles an innocuous, rightful way of (inter-process) communication. Kemmerer defines a covert channel as "[they], in contrast, use entities not normally viewed as data objects to transfer information from one subject to another.

These non-data objects, such as file locks, device busy flags, and the passing of time, are needed to register the state of the system" [28].

The methodology utilizes a two-dimensional matrix with operational primitives (e.g. write file, unlock file) as the column headings and resource attributes (e.g. process id, file value) as the y-axis headings for all enumerable resources. In the next step, every field is evaluated in the feasibility for covert communication. This methodology is a combination of system description analysis, formal operations and constraint satisfaction for the discovery of covert channels. It considers legitimate channels, usefulness and identity of the receiving process to draw conclusion on potential storage channels.

2.5 Existing Optical Covert Channels

In this section, we want to give an overview of the existing optical covert channels that use similar approaches. Section 2.5.2 describes a optical covert channel using embedded matrix bar codes. In Sec. 2.5.3, we describe an optical covert channel using indicator lights of data communication equipment.

2.5.1 Steganography

Image steganography is a technique to embed secret information into legitimate images such that they are still perceived as non-altered and legitimate with the goal of covertly transmit information. Traditionally, least significant bit [25] (LSB) modifications are used to embed text or images within images. The LSB is the last bit of a information byte, e.g. the ASCII letter "T" is represented as `01010100` where `0` is the LSB. Encoding the information `1` into "T" however would result in the representation `01010101` which is decoded into "U". Changing letters in a text this way might raise awareness of a steganography attempt very quickly. This is why we use image steganography, where information is altered similarly, but without raising visual attention (see Figure 2.10). We encode a URL handle into a very common image used in optical publications (instead of the `.tiff` original Lena image, we use a `.png` representation of the same dimension to increase interoperability).

The efficacy can be measured through the Peak Signal-to-Noise Ratio (PSNR) and compared to other steganography methods. It is determined by the fraction of the power of the input signal (desired) and the power of the background noise (unwanted) and typically measured in Decibel. A value higher than 0dB indicates that more signal power than noise power is present in the resulting image. When embedding QR-Codes, the version influences the PSNR. Higher QR-Code versions have a higher PSNR since the space covered by the QR-Code reduces the noise available to hide the signal.

2.5.2 Invisible QR-Codes

In a paper, Guri [20] demonstrates the successful ex-filtration of sensitive information from air-gapped computers through their computer screen. The optical covert channel

Figure 2.10: Original image (left), encoded image carrying the information "https://orcid.org/0000-0003-4216-302X" using steganography (right)

uses fast blinking, encoded visual objects (e.g. QR-Codes) that are presented in a way such it is hard to see them with bare eyes but not when capturing them using a camera. He elaborates on the applicability of displaying the QR-Code covertly in nearly all situations where the display can be observed by a camera within line of sight.

This methodology is also applied to embedding images and text instead of encoded data. To maximize the reconstruction rate, Guri used a professional grade Digital Single-Lens Reflex (DSLR) camera Canon 5D Mark III. However, the reconstruction rates for QR-Codes (version 3, medium error correction level) are better in almost all cases (except for a distance of 50 centimeter). Unfortunately this circumstance is not explained in the paper. Additionally, the methodology described considered six different cameras that are diverse in price and also used photo- and video recording as capturing method. For white on white blinking QR-Codes, the professional DSLR (35 millimeter) performed the best for all distances, the webcam (Microsoft LifeCam Studio) performed similar, while all other devices (Samsung Galaxy S5, Google Glass Explorer Edition, GoPro Hero 4) did not have good reconstruction rates. Guri concludes on possible countermeasures that "[the] practical implementation seems nontrivial, unless one knows what to look for; in this case, anomaly detection techniques may be helpful". However, he states that polarization filters can be used to minimize the line of sight for an attacker with a camera.

2.5.3 Computer Status LED

In their practical research paper, Loughry et al [32] use status indicators on data communication equipment such that the modulated optical signal of light emitting diodes correlates with a sublime message. Their work is relevant in a way, that the

transmission rate is outstanding (compared to the reported threshold of 100 bit/s by Levenshtein [59]) and the realistic experimental evaluation. Additionally, their approach categorizes equipment into three categories according to the amount of information potentially carried to an adversary.

In their categorization into class I (e.g. unmodulated indicators such as a power-on indicators), class II (e.g. activity lights and front panel lights) and class III (e.g. modem "transmitted data" or "received data"). Their experimental evaluation consists of 39 devices with heterogeneous usage (e.g. local-area network, wide-area network, mainframe computers) within a networked environment where only sufficiently good transmission capabilities to restore the original signal are considered. The indicator signals were collected using a photo-detector within a measured distance from 5 to 38 meters (with steps of 5 meters) and test data transmissions of 300 to $19,200.0$ bit/s (with doubling the data transmission rates). Within this transmission rate, their approach was error-free.

They experienced significant optical noise from artificial sources such as lamps or industrial lighting (interestingly it is claimed that sun light is easy to filter out). One of their main finding is, that the most frequent noise is artificial light operating at 120 Hz and multiples of this due to the standard 60 Hz alternating current power supply in their area. Countermeasures to this attack mainly consists of slowing down-, course grain or disable the indicator light emitting diodes.

2.5.4 Hard Disk Drive LED

Guri et al [21] modulate the hard disk drive (HDD) activity LED to massively ex-filtrate data from environments that are isolated using an air-gap. Similar to other methods (see Sec. 2.5.3 for example), the proposed covert channel uses a sequence of legitimate operations to encode the covert timing channel and ultimately ex-filtrate sensitive data. It is claimed, that the HDD activity LED cannot be manipulated directly through software. However, through playing a sequence of read and write operations on the HDD with respect to its response time it can be modulated sufficiently. Since common computers, laptops and servers have a HDD activity LED which continuously flickers, a modulation of this information can be masked and does not draw special attention to a potential victim.

Their methodology consists of four phases: (a) infection using e.g. social engineering or supply chain attacks (b) transmission of the binary data through the blinking HDD encoding scheme (c) reception which requires a line of sight to the HDD activity LED as part of a "malicious insider" or as remote camera scenario through e.g. local hidden camera, drone outside of the premise, compromised security camera (d) decode to demodulate and decoded back into the original binary form. The experimental evaluation consists of three off-the-shelf standard desktop PCs with distinct HDD activity LEDs in the computer case and seven different receivers to record the optical transmission.

Their experimental evaluation showed that a heterogeneous band of cameras achieves a maximum bandwidth of $15 - 120$ bit/s while a commercial photo diode sensor achieves

nearly 4,000 bit/s. Additionally, the HDD activity LED can be detected from more than 30 meter away. Countermeasures proposed by Guri et al include placing hardware carrying sensitive information into secure rooms where only authorized personnel has access and banning all types of cameras from these rooms as well as e.g. detecting algorithms that trigger frequent changes of the HDD activity LED. A very concrete and easy to implement measure is proposed: adding a low pass filter consisting of a resistor and capacitor between motherboard and HDD LED that constraints the blinking frequency.

2.6 Summary

In this Chapter, we introduced the concept of virtualization technology and a virtual machine (VM) controlled by a hypervisor that provides an interface for interacting with the operating system running in the VM. We explain the internal components within the KVM hypervisor and introduce also lightweight virtualization and argue why it is not sufficient for our reference implementation. Further, we introduce other secure data infrastructures deployed by institutes and compare the setting as well as use-cases. We introduce the concept of covert channels and explain a methodology of how to identify them along with exemplary optical covert channels.

CHAPTER 3

Open Source Secure Data Infrastructure and Processes

This chapter describes our reference implementation of the open source secure data infrastructure and processes, extensively described in [62]. We present the underlying security concepts implemented and the reasons behind them, as well as a general understanding of the processes within the infrastructure. The rest of this chapter is structured as follows: Sec. 3.1 gives an overview of our secure data infrastructure reference implementation, Sec. 3.2 describes the components of the architecture in detail supporting the five safes and the controls in Sec. 3.3.

3.1 Overview

A secure data infrastructure provides powerful security guarantees through a combination of technical, legal and procedural mechanisms for individuals that want to share sensitive data to experts. These individuals are referred to as data owners in the rest of this thesis, since they oversee who interacts with their data and allow/dismiss access to it. We describe a secure data infrastructure [61] that implements security controls that target the most relevant technical and organizational aspects specific to the setting of data visiting; that is to invite experts to visit the infrastructure and process sensitive data in a secure, isolated environment specific to their task.

In the context of the recent pandemic of the novel virus SARS-CoV-2 [48] this especially holds. A naïve approach to solve this problem may evolve around de-identified, possibly aggregated data sets that are released to the public on a published period. We argue that this has several disadvantages:

(a) The arguably necessary de-identification to protect the rights of individuals strips away information that may increase the value of the research like age, sex or country

of residence. With no possibility of the individual to agree to the disclosure of the personal information or re-identification, the impact of scientific work may decrease although the problem is not the disclosure, but the risk of sensitive information leakage;

(b) Access to data that is available only aggregated may not meet the scientific standards necessary to further process them, even when the researcher is not even interested in the identification of an individual;

The key concept is, that sensitive data is stored in a Data-VM which is surrounded by a strict firewall barrier that only allows process-approved connections to selected virtual machines (VMs) to it. Additionally, the experts never receive access to the Data-VM, but to an excerpt that is provided on a dedicated compute virtual machine (Analyst-VM) together with the tools required to perform the analysis. Access to this Analyst-VM is never provided directly, but through a dedicated Remote Desktop-VM to introduce a media break and avoid any data flowing off via e.g. copy and paste.

Since our method only provides visual access (pixel-by-pixel) to the data and the VNC suite does not offer native audio support (and no ssh tunnel is available), the only remaining way to extract data in a illegitimate way is through a visual covert channel. When the analyst (c.f. Section 3.3.1) is finished with analyzing the provided data set, the resulting artifacts e.g. trained models can be exported through dedicated Data Owner-VMs that require initial clearance by the Data Owner who inspects the contents before approving their export.

3.2 System Architecture

Our method provides a secure environment by providing confidentiality through implementing technical and organizational measures, integrity by five controls that are explained in the following subsections and availability by deploying the system on specialized server hardware. For our reference implementation, we only consider best-practice open source software to further strengthen resilience of our system.

The core VMs in the secure data infrastructure are orchestrated by a hypervisor (or "virtual machine monitor") that also isolates VMs from each other at all times [35]. We extend this isolation to a degree where only the responsible role and the secure data infrastructure itself are able to operate the VM in a limited boundary, further restricting the recommended system hardening on Red Hat Enterprise Linux[1] like distributions like Rocky Linux.

[1]Red Hat Inc. "Red Hat Enterprise Linux". [Online]. URL: `https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux`, accessed 2021-09-07

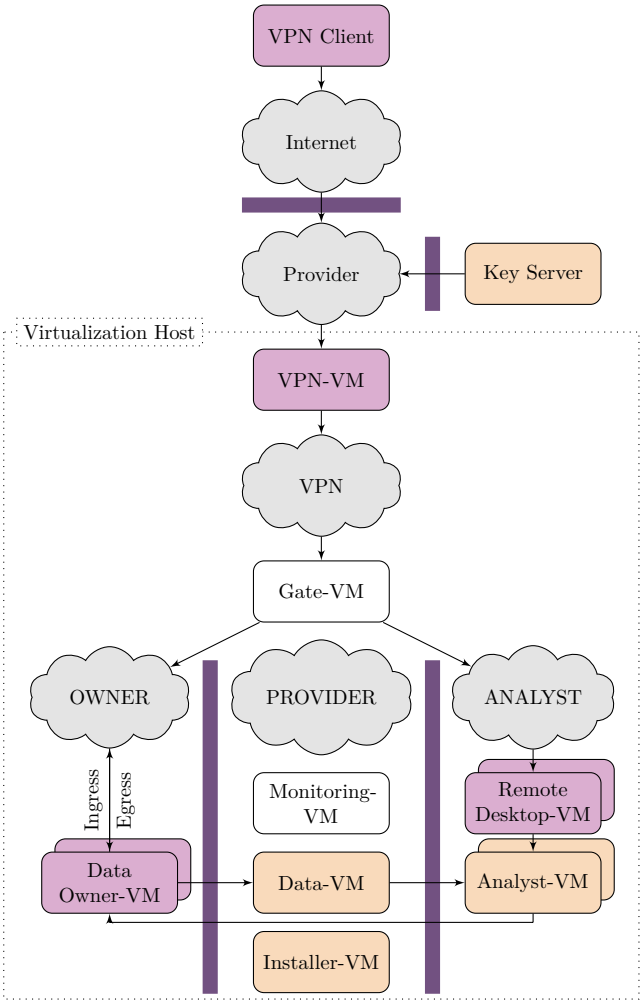

Figure 3.1: Multiple security layers in our architecture. Everything below the Provider network is virtualized on our dedicated virtualization host, hence the capitalized network names and VM suffixes.

3.2.1 VPN-VM

We run a virtual private network (VPN) VM on the virtualization host that serves as endpoint for the Data Owner and Analyst. Initially it was a community maintained set-up script of *libreswan*[2] for the IPSec server and *xl2tpd* as L2TP provider. This implementation was hard to configure in a different cluster and firewall, so we switched to a community maintained OpenVPN Access Server[3] set-up script using a community-maintained certificate authority toolkit.

[2]"Libreswan VPN Library". [Online]. URL: `https://libreswan.org/`, accessed 2021-09-07

[3]OpenVPN. "Access Server Self-Hosted VPN". [Online]. URL: `https://openvpn.net/access-server/`, accessed 2021-09-07

3.2.2 Gate-VM

There are two firewall configurations in the secure data infrastructure. Additionally to the virtualization host, we configure the default firewall service on the Gate-VM to only allow any core infrastructure component ("core VMs") to access the installer component as well as to specific Data Owner-VMs and Remote Desktop-VMs. Since the Analyst accesses the Analyst-VM through a dedicated Remote Desktop-VM and the Data Owner through a dedicated Data Owner-VM we restrict all traffic but to these specific VMs for a given user. We do this by configuring the standard firewall of the *netfilter* module of the Linux kernel. The Gate-VM connects the VPN network (open to the Internet) with the restricted virtual machines and can be seen as the gate of both the VPN network and the Owner- and Analyst networks in terms of routing.

3.2.3 Data-VM

The central Data-VM (virtualized on the virtualization host in the reference implementation, usually a dedicated air-gapped machine) runs a *PostgreSQL* server and follows RDA recommendations for dynamic data citation [44] through time-versioned tables. The Data-VM is also used while copying the data to or from the Data Owner-VM.

3.2.4 Installer-VM

The first component that is deployed during set-up is the Installer-VM. It is put in place to standardize all VM set-ups of the secure data infrastructure and to provide a cached Rocky Linux image along with core repository packages. Additionally it distributes the trusted keys for SSH connections for e.g. backup access to the VMs and manages user details for the notification tasks. The Installer-VM also maintains a custom software repository to provide the Analyst with required software on the Analyst-VM.

3.2.5 Monitoring-VM

The Monitoring-VM is a virtualized component on the virtualization host and runs a *rsyslog*[4] endpoint that receives events happening in the secure data infrastructure through datagrams. We save the activity in the form of audit trails that are read-only to the Provider role and anybody else. Only the secure data infrastructure can write to the log files. Improvements may be introduced using e.g. a controlled write-once storage medium for the log files or a hardware-controlled write-once storage medium for the log files. The Monitoring-VM also stores the VNC stream from the Remote Desktop-VMs through starting a proxy[5] that stores the whole stream that can be replayed like a video on the TigerVNC[6] client.

[4] "The Rocket-fast Syslog Server". [Online]. URL: `https://www.rsyslog.com/`, accessed 2021-09-07

[5] "VncProxy". [Online]. URL: `https://github.com/amitbet/vncproxy`, accessed 2021-08-29

[6] "TigerVNC". [Online]. URL: `https://tigervnc.org/`, accessed 2021-07-27

3.2.6 Key Node

For encryption of the VM storage, a pseudo-random encryption key is generated that is further encrypted with a password. We store this password on a separate physical machine that is outside of the server room to minimize effects of stealing the storage of the Virtualization Host.

3.2.7 Data Owner-VM

For each case where a Data Owner wants to submit data (ingress) and extract data (egress), our system deploys a fresh VM with sufficient encrypted memory available to achieve this. The Data Owner role receives by default a 20 gigabyte volume for this task that is mounted in the root directory of the Data Owner-VM. In order to establish a connection to the Data Owner-VM, the respective role has to enter a two-factor authentication code before connecting to the VPN-VM.

3.2.8 Analyst-VM

The Analyst is provided with a temporary VM, created individually for each analysis and data processing task. Upon creation with a 0ed storage region (in the reference implementation we format the encrypted virtual partitions with 0s as first step in the set-up when creating the file system) these are equipped with a copy of the data subset and the tools required by the Analyst. The Analyst-VM operates within an isolated network and can only be accessed through the corresponding Remote Desktop-VM. The system administrator can allow connections to verified license servers when required by analysis tools and when transferring data to- or from a Data Owner-VM as part of the *safe return* or data ingress process.

3.2.9 Remote Desktop-VM

In order to access the Analyst-VM, the Analyst needs to first connect to a temporary Remote Desktop-VM, created for a selected Analyst-VM – these only occur in pairs. We use *TigerVNC* as software implementation that runs inside the Remote Desktop-VM as a process, providing windowing system capabilities for using graphical tools at the Analyst-VM, configured to provide only video connectivity and without any cut-and-paste capability offered by *Xvnc*. The Remote Desktop-VM offers 1080p display resolution (1920x1080 pixels) by default in the reference implementation. The Remote Desktop-VMs sends all interactions and the video stream to the Monitoring Node. To save space, we use the RFB protocol that only stores changes in the video capture instead of frames. Even though the VNC protocol suite is not able to transfer audio, we disable virtual sound capabilities (through detaching the virtual sound card) on all VMs to reduce the likelihood of establishing an acoustic covert channel. An exemplary Remote Desktop-VM deployment using *RStudio*[7] is shown in Figure 3.2, depicting how the Analyst has direct

[7] "RStudio". [Online]. URL: `https://www.rstudio.com/`, accessed 2021-09-07

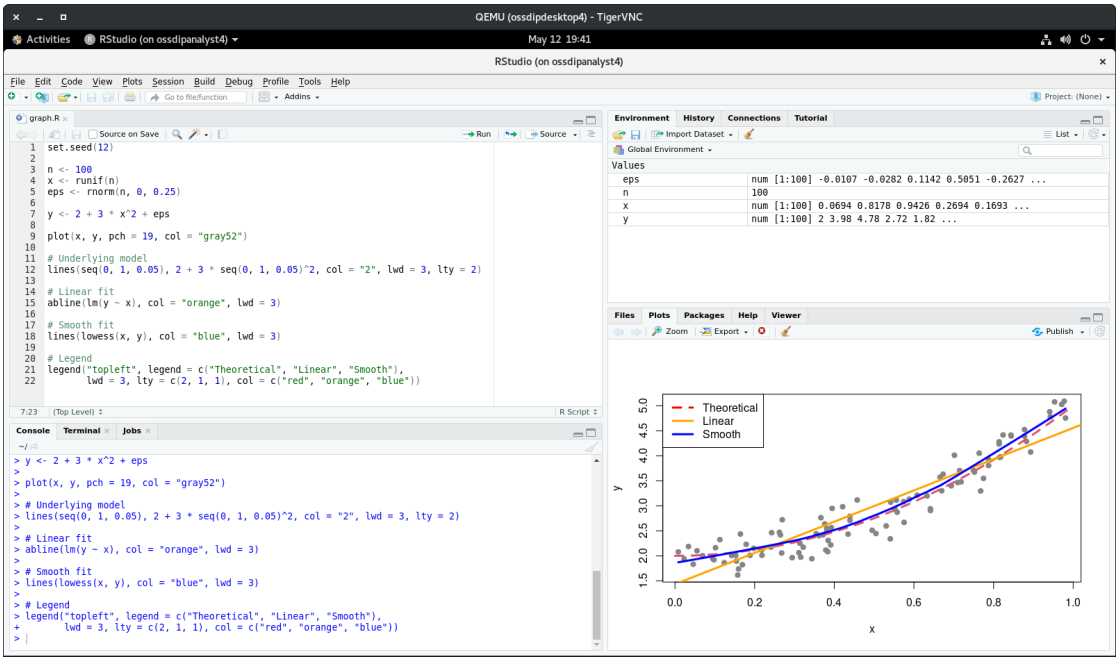

Figure 3.2: The Analyst can work directly with the sensitive data at the Analyst-VM through the x-window-system through the Remote Desktop-VM (exemplary data).

graphical access to the sensitive data.

3.3 Secure Data Infrastructure Controls

We use the six saves framework [24] on our reference implementation in a similar as intended by the authors. We split them into infrastructure- and process controls to explain the principles on a technical and organizational level. The infrastructure controls of our reference implementation secure the control of sensitive data from the physical storage location to the pixel displayed on the Analyst notebook's screen. In the rest of this section we explain the technical controls of our approach that protect the data. The organizational processes can be found in our reference implementation.

3.3.1 Roles and Controlled Access

To protect the sensitive data in the infrastructure, access to the server hardware must follow a strict concept. We recommend to host the secure data infrastructure on-premise and in a dedicated locked server rack where only the designated and certified operator can open the lock. According to the four eye principle, the server room should only be accessible with a key card that is held solely by another operator.

In our reference implementation, we use the role-based access control [14] concept that assigns an individual a set of roles that have associated privileges. These privileges serve

as rules which allow them to access defined components of the secure data infrastructure. Previous research has shown that this concept is vulnerable to privilege escalation [41] attacks. We therefore equip each role only with a bare minimum of privileges on a need-to-know principle for data access. In total, we defined six roles that participate in the secure data infrastructure and are not exclusive (a single entity can occupy more than one role, e.g. Data Owner and System Administrator). For better reading, we Capitalize the role names in the following and give an overview.

The *Data Owner* wants to maintain strict control of sensitive data while also allowing an identified expert to access the data and work with it. Our secure data infrastructure provides the expert with temporary and isolated Data Owner-VMs that allow ingress of data into a secure component that is destroyed after the import has finished. It provides the Data Owner with full control over the data and who it is provided to by allowing access to (excerpts) of the audit trails that monitor all events concerning the data. Data Owner-, Analyst- and Remote Desktop-VMs require a two-factor authentication from the role that is using them. Our implementation requires the setup of a second factor application during the first successful login, currently only Google Authenticator is supported.

The *Analyst* must have a clear understanding on what research questions need to be addressed and what data from the Data-VM is required to answer them in order to interact with the secure data infrastructure. The Analyst accesses the infrastructure through commodity hardware, e.g. notebook ("Analyst notebook"). It can be assigned to experts that need to analyze or process the data, but where sharing the data is not feasible. With the permission of the Data Owner, the Analyst is able to use a subset of the data in isolated VMs that have compute-optimized hardware configurations. The access to these temporary VMs is only granted for a limited time period, network connections to other VMs are prevented by the firewall running in the Gate-VM, Analysts can only access their own VMs. Following an approval process involving the Data Owner, access to the temporary VMs can be extended. The Analyst can have access to more than one VM at a time for different analysis tasks.

The *Data Provider* processes the data on behalf of the Data Owner. It manages the services and takes care of all operations of creating the respective isolated VMs for Data Owner- and Analyst roles, monitors user interactions, and handles the legal contracts required. It may outsource the actual hardware provisioning to a sub-contracted Carrier role, although frequently these two roles will be combined to reduce complexity and risks. From our experience, the Data Owner may itself decide to provide the services, leading to a simpler set-up and accordingly simpler processes for data ingress by combining the Data Owner- and Data Provider roles.

The *Carrier* is the organization that operates the secure data infrastructure, typically this role is integrated in the Data Provider role. This has the benefit of simplifying some processes and increasing the level of (immediate physical) control over the data when an on-premise infrastructure is used. Our system design however allows for outsourcing this role to a third party private cloud provider acting as Carrier. This may has the benefit of

predictable infrastructure costs, automated backups and economical scalability without the need of maintaining the infrastructure from a hardware side.

The *System Administrator* is responsible for maintaining the secure data infrastructure including the hardware, except for the Data-VM. For our reference implementation this would be the OpenStack[8] cloud computing infrastructure software running on a general purpose cluster at TU Wien. This role may be further sub-divided to assign, e.g. two different individuals, the respective role for the Data-VM and the Monitoring-VM [62].

The *Database Administrator* is responsible for maintaining the Data-VM. It is nominated by the provider organization and is mutually exclusive with the System Administrator. This exclusivity is required to not enable a change in the central Data-VM and subsequently manipulate the logs in the Monitoring-VM. This role also has to maintain the Key Node that stores the passphrase for decrypting the encryption key for the node disks for similar reasons.

3.3.2 Data Segmentation

Another control that improves the system's security is the data segmentation that separates the streams for different roles. We describe the data segmentation that also implements the RDA WGDC recommendations on dynamic data citation [44], [45] by placing a Data-VM into the management network of the infrastructure provider. The system never provides direct access to this Data-VM for any role. Moreover a process of versioning when appending data, storing queries executed when extracting a subset and optionally put related data back is implemented.

The recommendation of *safe data* is not met directly by our reference implementation since we do not explicitly differentiate between de-identified or open data, but instead use the general term of sensitive data. However, the ways both technical and organizational nature of our secure the data infrastructure by never providing direct access to the Data-VM to the Analyst and requiring the Carrier to monitor all actions that allow the Data Owner to comprehend the tasks that the Analyst executes, the Data Owner always has power to simply shut off any operations involving owned data. Requiring the Data Owner and Analyst to provide a second factor to authenticate against the respective VM ensures that the account is not used by e.g. a System Administrator. Since we encrypt the storage using a passphrase that is not accessible to the System Administrator (since it is managed by the Database Administrator), the System Administrator cannot access the storage of the VMs either.

Additionally, we employ the concept of *safe return*; however again, not in a direct way. The original recommendation defines a re-identification of the research artifacts and mapping it to the respective individual sensitive record. In our implementation, we only consider the use of sensitive data. Since every Data Owner use-case has a dedicated

[8] Open Source Cloud Computing Infrastructure. [Online]. URL: `https://www.openstack.org/`, accessed 2021-07-27

database in the Data-VM, the Database Administrator can copy artifacts on the Analyst-VM after clearance of an approval board and the Data Owner back to the Data-VM. Since the Data-VM is air-gapped, the Database Administrator has to open a connection through e.g. connecting the Data-VM to the isolated `OWNER` or `ANALYST` network for the duration of the transfer and then remove the connection. In the reference implementation this is done through the SPICE Server[9]

3.3.3 Network Segmentation

In our reference implementation, we split the network into six virtual networks without overlapping addresses following current best-practices. Using the standard *netfilter* module of the Linux kernel, additional restrictions are put in place where bridges are allowed. The Gate-VM is the single point where information can cross more than one network boundary.

The networks `OWNER` and `ANALYST` are completely isolated from the rest of the secure data infrastructure with the only meeting point of the Gate-VM. To still enable the Data Owner- and Analyst-VMs to install updates, they are allowed incoming connections to update repositories of the open Internet and selected license servers as well as potentially local mirrors of supported software. To the respective roles, they are only accessible through successful authentication against the VPN-VM. For Analyst-VMs, the Analyst cannot access the VM through SSH directly, but first needs to connect to a Remote Desktop-VM which then enables a connection via SSH to the Analyst-VM.

3.3.4 Automation

Our reference implementation supports multiple levels of security, starting with the secure generation of isolated volumes for the virtual machines. During creation of any VM, the required storage space is marked as exclusive, encrypted using the Linux Unified Key Setup (LUKS) and an encryption key of 4096 bits length. The key is further encrypted using a passphrase that resides on a dedicated external server (Key Node). It can be retrieved through SSH authentication, the passphrase is simply returned to the caller and only resides within the random access memory (RAM) of the Virtualization Host. It is queried again in case the server needs to restart or shuts down unexpectedly after system boot to prevent boot loader attacks [64]. All this happens automatically through Ansible playbooks.

Ansible[10] is a automation engine that allows the System Administrator to execute a set of tasks that occur often inside of a system. For our reference implementation, we use Ansible playbooks to set-up the core VMs, start them, create Data Owner- and Analyst-VMs, import and export data to the Data-VM and tear down the core VMs if necessary. The LUKS encryption key during the whole Virtualization Host lifetime resides

[9] "SPICE Project". [Online]. URL: `https://spice-space.org/`, accessed 2021-08-17

[10] Red Hat Inc. "Ansible is Simple IT Automation". [Online]. URL: `https://www.ansible.com/`, accessed 2021-09-07

only inside the RAM. This ensures the physical safety of the Virtualization Host disks. Although the reference implementation is designed to destroy the LUKS-encrypted disks and Data Owner- and Analyst-VMs handling is also automated with Ansible playbooks. We stress that in production deployment every VM should have a dedicated node, the temporary-VMs can share a physical node. The Data-VM should additionally have a dedicated Data Monitoring node and should not be connected to any other node than the Data Monitoring node (which itself is only connect to the Data node).

3.3.5 Monitoring

For the Data Owner it is important to comprehend the analyzing tasks performed by the Analyst on the Analyst-VM. To enable a crosscheck between the submitted proposal to analyze the data and the actual task, the Analyst must sign a full agreement of monitoring. Currently, the reference implementation is configured to append everything of importance to the Monitoring-VM. Since we capture human interaction with the secure data infrastructure, we give opportunity for automatic discovery of malicious pattern recognition, although our reference implementation does not automatically analyze these patterns. However, any System Administrator can connect the organization's commercial endpoint protection software to the Monitoring-VM in order to do so. In the context of this thesis, we connect to the Monitoring-VM and develop heuristics to discover our optical covert channel presented in Chapter 5.

3.4 Secure Infrastructure Processes

In this section we present the controlled interaction with the secure data infrastructure involving the role definitions from Sec. 3.3.1. We focus on the interactions involving the handling of sensitive data spanning from the ingress of data over the data provisioning to the data egress, covering the most important data lifecycle phases. These processes may require some training as part of a workshop, although processes such as log supervision, maintenance, account management, etc. are part of standard infrastructure operation and thus require no explicit training for the operating organization.

3.4.1 Data Ingress

This process is started, when the Data Owner wants to import data into the secure data infrastructure with the goal to make it available to the Analyst in the future (through the Data Provider). When Data Owner and Data Provider are the same entity, the simplified process can be used. We provide the full process chart in Figure 3.3 mark it with an Asterisk* and coloring the optional steps for the simplified operation in gray. For the non-combined process, the Data Owner needs to accept the Data Provider's data use agreement to provide a legal basis for the processing on the Data Provider's hardware. The Data Providers provides the Data Owner with training material in the form of documentation as well as workshops and accomplishes the training.

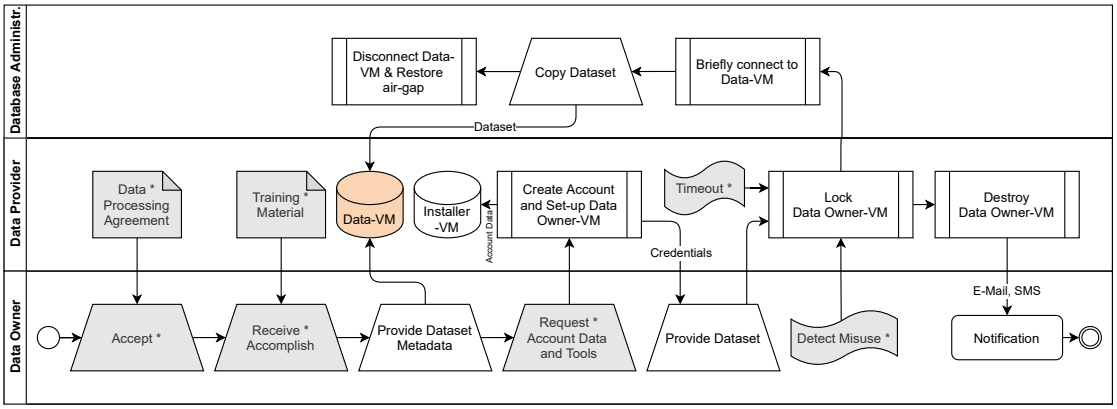

Figure 3.3: Providing the data to the Data Owner-VM.

Afterwards, the Data Owner provides the meta data to make it findable in a public data catalog. For the non-combined process only, the Data Owner needs to provide identification possibilities and ways to later receive notifications to the Data Owner-VM within the secure data infrastructure and the required tools while the Data Provider creates the Data Owner-VM and stores the credentials in the credential store (database inside the Installer-VM). The Data Owner also needs to be identified by the Data Provider through using e.g. a state-issued identity or video conferencing tool. The Data Owner is also provided with a password and a two-factor time-based one-time password for authentication. In our reference implementation, the Database Administrator copies the data set from the Data Owner-VM to the Data-VM and afterwards notifies the Data Provider to delete the Data Owner-VM.

3.4.2 Data Access

This process is started when an Analyst wants to visit and work with sensitive data of the Data Owner through the computational resources and training of the Data Provider (see Figure 3.4). First, the Data Owner needs to verify the identity of the Analyst. Then, the Analyst requests access to a (sub-)set of the sensitive data specifying the intended tasks and research questions that should be answered or processed to be performed for e.g. data enhancements to be imported back to the Data-VM. After (manual) check of the identity, the Data Owner and additional committees such as boards, review the research questions and grant or reject permission to use the data. When granted, the *data access agreement* between Data Owner and Analyst must be accepted including the conditions of use: (i) prohibition of data download (ii) prohibition of de-anonymization (iii) non-disclosure agreement and (iv) agreement to extensive monitoring. Upon acceptance, a request for an account is issued providing the required information like e-mail address and mobile phone number to send messages.

Later, the Data Provider extracts the data set on behalf of the Data Owner and provisions a new Remote Desktop- and Analyst-VM. The Analyst-VM is provisioned with the data

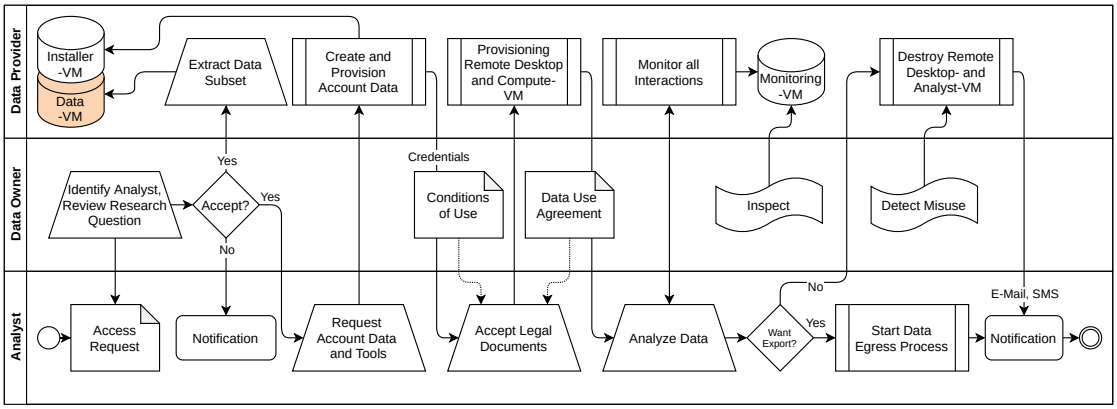

Figure 3.4: Analyst accessing sensitive data in our reference implementation.

excerpt from the Data-VM by briefly deactivating the air-gap of the Data-VM. The Analyst needs to accept the data processing agreement to further request credentials to visit the data and perform data processing tasks. Since these tasks are very heterogeneous, the Analyst can select standard tools to work with and optional custom installations of e.g. commercial software. The Data Provider then creates and provisions account data (the same way as in creating Data Owner credentials in Sec. 3.4.1) and saves them in the credential store of the Installer-VM as well as providing it to the Analyst. After accepting the legal documents from the Data Owner.

Then, the Analyst can work with the data while the Data Owner can inspect all interaction with the data at any time through requesting excerpts from the Monitoring-VM and viewing the VNC stream. In case the Analyst misuses the provided subset and the Data Owner can detect it, the Analyst-VM is locked immediately through locking the user account, thus preventing it from authentication and optionally scheduling Remote Desktop-VM and Analyst-VM for deletion. As long as the Analyst does not misuse the data, the Analyst can work with the data until the pre-defined timeout e.g. 100 days from creation, the Data Owner can permit extensions. After that, the Analyst is notified of access revocation through two different channels.

3.4.3 Data Egress

At the end of the data access process, the Analyst may wish to export artifacts such as graphs, reports, aggregated anonymous data, etc. To accomplish this, the Analyst places the artifacts in the export folder on the Analyst-VM and issues an export request containing the list of files to export and reason for doing so to the Data Owner who inspects it and gives clearance. Upon rejection, the Data Owner sends a notification to the Analyst and the process ends. When the Data Owner approves the export, the Data Provider extracts the artifacts from the Analyst-VM and then provisions a Data Owner-VM with copies of the artifacts for the Data Owner to inspect. After inspecting the artifacts and giving a second clearance, the Data Owner approves a re-import of the

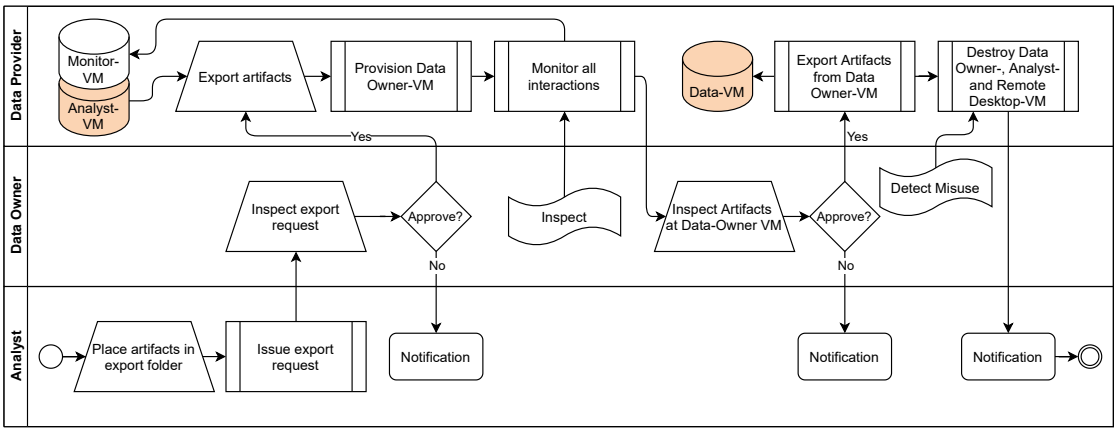

Figure 3.5: Safely exporting data out of the secure data infrastructure.

artifacts into the Data-VM. Note that the artifacts never leave the infrastructure, the Analyst needs to write results down from the screen. The Data Owner can also lock access to the Data Provider at any time and subsequently issue a destruction of the Data Owner-VM. In every case the Analyst is notified about the changes in the accessibility of the Data Owner-VM.

3.5 Summary

In this chapter we described our reference implementation demonstrating a sample deployment of the more difficult, enterprise production system described in [62]. We gave a motivation to implement such a system and described its architecture along with the components (VMs) involved and how they interact with each other. Also, we describe the roles that are interacting with the reference implementation and describe the controlled access that form the multiple layers of technical measures that protect the sensitive data from being leaked. We also give organizational processes for the most important actions (data ingress, data access and data egress) and the roles involved.

CHAPTER 4

System Modeling

In this chapter, we only consider optical covert channels since we strip the VMs from non-process network communication possibilities (c.f. Section 3.1). Even if not supported by the remote desktop protocol VNC, we configure all VMs to not use a sound card, eliminating acoustic covert channels [9] as explained in Section 3.2.9. Since access to the server room is only possible via the four eye principle for the system administrator (c.f. Section 3.3.1), electromagnetic or acoustic covert channels that use hardware are not likely to occur. Since the roles of the system administrator and database administrator are not able to access the sensitive data without leaving audit trails, the only remaining undetectable way is through the monitor of a Remote Desktop-VM and connected Analyst-VM. We use the methodology of Qian et al [17] to formulate a system model and introduce the optical covert channel. We enumerate all shared resources in a resource matrix as proposed by Kemmerer [28] and compare them by their suitability of the optical covert channel.

4.1 Communication Model

The communication model of Lampson consists of a sender ("customer") and a receiver ("service"), hence the communication model for a covert channel exists between a customer that interacts with a service. We use the notion of Qian et al [42] to design a covert channel communication model that is used by a malicious analyst in Figure 4.1. It shows the transmitter component $\mathcal{A}$ that is placed inside the secure data infrastructure with access to sensitive data within the control of the analyst. The receiver component $\mathcal{B}$ (Analyst notebook) is selected among the existing components of the secure data infrastructure, in order to deceive the detection mechanisms that are in place to detect data loss.

Our communication model for the optical covert channel makes use of the transmitter- and receiver component to prevent detection from the detector $\mathcal{X}$. Typically, the detector

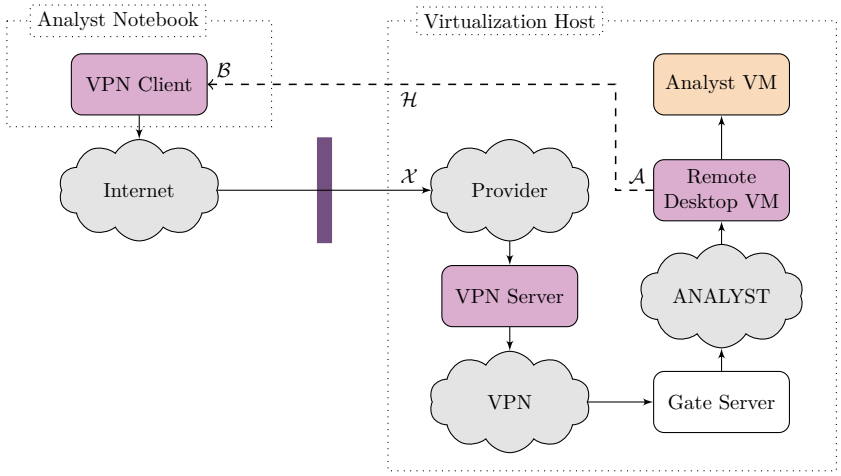

Figure 4.1: Covert Channel Communication Model ($\mathcal{A}$ is the transmitter, $\mathcal{B}$ the receiver, $\mathcal{X}$ the detector and $\mathcal{H}$ the optical covert channel).

is not a single component but is composed of multiple components that work together to detect malicious behavior. In the case of our secure data infrastructure we use a central logger (Monitoring-VM) that collects all interaction events. An optical covert channel $\mathcal{H}$ in our case is a one-way oriented data flow from the transmitter component $\mathcal{A}$ to the receiver component $\mathcal{B}$. It evades the security policies implemented in the Remote Desktop-VM, Gate-VM and the VPN-VM.

According to Lampson [30], it is necessary to safeguard data from unauthorized access or modification in secure data infrastructures. Even when all unauthorized access is prevented, a service of the secure data infrastructure may not perform as advertised or leak sensitive data. While the former sometimes can be observed by a communication partner and externally certified, the latter may happen without anyone to notice. The service can be constrained ("confined") in a way that it will be unable to leak data although this does not constitute an essential property of a program, we argue different. Our reference implementation is designed to confine multiple services in a way that they form multiple layers of security, forming a collection of programs that interact with the Analyst. Even in the presence of a leak of sensitive data in one layer, the containing (outer) layer most likely will prevent data leakage since multiple well-studied and core Internet protocols such as SSH/VNC/VPN must be vulnerable at once.

4.2 Discover Candidates

To identify all possible optical covert channels, we use a modified approach of Kemmerer [28] and evaluate their suitability for covert communication using the communication model in Figure 4.1. In the original approach, Kemmerer uses a table of attributes and (file) primitives such as read, write, etc. Since we have a more sophisticated setting,

	Attribute \ Component	VPN Client	VPN -VM	Key -VM	Gate -VM	Monit. -VM	Data -VM	Installer -VM	Data Owner -VM	Remote Desktop VM	Analyst -VM	Virtualization Host
Hypervisor	State	R,M	R									
	User	R,M								R	R	
	Software	R,M	R							R,M	R,M	
	Hardware	R,M										
X11	State	R,M								R	R,M	
	Forward	R,M								R	R	
	Auth	R,M								R	R	
SSH	State	R,M	R		R					R	R	
	Password	R,M	R		R					R	R	
	Keypair	R,M	R		R					R,M	R,M	
	Tunneling	R,M	R		R					R	R	
	Copy	R,M	R		R					R	R	
GNOME	State	R,M								R		
	Brightness	R,M								R,M		
	Application	R,M								R		
Firewall	State	R,M	R							R	R	
	Ports	R,M	R		R					R	R	
	Route	R,M	R		R					R	R	
	Interfaces	R,M	R							R	R	
Artifact	State	R,M								R	R,M	
	Text	R,M								R	R,M	
	Graphic	R,M								R	R,M	
	Database	R,M								R	R,M	
	Model	R,M								R	R,M	

Table 4.1: Shared resources matrix for our reference implementation that can be R=referenced or M=modified by the Analyst.

we use components of the reference implementation instead of primitives. This is valid since we fulfill properties (a) to (c) of Kemmerer's approach.

4.2.1 Marking Algorithm

As already outlined in Sec. 2.4.2, we first start with enumerating all attributes that can be used to transfer information. The next step is to fill in the column headings for the used components. We fill the matrix in Table 4.1 with "R"s and "M"s when the resource attribute is referenced in the component and when modified (=used) respectively. After the transitive closure of the matrix (one attribute may be modified by another attribute), it is analyzed in respect to locate potential storage and timing channels. According to Kemmerer, this displays the more sophisticated channels that involve multiple attributes. We give some examples to better understand Table 4.1 as follows.

Example 1 Take the attribute "X11 Forward" for example: the Analyst may want to access applications through the X11 protocol on the Analyst-VM. The Analyst has the ability to change the state of X11 by specifying a `-X` flag to the `ssh` command, the Analyst therefore references the attribute ("R") on the Remote

Desktop-VM. This also affects the ability to forward graphics on the Analyst-VM where the Analyst can still not modify the attribute ("R"). Note that secure copy (SCP) is disabled for the Analyst as well as opening a tunnel within SSH (through configuring the SSH daemon on the temporary VMs).

Example 2 For "GNOME Brightness" the shared resources at the Remote Desktop-VM allows for a screen brightness change since it is not disabled by our implementation and installed by default. This allows the Analyst to reference and modify ("R,M") the attribute on the Remote Desktop-VM, but e.g. not on the Analyst-VM since it does not provide a graphical environment.

Example 3 The attribute "Artifact Model" may contain sensitive information that is not obvious to the Access Review Committee (c.f. Sec. 2.2.1) or the Data Owner [56]. The Analyst can train the model on the Analyst-VM ("R,M"), but cannot export it to the Remote Desktop-VM since the SSH Tunnel is disabled ("R").

We focus on storage channels, since we do not consider timing covert channels provide sufficient data exfiltration capabilities. In fact, we want to transport arbitrary information and achieve a substantial bandwidth (above the postulated threshold [59] of 100 bit/s) and therefore neglect some contributions a timing channel provides.

4.2.2 Marking Channel Legibility

With the marking procedure in Sec. 4.2.1, we gain a overview on attributes that may be modified by the Analyst. By applying Kemmerer's methodology of shared resource matrices again, we generate the summarized analysis obtainable in Table 4.2. Cells marked with "L" denote an existing legal channel with access control mechanism, "N" denote that no useful information can be gained from channel (e.g. when tampering with the SSH key pair, the Analyst cannot increase the attack possibilities), "S" denote that the process for sending and receiving is the same and "P" denote a potential covert channel. For the most attribute/component combinations, the gained information is useless for an Analyst or adversarial Data Owner to extract sensitive information out of the system.

Candidate 1 We look on the "X11 Forward" attribute and observe a covert channel candidate on the Analyst-VM. Since we store the sensitive data on the Analyst-VM as well as as all tools necessary to work with the data, and the Analyst can forward graphical applications through the window system, the covert channel is open and can be used to display sensitive data.

Candidate 2 The "Artifact Text" attribute provides a potential covert channel to the Analyst since egressed (c.f. Sec. 3.4.3) artifacts may contain sensitive information either directly or in combination with other information (linking) becomes sensitive. Countermeasures to data exfiltration used in this covert channel are known already.

	Attribute \ Component	VPN Client	VPN Server	Key Server	Gate Server	Logging Server	Data Server	Installer Server	Data Owner VM	Remote Desktop VM	Analyst VM	Virtualization Host
Hypervisor	State	L	N									
	User	L								N	N	
	Software	L	N							N	N	
	Hardware	L										
X11	State	L								N	S	
	Forward	L								N	P	
	Auth	L								N	N	
SSH	State	L	N		N					N	N	
	Password	L	N		N					N	N	
	Keypair	L	N		N					S	S	
	Tunneling	L	N		N					N	N	
	Copy	L	N		N					N	N	
GNOME	State	L								N		
	Brightness	L								L		
	Application	L								N		
Firewall	State	L	N							N	N	
	Ports	L	N		N					N	N	
	Route	L	N		N					N	N	
	Interfaces	L	N							N	N	
Artifact	State	L								N	N	
	Text	L								S	P	
	Graphic	L								S	P	
	Database	L								L	S	
	Model	L								S	P	

Table 4.2: Potential covert channels in our reference implementation

Song et al [56] use typical machine learning pipelines that produce malicious machine learning algorithms who satisfy standard quality metrics (accuracy, generalizability) that remember sensitive data without exposing malicious activity in the code. They further point out that the malicious intent may be detected when the data owner knows how "normal" parameters look like and can compare it to the malicious parameter distributions. For this reason and the low capacity, we do not consider this channel in this thesis further.

Candidate 3 The Analyst may use a forged graphic in the egressed artifact (attribute "Artifact Graphic") disguised as a legitimate graphic. We observe that the Analyst is capable to generate graphics using e.g. RStudio that contains sensitive data which may not be detected when using steganography [26] or other techniques. Again, the capacity of the resulting covert channel is minimal as Johnson and Jojodia show with embedding a 518-byte plain text message into a 24-bit image invisible to the human eye, therefore we do not consider this covert channel in this thesis.

Candidate 4 As for the "Artifact Model" attribute, we obtain that trained machine-learning models may reveal some information about the sensitive data covertly. This can again be detected by the approach of Song et al [56]. Since this attack is already described in great detail, we omit it in this thesis.

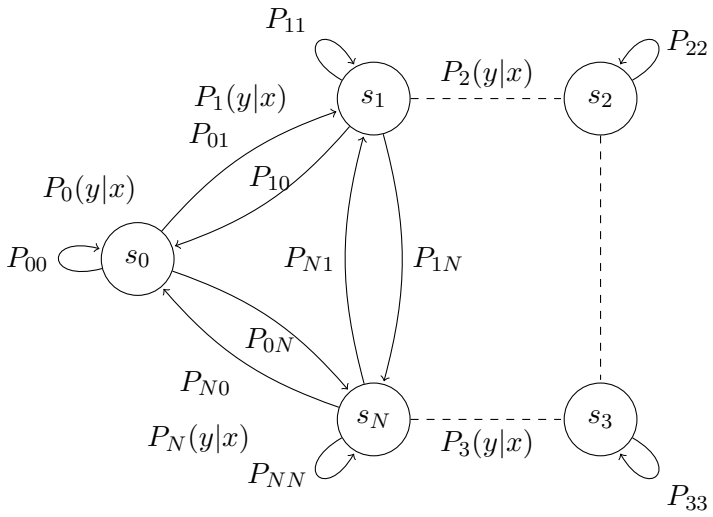

Figure 4.2: Model for the covert channel using a Markov chain with finite states.

This leaves us with only one candidate: connections using the window system on the Analyst-VM. Since we want to provide the Analyst with visible-only access to data on the Analyst-VM, this channel cannot be closed. As explained in Sec. 3.3.1, the Analyst wants to work with data available in the Analyst-VM from the Remote Desktop-VM. The covert channel uses LaTeX that is present on the Analyst-VM to generate QR-Codes and display them via the evince PDF display tool, both services that likely need to be made available on the Analyst-VM[1].

4.3 Covert Channel Analysis

Now that we have discovered a potential covert channel that can be implemented by an attacker in our reference implementation, we use the method of Markov chains [19] to model the covert channel noise varying over time which is used for channel robustness and -capacity calculation. After that, we apply the approach of Qian et al [42] to select the optical covert channel with the highest level of robustness, undetectability and capacity.

4.3.1 States

Using the original approach of Goldsmith et al [19] this results in the time-varying noise as a Markov chain displayed in Figure 4.2. It displays N different discrete (time steps are discrete), memory-less (states do not depend on the outcome of others) channels that form an irreducible (only one communicating class = the state space), periodic, stationary (distribution of a variable) Markov chain. This means a time-varying noise

[1]For example the R statistic language depends on a full installation of TeXlive on Rocky Linux at the time of writing

(of the data transmission) can be modeled and calculated using conditional probabilities on different system states determined by the channel noise. In short, the noise of the channel can be graded into several states (that all depend on the previous and initial state) and correspond to, a thorough explanation is given in Qian et al [42]. We can use this approach, because the properties of our QR-Code Optical Covert Channel are equal: our transmission is ordered, whenever a transmission is not successful, the received QR-Codes are still in order and are not re-ordered during transmission. For example: the Analyst transmits QR-Codes $\{c_1, c_2, c_3, c_4, c_5\}$ and due to noise in the channel only receives $\{c_1, c_3, c_4\}$.

As for their model, we use a simplified reduction that is explained in the following. Since an instance of their covert behavior channel uses legitimate protocol sequences (such as from the File Transfer Protocol) and our approach uses QR-Codes, we need to create an instance of the QR-Code Optical Covert Channel by using the definition of the covert behavior channel of Qian et al. Their instance (S, W) consists of a code set $s_i \in S$ that maps to a sequence of a word $w_i \in W$ of a collection of words W. For example, when constructing a encoding table $W = \{w_{j_1} = 1, w_{j_2} = 2, w_{j_3} = 3, w_{j_4} = 4\}$ and the Analyst wants to transmit the code set $S = \{s_3, s_2\}$, the instance maps $s_3 \mapsto w_{j_3}$ since w_{j_3} is encoded to 3. This applies for $s_2 \mapsto w_{j_2}$ analogously.

We informally construct an instance of the QR-Code Optical Covert Channel (S', W') via setting the original code set $S = S'$. The instance construction therefore holds and we can use the equations of Qian et al to determine capacity and robustness.

4.3.2 Robustness

In a noise-free environment, a covert channel has no need for robustness since the information embedded is never influenced by e.g. transmission errors. However since we operate our optical covert channel candidates not in a noise-free environment, we depend on robustness of the information embedded by implementing redundant information encoding via error correcting codes. We use the work of Qian et al [42] to compute the robustness bound and map the robustness metric to the modified robustness metric in Equation 4.3.2. We do not have a number of command sequences, but a number of QR-Codes N. Also, instead of the length of the command sequence L we use the QR-Code error correction E. Since we describe an optical covert channel we can set this equal as the robustness depends on the error correction level of the QR-Code being transmitted.

$$(1 - \frac{2}{N}) + \frac{2}{N} \cdot \underbrace{\frac{1}{E}}_{L} \tag{4.1}$$

Figure 4.3 displays the theoretical lower bound ξ_{low} for the different QR-Code configurations possible for the *qrencode* library used in the optical covert channel (c.f. Chapter 5)

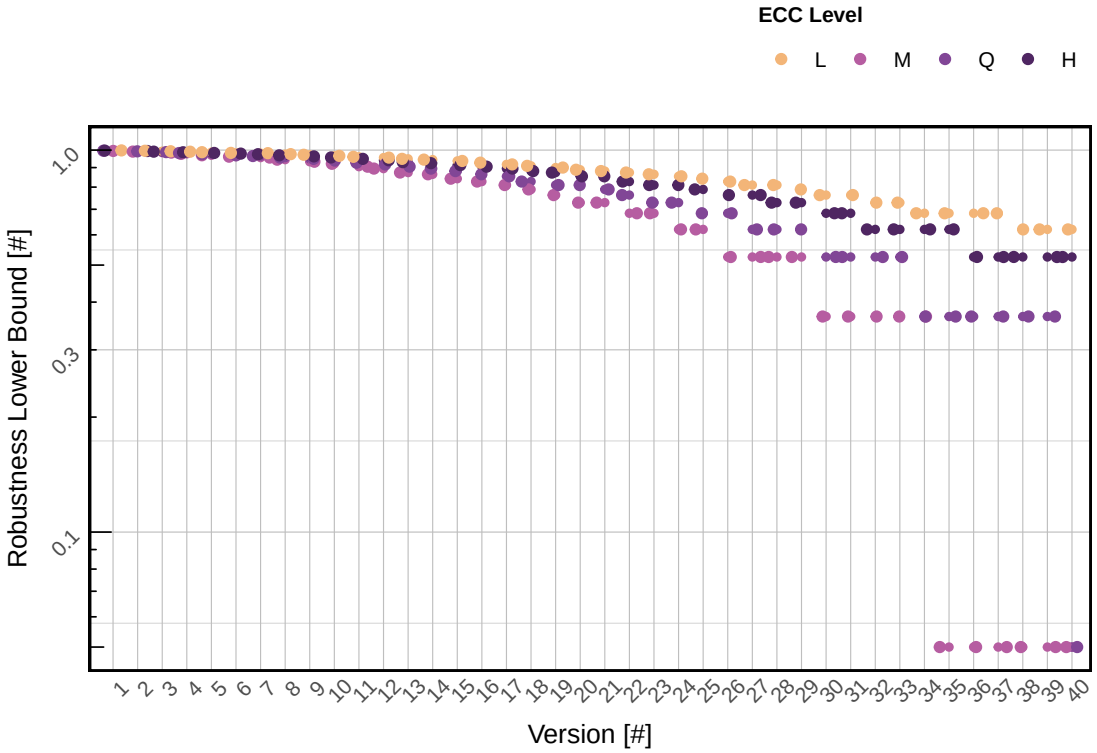

Figure 4.3: Calculation of the robustness lower bound ξ_{low} for QR-Code version 1 to 40 and different error correction levels (L being the lowest, M, Q and H being the highest)

and subsequently in the LaTeX document to create them. We see that the lower robustness bound ξ_{low} for low versions (up until 10) are close to 1, describing a very robust covert channel. This declines slowly for up until versions 35 onwards where it erratically jumps to very low robustness values, the smallest being $5.28 \cdot 10^{-6}$.

4.3.3 Capacity

The capacity of a covert channel is the amount of information transmitted over time. Qian et al [42] propose a modified approach of Goldsmith [19] to determine a covert channel capacity that models the real-world better, since it does not assume a noise-free covert channel. We can compute the capacity of the covert channel with N being the number of states, finite input alphabet X (e.g. input x_n of the covert channel at time n) and output alphabet Y (e.g. output y_n of the covert channel at time n) c.f. [19], [42]. Since we do not know the exact probabilities p_1, p_2 of the state model in Figure 4.2, we have to approximate the state change probabilities with the observed accuracy η and lower robustness bound ξ_{low}. We modify the capacity metric to retrieve the modified capacity metric shown in Equation 4.2.

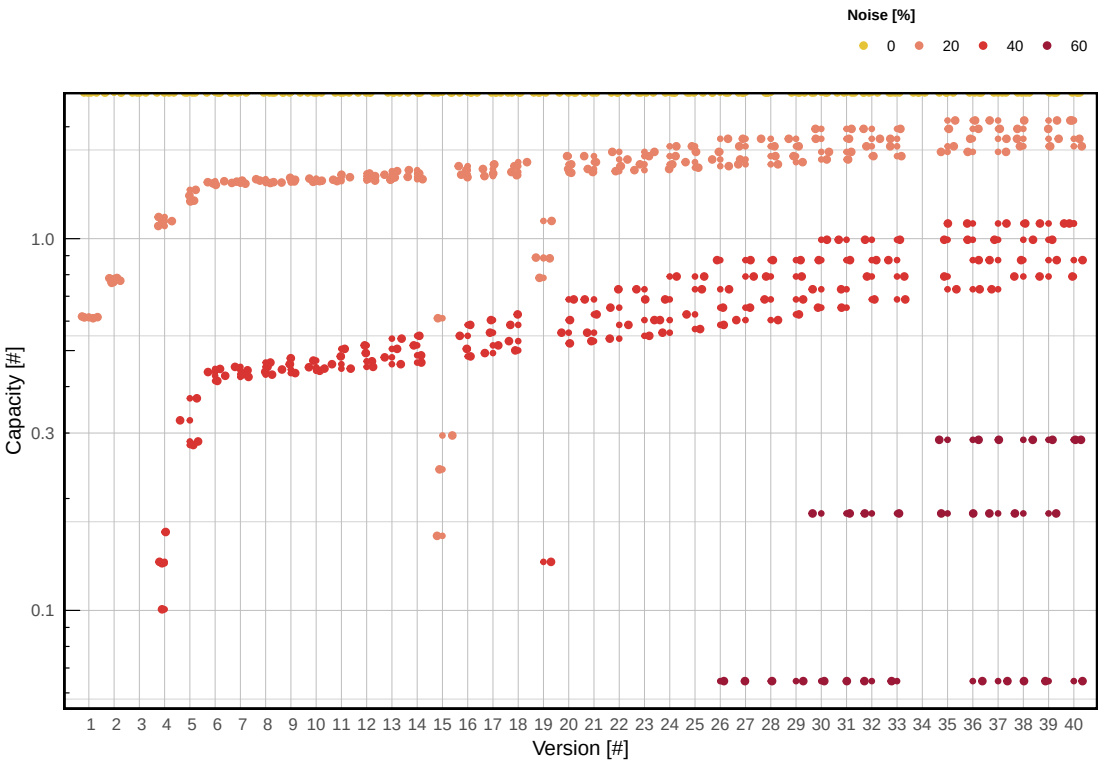

Figure 4.4: Estimation of the covert channel capacity for different versions and noises.

$$\frac{1}{n}\sum_{i=1}^{n}\Bigg(-\log\sum_{i=1}^{N}\overbrace{\lambda\cdot N\cdot\eta}^{p_1}\cdot\overbrace{(1-\eta)\cdot\lambda}^{\rho_i(l)}+\log\sum_{l=1}^{N}\overbrace{\lambda\cdot N\cdot(1-\eta)}^{p_2}\cdot\overbrace{(1-\eta)\cdot\xi_{\text{low}}}^{\pi_i(l)}\Bigg) \tag{4.2}$$

We map the original conditional state distribution $\rho_i(l)$ to a product of the inverted noise rate $1-\eta$ and the accuracy λ to describe the conditional state distribution in our covert channel. As for the original state probability p_2, we need to approximate again for the conditional state output probability (for each state l from $1\ldots N$) and use a product of accuracy, number of states and the inverted noise to describe the successful output receiving $\lambda\cdot N\cdot(1-\eta)$. We describe the conditional state distribution for in- and output $\pi_i(l)$ as the product of the inverted noise rate $1-\eta$ and lower robustness bound ξ_{low} from Sec. 4.3.2.

From Figure 4.4, we observe that the capacity is nearly equal for all versions when the noise is zero. We observe that the theoretical capacity for some configurations is nearly identical for high versions as the dots cover each other in e.g. configuration ($size = 20$, $ver = 26$, $ecc = \text{L}$). We do not distinguish the attribute *size* and *ecc* in the plot.

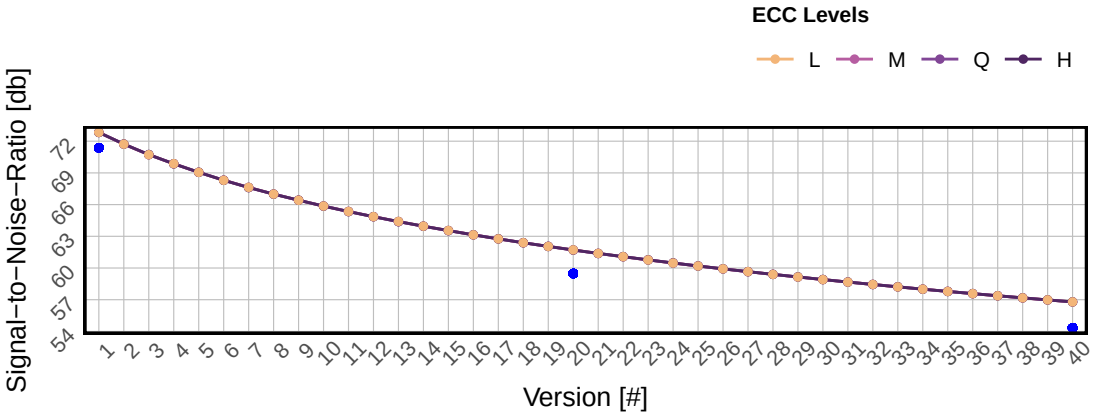

Figure 4.5: Signal-to-Noise-Ratio for QR-Code versions 1 to 40 and ECC levels L to H. Blue dots are reference values for the same image.

4.3.4 Undetectability

We use the work of Johnson et al [25] to obtain the SNR methodology and Hadjuk et al [22] to compare our results with the reported results. Since Hadjuk et al use Lena and QR-Codes too to embed information in an image, we can put some of our results in perspective. Since we use Steganography to hide images within images, the SNR methodology is a valid metric to determine the covertness of the channel. A system can take automated screenshots and use contour detection by Suzuki & Abe [57] to find the biggest shape (triangle, rectangle, pentagon) containing the QR-Code with high likelihood since (1) performing the Steganography decoding is computationally expensive; (2) reading of the decoded screenshot is prone to error when the noise is complex.

In Figure 4.5, we obtain the SNR for multiple QR-Code configurations. As expected, the SNR values for different ECC levels are nearly identical since the contained information is the same. We marked the relevant comparison values from Hadjuk et al [22] with blue dots since the configurations align with the reported configurations in the paper. We see that our SNR is constantly higher than the comparison values and interpret this as that our approach is not significantly worse than the approach by Hadjuk who extensively manipulate the QR-Code before embedding it into the image.

4.4 Summary

In this chapter, we defined the covert channel communication model that describes the involved components of the secure data infrastructure where a transmitter component sends a covert information to the receiver component without being detected by the detector component. We use a well-studied method that evaluates shared resources in the reference implementation that potentially can carry a covert channel and marks four

potential covert channels. Even though all four are promising candidates, we select only one optical covert channel that is not evaluated in well-known literature to the best of our knowledge. We further analyze the newly found covert channel candidate and evaluate the states and the properties of robustness, capacity and undetectability with a recognized statistical approach.

CHAPTER 5

QR-Code Optical Covert Channel

This chapter describes the novel QR-Code Optical Covert Channel in ample detail. First, we give an overview on the seven steps in Sec. 5.1, then propose an asynchronous communication protocol in Sec. 5.2 to explain how information is exchanged in the secure data infrastructure reference implementation and summarize on the chapter in Sec. 5.3. This channel matches the covert channel definition of Simmons [52] and was detected using the shared resource methodology of Kemmerer [28]. Having an optical covert channel specification from chapters 2 and 4, we construct a backdoor that enables a successful delivery of the covert channel.

5.1 Overview

Since our secure data infrastructure reference implementation virtually air-gaps all sensitive components from the open Internet and closely monitors all interaction on the edge and within the system, it is not a good idea for an Analyst to bluntly transmit sensitive data through the virtual networks and firewalls. In our selection process we already found that only optical ways can be used by an Analyst to illegitimately extract sensitive data out of the system, possibly without sufficient oversight of the security mechanisms. To do this efficiently, an Analyst must define a protocol that handles the life cycle of the information flow entirely which consists of seven steps:

(1) Manipulate the data set and extract only the most relevant data. Encode the extracted data set using a symmetrical method. Optionally this step can include the use of encryption to minimize the chance of detection to establish a secure covert channel with plausible deniability.

(2) Slice the encoded data stream into chunks of the available capacity for the QR-Code. This step also includes formatting the input (e.g. removing white space that can be restored later) to maximize the available capacity.

(3) Embed the QR-Codes into the masking image such that the QR-Code is not visible to the human eye using the sum method.

(4) Transmit the encoded information using a uni- or bidirectional channel. Our QR-Code Optical Covert Channel does not require a bidirectional channel, since the receiver does not need to echo back information e.g. acknowledge successful transmissions.

(5) Extract the QR-Code from the captured screenshot through feature recognition.

(6) Read the encoded information of the extracted feature image using Optical Character Recognition technology.

(7) Decode the received data which possibly includes decryption if used before in step two.

5.2 Communication Protocol

We explain the steps in the following sections in detail.

5.2.1 Encode

This step is not trivial and requires careful consideration and evaluation of available technology and techniques. There is a broad selection of existing encoding schemes available that transform raw binary data into a cipher that can be exchanged in channels that do not allow binary data transmission or are not 8-bit clean e.g. early *ASCII*. The binary encoding scheme *base64* is considered best-practice as it uses a large character set to encode binary information and is largely compatible with modern operating systems since it only uses 64 characters of ASCII.

This increases the file size by approximately $\approx 33\%$. A more storage-efficient way to encode binary data would be to use *base85* that increases the file size by approximately $\approx 25\%$, but uses computationally more expensive operations. Unfortunately, this package is not available in a standard installation of the Remote Desktop-VM or Analyst-VM, we are therefore left only to choose from hexadecimal encoding *xxd*, *base32* and *base64*. Due to the large overhead of additional data, hexadecimal encoding is not considered as candidate. The more restricted *base32* encoding uses only half of the ASCII characters available to base64 which results also in a approximately double-sized overhead. We therefore only consider base64 in the encoding step.

5.2.2 Slice

Rahim et al [43] use *base64* for hiding sensitive data in arbitrary file containers to printable text using the 64 ASCII characters available. Theoretically, the encoded information now can be displayed in the terminal (console) and recorded using OCR systems. This approach however is not reliable enough to extract data. For Arabic letters, Dinges et

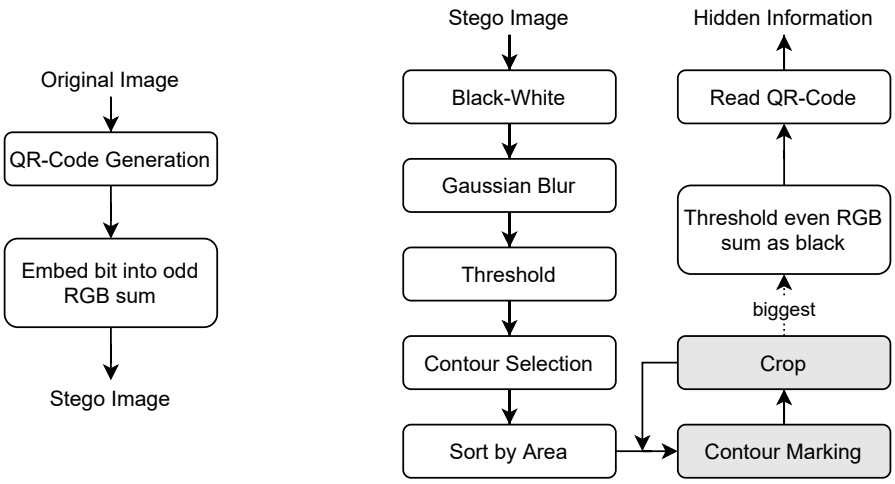

Figure 5.1: Processes to encode data into the original image and retrieve information of the Stego (encoded) image to obtain the hidden information

al [11] developed an OCR method that can recognize them with a confidence interval of in between 93.2 to 94 percent. Hassan et al [23] can recognize Bengali numerals at 96.7 percent confidence, Tamil script [31] at 81% percent confidence and finally Bedruz et al who developed a OCR approach for the Latin alphabet [2] that recognizes letters with a confidence of 90.75%. We claim that using QR-Codes and OCR technology, this can be improved significantly as we propose in Sec. 5.2.7.

Geo et al [16] state that single dimension bar codes such as e.g. EAN, UPC-A offer very limited capacity and QR-Codes offer the highest capacity for both binary and alphanumeric data, we favor them in regards to *DataMatrix*, *PDF417*, etc. We generate QR-Codes that optically transmit the sensitive information encoded in base64 through a optical stream of images based on Equation 5.1. It describes the function $n : \mathbb{N} \mapsto \mathbb{N}$ that takes a natural number $size$ as input that is the absolute length of the original sensitive data in bit and outputs the number of QR-Codes required to send the encoded message. This depends on the length of the base64-encoded message size returned by $base64_{es} : \mathbb{N} \mapsto \mathbb{N}$ that takes $size$ as input and maps it to the size of the base64-encoded message in bits. The number of QR-Codes required increases, when the capacity $cap : \mathbb{N} \times \mathbb{N} \mapsto \mathbb{N}$ decreases. It takes the version ver and ECC level ecc as input and returns the available number of bits available for storage. For example, a QR-Code of version 3 with ECC level H has a capacity of $cap(3, H) = 192$ bits.

$$\mathrm{n}(size, ver, ecc) = \left\lceil \frac{\mathrm{b64}_{es}(size)}{\mathrm{cap}(ver, ecc)} \right\rceil \tag{5.1}$$

Furumoto and Morii [15] evaluate *obfuscated* QR-Codes (that are not readable by the conventional method of Euclidean encoding) and propose a method that increases

readability of these QR-Codes, especially in situations where the conventional method does not decode the code anymore. Their method works with obfuscated QR-Codes that suffer from image noise (e.g. stains) up to 46%. Since binary data would offer the highest capacity in QR-Codes, an Analyst would probably use this encoding type. However this is not possible for our secure data infrastructure, because the LaTeX implementation does not offer native support for creating QR-Codes directly, but only text-based QR-Codes, offering only limited capacity.

```
#!/usr/bin/env python
import os
import sys
import cv2 as cv
from PIL import Image
from shape.QrTools import generate
im = Image.open(os.path.join(sys.path[0], "mask.png"))
qr = Image.open(os.path.join(sys.path[0], "qr.png"))
qr_pixel_map = qr.load()
im_pixel_map = im.load()
for x in range(qr.size[0]):
  for y in range(qr.size[1]):
    pixel = im_pixel_map[x, y]
    if qr_pixel_map[x, y]:
      if sum(pixel) % 2:
        continue
      else:
        change = True
    else:
      if sum(pixel) % 2:
        change = True
      else:
        continue
    if change:
      im_pixel_map[x, y] = (pixel[0] - 1, pixel[1], pixel[2])
im.save(os.path.join(sys.path[0], "output.png"))
```

Algorithm 5.1: Encoding of secret information bits within the red color channel of each pixel within an image, starting from the left top.

Since we enable Analysts to use the Remote Desktop-VM to access the Analyst-VM which has a full installation of TeX live present in order to generate reports, it can be used to generate a QR-Code using a very basic template and the *qrcode* package (which does not rely on the *qrencode* software package). To optimize the generation, we trim the base64 output to remove newlines after 76 characters (as required by RFC 2045), this reduces the number of QR-Codes generated by approximately $\approx 11\%$.

5.2.3 Embed

The process takes a masking image with high entropy (pixel color diversity) and generates a QR-Code for the base64 encoded data. The bit embedding follows a similar approach

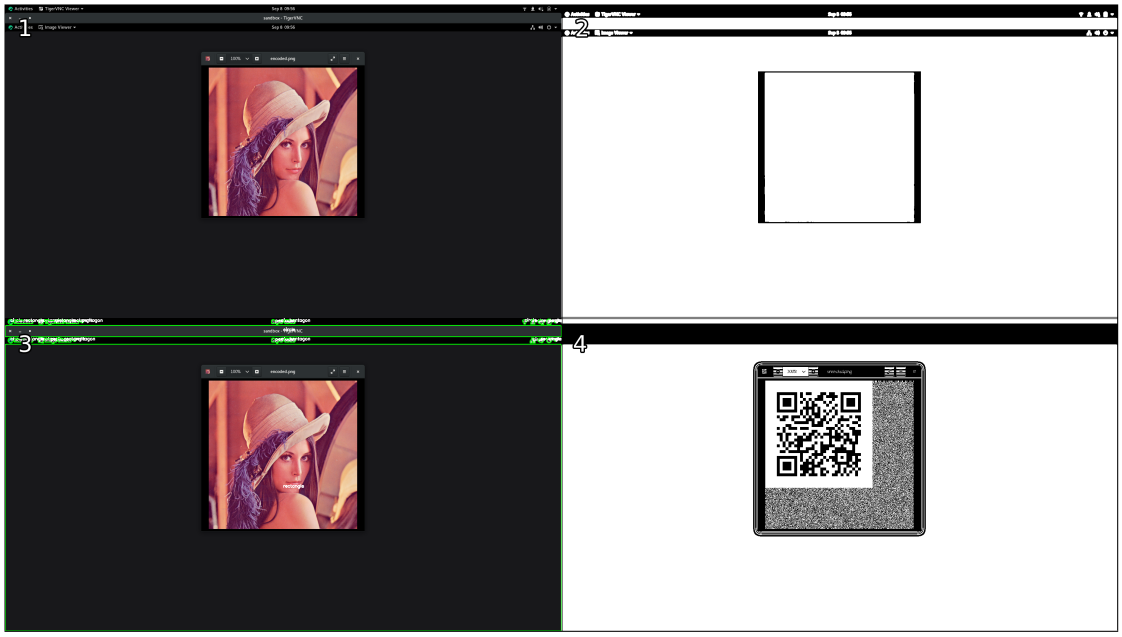

Figure 5.2: Artifacts generated during obtaining the screenshot (1), apply the black/white, blur and threshold (2) to the screenshot and running the contour algorithm (3), selecting only the biggest area and decoding it (4).

as to LSB encoding. We encode a secret bit into the red color channel of each pixel in Algorithm 5.1 where the sum of red, green and blue is odd by reducing the red color by one. The process is on the left side of Figure 5.1 and only contains two steps.

5.2.4 Transmit

In the fourth step that is executed between the Remote Desktop-VM and the Analyst-VM, the encoded data still needs to be transmitted to the Remote Desktop-VM to egress it out of the reference implementation. Contrary to the Remote Desktop-VM that comes with a full desktop environment, the Analyst-VM only has a window system environment (we use Rocky Linux' *base-x* group) installed in the standard configuration to not introduce additional software packages exposed to the sensitive data that can be used as tools for an Analyst. The Analyst displays these codes in an automatic fashion using the default image viewer available in GNOME and through the window system.. To allow the Analyst visual access to the graphical applications on the Analyst-VM, we must offer at least visual access. The difficulty (as evaluated in the experimental evaluation) is to preserve the hidden property through the Remote Desktop-VM, the approach is not very robust.

The Analyst notebook makes systematic screenshots using the *pyscreenshot* library and *Python*. In our proof of concept, the Analyst notebook has a standard GNOME workstation environment (e.g. RPM-based operating systems) installed and the screenshots

are captured directly through GNOME's *dbus* message middleware and saved to the local storage as image. Since the Analyst notebook and the Remote Desktop-VM cannot communicate automatically, the Analyst needs to make at least twice as often screenshots as the Remote Desktop-VM changes the QR-Codes to display according to the Nyquist-Shannon sampling theorem [51].

5.2.5 Extract

Recall Figure 5.1, on the right side we can obtain the non-trivial process of obtaining the encoded image with optional steps marked in gray background color. Since we want to achieve a robust selection of the encoded image to better read the QR-Code in the last step (it is computationally expensive), we transform the screenshot containing the Stego Image into black and white, apply a 5×5 Gaussian blur over it (making it less sharp) and apply binary threshold to generate a binary image, containing only white $(255, 255, 255)$ and black $(0, 0, 0)$. This allows for the contour detection algorithm [57] to select shapes and classify them. Algorithm 5.2 shows the algorithm that is implemented in the *OpenCV* package. Line 12 especially is of relevancy as it crops each region of interest (ROI) that is later marked and saved in a separate file. The artifacts and resulting feature image (containing all the ROIs) that are generated during the process are displayed in Figure 5.2. The algorithm returns the largest area nevertheless, the ROI marking is optional but may increase the likelihood to read the decoded QR-Code at a later step.

```
1  def detect_contours(ifile, ffile, tfile):
2    image = cv.imread(ifile)
3    gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)
4    blurred = cv.GaussianBlur(gray, (5, 5), 0)
5    thresh = cv.threshold(blurred, 3, 255, cv.THRESH_BINARY)[1]
6    cv.imwrite(tfile, thresh)
7    cnts = cv.findContours(thresh.copy(), cv.RETR_EXTERNAL,
8                           cv.CHAIN_APPROX_SIMPLE)
9    cnts = sorted(imutils.grab_contours(cnts), key=cv.contourArea)
10   for c in cnts:
11     x, y, w, h = cv.boundingRect(c)
12     ROI = image[y:y + h, x:x + w]
13     M = cv.moments(c)
14     cX = int((M["m10"] / M["m00"]))
15     cY = int((M["m01"] / M["m00"]))
16     shape = detect(c)
17     # draw shape and text
18   return cnts[-1]
19
20 def read(img):
21   detector = cv.QRCodeDetector()
22   data, _, _ = detector.detectAndDecode(img)
23   print("Hidden information: " + str(data))
24   return data
```

Algorithm 5.2: Contour detection and marking algorithm.

5.2.6 Read

After extracting the ROI containing the QR-Code, the flipped bit in the red channel needs to be corrected according to Algorithm 5.3. Since there is for our implementation no way of guessing which bits have been flipped and which not, the whole image is transformed using the sum of RGB channel method already proposed in Section 5.2.3. In all cases where the bit is assumed to be flipped (line 7), a black pixel is drawn, in all other cases a white pixel is drawn. This results in a binary image again, at best containing the QR-Code. The information is then relayed to a Python wrapper of the *libzbar* library which uses OCR technology to extract firstly the QR-Code in the screenshot and then reads its contents.

```
1  def restore(ofile):
2    im = Image.open(ofile)
3    im_pixel_map = im.load()
4
5    for x in range(im.size[0]):
6      for y in range(im.size[1]):
7        if sum(im_pixel_map[x, y]) % 2:
8          im_pixel_map[x, y] = (255, 255, 255)
9        else:
10         im_pixel_map[x, y] = (0, 0, 0)
11   return im
```

Algorithm 5.3: Restoring the QR-Code within the encoded picture.

5.2.7 Decode

We outlined a transmission protocol that does not include acknowledgments in Sec. 4.3.1. Since the sending and receiving process are continuous and independent at a constant average rate, we can conclude that it follows the natural exponential distribution in regards to error rate. Recall that n is the number of messages transmitted, then we assume for small n, no message is dropped with high probability, but for large n, some messages inevitably will be dropped and malfunctions will occur. To slightly alleviate the inevitable error occurrence, we use error correcting codes combined with OCR technology. The reason behind this is manifold: although we directly capture the display contents from the Remote Desktop-VM, our approach (in theory) can work with captured images from the distance with some alterations.

This approach is already studied in detail by Guri [20], who uses embedded QR-Codes as additional layer on top of other displayed content and evaluates the reconstruction rate from a range of 0.5 to 8 meters. The transmission within our QR-Code Optical Covert Channel is unidirectional, which is sufficient for our purposes. As our experimental evaluation shows, the most performing 5% QR-Codes are capable of storing between $[1,465,\ 2,563]$ bit of data.

5.3 Summary

In this Chapter we defined our QR-Code Optical Covert Channel within our OSSDIP [62] reference implementation along with a specification of the communication protocol. It motivates the necessity to implement a optical- rather than the more common network communication protocol due to the closely monitored environment and strict firewall rules. We thoroughly explain the seven steps taken in the protocol that leads to communication with the Analyst wanting to extract the data over the QR-Code Optical Covert Channel. The channel uses Steganography to embed the sensitive information first in a QR-Code, then in an ordinary image by decreasing red-channel values of pixels depending on their parity (even/odd). This can easily be restored when known and due to the error correcting code property of QR-Codes, faulty transmissions still result in correct information when decoded.

CHAPTER 6

Experimental Evaluation

In this chapter we practically evaluate the performance of the established covert channel by conducting experimental tests ("benchmark"). The rest of this chapter is structured as follows: Sec. 6.1 describes the experiment set-up and overall data set description, in Sec. 6.2 the measurements before the QR-Code sending phase of the Optical QR-Code Covert Channel are presented, Sec. 6.3 present the measurements during the transmission phase and Sec. 6.4 presents the measurements after the transmission phase of the Optical QR-Code Covert channel.

6.1 Overview

In this section, we give an overview of the experimental evaluation methodology.

6.1.1 System Deployment

Our experiment set-up consists of the standard, publicly available, `OSSDIP` reference implementation deployed on a general-purpose cluster with enterprise server hardware at TU Wien. From the available quota, we currently use three virtual machines with a total of 34 vCPUs and 132GB RAM and 870GB SSD storage with a large bandwidth Internet connection. We deploy the reference implementation as nested VMs inside the largest VM "Virtualization Host" (32 vCPUs, 128GB RAM, 850GB SSD) running on Intel Xeon (Broadwell) hardware and available hardware virtualization for KVM. The Analyst notebook is standard commodity hardware (4 CPUs, 8GB RAM, 500GB SSD) provided by TU Wien as part of our regular tenure. They are connected through the internet service provider and commodity network hardware (TP-Link Archer C80, 1300 Mbps) located outside of TU Wien's premises.

6.1.2 Software Methodology

The software in use by OSSDIP is described in detail by Weise and Rauber [62]. For the data manipulation on the Analyst notebook, we use the Rocky Linux[1] operating system and the statistical computing language *R* along with *RStudio*. We used LibreOffice Calc[2] for aggregating the data set and publishing. In order to establish the covert channel on the Analyst-VM, the Python libraries *qrcode*, *OpenCV*, *Pillow*, *imutils* and *numpy* need to be present. Since we allow the Analyst also to perform neural network analysis, all of the dependencies are met.

6.1.3 Statistical Heuristics

We use a simple encoding heuristic H_e defined in Equation 6.1 to find the most covert configuration from the 800 possible configurations. The heuristic penalizes QR-Codes with small capacity $\text{cap}(ver, ecc)$ in bits, e.g. version 1 evaluates significantly worse than version 40 since it results in a higher H_e, therefore more likely to raise suspicion. Similarly, configurations that require more QR-Codes $n(size, ver, ecc)$ to transmit a data block than other configurations are penalized as well as configurations with high generation duration d_g.

$$H_e = \frac{d_g \cdot n(size, ver, ecc)}{cap(ver, ecc)} \tag{6.1}$$

To evaluate the efficacy of the decoding after transmission, we present the decoding accuracy heuristic H_d provided in Equation 6.2. It estimates the accuracy of the optical transmission from the Analyst-VM to the Analyst notebook. We use the Levenshtein distance [67] to compare the *base64* encoded original data block and the *base64* encoded received data block (concatenation of all decoded QR-Codes). Another approach would be to quantify the similarity between two binary objects using the Russel-Rao coefficient [13]. When the Levenshtein distance $\text{l}(base64(block_{orig}), block_{recv})$ is the same as the *base64* encoded original data block $base64(block_{orig})$, the received block $block_{recv}$ is equivalent to the original data block $block_{orig}$. We use it in the accuracy heuristic H_d as numerator and the size of the received encoded block $s_d = \text{s}(base64(block_{recv}))$ as denominator and invert the result to express the similarity in percent (0% = data blocks differ entirely and 100% data blocks are equivalent).

$$H_d = 1 - \frac{\text{l}(base64(block_{orig}), block_{recv})}{s_d} \tag{6.2}$$

This creates a linear distribution of the values which we later need in the overall heuristic index. We want to select the candidates that have the lowest suspicion heuristic index H_e

[1] "Rocky Linux". [Online]. URL: https://rockylinux.org/, accessed 2021-08-27, version 8.4.

[2] The Document Foundation: "LibreOffice". [Online]. URL: https://www.libreoffice.org/discover/calc/, accessed 2021-08-27, version 6.4.7.2.

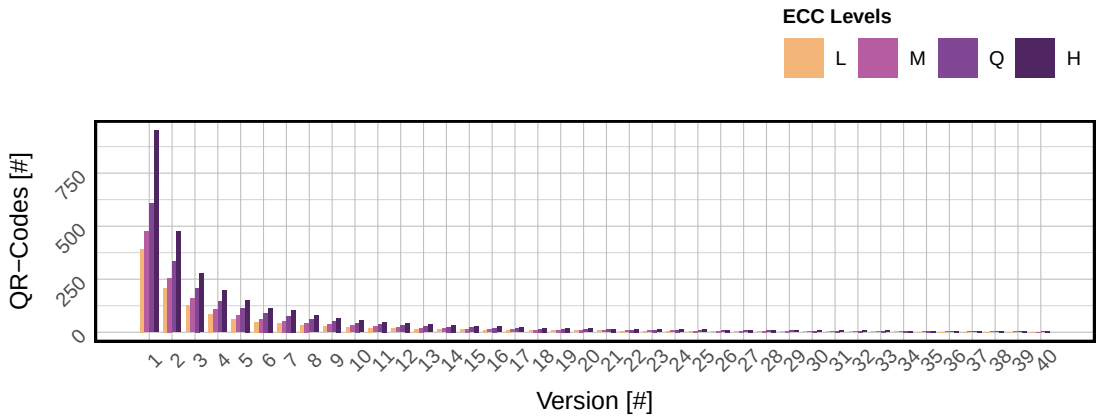

Figure 6.1: Number of QR-Codes needed to encode the given data.

and highest accuracy H_d in Equation 6.3. It rewards small suspicion heuristic values H_e and also high accuracy heuristic values H_d and evaluates to a number between 0 (worst) and 1 (best) for evaluations. Some measurements exceed this range in the negative and can reach values of $\min\{H_d\} = -219.69$ for the configuration $(size = 80, ver = 3, ecc = \text{H})$ that we later discuss are statistical outliers.

$$H_o = 1 - \frac{H_e}{H_d \cdot \max\{H_e\}} \tag{6.3}$$

6.1.4 Data Set

We identified three independent variables (i) version and therefore maximum capacity of a generated QR-Code; (ii) error correction level selection; (iii) size of the generated QR-Code that affect the outcome of the experiment steps and two dependent variables (a) number of generated codes trivially depending on version and error correction level (b) stream resolution depending on the encoding and compression settings. Our QR-Code generation data set [60] consists of $n = 39,195$ QR-Codes for 40 QR-Code versions $1 - 40$, four different ECC levels from L to H and five QR-Code image sizes (20, 40, 80, 160 and 320 millimeter). We

This leads to a specific number of QR-Codes per setting for version, ECC level and image size, this is extensively explained in Sec. 2.3. For each encoding and decoding step, we save the start and end time, count the number of QR-Codes generated for a 5 kilobyte large block of random data such that all QR-Code versions need to create at least two QR-Codes that need to be transmitted. As seen in Table 6.1, some QR-Code configurations produce (Count_e) only 2 QR-Codes $(size = 20, ver = 35, ecc = \text{L})$ for the same 5 kilobyte block while others may produce 952 QR-Codes $(size = 20, ver = 1, ecc = \text{H})$. The Duration_e describes the timespan started immediately before creation of the QR-Code

and (immediately) after creation. Similarly, the Duration_d describes the timespan between start of the QR-Code decoding and finish.

The remaining part is structured as follows: due to the complexity of the events and their separability, we split the experimental evaluation depending on their occurrence in time relative to the optical transmission of the QR-Codes in before-, during- and after transmission. First we give a short introduction of the phase, present the results that are obtained in this phase and interpret the results afterwards.

6.2 Before Transmission

In order to establish the QR-Code Optical Covert Channel, the Analyst first needs to plant the encoding script on the Analyst-VM in order to create the sequence of QR-Codes for a later step. The script needs to be typed manually into the Analyst-VM. Assuming a mean typing speed of 0.15 seconds per key (byte) as reported by Kinkead [29], this can be accomplished in approximately ≈ 97.95 seconds for the 653 characters (including spaces) in Algorithm 5.1.

6.2.1 Results

The experimental evaluation started on Sunday 21.02.2021 at 14:56:42 and took about 21 hours to complete with the last QR-Code created on Monday 22.02.2021 at 12:16:46. The original data blocks occupy a total of 4.00 megabytes while the encoded data blocks occupy a total of 4.68 megabytes on the disk. A total of $n = 39,195$ QR-Codes were generated during this time by the Analyst-VM. We visualize the encoding duration for the 800 configurations in Figure 6.3 A and interpret it in the next Section. The time

Attribute	min	max	median	mean	std.dev
Count_e	2.00	952.00	11.00	48.99	114.67
Duration_e	46.41	442.90	87.66	95.99	45.80
Duration_d	0.57	925.19	29.03	73.46	115.72
Data Size_e	6,668.00	6,668.00	6,668.00	6,668.00	0.00
Data Size_d	6,668.00	6,668.00	6,668.00	5,840.93	1,817.06
$\text{Levenshtein Distance}_d$	0.00	6,668.00	0.00	827.03	1,816.94
Accuracy_d	0.00	100.00	100.00	87.60	27.25
$\text{Stegano Duration}_e$	2.10	4.06	2.71	2.80	0.54
Index H_e	120.37	1,106.73	565.53	602.10	207.16
Index H_d	0.00	1,854.18	0.00	75.38	350.40
Index H_o	−219.69	$^{\dagger}0.99$	$^{\dagger}0.99$	0.68	7.81

Table 6.1: Optical QR-Code Covert Channel data set of attributes x before transmission x_e and after x_d transmission, values marked with †dagger are not rounded to reflect that no configuration reaches 100% in the overall heuristic index H_o ($p = 0.95$).

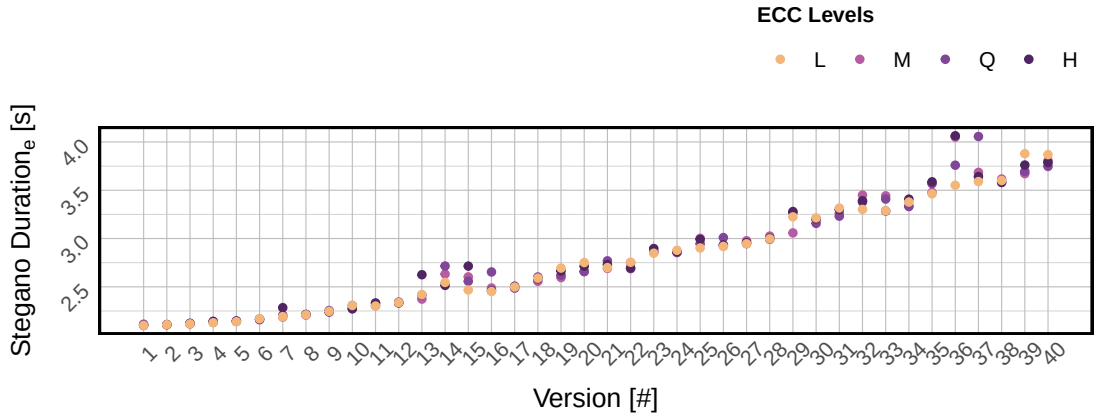

Figure 6.2: Mean duration to encode the QR-Code into the masking (legitimate) image using our image embedding approach.

needed for the steganography encoding is depicted in Figure 6.2 and shows that the encoding time increases with the version number. From a monitoring view, the opening and closing of a text editor is easily observable in the logs created by *rsyslog* at the Monitoring-VM. For each start of a text editor (e.g. *vim*), an entry is logged both locally on the Analyst-VM (in `/var/log/secure`, not readable by the Analyst) and the Monitoring-VM and shows both the editor usage and the file being opened.

6.2.2 Interpretation

We interpret the results of the previous Section and first evaluate Figure 6.3 A, it displays the encoding duration for all 800 configurations. We notice that lower ECC levels perform faster in nearly all cases, but for version 34. Otherwise it can be seen that with higher version, the generation time slightly increases too. Further, we see in Figure 6.3 B that the encoding heuristic penalizes configurations on the left side (low capacity) and rewards configurations on the right side (high capacity). With increasing capacity, the index H_e reaches nearly 0 for ECC levels L and M estimating that less transmissions with increased optical capacity will likely go unnoticed, this behavior is as expected and fits the design of the heuristic.

The QR-Code generation duration (Duration$_e$) varies largely from 46.41 seconds to 442.90

Attribute	**Duration$_e$**	**Count$_e$**	**Index H_e**
Duration$_e$		0.79	0.73
Count$_e$	0.79		0.81
Index H_e	0.73	0.81	

Table 6.2: Correlation matrix of the attributes relevant before transmitting the QR-Codes.

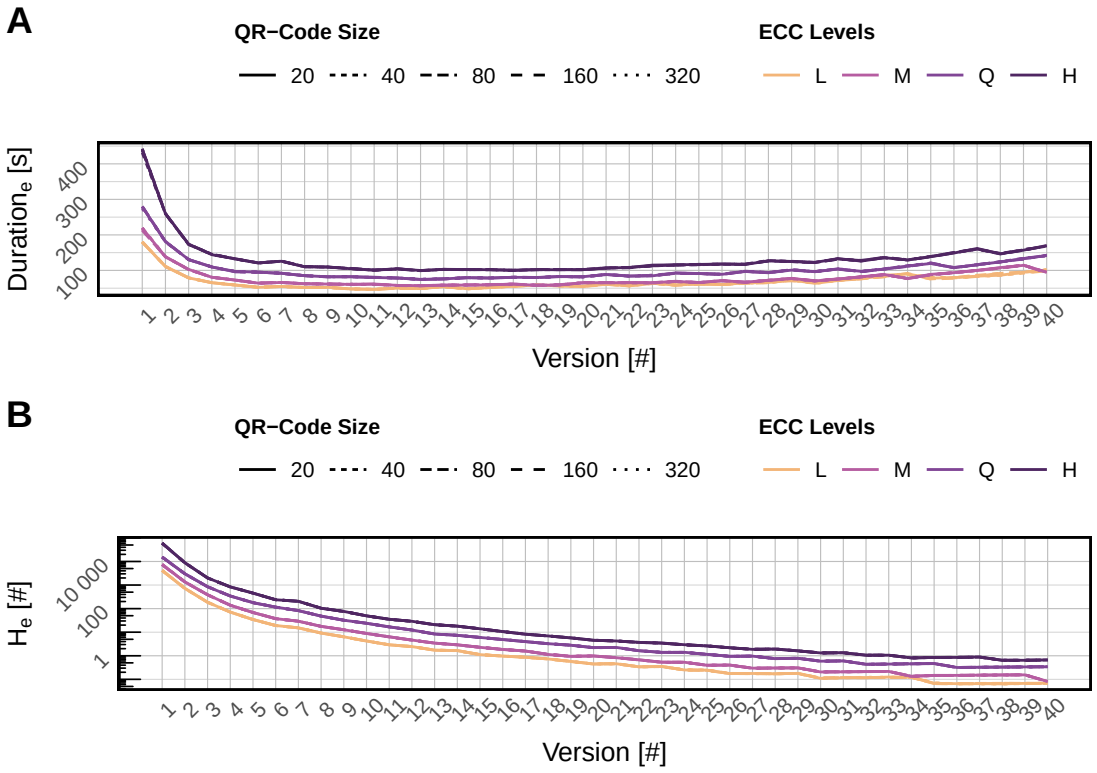

Figure 6.3: Visualization of the encoding duration (time between start of the script and finish) and the log-encoding heuristic H_e for different configurations.

seconds and an arithmetic mean of 95.99 seconds at a small standard deviation of 45.80 seconds. This means, that the data is centered near the mean. The correlation matrix in Table 6.2 shows that the encoding Duration_e correlates high with the number of generated QR-Codes (Count_e), we conclude that the *qrencode* and embedding takes longer for QR-Codes with lower capacity (as more QR-Codes need to be generated), since they correlate strongly. This is also reflected in Figure 6.3 A where up until version 3 an exponential generation time can be obtained in comparison to the other configurations with higher versions. Also, the suspicion heuristic index H_e correlates strongly with both the Duration_e and Count_e which indicates that the heuristic performs as intended (c.f. Sec. 6.1.3).

6.3 During Transmission

In this phase, the experimental evaluation consists of the saved video stream that the Remote Desktop-VM takes and sends to the Monitoring-VM. The produced video stream files are stored on the `/var` partition at the Monitoring-VM.

6.3.1 Results

Since all logs from `/var/log/secure` on the Analyst-VM are sent to the Monitoring-VM, we can examine the logs as if they would be stored on the Analyst-VM, but without the possibility to change them. The logs show that for each of the $n = 39,195$ generated QR-Codes that are in transmission, the *evince* image viewer was opened and closed. It shows the involvement of GNOME's *dbus* message middleware that communicates with the *systemd* manager. During transmission a file activity can be observed in the working directory where the encoding script is placed since it creates a *base64* encoded data block from the original data block for an encoded image and a QR-Code.

We give a statistical description of the auxiliary steps in Table 6.3. It shows a small mean screenshot duration of 0.21 seconds (does not include the time needed to store the screenshot to the storage). In rare cases, the duration of the screenshot capturing takes 0.64 seconds. Obtaining the information embedded in the screenshots (Decoding Duration) takes the *pyzbar* library between 0.10 seconds and 0.20 seconds. We present the data set of Table 6.3 also visually in Figure 6.4 and compare their behavior for each iteration.

Notice that the decoding duration only has $n = 24,308$ iterations (one iteration is producing one QR-Code) in subfigure B, compared to $n = 39,195$ iterations for subfigure A since we only consider successful transmissions (with accuracy = 100%). The yellow lines show linear modeling (the mean) and the pink line shows the generalized additive model (GAM) that tries smoothing the Screenshot Duration variable with a prediction function and enhances intractability.

6.3.2 Interpretation

The relatively low median Screenshot Duration of 0.19 seconds compared to the arithmetic mean of 0.21 shows that the duration is distributed heavily to the left (the histogram of values will have the highest density left of the mean) that shows that the methodology is fast for most of the configurations.

Unfortunately, since in Figure 6.4 the dimensions do not match we can not correlate the duration of these auxiliary phases. The missing time information of about 37.98% entries originates from only considering accurate decoding iterations during the decoding process, disregarding the unsuccessful decoding iterations. We can however take a look on the distribution of values for e.g. the Screenshot Duration. The linear fitting mean (yellow) line shows that the duration of taking a screenshot increases nearly linearly compared to

Attribute	min	max	median	mean	std.dev
Screenshot Duration	0.13	0.64	0.19	0.21	0.06
Decoding Duration	0.10	0.20	0.11	0.11	0.01

Table 6.3: Duration of auxiliary phases (screenshot duration, decoding duration).

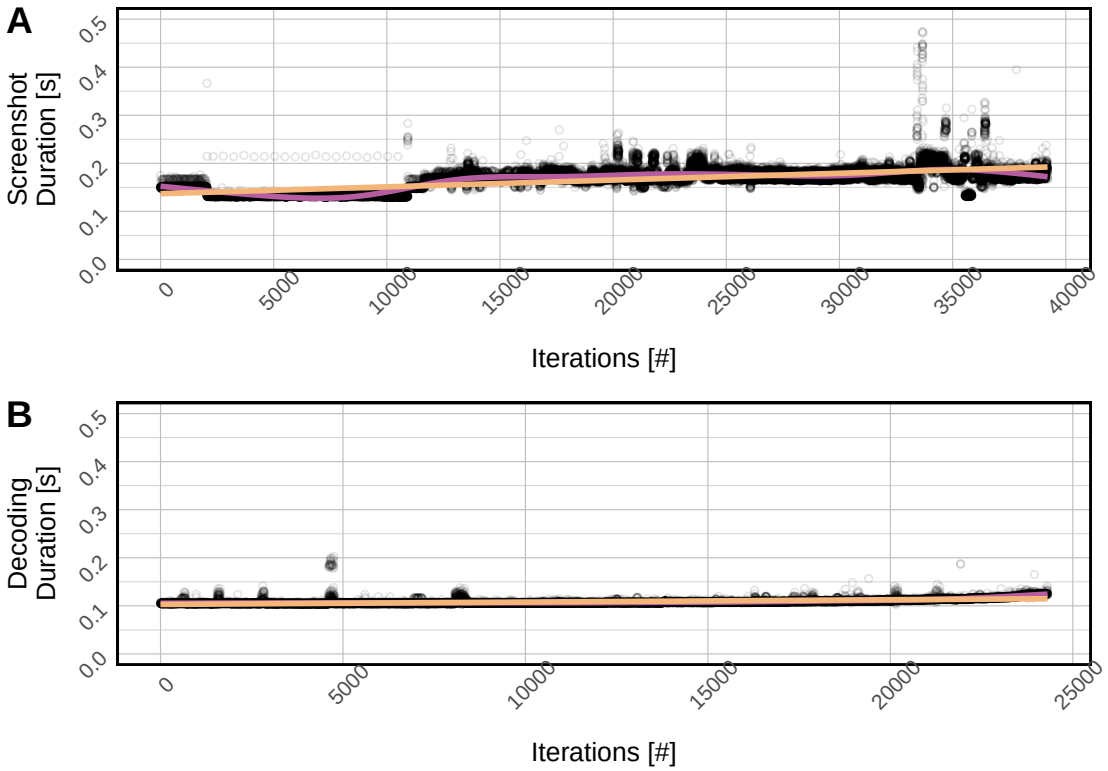

Figure 6.4: Duration of auxiliary phases (screenshot duration, decoding duration).

the (pink) line for GAM. We explain this with increased workload on GNOME's *dbus* message middleware as the experimental evaluation continues for subfigure 6.4 A, since capturing a screenshot needs to access this bus. For the decoding duration in subfigure B, we interpret it as a linear function (since the linear model in pink and GAM in yellow match) that slightly increases with the time. This may be also an increased workload on the Analyst notebook, but can have a multitude of reasons like ineffectiveness in the underlying *pyzbar* library.

6.4 After Transmission

This phase is commenced, when the Analyst has finished displaying the Stego Images and starts decoding them on the Analyst notebook. In a real setting, the Analyst would remove the generated artifacts to clear evidence after completing the benchmark. For our purposes however, we want to give insight on what artifacts were generated and where they occur along with characteristics.

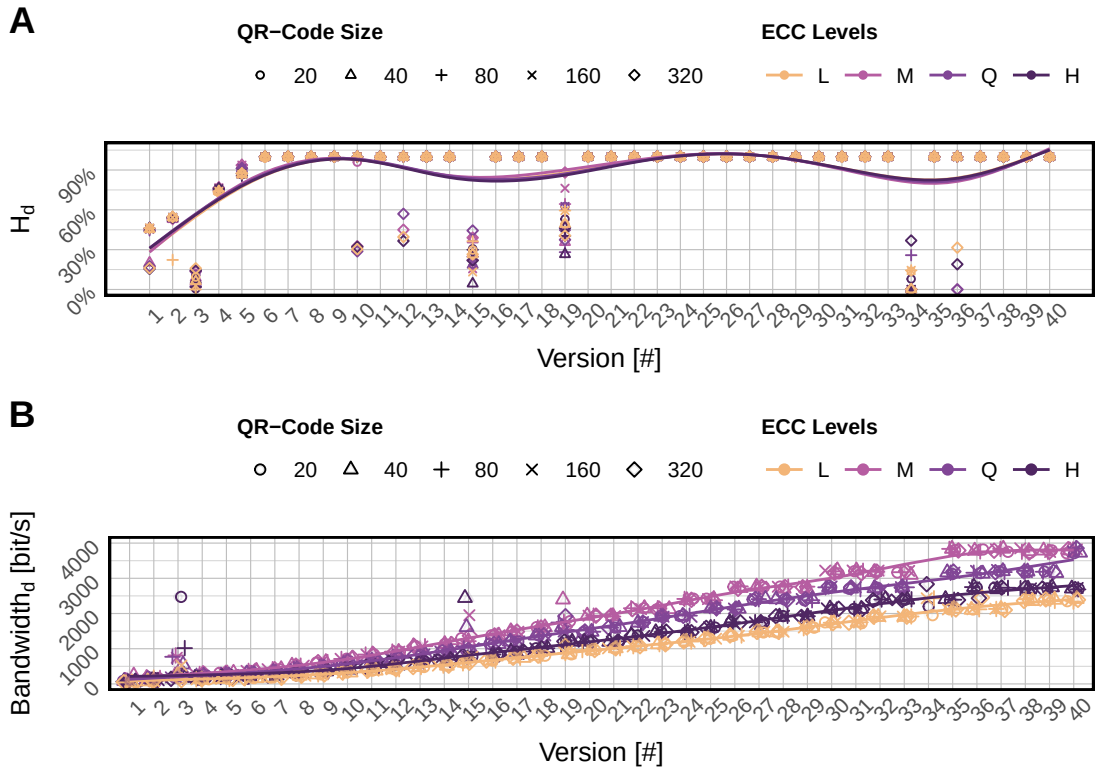

Figure 6.5: Visualization of the decoding accuracy and bandwidth for different configurations $(size, ver, ecc)$ with generalized additive model (GAM) for each ECC Level.

6.4.1 Results

Table 6.1 shows that the Duration$_d$ range is approximately 2.33 times larger than the Duration$_e$ range. To better understand and interpret the Accuracy$_d$ at a later step, we visualize it in Figure 6.5 A. From the data obtained through the evaluation this accuracy is calculated through comparing the original data block with the decoded data block at the Analyst notebook (c.f. Sec. 6.1.3). The Bandwidth$_d$ is displayed in subfigure B for all 800 configurations. It is calculated as the original data block size in bits (Data Size$_e$) divided by the decoding Duration$_d$.

Sorting the overall heuristic index H_o by number (highest first), we find the most promising candidates for the Optical QR-Code Covert Channel. Table 6.4 shows the three best QR-Code configurations in terms of our overall heuristic index H_o. The best configuration $(size = 320, ver = 38, ecc = \text{L})$ needs about 86.31 seconds to generate 2 Stego Images and about 10.73 seconds to transfer from Analyst-VM to the Analyst notebook. From the 800 configurations, exactly 626 configurations reached an accuracy of 100%, this makes the mean accuracy of our method 78.25%. The remaining data can be viewed in the publicly available data set [61]. As seen in Figure 6.6, the encoding

mechanisms Hextile and ZLRE are performing the best, allowing a successful extraction of the Stego Image through VNC at all JPEG compression levels.

When no compression is used, all encoding mechanisms are suitable for the Optical QR-Code Covert Channel. However, if using even the slightest compression the Tight and Raw encoding mechanisms of VNC are not transmitting the Stego Image correctly anymore. We want to note that changing the JPEG compression does not affect the validity of the transmission. Inspecting the TigerVNC source code reveals that the JPEG compression option is not used within the implementation.

6.4.2 Interpretation

In this section, we present the results of our experimental evaluation to find good candidates for the QR-Code Optical Covert Channel through the use of encoding and decoding heuristics. We obtain that QR-Code configurations with versions 1 to 5 perform significantly worse than versions 6 to 9 as well as 13, 14 and 16 to 18 and 20 to 33, 35 and 37 to 40 in terms of the decoding heuristic index H_d (accuracy). Especially versions 3, 15, 19 and 34 perform significantly worse than others as obtainable in the subfigures A and B. They are showing statistical outliers, since the reported values for the heuristic decoding index H_d are not near the generalized additive model that aligns for all configurations.

The indices of the best QR-Code configurations are relatively close. We interpret the bandwidths of the best QR-Code configuration of 3, 729.53 bit/s. We want to compare this bandwidth with more naïve approaches, like typing on a typewriter to put the achieved bandwidth in perspective. We only consider the general case where binary data is transmitted, thus some form of encoding needs to be chosen in order to type it on a typewriter, we consider *base64* again. A trained typist can reach speeds of 0.15 seconds per key (byte) as reported by Kinkead [29]. We assume that this also holds for *base64* encoded texts which results in a bandwidth of 39.99 bit/s, significantly less than our reported bandwidth.

$$\frac{1}{5000} \cdot \frac{15}{100 \cdot 8} \cdot 6668 \approx 39.99 \text{ bit/s} \tag{6.4}$$

Therefore an approximate time of 125 seconds to accomplish this task as we need to

#	Version	ECC	Size	Index H_e	Index H_d	Index H_o	Bandwidth
745	38	L	320	0.06	1.00	†0.99	3729.53
742	38	L	40	0.06	1.00	†0.99	3769.35
741	38	L	20	0.06	1.00	†0.99	3721.62

Table 6.4: Best three QR-Code configurations, values marked with †dagger are not rounded to reflect that no overall heuristic index reaches 100%. The leftmost column is the row number in the publicly available data set.

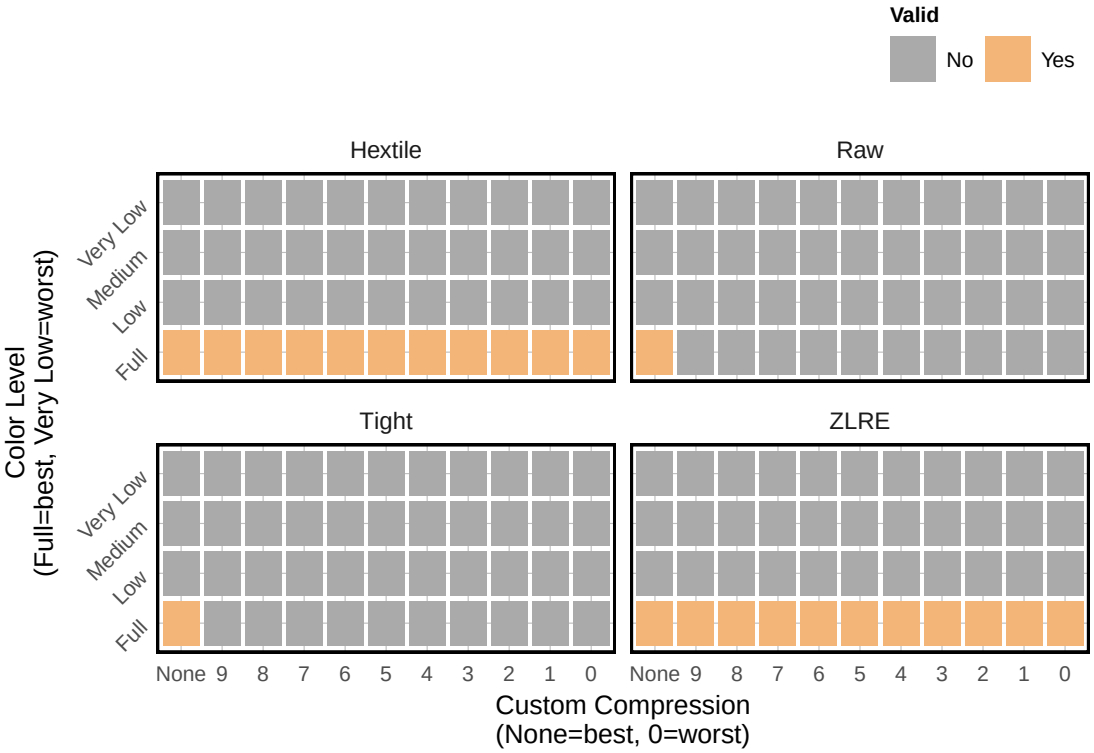

Figure 6.6: Decoding accuracy of a QR-Code ($size = 320, ver = 38, ecc$ =L) embedded via red channel encoding in a legitimate image by using Steganography.

convert the 0.15 byte/s to bit/s and multiply it with he encoded random data block size (c.f. Equation 6.4).

6.4.3 Detection

We experimentally evaluate the system behavior to conclude, whether the QR-Code Optical Covert Channel can be detected. Since our reference implementation logs every interaction with it, the Monitoring-VM records all activities that happen both on the Virtualization Host and within the guest-VMs. Our reference implementation sends all logs (wildcard $\star.\star$) to the Monitoring-VM, no activity of generating the QR-Codes using the Python script is captured in the logs. As for the systematic displaying of the QR-Codes, we were able to find traces of this activity in the Monitoring-VM. For each QR-Code display, we found entries of GNOME's *dbus* message middleware that requests a activation of the *evince* service, a subsequent log of the *systemd* manager itself and one line of a successful start of the *evince* application. `OSSDIP` records the VNC stream of the Remote Desktop-VM and stores it at the Monitoring-VM. Additionally the Data Owner has the possibility to access the VNC video stream through a VNC proxy and view the Analyst working live.

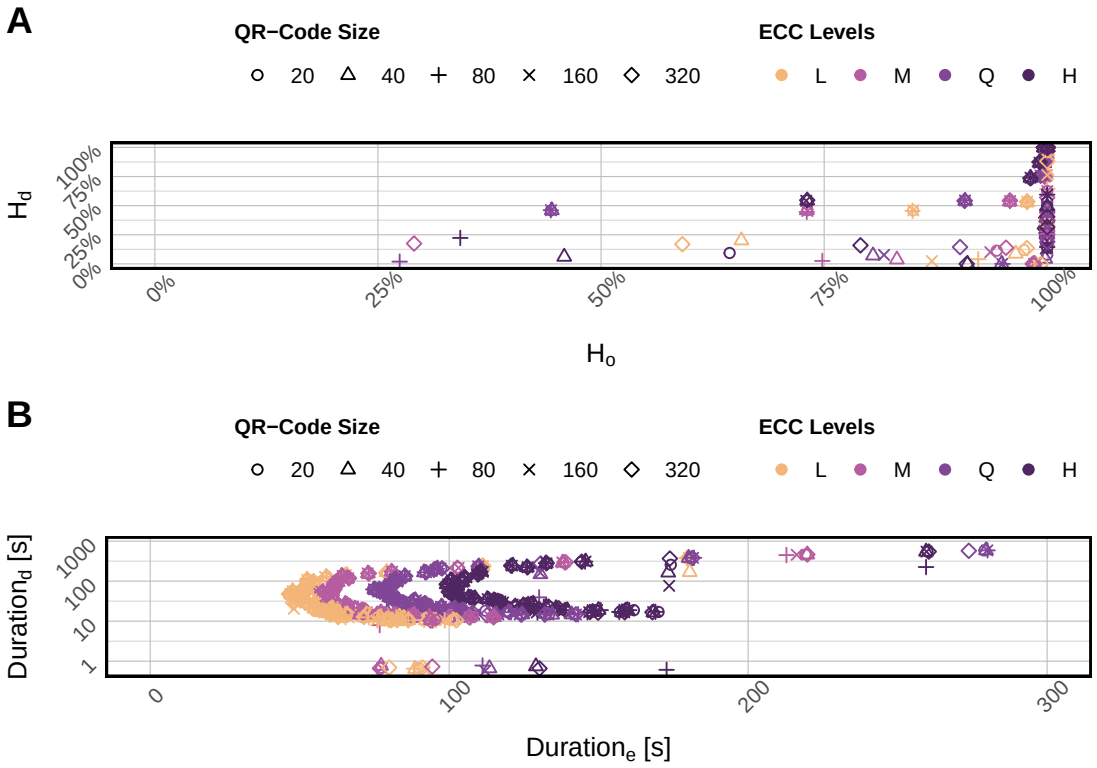

Figure 6.7: Visualizations of correlations between duration and the decoding heuristic index/overall heuristic index.

Similarly to Hadjuk et al [22] and Wu et al [65], we determined the peak Signal-to-Noise-Ratio (SNR) to compare the covertness of our Stego Image (within the screenshot) and obtain Table 6.5. Wu et al do not disclose the QR-Code version.

6.5 Summary

We practically evaluated the performance of the established covert channel through experimental evaluations on a deployed system. Additionally, we presented the software used and propose heuristic indices that model the behavior of the covert channel before,

Version	1	20	40
PSNR	72.83	61.70	56.79
PSNR [22]	71.38	59.48	54.33
PSNR [65]	30.85		

Table 6.5: Comparison of our approach with research (lower is better).

during and after the transmission of images that carry sensitive information embedded via Steganography and QR-Code technology. Further, we described the data set using statistical key attributes and interpreted the results before-, during- and after transmission of these images.

We presented the encoding duration (Duration_e) and suspicion heuristic indices (H_g) of the phase before transmission for all $n = 39,195$ generated QR-Codes and interpret it subsequently. It shows that the generation of QR-Codes varies largely and performs best with ECC level L in nearly all cases except for version 34. In very low versions 1-3, the duration time is exponential, similarly as the suspicion heuristic H_g. During transmission, we obtain that taking screenshots takes on mean only 0.21 seconds, the decoding 0.11 seconds on mean. After transmission, we found that 78.25% of the transmitted QR-Code configurations were able to reach 100% accuracy in terms of reading the embedded information from the Stego Image and obtain the secret information encoded into the QR-Code.

CHAPTER 7

Conclusions and Future Work

In this chapter we want to conclude on the findings in Sec. 7.1, recap what is already known in Sec. 7.2, put the main findings into context in Sec. 7.3 and show limitations of this work in Sec. 7.4.

7.1 Conclusion

We presented an architecture and processes for operating a secure data infrastructure that supports secure data visiting. Data Owners can make their data accessible to visitors (experts, analysts) for specific tasks while maintaining full control of how their data is being used and preventing data leakage. The concept is based on providing access to the subsets of (potentially fingerprinted and/or anonymized) data required for a specific activity via isolated virtual machines that are monitored and accessible only via remote desktop access, but never directly. We implement this through well-defined processes for data provisioning, safe computing, -returns and -outputs as well as according contractual regulations.

7.1.1 Research Question 1

Kemmerer [28] describes a shared resource matrix methodology that lists all resources that can be referenced or modified by a subject. This methodology is valid since the assumptions (both subjects are processes of the system and share a single processor) apply to our method since we have a shared secure data infrastructure as system and a processor that is the Remote Desktop-VM. The methodology of Qian et al [42] allows evaluating the efficacy of a network behavior (covert) channel in terms of capacity-, robustness- and security analysis. The use of this methodology is valid in this thesis since we transform the problem of the behavior channel into our optical covert channel to be able to apply their fundamental equations.

We answer this research question to display the Analyst scenario and we enumerated all shared resources with the methodology of Kemmerer [28] to list their ability to be referenced or modified. Using the transitive closure (one attribute may have been modified by another component), we derive a potential covert channel candidate that can be used by the Analyst. Only considering the robustness lower bound ξ_{low}, candidates for an optical covert channel are configurations with versions up to ten, since ξ_{low} is close to 1. Taking only capacity into consideration, a set of possible candidates can not be obtained since the configurations scatter across the capacity/version graph. From only considering the PSNR, a set of possible candidates can be identified through selecting a configuration with high version $30 - 40$.

7.1.2 Research Question 2

Qian et al. use a system communication model that consists of a sender $\mathcal{A}$ and receiver $\mathcal{B}$ that is placed in an infrastructure with a detector $\mathcal{X}$ that listens for the presence of a covert channel $\mathcal{H}$. We use this methodology to construct a communication model and present the channels that can be used by the Analyst to establish an optical covert channel. The methodology is applicable since it models a covert channel in general and our optical covert channel is only a special case of the general case. Within the `OSSDIP` setting, we defined a communication model that uses the x-window-system after generating QR-Codes covertly on the Analyst-VM in its technical implementation. Through using Steganography and a red channel encoding, we can embed a QR-Code within a legitimate image and recover it through VNC when the server allows for standard encoding. This backdoor is not difficult to design, the methods are well-known and the libraries required to generate the QR-Code and embed it into the legitimate image are present on the target VM by default.

7.1.3 Research Question 3

All activity on the Analyst-VM is sent to the Monitoring-VM within our reference implementation. The Data Owner is allowed to inspect logging information for approved Analyst-VMs at any time. Due to the extensive logging within the infrastructure, the automated display of visually similar Stego Images most likely will raise suspicion if inspected by the Data Owner. The mechanism currently consists of the Data Owner viewing the activities of the Analyst live and access the log file after manual filtering of the System Administrator to only contain information about a single Analyst-VM in question. Since the reference implementation also monitors the VNC stream and records it, the QR-Codes will be visible in the stream and will be detected there.

7.1.4 Research Question 4

Our proposed QR-Code Optical Covert Channel uses the $(size = 320, ver = 38, ecc = \text{L})$ configuration which is capable of transmitting a 5 kilobyte raw sensitive data file in 10.73 seconds when displaying each Stego Image for 2 seconds from Analyst-VM to the Analyst

notebook with an 100% accuracy, which is equivalent to a bandwidth of $3,729.53$ bit/s, about 37 times larger than the significant threat bound postulated by Gligor [59]. We compared the bandwidth with a trained typist that is capable of writing a correct byte in 0.15 seconds postulated by Kinkead [29], resulting in a bandwidth of approx. 39.99 bit/s. By only looking at the encoding heuristic index H_e, QR-Code configurations that are generated faster with high capacity are rewarded, which results in configurations with mostly versions 29-37 at ECC Level L being the most suitable candidates For the capacity model of Qian et al [42], we calculate the covert channel capacity as $3,729.53$ bit/s for the selected configuration ($size = 320, ver = 38, ecc = \mathrm{L}$). Thus we found a significant threat for our reference implementation.

7.2 What is already known

Network covert channels are a ongoing field of research. When established, an Analyst can use one or more to covertly communicate inside network protocols and extract sensitive information from secured environments. Most covert channels require in-depth understanding of all involved communication partners and network protocols in order to fully comprehend and prevent them.

We presented related infrastructures and covert channels in Chapter 2. Approaches of Guri [20] showed, that QR-Codes are suitable to export a large amount of data from highly-protected systems, when visual access cannot be restricted. Their experiment showed, that an Analyst with direct line of sight to a compromised monitor can ex-filtrate invisible QR-Codes embedded in legitimate visual representations.

7.3 What this thesis adds

This thesis uses well-studied methods to determine the optimal optical covert channel an authenticated Analyst can use for our reference implementation. To the best of our knowledge, there exists little to no work on the practical evaluation of optical covert channels through QR-Codes where the display is not exposed to not only the Analyst. In our setting, the Analyst can operate the QR-Code Optical Covert Channel in a room where anyone is prohibited from entering e.g. a private room. We argue that this simplification still introduces a novel approach, since data visiting infrastructures and the ex-filtration of sensitive data thereof is an active research domain.

7.4 Limitations

A naïve approach would be to display the QR-Codes in full screen (e.g. through a diashow) to increase detection by the Analyst decoding script repeating the transmission sequence or using the discussed acknowledge keystroke re-transmission scheme. This, however, results in a very predictable behavior, since the image covers up the whole screen and the entropy [50] is always below 1 and easy to detect. Witha median transmission duration of 2

seconds, this approach will most likely be detected, since it may not match common screen entropy distributions. Since our approach makes use of the default GNOME document viewer *evince*, the PDF documents open at the default position. However, this behavior can be changed to display each window in the center by setting `org.gnome.mutter center-new-windows true`. It is assumed that this alteration does not improve of worsen the entropy and accuracy of the method. Since we are not able to set the exact position, size or zoom level using the document viewer or by changing the GNOME desktop settings, the placement of the QR-Code is left with the default settings of the desktop environment. Using our detection mechanisms with inspecting the logs, we can only examine past events, which leaves the Data Owner with limited capabilities to detect the theft instantly. In fugure work, we want to examine how to detect the exfiltration in real-time and develop a module for data leak prevention solutions.

7.5 Future Work

We give an overview of identified future work approaches related with our work presented. Our vision for future work is manifold. To detect our proposed QR-Code Optical Covert Channel within the secure data infrastructure, the security analyst has multiple methods: (1) using the Kolmogorov-Smirnov test between two empirical distribution functions with two samples [55] to compare similarity of a distribution with a known distribution. Peng et al [36] show the effectiveness of this approach to detect watermarked inter-packet delays for a covert timing channel; (2) using the Corrected Conditional Entropy (CCE) approach, Gianvecchio and Wang [17] have shown that their approach detects covert channels equally effective.

In future releases, we want to implement both of the approaches and evaluate their effectiveness. Additionally, in the future we consult with system administrators to propose improvements on the deployment process, receive insight in the secure processes and how to improve them, in terms of security and handling. We want to develop a mechanism for data leak prevention solutions that can detect this covert channel for real-time notification of the Data Owner that the Analyst may tries to ex-filtrate data. Contrary to previous approaches that protect intellectual property on a low level or use statistical methods, Piec and Rauber [39] use optical mechanisms to identify information leak within organizations, where an Analyst is able to view sensitive information using a screen. Their approach uses hidden watermarks that are placed on-top of interesting parts of the screen, embedding information that makes it possible to identify the source of the information leak, once decoded.

This has major advantages to organizations that want to protect their intellectual property, since it is decoupled from the content displayed and can be used on any underlying application. Since the watermark is placed on interesting regions that are determined using the FAST feature detection algorithm [47], their approach can be applied to rapid content changes due to the well-studied implementation of the FAST algorithm.

Due to the large amount of video stream capture, we currently evaluate if only the last

24 hours VNC stream should be available as video and use converters[1] to save only screenshots in intervals of seconds for longer time periods.

[1] "rfbproxy". [Online]. URL: https://lwn.net/Articles/452098/, accessed 2021-08-29

Supplementary Material

To provide the reader with the original benchmark data in the form of an accessible data set in the comma-separated values file format for $n = 39,195$ QR-Codes. The data set in this thesis is available online [60].

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] I. Barkow, T. Leopold, M. Raab, D. Schiller, K. Wenzig, H.-P. Blossfeld, and M. Rittberger, "RemoteNEPS: Data Dissemination in a Collaborative Workspace", *Zeitschrift für Erziehungswissenschaft*, vol. 14, no. 2, pp. 315–325, May 2011, ISSN: 1862-5215. DOI: 10.1007/s11618-011-0192-5.

[2] R. A. Bedruz, E. Sybingco, A. R. Quiros, A. C. Uy, R. R. Vicerra, and E. Dadios, "Fuzzy Logic Based Vehicular Plate Character Recognition System Using Image Segmentation and Scale-invariant Feature Transform", in *2016 IEEE Region 10 Conference (TENCON)*, 2016, pp. 676–681. DOI: 10.1109/TENCON.2016.7848088.

[3] M. Bicher, C. Rippinger, C. Urach, D. Brunmeir, U. Siebert, and N. Popper, "Evaluation of Contact-Tracing Policies Against the Spread of SARS-CoV-2 in Austria: An Agent-Based Simulation", *Medical Decision Making*, p. 0272989X211013306, May 2021. DOI: 10.1177/0272989X211013306.

[4] M. Bicher, C. Urach, and N. Popper, "GEPOC ABM: A Generic Agent-Based Population Model for Austria", in *Proceedings of the 2018 Winter Simulation Conference*, ser. WSC '18, Gothenburg, Sweden: IEEE Press, 2018, 2656–2667, ISBN: 978153866570. DOI: 10.1109/WSC.2018.8632170.

[5] J. K. Blitzstein and J. Hwang, "Introduction to Probability, 2nd Edition", in. Abingdon, Oxon, UK: Chapman and Hall/CRC Texts in Statistical Science, 2019, p. 405, ISBN: 9781138369917.

[6] T. C. Bressoud and F. B. Schneider, "Hypervisor-based Fault Tolerance", *ACM Transactions on Computer Systems*, vol. 14, no. 1, 80—107, Feb. 1996. DOI: 10.1145/225535.225538.

[7] C. J. Brooks, J. W. Stephens, D. E. Price, D. V. Ford, R. A. Lyons, S. L. Prior, and S. C. Bain, "Use of A Patient Linked Data Warehouse to Facilitate Diabetes Trial Recruitment from Primary Care", *Primary Care Diabetes*, vol. 3, no. 4, pp. 245–248, Nov. 2009. DOI: 10.1016/j.pcd.2009.06.004.

[8] S. Brophy, K. H. Jones, M. A. Rahman, S.-M. Zhou, A. John, M. D. Atkinson, N. Francis, R. A. Lyons, and F. Dunstan, "Incidence of Campylobacter and Salmonella Infections Following First Prescription for PPI: a Cohort Study Using Routine Data", *Official Journal of the American College of Gastroenterology*, vol. 108, no. 7, 2013. DOI: 10.1038/ajg.2013.30.

[9] B. Carrara and C. Adams, "On Acoustic Covert Channels Between Air-Gapped Systems", in *Foundations and Practice of Security*, F. Cuppens, J. Garcia-Alfaro, N. Zincir Heywood, and P. W. L. Fong, Eds., Cham: Springer International Publishing, 2015, pp. 3–16. DOI: 10.1007/978-3-319-17040-4_1.

[10] T. Desai, F. Ritchie, and R. Welpton, *Five Safes: Designing Data Access for Research*, https://uwe-repository.worktribe.com/output/914745/five-safes-designing-data-access-for-research, accessed 2021-08-04, 2016.

[11] L. Dinges, A. Al-Hamadi, M. Elzobi, and A. Nürnberger, "Automatic Recognition of Common Arabic Handwritten Words Based on OCR and N-GRAMS", in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3625–3629. DOI: 10.1109/ICIP.2017.8296958.

[12] N. Elhage, *Virtunoid: Breaking out of KVM (Kernel Virtual Machine)*, Aug. 2011. DOI: 10.5446/40606.

[13] D. P. Faith, "Asymmetric Binary Similarity Measures", *Oecologia*, vol. 57, no. 3, pp. 287–290, 1983.

[14] D. Ferraiolo and R. Kuhn, "Role-Based Access Controls", in *Proceedings of the 15th National Computer Security Conference*, vol. 1, Oct. 1992, pp. 554–563.

[15] K. Furumoto and M. Morii, "Recognition Accuracy Improvement of Obfuscated QR-Code by Using Reliability Information and GMD Decoding", *Journal of Information Processing*, vol. 26, pp. 350–357, 2018. DOI: 10.2197/ipsjjip.26.350.

[16] J. Z. Gao, L. Prakash, and R. Jagatesan, "Understanding 2D-BarCode Technology and Applications in M-Commerce - Design and Implementation of A 2D Barcode Processing Solution", in *31st Annual International Computer Software and Applications Conference (COMPSAC 2007)*, vol. 2, 2007, pp. 49–56. DOI: 10.1109/COMPSAC.2007.229.

[17] S. Gianvecchio and H. Wang, "An Entropy-Based Approach to Detecting Covert Timing Channels", *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 6, pp. 785–797, 2011. DOI: 10.1109/TDSC.2010.46.

[18] R. P. Goldberg, "Survey of Virtual Vachine Research", *Computer*, vol. 7, no. 6, pp. 34–45, 1974. DOI: 10.1109/MC.1974.6323581.

[19] A. J. Goldsmith and P. P. Varaiya, "Capacity, Mutual Information, and Coding for Finite-state Markov Channels", *IEEE Transactions on Information Theory*, vol. 42, no. 3, pp. 868–886, 1996. DOI: 10.1109/18.490551.

[20] M. Guri, "Optical Air-Gap Exfiltration Attack via Invisible Images", *Journal of Information Security and Applications*, vol. 46, pp. 222–230, 2019, ISSN: 2214-2126. DOI: 10.1016/j.jisa.2019.02.004.

[21] M. Guri, B. Zadov, and Y. Elovici, "LED-it-GO: Leaking (A Lot of) Data from Air-Gapped Computers via the (Small) Hard Drive LED", in *Proceedings of the International Conference on Detection of Intrusions, Malware and Vulnerability Assessment*, Cham: Springer International Publishing, 2017, pp. 161–184. DOI: 10.1007/978-3-319-60876-1_8.

[22] V. Hajduk, M. Broda, O. Kováč, and D. Levický, "Image Steganography with using QR Code and Cryptography", in *2016 26th International Conference Radioelektronika*, Apr. 2016, pp. 350–353. DOI: 10.1109/RADIOELEK.2016.7477370.

[23] T. Hassan and H. A. Khan, "Handwritten Bangla Numeral Recognition Using Local Binary Pattern", in *2015 International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, 2015, pp. 1–4. DOI: 10.1109/ICEEICT.2015.7307371.

[24] T. Hubbard, G. Reilly, S. Varma, and D. Seymour, *Trusted Research Environments Green Paper*, version 2, Jul. 2020. DOI: 10.5281/zenodo.4594704.

[25] N. Johnson, Z. Duric, and S. Jajodia, "Information Hiding: Steganography and Watermarking-Attacks and Countermeasures", in, 1st ed. Springer US, 2001, vol. 1, ch. 2, pp. 15–45. DOI: 10.1007/978-1-4615-4375-6.

[26] N. F. Johnson and S. Jajodia, "Exploring Steganography: Seeing the Unseen", *Computer*, vol. 31, no. 2, pp. 26–34, 1998. DOI: 10.1109/MC.1998.4655281.

[27] K. H. Jones, D. V. Ford, C. Jones, R. Dsilva, S. Thompson, C. J. Brooks, M. L. Heaven, D. S. Thayer, C. L. McNerney, and R. A. Lyons, "A Case Study of the Secure Anonymous Information Linkage (SAIL) Gateway: A Privacy-protecting Remote Access System for Health-related Research and Evaluation", *Journal of Biomedical Informatics*, vol. 50, pp. 196–204, 2014. DOI: 10.1016/j.jbi.2014.01.003.

[28] R. A. Kemmerer, "Shared Resource Matrix Methodology: An Approach to Identifying Storage and Timing Channels", *ACM Transactions on Computer Systems*, vol. 1, pp. 159–173, 3 Aug. 1983. DOI: 10.1145/357369.357374.

[29] R. Kinkead, "Typing Speed, Keying Rates, and Optimal Keyboard Layouts", *Proceedings of the Human Factors Society Annual Meeting*, vol. 19, no. 2, pp. 159–161, 1975. DOI: 10.1177/154193127501900203.

[30] B. W. Lampson, "A Note on the Confinement Problem", *Commun. ACM*, vol. 16, no. 10, 613–615, Oct. 1973. DOI: 10.1145/362375.362389.

[31] C. Liyanage, T. Nadungodage, and R. Weerasinghe, "Developing a Commercial Grade Tamil OCR for Recognizing Font and Size Independent Text", in *2015 Fifteenth International Conference on Advances in ICT for Emerging Regions*, 2015, pp. 130–134. DOI: 10.1109/ICTER.2015.7377678.

[32] J. Loughry and D. A. Umphress, "Information Leakage from Optical Emanations", *ACM Transactions on Privacy and Security*, vol. 5, no. 3, 262—289, Aug. 2002, ISSN: 1094–9224. DOI: 10.1145/545186.545189.

[33] J. McGregor, C. Brooks, P. Chalasani, J. Chukwuma, H. Hutchings, R. A. Lyons, and K. Lloyd, "The Health Informatics Trial Enhancement Project (HITE): Using Routinely Collected Primary Care Data to Identify Potential Participants for a Depression Trial", *Trials*, vol. 11, no. 1, p. 39, Apr. 2010. DOI: 10.1186/1745-6215-11-39.

[34] R. A. Meyer and L. H. Seawright, "A Virtual Machine Time-sharing System", *IBM Systems Journal*, vol. 9, no. 3, pp. 199–218, 1970. DOI: 10.1147/sj.93.0199.

[35] M. Mishra, A. Das, P. Kulkarni, and A. Sahoo, "Dynamic Resource Management Using Virtual Machine Migrations", *IEEE Communications Magazine*, vol. 50, no. 9, pp. 34–40, 2012. DOI: 10.1109/MCOM.2012.6295709.

[36] Pai Peng, Peng Ning, and D. S. Reeves, "On the Secrecy of Timing-based Active Watermarking Trace-back Techniques", in *Proceedings of the 2006 IEEE Symposium on Security and Privacy (SP'06)*, 2006, 15 pp.–349. DOI: 10.1109/SP.2006.28.

[37] E. Perez-Alba, L. Nuzzolo-Shihadeh, J. E. Espinosa-Mora, and A. Camacho-Ortiz, "Use of Self-Administered Surveys Through QR Code and Same Center Telemedicine in a Walk-In Clinic in the Era of COVID-19", *Journal of the American Medical Informatics Association*, vol. 27, no. 6, pp. 985–986, May 2020. DOI: 10.1093/jamia/ocaa054.

[38] T. Peterseil, "DEXHELPP", Aug. 2021, Personal Conversation.

[39] M. Piec and A. Rauber, "Real-time Screen Watermarking Using Overlaying Layer", in *Proceedings of the 9th International Conference on Availability, Reliability and Security*, 2014, pp. 561–570. DOI: 10.1109/ARES.2014.83.

[40] N. Popper, F. Endel, R. Mayer, M. Bicher, and B. Glock, "Planning Future Health: Developing Big Data and System Modelling Pipelines for Health System Research", *Simulation Notes Europe*, vol. 27, pp. 203–208, 4 2017. DOI: 10.11128/sne.27.tn.10396.

[41] N. Provos, M. Friedl, and P. Honeyman, "Preventing Privilege Escalation", in *Proceedings of the 12th USENIX Security Symposium*, Aug. 2003, pp. 231–241. DOI: 10.5555/1251353.1251369.

[42] Y. Qian, T. Sun, J. Li, C. Fan, and H. Song, "Design and Analysis of the Covert Channel Implemented by Behaviors of Network Users", *Security and Communication Networks*, vol. 9, no. 14, pp. 2359–2370, 2016. DOI: 10.1002/sec.1503.

[43] R. Rahim, H. Nurdiyanto, R. Hidayat, A. S. Ahmar, D. Siregar, A. P. U. Siahaan, I. Faisal, S. Rahman, D. Suita, A. Zamsuri, D. Abdullah, D. Napitupulu, M. I. Setiawan, and S Sriadhi, "Combination Base64 Algorithm and EOF Technique for Steganography", *Journal of Physics: Conference Series*, vol. 1007, p. 012 003, Apr. 2018. DOI: 10.1088/1742-6596/1007/1/012003.

[44] A. Rauber, A. Asmi, D. van Uytvanck, and S. Proell, *Data Citation of Evolving Data: Recommendations of the Working Group on Data Citation (WGDC)*, Oct. 2015. DOI: 10.15497/RDA00016.

[45] A. Rauber, A. Asmi, D. van Uytvanck, and S. Pröll, "Identification of Reproducible Subsets for Data Citation, Sharing and Re-Use", *Bulletin of the IEEE Technical Committe on Digital Libraries (TCDL)*, vol. 12, no. 1, May 2016. DOI: 10.5281/zenodo.4048304.

[46] S. E. Rodgers, R. A. Lyons, R. Dsilva, K. H. Jones, C. J. Brooks, D. V. Ford, G. John, and J.-P. Verplancke, "Residential Anonymous Linking Fields (RALFs): A Novel Information Infrastructure to Study the Interaction between the Environment and Individual's Health", *Journal of Public Health*, vol. 31, no. 4, pp. 582–588, May 2009. DOI: 10.1093/pubmed/fdp041.

[47] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking", in *Proceedings of the Tenth IEEE International Conference on Computer Vision*, vol. 2, Nov. 2005, pp. 1508 –1515. DOI: 10.1109/ICCV.2005.104.

[48] H. A. Rothan and S. N. Byrareddy, "The Epidemiology and Pathogenesis of Coronavirus Disease (COVID-19) Outbreak", *Journal of Autoimmunity*, vol. 109, p. 102 433, 2020, ISSN: 0896-8411. DOI: 10.1016/j.jaut.2020.102433.

[49] M. J. Scheepers, "Virtualization and Containerization of Application Infrastructure: A Comparison", in *21st Twente Student Conference on IT*, 2014, pp. 1–7.

[50] C. E. Shannon, "A mathematical theory of communication", *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[51] C. Shannon, "Communication in the Presence of Noise", *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949. DOI: 10.1109/JRPROC.1949.232969.

[52] G. J. Simmons, "The Prisoners' Problem and the Subliminal Channel", in *Advances in Cryptology: Proceedings of Crypto 83*, D. Chaum, Ed., Boston, MA: Springer US, 1984, pp. 51–67. DOI: 10.1007/978-1-4684-4730-9_5.

[53] G. J. Simmons, "Message Authentication Without Secrecy", in *Secure Communications and Asymmetric Cryptosystems*. Boulder: Westview Press, 1982, pp. 105–139.

[54] J. Skopek, T. Koberg, and H.-P. Blossfeld, "RemoteNEPS – An Innovative Research Environment", in *Methodological Issues of Longitudinal Surveys: The Example of the National Educational Panel Study*, H.-P. Blossfeld, J. von Maurice, M. Bayer, and J. Skopek, Eds., Wiesbaden: Springer Fachmedien Wiesbaden, 2016, pp. 611–626. DOI: 10.1007/978-3-658-11994-2_34.

[55] N. V. Smirnov, "On The Estimation of the Discrepancy between Empirical Curves of Distribution for Two Independent Samples", *Moscow University Mathematics Bulletin*, vol. 2, no. 2, pp. 3–14, 1939.

[56] C. Song, T. Ristenpart, and V. Shmatikov, "Machine Learning Models That Remember Too Much", in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17, New York, NY, USA: Association for Computing Machinery, 2017, 587–601, ISBN: 9781450349468. DOI: 10.1145/3133956.3134077.

[57] S. Suzuki and K. Abe, "Topological Structural Analysis of Digitized Binary Images by Border Following", *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985. DOI: 10.1016/0734-189X(85)90016-7.

[58] S. Tiwari, "An Introduction to QR Code Technology", in *2016 International Conference on Information Technology (ICIT)*, 2016, pp. 39–44. DOI: 10.1109/ICIT.2016.021.

[59] G. Virgil, *A Guide to Understanding Covert Channel Analysis of Trusted Systems*, Nov. 1993. DOI: 10.21236/ada477386.

[60] M. Weise, "QR-Code Optical Covert Channel Benchmark", version 2, Aug. 2021, [Dataset]. DOI: 10.5281/zenodo.5502675.

[61] M. Weise and A. Rauber, "A Data-Visiting Infrastructure for Providing Access to Preserved Databases that Cannot be Shared or Made Publicly Accessible", in *Proceedings of the 17th International Conference on Digital Preservation*, Beijing, China: iPRES, 2021.

[62] M. Weise and A. Rauber, "Open Source Secure Data Infrastructure and Processes Supporting Data Visiting", *Zenodo*, 2021. DOI: 10.5281/zenodo.4632903.

[63] R. Wilf-Miron, V. Myers, and M. Saban, "Incentivizing Vaccination Uptake: The "Green Pass" Proposal in Israel", *JAMA*, vol. 325, no. 15, pp. 1503–1504, Apr. 2021, ISSN: 0098-7484. DOI: 10.1001/jama.2021.4300.

[64] R. Wojtczuk and C. Kallenberg, "Attacking UEFI Boot Script", in *31st Chaos Communication Congress*, Hamburg, Germany, Jan. 2015.

[65] W.-C. Wu, Z.-W. Lin, and W.-T. Wong, "Application of QR-Code Steganography Using Data Embedding Technique", in *Information Technology Convergence*, J. J. J. H. Park, L. Barolli, F. Xhafa, and H.-Y. Jeong, Eds., Dordrecht: Springer Netherlands, 2013, pp. 597–605.

[66] Z. Wu, Z. Xu, and H. Wang, "Whispers in the Hyper-Space: High-Speed Covert Channel Attacks in the Cloud", in *Proceedings of the 21st USENIX Conference on Security Symposium*, ser. Security'12, Bellevue, WA: USENIX Association, Aug. 2012, p. 9. DOI: 10.5555/2362793.2362802.

[67] L. Yujian and L. Bo, "A Normalized Levenshtein Distance Metric", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1091–1095, 2007. DOI: 10.1109/TPAMI.2007.1078.

[68] G Zauner, N. Popper, and F Breitenecker, "State of the Art Research in Austria: DEXHELPP – Decision Support for Health Policy and Planning: Methods, Models and Technologies Based On Existing Health Care Data", *Security and Communication Networks*, vol. 17, no. 4, Nov. 2014. DOI: 10.1016/j.jval.2014.08.1222.

[69] M. Zhu, B. Tu, W. Wei, and D. Meng, "HA-VMSI: A Lightweight Virtual Machine Isolation Approach with Commodity Hardware for ARM", in *Proceedings of the 13th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, ser. VEE '17, Xi'an, China: Association for Computing Machinery, 2017, 242–256. DOI: 10.1145/3050748.3050767.