Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

ZHAW - MAS Informatik

MAS Thesis

Controlled communication of seized mobile devices in the IT Forensics Unit of Zurich Metropolitan Police

A controlled network environment that completely blocks the data communication of seized mobile phones and only allows essential communication through whitelisting.

Alexander Wüst
Kronenwis 19
8864 Reichenburg

Abgabe 14. May 2025
Betreuer:in Peter Heinrich

Zürcher Fachhochschule

Abstract

Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text. Dies ist ein Platzhalter-Text.

Table of Contents

1. Introduction

1.1. Preface

Electronic evidence plays a central role in today's criminal prosecution. Almost no criminal proceedings can do without the analysis of digital data - be it data from mobile phones, computers, IoT devices, cloud data and many other data sources. Digital forensics plays a crucial role in this: IT forensic experts prepare the seized devices according to forensic standards to ensure that the data collected can be used in court.

It's important does the collected devices are completely isolated from data networks after seizing them to maintain data integrity. If this is not possible due to security settings, the Zurich Metropolitan Police use a Faraday room[1]. This prevents data communication from the mobile phone and data on the device from being changed. As soon as a mobile phone is reconnected to the internet it will updating apps and synchronising data, such as cloud services, messages and other application data and this of course will trigger write processes on the data storage. Also, there is the risk of a remote deletion of the device which needs to be avoided.

A fundamental principle of forensic work is the reproducibility of the results. Once write operations occur on the data storage, this reproducibility can no longer be guaranteed. So, it would be clear, establishing a network connection or even simply powering on a device seems not to be an option and needs to be avoided.

However, best practices at the Metropolitan Police Zurich have shown that powering on a device is necessary to verify whether the data acquired by the forensic hardware and software has been processed correctly. A defined protocol is followed to check for any apps or entries that may not have been parsed correctly – In fact there are often problems during parsing in practice. Especially apps just known in Switzerland like “Twint” are not parsed automatically.

Without powering on the device, still in Airplane mode, it is not possible to detect parsing errors. For example, when no WhatsApp messages are shown at all or to identify if any media files such as photos are missing. In recent years, when physical access to a device was possible, like with a known or brute-forced[2] passcode, it has proven useful to use this access to validate the acquisition and ensure no data was lost or misinterpreted during the process. The described procedure, although it also causes write operations, but it is considered as essential to ensure a good and complete data acquisition.

It is more and more common that not everything that is accessible on the device is saved on the device. Common examples are images and videos that are stored directly on provider servers (cloud). When manual reviewing chat conversations trough a police investigator, it can happen that the content of chats clearly goes in one direction, but the images and videos taken are missing because they are no longer or have never been stored locally on the device. For example, an image that could be identified as an offence or a prohibited media file containing violence or even child pornography may be missing from the device and because that it will not allow any clear conclusions. Or the communication seems to be clear, but the pictures are harmless? So, it is important to give during a search the best possible

[1] Faraday room or Faraday cage, which prevents communication from or into the room. A great and quite simple setup was made by the University of Gottingen [1].

[2] Brute force refers to methods that systematically attempt all possible passcode combinations.

picture. But how to gain this extra information without getting the device online? The best way is to access the data directly with a cloud acquisition method. For this method the service needs to be supported, and a valid token needs to be extracted from the device. Sometimes even that method is not possible due to different reasons. Then it will normally give a consultation between the public prosecutor, the police officer who is in charge for the case and the forensic examiner. One of the biggest problems is the possibility of a remote deletion when taking the device online. As the acquiring trough forensic hard- and software took already place it will still be possible to have the original copy of the state of the device when it was seized.

There is no easy solution that makes it possible to take a device online in a controlled environment without risking the remote deletion of data and at the same time only allowing the necessary connections to the Internet. This MAS thesis aims to close this gap.

1.2. Goal of the work

The primary objective of this thesis is to implement a wireless network with an integrated firewall, which by default blocks all internet access for connected devices. A central focus of the project is the development of a custom web interface that interacts with an firewall solution via its API. Through this interface, the forensic examiner will be able to monitor and configure firewall settings, define individual access rules for connected devices.

The solution is designed to improve usability and control by allowing the forensic examiner to make informed decisions about which connections should be allowed and which should remain blocked. All connections will be logged, enabling transparency and accountability. This is especially important in a forensic context.

To meet these goals, the project includes the development of an application with the following core capabilities:

- Centralized management of device-based firewall rules
- Monitoring of network activity
- Logging of all connection attempts and allowed traffic
- Easy addition and removal of rules through a user-friendly interface

Therefore suitable hardware will be evaluated and bought to ensure compatibility with the firewall solution and to meet performance and environmental requirements (e.g., rugged form factor, multiple network interfaces).

Reference to use cases ???…

1.3. Scope Limiations

This master's thesis explicitly excludes the following topics from its scope:

- Implementation or evaluation of two-factor authentication methods using mobile networks (e.g., MobileID, SMS-based authentication)
- Analysis of write operations on mobile devices that may occur as a result of simply powering them on, even if considered best practice in forensic handling
- Integration of Deep Packet Inspection (DPI) or content-level traffic analysis within the firewall
- Examination of legal considerations or compliance issues, such as data protection laws, admissibility of evidence, or regulatory frameworks

The focus of this work remains on the technical implementation of a device-aware firewall management system for controlled network access and forensic transparency.

2. Background

2.1. Evolution of Firewalls

As computer networks began to connect to each other, the need to protect one network from the other and avoid external threats became increasingly important. The name “Firewall” came from physical barriers used in the architecture and history: Walls that protected cities like the Great Wall of China or structural firewalls that prevented a fire in the kitchen to spread out on other parts of the building. [2]. Similar, in networking, a firewall serves as a barrier to keep not wanted traffic outside and to allow other traffic to go through. Even if the “external side” is burning the firewall is intended to keep the internal network safe and unaffected.

The development of firewalls has progressed through several generations. First-generation firewalls in the 1990s used basic packet filtering to allow or block traffic based on IP addresses and ports.

In the Early 2000s the second generation, stateful inspection firewalls emerged, introducing the ability to track the state of connections and inspect traffic in context, offering enhanced control. Then application-layer firewalls enabled content-aware filtering and protocol-specific inspection, further increasing security.

Around 2008 the third Generation or also called Next-Generation Firewalls (NGFW) integrated traditional firewall functions with deep packet inspection (DPI), intrusion prevention systems (IPS), and application awareness.

The fourth firewall generation was launched around 2020. These firewalls use machine learning to detect zero-day threats in real time, going beyond traditional signature-based methods. Key features include zero-delay signature updates, automated security policy recommendations, and IoT device visibility based on behavioral analysis. It is continously learning from the network traffic. It aims to reduce the manual intervention. [2], [3]

2.2. Overview of current firewall solutions

This section provides an overview of actual avaiable firewall solutions focused on Open Source and Enterprise solutions.

2.2.1. System Level Firewalls

Within the Linux systems there are essential tools to control network traffic and indivudal hosts. The most common solutions are iptables, nftables, ufw and firewalld. They are varying from complexity, flexibility and how user friendly they are.

Iptables – Is a tradtional packet-filtering framework which is included directly in the Linux kernel.

Nftables – Is like a new version of Iptables with many advantages like rule changes at once. All rules like Ipv4 and Ipv6 are possible in the same code base. [4]

ufw

2.2.2. Open Source Firewalls

An open-source firewall is a firewall solution whose source code is publicly available and freely accessible. The source code can be reviewed and modified. These firewalls are typically community-driven and cost-effective. One of a big advantage is also the opportunity to customize the product. Therefore many projects offer commercial services like professional support and additional enterprise features. This helps the organisation behind to maintain the software and to fund the needed infrastructure. [5]

pfSense - Is a flexible, open-source firewall and router platform that offers NAT, packet filtering, and next-generation firewall features. It supports multiple interfaces, scales well, and provides a command-line interface for advanced configuration.

OPNsense - Is a user-friendly, web-managed next-generation firewall with built-in intrusion detection (IDS), web filtering, and VPN support. It combines strong security features with an intuitive interface, making it suitable for diverse network environments.

VyOS - Is a community-driven, fully open-source firewall that aims for high availability and uptime. It includes stateful inspection, NAT, and routing features, and is often used in hardware appliances for continuous performance.

ClearOS - Is a simple, stateful firewall solution aimed at users with basic network protection needs. It is easy to manage and configure, though it lacks advanced features like NAT and packet filtering.

2.2.3. Enterprise Firewalls

Enterprise firewalls are typically closed-source and unlike open-source alternatives, not freely accessible or modifiable. These solutions often come with a wide range of additional services, including professional support, subscription-based threat intelligence, and service-level agreements (SLAs), making them particularly attractive for organizations that require guaranteed uptime, vendor accountability, and integrated security management. Well known brands are Palo Alto, Fortinet, Cisco, Check Point, Sophos.

2.2.4. Requirements for the firewall solution

Early in the project phase, it was essential to determine the firewall platform for the project. The Primary goal was to build an extended interface for our specific need than developing a complete firewall solution from scratch. The selected system needed to provide a robust and flexible base. For evaluation purpose a minimal set of functional and technical requirements was defined:

Open-Source Licensing The firewall must be distributed under an open-source license. This ensures the solution is free of licensing costs and avoids vendor lock-in. Furthermore, open-source access enables future customization, which may become necessary in advanced stages of the project.

Self-Hosting on Local Infrastructure The solution must be fully deployable on the organization’s own infrastructure without relying on any external cloud components or third-party services. This is essential for maintaining full control over the processed data, meeting forensic standards, and following with the information security and data protection policies (ISDS) of the City of Zurich.

API Support A modern well documented API. The project aims to integrate firewall control into a custom web interface, without the need to interact directly with the firewall's native user interface. API support is also essential for any automation workflows.

Device-Aware Rule Management The firewall must support rules that can be defined per device, based on identifiers such as IP or MAC addresses. This capability is necessary for implementing granular access policies. For example, permitting one device to access a specific IP while restricting another.

Active Development The firewall must have an ongoing development with regular updates and security patches.

Feature-Rich Environment The solution should include a wide range of network security features out of the box. Minimize the need for third-party tools and to have the capability for later processing steps.

2.2.5. Compare firewall solutions based on requirements

As the enterprise firewall solutions (e.g., Palo Alto, Fortinet, Cisco, Check Point, and Sophos) are proprietary and do not meet the open-source licensing requirement, they were excluded from the comparison table below. These products are therefore considered out of scope for this project, which is focused solely on open-source, self-hosted solutions.

Firewall	Open Source	Self-Hosting	API Support	Device-Aware Rules	Active Development	Feature-Rich
iptables[6]	Yes	Yes	No	Limited	Yes	Moderate
nftables[7]	Yes	Yes	No	Limited	Yes	Yes
pfSense[8]	Yes	Yes	Limited	Yes	Yes	Yes
OPNSense[9]	Yes	Yes	Yes	Yes	Yes	Yes
VyOS[10]	Yes	Yes	Limited	Limited	Yes	Yes
ClearOS[11]	Yes	Yes	Limited	No	No	Decreasing with time as no active development is done

Table 1: Comparison of different firewall solutions

After evaluating the available open-source firewall projects and consulting their official documentation and community resources, **OPNsense** was selected as the source firewall for this project. OPNsense has an actively maintained platform with a huge user and developer community.

A great advantage of OPNsense is the officially supported and documented API, which already covers most when not all needed functions for the later project. The project offers regular updates, an open roadmap and detailed release notes.

Although the project would be possible on a base like nftables which is integrated directly into the Linux kernel and with the needed implementation time it would be possible as well. However, the expected additional development time was looked at as too extensive, this is why a ready to use solution was selected.

2.3. Hardware evaluation

Hardware selection was not a primary focus at the outset of the project. After the decision was made to use OPNsense as the firewall platform, further research was made using online sources, including general searches (e.g., Google) to identify suitable hardware configurations for a cost-effective setup.

The official OPNsense hardware appliance offerings [12] were reviewed but ultimately looked as out of scope due to their cost. These systems are primarily targeted at enterprise customers, with prices often exceeding the budget. The project aimed to stay within a budget of just a few hundred Swiss francs, ideally utilizing small-form-factor, low-cost hardware such as a Raspberry Pi 5.

While the Raspberry Pi 5 initially appeared to be a promising candidate due to its affordability, it was ultimately excluded. Although it meets the official recommended hardware requirements for OPNsense [13], including:

Processor: Minimum 1.5 GHz multi-core CPU

RAM: 8 GB

It failed to meet the additional project-specific requirements, which were essential for practical deployment in a forensic lab environment:

Industrial enclosure: The device must be housed in a durable case suitable for continuous operation in a professional setting.

Networking capabilities: The device must support at least one WAN port and two or more LAN ports (e.g., one for Wi-Fi and one or more for Ethernet-connected devices).

However rugged enclosures for Raspberry Pi exist, they are still part of a DIY solution and often require additional adapters or USB-to-Ethernet dongles, which compromise reliability and performance. When evaluating networking options specifically, it became clear that no off-the-shelf Raspberry Pi configuration could fulfil the requirements.

In the OPNsense forum [14] many community members shared links to low-cost firewall hardware available on platforms such as AliExpress. However, this option was not considered further due to concerns regarding warranty coverage, lack of technical support, and the potential for long delivery times - often several weeks. These factors would have introduced unnecessary delays and risk to the project timeline.

Finally, the decision was made to purchase hardware from Protectli, a company with a location in Rossdorf, Germany. Protectli specializes in open-source firewall appliances and is well-regarded for its compatibility with OPNsense. After evaluating several of the company's models, the Protectli V1410 was selected, as it fulfilled all functional requirements for the project, including port availability, compact form factor, hardware durability, and full compatibility with the OPNsense software platform.

Selected Device: **Protectli V1410**

CPU: Intel® N5105 Quad-Core, 2.0 GHz (Turbo up to 2.9 GHz)

Networking: 4 × Intel® I226-V 2.5 GbE RJ-45 Ethernet ports

Memory: 8 GB LPDDR4 (on-board)

Storage: 32 GB onboard eMMC and 250 GB Kingston NVMe (NV2-250G)

Expansion: M.2 slots for optional Wi-Fi or LTE modules

Power Supply: 12 V with screw-in connector (included)

Other: Fanless, silent operation, coreboot-supported, compact form factor

Price: €284.55 (excluding VAT, as of February 17, 2025)

This hardware provides sufficient performance and operational flexibility for use in a forensic laboratory environment. It fully meets the hardware requirements for OPNsense and includes additional storage capacity for extended logging or future use cases. The fanless design and industrial-grade build further support long-term maintainability and stability under continuous operation.

2.4. Remote wipe

2.4.1. Apple iOS

2.4.2. Google Android

3. Implementation

4. Analysis / Results

5. Discussion / Conclusion

References

[1] M. Mohler, ‘Der Faradaysche Käfig’. Accessed: Jan. 13, 2025. [Online]. Available: https://lp.uni-goettingen.de/get/text/833

[2] K. Ingham and S. Forrest, ‘Network Firewalls’, University of New Mexico, 2002. [Online]. Available: http://iar.cs.unm.edu/~forrest/publications/firewalls-05.pdf

[3] Palo Alto Networks, ‘The History of Firewalls | Who Invented the Firewall?’ Accessed: Aug. 05, 2025. [Online]. Available: https://www.paloaltonetworks.com/cyberpedia/history-of-firewalls

[4] A. Dubey, ‘Comprehensive Guide to Linux Firewalls: iptables, nftables, ufw, and firewalld’, Medium. Accessed: Aug. 05, 2025. [Online]. Available: https://medium.com/@amandubey_6607/comprehensive-guide-to-linux-firewalls-iptables-nftables-ufw-and-firewalld-9e86e0a49979

[5] C. Dilmegani, ‘Top 7+ Open Source Firewall Options in 2025: Features & Types’, AIMultiple Research. Accessed: Jun. 02, 2025. [Online]. Available: https://research.aimultiple.com/open-source-firewall/

[6] ‘iptables(8) - Linux manual page’. Accessed: May 14, 2025. [Online]. Available: https://man7.org/linux/man-pages/man8/iptables.8.html

[7] ‘nftables wiki’. Accessed: May 14, 2025. [Online]. Available: https://wiki.nftables.org/wiki-nftables/index.php/Main_Page

[8] ‘pfSense® - World’s Most Trusted Open Source Firewall’. Accessed: May 14, 2025. [Online]. Available: https://www.pfsense.org/

[9] OPNSense, ‘OPNsense Documentation’, OPNsense. Accessed: Feb. 12, 2025. [Online]. Available: https://docs.opnsense.org

[10]‘VyOS – Open source router and firewall platform’, VyOS. Accessed: May 14, 2025. [Online]. Available: https://vyos.io/

[11]‘ClearOS’. Accessed: May 14, 2025. [Online]. Available: https://clearos.com/

[12]‘Hardware – OPNsense® Shop’. Accessed: May 14, 2025. [Online]. Available: https://shop.opnsense.com/product-categorie/hardware-appliances/

[13]‘Hardware sizing & setup — OPNsense documentation’. Accessed: May 14, 2025. [Online]. Available: https://docs.opnsense.org/manual/hardware.html

[14]‘Hardware and Performance’, OPNsense Forum. Accessed: May 14, 2025. [Online]. Available: https://forum.opnsense.org/index.php?board=21.0

Appendix

Hier sind die in der Arbeit referenzierten Anhänge aufzuführen.

Declaration of Originality

Bitte Wortlaut aus «Merkblatt Erstellung Abschlussarbeit in CAS, DAS und MAS» übernehmen.