University *POLITEHNICA* of Bucharest
Faculty of Automatic Control and Computers, Computer Science and Engineering Department

PhD Thesis Summary

in Computer Science, Information Technology and System Engineering

Privacy Enhancements for Scalable Storage Systems in Public Cloud Environments

Îmbunătățirea confidențialității sistemelor de stocare scalabile în medii Cloud publice

presented by

Drd.ing. Gabriel Apostol

supervised by

Prof.dr.ing. Florin POP

2023
Bucharest, Romania

Contents

Abstract

Cloud data storage services have in general numerous research challenges, primarily attributed to the lack of physical control over the infrastructure. These challenges have a substantial impact on the security and performance of Cloud systems. Adding to these challenges security constraints such as data confidentiality, integrity, and availability leads to complex and valuable research questions that have a significant impact in the field of Information security and Scalable Cloud-based storage systems. The protection of personal data has become a critical concern in contemporary society. With the growing popularity of Cloud-based platforms, individuals are entrusting a significant amount of information to third-party entities. Therefore, it is crucial to prioritize the security of this information, as it may contain sensitive data. Addressing the research challenge for security enhancements of Scalable Storage Systems in Public Cloud Environments requires a holistic approach, considering scalability, data privacy, multi-tenancy, threat detection, compliance, and collaboration.

The main objective of this thesis is to research, design, implement, and evaluate a multi-layer security enhancement framework for Scalable Storage Systems in Public Cloud Environments that include: (i) a secure transport layer mechanism, the uses both cryptographic mechanisms and complex communication protocols that leverage data from mobile devices, (ii) a data-at-rest solution that enhance the confidentiality of data in public Clouds using a novel steganographic approach, and (iii) a data in use data security enhancement mechanism based on the cutting-edge and secure overlay that implements the SOCKS5 protocol.

1 | Context and Goals

The collection of personal data has become a widespread business practice in today's online world. The major players in this field are Internet Service Providers (ISPs), which are private companies that rent or own the infrastructure and have access to all information regarding the nature of online activities performed by their customers. Even if an individual starts an encrypted browsing session, he still might be the subject of pattern-based profiling. An adversary could exploit the fact that a site is partially encrypted, in order to get more insight into the interests of the average user. An alternate profiling method is the analysis of non-encrypted DNS queries used by every internet-connected application in order to communicate over public networks. These queries might leak potentially sensitive data such as health data, political interests, shopping activity, and even information regarding installed software. Even if the employment of network overlays for privacy enhancement poses some ethical issues, they are certainly centered around the principles of freedom. Therefore, their constant improvement [10] should be considered a priority, because confidentiality is a fundamental human right.

1.1 Problem Statement and Objectives

The scope of this section is to investigate various research questions pertaining to improving the privacy and security of data.

RQ 1: What are the security and privacy risks associated with storing personal data on public cloud storage services, and how can consumers minimize them? We analyzed the most popular free cloud storage service providers and evaluated their methods for transferring, storing, and processing user data.

RQ 2: Considering that numerous smartphones enhance their capabilities by utilizing cloud services, how can the diverse range of hardware features they possess enhance security and build trust? we analyzed how smartphones can enhance security and build trust by providing robust mechanisms for user authentication, secure storage, and encryption through biometric interfaces.

RQ 3: How could cloud service providers and their clients reduce their energy and computational resource consumption, without compromising on security? We conducted a study on the current state of low-energy cryptographic primitives, providing a comprehensive overview of recent advancements in the field. Additionally, we have shifted our focus to contactless smart cards, as they require minimal energy to perform cryptographic tasks in a highly secure manner.

RQ 4: What insights can be gained from analyzing cloud storage network access patterns, and how can intentionally obscuring these patterns mitigate associated risks? To mitigate network layer threats such as profiling, reconnaissance, and targeted attacks, we have integrated multiple privacy-focused overlay networks. Our main focus was the Tor Network, as it enables users to utilize pluggable transports, which are essentially plugins that further obscure network access patterns.

RQ 5: Why might it be necessary to supplement cryptography with steganography, and what advantages does this combination offer in terms of security and privacy in cloud environments? We present a mechanism for enhancing privacy by combining video steganography with encryption methods. Our main focus was to create a tool capable of modifying MKV files in order to transform them into steganographical containers, in a manner that would not affect the video quality or entropy.

1.2 Research Objectives

Taking into consideration the main objective of this thesis and the research questions presented in Section 1.1, we identified the following objectives of this research:

1. Research and define the threat model for public cloud storage and investigate mitigation of security and privacy risks with respect to data security and user privacy;
2. Research, implement and evaluate a robust mechanism for user authentication, secure storage, and encryption through biometric interfaces that uses mobile sensor data;
3. Research, implement and evaluate a privacy-first communication protocol that uses the identifiers of various hardware subsystems to generate a unique identifier for each mobile device;
4. Research, design, implement and evaluate a resilient software architecture that encompasses contact-less smart cards that can store non-exportable private cryptographic keys, providing enhanced security;
5. Research, architect, implement and evaluate an enhanced mitigation network layer that uses a cross-platform SOCKS5 interface that translates various protocols into XMPP messages;
6. Research, design, implement and evaluate an extensive security layer for security enhancement in cloud environments based on steganography techniques.

1.3 Thesis Outline

This thesis focuses on privacy enhancements for public cloud storage systems. We conducted a technology assessment of several public cloud storage providers to identify their key features, strengths, and weaknesses, as outlined in Chapter 2, Privacy Enhancements for Cloud-Edge Environments. We have analyzed the potential of enhancing the security of mobile user sessions by collecting data from both hardware and virtual sensors, as it can be seen in Chapter 3, Trust Enhancement Algorithms for Opportunistic Communication. In Chapter 4, Low Energy Encryption, we analyzed the energy consumption impact of various low-energy encryption algorithms. In Chapter 5, Highly Scalable Overlay Systems we analyzed the fundamental components of software-defined network overlays, such as Tor and I2P. This was done to introduce the essential concepts, principles, and technologies required to achieve online privacy, security, and anonymity. In Chapter 6, Enhancing Privacy with Steganography, we discussed the implementation of various steganography techniques for digital multimedia files. Our objective is to provide an overview of the advancements that have emerged over time and to raise awareness about the methods and reasons for concealing and covertly transmitting the information. Furthermore, we have implemented a novel steganographic method that leverages the flexibility of the MATROSKA video format to hide dm-crypt encrypted containers. Finally, in Chapter 7, Conclusion and future directions we conclude this Thesis, and we outline the original contributions of this Doctoral Thesis.

2 | Privacy Enhancements for Cloud-Edge Environments

The increasing demand for high-speed networks and storage resources has led to widespread adoption of Cloud-based services in various industries. While the benefits of this trend are evident, it also raises significant privacy concerns related to the confidentiality and integrity of data. To address these concerns, Cloud providers and third-party software applications have started offering simplified on-the-fly data encryption solutions.

2.1 Safeguarding Intellectual Property

In order to reduce traditional storage costs, some companies have started to outsource data storage to third parties to a certain extent. Even if all data should be subject to storage outsourcing, we should focus on sensitive data and how it is managed, in order to keep industry assets as safe as possible.

The last approach implies the protection of assets using the **trade secret** methodology in order to prevent competitors from gaining knowledge gathered from company-funded research and development.

While each intellectual property method of protection has its own advantages and disadvantages, according to Table 2.1, we can observe that trade secrets could be protected using strong privacy Cloud storage overlays.

2.2 Public Cloud Storage Service Providers

File synchronization and backup over Cloud platforms have become more and more ubiquitous in recent years. In order to observe the benefits of public cloud storage

Table 2.1. Advantages and disadvantages of intellectual property protection methods.

	Trade secrets	**Other means**
Advantages	Unlimited protection time. Publication is not necessary. Suitable for unprotectable work.	Simplified licensing procedures. Grants exclusive rights are opposable to anyone, including independent inventors.
Disadvantages	Loss has immediate effects. No protection against independent inventors. Some products are subject to reverse engineering.	Limited protection time. Once published, it can be subject to research and development by competitors. Reproducible by competitors after expiration.

platforms, we have performed a comparative analysis, based on white papers and online sources [3] regarding Dropbox, Google Drive, and Amazon Cloud Drive, which are convenient means of data backup and synchronization (Table 2.2).

Table 2.2. Comparison of storage Cloud services.

	Dropbox	**GoogleDrive**	**Amazon Cloud Drive**
File size limit	10 Gb	5 Tb	2 Gb
Transport layer security for data in transit	SSL\TLS [2048 bit keys]	SSL\TLS [2048 bit keys]	SSL\TLS [2048 bit keys]
Free storage	2 Gb	15 Gb	None
Encryption of data at rest	AES [256 bits keys]	AES [128 bits keys]	None

2.3 Challenges to Public Cloud Storage Adoption

Public Cloud storage services have become increasingly popular over time due to their numerous advantages, such as scalability, efficiency, and usability. However, despite their advantages, there are also potential risks associated with their use, particularly in terms of security.

Data loss is another significant security risk. Hardware malfunctions, human errors, and natural disasters can all lead to data loss when using Cloud storage. Although Cloud service providers typically implement redundancy and backup procedures to minimize the risk of data loss, it is still possible to experience such loss in a worst-case scenario.

In the event that data is stored in the Cloud, users may not have complete control over it. Under certain conditions, Cloud service providers may be granted specific rights to access, use, or even delete data. Concerns regarding data confidentiality and privacy may arise from this issue.

For both individuals and businesses, Cloud storage is a crucial resource, and therefore it is important to recognize the associated privacy and confidentiality risks. The security of Cloud-stored information can be enhanced by implementing robust encryption, secure transmission protocols, and multi-factor authentication.

2.4 Third-party Data Encryption Providers

While many individuals have already embraced Cloud storage, new security concerns are beginning to rise in the aftermath of several high-profile data breaches [4] targeted toward the major players in the Cloud storage industry. The most advertised apps in the domain of Cloud storage security overlays are Boxcryptor, Cloudfogger, AES Crypt, SpiderOak and Viivo. These applications help their users to protect their data confidentiality before it reaches any server, adding an additional layer of security.

2.4.1 Viivo

Viivo is a FIPS-140-2 validated software implementation that is free for personal use. It was created by the PKWARE company and it integrates with Dropbox, Box, OneDrive, Google Drive, and Copy. The cryptographic primitives used by Viivo are RSA-4096 and

AES-256, used together in order to achieve a higher degree of confidentiality. Viivo can be used on multiple operating systems, such as Apple Mac OS X, Windows and Android, ensuring cross-device interoperability and integration.

2.4.2 AES Crypt

AES Crypt is a software implementation available for iOS, Android, Linux, Php and Java that uses the standard AES and HMAC-SHA-256 to ensure data confidentiality and integrity. Even if it does not implement any type of federation, this application is very simple to use across many devices and it has an open-source implementation, suitable for peer review. The encryption process can be started by the user after a file and a password have been chosen. Regardless of the password complexity, the PBKDF2 algorithm is used in order to derive the AES algorithm key and iv, and the HMAC-SHA-256 key.

2.5 Conclusions

Although encryption is an essential security measure for safeguarding sensitive data, there are still issues related to its application. The most sensitive aspects associated with encryption are:

- Key management: Managing encryption keys can be difficult, especially when working with large amounts of data and many users. If keys are improperly managed, misplaced, or compromised, this could lead to unwanted disclosures;
- Key compromise: This occurs when an attacker obtains the ability to decrypt data using an encryption key. This can happen if the encryption algorithm is weak or if the encryption key has low entropy;
- Implementation flaws: When encryption algorithms are used incorrectly or are deprecated, vulnerabilities can be abused by an attacker. A poorly implemented encryption algorithm, for instance, might be vulnerable to attacks such as chosen ciphertext or side-channel attacks;
- Problems with compatibility: Depending on the cipher suite offered by a secure communication protocol, different encryption algorithms and keys may not be fully supported, making it impossible to share data between different systems in a highly secure manner;
- Performance overhead: Because encryption and decryption can be resource-intensive operations, they can impact overall system performance and add to processing costs.

To enhance the security of their data, users may opt to utilize encryption overlays prior to uploading their data to the Cloud.

The implementation of encryption overlays can serve as a protective measure against security breaches and ensure adherence to regulatory requirements by guaranteeing that data is encrypted prior to its transfer to the Cloud.

3 | Trust Enhancement Algorithms for Opportunistic Communication

In the last decade, a large number of people have become aware that the privacy and confidentiality of data can be trespassed upon and therefore began searching for alternatives. In order to communicate freely and securely, a group of people can decide to create a decentralized network.

3.1 Mobile Device Biometrics: Pitfalls and Challenges

In the last years, the number of smartphones capable of reading user fingerprints in order to perform authorization and authentication has tripled.

Unfortunately, previous work highlighted the fact that this implementation can allow some electrically conductive materials, molded in the victim's fingerprint, to bypass the sensor and permit access. Besides this, other pitfalls were proven, like: (i) **Confused Authorization Attack**. This type of attack is a technique used by malware-backed applications to trigger a biometric authentication prompt on the device screen; (ii) **Fingerprint Sensor Spying Attack**. In some implementations, the fingerprint sensor is not fully locked down and is handled through the Linux Kernel, thus enabling attackers to sniff data passing on the Serial Parallel Interface (SPI) bus; (iii) **Trusted fingerprint sensors exposed to the untrusted world**;

3.2 Mobile Device Fingerprinting

Device fingerprinting is a methodology based on standard or non-standard procedures, used to uniquely identify a specific physical device in a network. The fingerprinting procedure could be performed in the following manners [5] [1]: (i) **Client-based**. This approach implies that the computation is made locally, using various data sources, such as hard-disk serial number, network card MAC address, or other identifiers obtainable by a non-privileged user. The resulting value should be unique, persistent, and tamper-proof; (ii) **Server-based**. The fingerprint value is computed on a profiling server using connectivity details, without the need to install additional client-side software on the given workstation or personal computer. This approach is mainly used in online banking and e-commerce websites; (iii) **Browser-based**. This approach relies on data collected from the user's browser, such as cookies, user agent value, generic and privacy settings, installed plug-ins, etc.; (iv) **HTTP-based**. This method uses a fingerprinting method and consists of extracting specific metadata transmitted through an HTTP connection. Some of the used parameters are the compression method name, the support for proxies, the supported cipher suite, etc; (v) **Operating System-based**. This approach implies computing the unique identifiable value using the operating system information obtained from a network connection with the physical device; (vi) **TCP-based**. Similar to the

HTTP-based approach, this method relays on TCP protocol stack information gathered when a network connection is established.

3.3 A Mobile-first Trust Enhancement Protocol

Mobile device fingerprinting is a procedure that comprises methods for uniquely identifying a specific machine, in order to ensure authorization and communication between the participating nodes of a decentralized network.

The privacy and confidentiality level of the transmitted messages is an open problem, which will be addressed by our proposed solution Figure 3.1.

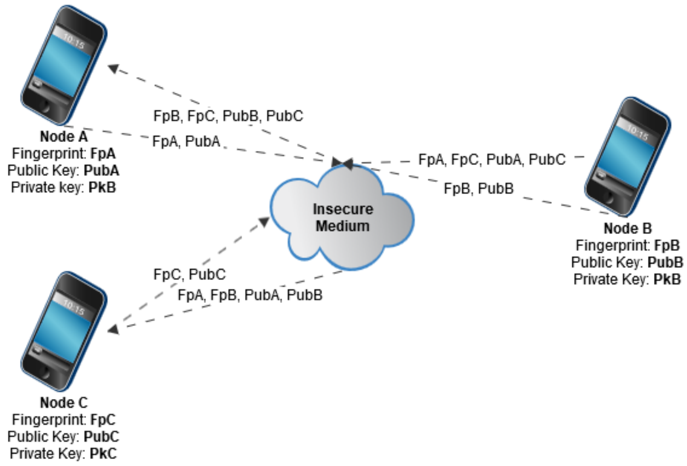

Figure 3.1. The simplified hand-shake phase of the proposed communication protocol.

In a simplified manner, as shown in Figure 3.1 that illustrates the basic building blocks of our proposed communication protocol:

- Each node in the spontaneous network has its own pair of asymmetric keys and a verifiable, uniquely-identifiable fingerprint.
- When a node advertises its presence, it broadcasts its fingerprint and public key for other nodes to learn.
- Receiving nodes correlate the public key with the fingerprint to verify the identity of the broadcasting node.
- If another node broadcasts an identical pair of keys or fingerprints, it should be rejected by surrounding peers.
- After a node broadcasts its details, surrounding peers begin a consensus to choose the oldest two nodes to verify the attending node's identity.
- If the verified entity fails to prove their identity, the verifiers propagate this information to prevent further connection attempts.
- If the node is validated, the censor peers advertise the result to allow further interactions.
- Nodes should broadcast heartbeat messages at regular intervals to detect nodes that are leaving.
- The maximum number of known peers for a node is restricted to 1000.
- The new peer verification procedure starts with a "hello message" broadcasted to the network to announce presence and discover available peers.

- The new participant sends a discovery message to announce its presence to the nearest available peer.

3.4 Device-based User Identification

In order to generate a unique identifier for a node, it is necessary to find a set of spoofing-resistant verifiable hardware and software attributes. When it comes to mobile devices, there is some information that is hardly changeable, such as the resolution of the screen, the hardware sensors list, the device manufacturer, and the operating system name.

Our solution combines the information obtained from two categories of pseudo-unique identifiers: NVRAM data and specific hardware features. From the NVRAM, the IMEI value is extracted automatically. The IMEI is considered to be hardly changeable without root permission.

On the software layer, the Android operating system does not provide reliable values which could be used as metrics in the fingerprinting process, because the Android ID and the Android Advertising ID are changed when the factory reset facility is called by the user. The method used for generating the unique identifier is based on the SHA-256 algorithm, in order to obtain a 256-bit length fingerprint value Figure 3.2.

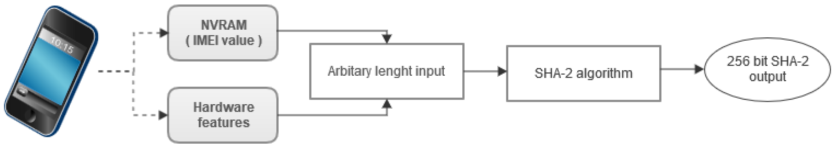

Figure 3.2. Fingerprint generation procedure.

The fingerprinting module computes the uniquely identifiable value associated with a node and a public key. The first problem encountered was that emulated devices cannot provide a valid IMEI, so for this reason, we had to implement a software module to generate this value once per device.

Even if the proposed solution is feasible on the Android or Windows operating system, the limited system privileges on iOS disallow sensitive information access, such as the **IMEI!** or environment data.

3.5 A Multi-layer Approach for Contextual Mobile Devices Fingerprinting

Our proposed solution aims to gather all raw data output provided by the specific device's sensors, without requesting privileged access to its resources, since rooting the mobile phone or tablet will diminish the security and privacy level guaranteed by the device's manufacturer. The sensor data can be used to provide information about the user's contextual environment and motion status.

Firstly, we chose to identify how the sensor data is influenced by the user's motion status due to the fact that gathering contextual environment data is related to the device's motion state. Therefore, the implementation was tested using three different scenarios, as follows: (i) When the user is in a stationary state (**Stationary State** - SS); (ii) When the user is moving (**Moving State** - MS); (iii) When the user is driving or is in a car (**Driving State** - DS).

The main goal of our research was to identify as much information as we could regarding the user's behavioral patterns, influenced by environmental and human-machine interaction, as perceived by an Android-based sensor stack. Our results have shown that modern mobile smartphones can sense the presence of a person that interacts with them and can provide some information about the usage context.

This approach might provide meaningful insights about user behavioral patterns without any special permission and without disclosing the actual owner identity or biometric data.

3.6 Mobile Device Reliability in the Context of Privacy

Mobile devices are susceptible to a range of security threats, including but not limited to malware, phishing attacks, and data breaches. The compromise of personal information poses a significant risk to individuals, including but not limited to financial fraud, identity theft, and cyberstalking.

Mobile devices have the capability to collect a substantial amount of personal information about their users, such as their geographic location, search history, and application usage. In order to minimize the amount of collected personal data, it is advisable to opt for devices that provide extensive privacy settings. Despite offering similar features, certain applications may collect a greater amount of data than others.

The reliability of the manufacturer is of utmost importance when selecting a device. It is crucial to choose a product from a reputable company that prioritizes the protection of personal information. It is recommended to choose companies that have a proven track record of protecting user information and effectively managing security vulnerabilities.

3.7 Conclusions

Due to inherent variations in the manufacturing process and other factors, it is possible for two identical devices to display sensor values that differ slightly under the same circumstances. There may be differences in the sensors themselves, the components utilized in constructing the device, or the operating environment, even when devices are manufactured according to the same specifications.

Additionally, sensors can be affected by a number of variables, such as temperature, humidity, and electromagnetic interference. These factors can cause sensors to produce different readings even when operating under identical conditions. To make sure that sensor readings are accurate and consistent across various devices, modern smartphones are usually calibrated.

If sensor data is used to enhance the user authentication process, it needs to undergo additional processing or filtering to eliminate any noise or adjust for variations in sensor readings caused by environmental factors and natural fluctuations. Even if the devices are not identical, filtering and normalizing sensor readings can help improve the accuracy and consistency of the authentication process.

4 | Low Energy Encryption

4.1 A survey on the efficiency of cryptographic primitives

In order to obtain a better cost/security ratio, it is necessary to analyze the time and computational resources consumption for classic and other emerging cryptographic primitives.

4.1.1 Confidentiality Primitives

In order to determine which of the modern cryptographic algorithms are suitable in our use case, it is necessary to evaluate the trade-off between CPU consumption, speed, and security displays more details about each cryptographic primitive, although, in our critical analysis, we used only the data from the Number of Rounds, CPU Cycles per Byte, and CPU Cycles to set up the key & IV.

Table 4.1. Encryption Algorithms CPU Consumption.

Algorithm	Number of rounds	Block size (bits)	Cycles per byte	CPU Cycles to set up key and iv
DES	16	64	54.7	1532
IDEA	8	64	49.9	1277
AES	10/12/14	128	12.6	1277
RC5	12/16	32/128	23.4	4665
RC6	20	128	17.3	5128
MARS	2 x 16	128	37.2	6435
XTEA	32	32	67.4	1165
Serpent	32	128	54.7	2191
Twofish	16	128	29.4	14121

4.1.2 Integrity Primitives

From a cost perspective, computational resource consumption is a metric equally important to the level of trust provided by the selected hashing method in power-constraint environments Table 4.2.

4.2 Low Power Cryptographic Primitives

The term "low-power cryptographic primitive" denotes a cryptographic operation or function that can be executed on low-power devices, including but not limited to mobile phones,

Table 4.2. Hashing Algorithms CPU Consumption.

Algorithm	Number of rounds	Input size (bits)	Hash size (bits)	CPU Cycles per byte
MD5	4	512	128	6.8
SHA-1	4	512	160	11.9
SHA-256	64	512	256	15.8
SHA-512	80	512	512	17.7
WH	10	64	64	30.5

Internet of Things (IoT) devices, and other battery-operated gadgets. The utilization of this particular type of primitive is crucial in guaranteeing secure communication and data transfer within environments that have limited resources.

Low-power encryption may be significantly impacted by processor evolution. More complex and computationally intensive cryptographic algorithms, which were previously impractical to implement in low-power environments, can now be supported by processors as they become more powerful and energy-efficient.

4.3 Smart Card Technology and Security

Smart cards have emerged as a reliable and effective solution for securely storing confidential cryptographic information and processing sensitive data [7]. Smart cards are small devices that contain embedded memory and microprocessors.

Smart cards are utilized in various applications and industries [2], including: (i) Banking institutions offer their customers smart cards to facilitate electronic payment systems, including both debit and credit cards; (ii) Transportation companies are implementing smart cards as a convenient payment method for users of public transportation; (iii) Governments issue identification documents, such as passports, national identity cards, and driver's licenses, which are based on smart cards; (iv) Healthcare systems utilize smart cards to store patient information and provide secure access to medical records; (v) Access control mandates the usage of a smart card for employee identification and building access.

The choice of programming language and platform depends on the specific requirements of the application, as each platform has its own unique strengths and weaknesses. A large community of developers, along with a wide range of tools and libraries, are readily available for Java Card, which is a well-established platform. DotNET Card is a relatively new platform that offers robust support for the .NET Framework and a cutting-edge development environment.

4.4 Mobile-first Storage Overlay

In this section, we propose a novel smartphone-based Cloud storage encryption overlay, resilient to key theft, trojans, keyloggers, inference, and account compromise. First, we will describe the architecture of the system and then other relevant aspects regarding the functionality.

4.4.1 Architecture

One of the most sensitive aspects when it comes to cryptographic systems is key management, because there is no secure place to store keys on an internet-connected computer.

In order to overcome this issue, we propose the migration of sensitive data to another trusted domain. This trust domain is a programmable smart card, enhanced with a cryptographical co-processor and powered by electromagnetic induction.

While a smart card requires additional set-up and hardware, it is a small cost in order to achieve a higher degree of security for sensitive data.

4.4.2 The Java Card Applet

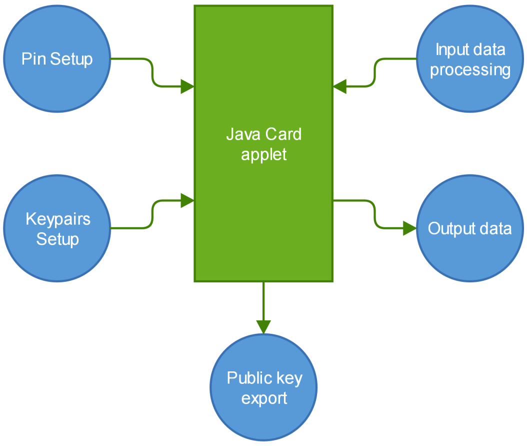

Figure 4.1. Java Card applet architecture overview.

Java Card is a technology targeting low-power embedded devices in order to allow them to securely run applications. The card ecosystem is very secure, forbidding applet disclosure and tampering, in order to comply with various policies imposed by banks and governments.

Our proposal Figure 4.1 regarding the applet implies a software implementation that is protected by a pin and securely stores one key pair in order to perform data encryption for the files which will be uploaded to public or private Cloud storage.

The communication with the card will be done through **NFC!** (**NFC!**) in order to perform applet deployment, **PIN!** (**PIN!**) authentication, symmetric encryption, symmetric decryption, integrity computations, integrity checks, asymmetric encryption, and asymmetric decryption.

4.4.3 Storage Layout and Data Structures

The storage layout Figure 4.2 of our project will be implemented in a manner that will not allow an adversary to learn information about file names, sizes, and content.

In order to achieve the previously mentioned characteristics, a data federation layer has to be implemented, in order to achieve integration with multiple storage providers.

The storage layout will be split into four distinct zones, the metadata store, the temporary metadata store, the block store, and the temporary block store. This separation will ensure file splitting, file name encryption, consistency, and data encryption.

The **metadata store** will contain encrypted **metadata blobs** holding data regarding real file name, file size, and block identifiers.

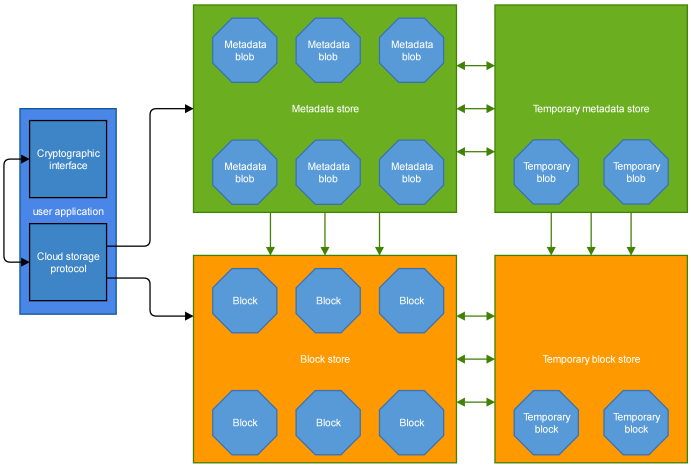

Figure 4.2. Cloud storage architecture overview.

The **temporary metadata store** is an intermediate location used to securely store intermediate **metadata blobs** during the encryption process.

The purpose of the **block store** is to hold individual files containing secure information resulting from file encryption and splitting operations. The blocks will consist of files with random file names, in order to prevent real file name disclosure.

In order to ensure consistency, until the file encryption operation has been completed, the intermediate data will be uploaded to the **temporary block store**.

After an encryption operation has been completed successfully, the data is moved from temporary stores to default stores. Any temporary file will have its name generated randomly.

4.5 Conclusions

The level of power consumption exhibited by an encryption algorithm does not necessarily correlate with its strength. The strength of an encryption algorithm depends on the mathematical complexity of the algorithm and the size of the encryption key used to secure the data.

In order to enhance security and protect highly sensitive data, telephones and smart cards can be used in conjunction. Contactless smart cards can utilize the Near Field Communication (NFC) interface to provide secure data encryption and digital signatures. If a smartphone is compromised, the private keys stored on the smart card cannot be accessed by an attacker. This use case has the potential to minimize the impact of an incident, as long as the adversary does not have physical access to the smart card.

5 | Highly Scalable Overlay Systems

Nowadays, the vast majority of websites are encrypted through **HTTPS!** (**HTTPS!**). However, there are still major concerns regarding the **DNS!** (**DNS!**). In such a scenario, before loading any encrypted website, the browser forwards a **DNS!** name resolution request in order to determine which internet protocol address is associated with the domain that the user has requested.

5.1 Privacy Tools and the Digital Age Censorship

Online activities are prevalent in our daily lives, but many people are unaware of how much information they leak by accessing various services. Even with encrypted traffic, internet service providers can infer private data regarding their client software, interests, and habits.

Another advantage of overlay networks is given by the fact that, if they are employed at client endpoints, they can transfer data by using virtual addresses rather than destination IP addresses. Since an IP address is not required to communicate with a peer, the number of possible virtual subnets can become much larger.

Additionally, depending on the monitoring strategy of an Internet service provider, the network access patterns associated with these tools can actually attract unwanted attention from law enforcement, even if no illegal activity is conducted by privacy enthusiasts.

5.2 Hiding Cloud-Edge Network Access Patterns

Internet privacy overlay tools, such as Tor, Freenet, I2P, GnuNet, and Loki Network, offer a variety of legitimate use cases for individuals who prioritize online privacy and security.

These tools can be utilized by journalists and whistle-blowers to safeguard their sources and ensure the secure transmission of sensitive information.

In order to further enhance privacy and to prevent advanced firewalls to block connectivity, Tor Project introduced Pluggable Transports (PTs), a collection of software modules that are utilized to obscure Tor traffic, rendering it more challenging for network censors to detect and obstruct it.

Tor Pluggable Transports are a crucial tool for users who are seeking to bypass internet censorship and safeguard their online privacy. By making it more challenging for network censors to detect and block Tor traffic, pluggable transports help ensure that the Tor network remains accessible to users worldwide.

5.3 Cross-platform Network Privacy Overlay Adapter

The evasion of various ISP censorship and surveillance can be achieved using our proposed solution, which is essentially an anonymizing set of tools meant to tunnel various protocols in a scalable manner.

In order to provide compliance with existing applications, our solution would implement a SOCKS5 proxy interface. The main task of this proxy would be the splitting and transmission of any outgoing messages to different communication channels like AllJoyn, Irc Chat, and XMPP chat.

Our implementation was designed for Cloud storage network access patterns obfuscation and low latency data transfer. In order to hinder the detection of traffic generated by our implementation on any network, the endpoints must be deployed both on a subscriber machine and inside the infrastructure of the Cloud service provider. Using this approach, the traffic patterns become more complex and, therefore, harder to identify.

5.3.1 Repurposing XMPP for Traffic Obfuscation

Our solution has to be configured with one or more options from a list of predefined transport channels. Since a transport channel is established between bots using a chat service Figure 5.1, account creation is required in order to ensure communication.

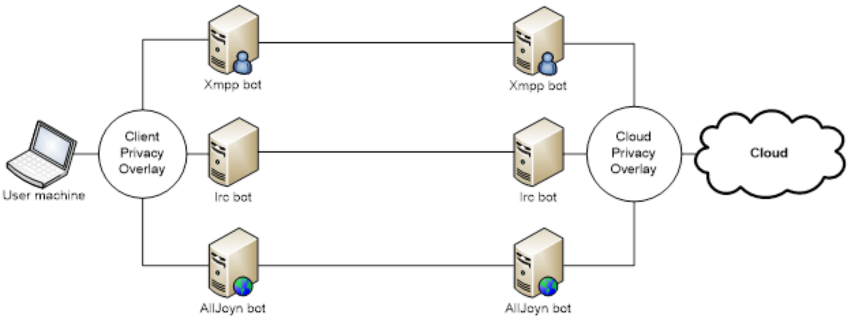

Figure 5.1. Proposed profiling resilient architecture.

Our solution relies on protocols operating on the OSI transport layer Figure 5.2, initially designated for text messaging. We have chosen messaging protocols because they can be easily repurposed to transfer any type of data, in a full duplex manner. In order to overcome possible speed issues caused by bandwidth throttling, we used different chat services simultaneously.

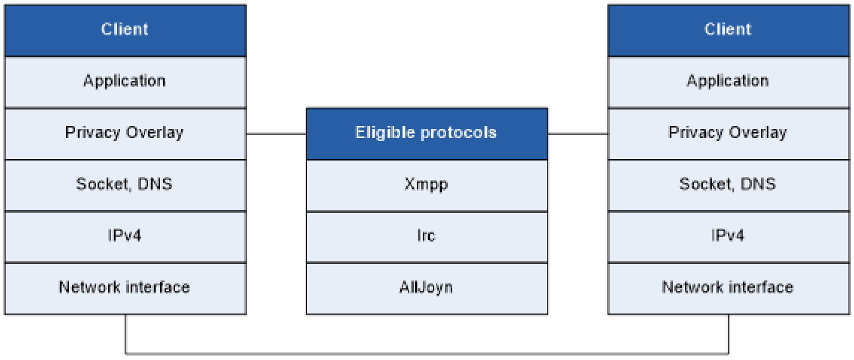

Figure 5.2. Privacy overlay integration with the well-known network layers.

Our proposed architecture is based on two high-level components, a network traffic obfuscator, and a network traffic deobfuscator.

The Socks5 compatible interface is a component specially created to ensure integration with existing software. This way, applications that can use a Socks5 proxy can benefit from this privacy overlay, without requiring any special modifications.

5.3.2 The SOCKS5 Overlay Interface

Our implementation aims to provide easy integration with existing applications, by combining a local SOCKS5 proxy with various legitimate protocols which allow input manipulations. Such protocols are used mostly by text chat services.

We implemented and benchmark a cross-platform SOCKS5 proxy server Figure 5.3, which will be used to capture application-generated data. Our proxy server uses a limited set of SOCKS5 instructions. Currently, it supports only the TCP CONNECT method and provides no authentication method.

The handshake manager runs in a separate thread, polling session queues. If any data has been pushed by the server, it employs a session-specific protocol handler to evaluate the requests and writes back its resolution afterward.

The TCP choreographer is responsible for the creation and management of multiple connections. It also routes the data generated by the actual requests and the listening server.

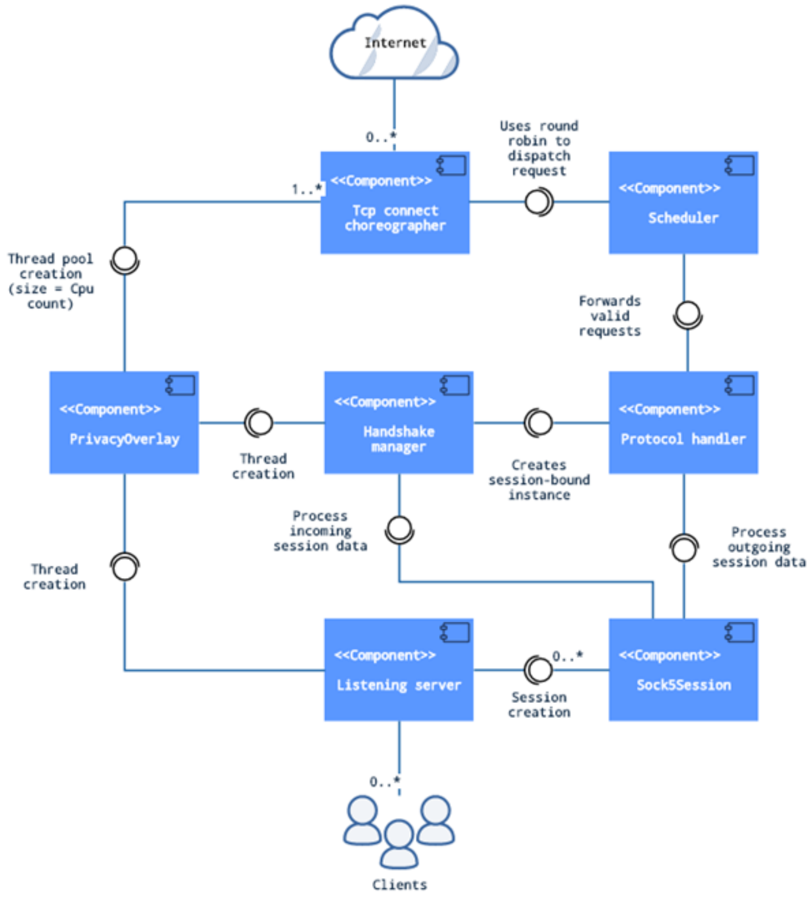

Figure 5.3. SOCKS5 local server high-level architecture.

We conducted the tests multiple times, modifying the default request time to decrease error rates. We achieved errorless execution when we configured the default request timeout value at 10000 milliseconds.

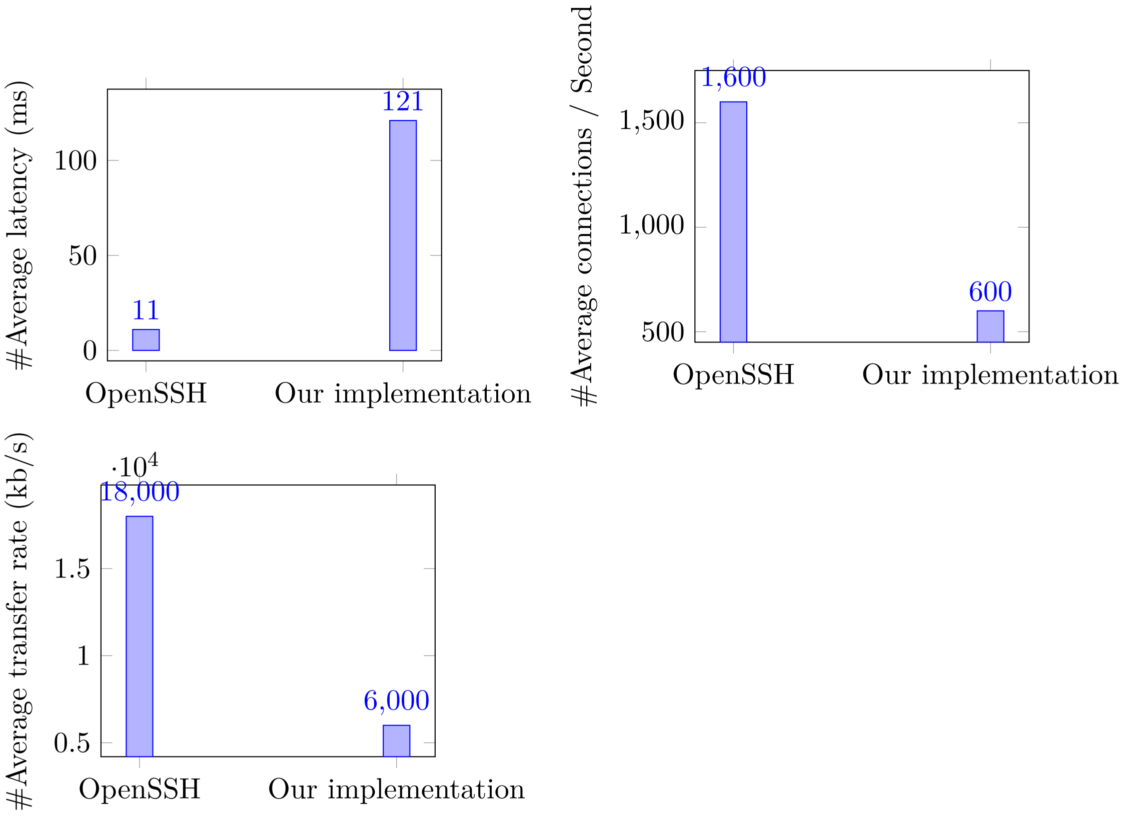

Figure 5.4. Benchmark results. Average latency, average connections / second, and average transfer rate.

5.4 Conclusions

As response to ISP surveillance, new privacy enhancement procedures are constantly researched in order to evade censorship and personal data misuse. Our proposed solution is centered on the idea that every person has the right to be anonymous and has complete control over his data.

Our solution aims to provide a platform-independent method of integration with software-defined networks by leveraging the existing SOCKS-5 protocol. Even if the idea is not new, we needed a software solution that had no dependencies on external libraries and a relatively small, maintainable, and trivial codebase.

However, there is still work to be done in specifying a reliable overlay protocol and in choosing a methodology to enhance trust between the communicating entities. Our proposed solution is not intended to replace any existing privacy overlay, but to interoperate with other existing services, in order to bypass censorship and surveillance. Even if this solution might be used for questionable purposes, we acknowledge the fact that every person has the freedom to decide how to use the internet.

6 | Enhancing Privacy with Steganography

Since ancient times, steganography has been used to conceal secret messages. Steganographic methods have been refined and transformed by many scientific breakthroughs over time. In computer science, steganography can be used to conceal private information through various methods and techniques.

This chapter explores the feasibility of using steganography in multimedia files to hide confidential data. This study examines the characteristics of MKV files and the available steganography techniques that can be used to reliably store sensitive data in files.

Our proposed implementation utilizes specific Matroska [6] elements to store additional data into the file, while not impacting the characteristics of the video frames.

6.1 The Matroska Video Container Format

Matroska Video (MKV) is a well-known multimedia container format that can store a variety of media types, including video, audio, subtitles, and still pictures. It was created in 2002 as an open-source, cost-free container format, and it is now maintained by the nonprofit group Matroska.org.

Extensible Binary Meta Language (EBML) is used to specify the file structure in the Matroska Video (MKV) format. Each element in the file is given a specific label by EBML using what are known as EBML IDs. These IDs are used to specify each element's location in the file, as well as its format.

The EBML IDs that are frequently used in MKV files include:

- EBMLHeader (ID: 0x1A45DFA3): This ID marks the beginning of the file and provides information about the version of EBML used and the size of the file.
- Segment (ID: 0x18538067): This is the top-level element in an MKV file, and it contains all other elements in the file.
- Info (ID: 0x1549A966): This element contains general information about the MKV file, such as the duration, date created, and title.
- Attachments (ID: 0x1941A469): This element contains additional files that are related to the MKV file, such as cover art, subtitles, or additional audio tracks.
- Void (ID: 0xEC): This element is a reserved type, and it is primarily used for padding or filling gaps within the EBML structure.

The Matroska specification, which is available on the `https://matroska.org/technical/basics.html` website, contains the full list of EBML IDs.

Further analyzing the specifications, we concluded that the elements of type Void or Attachment would be an ideal carrier for hidden information, since altering them does not impact video quality.

Given the fact that steganographically hidden data can be detected, it is highly advisable for users to protect its confidentiality with reliable encryption methods.

6.2 Encrypting Volumes Using dm-crypt

Dm-crypt [9] is a subsystem for disk encryption in Linux that offers transparent encryption of block devices. It is a cryptographic module at the kernel level that facilitates the creation of encrypted partitions or volumes, thereby enabling secure storage of data on storage devices such as solid-state drives (SSDs), hard drives, or USB drives.

Dm-crypt is widely used in Linux-based systems to provide data-at-rest encryption, ensuring that even if the storage device is physically compromised or stolen, the data remains protected [8].

In typical usage scenarios involving dm-crypt, encryption is applied to a partition or volume starting from the beginning, at offset 0. In our specific scenario, which involves steganography, it is desirable to designate a specific volume within a carrier file while leaving the remainder unaltered.

6.3 Proposed solution: Hidden in the Void

Every time a user creates an encrypted volume to use with dm-crypt, they can use the "plain dm-crypt with offset" encapsulation. The resulting file will contain random data and an encrypted volume at the designated offset.

From a high-level perspective, the purpose of our software is to read all MKV metadata from a file and combine it with a dm-crypt container, in order to obtain a new MKV file that can be opened in any media player, but also in mounted as a volume, in order to access encrypted data.

```
 private static void bindFiles() throws IOException {
   final List<Element> elementList =
    Reader.getAllElementsFromFile(INPUT_MKV);
   final Writer writer = Writer.aBuilder()
         .setElementList(elementList)
         .setInputFile(INPUT_MKV)
         .setOutputFile(OUTPUT_MKV)
         .setEmbeddableFile(CONTAINER)
         .build();
   writer.assemble();
}
```

Listing 6.1. Top level code snippet

Updating the EBML elements of an MKV file can affect the compatibility of the file with various media players, therefore, we generate a new output file whenever we want to add extra data. This also makes our implementation easier to understand and maintain.

6.4 Results

Using our solution can have complex and context-specific implications. It can be a useful tool for security and privacy, but it can also be used for wrongdoing or unethical behavior. From a security perspective, it can be applied as a method to safeguard sensitive or private information. Data can be protected from prying eyes and unauthorized access by being hidden in plain sight.

To demonstrate the advantages of our proposed solution, we conducted an experiment to measure the entropy variation induced by our tool. We obtained the initial MKV file entropy and the carrier file entropy by using the Linux command-line tool "ent." The results of our study demonstrate that the employed methodology resulted in a reduction of entropy, even when an encrypted container was used. This phenomenon arises as a result of introducing low-entropy padding data into the Void element.

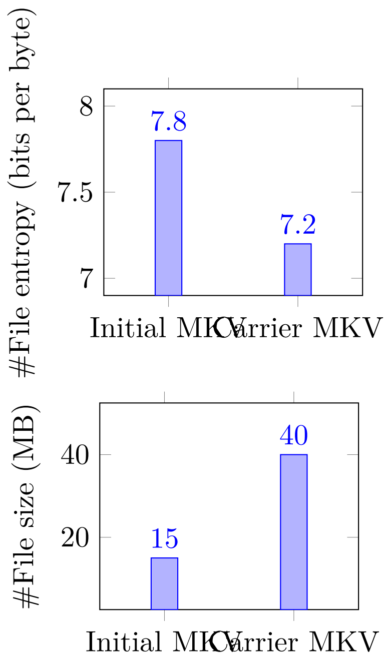

Figure 6.1. File entropy and size comparison using the Linux ent tool.

Another use for our implementation is automated steganography detection. Our software can parse MKV files accurately while disallowing unknown elements. Attempting to load a malformed MKV file will always result in an error. Our solution can detect invalid MKV elements three times faster (Figure 6.2) than the open-source tool, MKVInfo. To demonstrate this fact, we developed a shell script and compared the speed at which both methods solve the problem.

```
#!/bin/bash
var=$(mkvinfo -v $@ | grep -c Unknown)
if [ $var > 0 ]; then echo "suspect for steganography"
else echo "valid"
fi
```

Listing 6.2. MKV anomaly detection script (mkvhunt.sh) .

We also compared the speed for various input file sizes, in order to observe that our method scales linearly. In this context, scalability refers to a system's capacity to manage growing workloads without experiencing a significant decline in performance. By benchmarking different input sizes, we evaluated the scalability of our system and determined that it can effectively handle larger data sets.

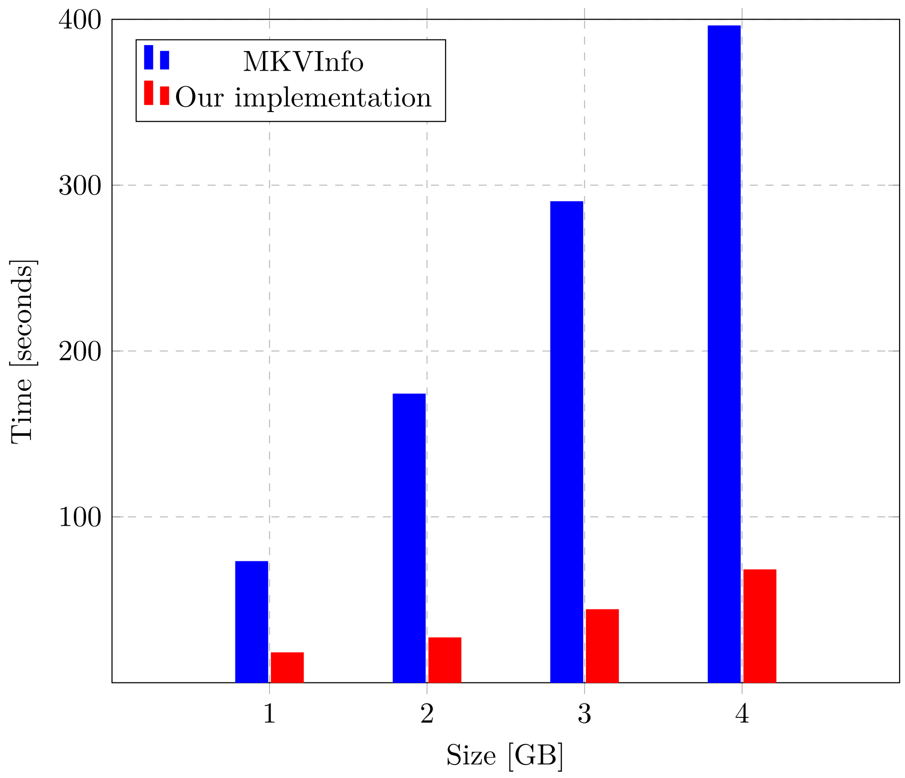

Figure 6.2. Speed comparisons using different file sizes.

As a potential future direction, our tool could be integrated with the open-source forensic software, Autopsy, to identify concealed data within MKV files. Furthermore, our software has the potential to be expanded to identify anomalies and calculate the entropy of individual EBML elements within an MKV file.

The unethical usage of our steganography tool is difficult to detect, mainly because it conceals data inside a valid MKV video file. In order to prevent unethical use of our solution, future research could be focused on developing digital forensics tools capable of detecting our methods. Since our software implementation was developed in Java, it can be easily integrated as an Autopsy plugin that detects various MKV anomalies, such as abnormally long MKV "void" elements.

7 | Conclusion and future directions

7.1 Conclusions

Encryption is a fundamental principle in the realm of information security and is of paramount importance in the protection of sensitive data. It involves the transformation of plain, comprehensible data into a coded format that can only be accessed or comprehended by authorized individuals.

In Chapter Privacy Enhancements for Cloud-Edge Environments we provided a comprehensive overview of encryption in the context of cloud computing:

- Encryption is important for protecting sensitive data, but there are issues related to its application, including, but not limited to key management, key compromise, implementation flaws, compatibility problems, and performance overhead are all potential risks associated with encryption.
- Robust encryption algorithms, careful key management, and adherence to industry best practices can help reduce these risks.

In Chapter Trust Enhancement Algorithms for Opportunistic Communication we analyzed how smartphones could enhance cloud-service authentication by leveraging sensor data in context-dependent scenarios.

- Identical devices can display slightly different sensor values due to manufacturing variations and environmental factors.
- Calibration during manufacturing helps ensure accurate and reliable sensor data.
- Filtering and normalizing sensor readings can improve the accuracy and consistency of user authentication processes.
- Establishing a threshold for sensor readings can eliminate extraneous data and highlight relevant patterns or behaviors for user profiling.
- The inclusion of a threshold can improve the precision and relevance of user profiling through mobile sensor data.

To establish communication with cloud-based systems, as well as resource-limited devices such as smartphones, the utilization of cryptography is imperative. In Chapter Low Energy Encryption we analyzed the impact of encryption on resource-constraint devices.

- Low-power encryption is important for devices with limited processing capabilities and battery life, such as IoT and mobile devices. Therefore, low-power encryption can improve device performance and battery life.
- The strength of an encryption algorithm depends on its mathematical complexity and the size of the encryption key, not just its power consumption.
- Smartphones are vulnerable to malware and physical theft, and using smart cards in conjunction with phones can enhance security.

Encrypted network activity may reveal certain patterns and metadata related to a user's actions. However, the actual content of the communication or specific details cannot be accessed without the decryption keys.

In Chapter Highly Scalable Overlay Systems we analyzed existing anonymization tools that provide overlay networking to address metadata leakage, in the context of digital surveillance and censorship.

- Network privacy enhancement methods are continuously being researched in order to evade censorship and prevent personal data misuse.
- Overlay networking, also known as software-defined networking, is a network-enhancing technique that can be defined using software, opening new opportunities in the field of security research.

In Chapter Enhancing Privacy with Steganography we analyzed well-known steganography techniques applied to multimedia files.

- Steganography has been used since ancient times to conceal secret messages. In computer science, steganography can be used to conceal private information through various methods and techniques.
- Multimedia files are a viable option for steganography due to their relatively large size. Additionally, they are frequently exchanged through various communication channels, making them an ideal carrier for hidden messages or data.

7.2 Original Contributions

The main contributions presented in these chapters provide valuable insights for discussing security in the context of cloud storage.

We can conclude that the primary contributions of the research to this thesis are:

- We conducted a technology assessment of several public cloud storage providers to identify their key features, strengths, and weaknesses, as outlined in Chapter Privacy Enhancements for Cloud-Edge Environments.
- As smartphones increasingly rely on cloud environments to perform multiple tasks, we have analyzed the potential of enhancing the security of mobile user sessions by collecting data from both hardware and virtual sensors, as it can be seen in Chapter Trust Enhancement Algorithms for Opportunistic Communication. This data can be utilized to initiate de-authentication events when device sensors detect unusual usage patterns.
- When communicating with devices that have limited battery and computational resources, it is crucial for a cloud storage provider to ensure confidentiality by implementing robust encryption algorithms. Therefore, in Chapter Low Energy Encryption, we analyzed the energy consumption impact of various low-energy encryption algorithms.
- In Chapter Highly Scalable Overlay Systems we analyzed the fundamental components of software-defined network overlays, such as Tor and I2P. This was done to introduce the essential concepts, principles, and technologies required to achieve online privacy, security, and anonymity.
- In Chapter Enhancing Privacy with Steganography, we discussed the implementation of various steganography techniques for digital multimedia files. Our objective is to provide an overview of the advancements that have emerged over time and to raise awareness about the methods and reasons for concealing and covertly transmitting the information. Furthermore, we have implemented a novel steganographic method that leverages the flexibility of the MATROSKA video format to hide dm-crypt encrypted containers.

7.3 List of Publications

To validate our findings we have published several papers in prestigious Conferences proceedings and International Journals as follows:

1. **Apostol, G. C.**, Mocanu, A.-E., Mocanu, B. C., Radulescu, D., Negru, C., Petre, I. & Pop, F., In Studies in Informatics and Control, 1220-1766, 2023. https://doi.org/10.24846/v32i2y202310;

2. Mocanu, A.-E., Mocanu, B.-C., **Apostol, G. C.**, Negru, C., Petre, I. & Pop, F., In 24th International Conference on Control Systems and Computer Science (CSCS24) 2023. IEEE. Accepted;

3. **Apostol, G. C.**, Mocanu, A. E., Mocanu, B. C., Radulescu, D. M., & Pop, F. (2023, February). CPSOCKS: Cross-Platform Privacy Overlay Adapter Based on SOCKSv5 Protocol. In Green, Pervasive, and Cloud Computing: 17th International Conference, GPC 2022, Chengdu, China, December 2–4, 2022, Proceedings (pp. 149-161). Cham: Springer International Publishing;

4. **Apostol, G. C.**, Mocanu, B. C., Radulescu, D. M., Petre, I., & Pop, F. (2022, September). Hiding cloud network access patterns for enhanced privacy. In 2022 21st RoEduNet Conference: Networking in Education and Research (RoEduNet) (pp. 1-5). IEEE;

5. Mocanu, B. C., **Apostol, G. C.**, Radulescu, D. M., & Serbanescu, C. (2022, July). TrustS: Probability-based trust management system in smart cities. In 2022 21st International Symposium on Parallel and Distributed Computing (ISPDC) (pp. 65-69). IEEE.

6. **Apostol, G. C.**, Borcea, L., Dobre, C., Mavromoustakis, C. X., & Mastorakis, G. (2021). A Survey on Privacy Enhancements for Massively Scalable Storage Systems in Public Cloud Environments. Big Data Platforms and Applications: Case Studies, Methods, Techniques, and Performance Evaluation, 207-223;

7. **Apostol, G. C.**, & Pop, F. (2016, May). MICE: Monitoring high-level events in cloud environments. In 2016 IEEE 11th International Symposium on Applied Computational Intelligence and Informatics (SACI) (pp. 377-380). IEEE.

7.4 Future Work

The research area of improving privacy for public cloud storage services presents numerous unresolved issues that require further investigation.

These issues can be presented as follows:

- The enhancement of user awareness and education is a crucial aspect in the context of cloud computing, as it serves to increase the comprehension of privacy risks and best practices among users.
- Future research should aim to address the privacy concerns that arise from the collection and modeling of sensor data on smartphones, with a particular focus on the measures that can be taken to ensure the security of such data.
- Future development should focus on the integration and compatibility of post-quantum encryption algorithms with the current smart card infrastructures, protocols, and standards. The aim is to ensure a seamless integration of these algorithms with the existing systems.

- The focus of future research could be on enhancing the resistance of TOR pluggable transports against detection and blocking efforts by censors. The objective is to develop strategies that can effectively mitigate the impact of such efforts on the functionality of pluggable transports.
- Future research should address the challenges posed by steganalysis, which is the process of detecting the presence of hidden data within media. Developing steganographic techniques that can withstand various steganalysis methods, while maintaining a low probability of detection, is a challenging task.

Bibliography

[1] Norihiro Fukumoto, Shigehiro Ano, and Shigeki Goto. Passive smart phone identification and tracking with application set fingerprints. *Proceedings of the Asia-Pacific Advanced Network*, 36:41–48, 2013.

[2] Mike Hendry. *Multi-application smart cards: technology and applications*. Cambridge university press, 2007.

[3] Dropbox Inc. Dropbox business security (white paper). https://www.dropbox.com/static/business/resources/Security_Whitepaper.pdf, 2015. Accessed: 2016-06-12.

[4] Umer Khalid, Abdul Ghafoor, Misbah Irum, and Muhammad Awais Shibli. Cloud based secure and privacy enhanced authentication & authorization protocol. *Procedia Computer Science*, 22:680–688, 2013.

[5] Threat Metrix. Device fingerprinting and fraud protection whitepaper.

[6] Nikolaos Pitropakis, Costas Lambrinoudakis, Dimitris Geneiatakis, and Dimitris Gritzalis. A practical steganographic approach for matroska based high quality video files. In *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, pages 684–688. IEEE, 2013.

[7] Wolfgang Rankl and Wolfgang Effing. *Smart card handbook*. Wiley, 2010.

[8] Jan Richter. An introduction to luks disk encryption. In *Annual Digital Forensic Research Workshop*. Citeseer, 2010.

[9] Herbert Robertson. dm-crypt: Transparent disk encryption subsystem for linux. In *Linux Symposium*, 2007.

[10] Irvin Zhan. Dnscatproxy: A pluggable transport based on dns tunneling, 2015.