

# CHALMERS



Evaluation of Privacy Enhancing Technologies

*Master of Science Thesis in Networks and Distributed Systems*

Elpidoforos Arapantonis

Chalmers University of Technology  
Department of Computer Science and Engineering  
Göteborg, Sweden 2014

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.

The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Evaluation of Privacy Enhancing Technologies

Elpidoforos Arapantonis

©Elpidoforos Arapantonis, January 2014.

Examiner: Erland Jonsson

Chalmers University of Technology  
University of Gothenburg  
Department of Computer Science and Engineering  
SE-412 96 Göteborg  
Sweden  
Telephone + 46 (0)31-772 1000

[Cover: The idea for the image was taken by the [1] and has been re-created in image processing software]

Department of Computer Science and Engineering  
Göteborg, Sweden January 2014

## Abstract

Anonymity, Privacy and Privacy Enhancing Technologies (PET) are the terms which are going to be discussed during the course of this project. The breakdown of the formal terminology and the theory will help the readers to follow the analysis of the existing implementations. The evaluation on the PETs will give some useful results that will be discussed in the conclusion.

The primary goal of this project is the in depth study of existing PET implementations. This will help not only to understand their main functionalities, but also to classify them in categories. A further elaboration on the classification models will help the readers to understand the present and future needs, concerning the additional features that the PETs need to cover.

To assess the effectiveness of an implementation, a series of tests will be conducted with the help of an existing web application as well as with a custom made test (fingerprinting). The web application will check the implementation's behavior by conducting tests in various potential vulnerable parts and the fingerprinting will test the user's uniqueness regarding to the traces, which are left behind. The results from the evaluation are going to be an essential guide for the users and will help them understand which implementation is suitable to their needs.

Finally, the conclusion will give a broader point of view regarding the preservation of the user's anonymity. It is clear that there are anonymity systems which are not included in this project. This presents a good opportunity for a future research either in a wider perspective or by focusing on specific implementations.



## **Acknowledgements**

I would like to express my deep regards to my supervisor Erland Jonsson for his patience and guidance throughout the course of this thesis.

I would also like to thank my parents and my friends for their great support and their constant encouragement without which this assignment would not be possible.

Göteborg 4/12/2013



# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Motivation on Privacy Enhancing Technologies . . . . .	2
1.2	The Goal . . . . .	3
1.3	Definitions . . . . .	3
1.4	Background . . . . .	4
1.4.1	Cryptography . . . . .	5
1.4.2	Mix Nets . . . . .	12
1.4.3	Onion Routing . . . . .	12
<b>2</b>	<b>Review of Existing Implementations</b>	<b>14</b>
2.1	Anonymity and pseudonymity tools for emails . . . . .	14
2.1.1	Type-0 remailers . . . . .	14
2.1.2	Type-I remailers . . . . .	15
2.1.3	Type-II remailers . . . . .	15
2.1.4	Type-III remailers . . . . .	15
2.2	Anonymity and pseudonymity systems . . . . .	15
2.2.1	Anonymous Proxy Servers . . . . .	16
2.2.2	Java Anonymous Proxy (JAP) . . . . .	17
2.2.3	Tor . . . . .	19
2.2.4	Crowds . . . . .	25
2.2.5	Freenet . . . . .	26
2.3	Encryption Tools . . . . .	27
2.3.1	TrueCrypt . . . . .	27
2.3.2	Pretty Good Privacy (PGP) . . . . .	28
<b>3</b>	<b>Taxonomy and Modeling</b>	<b>32</b>
3.1	Historical Overview . . . . .	32
3.2	Existing Classification Models . . . . .	32

---

<b>4</b>	<b>Evaluation of Privacy Enhancing Technologies</b>	<b>38</b>
4.1	The safe-surf option . . . . .	39
4.2	Test in ip-check.info with default options in web browser . . . . .	40
4.3	Test with the custom made test . . . . .	42
4.4	Test the Implementations . . . . .	43
<b>5</b>	<b>Results of Evaluation</b>	<b>46</b>
5.1	Safe-Surf Option in ip-check.info . . . . .	46
5.1.1	Low Anonymity Java/Flash activated vs. Java/Flash non activated . . . . .	46
5.1.2	Medium Anonymity Java/Flash activated vs. Java/Flash non activated . . . . .	47
5.1.3	Medium Anonymity Java/Flash activated vs. Java/Flash non activated . . . . .	47
5.2	Java Anonymous Proxy with Mozilla Firefox default user profile vs. JonDoFox user profile . . . . .	48
5.3	Tor . . . . .	49
5.3.1	Mozilla Firefox in Tor bundle . . . . .	49
5.3.2	Mozilla Firefox with default user profile . . . . .	49
5.3.3	Mozilla Firefox with JonDoFox user Profile . . . . .	49
5.4	Java Anonymous Proxy vs. Tor . . . . .	49
5.5	Results from Custom Test in anonymity systems . . . . .	49
5.6	Vulnerabilities in TrueCrypt and PGP . . . . .	51
5.6.1	Brute Force Attack . . . . .	52
5.6.2	Side Channel Attack . . . . .	52
<b>6</b>	<b>Conclusion</b>	<b>54</b>
	<b>Bibliography</b>	<b>59</b>
<b>A</b>	<b>Appendix</b>	<b>60</b>
A.1	TheTest.html – Extract Java version . . . . .	60
A.2	Thelog.php – Keep Log records . . . . .	61
A.3	Stringcomp.php – Compare the Logs and return the % of similarity . . . . .	65



## List of Abbreviations

PET	Privacy Enhancing Technologies
PII	Personally Identifiable Information
IOI	Items of Interest
DES	Data Encryption Standard
AES	Advanced Encryption Standard
EFF	Electronic Frontier Foundation
NIST	National Institute of Standards and Technology
ECB	Electronic Codebook Mode
CBC	Cipher Block Chaining
CTR	Counter mode
RSA	Rivest, Shamir, Adleman, a public-key cryptosystem
ECC	Elliptic Curve Cryptosystem
MAC	Message Authentication Code
MITM	Man in the Middle
DH	Diffie-Hellman
OR	Onion Router
JAP	Java Anonymous Proxy
AN.ON	Anonymitat Online
TCP	Transmission Control Protocol
SOCKS	Socket Secure, an Internet Protocol
TC	Tor Client
PGP	Pretty Good Privacy
HTTP	Hypertext Transfer Protocol
HTML	HyperText Markup Language
NAT	Network Address Translation
CSS	Cascading Style Sheets
SSL	Secure Sockets Layer
TLS	Transport Layer Security
FTP	File Transfer Protocol
PHP	Hypertext Preprocessor
OS	Operating System
WDE	Whole Disk Encryption



# 1

## Introduction

”The quieter you become, the more you can hear.”  
-Ram Dass

### 1.1 Motivation on Privacy Enhancing Technologies

Nowadays the vast growth of Internet has resulted on putting at risk the personal privacy. It is difficult for new users to realize that their online activity might cause them to reveal their personal information. Security issues are dangerous for the users and as a countermeasure a set of tools and technologies/mechanisms can be used to help them protect their identity. The term PETs (Privacy Enhancing Technologies) is referring to these technologies and can be defined as:

”Privacy-Enhancing Technologies is a system of ICT measures protecting informational privacy by eliminating or minimizing personal data leakage thereby preventing unnecessary or unwanted processing of personal data, without the loss of the functionality of the information system” [2].

During the design and the operation of modern computers systems, the need to protect the users’ privacy is identified. Hence, the protection of personal information, i.e. Personally Identifiable Information (PII), must be cared for. More and more applications are using personal data to provide new experience to the user. The ease of use of these applications tends to create a false impression of trust to the user. At the same time new threats are coming up continuously and increase the major problem of privacy when the user exchanges information with an on-

line environment. The comprehension of the main terminology such as anonymity, identity management and privacy will help to better understand the whole concept of the Privacy Enhancing Technologies.

## 1.2 The Goal

The main aim for this master thesis is to research and document the available PET technologies according to recent studies. The results from the documentation-analysis will be evaluated with the help of a test environment.

## 1.3 Definitions

This section will be a reference to the terminology, which is used in the report. This will make it easier for the reader to understand the basics of each topic before dive into more details.

**Privacy:** an attempt to define privacy is knotty. One of the best definitions is by Louis Brandeis and Samuel Warren in their pioneering paper on privacy during the 1890, “is freedom from interference or intrusion or else “the right to be let alone” [3, 4].

**Identity:** this term depends on the way that it is used can be either a human being of a legal person or a computer and as is it described in [5] “identity can be attached to any subject”. A better description can be “An identity is any subset of attribute values of an individual person, which sufficiently identifies this individual person within any set of persons”. So usually there is no such thing as “the identity”, but several of them [5]. It might not be so clear what these phrases mean, but it will become clearer by reading the examples and implementations in the following chapters.

**Identity management:** is when you handle identities which belong to a specific identity (person, computer, etc) and they are “usually denoted by pseudonyms” [5, 6].

**Anonymity:** according to [5, 7] the term anonymity from the users’ perspective is “The state of being not identifiable within a set of subjects“. To be anonymous depends on the point of view. Later in the report you will find an elaboration to understand the positive and the negative parts of being anonymous, and how this can be achieved.

**Unlinkability:** this aspect can be defined with help of [5, 7] “Unlinkability of a pseudonym or a subject’s actions refers to a situation where any actions or appearance of a subject on a system cannot be identified to belong to any other action of this subject” or “Unlinkability within a system means that the attacker cannot sufficiently distinguish whether several items of interest (IOI) are related or not”. In simple words it is when someone cannot connect a person with a specified action.

**Privacy-enhancing identity management:** is the application of the identity management by maintaining the unlinkability between the identities [5].

**Undetectability:** as it is referred to in [5] as “Undetectability of an item of interest (IOI) from an attacker’s perspective means that the attacker cannot sufficiently distinguish whether the IOI exists or not.” Undetectability is very important in anonymity systems.

**Unobservability:** this term is connected with the undetectability and by using [5] “Unobservability of an IOI means undetectability of the IOI with respect to all subjects uninvolved in the IOI”.

**Pseudonym:** the roots of this word are coming from Greek and means the “false name” (pseudo = false, onyma = name). By taking the definition from [5], “A pseudonym is an identifier of a subject other than one of the subject’s real names”. The users tend to use pseudonyms to hide their real identity for different reasons. In upcoming sections there is going to be an analysis of the reasons and the tools that they use to achieve that.

**Identifiability:** a simple explanation it is how an attacker can distinguish a person among others (by considering the person’s actions), or as it is described in [5] “Identifiability of a subject from an attacker’s perspective means that the attacker can sufficiently identify the subject within a set of subjects”

## 1.4 Background

This section is dedicated to the theoretical background of the privacy enhancing technologies. To start with, an analysis of the fundamental theory (cryptography) will take place and then will follow more specific theoretical parts of the privacy enhancing technologies.

### 1.4.1 Cryptography

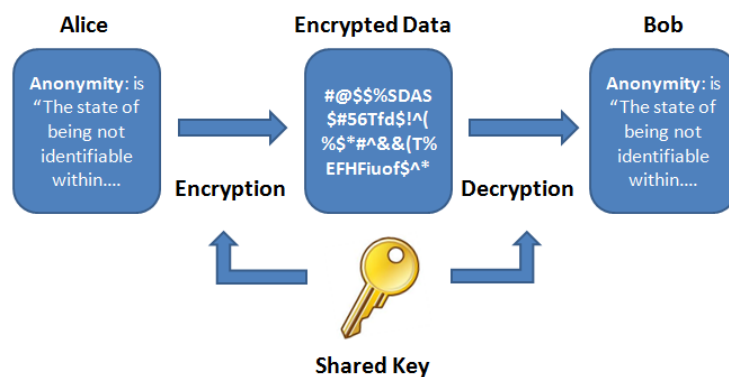
In this chapter we will analyze some basic cryptographic protocols and how they are used on the Internet. This will help the readers to understanding better the anonymity and pseudonimity systems.

#### Symmetric Key encryption

There are two facts, which can be applied to the symmetric key encryption.

1. One Symmetric Key for both the operations of encryption and decryption.
2. Block and Stream ciphers, one time pad.

The first one is straightforward. As we can see in the Figure 1.1, Alice and Bob are sharing the same secret key, which is used to encrypt and decrypt the data.



**Figure 1.1:** Symmetric Key encryption

The second needs more elaboration. Symmetric key encryption is using block ciphers and stream ciphers. But before proceed with that let's talk about the simplest and most secure: one-time pad. The idea behind the one-time pad is that, a key is used only once to encrypt a plain text. Unfortunately the simplicity of the cipher also makes it difficult to implement. More pragmatic symmetric key encryption algorithms are the block and the stream ciphers. In a quick overview of the important algorithms, you can find the following: DES, 3DES, AES and RC4, RC5, RC6.

**Data Encryption Standard (DES):** Is designed by IBM in 1975, published in 1977 and became a standard in 1979. It was the standard encryption method for 25 years. The first version of the algorithm was providing a small key length (64 bits which turned to be 56 bits) which is insecure according to the nowadays

standards. Although the flaw, because of the key size, it is worth mentioning that DES was the main standard encryption algorithm for many years. DES managed to confront cryptanalysis for all these years and still nowadays, the best practical attack is the exhaustive key search [8, 9, 10].

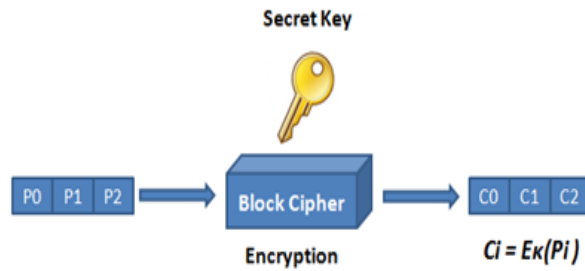
Some of the efforts for cracking DES succeeded. The first one was in 1988 with the DES Cracker (Deep Crack) by the Electronic Frontier Foundation (EFF) which managed to break it in 56 hours. The second attempt was a cooperation of Deep Crack and distributed.net in 1999-2000 where they used the CPU power of 100.000 computers and managed to break the DES in 22 hours and 15 minutes [8, 9, 10]. The next generation of the DES was the 3DES. Triple DES was offering three different key sizes 56, 112 and 168 bits, with a standard block size of 64 bits. The algorithm is based on encrypting and decrypting the plaintext in rounds [11].

**Advanced Encryption Standard (AES):** The need for a better encryption algorithm made the National Institute of Standards and Technology (NIST) to announce a competition in 1997. After four years in 2001, NIST selected the Rijndael cipher, which had been proposed by two Belgian cryptographers. The AES is based on permutations and substitutions of fixed block size of 128 bits and supports a key size of 128, 192 and 256 bits [8, 9]. Cryptanalysis of the AES since 2001 offered some ways to attack the algorithm but only in its smaller round version of AES-128. A more comprehensive study regarding AES can be found in [10, 12].

**Stream Ciphers (A5/1, RC4):** The stream ciphers are characterized by low complexity, which leads to high speed. The main idea is that they are using a random keystream which has been XORed with the plaintext to achieve encryption. As it can be obvious the security of the cipher depends on the randomness of the keystream and that is why usually the stream ciphers are weak. The most common ciphers are A5/1 and RC4 [13, 14].

**Block ciphers** have different modes of operations so that they can manage to provide better diffusion and confusion between plaintext and ciphertext [9, 10]. The first mode is called Electronic Codebook Mode (ECB) where the plaintext is divided in blocks and each block is encrypted and decrypted independently. This implementation has a lot of problems arising by the design. An attacker can search for patterns between plaintext and ciphertext because the same plaintext blocks are encrypted into the same ciphertext blocks. In Figure 1.2 the corresponding plaintext (P0-3) is encrypted block by block into the ciphertext (C0-3).

The second mode of operation in block ciphers is called Cipher Block Chaining (CBC). This mode is overriding the problems of the ECB by XORing each block

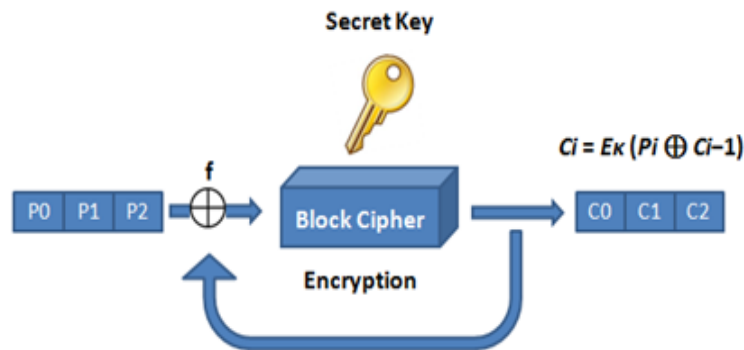


**Figure 1.2:** Symmetric Key encryption using ECB [9]

of plaintext with the previous block of ciphertext. To achieve that there is a need for an initialization vector  $C_{-1}$  on which the sender and the receiver agree in the beginning of the “conversation”. This vector will be used during the encryption of the first block [8, 9, 10]. The encryption and the decryption procedures are:

- $C_i = E_k(P_i \oplus C_{i-1}), i \geq 0$
- $P_i = D_k(C_i) \oplus C_{i-1}, i \geq 0$

Where  $C_i$ : Ciphertext,  $E_k$ : Encryption using the key k,  $P_i$ : Plaintext,  $C_{-1}$ : Initialization vector,  $D_k$ : Decryption using the key k and  $\oplus$ : Exclusive-OR [9].



**Figure 1.3:** Symmetric Key encryption using CBC [9]

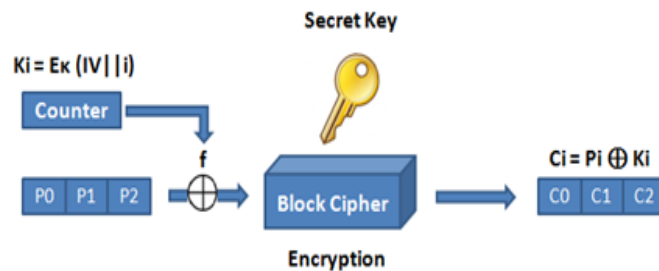
The last mode is the Counter mode (CTR). CTR mode is not one of the standard cipher modes. The operation of the cipher is based on a randomly generated stream (keystream which is like the IV in earlier modes). The keystream encrypted and then XORed with the plaintext and creates the ciphertext. The encryption procedure:

- $K_i = E_k(IV \parallel i), i \geq 0$



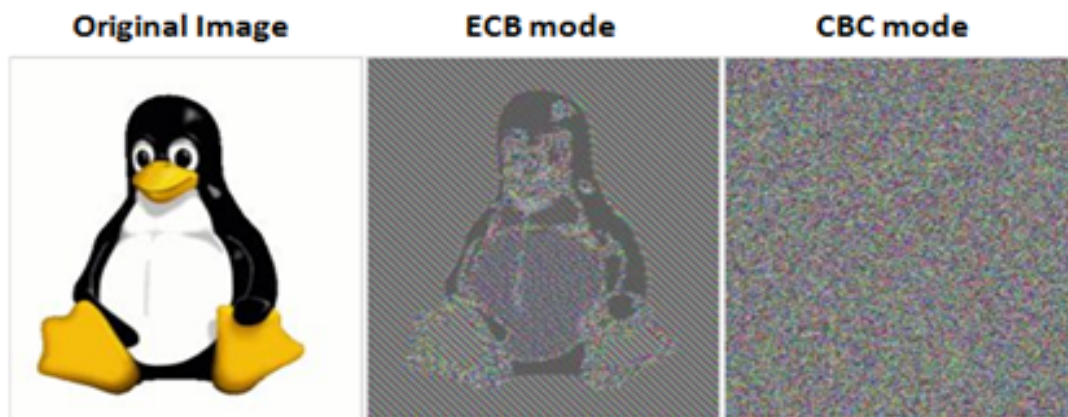
$$\bullet C_i = P_i \oplus K_i, i \geq 0$$

Where  $K_i$ : Generated keystream,  $E_k$ : Encryption using the key  $k$ ,  $P_i$ : Plaintext,  $IV$ : Initialization vector,  $C_i$ : Ciphertext,  $\parallel$ : Concatenation and  $\oplus$ : Exclusive-OR [9].



**Figure 1.4:** Symmetric Key encryption using CTR [9]

This was a quick review of some important block/stream ciphers and their modes of operation. In the next section we will continue with the asymmetric key encryption, but before continue it is better to talk about the different modes of operation in the AES encryption. The upcoming Figure 1.5 will show how the ECB mode can leave patterns between plaintext and ciphertext and how these patterns are gone in the CBC mode.



**Figure 1.5:** Comparison between ECB-CBC modes [15]

### Asymmetric Key encryption

The idea of the asymmetric encryption (public key encryption) goes back to the 1976, where Whitfield Diffie and Martin Hellman published a method for public

key agreement. The public key encryption is based on hard to solve mathematical problems, like factorization of large prime numbers. The idea is very simple, there are two keys, one to encrypt and the other to decrypt. The one is private key and it is used for decryption and the other it is public and it is used for encryption [8, 9, 10].

The drawback of this scheme is the speed because it is using large numbers. There are mainly two ciphers for public key encryption Rivest, Shamir, Adleman (RSA) and Elliptic Curve Cryptosystem (ECC) which are using key sizes of 1024-4096 bits and 163-512 bits respectively [8, 9, 10]. A user have two keys (encryption and decryption) one public and one private, but which one will make public? There are two options for different implementations [8, 9, 10].

1. Decryption key public, so that the owner can only encrypt, and someone else can verify the identity of the encrypted data (used to sign data).
2. Encryption key public, to help someone to send encrypted data and only one person can decrypt them (owner of the decryption key).

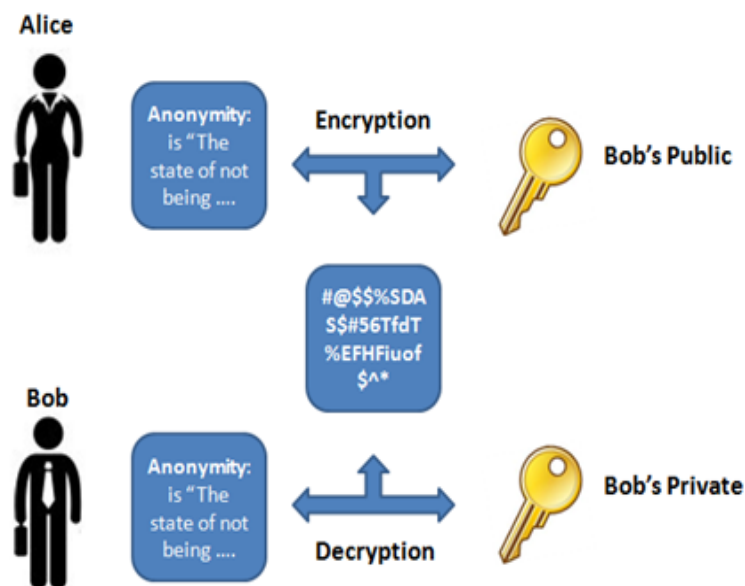


Figure 1.6: Public Key Encryption

## Hash Functions

A hash function can be defined as an algorithm which can take an arbitrary-length message as input and return a fixed-length bit string output. This algorithm has some attributes:

1. It is one-way function (infeasible to compute the data block by a given hash value)
2. Easy to compute the hash value regardless the input
3. Modification of the message always leads to a different hash value
4. Need to be collision-resistant

The cryptographic hash functions are used mainly for authentication reasons (digital signatures, MACs, etc.) but can be also useful in other implementations like fingerprinting or as a checksums [8, 9, 10]. A hash function is creating a so called “avalanche effect” between the input and the output (a minor change in the input create a major changes in output) [8, 9, 10]. Some of the popular hash functions are MD5, SHA-1, SHA-2 and SHA-3/Keccak. The MD5 (creates 128-bits string) is not considered secure nowadays and the users are advised to use SHA-1 (160-bits), SHA-2 (up to 512-bits) or SHA-3 (up to 512-bits) [8, 9, 10]. One very important attribute for a hash function is the collision resistance. This property means that it is theoretically infeasible to find two different messages ( $m_1$ ,  $m_2$ ) which are producing the same output/hash value.

$$\bullet H(m_1) \neq H(m_2)$$

Where:  $H()$ : The hash function and  $m$ : Plaintext [9]. In theory there are two forms of attacks against the hash functions:

1. Attack against the one way function by brute force. This attack will need  $O(2^n)$  attempts, which means that the security of the algorithm is n-bits.
2. Find a collision such that  $H(m_1) = H(m_2)$ .

But concerning the second one, how many tries is this going to take? Here comes the “birthday paradox” which proves that the probability to have a collision between two messages is 50% and it is close to square root of different combinations. As a real life example: “How many students need to be inside a classroom to have at least 50% probability that two of them have birthday the same day?”

$$\bullet \frac{364}{365} \times \frac{363}{365} \times \dots \times \frac{365 - (k - 1)}{365}$$

The result will be less than 0.5 if  $k = 23$  [8, 9, 10]. Hash functions cannot assure the authenticity of the message. This issue led the designers to form the keyed hashed.

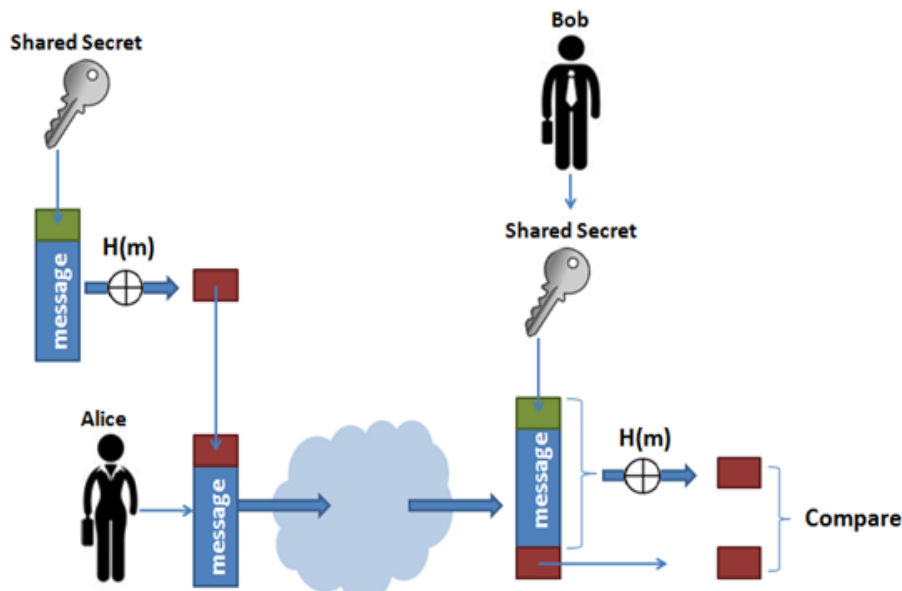


Figure 1.7: Keyed Hash function [8, 9, 10]

In Figure 1.7 Alice wants to send a message to Bob, but also to assure that the message is not going to be altered by someone (Man in The Middle attack) and finally Bob wants to assure that the originator of the message is Alice. With the above procedure the two parties can verify the integrity of the message and also authenticate Alice to Bob. The drawback of this implementation is the non-encryption of the message but on the other hand the procedure is fast. One very good example is the HMAC [8, 9, 10].

### Diffie-Hellman (D-H) Key Agreement

The need of a secure key exchange scheme drove to design a standard, which has some interesting properties. The D-H key agreement published in 1976 by Whitfield Diffie and Martin Hellman and permits two parties to exchange shared secret keys over an insecure communication channel. The principles of this method are based on the following formula:  $(g^x)^y = (g^y)^x$  [8, 9, 10].

In D-H key agreement, the calculations are made in modulo operation with large prime numbers [Figure 1.8]. The protocol description [16] needs to be followed in order to assure secure communication in a system. Nowadays a lot of implementations are using D-H key agreements (SSH, IPSec etc.) but need to

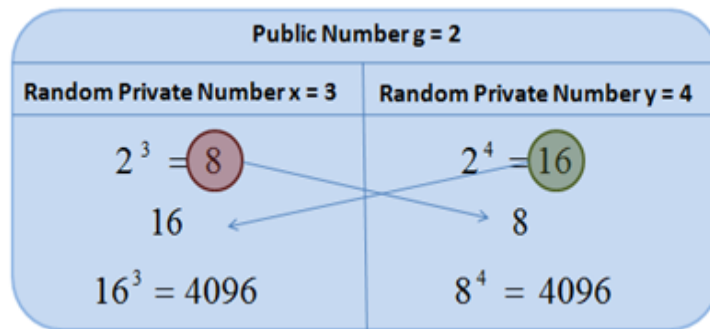


Figure 1.8: Hellman Simple Example [8]

considered that does not offer protection against some attacks and does not provide authentication [8, 9, 10].

### 1.4.2 Mix Nets

During 1981 Chaum proposed the mix net system [17]. The purpose of this system is to provide sender-receiver anonymity/unobservability. The system is using a chain of proxy servers (nodes) and each message is encrypted with public key encryption. A message is encrypted by using each node's public key. Then the message is transmitted and each node decrypts one of the encryption layers by using its own private key. Finally blend the messages and forward them to the next node [Figure 1.9].

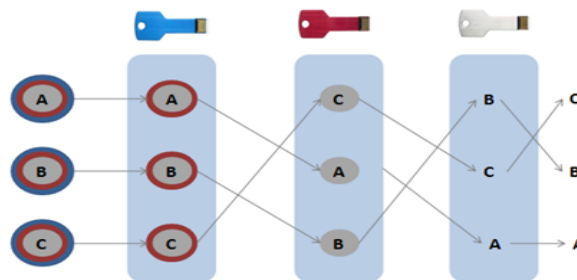


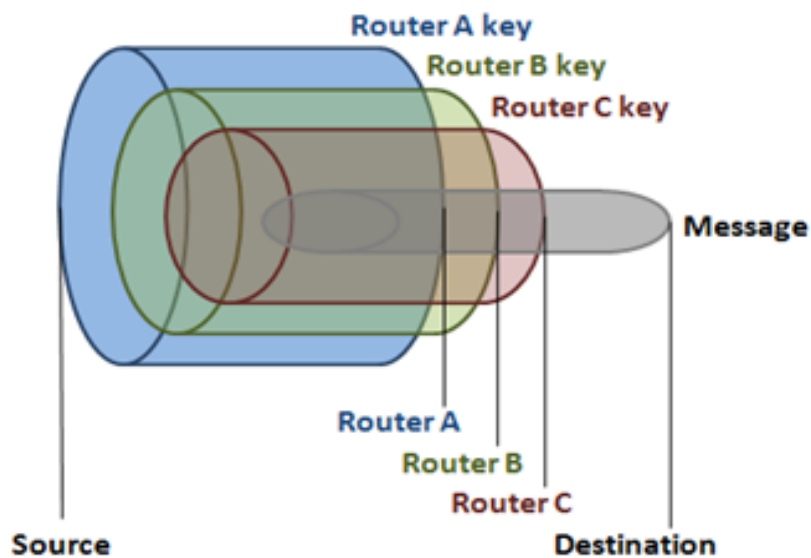
Figure 1.9: Decryption procedure in the Mix Nets [18]

### 1.4.3 Onion Routing

As an alternative to mix nets the US Naval research lab created the Onion Routing project during the 90s [19, 20]. The goal of this project was to design an

anonymous low-latency communication system. The specifications of this system include also, resistance to known attacks (traffic analysis, eavesdropping, insiders, outsiders etc).

In contrast with the previous systems (e.g. Mix nets) Onion Routing is also providing anonymity by using the principles from Chaum [17]. The advantage of this system is the unconditional trust between the nodes of the network (onion routers or ORs). Each router of the system when receives a message, re-encrypts it and forwards it to another OR so even if a router has been compromised the communication path is still safe. This encrypted message (onion) [Figure 10] has layers of encryption which are removed every time the message changes OR [7]. The system does not offers end to end encryption between the sender and the receiver because the last OR, (exit node) can access the unencrypted message (depends if the message is encrypted with end to end encryption scheme p.x. SSL) [7].



**Figure 1.10:** An onion [21]

To sum up, an OR is a network on top of internet (overlay), which is using layered encryption technology. If someone observes the network, knows only that there is a communication which is taking place [7].

# 2

## Review of Existing Implementations

**T**HIS chapter covers some important PET implementations. This review will help to classify the existing implementations in categories.

### 2.1 Anonymity and pseudonymity tools for emails

The first systems which will be examined are for electronic mail anonymity and pseudonymity. The idea belongs to the Chaum, who described the concept of mix networks in his paper in 1981. The main function of these systems is to hide the sender's identity, by providing not only anonymity but also undetectability. It is also possible to provide pseudonymity, which means that it can create a false identity for someone, so that she can send and receive emails without compromising her real identity. These systems belong to the high-latency tools. [22].

#### 2.1.1 Type-0 remailers

The type-0 remailer or pseudonymous remailer has a very simple implementation. The system removes the email address of the sender and assigns a pseudonym to her. Then it forwards the email to the receiver. The drawback of this system is that keeps a list/table with the matches between pseudonyms-real emails. This list can be a single point failure for the system and if someone takes over this list the anonymity of the system is collapsing [22].

### 2.1.2 Type-I remailers

The type-I remailers are also known as Cypherpunk anonymous remailers. To understand this implementation I will give an example.

Alice wants to send an email to Bob. She is sending the email to the type-I remailer, which removes all her personal information and then forwards the email. Now comes the good part. Instead of sending it directly to the receiver the first type-I remailer forwards the email to a second type-I remailer and so on. Finally the last remailer is delivering the email to the recipient. By compromising any point in this system an attacker cannot identify who is communicating with whom.

Apart from this improvement, the type-I remailer introduces an encryption scheme, where each remailer in the chain has the possibility to decrypt the email but the only data that it has access to is the address of the next remailer in the chain. Only the final remailer has access to the unencrypted text and the address of the recipient [Figure 1.9].

The last improvement that was introduced in type-I remailer is called mixing. When a remailer is receiving a lot of emails, it does not forward them with a specific pattern (FIFO, LIFO etc) but instead of that it is rearranging all the emails and sends them in a random order [22].

### 2.1.3 Type-II remailers

The type-I remailer introduced some improvements, but there were still some vulnerabilities, which it was not able to address (size correlation attacks, replay attacks). The Mixmaster (Type-II remailer) was addressing issues regarding the aforementioned attacks. The drawback of the type-II remailer was the need of a special software [23, 24].

### 2.1.4 Type-III remailers

The last remailer, it is the type-III and its standard implementation, the Mixminion. The new type-III remailer is providing some improvements compared to the previous implementations (replay prevention, forward anonymity). The type-III remailer is not so popular so far and this is one of its largest drawbacks [22, 24].

## 2.2 Anonymity and pseudonymity systems

In this section we will describe systems which can be used to provide anonymity, pseudonymity and unlinkability. The analysis will focus only on interactive (low-



latency) systems because they are becoming impractical as far as the latency increases.

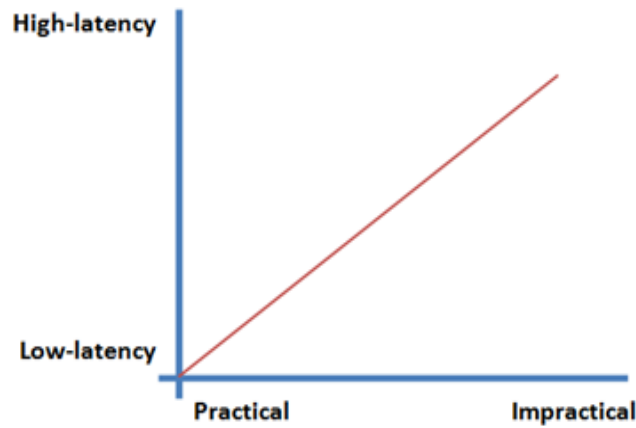


Figure 2.1: High/Low Latency vs. Practicability

### 2.2.1 Anonymous Proxy Servers

The simpler implementation, which provides online anonymity is the anonymous web proxy server, which can also be referred to as a forward proxy. The implementation is simple, a user requests from a proxy server a target and the server retrieves this information and connects the user to the target.

The anonymity part is taking place when the user connects to the proxy server. The proxy is sending the requests on behalf of the user. As a result the target believes that it is communicating with the proxy server but instead of that it is communicating with the user.

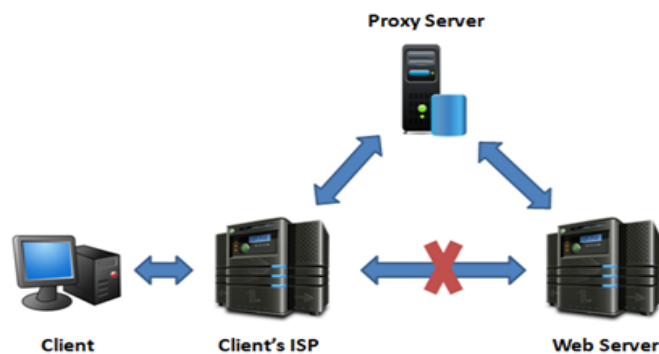


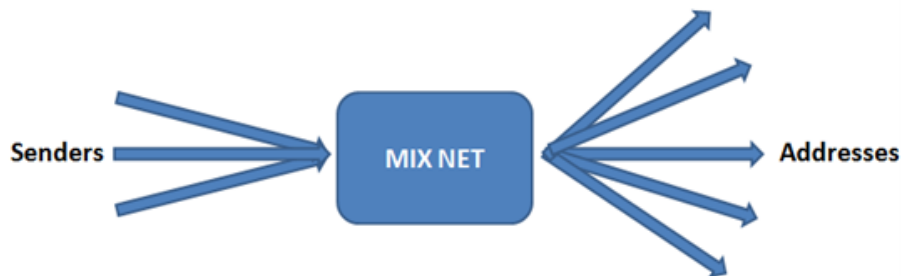
Figure 2.2: Proxy Server Connection [25]

There are a lot free anonymous web proxy servers/providers online. Some examples are Anonymouse, KProxy, WebProxy.ca etc. This list is very big but as we will analyze later the web proxy servers are providing limited protection [25].

### 2.2.2 Java Anonymous Proxy (JAP)

The University of Dresden around 2001 was working in an anonymity project (AN.ON, Anonymitat Online). The JAP is the client made for AN.ON project (is also known as JonDo) and it is a web based tool that is using the type-II remailer techniques [26].

The implementation of the system is based on the mix nets. The user is connected with a web server by using several nodes which are called mixes. When a message enters a mix blends with messages from other users in order to achieve unobservability and avoid traffic analysis. If someone observes this mix from outside or inside, cannot determine which data traffic belongs to whom [27]. A way to group various mixes is into "Mix Cascades" or "Cascades" where each cascade has a specific path through the network [26, 28].



**Figure 2.3:** Users inside one Mix [29]

In Figure 2.4 is presented the architecture of the network. The JAP is the client software which installed in the user's computer and in combination with the web browser offers anonymization. The Info Service provides information regarding the mixes (traffic load, online users, available cascades etc) and finally the Cache Proxy optimizes the connection between the mixes and the internet (re-using frequently visited sites, reduce bandwidth, improve response times ) [27, 30].

The JAP client connected with the info service and retrieves a list of the available cascades. Then the user chooses the desired cascade and the JAP client connects the user in this cascade. Finally, the JAP client checks the network load and other information, related to the connected cascade [27].

Thereafter JAP is creating a connection with the first mix in the cascade. The user's data traffic gets encrypted and send to the first mix in the cascade. The mix blends this data traffic, with data traffics from all the other users who are connected in the same mix. Then the first mix forwards the data to the second mix and goes forth. When the data reach the final mix, get decrypted and delivered to the final destination through the cache proxy [27].

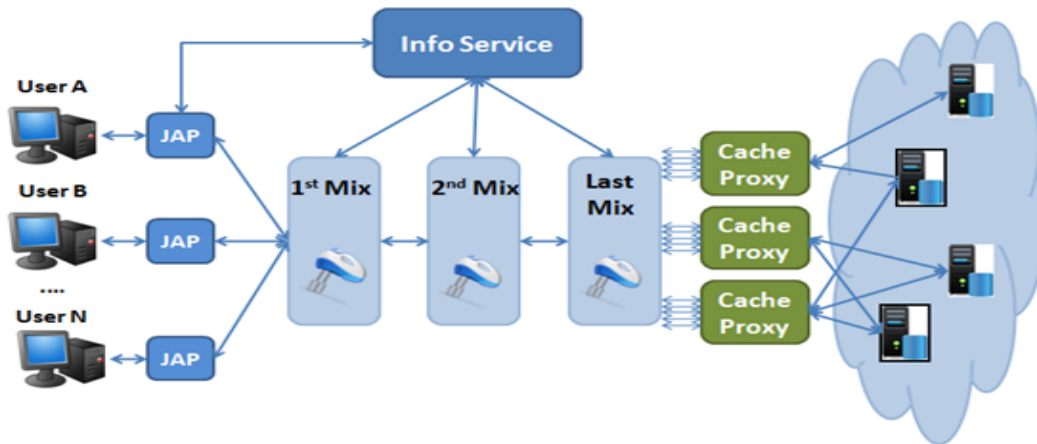


Figure 2.4: JAP Architecture [27]

One important part of JAP is the encryption of the data traffic. The encryption is in respect with the cascade which will be used. Every layer of encryption is created according to the mix which will follow [Figure 1.9]. If an eavesdropper intercepts messages is unable to read them and cannot conclude about the sender's identity [27]. The encryption method is a hybrid form. Symmetric encryption (RSA 1024+ bit key length) is used for key encryption (between the mixes) and asymmetric encryption (AES 128 bit key length) for the data traffic for better efficiency [27]. The whole communication is bi-directional and the JAP client is the creator of the symmetric encryption keys which are going to be used to re-encrypt the data traffic which is coming from the web server. The final step is the decryption of the data by the JAP client which owns the decryption keys [27]. As it is clear the encryption is taking place only inside the cascade which means, when the traffic leaves the last mix, if the user does not provide alternative encryption method (i.e. SSL) the data are unencrypted and everyone can have acces to them [27]. The AN.ON project is offering a Mozilla Firefox profile (JonDoFox) which is a special user profile with privacy orientation. This user profile has already installed plug-ins to:

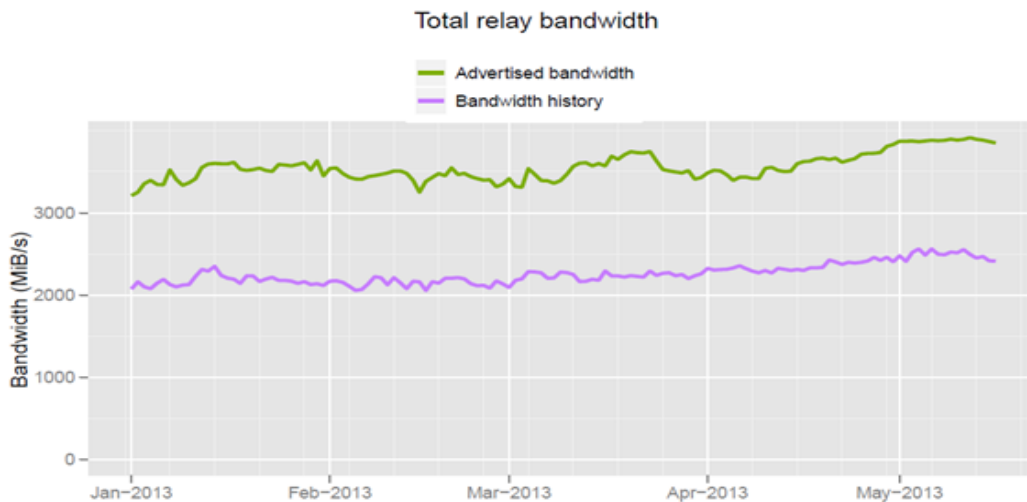
- Avoid execution of Flash in web pages
- Avoid execution of Scripts in web pages

- Force websites to use https connection
- Cookie management
- Profile switcher to change between JAP and Tor
- Ad Blocker

Also has activated an option to delete the history when the web browser is turn off. As a part of the AN.ON project is highly recommended to use JonDoFox in combination with JAP to minimize any possible leak of information [27].

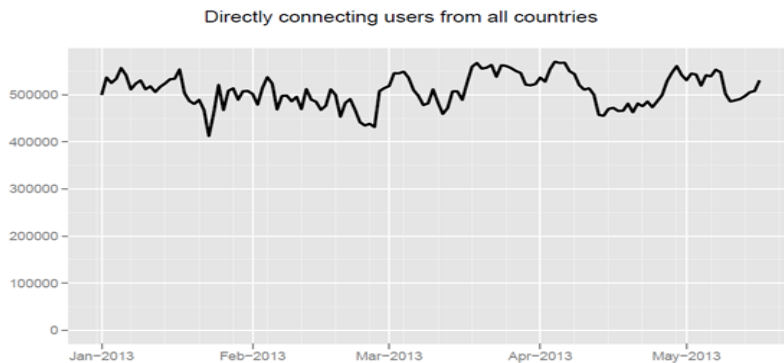
### 2.2.3 Tor

A newer generation of anonymity network appeared in 2002 and it was the Tor [31, 32]. Tor is based on the onion routing technology and it is the most popular project nowadays with 3-4 Gbits of traffic [Figure 2.5] and over 500.0000 users [Figure 2.6] every day. Originally it has been developed by the US Naval Research Lab and its privacy orientation was suitable for governmental communications.



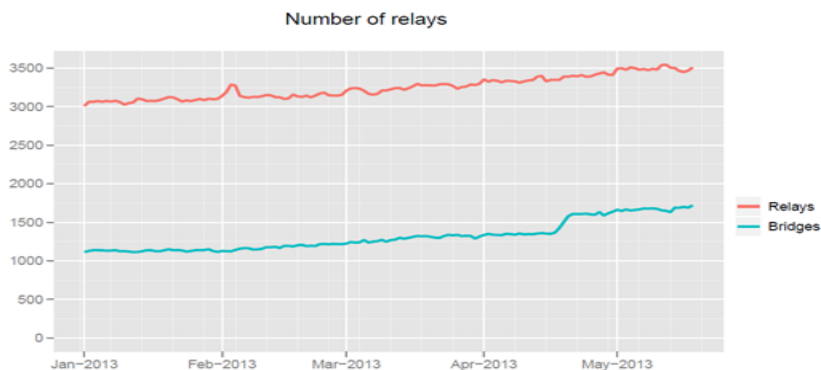
**Figure 2.5:** Tor BW from relays [33]

The project is based on volunteers who are maintaining Tor nodes by offering their bandwidth. In Tor network a node can be either a “Tor relays” or “Bridge relays”. If a user is a bridge relay in contrast with the Tor relay, its address is not listed in the public Tor directory. This helps users, whose access in Tor networks is blocked (ISP, governments etc.) to regain access by using the relays which cannot be detected (bridges) [31, 32].



**Figure 2.6:** Users in Tor [33]

Tor project mentions: “If you can offer a lot of bandwidth be a normal relay, if can offer just a little bit of bandwidth be a bridge” [32]. Another way to classify the users inside the Tor network is, if they are exit relays. As can be seen in the Figure 2.8 the node in the red circle is an exit relay, and it is the last node at the edge of Tor network [32].



**Figure 2.7:** Relays and Bridges in Tor Network [33]

Tor is a distributed anonymous network, where each user establishes a TCP connection (by using SOCKS) with some other nodes inside the network. Then she uses these intermediate step stones (Tor nodes) to reach the desired final destination [32].

There is a basic problem during a TCP communication between two points. Even if the data are encrypted and an eavesdropper cannot decrypt them, she can still gather useful information about the two parties which are communicating. This information is kept in the headers of the packets and can reveal information

regarding source address, destination address, timing and size of the data [32].



**Figure 2.8:** Using TOR [32]

In Figure 2.8 are the steps that are taking place to communicate 2 parties. Alice first communicates with Dave (directory server) and receives a list with the Tor relays. Then Tor client selects a path (circuit) through the Tor network randomly and sends the data through the network. The trick in the whole procedure is that the circuit has been created one hop at a time, which means that each intermediate relay knows only the address of the previous relay and of the next relay. To achieve even higher privacy the client negotiates the encryption keys for every hop separately. If an attacker manages to compromise one relay it is not impossible to correlate the data traffic with a source and a destination address [32]. Every specific time frame Tor forces the user to change circuit so that an eavesdropper cannot create a link between old and new connections [32].

**The Design of Tor:** The communication between the ORs and the Tor client (TC) is carried out by exchanging packets, which are called “cells”. The exchanged data are protected as far as the whole procedure is taking place over encrypted TLS tunnels, which are created between the desired nodes [32]. The cells have a fixed size of 512 bytes and they are divided in two parts, the header and the data part (payload). The header contains information regarding the circuit ID, Stream ID (if we have multiple streams per circuit) and other useful information (i.e. creates or destroys a circuit etc.) [32]. The TC creates a circuit which can be shared among many TCP streams at a time. To avoid delays due to “one hop at the time” construction, each TC creates new circuits periodically on the background. The reason is to avoid traffic analysis and the users are forced to change over a

new circuit once a minute [32]. The next step is the explanation of the encrypted communication between OR-OR or TC-OR. We can visualize the communication by using the Figure 2.9 [32].

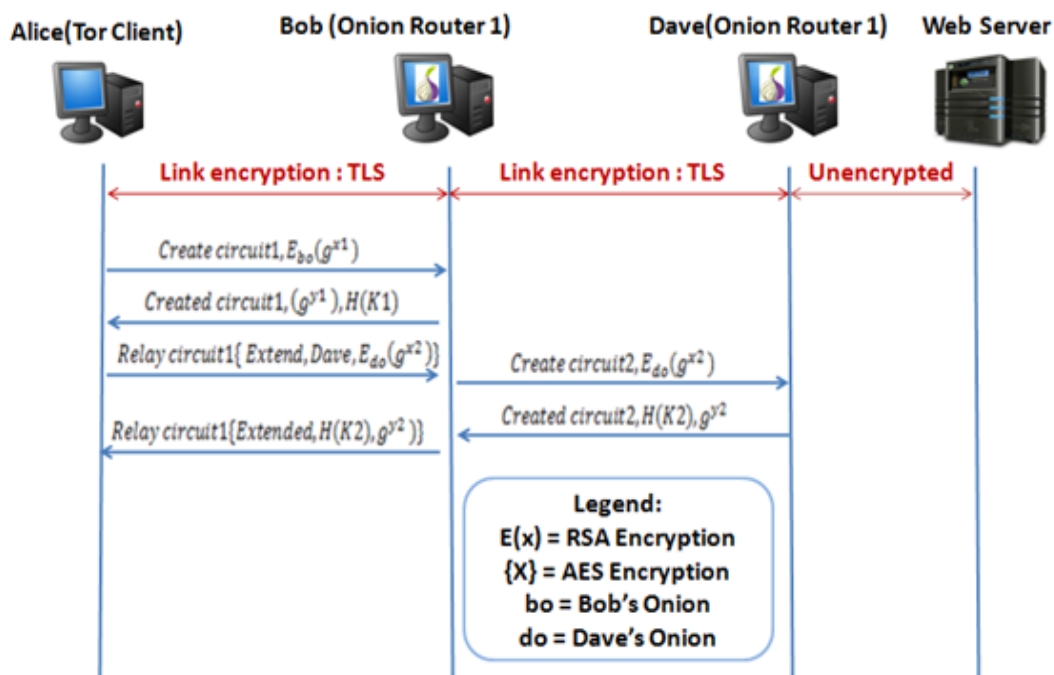


Figure 2.9: Tor Communication [32]

Alice starts the procedure by sending a “create circuit” cell to Bob. Then she is choosing a circuit ID and inside the data part of the cell enters her part of the Diffie-Hellman (D-H) handshake ( $g^{x1}$ ) which is encrypted with the onion key of Bob. Afterwards, Bob answers back with a “created circuit” cell which includes his part of D-H handshake ( $g^{y1}$ ) plus the hash of the key ( $K_1 = g^{x1} * y_1$ ) [32].

By the end of this step Alice can communicate with the first OR. Then she extends the circuit further than Bob she sends an “extend” cell to Bob by specifying him the address of the next OR and the first half of the D-H for the next OR ( $g^{x2}$ ), in our case Dave. Bob now is responsible to forwards the handshake to Dave. Then Dave replies back to Bob with a “created circuit”, Bob insert this info into an extended cell and sends this cell back to Alice [32].

If Alice wants to extent further the circuit needs to continue by taking the same steps. At the end of the communication Alice terminates the circuit by sending a “destroy” cell and close all streams [32].

Another interesting feature of Tor is the hidden services. A user of Tor can

setup a web server which has a name (like in DNS system) and it is possible to be resolved only through the Tor network [32, 34].

When a user (Bob in our case) wants to offer a hidden service chooses some Tor relays and creates circuits to them. Then he asks them to be “Introduction points” and he shares hidden service’s public key with them. Tor network is providing the appropriate anonymity to Bob [32].

The next step for Bob is to help other users to reach his hidden service. He is creating a “hidden service descriptor” which contains the introduction points, the public key and also signs the descriptor with his private key (verification of his identity). Then Bob makes the descriptor public by uploading to a “distributed hash table”. The format of the descriptor is “ABCD.onion” [32].

When a user, (Alice) wants to communicate with the hidden service, she is learning (by using any sources) the ABCD.onion address and then, downloads the descriptor from the “distributed hash table”. Now she has all the appropriate information about the service (introduction points and public key). At the same time Alice communicates with some other Tor relay and she asks if it can be a “rendezvous point” (red circle in Figure 2.10). She also sends to this relay an “one time secret” [32]. These were the first 4 steps in the hidden service creation.

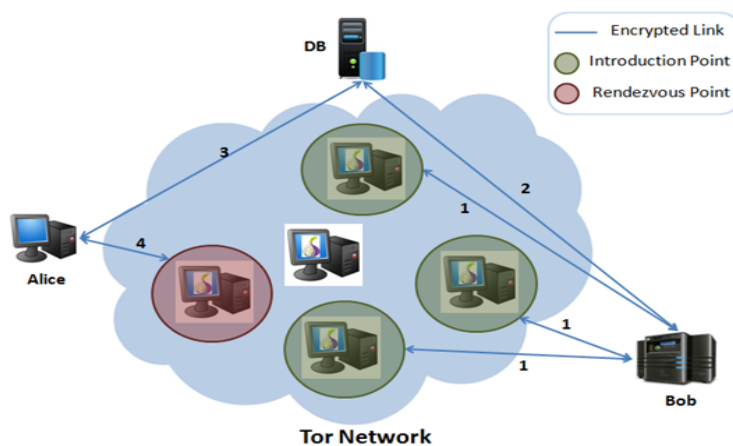


Figure 2.10: Hidden Service Steps 1-4 [32]

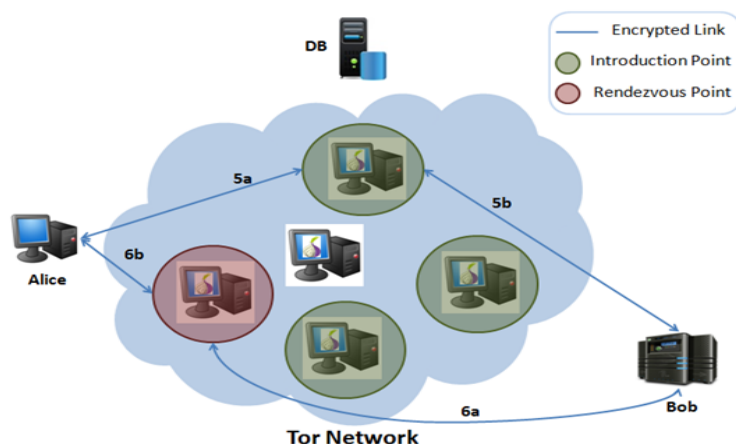
When Alice has a “rendezvous point”, she creates an “introduce message” that embodies the “rendezvous point’s” address and the “one time secret”. She sends this message to an introduction points and she is asking this point to deliver the “introduce message” to Bob’s hidden service. The “introduce message” is encrypted



with the provided public key [32].

The hidden service receives the “introduce message”, decrypts it and recovers the “rendezvous point” and the “one time secret”. The communication between the “rendezvous point” and the hidden service is established by sending a “rendezvous message” [32]. This was the last step regarding the establishment of the circuit, now Alice is able to communicate with the hidden service through the “rendezvous point” which relays the traffic from Alice to Bob [32] [Figure 2.11].

One key feature in the Tor Network (and therefore to hidden services) is the “entry guards”. When a user creates a circuit is using only few entry relays (the first node in the Tor Network). The reason for this action derives by statistics. Tor Network is based on the fact that an attacker cannot control the whole part of the network which includes among other an entry and an exit point. These two points are crucial, because if someone manages to control these, can correlate data traffic (traffic analysis) with a user. Let’s suppose that the whole Tor network has  $N$  relays, and an attacker can control/observe  $E$  nodes, if a user changes the exit and the entry point every time that she is using the Tor network then the attacker has  $(E/N)^2$  probability to correlate traffic with a user [32].



**Figure 2.11:** Hidden Service Steps 5-6 [32]

By using some fixed entry relays the attacker has lower probabilities of succeed in such an attack, even if she can observe some entry points the possibilities are lower than before [32]. More information can be found in related studies [35, 36, 37].

Tor is only working with TCP streams and with SOCKS supported applications. By using Tor with applications that are not supposed to cooperate suc-

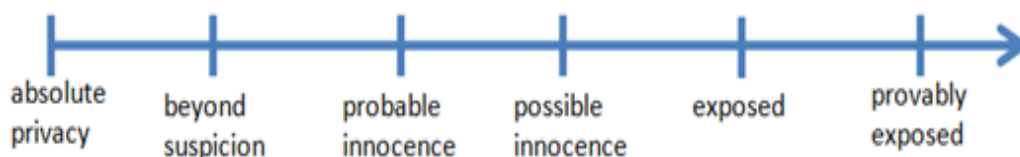
cessfully with Tor can lead to leak the real IP of the user. More information and extended documentation can be found in Tor's project official webpage [32].

### 2.2.4 Crowds

Crowds is an anonymous network, which according to its authors can provide anonymity by "blending the user into the crowd". The idea follows the basics of the other anonymous networks [38].

A user first is accessing a network (crowd) by using a process which is called "jondo". The next step is to ask to grand access to the crowd by using a server which is called "blend". Then the user sends a request to a web server, but instead of a direct communication, the user forwards this request to another random user inside the crowd. This second user decides independently if she will forwards the message to the final destination or she will keep forwarding the request through the crowd [39].

The authors of crowd introduce the notion "degrees of anonymity" and also define the points between no anonymity and anonymity [38].



**Figure 2.12:** Degrees of Anonymity [38]

The new feature, which introduced in this anonymous network in comparison with previous implementations, was the totally random selection of the next node in the forwarding procedure. Each user inside the crowd network is taking the decision to forwards to another node or not by flipping a biased coin (each user decides to forward or not, independently to the prior result of the coin with probability  $p_f > 1/2$ ). The random construction of an anonymizing tunnel though random selected nodes helps to achieve maximum anonymity [38].

All the anonymous networks have flaws and Crowds is not an exception. Although Crowds is secured against a local attacker/observer, there are vulnerabilities on the members of the network regarding predecessor attacks [39].The Crowds is considered to be vulnerable in a global attacker/observer [38].

### 2.2.5 Freenet

During 2000 Ian Clarke created "a distributed anonymous information storage and retrieval system" aka the Freenet [40]. The Freenet is a peer-to-peer network used for anonymous distributed data storage.

The idea is simple; a user joins the network and provides a small space of her hard disk drive. This part gets encrypted and now she can share and lookup information to and from other users.

The approach of Freenet is different than in the classic peer-to-peer systems. The integrated proxy of the Freenet software has been used to gain access to the content, which is published within the Freenet (freesites). A comparison with a similar approach can be the Tor hidden services [40].

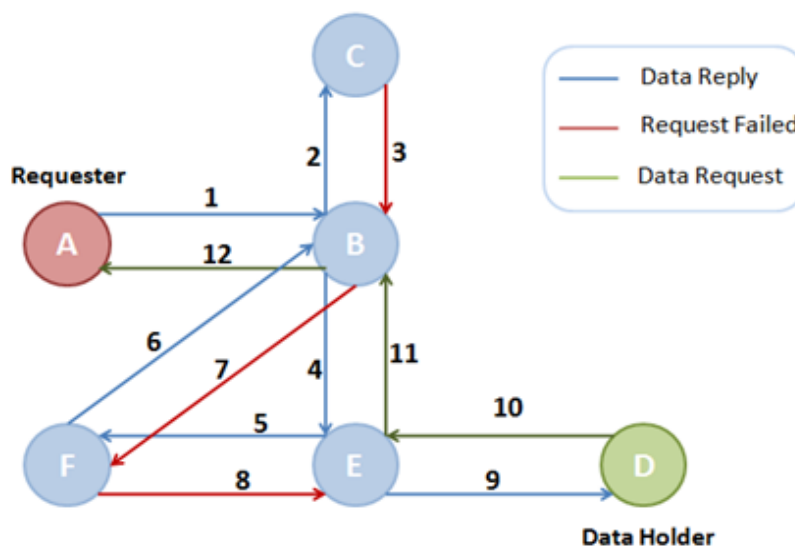


Figure 2.13: Request Sequence in Freenet [40]

When a user shares something with the network (upload), the data fragmented in small parts and shared among other users. The user who is uploading the data does not need to be online all the time so that someone else can download them. If a user wants to download the data, the network retrieves all the fragmented parts and reassembles them [40].

The network is anonymous and if a user wants to access some data, does not connect directly to users who have some of the pieces but she is rerouting inside the network through many nodes. The final result is that none of the intermediate nodes know who made the request and who owns the data [40].

As it is clear the bandwidth restriction is the drawback of the Freenet because to reroute a request across many nodes inside the network is costly, thus it will take more time to complete the transfer [40]. The last big update during 2008 included the two modes of operation which are connected with the levels of security OpenNet and DarkNet. The two implementations are straightforward. In OpenNet approach, a user connects with random users of the network and in DarkNet approach the user connects only with some trusted parties (friends). Furthermore she is also exchanging keys to increase the anonymity and the unobservability [40].

Freenet is an overlay network and uses the current infrastructure (Internet). All the nodes of Freenet can be possible neighbors, as far one node can access directly some of them and there is no hierarchy. To exchange data, a node (A) forwards data packets to another node (B) without considering if B will continue forwarding the data (when i.e. there is a request). This property is similar with the one in the crowds [40].

Freenet is used by third-party software, which are taking advantage of its architecture. Some examples are Infocalypse (code sharing), FlogHelper (blogging), Sone (social network) and Web of Trust [40].

## 2.3 Encryption Tools

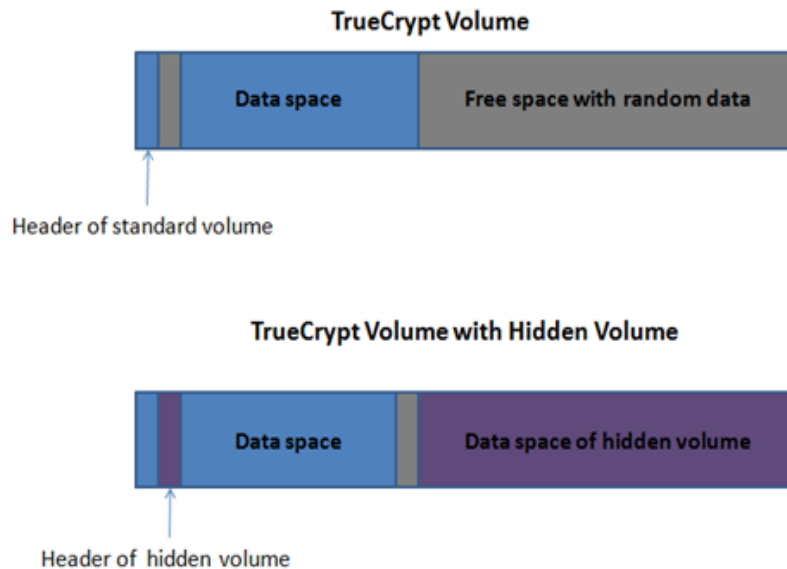
Previously were covered anonymity implementations. It is time to continue by registering some important implementations for encrypting files/disks and emails.

### 2.3.1 TrueCrypt

TrueCrypt is real time disk encryption software, which is provided under the open source TrueCrypt license. TrueCrypt is used to create virtual encrypted disks, to encrypt partitions or even a whole hard disk drive and can be used under multiple platforms (Microsoft Windows, LinuxOS and MacOS). The software uses the AES, Serpent and Twofish encryption algorithm in XTS mode. Also uses RIPEMD-160, SHA-512 and Whirlpool hash functions [41].

It is considered one of the best open source encryption tools and it supports the concept called “plausible deniability” in two ways:

1. The user can create a hidden volume or a hidden operating system [Figure 2.14], so even if she is forced to reveal her password for the first drive, the intruder cannot prove that there is another volume inside the first one. This is based on the TrueCrypt implementation where, the free space in a volume is always filled with random data. The user can also install a hidden operating system, inside a TrueCrypt hidden volume [41].



**Figure 2.14:** Hidden Volume in TrueCrypt [41]

2. The second important part of “plausible deniability” is that, until the decryption of a volume, the space seems to be filled with random data. These data cannot be recognized as been TrueCrypt encrypted.

TrueCrypt is very popular but has also vulnerabilities (Hibernation file, Memory dump files, paging files etc.). TrueCrypt Foundation proposes a list with some security requirements and precautions to avoid data leakage [41].

Despite the fact that it is so popular there are some rumors around the TrueCrypt. It is said; that there is a possible backdoor intentionally left, but there is not any real case scenario or proof of these accusations ‘til now [42].

### 2.3.2 Pretty Good Privacy (PGP)

PGP is one of the first data encryption/decryption software which was released during the early 90s by Phil Zimmermann. The first version of the software released in 1991. Later on, during 1996, Zimmerman and his team merged with Viacrypt, which later renamed as PGP Incorporated. Nowadays, PGP belongs to the Symantec Corporation [43, 44].

By taking a look into the technical details of PGP, we can distinguish that it is a hybrid form of encryption (symmetric and asymmetric). To give an example,

if Alice wants to encrypt data by using PGP, the first step is to compress them (saves disk space and creates obfuscation). Then she creates a session key, which is a form of one-time key. The session key has been created by using entropy coming from the user (random mouse movements, mouse clicks etc.). The session key is used to encrypt the data (symmetric encryption algorithm). Finally the session key gets encrypted with an asymmetric encryption algorithm (public key encryption) and transmitted to the recipient [43, 44].

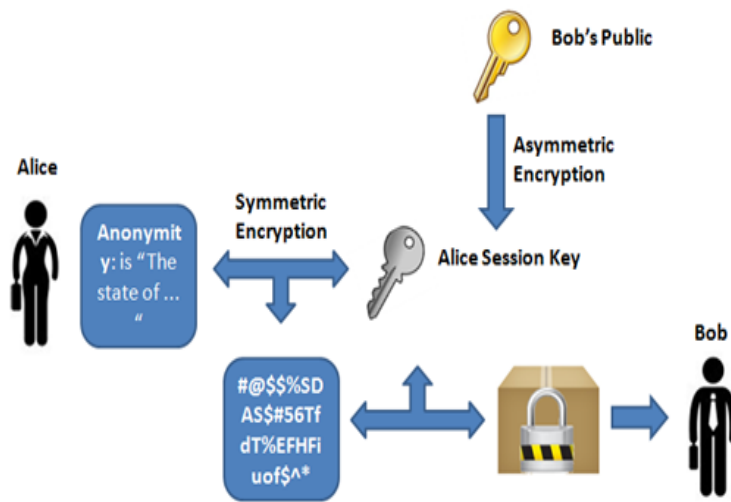


Figure 2.15: PGP Encryption [45]

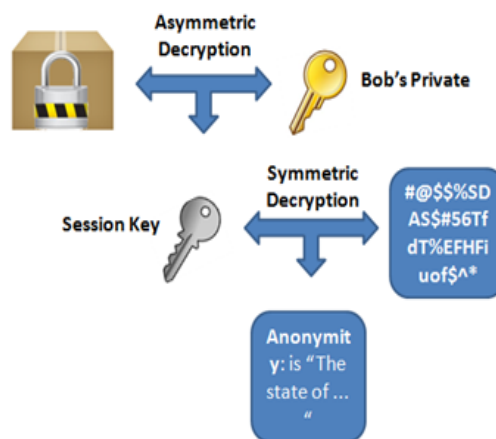


Figure 2.16: PGP Decryption [45]

The decryption procedure is quite simple because it is the reverse of encryption. The recipient uses his private key to decrypt the session key and when he

recovers it, he uses the session key to decrypt the data. The reason of using a hybrid encryption scheme is to increase the performance without sacrificing the security [43, 44].

The combination of the symmetric/asymmetric encryption can offer high level of confidentiality but also supports authentication and integrity by using digital signatures (hash operation) [43, 44].

An alternative to PGP is the GNU Privacy Guard which is in compliance with the OpenPGP Internet Standard. The design group is the Free Software Foundation and like the PGP, is used for encryption and for signing data [45]. Some other examples are the Enigmail which is a plug-in for Mozilla Thunderbird (email encryption and signing) and Gpg4Win (compatible with Microsoft WindowsOS) for also file and email encryption (both Enigmail and Gpg4Win support OpenPGP).





# 3

## Taxonomy and Modeling

**I**N this chapter will be covered the existing classification and taxonomy models. There will be a discussion if these models need to be adjusted.

### 3.1 Historical Overview

The story begins in 1981, when David Chaum introduced the world to how to communicate while considering anonymity and unobservability. His idea does not need a centralized authority and two entities can communicate anonymously and without been traced [7, 17]. The methodology that he describes was called “Mix”. This concept was a breakthrough in 1981 and became a reality in early 2000s by an implementation called the Mixmaster, which is an anonymous remailer. More information about this is presented in the previous chapter [7].

The next step came with the publication of [46], where Dutch and Canadian authorities, were trying to prove that it is possible to use the recent technology to protect the confidentiality and integrity of the personal information. It is worth mentioning that the handbook in [47] was an important step in evolution of PETs because it was not written by researchers or technicians but by the authorities [46, 47].

The last 15 years the privacy challenges became part our everyday life and the research community put effort into research programs for this scope [7].

### 3.2 Existing Classification Models

This chapter will cover classification models, which are proposed by researchers and by the community.

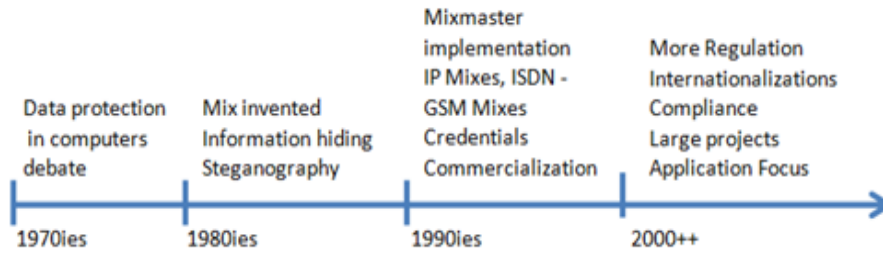


Figure 3.1: History of the PETs [7]

Privacy has many interpretations and it is difficult to restrict the definition of what is privacy. It is also important to have a taxonomy system which covers all the different privacy issues. In 2006 Solove [48], tried to "to identify and understand the different kinds of socially recognized privacy violations", by dividing the information privacy in four challenges [7, 48] :

1. Information Collection
2. Information Processing
3. Information Dissemination
4. Invasion of Privacy

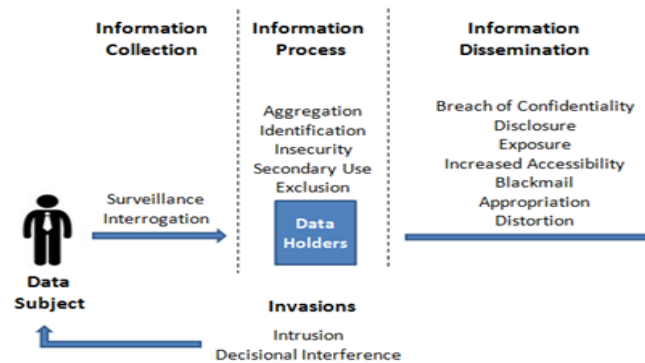


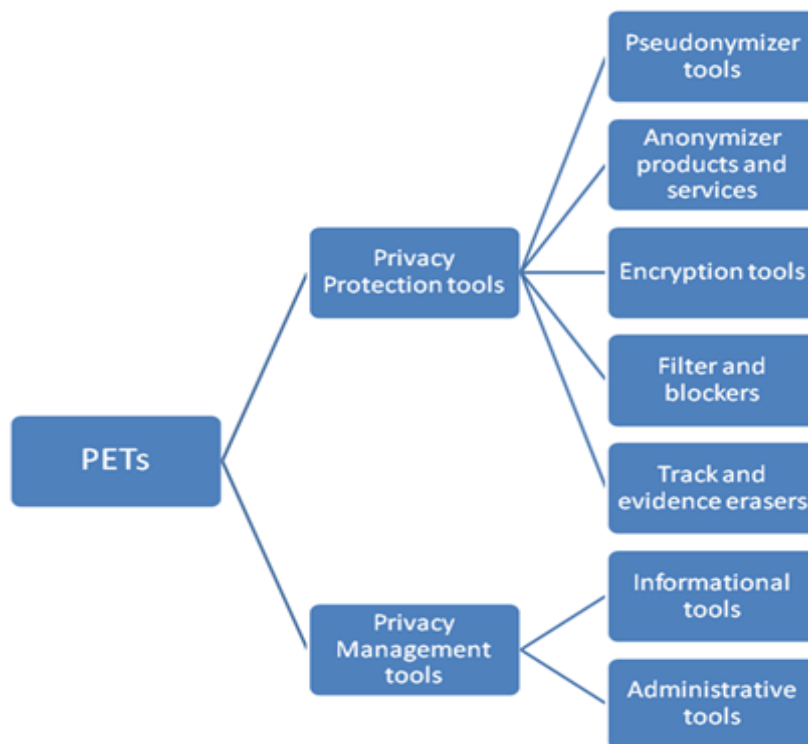
Figure 3.2: Taxonomy of Privacy [7]

In first, second and third challenge the terms collection, processing and dissemination are the acts regarding personal information by a third party (person or organization), while in fourth the author it is clarifying that the "invasion" is the intrusion of personal space and can possible influence the decisions. Solove makes

a detailed analysis in these challenges in his paper [7].

Studies during the last years, helped to categorize the PETs. By using [7] and FIDIS project (FIDIS 2003/2007) PETs can be separated as Opacity tools or Transparency tools. Opacity tools are trying to anonymize the user's identity by diminishing the visibility of the personal information. Transparency tools are using methods to elevate the understanding of how the PII are processed by a program.

Later in 2005 the Danish Government [49] released a study which separates the PETs in Privacy Protection tools or Privacy Management tools. In continuation to the previous model, privacy protection tools are the equivalent with opacity tools and privacy management with transparency tools, In Figure 3.3 these two big categories are divided in subcategories [7, 49].



**Figure 3.3:** Privacy Protection Classification [7, 49]

According to studies [49], PETs need to cover three main features anonymity, unobservability and unlinkability. By reviewing the terminology in previous chapters we came across the undetectability, which also needs to be included.

Anonymity networks among others are used to help people who live in Internet censored countries. When someone wants to surf anonymously a user can utilize an anonymous network (e.g. Tor) and no one will bother her (i.e governmental authority, e.t.c). But there are also exceptions, where even the suspicion that someone is using an anonymous network can cause her troubles. This is the main reason why the undetectability needs to be introduced as one of the main features.

To give an example, Tor network is using obfuscations techniques to masquerade its traffic with the so called obfuscation bridges, which have some special plug-ins, the pluggable transports [32].

The upcoming Figure 3.4 includes the tools that have been analyzed in previous chapter and will exhibit their main features.

Implementations	Features			
	Anonymity	Unobservability	Unlinkability	Undetectability
Type-0 Remailer				
Type-I Remailer				
Type-II Remailer				
Type-III Remailer				
Anonymous Proxy				
Tor				
JAP				
Crowds				
Freenet				
True Crypt				
PGP				

	Include Feature
	Under Conditions

**Figure 3.4:** PETs and features [49]

The undetectability in Tor and JAP were set as a feature “under conditions”. To preserve online privacy the use of an anonymous network is not enough. The users need to be cautious and also to follow certain standards to maintain a high level of anonymity.

Privacy Enhancing Technologies					
Privacy Protection Tools				Privacy Management Tools	
Pseudonymizer	Anonymizer	Encryption	Filters and Blockers	Informational	Administrative
Type-0 Remailer	Anonymous Proxy	Freenet	-	-	-
Type-I Remailer	Tor	True Crypt	-	-	-
Type-II Remailer	JAP	PGP	-	-	-
Type-III Remailer	Crowds	-	-	-	-

**Figure 3.5:** Categories of Implementations

The classification of the implementations which was reviewed in previous chapter as Privacy Protection or Privacy Management tools can be found in the above Table [Figure 3.5].



# 4

## Evaluation of Privacy Enhancing Technologies

**I**N previous chapters, implementations which are offering a high level of privacy were reviewed. The next step is to find a common way to evaluate them.

The first evaluation will be conducted to the anonymity and pseudonymity systems (Proxy servers, Tor and JAP). The web applications, which will be used are the ip-check.info that belongs to the JonDonym (Private and Secure Web Surfing) project and also a custom made test, which is created for the scope of this project work [27].

The ip-check.info is using a series of anonymity tests and presents the results in an understandable way [50]. The tests are divided in three sections:

- **Plug-in tests (Flash, Java)**
- **HTTP / HTML tests**
- **CSS / JavaScript tests**

We will not analyze in depth the details of each test, because a complete documentation can be found in ip-check.info homepage. It is more efficient if we highlight the most important parts [50].

To conduct the test a Microsoft Windows 7 32-bit Operating System will be used in combination with the 4 most popular browsers: Google Chrome, Mozilla Firefox, Internet Explorer 9 and Opera with the latest updates.

The tests scenarios apart from the three aforementioned systems (Proxy servers, Tor, JAP) will also include the browser's "Safe-Surf" option.

Regarding the anonymous proxy servers, lists from the <http://www.idcloak.com> will be used. The IdCloak divides the proxy servers within three levels of anonymity: High, Medium and Low. The last two tests will be conducted in Tor and in JAP.

We need to consider that the failure of an anonymous system tends to be due to the browser and especially the plug-ins, which are usually installed by the users. In the upcoming tests Adobe Flash player and Java Plug-in will be activated. Afterwards, we will use the JonDoFox user profile (safe-surf user profile which deactivates all the unnecessary plug-ins) [27] in Mozilla Firefox and after that we will conduct the tests again.

The second test will try again to take advantage of a vulnerability in user's browser. Each user has a unique setup on her browser/computer and each time that accesses a webpage it is possible for an attacker to extract useful information.

The setup of the test includes three files which are developed to keep a log with the following: user's IP-Address, if the IP-Address is Tor or JAP exit node, User Agents and the Java version that is used by the browser.

We will test a variety of operating systems and a variety of web browsers (Firefox, Chrome, Opera, Safari and Internet Explorer 8) to check their behavior. As in the previous test we will also use the JonDoFox user profile in Mozilla Firefox in combination with JAP and Tor.

## **4.1 The safe-surf option**

Each browser has a "safe-surf" option which has different names in each browser [51].

- **Google Chrome: Incognito Window**
- **Mozilla Firefox: Private Window**
- **Internet Explorer 9: InPrivate Browsing**
- **Opera: Private Window**

By comparing the browsers, the respective safe-surf options have a lot of similarities. When a user enables this option the browser does not keep:

- **Track of the visited pages (Browsing History)**
- **New cookies**
- **Temporary internet files**
- **Auto complete forms**



- Login credentials (username, password)
- General changes in the browser (bookmarks, install plug-ins etc.)

## 4.2 Test in ip-check.info with default options in web browser

First let's take a look to a default test from ip-check.info in Mozilla Firefox. In Figure 4.1 we can see the first field which contains information regarding the user's IP, location, net provider and Reverse DNS of the IP [51].


<b>Your IP</b>	46 [redacted]	<a href="#">Traceroute</a>
Your location	 Vastra Gotaland, Göteborg	<a href="#">Show on map</a>
Your net provider	Student housing networks	<a href="#">Whois IP</a>
Reverse DNS	 sgsnet.se	<a href="#">Whois Domain</a>

Figure 4.1: IP Field [50]

The next field has information taken from HTTP / HTML tests [Figure 4.2] [50] and checks for cookies, authentication data, http session duration, referrer, signature and user-agent [50].

Attribute	Value	Rating
Cookies	Third party sites get your cookies and may track you.	bad
Authentication	Your unique ID: [redacted]	bad
Cache (E-Tags)	Your unique ID: [redacted]	bad
HTTP session	unlimited	bad
Referer	Original: Websites may see from which other website you come from!	medium
Signature	8ab3a24c55 [redacted] (Firefox)	medium
User-Agent	Mozilla/5.0 (Windows NT 6.1; rv:21.0) Gecko/20100101 Firefox/21.0	bad
SSL_session_id	643EA53DDAEF087760B4BE04380F [redacted]	neutral
Language	en-US,en;q=0.5	medium
Content types	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8	medium
Encoding	gzip, deflate	good
Do-Not-Track		medium

Figure 4.2: HTML / HTTP tests [50]

The next two fields, test Adobe Flash player and Java Plug-in [Figure 4.3] [50] and reveal information regarding user's OS and the internal network environment.

In our case the computer has an internal IP address (behind NAT) 10.0.2.15, its OS is Microsoft Windows 7 32bit and obviously the Flash player and Java Plug-in are activated and can be executed by the browser. By identifying the version of the OS an attacker is able to check for vulnerabilities so that she can design a specific attack [50].

Your internal IP	10.0.2.15 (Click here to fix this problem)	More
Java VM	Oracle Corporation 1.7.0_17	
Operating system	Windows 7 x86 Version 6.1	
Language	English, United States	
Flash Cookies	ON (Click here to fix this problem)	
Fonts	144	
Flash Player	Adobe Windows [WIN 11,7,700,202]	
Operating system	Windows 7 [en, Fri May 31 2013 12:19:29 AM]	
Screen	1366*768, 72 DPI	

Figure 4.3: Flash and Java tests [50]

The final test is on CSS and JavaScript [Figure 4.4] [50] and begins, by identifying if the JavaScript is activated (crucial information for an attacker). It is worth mentioning that by using JavaScript numerous privacy risks arising. In our example the attacker can read the list of the installed plug-ins and she can take advantage of a vulnerability in one of them [50].

JavaScript	JavaScript is activated! (Version: 1.5)	medium
Plugins	Found 4 plugins. Java and Flash are active!	bad
Mime types	Found 43 mime types that your browser supports.	bad
Tab name	"window.name" is traceable. Your unique ID: 7046754	bad
Tab history	There are 6 pages in your tab history.	medium
Local storage	Local storage is enabled. Your unique ID: ██████████	medium
Browser bars	MenuBar PersonalBar StatusBar ToolBar ScrollBars LocationBar	good
Browser type	Mozilla/5.0 (Windows) 20100101/20130511120803 Netscape (en-US)	medium
System	Windows NT 6.1 Win32 (Fri May 31 2013 00:21:20 GMT+0200)	medium
Fonts	132 installed fonts have been found on your computer.	bad
Browser history	Protected.	good
JavaScript	JavaScript is activated! (Version: 1.5)	medium
Mime types	Found 43 mime types that your browser supports.	bad
Tab name	Your browser seems to destroy the "window.name" attribute.	medium
Tab history	There are 6 pages in your tab history.	medium
Local storage	Local storage is enabled. Your unique ID: 17046754	medium
Browser window	1382 x 744 pixels. 1366 x 624 pixels (inner size). Zoom: 100%	medium
Browser bars	MenuBar PersonalBar StatusBar ToolBar ScrollBars LocationBar	good
System	Windows NT 6.1 Win32 (Fri May 31 2013 00:19:29 GMT+0200)	medium

Figure 4.4: CSS and JavaScript tests [50]

The results have three properties, "Attribute", "Value" and "Rating". The attribute is the title of the test and is presented by using a color code with three levels of caution: green, orange and red. The "value" is the actual result of the test with a countermeasure proposal wherever is needed. The last property is the "rating" of ip-check evaluation [50].

To present the results the same color code will be used. A field, which is green its rating is "good" and this means that according to this test the user has selected the optimal settings. A field which is orange is rated as "medium" and that is a speculation about a possible a privacy problem [50]. Finally a red field is rated as "bad" and the user needs to change some of its settings [50].

### 4.3 Test with the custom made test

The setup of the test includes an HTML webpage (<http://www.pcgadget.gr/test/test.html>) which runs a script to return the Java Version. Then it forwards the user to another page (<http://www.pcgadget.gr/test/thelog.php>) where returns more results regarding the user's browser/computer setup. Finally, forwards to the final webpage (<http://www.pcgadget.gr/test/index.html>) where the user sees the message "Welcome to my Thesis web page!"

Behind the scene a log is taken with all the previously described information. Let's see how looks like a log, which is taken by using Mozilla Firefox browser:

```

1 [1]
2 Java Version: 1.7.0_25
3 The IP Address 90.2 [REDACTED] is NOT Tor Exit Node!
4 The IP Address 90.2 [REDACTED] is NOT JAP Exit Node!
5 IP Address: 90.2 [REDACTED] Time: Oct 25 10:09:51 2013 Referrer: http://www.pcgadget.gr/test/test.html
6 User Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0 Query:
7 [Host :www.pcgadget.gr, User-Agent :Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0,
8 Accept :text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8, Accept-Language :en-US,en;q=0.5,
9 Accept-Encoding :gzip, deflate, Referer :http://www.pcgadget.gr/test/test.html, Connection :keep-alive,
10 Content-Type :application/x-www-form-urlencoded, Content-Length :14]

```

**Figure 4.5:** Log Record

The log record [Figure 4.5] in the first line contains a log counter and right after the results from the Java version check. The next two consecutive lines are checks regarding the IP Address and if it is part of Tor/JAP network (lines three and four). On lines five and six are information about the IP Address, the access time, the referrer, the contents of the user agent and if there is any query string [52]. The last four lines contain information by using the `apache_request_headers`

function "to fetch the HTTP request headers" [52] .

The main goal of this test is to try to find patterns between the logs so that can reveal the real IP address of the user.

## 4.4 Test the Implementations

To sum up, the evaluation tests which will be conducted are:

1. Test in browser's Safe-Surf option
2. Test in anonymous Proxy Servers from ip-check.info
  - (a) Low anonymity proxy with Java/Flash activated
  - (b) Low anonymity proxy with Java/Flash non activated
  - (c) Medium anonymity proxy with Java/Flash activated
  - (d) Medium anonymity proxy with Java/Flash non activated
  - (e) High anonymity proxy with Java/Flash activated
  - (f) High anonymity proxy with Java/Flash non activated
3. Test in Java Anonymous Proxy (JAP) from ip-check.info
  - (a) Mozilla Firefox with default user profile
  - (b) Mozilla Firefox with JonDoFox user profile
4. Test in Tor from ip-check.info
  - (a) Mozilla Firefox in Tor bundle
  - (b) Mozilla Firefox with default user profile
  - (c) Mozilla Firefox with JonDoFox user profile
5. Custom Test in anonymity systems and in proxy servers

Before proceed in the next chapter let's give a glimpse of the procedures.

**Safe-Surf option:** To proceed with this test, just activate the "anonymous mechanisms" of each browser and then check the results.

**Anonymous Proxy Servers:** The proxy servers which will be used are taken by the idcloak.com. The anonymity levels are divided in Low - Medium - High.

Tests in different proxy servers will be conducted with the ip-check.info, to identify the levels of anonymity and how this affects the user's privacy.

Mozilla Firefox browser will be used with the default user profile which means that, Java plug-in and Adobe Flash player will be executed. Afterwards the JonDoFox profile will be used with Mozilla Firefox.

To use a proxy server with Mozilla Firefox we need to setup properly: Firefox->Options->Network->Connection -> Settings and then Manual Proxy Configuration [Figure 35]. In the HTTP Proxy field setup the IP address of the desired proxy server and then it is important to tick the "Use this proxy server for all protocols", because a leakage of information can occur by a protocol which is not used. Later we will give an example to make this clear. The tests are going to be divided in three levels of anonymity Low - Medium - High and each one will use two different user profiles in Mozilla Firefox, the default user profile and the JonDoFox user profile (deactivated Flash/Java) [27]. In the results there will be a comparison between the two user profiles between the different levels of anonymity.

**Java Anonymous Proxy (JAP):** The JAP will be the next anonymity system/software, which will be subjected to tests. First need to download and install the newest JAP/JonDo [27]. Afterwards will fire up the program and choose the desired cascade. There are two types of cascades, with free mixes and with premium mixes (users need to pay to use them) The differences between cascades composed by premium mixes and the free ones are:

1. The premium composed by three mixes while the free ones have only one or two mixes
2. The response time in premiums is better
3. The premium has higher availability
4. The speed is higher in premium cascades

In this test two different free cascades and web browser configurations will be used. The first one will be with the default Mozilla Firefox user profile and the second one will be with the JonDoFox user profile.

**Tor:** The most popular among the anonymity projects is Tor. To evaluate Tor the same procedure will be followed. By downloading the bundle version of Tor a user receives the Vidalia control panel (user interface) where he can setup Tor and also a properly configured, portable version of Mozilla Firefox.

By executing the "Start Tor Browser.exe" the user gets connected to Tor network. Then Mozilla Firefox fires up in a homepage where checks if Tor is active and is giving the appropriate feedback to the user.

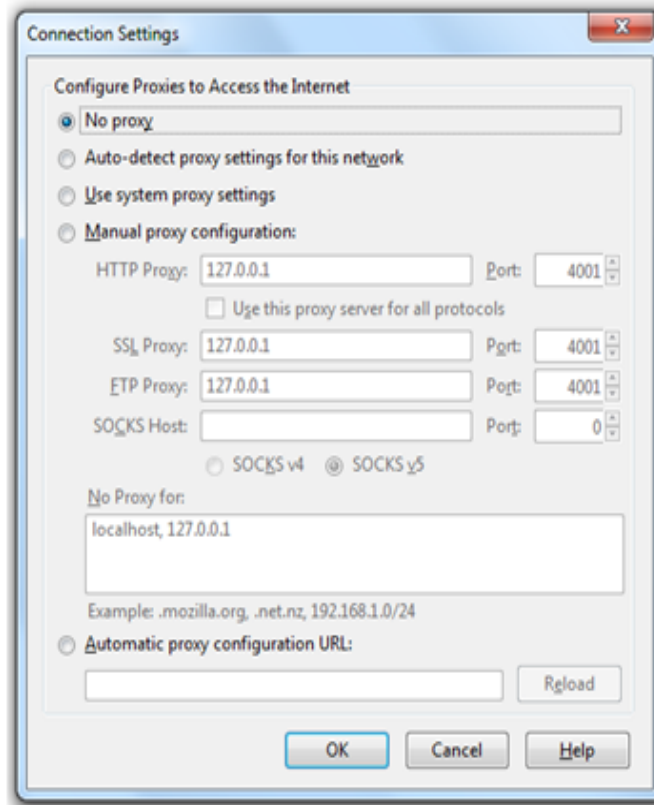


Figure 4.6: Proxy Server Setting in Mozilla Firefox

To conduct the tests we use three different configurations. The first will be in the browser which comes within the Tor bundle, the second will be in the default user profile in Mozilla Firefox and the last one in JonDoFox profile. The setup of the network settings will be according to the Tor's project instructions.

**TrueCrypt and PGP:** In the next chapter a discussion regarding the vulnerabilities of these implementations will be held. Also, there will be a reference to speculations concerning the origin of TrueCrypt.

# 5

## Results of Evaluation

**I**N this chapter an analysis of the results of the tests will be given as well as, a comparison of the implementations with respect to their effectiveness.

### 5.1 Safe-Surf Option in ip-check.info

As it was expected the results are exactly the same with the first default test, because the safe-surf option affects only the browser. This creates a fake feeling of anonymity, which can have negative effect in the user's online behavior.

This option is useful only in public computers and if the user does not want to leave traces behind (history, auto complete forms, searches, etc.).

#### 5.1.1 Low Anonymity Java/Flash activated vs. Java/Flash non activated

With the Java plug-in activated, the real IP Address is revealed and furthermore an attacker can also indentify that this user is behind a NAT (internal IP Address). The X-FORWARDED-FOR is a header (HTTP header) that is used by the proxy servers to get the IP Address of the user. By using Javascript it is also possible to observe the installed plug-ins and check for possible vulnerabilities.

Then deactivate the Java/Flash player and rerun the test by using the Firefox with JonDoFox profile. Although it was not possible to eavesdrop the IP of the user by taking advantage of Java plug-in (is not active anymore), the low anonymity

proxy server is using the” X-FORWARDED-FOR” and as a result the real IP address is revealed.

### 5.1.2 Medium Anonymity Java/Flash activated vs. Java/Flash non activated

By running the second test with the Java/Flash activated, the user’s identity was revealed. This medium anonymity server is not using the X-FORWARDED-FOR but the problem with Java/Flash remains. By using the JonDoFox profile and rerun the test this server does not reveal any information regarding the IP address of the user.

Now it is time to try something different. By keeping the same user profile in Firefox let’s make one small change. Instead of using proxy server for all the protocols (SSL, FTP) we will not use it for SSL and FTP. Then rerun the test to check the results [Figure 4.6].

Although Java and Flash are not running, the faulty configuration of the web browser leads to reveal the source IP address [Figure 5.1].

<b>Your IP</b>	125.216.144.100 (Proxy) 46.209.254.203 (HTTPS) 46.209.254.203 (FTP)	<a href="#">Traceroute</a>
Your location	 China	<a href="#">Show on map</a>
Your net provider	<a href="#">Guangzhou Auto (China)</a> <a href="#">View their privacy policy</a>	<a href="#">Whois IP</a>
<b>HTTP session</b>	unlimited	<a href="#">bad</a>
<b>Referer</b>	hidden (changed when switching the website)	<a href="#">good</a>
<b>Cookies</b>	Your browser does not store any cookies.	<a href="#">good</a>
<b>Authentication</b>	protected	<a href="#">good</a>
<b>JavaScript</b>	JavaScript is currently turned off.	<a href="#">good</a>

Figure 5.1: Faulty proxy configuration [50]

### 5.1.3 Medium Anonymity Java/Flash activated vs. Java/Flash non activated

As it is expected the results are the same as before. The Java leaked the source IP address and revealed user’s identity. The high anonymity server does not use the X-FORWARDED-FOR so the leakage is happening due to the plug-ins and also due to wrong configuration. By activating the JonDoFox profile the proxy server does not reveal any crucial information about the user.

The last step is to use the same wrong configuration as before and rerun the test where the results are that the same as before (user’s IP is disclosed).



The upcoming table has the aggregated result from the previous tests and it will use the color code from the ip-check.info [Figure 5.2].

Attack	Information Leakage Risk					
	Proxy Server					
	Low Anonymity		Medium Anonymity		High Anonymity	
	Default profile in Firefox	JonDoFox user Profile	Default profile in Firefox	JonDoFox user Profile	Default profile in Firefox	JonDoFox user Profile
IP Check	Red	Green	Red	Green	Red	Green
Location	Red	Green	Red	Green	Red	Green
Net Provider	Red	Green	Red	Green	Red	Green
Reverse-DNS	Red	Green	Red	Green	Red	Green
Cookies	Red	Green	Red	Green	Red	Green
Referer	Yellow	Green	Yellow	Green	Yellow	Green
HTTPS Session	Red	Green	Red	Green	Red	Green
X-FORW-FOR	Red	Green	Red	Green	Red	Green
Authentication	Red	Green	Red	Green	Red	Green
Java Test	Red	Green	Red	Green	Red	Green
Flash	Red	Green	Red	Green	Red	Green
JavaScript	Red	Green	Yellow	Green	Yellow	Green
Plugins	Red	Green	Red	Green	Red	Green
Tab History	Yellow	Green	Yellow	Green	Yellow	Green
Tab Name	Red	Green	Red	Green	Red	Green
Mime Types	Red	Green	Red	Green	Red	Green

Figure 5.2: Proxy Server tests

## 5.2 Java Anonymous Proxy with Mozilla Firefox default user profile vs. JonDoFox user profile

The first step is to activate the JAP and choose a cascade that consists of one mix and run the ip-check test with Java and Flash activated. The result is that the IP address of the users is revealed. Then a cascade that consist of two mixes is used and rerun the test. There is no improvement and the results are identical.

The second step is to open the Mozilla Firefox with the JonDoFox profile. With the combination of JonDoFox and JAP there is no IP Address leakage. By using two different cascades (one or two mixes) the only difference is in the HTTP Session, where in the cascade with only one mix is unlimited whereas in the two mix cascade is stateless (each HTTP request is anonymized independently by the others and cannot be connected with older requests) [27].

## 5.3 Tor

The next anonymous system, which will be subjected in tests is Tor.

### 5.3.1 Mozilla Firefox in Tor bundle

Mozilla Firefox does not deactivate the JavaScript due to better user experience. Although this fact, the real IP address of the user is not revealed, but the referrer and the tab history values can give some useful information regarding the user browsing history

### 5.3.2 Mozilla Firefox with default user profile

With Flash and Java activated the results are as expected. The original IP Address of the user is revealed, though (FTP) although is running though Tor network. This is the same kind of leakage, as in anonymous proxy servers when there was a miss-configuration in the network settings.

### 5.3.3 Mozilla Firefox with JonDoFox user Profile

The final test reveals also user's real IP address. The miss configuration of the FTP which is not Tor compatible leads to this leakage.

## 5.4 Java Anonymous Proxy vs. Tor

To present the results of the tests in JAP and Tor a table has been created with the most important attributes and their rating according to ip-check.info [Figure 5.3] [27].

## 5.5 Results from Custom Test in anonymity systems

Earlier in the analysis of this test we define as main goal to find patterns between the logs. Now is time to check some of the records. In the upcoming picture we used the default Firefox browser of the Backtrack Operating System with the default user profile to surf on the Internet. Then we used it again to surf anonymously through the Tor network by just changing the network properties.

To evaluate the result between the two log records we will compare them with a custom String Evaluator ([http://www.pcgadget.gr/evalstring/string\\_omp.php](http://www.pcgadget.gr/evalstring/string_omp.php)),

Attack	Information Leakage Risk				
	Tor Project			JAP AN.ON Project	
	Mozilla Firefox in Tor	Default profile in Firefox	JonDoFox user Profile	Default profile in Firefox	JonDoFox user Profile
IP Check	Green	Red	Red	Red	Green
Location	Green	Red	Red	Red	Green
Net Provider	Green	Red	Red	Red	Green
Reverse-DNS	Green	Red	Red	Red	Green
Cookies	Yellow	Red	Green	Red	Green
Referer	Yellow	Red	Green	Yellow	Green
HTTPS Session	Yellow	Red	Yellow	Red	Green
X-FORW-FOR	Green	Red	Green	Red	Green
Authentication	Green	Red	Green	Red	Green
Java Test	Green	Red	Green	Red	Green
Flash	Green	Red	Green	Red	Green
JavaScript	Yellow	Red	Green	Yellow	Green
Plugins	Green	Red	Green	Red	Green
Tab History	Yellow	Red	Green	Yellow	Green
Tab Name	Red	Red	Green	Red	Green
Mime Types	Green	Red	Green	Red	Green

Figure 5.3: JAP vs. Tor

```

1 [1] BackTrack Default Firefox
2 Java version not found!
3 The IP Address 90.2... is NOT Tor Exit Node!
4 The IP Address 90.2... is NOT JAP Exit Node!
5 IP Address: 90.2... Time: Oct 23 8:32:38 2013 Referrer: http://www.pogadget.gr/test/test.html
6 User Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1 Query:
7 [Host :www.pogadget.gr, User-Agent :Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1,
8 Accept :text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8, Accept-Language :en-us,en;q=0.5,
9 Accept-Encoding :gzip, deflate, Accept-Charset :ISO-8859-1,utf-8;q=0.7,*;q=0.7, Keep-Alive :115,
10 Connection :keep-alive, Referer :http://www.pogadget.gr/test/test.html, Content-Type :application/x-www-form-urlencoded, Content-Length :6]
11
12 [2] BackTrack Default Firefox in Tor
13 Java version not found!
14 The IP Address 5.1... is Tor Exit Node!
15 The IP Address 5.1... is NOT JAP Exit Node!
16 IP Address: 5.1... Time: Oct 23 8:40:00 2013 Referrer: http://www.pogadget.gr/test/test.html
17 User Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1 Query:
18 [Host :www.pogadget.gr, User-Agent :Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1,
19 Accept :text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8, Accept-Language :en-us,en;q=0.5,
20 Accept-Encoding :gzip, deflate, Accept-Charset :ISO-8859-1,utf-8;q=0.7,*;q=0.7, Keep-Alive :115,
21 Connection :keep-alive, Referer :http://www.pogadget.gr/test/test.html, Content-Type :application/x-www-form-urlencoded, Content-Length :6]
22

```

Figure 5.4: Log Comparison

which is using the `similar_text` PHP function and returns the percentage of similarity between two strings [52].

By running this evaluation and by including all the parameters of these logs (IP address, access time, log counter e.t.c) they are 96.08% similar. By removing the obvious different parts (IP address, access time, log counter) the two logs are 99.81% similar, which means that they are basically the same.

Next we will repeat the same test but instead of using the default Firefox to surf through Tor, the Tor's Firefox browser will be used [Figure 5.5].

By repeating the same comparisons as before the two logs (including all the

```

1 [1] BackTrack with Tor browser
2 Java version not found!
3 The IP Address 3 0.1 is Tor Exit Node!
4 The IP Address 3 0.1 is NOT JAP Exit Node!
5 IP Address: 3 0.1 Time: Oct 23 8:36:18 2013 Referrer: http://www.pcgadget.gr/test/test.html
6 User Agent: Mozilla/5.0 (Windows NT 6.1; rv:17.0) Gecko/20100101 Firefox/17.0 Query:
7 [Host :www.pcgadget.gr, User-Agent :Mozilla/5.0 (Windows NT 6.1; rv:17.0) Gecko/20100101 Firefox/17.0,
8 Accept :text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8, Accept-Language :en-us,en;q=0.5,
9 Accept-Encoding :gzip, deflate, Connection :keep-alive, Referer :http://www.pcgadget.gr/test/test.html,
10 Content-Type :application/x-www-form-urlencoded, Content-Length :6]
11
12 [2] BackTrack with default Firefox
13 Java version not found!
14 The IP Address 9 38 is NOT Tor Exit Node!
15 The IP Address 9 38 is NOT JAP Exit Node!
16 IP Address: 9 38 Time: Oct 23 8:32:38 2013 Referrer: http://www.pcgadget.gr/test/test.html
17 User Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1 Query:
18 [Host :www.pcgadget.gr, User-Agent :Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1,
19 Accept :text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8, Accept-Language :en-us,en;q=0.5,
20 Accept-Encoding :gzip, deflate, Accept-Charset :ISO-8859-1,utf-8;q=0.7,*;q=0.7, Keep-Alive :115,
21 Connection :keep-alive, Referer :http://www.pcgadget.gr/test/test.html, Content-Type :application/x-www-form-urlencoded, Content-Length :6]

```

**Figure 5.5:** Log Record Tor Firefox browser versus Default Firefox

information) are 88.16% similar and if we exclude all the obvious different parts the two logs are 86.86% similar. The results from these comparisons are extremely useful and lead us to an indisputable result. The user must not make use of her default browser to surf the internet because as it is obvious if she visits the same webpage twice can be identified.

In the second subpart of this custom test we will provide some information regarding the behavior of each browser when we ran the test from <http://www.pcgadget.gr/test/test.html>.

As we can see in the Table 5.6 the browsers with the NoScript plug-in installed did not leak any information regarding the Java version. During the test Opera and Safari did not pop up any notification windows or message and also the Java version has been recorded in the log file. The attacker by gathering these information can harm the user with a targeted attack for the specified version of Java (possible vulnerable if older version).

## 5.6 Vulnerabilities in TrueCrypt and PGP

In this part there will be a reference to the vulnerabilities of the very famous encryption software TrueCrypt and PGP. To attack an encryption software there are two ways, the brute-force and the side-channel.

Browsers in Win7 OS	NoScript plug-in installed	Java prompt message	Recorded Java version in Log file
Default Firefox	No	Yes	Yes
Firefox JonDoFox Profile	Yes	Yes	No
Firefox (Tor bundle)	Yes	Yes	No
Chrome	No	Yes	Yes
Opera	No	No	Yes
Safari	No	No	Yes
IE8	No	Yes	No

Figure 5.6: IP Field [50]

### 5.6.1 Brute Force Attack

The brute-force attack of encryption software (SW) is straightforward; the attacker needs two constituents, a good dictionary and a lot of patience. The attack although it is possible, it is not feasible and depends absolutely on the user's password policy. There are tools where the attacker can create dictionaries with passwords up to a specific length or passwords, which contain specific characters or numbers (John the Ripper, TCBrute 2, OTFBrutus, TCHead etc) .

### 5.6.2 Side Channel Attack

On the other hand, there are the side-channels attacks, where the attacker tries to gain access to the encrypted data by taking advantage of vulnerabilities in the implementation of the cryptosystem. Currently there are three major well know vulnerabilities.

1. **Hibernation Files:** A lot of users instead of power off their computers, they hibernate them. This procedure is something like a power saving mode where the operating systems takes an instance of the system memory (RAM) and write all the data in a hibernation file. By power on the computer everything is where you left them [41].
2. **Paging Files (Swap Files):** The Operating System (OS) a lot of times instead of keeping the parts of programs or data in the RAM (due to lack of memory), it is transferring these data to the hard drive. As it is described [41] when an encrypted i.e. text file opens, all its contents are transferred

unencrypted in RAM. If the OS decides to free some memory and transfers some data in hard drive are going to be unencrypted and accessible [41].

3. **Memory Dump Files:** When a system crashes it keeps a snapshot of its memory prior to the crash to the hard drive for debugging reasons. It is the same concept with paging files; the data which were in the system memory unencrypted were transferred in the hard drive [41].

The question now is how these vulnerabilities affect the security of the system. An attacker by using the hibernation file can have access to encrypted data by analyzing this file. The user when decrypts a file the master key is saved in RAM unencrypted. When the computer hibernates, the system memory is saved in the hard drive and all the data are accessible even if the computer is turned off.//

The designers of TrueCrypt propose to encrypt the whole disk or partition and activate options to dismount the drives prior to hibernation. The PGP is deactivating the hibernation in the OS and also Symantec is proposing the PGP Whole Disk Encryption (PGP WDE) which encrypts the whole volume (swap files, boot sector, system file) [41, 53].

The paging file vulnerability can be avoided by forcing the OS to keep the paging files into the volume, which is encrypted. This is not always happening because i.e. Microsoft Window can keep the paging files in an unencrypted volume whereas. Linux distributions they always encrypt the swap space which contains the paging files and the hibernation file [41, 53].

The last vulnerability, which is the most difficult to protect from it is the memory dump files. A crash can happens unexpectedly. The encryption SW, as far it stops working cannot take care of the data and the attacker can find a complete (depends on the OS) memory dump (the whole system memory data). When the encrypted volume is opened the master key is stored in the system memory and it will be accessible though the memory dump file. To protect against this vulnerability the solution is to disable the memory dump file[41, 53].

Lately, computer SWs (Passware Kit Forensic 12.5, Elcomsoft Forensic Disk Decryptor) can take advantage of these vulnerabilities.

All in all these vulnerabilities bring out the greater vulnerability of all, the human factor. The only way to gain access in a system encrypted with TrueCrypt or PGP is only through a misconfigured system.

# 6

## Conclusion

**I**N this report three different types of anonymity systems (JAP, Tor and Proxy Servers) have been evaluated. The results regarding the user's privacy are dispiriting. Most of the times the tests, managed to reveal user's original IP address or to leak the user's private information. The severity of the privacy breach depends on the type of leaked information. As obvious from Figure 5.3 only the combination of JonDoFox profile and JAP was able to preserve all information about the user's identity during the test.

During the tests we realized that all the proxy systems leaked the IP Address in the Java plug-in. This happens because they cannot bind third party plug-ins and the user must use special software to achieve that.

Although the anonymity systems did not succeed in all tests we need to consider that when the system was properly configured, Tor and JAP managed to hide the user's IP address. The user needs to carefully configure the web browser and the anonymity system to gain all the benefits of its use.

Users tend to misunderstand the topic of anonymity. They must always be aware that they need to know who they are trying to be protected from and also need to consider that to be anonymous they need other people around them. Finally, they need to look and behave in the same way as others. Anonymity come with company.

The anonymity is also linked with the term fingerprinting (usually in terms of a web browser), where if someone differs from the crowd that she belongs to, she risks losing her anonymity. A quick example is the Live CD (a bootable operating system) in combination with an anonymity system. This combination is proposed by a lot of sources, so that a user can maintain her anonymity while she surfs on the internet. Generally this system is safe, but we will argue for the fact that if a

user does not have a new Live CD, she will end up surfing with an outdated Linux distribution by using an old version of a browser. A general statement, which can be used is the following: "a user, who is different in an environment in which all the other users look alike can be identified".

Finally, regarding the encryption tools (TrueCrypt and PGP) it is worth mentioning that they are as good as the user as far they use passwords, which are created by the user to protect the data. Every year studies reveal that passwords such as "password", "123456", "qwerty" or "111111" are on the top 10 of the most common passwords. Even if solutions like the aforementioned are the best in their category they cannot protect against a weak password.

Lately anonymity has become very popular (after some incidents). More and more users start to be aware about how they unknowingly share their personal information and what protection measures need to take to achieve a decent level of anonymity.



# Bibliography

- [1] Fingerprint Cover Image.  
URL <http://www.presentermedia.com/>
- [2] J. O. G.W. van Blarkom RE, J.J. Borking, Handbook of Privacy and Privacy Enhancing Technologies, The case of Intelligent Software Agents.  
URL <http://www.andrewpatrick.ca/pisa/handbook/handbook.html>
- [3] L. D. B. Samuel D. Warren, The Right to Privacy 4 (5) (1890) 193–220.
- [4] S. Michael McFarland, What is Privacy.  
URL <http://www.scu.edu/ethics/practicing/focusareas/technology/internet/privacy/what-is-privacy.html>
- [5] M. H. Andreas Pftizmann, A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management.  
URL [http://dud.inf.tu-dresden.de/Anon\\_Terminology.shtml](http://dud.inf.tu-dresden.de/Anon_Terminology.shtml)
- [6] The Preciosa Project.  
URL <http://www.preciosa-project.org/privacy-terminology>
- [7] L. Fritsch, State of the art of Privacy-enhancing Technology (PET).
- [8] T. Olovsson, Course slides in Network Security.
- [9] B. von Sydow, Course slides in Cryptography, chalmers University of Technology.
- [10] W. Stallings, Cryptography and Network Security, Principles and Practice, fifth edition Edition, Prentice Hall, 2011.

- [11] I. T. Laboratory, Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, William C. Barker, Elaine Barker Computer Security Division MD 20899-8930.
- [12] F. I. P. S. P. 197, Announcing the Advanced Encryption Standard (AES).
- [13] K. I. Subhamoy Maitra, Indian Statistical Institute, RC4 Stream Cipher and Its Variants.
- [14] A. Jeremy Quirke, Security in the GSM system (May 2004).  
URL <http://www.ausmobile.com>
- [15] Wikipedia ECB-CBC image comparison.  
URL [http://en.wikipedia.org/wiki/Block\\_cipher\\_mode\\_of\\_operation#Electronic\\_codebook\\_.28ECB.29](http://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#Electronic_codebook_.28ECB.29)
- [16] W. Diffie, M. E. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory Vol. IT-22.
- [17] D. L. Chaum, Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms, University of California, Berkeley.
- [18] Wikipedia image, decryption operation in Mix nets.  
URL [http://en.wikipedia.org/wiki/File:Decryption\\_mix\\_net.png](http://en.wikipedia.org/wiki/File:Decryption_mix_net.png)
- [19] P. S. David Goldschlag, Michael Reed, Onion routing, Commun, ACM 42(2) (1999) 39–41.
- [20] P. S. David Goldschlag, Michael Reed, Hiding Routing Information.
- [21] Wikipedia image, Figure for Onion Routing.  
URL [http://en.wikipedia.org/wiki/File:Onion\\_diagram.svg](http://en.wikipedia.org/wiki/File:Onion_diagram.svg)
- [22] I. G. R, Privacy Enhancing Technologies for the Internet III: Ten Years Later.
- [23] U. M. Len Sassaman, Mixmaster.  
URL <http://mixmaster.sourceforge.net>
- [24] D. H. N. M. George Danezis, Roger Dingledine, Mixminion: Design of a Type III Anonymous Remailer Protocol, IEEE Symposium on Security and Privacy, 2003.
- [25] Proxy Server.  
URL <http://www.publicproxyservers.com/>

- [26] H. Federrath, JAP Anonymity and Privacy.  
URL <http://anon.inf.tu-dresden.de/index/en.html>
- [27] Project: AN.ON Anonymität Online.  
URL <http://anon.inf.tu-dresden.de/>
- [28] P. S. Roger Dingledine, Reliable Mix Cascade Networks through reputation.
- [29] Wikipedia image, decryption operation in Mix nets.  
URL [http://en.wikipedia.org/wiki/File:Decryption\\_mix\\_net.png](http://en.wikipedia.org/wiki/File:Decryption_mix_net.png)
- [30] Cache Proxy .  
URL <http://www.squid-cache.org/>
- [31] P. S. Roger Dingledine, Nick Mathewson, Tor: The Second-Generation Onion Router, 2004, 13th USENIX Security Symposium, San Diego, CA.
- [32] N. M. Roger Dingledine.
- [33] Tor Metric Project.  
URL <https://metrics.torproject.org>
- [34] J. Lindqvist, Practical privacy enhancing technologies for mobile systems, Ph.D. thesis, Helsinki University of Technology (2009).
- [35] P. S. Lasse Øverlier, Locating Hidden Servers, 2006, IEEE Symposium on Security and Privacy.
- [36] B. N. L. C. S. Matthew Wright, Micah Adler, An Analysis of the Degradation of Anonymous Protocols, 2002, network and Distributed Security Symposium - NDSS.
- [37] B. N. L. C. S. Matthew Wright, Micah Adler, Defending Anonymous Communication Against Passive Logging Attacks, 2003, IEEE Symposium on Security and Privacy.
- [38] A. R. Michael Reiter, Crowds: Anonymity for Web Transactions, Vol. 1(1), in ACM Transactions on Information and System Security.
- [39] B. N. L. Matthew K. Wright, Micah Adler, The Predecessor Attack: An Analysis of a Threat to Anonymous Communications Systems.
- [40] B. W. T. W. H. Ian Clarke, Oskar Sandberg, Freenet: A Distributed Anonymous Information Storage and Retrieval System, 2001, in Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability.

- [41] TrueCrypt project.  
URL <http://www.truecrypt.org>
- [42] Analysis: Is there a backdoor in Truecrypt? Is Truecrypt a CIA honeypot?  
URL <http://www.privacylover.com/encryption/analysis-is-there-a-backdoor-in-truecrypt-is-truecrypt-a-cia-honeypot/>
- [43] P. R. Zimmermann, The Official PGP User's Guide (Jan. 1996).
- [44] How PGP works.  
URL <http://users.ece.cmu.edu/~adrian/630-f04/PGP-intro.html>
- [45] GNUPG project.  
URL <http://www.gnupg.org/>
- [46] Information, T. N. Privacy Commissioner/Ontario Canada, Registratiekamer, Privacy-Enhancing Technologies: The Path to Anonymity 1.
- [47] J. O. J. H. G.W. van Blarckom, J.J. Borking, Handbook of Privacy and Privacy-Enhancing Technologies - The case of Intelligent Software Agents, 2003, pISA Consortium, The Hague.
- [48] D. J. Solove, A Taxonomy of the Privacy.
- [49] M. of Science Technology, M. G. Innovation, Privacy Enhancing Technologies.
- [50] The IP Check project .  
URL <http://www.ip-check.info>
- [51] Anonymous Window in Browsers, Chrome, Firefox, IE9, Opera .  
URL <https://support.google.com/chrome/answer/95464?hl=en>
- [52] PHP Manual.  
URL <http://php.net/manual>
- [53] Security of PGP.  
URL <http://www.symantec.com>
- [54] J. Brody, P. Yager, R. Goldstein, R. Austin, Biotechnology at low Reynolds numbers, Biophysical Journal 71 (6) (1996) 3430–3441.  
URL <http://linkinghub.elsevier.com/retrieve/pii/S0006349596795383>

# A

## Appendix

### A.1 TheTest.html – Extract Java version

```
<!doctype html>
<html lang="en">
  <head>
    <title>Check My Page!!!</title>

    <form id="testForm" name="testForm" method="post"
      action="http://www.pcgadget.gr/test/thelog.php">
      <input type="hidden" name="total" id="total"
        value="">

    </form>

    <script type="text/javascript"
      src="http://java.com/js/deployJava.js"></script>
    <script type="text/javascript">
      var version = deployJava.getJREs();
      document.testForm.total.value = version;
      document.forms["testForm"].submit();
    </script>

  </head>
  <body>
    <h1>Welcome!</h1>
```

```
</body>
</html>
```

## A.2 Thelog.php – Keep Log records

```
<?php

// Variables
$time = date("M_j_G:i:s_Y");
$ip = getenv('REMOTE_ADDR');
$userAgent = getenv('HTTP_USER_AGENT');
$referrer = getenv('HTTP_REFERER');
$query = getenv('QUERY_STRING');

// Log Entry
$data = "IP_Address:_" . $ip . "_Time:_" . $time . "_Referrer:_" .
        $referrer . "_User_Agent:_" . $userAgent . "_Query:_" . $query;

// Format the meta-data / Headers
$head = apache_request_headers();
$sats = arrayToString($head);
$plus_head = $data . "\n" . $sats . "\n" . "\n";

// Functions
// Check if the IP Address is Tor Exit Node
function check_tor(){
    $row = 1;
    $the_ip = getenv('REMOTE_ADDR');
    $bool = False;
    if (($handle = fopen("http://torstatus.blutmagie.de
        _____/ip_list_exit.php/Tor_ip_list_EXIT.csv", "r"))
        !== FALSE) {
        while (($data = fgetcsv($handle, 1000000, ","))
            !== FALSE) {
            $num = count($data);
            for ($c=0; $c < $num; $c++) {
                if ($the_ip == trim($data[$c])){
                    $bool = True;
                }
            }
        }
    }
}
```

```

        }
    }
}
fclose($handle);
}
//Print Message
if($bool == True){
    return "The_IP_Address_". $the_ip. "_is
    =====Tor_Exit_Node!";
}
else{
    return "The_IP_Address_". $the_ip. "_is_NOT
    =====Tor_Exit_Node!";
}
return $bool;
}

//Check if the IP address is part of JAP network
function check_jap(){
$row = 1;
    $the_ip = getenv('REMOTEADDR');
    $bool = False;
    if (($handle = fopen("http://www.pcgadget.gr
    =====/checkip/jp.txt", "r")) != FALSE) {
        while (($data = fgetcsv($handle, 1000000, ",,"))
            != FALSE) {
            $num = count($data);
            for ($c=0; $c < $num; $c++) {
                if ($the_ip == trim($data[$c])){
                    $bool = True;
                }
            }
        }
        fclose($handle);
    }
    //Print Message
    if($bool == True){
        return "The_IP_Address_". $the_ip. "_is_JAP
        =====Exit_Node!";
    }
}

```

```

    else{
        return "The_IP_Address_".$the_ip."_is_NOT
        _____JAP_Exit_Node!";
    }
    return $bool;
}

//Function to keep the logs on file
function writeLog($the_data){
    $day = date("Y_m_d");
    $my_file = $day.'_logs.txt';
    $handle = fopen($my_file, 'a') or
    die('Error on open file: '.$my_file);
    fwrite($handle, $the_data);
    fclose($handle);
    //echo 'Write on file OK!';
}

//Function to redirect to the homepage
function Redirect($url, $per = false){
    if (headers_sent() == false){
        header('Location: ' . $url, true,
            ($per == true) ? 301 : 302);
    }
    exit();
}

//Function to keep transform the data
//from http://www.daniweb.com/
//web-development/php/threads/8158/array-to-string

function arrayToString($array){
    $i = 0;
    $string = '';
    foreach ($array as $index => $value ){
        if($i != count($array)-1){
            $string .= "$index_: $value, ";
        }else $string .= "$index_: $value";
        $i++;
    }
}

```



```

$string = '['.$string.'];';

return $string;
}

//Keep track of log count
function readCount(){
    $my_file = 'count.txt';
    $handle = fopen($my_file, 'r') or
    die('Error on open file: '.$my_file);
    $counter = fread($handle, filesize($my_file));
    fclose($handle);
    return $counter;
}

//Keep track of log count
function writeCount($the_data){
    $my_file = 'count.txt';
    $handle = fopen($my_file, 'w') or
    die('Error on open file: '.$my_file);
    fwrite($handle, $the_data);
    fclose($handle);
}

if(strlen(preg_replace('/\s+/u', '', readCount())) == 0){
    writeCount('1');
}

$log_counter = readCount();
$tor = check_tor();
$jap = check_jap();
$j = $_POST['total'];

if(strlen(preg_replace('/\s+/u', '', $j)) == 0){
    $java = "Java_version_not_found!";
}
else{
    $java = "Java_Version:_" . $_POST['total'];
}

```

```

$a = [ ". $log_counter." ] . "\n" . $java . "\n" .
      $tor . "\n" . $jap . "\n" . $plus_head ;
writeLog($a);
$count = intval($log_counter) + 1;
writeCount($count);

Redirect('http://www.pcgadget.gr/test/index.html', false);

?>

```

### A.3 Stringcomp.php – Compare the Logs and return the % of similarity

```

<!doctype html>
<html lang="en">
  <head>
    <title>String Evaluator</title >
  </head>
  <body>
    <h1>Welcome to the String Evaluator!</h1>
    <form NAME ="form1" METHOD ="POST" ACTION = "">
      <textarea name="TEXT1" cols="45" rows="25"
                required></textarea>
      <textarea name="TEXT2" cols="45" rows="25"
                required></textarea>
      <br>
      <input TYPE = "Submit" NAME = "Submit1"
            VALUE = "Submit">
    </form>
    <br>
    <br>
    <b>
      <?php
        $var1 = $_POST[ 'TEXT1' ];
        $var2 = $_POST[ 'TEXT2' ];
        similar_text ( $var1, $var2, $result );
        print( "The_2_string_are:_". $result .
              "%_similar" );
      ?>

```

```
    </b>  
  </body>  
</html>
```