

UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA



**CERTIFICATE POLYGAMY
A MATTER OF TRUST**

Paulo de Mendonça Dias

MESTRADO EM SEGURANÇA INFORMÁTICA

Abril 2011

UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA



**CERTIFICATE POLYGAMY
A MATTER OF TRUST**

Paulo de Mendonça Dias

Tese orientada pelo Prof. Dr Adrian perrig
e co-orientada pelo Prof. Dr Nuno Ferreira Neves

MESTRADO EM SEGURANÇA INFORMÁTICA

Abril 2011

Resumo

O acesso a serviços disponíveis na Internet expõe os utilizadores a diversos ataques, tal como o *Man-in-the-Middle (MitM)*. As defesas para estes ataques, tais como autenticação mútua através de uma Public Key Infrastructure (PKI), baseiam-se em infra-estruturas complexas que os utilizadores não estão disponíveis para utilizar e suportar. A enorme aceitação de métodos de autenticação designados por “acto de fé” (leap-of-faith) ou “confiar na primeira utilização” (TOFU, trust-on-first-use), utilizado em implementações comuns de SSH e TLS/SSL, dão sinais claros da pré-disposição dos utilizadores em sacrificar a segurança em prol de uma melhor usabilidade. Aliás, este é um comportamento comum na vida quotidiana das pessoas. Se alguém se apresentar apenas com um cartão de visita, teremos tendência a confiar no seu conteúdo. Apenas desconfiaremos se, mais tarde, outra identificação for apresentada. Por outras palavras, confiamos nas primeiras credenciais apresentadas.

Esta temática foi abordada por soluções como o *Perspectives*, que fornecem autenticação tipo SSH com sondagens através de múltiplos caminhos/acessos, descrito em [1]. Através da observação e recolha das chaves públicas observadas ao longo do tempo por servidores espalhados geograficamente, designados por *Notários*, o *Perspectives* impede muitos dos ataques possíveis num cenário de TOFU. Um utilizador pode solicitar o historial de chaves de um determinado serviço, comparando-o à chave oferecida na utilização corrente, e com esse historial tomar uma decisão mais informada quanto ao aceitar uma chave que não exista em cache.

No entanto, o *Perspectives* assume um certificado por sítio, o que não é um pressuposto válido em muitos casos. Nesse caso, como pode o utilizador distinguir entre um certificado adicional introduzido pelo serviço a que está a aceder, e uma situação de ataque, em que o certificado está a ser fornecido pelo atacante? A presente tese endereça esta temática de poligamia de certificados, aumentando a visão dos *Notários* por forma a fornecer uma visão consolidada de diversos certificados. Adicionalmente, sugerimos alterações a alguns módulos do *Perspectives*, nomeadamente o módulo de sondagem (probing) por forma a lidar com questões tais como existência de mecanismos de caching acoplados aos serviços, pela utilização de, por exemplo, proxies.

Palavras chave: Certificados digitais, Perspectives, segurança, Certification Authority, SSL

Abstract

Users are vulnerable to attacks, such as Man-in-the-Middle (MitM) attack, whenever they resort to services in the Internet. Common defenses for these attacks, like mutual authentication based, for example, on a Public Key Infrastructure (PKI), rely on complex infra-structures that users are unwilling to support. Huge acceptance of simple methods like Trust-on-first-use (TOFU, also known as “leap-of-faith” authentication), employed by popular implementations of SSH and TLS/SSL, clearly indicate that users are prepared to sacrifice security for the sake of low-cost and more usable solutions. Moreover, this is a behavior that users are familiar with. If one meets a person who hands over some credentials, such as nickname, email address or even a business card, one will bind those credentials to that person in all future contacts, without initially asking for his or her ID. In other words, one trusts these credentials on the first time they are seen, and then uses them in all future interactions with that person.

This topic has been addressed previously in solutions like *Perspectives*, which provides SSH-style Host Authentication with Multi-Path Probing, as described in [1]. By observing and collecting the server’s public keys over time, maintaining them in a set of geographically disperse servers known as “Notaries”, *Perspective* thwarts many of the attacks that are possible in a TOFU scenario. A user can download such records on demand, comparing them with the current key provided by the site being accessed. Although not secure to all attacks, users can make a more educated decision on accepting or rejecting each certificate.

However, *Perspectives* assumes one certificate per site, which is a false assumption in some cases. So, how can users differentiate between a distinct, legitimate certificate provided by the site, and a fake certificate provided by an attacker? This thesis addresses this certificate polygamy issue, by enhancing the concept of the Notaries used in *Perspectives*, and provides a consistent view of a set of certificates to the user. Moreover, it suggests changes in modules like the probing module, to keep a clear and consistent observation of certificates, despite caching and reutilization made by components such as proxies.

By allowing the user (or, by company policies) to fine tune some configuration parameters, the proposed solution will provide different levels of confidence to the observed server’s public keys, thus satisfying distinct levels of security, or user proficiency.

Acknowledgments

A special recognition to the Carnegie Mellon University, to Faculdade de Ciências da Universidade de Lisboa, and to the CMU|Portugal program for providing the conditions that allowed my participation in this outstanding Master.

I would like to thank Professor Adrian Perrig for the opportunity to work on *Perspectives*, and his support to this thesis preparation during my stay in Pittsburgh, and Professor Nuno Ferreira Neves, without his guidance and support this writing would not have been possible.

To my family, for the wisdom to understand and support my absence.

A special recognition to Professor Paulo Veríssimo and Professor Miguel Correia, for their support and enthusiasm when things looked impossible.

Finally, thanks to Filipe Ribeiro for his help on the Figures and to Carlos Ferreira da Silva for more than I can detail.

Contents

Resumo	I
Abstract.....	III
Acknowledgments	V
Chapter 1 Introduction.....	1
1.1 Overview	1
1.2 Summary of the Solution.....	8
1.3 Organization of the document	10
Chapter 2 Background.....	11
2.1 General problems	11
2.2 Current status.....	20
2.3 The <i>Perspectives</i> approach.....	23
Chapter 3 Solution for Certificate Polygamy	27
3.1 Handling Multiple Certificates	27
3.2 Probing Module.....	28
3.3 Threshold and Quorum Parameters	31
3.4 Client Policies Enhancement.....	35
3.5 Cross-validation Protocol and Shadow Servers.....	37
Chapter 4 Testing and Analysis.....	39
4.1 Evaluation	39
4.2 Example attacks on LAN	39
4.2.1 Attack on a hot spot	39
4.2.2 Attack on home router.....	43
4.3 Probing and Caching	43
Chapter 5 Conclusions and Future work	47
5.1 Conclusions	47
5.2 Future Work	47
Annex I The Coupon Collectors Problem applied to digital certificates	49
Conventions	53
References.....	55

List of Figures

Figure 1 – One-time password generator key device.....	2
Figure 2 – Display card token generator.....	2
Figure 3 - Access from an insecure page to an https page for the user to request a digital certificate.....	3
Figure 4 - Legitimate link to request a digital certificate.....	4
Figure 5 – Attacker replaces the link to a malicious site.....	5
Figure 6 – Access to a site with a certificate issued by an unknown CA, using Microsoft IE 8.....	6
Figure 7 – Access to a site with a certificate issued by an unknown CA, using Mozilla Firefox 3.6.10.....	6
Figure 8 – SSH warning about a new certificate.....	7
Figure 9 – Sequence number in TCP packet.....	11
Figure 10 – Source route to circumvent failed node.....	12
Figure 11 – Architecture of the TCP/IP protocol suite.....	13
Figure 12 – Rogue Certification Authority.....	16
Figure 13 – Alert for detected certificate issue on IE 6.....	18
Figure 14 - 10-day Key History for caixaebanking.cgd.pt:443.....	21
Figure 15 - 200-day Key History for caixaebanking.cgd.pt:443.....	21
Figure 16 - 10-day Key History for bes-sec.bes.pt:443.....	22
Figure 17 - 200-day Key History for bes-sec.bes.pt:443.....	22
Figure 18 - 10-day Key History for citibank.com:443.....	23
Figure 19 – Geographical distribution of Notaries.....	24
Figure 20 – Worldwide geographical distribution of Notaries.....	32
Figure 21 – Worldwide geographical distribution of Notaries, by zone.....	33
Figure 22 – Geographical distribution of Notaries, by smaller zones.....	34
Figure 23 – A client (C) contacting shadow server (SH) for a shadow copy of Notary.....	38
Figure 24 – Login page for Internet access on a Hotel hotspot.....	40
Figure 25 – Source code for the login page for Internet access on a Hotel hotspot.....	40
Figure 26 – Sample of server source code.....	41
Figure 27 – Server challenge.....	41
Figure 28 – Client response to challenge.....	42
Figure 29 – Detail of client response to challenge.....	42
Figure 30 – Probing for certificates.....	43
Figure 31 – Probing results with 1 second delay between probes, on Citibank:443.....	44
Figure 32 – Another 10-day Key History for citibank.com:443.....	44
Figure 33 – Another 200-day Key History for citibank.com:443.....	45
Figure 34 – Another 200-day Key History for bes-sec.bes.pt:443.....	45
Figure 35 – Probing results with 10 second delay between probes, on Citibank:443.....	46
Figure 36 – Expected number of probes to retrieve all digital certificates.....	50

Figure 37 – Probing algorithm 50
Figure 38 – Probing algorithm with minimum number of probes 51

List of Tables

Table 1- SSL session establishment.....	18
Table 2- Probability of more than 1 certificate after p probes	29

Abbreviations

3G	3 rd Generation Mobile Telecommunications
ARP	Address Resolution Protocol
BES	Banco Espírito Santo
CA	Certification Authority
CGD	Caixa Geral de Depósitos
DNS	Domain Name System
DNSSec	Domain Name System Security Extensions
DoS	Denial of Service
EGP	Exterior Gateway Protocol
FTP	File Transfer Protocol
GPRS	General Packet Radio Service
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
ISP	Internet Service Provider
LAN	Local Area Network
MD5	Message digest algorithm 5
MitM	Man-in-the-Middle
NA	Notary Authority
NTP	Network Time Protocol
PKI	Public Key Infrastructure
PT	Portugal Telecom
QoS	Quality of Service
RFC	Request For Comments
SAN	Storage Area Network
SNMP	Simple Network Management Protocol
SP	Service Pack
SSH	Secure Shell
RIP	Routing Information Protocol

SSL	Secure Sockets Layer
TCB	Trusted Computing Base
TLS	Transport Layer Security
TOFU	Trust-on-first-use or leap-of-faith
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VM	Virtual Machine
WLAN	Wireless Local Area Network

Chapter 1

Introduction

1.1 Overview

Whenever surfing the Internet, our expectations and needs regarding authentication vary according to the nature of the task we are performing, and the value of the information we are accessing. We can categorize the authentication needs as:

▶ Anonymous access

As stated by Ed Schwartz [2], anonymity can be beneficial in a variety of applications. Web sites for whistle-blowers wanting to report abuses without fear of reprisal, like online support groups for sensitive issues such as victims of domestic violence, police abuse or corrupt political and public service employees. However, there is a thin line, when using anonymity, separating legitimate well-behaved users and malicious ones that resort to anonymity to perform malicious actions (such as spam or virus dissemination and later carry out attacks like distributed denial of service). Unfortunately, when it comes to protecting the consumer (at least the legitimate, well-behaved consumer), the balance favors the Internet Service Providers (ISP), allowing them to de-anonymize or black-list users for any reason at any time. Some proposals, such as the one from Ed Schwartz, establish the concept of contract anonymity, where the service provider guarantees anonymity and unlinkability as long as the user complies with the contract.

▶ One way authentication – Service to User

This is the case where the Service needs to authenticate to the User, but there is no need for the User to authenticate to the Service. This can occur, for example, while doing digital cash donations to a social charity organization. The donor wants to be sure that he or she is helping the legitimate organization, by accessing its site and not a fake one, but keeping his identity hidden.

▶ One way authentication – User to Service

This is the case opposite the previous one, that is, where the user needs to authenticate to the service, but there is no need for the Service to authenticate to the user. Voting for a TV reality show may prevent users from voting twice by authenticating them, but the service authentication overhead may not be necessary.

► Two way authentication

This is the more complex scenario, where both user and service need to authenticate to each other. Common cases include home banking, as well as finance and public administration, due to financial impact, credibility and damages, among others.

If one maps the previous categories of authentication needs to real world implementations, one will find different usage of each method. For example, home banking solutions are sometimes implemented without two way authentication or with weak forms of client authentication. In the following cases, which are representative due to the importance of the institutions, they resort to simpler types of authentication. They only provide weak security mechanisms for the user to authenticate to the Home Banking Service, instead of what we call *client authentication* in the sense of the establishment of a secure communication channel. Some examples of this client authentication include the use of a PKI where the customer possesses a digital certificate, possibly safeguarded by a Trusted Computing Base (TCB) or a less complex solution using a token generated by a physical device, such as the one used by Paypal, in Figure 1 or the display card from ActivIDentity, in Figure 2.



Figure 1 – One-time password generator key device



Figure 2 – Display card token generator

Is this simplicity of username/password to authenticate the customer acceptable, from the bank's perspective? Delivering a service vulnerable to so many attacks, when the image that they are trying to transmit to the consumer is security? Yes, because contractually, they assign to the user the responsibility with regard to any security breach, so their calculations are quite simple: it is cheaper to blame the user than to provide an adequate infra-structure to deliver two way authentication. Is this legal? Yes, for the time being. Do users allow it? By the success and use of home banking, we know they (we) do.

Consider as an example home banking applications. In most cases, the user is not aware of the changes in the pages as he navigates until the final destination. Let us analyze the access to the site of BES – Banco Espírito Santo, a leading bank in Portugal (starting from Figure 3).

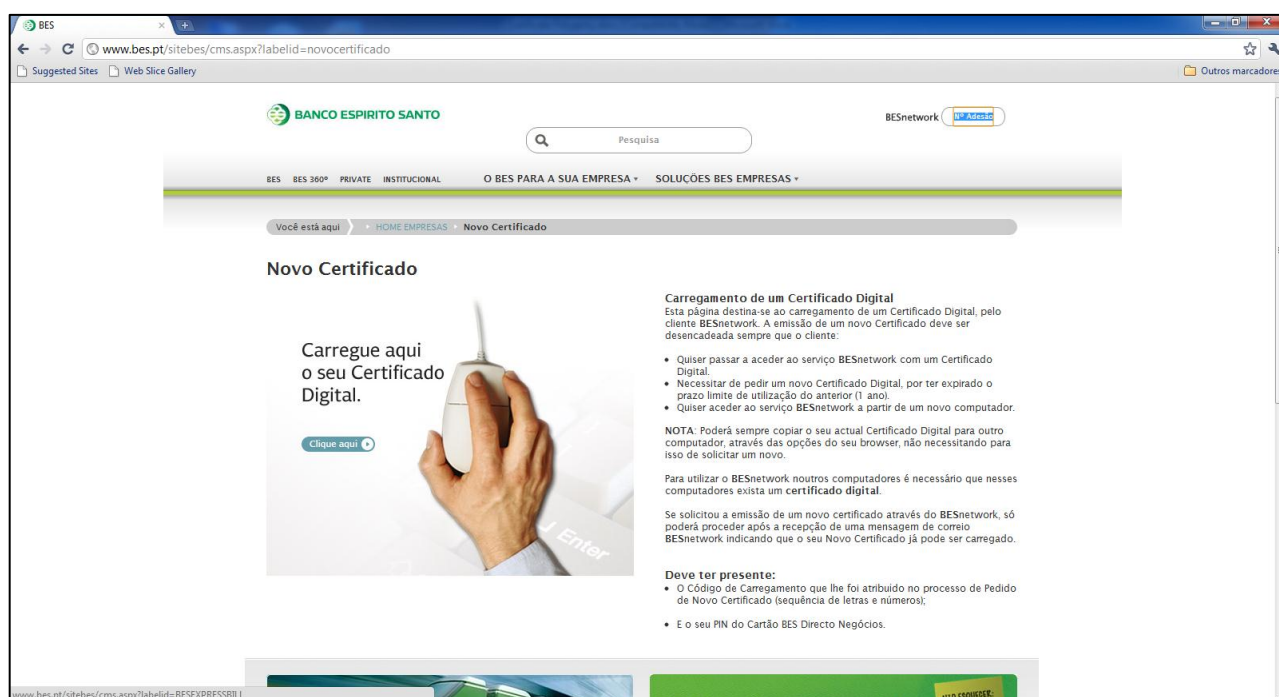


Figure 3 - Access from an insecure page to an https page for the user to request a digital certificate

By clicking on the image, the user would be redirected to <https://bes-sec.bes.pt/wclientes/cb/tplsp.asp> as illustrated in Figure 4.

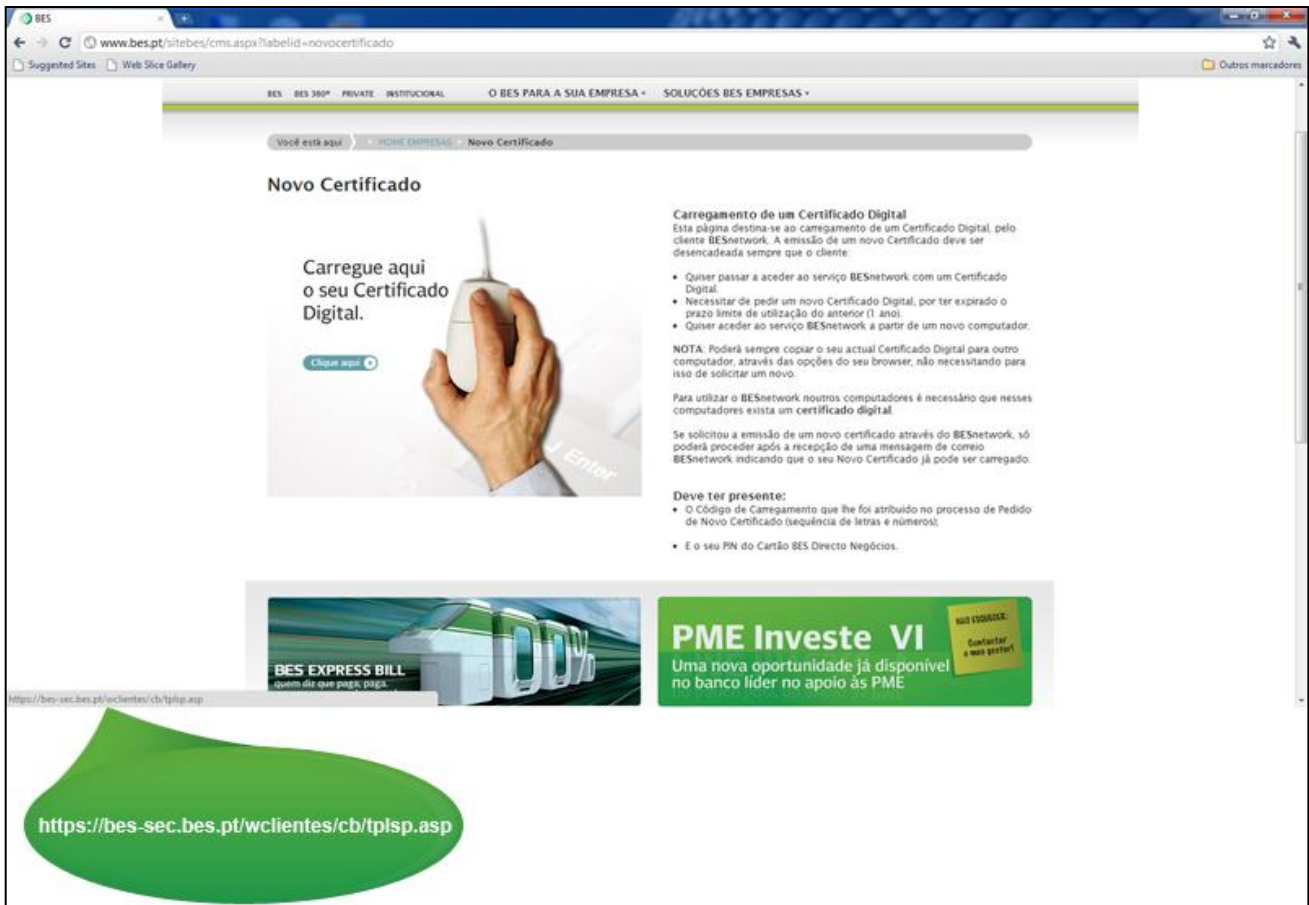


Figure 4 - Legitimate link to request a digital certificate

This simple step performed by the user is potentially vulnerable to a Man-in-the-Middle (MitM) attack. For example, malicious hosts on a Wireless Local Area Network (WLAN) can spoof ARP/DNS requests (e.g., ARPFrame [4] is a worm that injects malicious HTML code into local HTTP traffic) to valid users as soon as access to the network is obtained. Public access points and home routers may be poorly administered or have known vulnerabilities. For example, Pharming attacks [5][6], where simple vulnerabilities in home routers expose end hosts to “drive-by pharming” attacks that use DNS to redirect clients fake versions of security-sensitive websites, or the more sophisticated Dynamic Pharming Attack [7] that hijacks DNS, sending to the victim’s browser malicious JavaScript, which then exploits DNS rebinding vulnerabilities and the name-based same-origin policy to hijack a legitimate session after authentication has taken place. As a result, the attack works regardless of the authentication scheme used.

A simple DNS spoofing attack will enable the replacement of the link associated with the image to request the certificate, from the *original* <https://bes-sec.bes.pt/wclientes/cb/tplsp.asp> to something malicious, like <https://bes-sec.banco-bes.com/wclientes/cb/tplsp.asp> or similar, using an apparently harmless domain name, available and easy to register (see Figure 5). This will forward the user to a

malicious site, making the user vulnerable to various types of attacks, including the theft of their legitimate credentials.

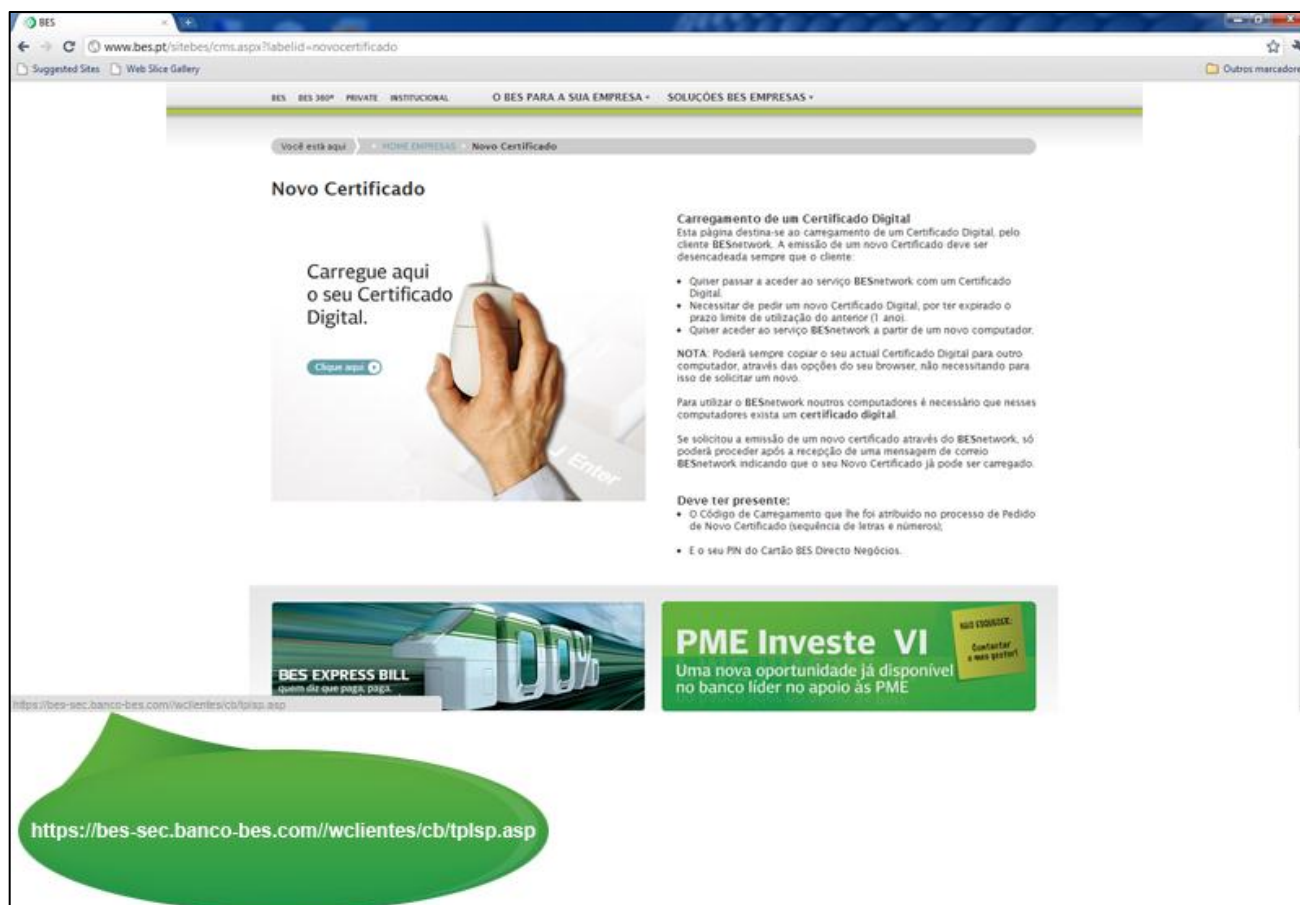
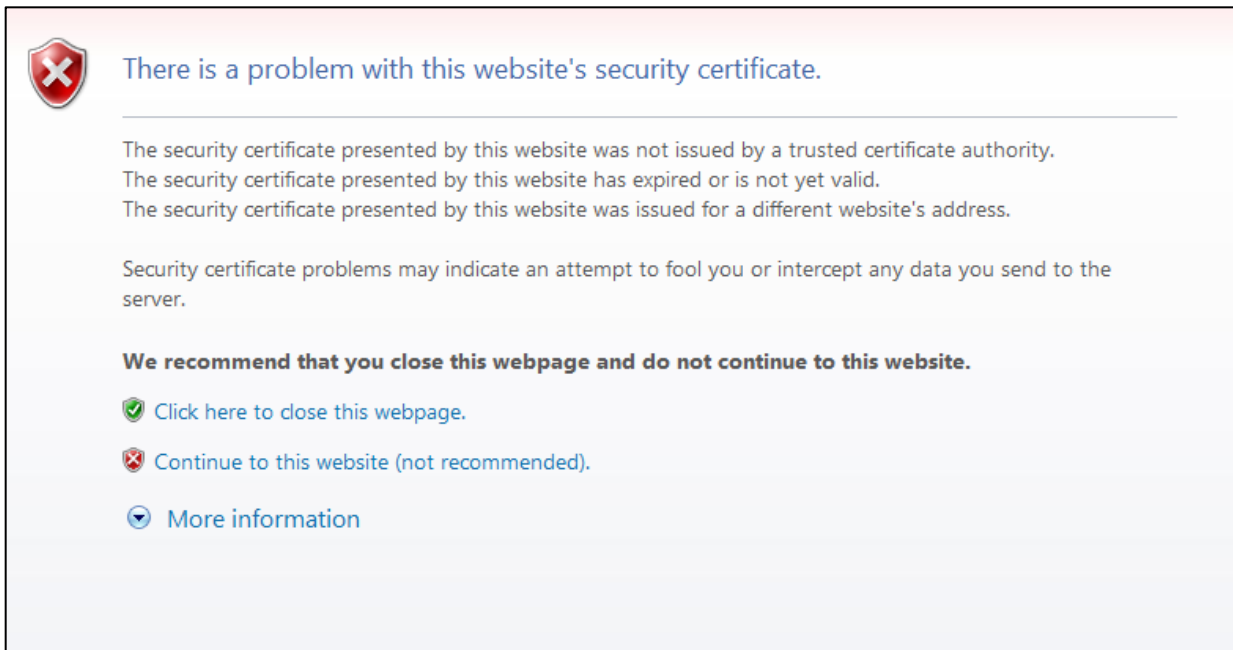



Figure 5 – Attacker replaces the link to a malicious site

In this case, the user will be provided with a certificate issued by the attacker, and since the domain is registered, and the certificate issuer would match the site, the user would be misled into accepting the certificate as a legitimate certificate issued by his bank.

Other related example is given by Bruce Schneier in [3], where users are redirected to “secure” sites with certificates owned by entities different than the ones managing the sites. According to PKI rules, any user should abort such connection, as this corresponds to a behavior equivalent to MitM attack (the situation that a PKI should prevent).

As users became aware of the security issues that might affect them, the problems mentioned above will become less prevalent. In the thesis, we will address a complementary problem which is related to the use of certificates that are not signed by a Certification Authority (CA) trusted by the browser. In this case, when the user accesses the site, he or she receives an alert similar to Figure 6 and Figure 7.



 **There is a problem with this website's security certificate.**

The security certificate presented by this website was not issued by a trusted certificate authority.
The security certificate presented by this website has expired or is not yet valid.
The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.




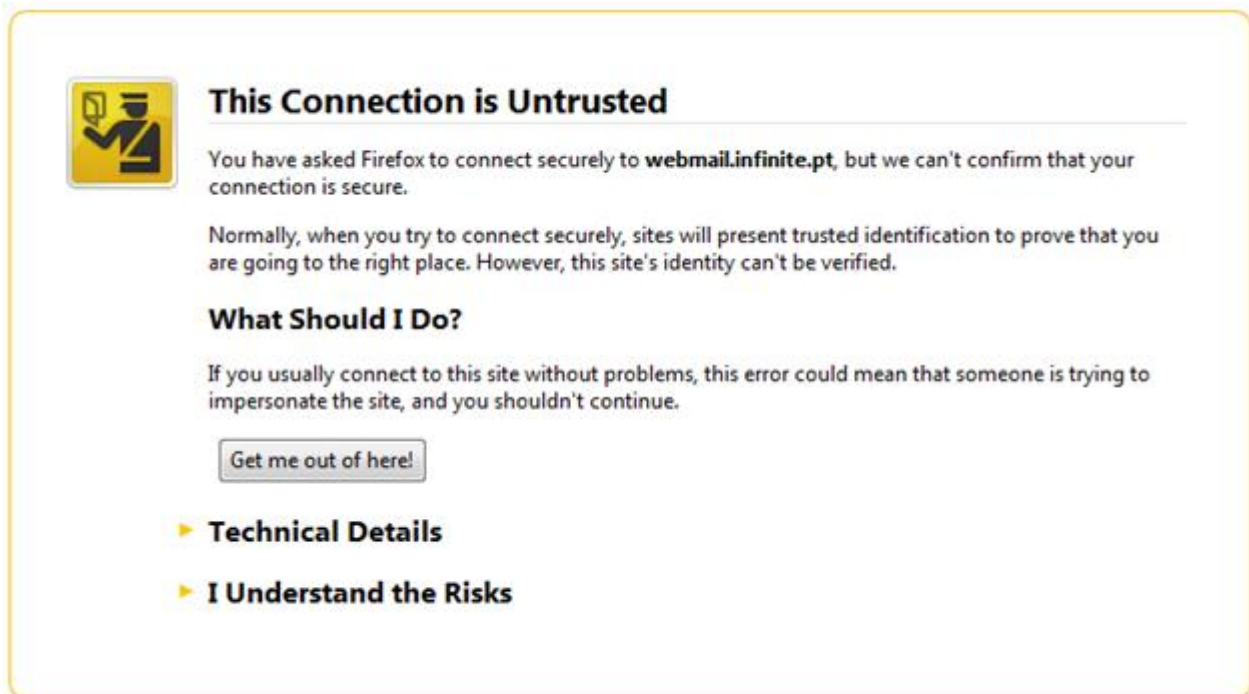

-  [Click here to close this webpage.](#)
-  [Continue to this website \(not recommended\).](#)
-  [More information](#)

Figure 6 – Access to a site with a certificate issued by an unknown CA, using Microsoft IE 8



 **This Connection is Untrusted**

You have asked Firefox to connect securely to **webmail.infinite.pt**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

- ▶ **Technical Details**
- ▶ **I Understand the Risks**

Figure 7 – Access to a site with a certificate issued by an unknown CA, using Mozilla Firefox 3.6.10

An equivalent warning is issued by popular SSH clients (see Figure 8). From the point of view of the users, these warnings are very unfriendly, since they provide no additional information, and require a difficult decision to be made – either trust the certificate and use the site, or do not trust it and then become unusable to carry out some action.

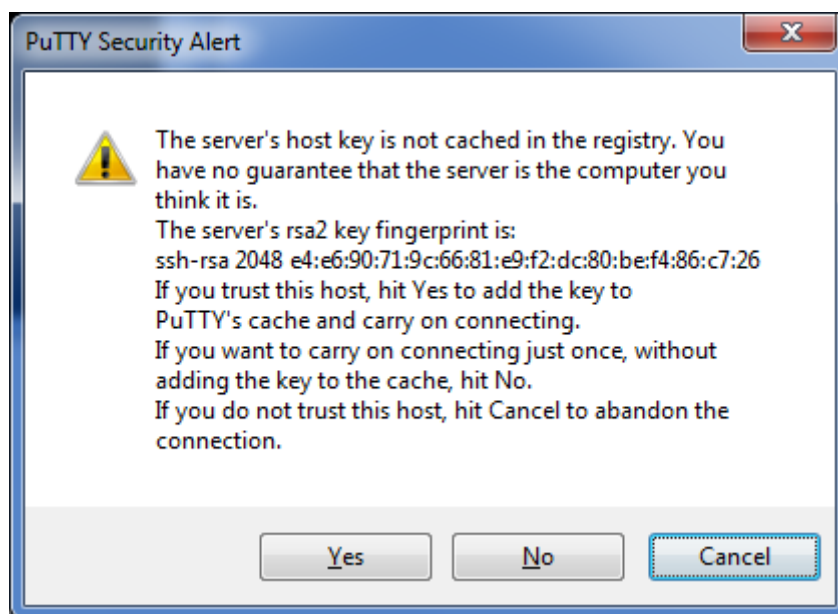


Figure 8 – SSH warning about a new certificate

In summary, there are basically three alternatives for a user to handle the Certificate received from the site:

► Public Key Infrastructure

Public Key Infrastructure relies on trusted entities (e.g., Verisign, UTN-USER) that issue certificates to entities that have been formally verified, usually with significant, manual effort. Users (and small companies) cannot afford to pay the high verification costs of such certificates, and end-up having to trust the browser policies to decide if a certificate is acceptable or not. Alternatively, the user is asked to indicate if he or she wants to accept the certificate, without much information to support the decision.

► Offline or off-path validation

The user might use a separate, secure channel, to validate the information. For example, he or she may use the phone and call the help desk to validate the 32 characters of the key fingertips as illustrated in Figure 8. This method ends up not being very useful in practice because often users are not willing to spend time with these calls.

► TOFU

Without a solution to protect users from totally insecure protocols, such as telnet, the SSH model of authentication emerged as an affordable and pragmatic solution to the user. SSH relies on verifying unknown public keys before accepting them as valid. Once a user accepts a public key, it is deemed valid by the client and cached locally to be used in future authentications with that same server. Although users should verify thoroughly each new key (for example, by using a separate secure channel or by validating certificate fingerprints), most users immediately accept the certificate on the first access. That is, the user is assuming that there is no adversary in that first connection and for this reason is called trust-on-first-use (TOFU). The same problem is present if no validation is made later on when a certificate changes, and a new one, different from the cached, is presented to the user. In both cases, during that period when the certificate is received for the first time, the user is vulnerable to Man-in-the-Middle attacks.

To help the user decide either to accept or reject the new certificate, *Perspectives* supplies to the user valuable additional information regarding the certificate (for example, SSH to access an end host or HTTPS to access a web site that uses a self-signed SSL certificate), leveraging views from multiple network vantage points, avoiding misinformation that could be introduced by localized attacks. It uses a set of publicly available servers (designated by Notaries), that keep track of the history of the public keys used by each network server, associated to a specific service, over time. By dispersing geographically the location of each Notary, a specific MitM attack only affects a specific Notary, allowing the remaining Notaries to provide the correct information (which can then be voted by the client). Much like to a real Notary, the Notary of *Perspectives* cryptographically signs (notarizes) statements saying that at time t it observed service S using public key PK_u .

1.2 Summary of the Solution

Our solution extends the scope of *Perspectives*, in two main areas. First, for globalization, extends the concept of Quorum acceptance based on political and/or geographical areas, introducing the concept of Zones, allowing the user the ability to accept a quorum achieved by Notaries localized in a minimum number of distinct political and/or geographical areas. Second, it handles cases where sites have a set of certificates, instead of the typical one-to-one relationship between site, service and certificate. Our improvements allows *Perspectives* to make a distinction between single and multiple certificate sites, applying the coupon collector solution based on the fact that at least two coupons (certificates) will denote a multiple certificate site, and using probabilities to validate the assumed population size. The proposed solution requires changes to the following modules of *Perspectives*:

► Probing

If the site/service has only a certificate, probing will determine if a certificate change has occurred. Only after a continuous, consistent observation of a new certificate, will a site be promoted to multiple certificate, since the Notary has to determine if the previous certificate is still observed (in this case, two certificates are seen consistently by the Notaries, denoting the introduction of another certificate) or it will no longer be seen, which is the case of a certificate replacement. Probing will be made to multiple certificate sites, until a level of confidence is achieved, based on the number of certificates observed versus the number of probes made so far. We will apply the Coupon Collector Problem to determine the level of confidence.

► Threshold and Quorum Parameters

We will suggest changes to the threshold and quorum parameters in order to consider single or multiple certificate sites, and also reflecting the case where caching may occur, which could prevent some certificates from being seen by specific Notaries.

► Client Policies

Client policies will be enhanced to reflect the zone concept, and the possible different values for threshold and quorum parameters, when encountering a single or a multiple certificate site. This will reflect the expectation of a higher consensus in single certificate sites, versus a multiple certificate site, since caching may hide some certificates based on the Notary IP address.

► Cross-validation Protocol and Shadow Servers

Shadow servers will replicate the certificates observed by each Notary. In single certificate sites, Notary and service will identify a unique entry in the shadow server. However, a multiple certificate site will contain as many entries as the number of certificates observed by each Notary/service. We will enhance the cross-validation protocol with an extra parameter to identify the certificate we are currently looking at.

1.3 Organization of the document

The rest of the thesis is organized as follows. Chapter 2 describes the current status of digital certificate handling, including an overview of *Perspectives*. Chapter 3 describes our solution. Our experiments and results are presented in Chapter 4, while Chapter 5 summarizes the conclusions and proposes future work directions.

Chapter 2

Background

2.1 General problems

Internet (in)security is a well-known problem that goes much further than simple physical eavesdropping, or altered or injected packets. These problems have been around for more than a decade, and they have been discussed in several papers, such as [11].

In this section, we address the following two questions:

- ▶ How insecure is the Internet today?
- ▶ Is the existent insecurity relevant to MitM attacks?

Amongst the most common vulnerabilities are:

- ▶ TCP Sequence Number Prediction

Initially described by Morris [12] in 1985, if the attacker can predict the sequence number in the TCP packet, he or she can construct a new packet and send responses to the host, without ever having established a connection with it. This allows, for instance, the attacker to spoof packets in the same network.

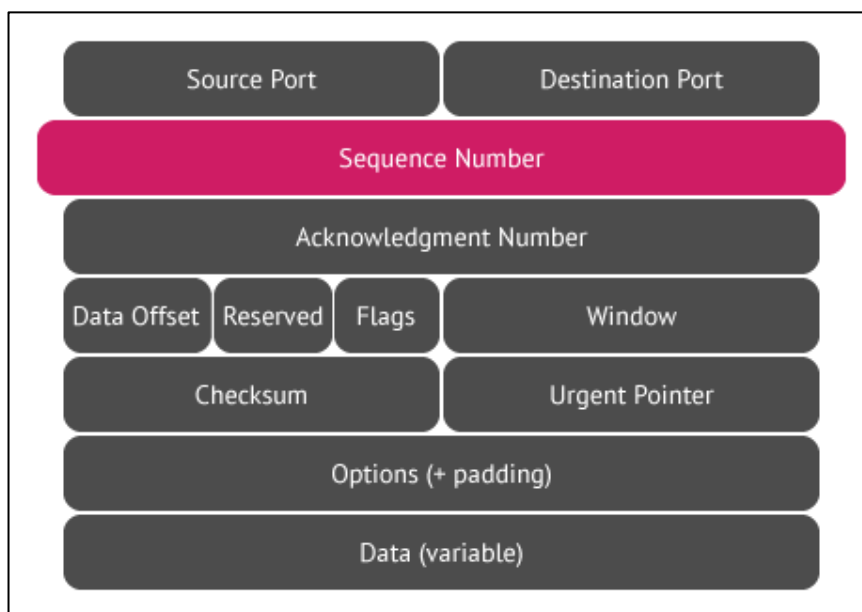


Figure 9 – Sequence number in TCP packet

► Source routing abuse

In source routing, the source specifies the nodes that packets should visit on the way to the destination, instead of using the default route. Source routing is useful to circumvent a failed node (Figure 15), or to choose a more appropriate path to comply with Quality of Service (QoS) requirements.

Since it is also expected that the destination replies follow the reverse path of the source route, an attacker can pick any address or alter the source route to include himself in the path.

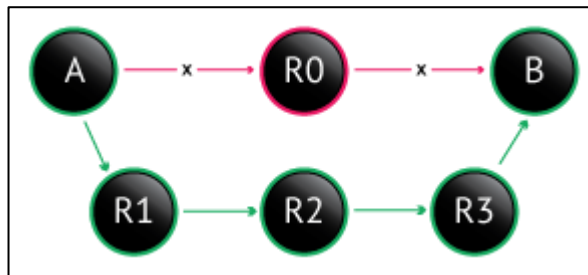


Figure 10 – Source route to circumvent failed node

► Routing Information Protocol (RIP)

RIP is used to propagate routing information on local networks. It is a simple protocol that allows an attacker to inject false or non-existing route information, leading traffic, for example, to him (eavesdropping the connection) or to a non-existing network (creating a black hole).

► Exterior Gateway Protocol (EPG)

Intended for communication between core network gateways, EPG is vulnerable to attacks such as impersonating that a gateway is temporarily down. Although this type of vulnerability is not the best to illustrate the cases of attacks at the edge, it shows the range of TCP Internet vulnerabilities.

► Internet Control Message Protocol (ICMP)

ICMP is a core protocol in the IP protocol suite (shown in Figure 16), and is used to report network errors when processing datagrams and provide administrative tools, status messages and reports. ICMP is neither secure nor reliable, lacks of any authentication mechanism and IP must act upon reception of ICMP messages. This makes ICMP extremely vulnerable and easily attacked. In particular, blind attacks can be made without specific knowledge of the connection's characteristics, enabling an off-path attacker to succeed with a very high probability.

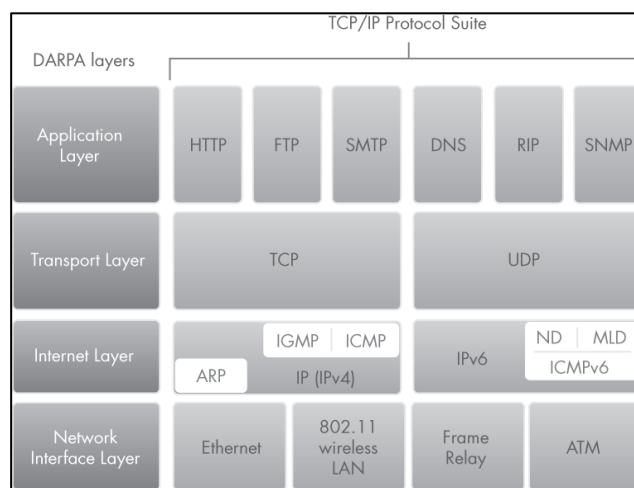


Figure 11 – Architecture of the TCP/IP protocol suite

► Finger Service

The Name/Finger protocol was designed to provide information on a particular computer system or a particular person at network sites. The goal was to provide information on other users of the network, such as their full name, phone number, mail address, etc. Information on who is logged-in was useful to check the availability of a person to meet. Running this service is giving away valuable information that can be misused, even if there were no vulnerabilities in the implementation. However this information can be used to crack passwords, as explained by Grampp and Morris in [14].

▶ Domain Name System (DNS)

DNS provides a distributed database for translating host names to IP addresses. Interfering with DNS is probably one of the most rewarding attacks, since it opens varied vulnerabilities, like denial of service, MitM and credential theft.

▶ File Transfer Protocol (FTP)

Although the protocol itself is not considered flawed, some aspects are potentially unsafe. Anonymous FTP is used often in the Internet, but care should be taken to protect those files. An infected file could be disastrous. Inherent to anonymous access is the problem of not knowing who has requested those files. When using authentication in FTP, the service relies on a user/password combination that for some applications might be considered inappropriate in terms of security. One-time password implementations are now becoming popular.

▶ Simple Network Management Protocol (SNMP)

SNMP is used for network management, and, as such, should be protected because it has access to network resources (routers, switches, servers, workstations, printers, racks, Storage Area Networks and much more). Although this principle is defended in the RFC 1157 - Simple Network Management Protocol (SNMP), it also allows weak implementations, including null authentication services. Here are some examples why protecting SNMP is considered, in areas like date and time:

▶ Time synchronization

From a security perspective, effective understanding of security incidents requires matching the events timestamps on all log files. Any discrepancies will complicate or sabotage legal proceedings.

▶ Software

There is a lot of software that requires accuracy to work, from development tools like make to RSA SecurID or Kerberos.

► Network Time Protocol

The Network Time Protocol (NTP) is a protocol for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks. Since a large variety of security algorithms and solutions rely on time, external control of it is a giant step towards insecurity.

Although specific defenses may be used to prevent some of these attacks, all of them would benefit from these generic protection mechanisms:

► Authentication

We need to rely on a more secure solution than the simple IP source address supplied by the protocol suite. Since we are avoiding complex solutions that rely on heavily managed infra-structures, one could use the same authentication mechanisms based on asymmetric keys and digital certificates, without the overhead of a full PKI implementation.

► Encryption

A suitable encryption implementation can be an effective mechanism to prevent most of the vulnerabilities outlined previously. Performing link-level encryption can even protect against physical intruders who connect directly to the wired or wireless medium.

► Trusted Systems

If we can manage all the systems involved in the operation we are trying to secure, we might have some confidence on the level of security available. However, when accessing global services in the Internet, most of the equipment we must rely on is outside our control. As we present later on, even home equipments are hard for a common user to guarantee its reliability. We must trust on mechanisms based on quorums to provide some level of trust on the provided information – this is the solution currently employed by *Perspectives*.

TLS/SSL and SSH provide a generic way for a user to connect to a remote server in a secure fashion. They provide a method to create a secure channel (authentication, integrity, and confidentiality) between the sender and receiver, which is able to mitigate most of the attacks discussed throughout this section.

These approaches, however, can still be attacked.

Popular applications like `dsniff` provide several tools, including `sshmitm` and `webmitm`, which implement active MitM attacks against redirected SSH and HTTPS sessions by exploiting weak bindings in PKI. Although a properly configured client should warn the user about issues related with the server's certificate, the attack is still possible. With a poorly configured client or a less stringent user, the attack becomes even easier.

A normal SSL session should authenticate the server to the client using a certificate. However, SSL does not require the client to authenticate to the server, which can create security flaws. An SSL client should warn the user if any of these conditions are met during a normal SSL session:

- ▶ The certificate was signed by a certificate authority that is not recognized

There are some reasons why the effectiveness of this warning is extremely unreliable. On one hand, some servers use self-signed certificates. Users have seen this warning quite often in the past, and tend to access the site despite the warning. Another reason is the legitimacy of the CA itself. Alexander Sotirov et al. [13] showed how to create a rogue certification authority certificate trusted by all common web browsers, by taking advantage of a weakness in the MD5 cryptographic hash function. This process, known as *collision*, allows the introduction of a rogue CA in the trusted list of the browser (Figure 12).

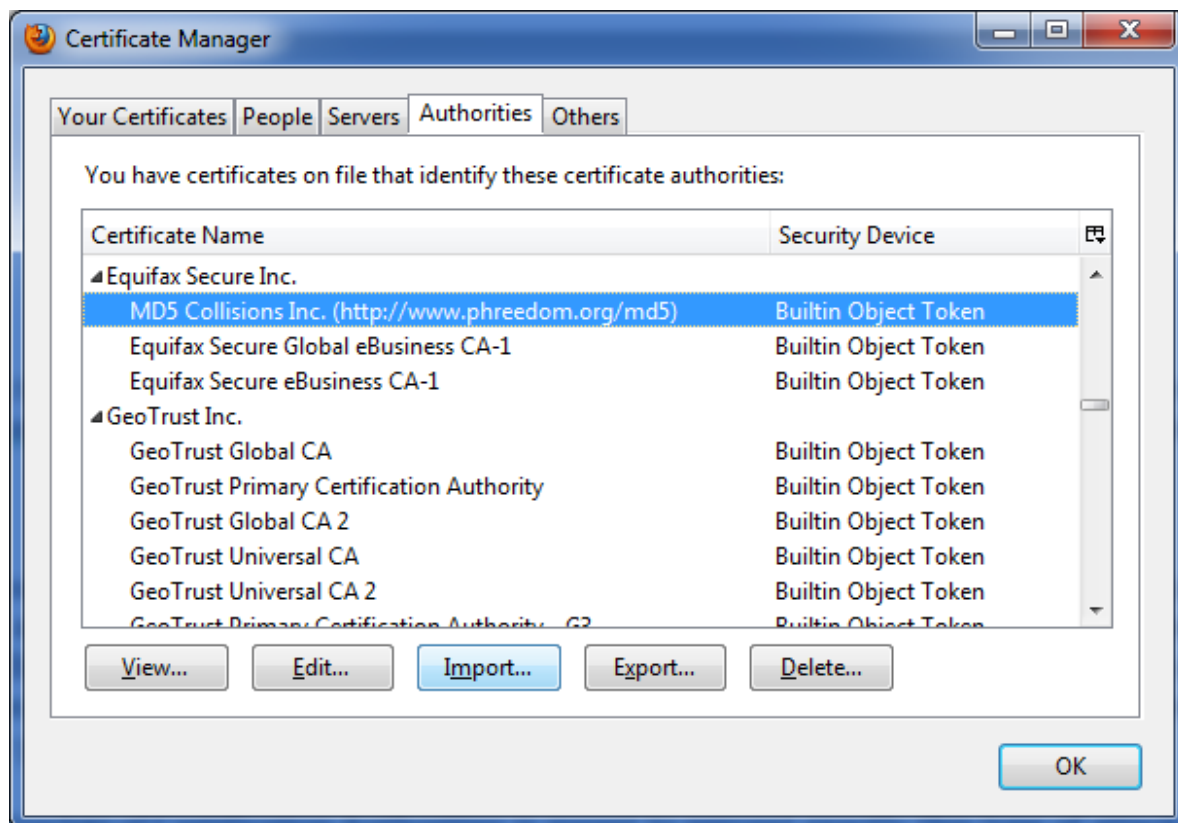


Figure 12 – Rogue Certification Authority

To complicate things even more for the user, the list of CA's is very long, with so many unknown entities, that is difficult to determine who to trust. Who are QuoVadis Limited, TürKie Bilimsel ve Teknolojik Araştırma Kurumu – TÜBİTAK or Sociedad Cameral de Certificación Digital – Serticámara S.A. appearing on Mozilla Firefox 3.6.15?

One can argue that users should rely on more well-known CAs like VeriSign or VISA, but their certificates tend to be more expensive, leading some servers to employ self-signed certificates. An alternative might be for local servers (such as a local bank) to use a local trusted CA. A Portuguese user might think of the Multicert CA, with renowned local stakeholders like SIBS (managing the ATM network, with more than 25 banks as their stakeholders), CTT (Portuguese Post-Office), INCM(National Press), PT-Prime and PT SGPS (both belonging to Portugal Telecom). But Multicert is not on the default list of trusted CAs of common browsers.

► The certificate is invalid or has expired

With such a warning no user should proceed with the connection, unless he is fully aware of a specific reason why this certificate is either invalid or expired. Probably it would expose the users to DoS, if an attacker manipulates the user's notion of time, but it is probably better to cancel and report the issue.

► The common name of the certificate does not match the DNS name of the server

Some online solutions do present certificates that do not match the name of the site. Although users may even be used to this behavior, browser policies should terminate immediately such connections.

User perception of SSL is also mistaken due to the way some versions of browsers handle a certificate, clearly stating that information exchanged with the site, using SSL, could not be viewed or changed by others (Figure 13).



Figure 13 – Alert for detected certificate issue on IE 6

Next, we briefly review the protocol used by SSL to create a connection, and indicate at each step potential problems that might occur while handling the certificates. Table 1 summarizes SSL negotiation, where shaded rows signal optional steps.

CLIENT	SERVER	PHASE
client_hello →		1
	← server_hello	
	← certificate	
	← server_key_exchange	2
	← certificate-request	
	← server_hello_done	
certificate →		3
client_key_exchange →		
certificate_verify →		
change_cipher_spec →		4
finished →		
	← change_cipher_spec	
	← finished	

Table 1- SSL session establishment

A `client_hello_message` is sent by the client to the server, including the list of supported cipher suites. The server responds with a `server_hello`, selecting an appropriate TLS version and cipher suite to use in the initial SSL session. The server also responds with its certificate, and finishes with a `server_hello_done` message. The client and server then create a symmetric key, and in the end each side sends a `change_cipher_spec` message to notify each other that subsequent information will be protected under the agreed key. Finally, each side sends a finished message to its peer.

The following actions may occur in each phase:

- ▶ Phase 1 – this phase intends to initiate a logical connection between peers and to establish the security capabilities that will be used. It is initiated by the client, with the following arguments:
 - ▶ The highest SSL version allowed by the client
 - ▶ A timestamp and a random number to prevent replay attacks during the key exchange
 - ▶ A session ID, used to indicate either a new session or the renegotiation of an existing one
 - ▶ Cipher Suite that contains a list of cryptographic algorithms supported by the client, with preferred one on the beginning of the list
 - ▶ A list of the compression methods supported by the client
- ▶ Phase 2 - Specific steps depend on the architecture deployed, namely the public-key encryption scheme used
- ▶ Phase 3 – The client validates the server certificate
- ▶ Phase 4 – Completion of the set-up of the secure connection, including that the key exchange and authentication processes were successful

Some problems that may occur during the exchange include:

- ▶ Downgrading

A poorly configured client (or a vulnerable one) may be tampered with to provide a Cypher Suite in Phase 1 omitting recent cryptographic algorithms, and only containing well known weak ones.

► Invalid Certificates

As we mentioned previously, a lesser astute user might be tempted to proceed with the connection, despite failed validation in Phase 3. We will describe next how this action can allow a successful MitM attack.

Let us review why the example we mention so often (for example, in Figure 3, in relation to the site of Banco Espírito Santo) is so dangerous.

As hypothesized by Moxie Marlinspike, in most cases SSL is not established directly through the use of https in the browser Uniform Resource Locator (URL), but instead through:

- Clicking on a link that contains an HTTPS URL
- Through a redirect from a HTTP site using response code 302, indicating the resource resides temporarily under a different URI, where the temporary URI is given by the location field in the response

These methods of redirection to the HTTPS site can lead to attempts to break, not the TLS/SSL connections itself, but the “switch” between the non-encrypted and the encrypted communications. The idea is to attack the transition from HTTP to HTTPS even before the establishment of SSL, and perform a MitM. It is therefore fundamental that the client has the capabilities to make a correct judgment while validating the certificate received from the remote server.

2.2 Current status

The following figures show the behavior of a single-certificate web site, in this case `caixaebanking.cgd.pt:443`, the site of the Portuguese State bank, Caixa Geral de Depósitos (CGD).

Figure 14 illustrates a period of time where only one certificate is observed. This means that, during this period, no attack was attempted (observed) on the path between the Notaries and the server, involving the presentation of a new certificate, and the owner neither replaced the old certificate nor added an additional one.

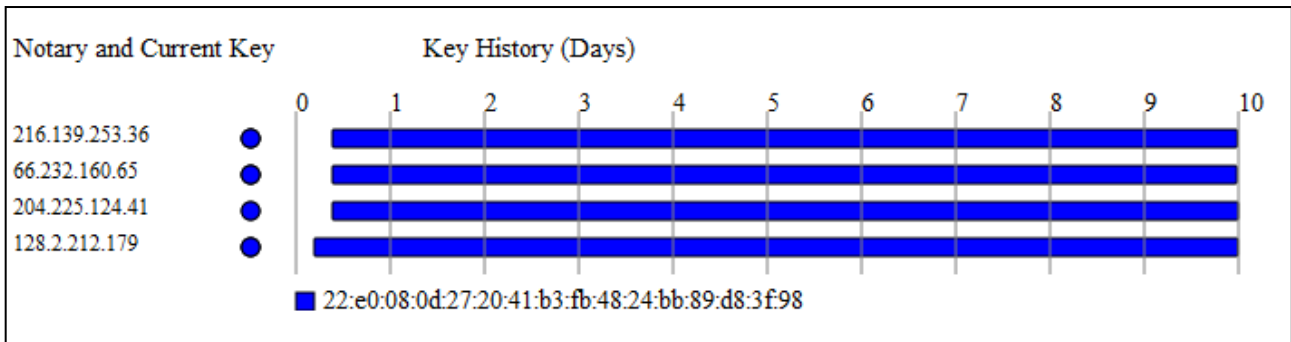


Figure 14 - 10-day Key History for caixaebanking.cgd.pt:443

Now, consider the observations for the same site, caixaebanking.cgd.pt, for a longer period of time (200 days).

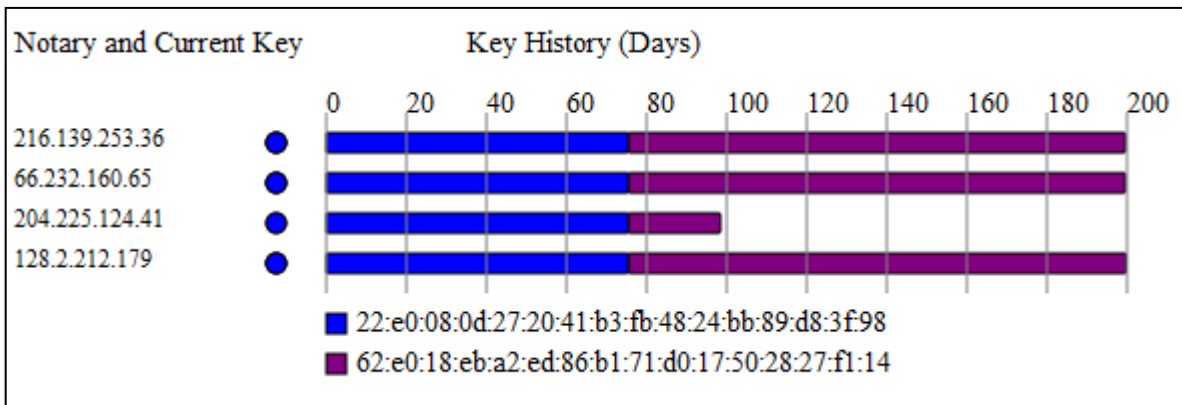


Figure 15 - 200-day Key History for caixaebanking.cgd.pt:443

Figure 15 illustrates the case where a replacement on the certificate was made. Notice that the older certificate does not appear anymore in the last 77 days, suggesting it has been removed and replaced by a new one.

Figure 16 displays a site with two certificates observed consistently during the last 10 days. Although only one certificate is observed by each site on some particular day, both certificates have been observed in other days. This situation lead us to conclude that this site has multiple certificates (two) that are consistently observed in space (by four Notaries) and in time (in this case, 10 days).

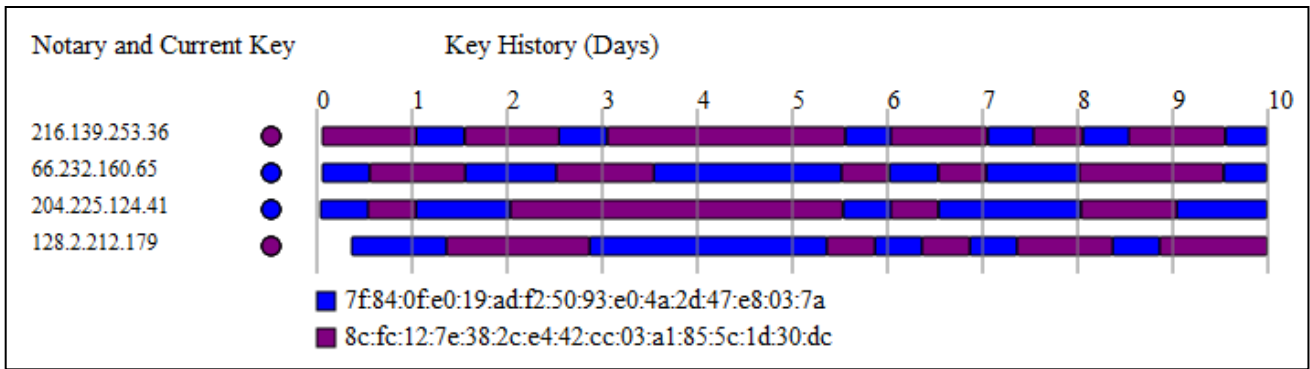


Figure 16 - 10-day Key History for bes-sec.bes.pt:443

Now, consider Figure 17, where the same site was observed for a longer period of time. This figure illustrates a particular situation, where two certificates appear consistently and two other appear sporadically in a specific reduced time frame. This can indicate either an attack or the temporary use of those certificates by the organization (for example, for testing purposes).

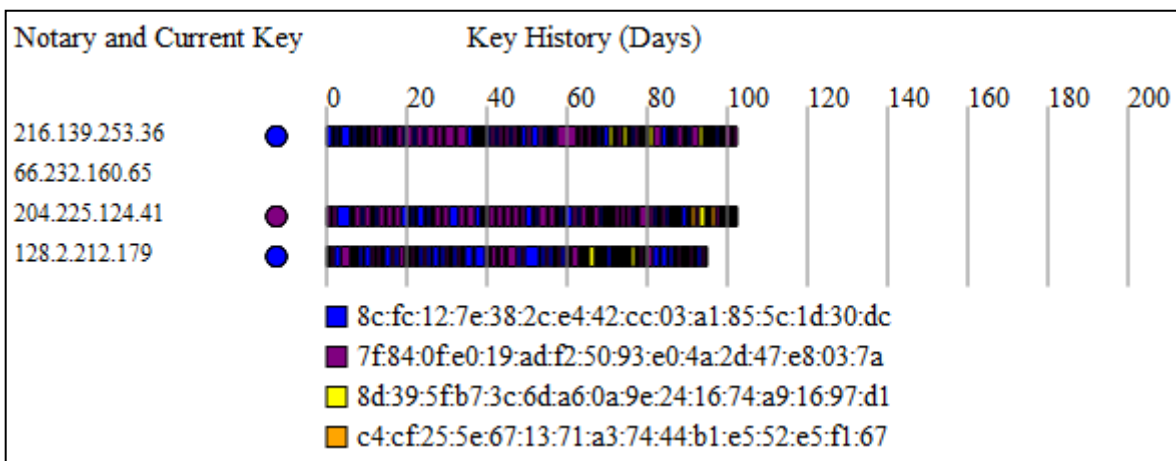


Figure 17 - 200-day Key History for bes-sec.bes.pt:443

In Figure 18 it is possible to observe the results for another site (citibank.com) with multiple (more than a dozen) certificates. In such cases, there is no consistency among observations. We can see that some Notaries are not even aware of keys observed by other Notaries.

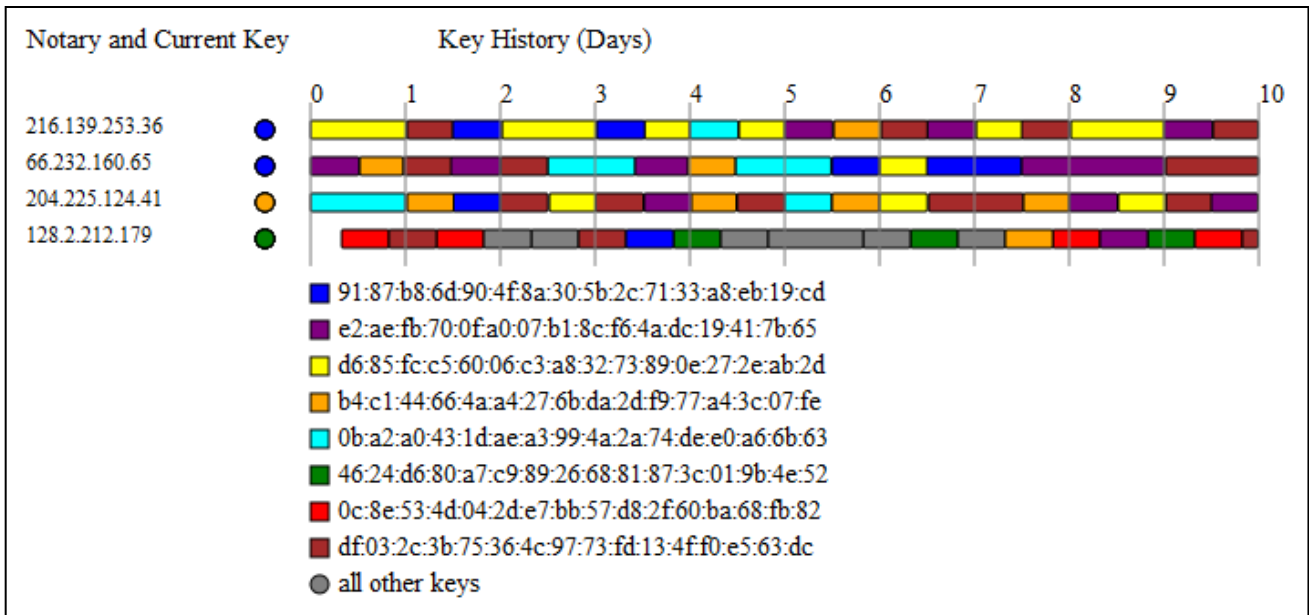


Figure 18 - 10-day Key History for citibank.com:443

2.3 The Perspectives approach

Perspective removes some of the uncertainty while in the TOFU process. By supplying the user with valuable information regarding the certificate provided by the service (for example, SSH access to an end host or HTTPS access to a web site that uses a self-signed SSL certificate), and leveraging views from multiple network vantage points, it avoids misinformation that could be introduced by localized attacks, and helps the user to determine whether to accept or reject the received public key certificate. It uses a set of publicly available servers (designated by Notaries), that keep track of the history of the public keys used by each network server, associated to a specific service, over time. The Notaries currently in use at the time of this writing are:

```

cmu.ron.lcs.mit.edu:8080 (128.2.212.179)
convoke.ron.lcs.mit.edu:8080 (204.225.124.41)
mvn.ron.lcs.mit.edu:8080 (66.232.160.65)
hostway.ron.lcs.mit.edu:8080 (216.139.253.36)

```

The geographical distribution of the sites is illustrated in figure 19.

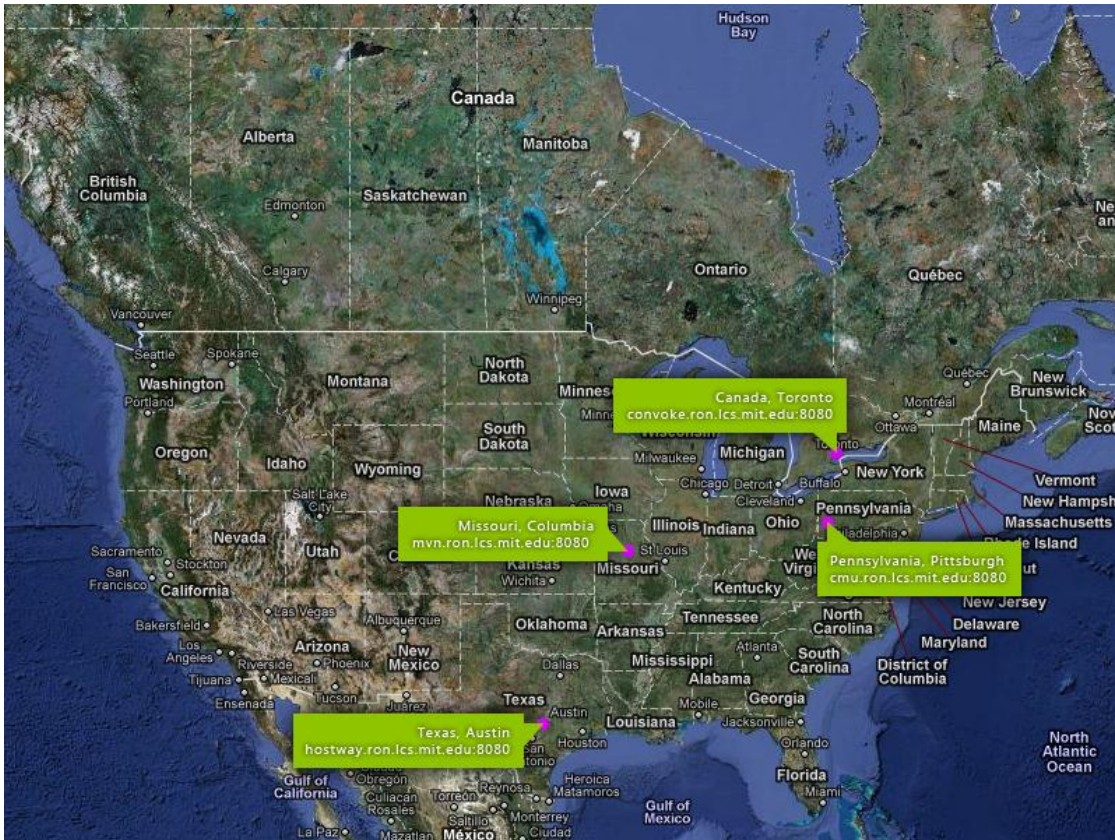


Figure 19 – Geographical distribution of Notaries

By dispersing geographically the location of each Notary, a specific MitM attack should only affect a specific Notary.

Similarly to a physical Notary, the Notary Server cryptographically signs (i.e., notarizes) statements saying that at time t it observed service S using public key PK_u .

When a client connects to a network service, it receives a public key (or certificate) in the reply. If the offered key has not been previously authenticated (i.e., it does not match an existing key in the client cache), the client must either accept the offered key, taking a security risk, or reject the key, losing the ability to communicate with the service. To obtain more information to support this decision, the client contacts the Notaries and requests from each one all observed key data for that service. Notaries get this information by periodically probing the servers for their certificate. This model has a simple deployment similar to SSH, since no central administration is required, working independently from both the servers that provide the certificates and clients requesting them.

The client then uses application-specific key-trust policies to interpret this data and accept or reject the key. These policies can be fine-tuned to each client, allowing customization to different user profiles or to match company policies, for example. These policies check for consistency between the offered key and the keys seen by each Notary, often allowing clients to distinguish between a legitimate key and an attack. For example, a key seen only by one Notary during last week will likely indicate a localized attack.

Perspectives is tolerant to attacks on any path, as well as in components of the Notary infrastructure itself. The only requirement is the server and the client being completely trusted, a standard requirement for host authentication.

The Notaries are organized in groups, and each group can be managed autonomously by an entity designated by Notary Authority (NA), with responsibilities to manage the Notary group, including determining the servers that are legitimate Notaries. Each NA has a public/private key, with the public key distributed to each Notary using an out-of-band mechanism. After the safe inclusion of a new Notary, the NA periodically publishes the list of certified Notaries, by signing with its private key, the IP and public key of each Notary.

The probing of keys is done daily by replicating the behavior of each service's client (e.g., SSH or HTTPS) until the Notary receives the offered service's public key, at which point it terminates the connection. Information about the public key is stored in the Notary database, where each entry contains a *service-type* (identifying the protocol) and a *service-id* (information required to contact the server, like hostname and port number). In addition, the database contains a history of the key observation over time, where each entry is designated by key timespan. Each *timespan* contains the observed key, and two timestamp entries containing the initial (t_{start}) and final (t_{end}) period where the key was observed. Timespan creation or update is ruled by the following protocol:

- ▶ if the observed key was observed in the last probing, t_{end} is updated
- ▶ if the key was not seen in the last probing, a new *timespan* is created with $t_{\text{start}} = t_{\text{end}}$

A client application is not required to contact the Notaries each time it accesses a service, but only when the offered key is not cached. This means that either the client never accessed this service before, or that the offered key is different from the cached one. To retrieve the observed data, the client queries the Notary by supplying the (service-type,service-id) pair.

The received information from the Notaries allows the client to define new rules (policies) to either trust the offered key (continuing the access to the service), or reject it and terminate the connection. Recall that this information includes the Notary (each Notary defines a different spatial location) and the *timespans* (defining consistency observation of the key over time).

Perspectives provides a framework to define spatial and temporal consistency, by defining the following properties.

Definition: For a set of n Notary servers, a service S , and a threshold q ($0 \leq q \leq n$) we say that a key K has quorum at time t iff at least q of the n Notaries report that K is the key for S at time t .

Intuitively, for values of q that are large relative to n , a key that has quorum indicates consensus among the observations made by the all Notaries at a single point in time. *Perspectives* use another threshold parameter to extend the concept of quorum into the temporal realm.

Definition: For a set of n Notary servers, a service S , and a quorum threshold of q , a key K has a quorum duration of d at time t iff for all t' such that $(t - d) \leq t' \leq t$ the key K had quorum with threshold q at time t' .

To detect malicious Notaries, Perspectives design includes data redundancy, a cross-validation protocol aiming to limit the malicious effect of a compromised Notary. This data redundancy capability is achieved by the role of a shadow server assumed by each Notary. A shadow server is a copy of the data of another Notary. Hence, when receiving key usage history from one Notary $N1$, the client has the capability of requesting that same information from the Notary $N2$, where $N2$ is the shadow copy of 1 , that is, $N2$ maintains a copy of the data observed by $N1$ (the client can even request shadow information from n Notaries, since there can be several shadow copies of each Notary). If the received information from the shadow copy differs from the information received by the Notary, the client can suspect that either one could be compromised.

Each Notary is responsible by the updating of its shadow server using a mechanism designated by *cross-validation protocol*. When a Notary contacts a service S , it updates its own local database, followed by the update of all shadow servers. Shadow servers are designed in a way to protect a malicious Notary from deleting previous entries. If a mismatch is detected, the shadow server stores both sets of data and signatures, signing both.

Chapter 3

Solution for Certificate Polygamy

3.1 Handling Multiple Certificates

Having multiple certificates provides considerable benefits. One of these benefits is related with redundancy. For load balancing, a web farm is usually created with multiple servers handling the client requests. Client requests are spread between available servers, using simple algorithms based on an evenly distribution of requests, or more sophisticated that use information from the network and the server itself to decide what server will handle each incoming request. If all servers used the same certificate, a compromised server (thus, a compromised certificate) would affect all servers in the web farm. Having a distinct certificate in each server allows the revocation of that single certificate, leaving all other servers fully operational, without any downtime. Sites like Citibank handle multiple certificates (Citibank has over a dozen different certificates). Therefore, *Perspectives* needs to catalog all the legitimate certificates, by using an approach like the coupon collector [8,9,10] to estimate how many certificates exist, or to decide when to stop probing.

The areas requiring investigation in order to *Perspectives* handle multi-certificate sites are:

► Probing modules

A probing module observes keys by connecting to the service and mimicking an ordinary client until it receives the service's public key, at which point it disconnects. With multiple certificates, this behavior has to change by performing multiple probes.

► Threshold and quorum parameters

Quorum parameters have to be adjusted for certificate polygamy, since Notaries can observe distinct valid certificates, but different from the one that the client is seeing. Considering a specific service S , for n Notary servers, and for a quorum q such that $0 \leq q \leq n$, q should be smaller compared to n , for servers handling multiple certificates. But this can impact single site certificates, so the suggested approach would be to separate thresholds for single and multi-certificate sites, possibly handling this at the client policy level.

► Client policies

On single-certificate sites, the replacement of a key is well defined in time. If two keys are observed by different Notaries in the same time frame, it indicates a possible attack. With certificate polygamy, this could be a consistent behavior, and therefore it is not necessarily an attack.

► Cross-validation protocol

With single-certificate sites, several checks are made considering one certificate, like timespans overlapping and common t_{start} values. This will not be an issue if care is taken while filtering with the current certificate.

► Database

A database entry is currently uniquely identified by the combination of a service-type, which identifies the protocol used to retrieve the key, and a service-id. Multiple keys already exist, but it is assumed valid keys do not overlap. With several certificates, although validations might have to change, no changes to the Notary database are expected, regarding multiple certificates. However, further investigation has to be done to evaluate the advantage of registering a flag indicating of the site/service is single or multi-certificate.

3.2 Probing Module

The probing module is a core component that needs to be aware of multi-certificate sites. For performance reasons, probing should be made in parallel, similarly to what is already done by the client querying n Notaries.

The following steps describe the new behavior of the probing module.

Step 1. Discover if the site has multiple certificates

This could be achieved through multiple probing, although observation of multiple certificates by several Notaries can also start this procedure.

For an estimated population of N certificates, the probability of observing O certificates after P probes is given by $(O/N)^P$. For example, for a population of 20 certificates, the probability p of observing 12 certificates after 30 probes is given by $(12/20)^{30}$, which is $\sim 2,2 \times 10^{-7}$. To discover if a site has multiple certificates, we can start by assuming that it has only 2 certificates, so we can calculate the probability using the formula $(1/2)^P$ (based on one observed certificate and an estimated population of 2):

Number of Probes	Probability %
1	50
2	25
3	12.5
4	6.25
5	3.125
6	1.5625
7	0.78125
8	0.390625
9	0.1953125
10	0.09765625
11	0.048828125
12	0.0244140625
13	0.0122070313
14	0.0061035156
15	0.0030517578

Table 2- Probability of more than 1 certificate after p probes

By defining a threshold Θ representing the minimum probability we want to achieve while probing, we can stop probing after a pre-defined number of probes, as detailed in Table 2. For $\Theta = 0.008\%$, then 14 probes would be sufficient to assume the site has only one certificate.

This procedure would only be started if we encounter a new certificate, since we do not require a daily probing to check for an increment of certificates, we do not observe any new certificate. This procedure will also allow determining if the certificate was replaced, since only the new one would be observed. We should recall that the procedure should not be done immediately, while the user is waiting for the access, since probing without delay would possibly generate cached responses from the server, returning the same certificate, as explained in section 4.3. Probing and Caching.

Step 2. If the site uses a single certificate, the current behavior still holds

In this case, no changes are required to the probing module.

Step 3. If the site has multiple certificates, we need additional probing:

Step 3.1. Estimate the number of certificates (N) and probe them. We suggest considering the same number of certificates in the beginning of the probe.

When determining the population estimation, one of two scenarios may occur:

- a) Service Type/Service-ID seen for the first time

We will need to apply the coupon collector solution considering the observed population size. Examples are presented in Annex I.

- b) Daily probing

For daily probing we assume the number of certificates to be N, the last observed value, and apply the coupon collector solution to obtain a coverage δ (that is, we make an observation of at least $\Theta\%$ probability).

While doing this probing, the following 4 situations are expected:

- i) # of certificates is correct. This is the most common case
We do not see a lot of collisions and no new certificates

- ii) # of certificates has declined

We infer this case when we see a lot of collisions

- iii) # of certs increased

This will be the case when we see a new certificate

- iv) # certificates is the same, but they were replaced

We see simultaneously new certificates and a lot of collisions

Annex I provides additional details and sample results to apply The Coupon Collector Problem to digital certificate probing.

3.3 Threshold and Quorum Parameters

In a single certificate (or public key) environment, *Perspectives* provides additional security to the users by allowing them to have an enhanced view of the site certificate, both in space and in time. This is achieved by viewing the certificate from distinct vantage points in space and in time.

Perspectives introduces the following definitions:

Definition: For a set of n Notary servers, a service S , and a threshold q ($0 \leq q \leq n$) we say that a key K has quorum at time t iff at least q of the n Notaries report that K is the key for S at time t .

Intuitively, for values of q that are large relative to n , a key that has quorum indicates consensus among the observations made by the all Notaries at a single point in time. Wendlandt, Andersen and Perrig [1] use another threshold parameter to extend the concept of quorum into the temporal realm.

Definition: For a set of n Notary servers, a service S , and a quorum threshold of q , a key K has a quorum duration of d at time t iff for all t' such that $(t - d) \leq t' \leq t$ the key K had quorum with threshold q at time t' .

In multiple certificate sites, the observed behavior should not change considerably when compared with single certificate sites. However, due to possible caching of replies when a client Notary probes the certificate, it is expected that a smaller set of Notaries report seeing a specific certificate. If a site caches the IP address of the Notary probing for the certificate, it will reply the same certificate in a future probe. This will result in this particular Notary being unaware of other valid certificates for the probed site. When a client later contacts the Notaries requesting information about the certificate, it is expected that a larger number of Notaries do not know of the existence of that particular certificate, when compared to a situation where caching does not occur. If the threshold q remains unchanged, many multiple certificate sites would trigger alerts on the browser that probably could lead to the rejection of the certificate.

To address this issue, one possible solution would be to decrease the value of q . However, this solution would have the following disadvantages:

- ▶ in single certificate sites, lowering the threshold q will raise the number of false negatives, since only a smaller set of Notaries would have to report the existence of a particular certificate to render it valid to the user;

► in multiple certificate sites, it is admissible to have a lower value of q , but additional care should be taken, namely to ensure a higher geographical distribution of the Notaries. This could increase the difficulty of the adversary to perform a MitM attack.

We suggest the inclusion of an additional threshold, z , to define the geographical zone of the Notary. Figure 20 already demonstrates the existing spatial redundancy. Even in a single certificate site solution, the geographical distribution can be expanded to a worldwide area.



Figure 20 – Worldwide geographical distribution of Notaries

The reason we expand this concept of spatial redundancy, even in the standard *Perspectives* solution for a single certificate site, is to leverage the concept to worldwide usage. This has two main advantages:

- Raises redundancy to a worldwide level

For example, a problem affecting the US servers (or link) would render *Perspectives* useless in Europe, if all Notaries are located in the US. If European users can also rely on Notaries placed in Europe, Asia and Australia, they still might gather a quorum large enough to trust the offered key

- Eliminates suspicions of Notaries based only in one country or region

Potentially, users in China would not trust a set of servers located in the US, or vice versa. Consensus through a quorum of servers located in distinct geographical areas is a more trustworthy scenario (at least, for a relatively large number of users) than the same number of quorum servers in the same area

The next step, illustrated in Figure 21, is to include each Notary location in a predefined Zone, according to a geographical criterion we previously determined. Since the geographical distribution aims at avoiding localized attacks, this criterion should be set considering the available Internet links, since compromising a link will compromise all the Notaries dependent on that particular link.



Figure 21 – Worldwide geographical distribution of Notaries, by zone

We now extend the criteria of acceptance of a certificate by requiring that, amongst the n Notaries that consistently see it, be localized in at least z Zones. This threshold, z , should be tuned considering the required policy and the number of available Z zones.

Going one step further, and considering an increment on the number of available Notaries, we can extend this concept to localized geographical distribution. For example, taking the European zone, we could further divide it into subzones. Surely the density of countries in Europe is high, when compared to, for example, Australia.



Figure 22 – Geographical distribution of Notaries, by smaller zones

We can now introduce the updated criteria.

Definition: For a set of n Notary servers, Z zones ($Z \leq n$), a service S , and a threshold q ($0 \leq q \leq n$) we say that a key K has quorum at time t iff at least q of the n Notaries, localized in at least z of the Z zones, report that K is the key for S at time t .

3.4 Client Policies Enhancement

Geographic zones can be used to provide additional information to the user, allowing him to take a more informed decision. We present for comparison some behavior scenarios with and without zone information. We use the term *cached key* to refer to a key already cached by the client, and the term *cached certificate* to refer the ability of a server providing a specific service to present the same certificate key based on the requester Notary IP address.

Single certificate site: No-Zone aware behavior, no server key cached

Although the key is not cached, it does not necessarily mean that the key is compromised or a MitM attack is in progress. If the user is accessing the service for the first time on her local machine, this is the expected behavior. However, if the user accesses this service on a frequent basis, she might be suspicious, but it does not mean that an attack is in progress. It may be that the provider of the service just renewed the certificate or introduced an extra one. The user should try to understand if the key has been observed by many Notaries, regardless of their physical location, and the duration of the key history. If the key replacement and quorum duration is satisfied, the policy provider may supply a message like (similar to the single certificate site version of *Perspectives*):

Key seen consistently for the past d days

If the observed quorum or duration does not achieve the minimum values defined by the policies, the user might be warned with a message like:

SUSPECTED ATTACK: Offered key is NOT consistent. Only X of Y Notaries currently see it.

WARNING: Server key has only been seen consistently for the past d days.

We may be facing either a new certificate or an attack. It is up to the user to decide the appropriate action to take. An experienced user might validate the key using an alternative channel while a non-expert user could be advised to reject the key and try later on, in the expectation that, during that time, more Notaries will also observe the key.

Multiple certificate site: No-Zone aware behavior with offered key different from the cached key

In this case, we can no longer assume an uncached key as a key change, but the behavior is not too different from a single certificate site because the space and time policies still apply. If we face a scenario where there is a proxy between the server and the Notary, which does caching, then it will be much harder to get a quorum. The main difficulty is that when the Notary probes the server, it will get the same cached certificate from the proxy, instead of a random certificate from the full set. In this case, we can expect more policy alerts, and as previously explained, one solution would be to lower the quorum value.

Lowering the threshold q will prevent additional rejections due to non-achievement of quorum, eliminating the effect of caching to reach quorum consensus. To prevent this change from affecting single certificate sites, we could introduce different quorums for single (q_s) and multiple (q_m) certificate sites, where $q_m \leq q_s$. However, this implies that there is a policy where $q_m \leq q_s$, which could only be effective with q_m and q_s defined for each site, according to their role. This solution implies a higher administration overhead. The administration could be too complex because it would have to be done at the server level, not at the Notary. As an alternative, we present the zone-aware alternative, with much less administration effort.

Single certificate site: Zone aware behavior

Compared with the scenario of no-zone aware behavior with single certificate site, no server key cached, security is only improved if, for the user, there is a special significance of a broader geographical location where certificates (hence, Notaries), are observed. Having defined z (zone threshold) and Z (total number of defined zones), the user might see a warning like:

WARNING: Key seen consistently for the past d days but only observed in X out of Z zones

But the biggest challenge will be to differentiate between a single certificate site doing a certificate replacement, and a single site starting to use a new certificate. In the latter case, it means that it is no longer a single certificate site, but it has been promoted to a multiple certificate site. However, this distinction should be done at the level of the probing module. The user she could receive a message like:

SUSPECTED ATTACK: Offered key is NOT consistent. Only X of Y Notaries currently see it in z of Z zones.

WARNING: Server key has only been seen consistently for the past d days and previous key is still in use

If the probing module considered that the observed key is an additional one (not a replacement) then at this point the site would be already considered a multiple certificate site and the policy would not consider the use of the previous key an issue.

Multiple certificate site: Zone aware behavior

As compared with the alternative of different qs and qm to distinguish sites with or without certificate caching, the zone-aware approach has the benefit of simplicity, because qs and qm would be parameters at the server level, while zone is at the Notary level. Obviously, the number of Notaries is significantly less than the number of servers, and zone maintenance would be managed as part of the Notary administration.

The behavior would be similar to the previous case, excluding the check made on the use of the previous key.

3.5 Cross-validation Protocol and Shadow Servers

To protect clients from accepting malicious data from a compromised Notary, each Notary keeps images of their data in replicated servers known as Shadow Servers. When a client contacts a Notary, it also contacts r shadows (defined by the client policy) to confirm data, and detected inconsistencies that would allow the detection of untrusted Notaries.

Currently, in single certificate sites, when the client contacts a shadow server for a copy of the observed key data for service s , it specifies the IP address of the Notary as well as the requested service. The shadow server replies with a service entry containing the key data and the signature created by the Notary, along with the shadow server signature of that data.

Since multiple certificates are expected, in the current version of the cross-validation protocol, to reflect key history, no changes are required. Figure 23 details the protocol, where the client specifies the IP address of N and the *service-id* from the original query to N . The shadow server replies with a service entry O_N (observed key data and signature) created by N , along with the shadow server's signature over that data.

Cross-Validation Protocol: $C \rightarrow SH : N, s$ $SH : DB_n = get_replica_datbase(N)$ $SH : O_N, \{O_N\}_{K_N^{-1}}, \{O_N, \{O_N\}_{K_N^{-1}}\}_{K_{SH}^{-1}}$
 $\quad \quad \quad = find_service_entry(DB_N, s)$ $SH \rightarrow C : O_N, \{O_N\}_{K_N^{-1}}, \{O_N, \{O_N\}_{K_N^{-1}}\}_{K_{SH}^{-1}}$

Figure 23 – A client (C) contacting shadow server (SH) for a shadow copy of Notary

N's observed key data for service s

Chapter 4

Testing and Analysis

4.1 Evaluation

In this chapter we evaluate how easy it is to perform a MitM attack on a LAN segment of an end user, our “victim”. The attack we want to perform depends on the access to the LAN where victim is located, and therefore we exploited some common vulnerabilities to get this access. First, we selected a hotspot at a hotel, and tried to obtain illegitimate access, without the need to identify ourselves with credentials. These environments are not familiar to users, since different hotels have different solutions, so any disturbance due to the attack would not generate additional suspicious to the victim, as a different behavior at his home or office would.

In the second part of the chapter, we probed sites with multiple certificate sites to understand how random certificate delivery is, and to test the effect of caching. These tests allowed us to conclude that the solution proposed in the previous chapter can be used to solve the problem of multiple certificates.

4.2 Example attacks on LAN

We tested a number of scenarios to conclude if the vulnerabilities are feasible to explore, with a relatively small effort. We selected a public place (hotel hotspot), home wireless networks, and, once access is possible, attacks on DNS in order to facilitate MitM attack.

4.2.1 Attack on a hot spot

The idea was to test unauthorized access on a hotel hotspot. A paid hotspot was found that could provide access to an open wireless network. Everyone in range could get access to the wireless LAN. Once connected, access was redirected to a login page, where the user was requested to supply a user/password to gain Internet access.

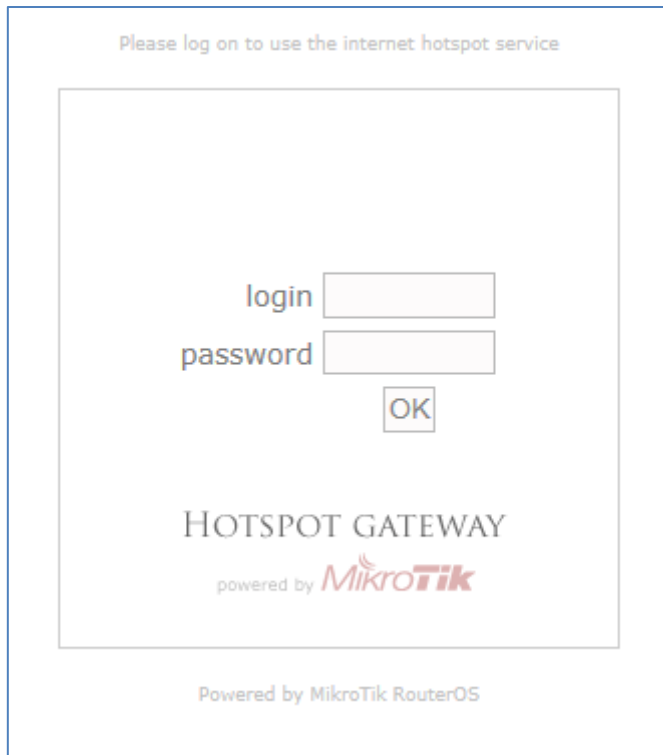


Figure 24 – Login page for Internet access on a Hotel hotspot

For our experience, it would be enough to get access to the wireless network, and spoof it to obtain other user's traffic information. By running Wireshark one could eavesdrop the network. However, to perform a successful MitM attack, one would need Internet access to forward the request. Of course that could be achieved with a 3G or GPRS connection, but we assume attackers do not want to use their own resources. So, we tried to get full access to the network, including Internet access.

We then viewed the login page source code.

```
<SCRIPT type="text/javascript" src="http://192.168.88.1/md5.js"></SCRIPT>
<SCRIPT type="text/javascript">
  <!--
    function doLogin() {
      document.sendin.username.value = document.login.username.value;
      document.sendin.password.value = hexMD5('\157' +
        document.login.password.value +
        '\305\122\134\207\247\125\147\041\161\330\060\127\119\254\224\130');
      document.sendin.submit();
      return false;
    }
  //-->
</SCRIPT>
```

Figure 25 – Source code for the login page for Internet access on a Hotel hotspot

We can easily conclude that the server sends a challenge with which the client builds a string including the supplied password. We can even see the server full page code from the link referenced in the source code in Figure 26, <http://192.168.88.1/md5.js>.

```
function strw2binl(str)
{
  var nblk = ((str.length + 4) >> 5) + 1 // number of 16-word
  blocks
  var blks = new Array(nblk * 16)
  for(var i = 0; i < nblk * 16; i++) blks[i] = 0
  for(var i = 0; i < str.length; i++)
    blks[i>>1] |= str.charCodeAtAt(i) << ((i%2) * 16)
  blks[i>>1] |= 0x80 << ((i%2) * 16)
  blks[nblk*16-2] = str.length * 16
  return blks
}

/*
 * External interface
 */
function hexMD5 (str) { return binl2hex(coreMD5( str2binl
(str))) }
function hexMD5w(str) { return binl2hex(coreMD5(strw2binl
(str))) }
function b64MD5 (str) { return binl2b64(coreMD5( str2binl
(str))) }
function b64MD5w(str) { return binl2b64(coreMD5(strw2binl
(str))) }
```

Figure 26 – Sample of server source code

We spoofed network traffic to see a challenge from the server (Figure 27), and the matching resolution from the client (Figure 28 and Figure 29).

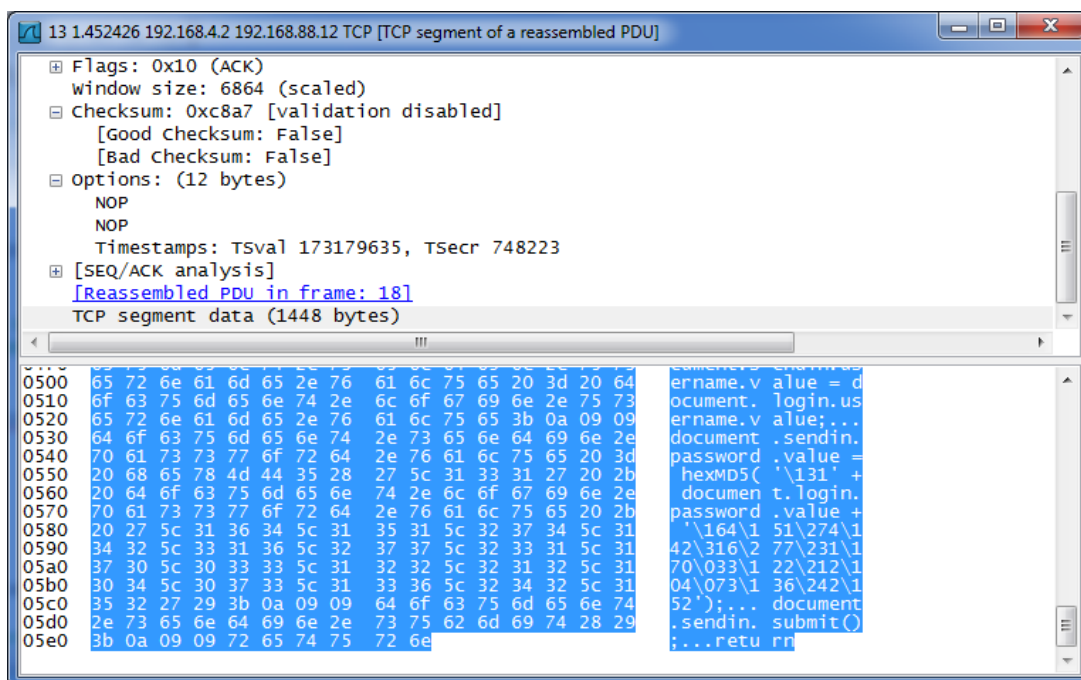


Figure 27 – Server challenge

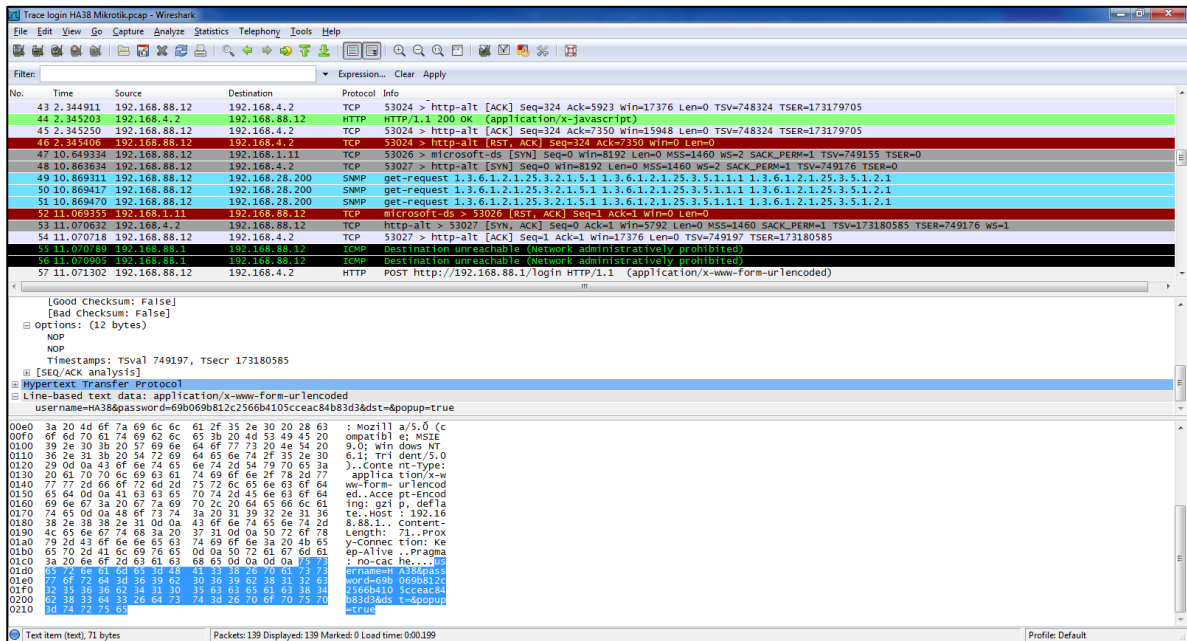


Figure 28 – Client response to challenge

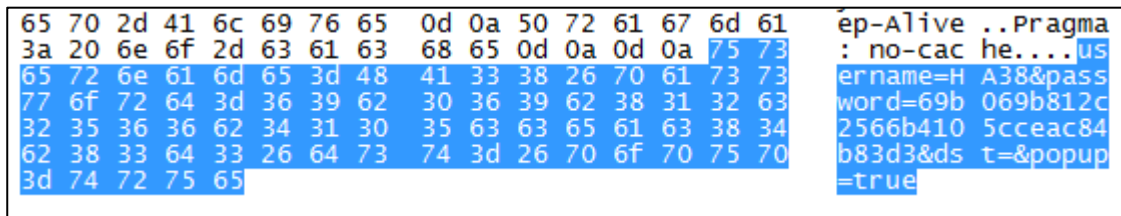


Figure 29 – Detail of client response to challenge

A brute force attack could reveal the password in an acceptable time frame, as long as the password policies are simple enough. Our plan was to try:

- ▶ Numeric passwords from 1 to 12 digits
- ▶ Lower case characters from 1 to 12 digit
- ▶ Upper and lower characters and digits, from 1 to 12 digits

We stopped at the first scenario since the password was a 4 number digit. Since the number of password can be calculated by

$$n = \sum_{i=\min}^{\max} C^i$$

where C denotes the character space for each position in the password, min the minimum password length, max the maximum password length and n the number of possible passwords, in this case we have:

$$n = \sum_{i=4}^4 10^i = 10^4 = 10000 \text{ possible passwords}$$

That allows an extremely quick brute force attack to get access to the Internet.

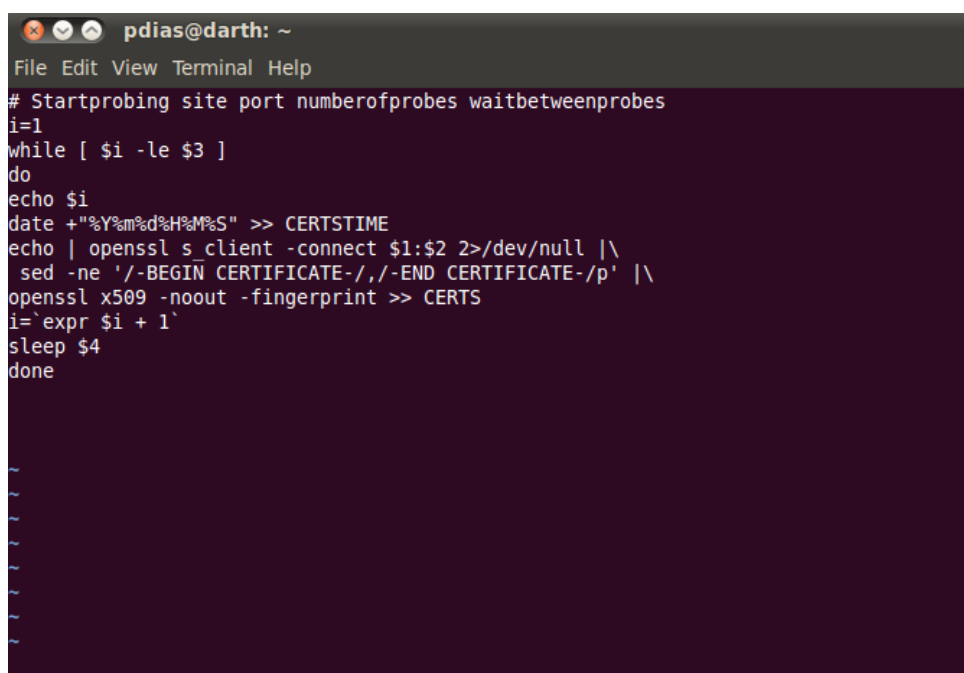
An attack spoofing DNS query request, replaced with our own IP would then create the initial conditions to try to perform a MitM attack.

4.2.2 Attack on home router

We access several home routers from PT. We used the same default credentials supplied in our own installation, with password “3!Play”. All routers installed by PT teams within reach have the same password.

4.3 Probing and Caching

In this experiment we wanted to evaluate the existence (or not) of caching when probing a site for a certificate, using the same origin IP. We created a shell script running on Ubuntu Linux, inside a Virtual Machine (VM) running VMWare workstation version 7.1.3 build-324285 on a Microsoft Windows 7 Professional with Service Pack (SP) 1 host.



```
pdias@darth: ~
File Edit View Terminal Help
# Startprobing site port numberofprobes waitbetweenprobes
i=1
while [ $i -le $3 ]
do
echo $i
date +%Y%m%d%H%M%S" >> CERTSTIME
echo | openssl s_client -connect $1:$2 2>/dev/null |\
sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' |\
openssl x509 -noout -fingerprint >> CERTS
i=`expr $i + 1`
sleep $4
done
```

Figure 30 – Probing for certificates

We performed 1000 continuous probes on Citibank, with no delay at all. Results are summarized in Figure 31.

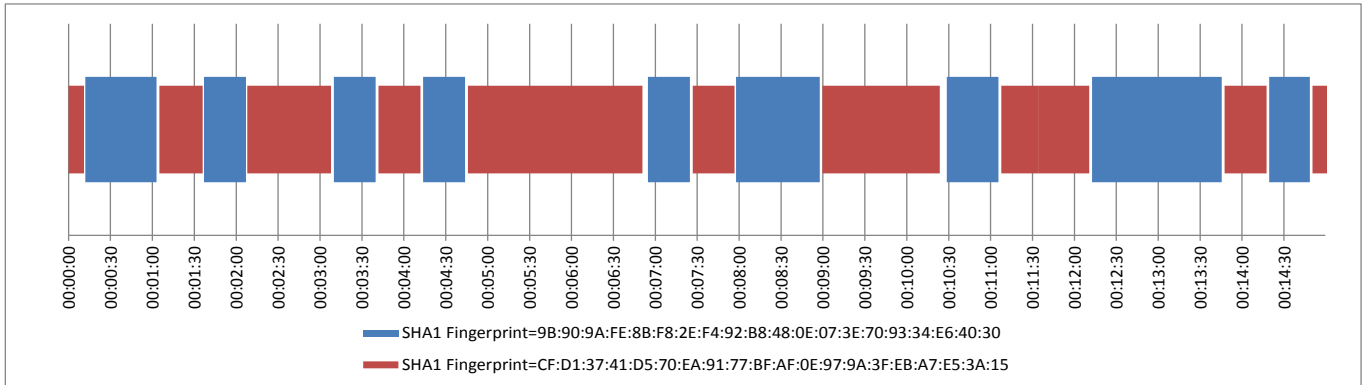


Figure 31 – Probing results with 1 second delay between probes, on Citibank:443

We can notice a cycle of around 30 seconds in which the returned certificate is always the same. We hypothesized that the server uses a 30 second validity cache, during which it always replies with the same certificate. However, we were expecting to see more than two certificates, since, in our initial probe, as mentioned in Figure 13, more than one dozen certificates were observed. Since more than 200 days have elapsed since the results presented in that figure were observed, we decided to probe again the Notaries for the data. Results are shown in Figure 32.

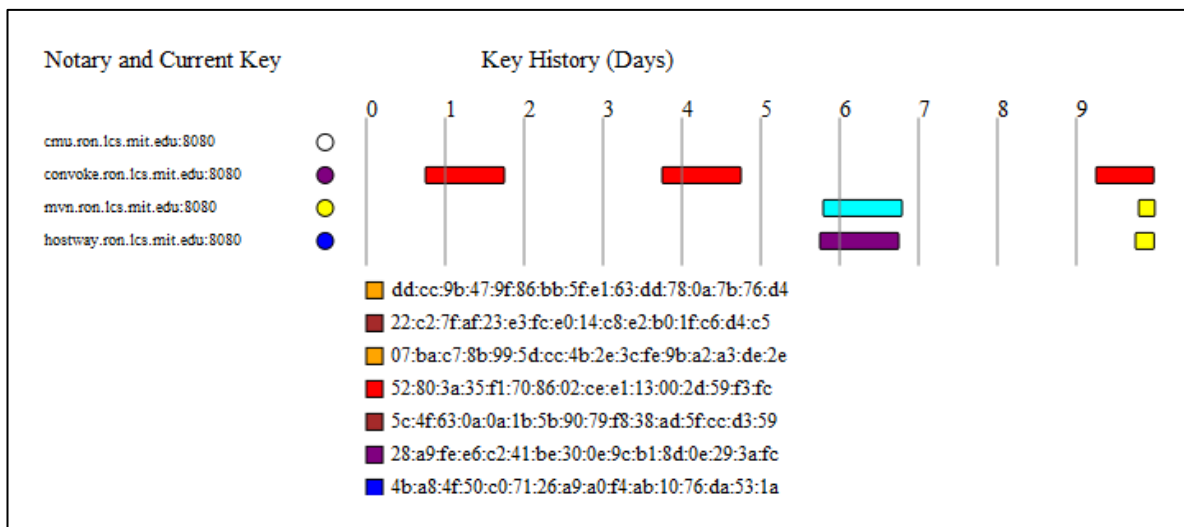


Figure 32 – Another 10-day Key History for citibank.com:443

We can observe an increased inconsistency in certificate observation, when compared to our first query for Notary results. Some Notaries do not observe any certificate, while others observe one certificate sporadically. We did query the Notaries again for a 200-day key history. Results are shown in Figure 33.

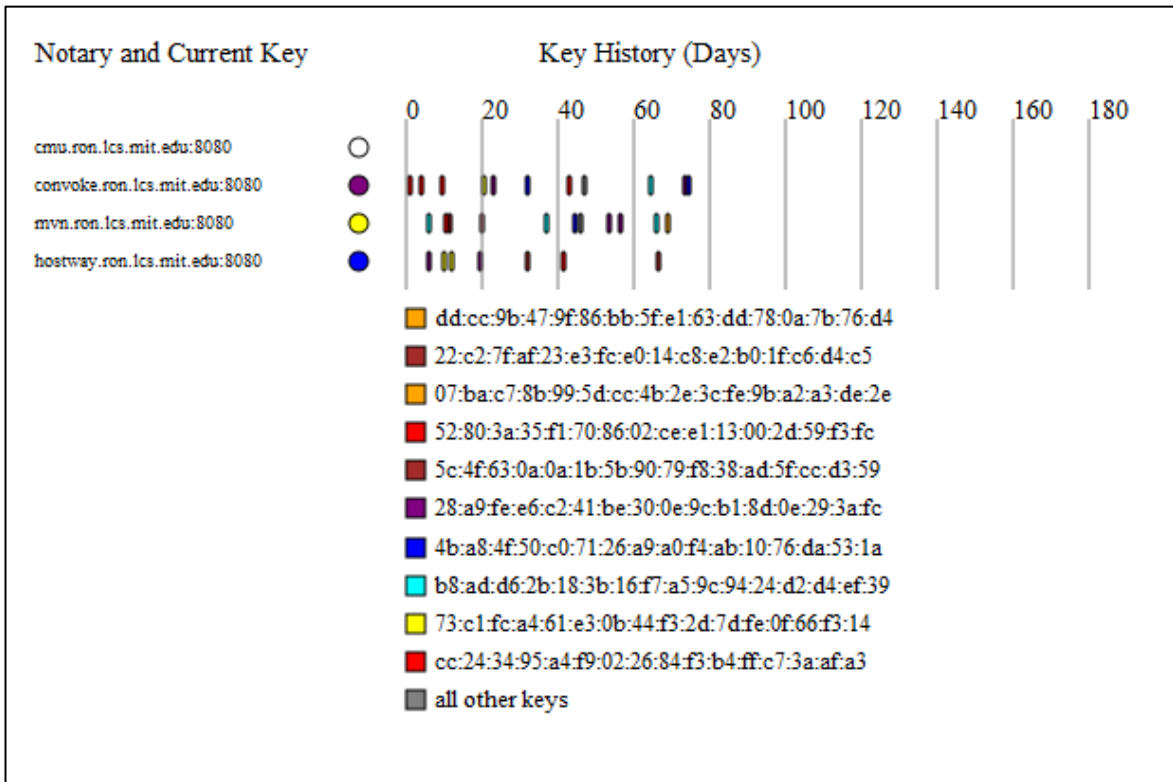


Figure 33 – Another 200-day Key History for citibank.com:443

Facing the results we hypothesized that Notaries results were cleared about 70 days ago, but that proved not to be the case, since history existed for other sites (like bes-sec.bes.pt:443), as shown in Figure 34.

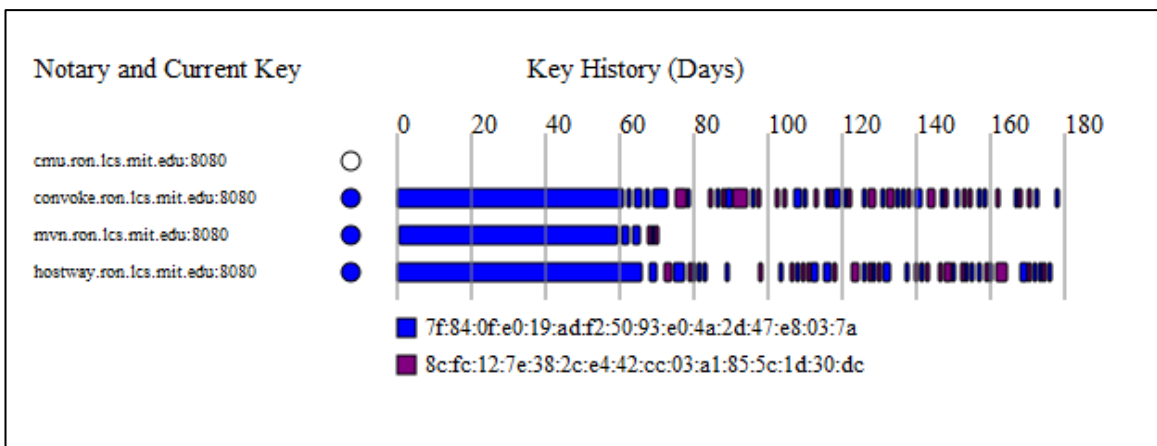
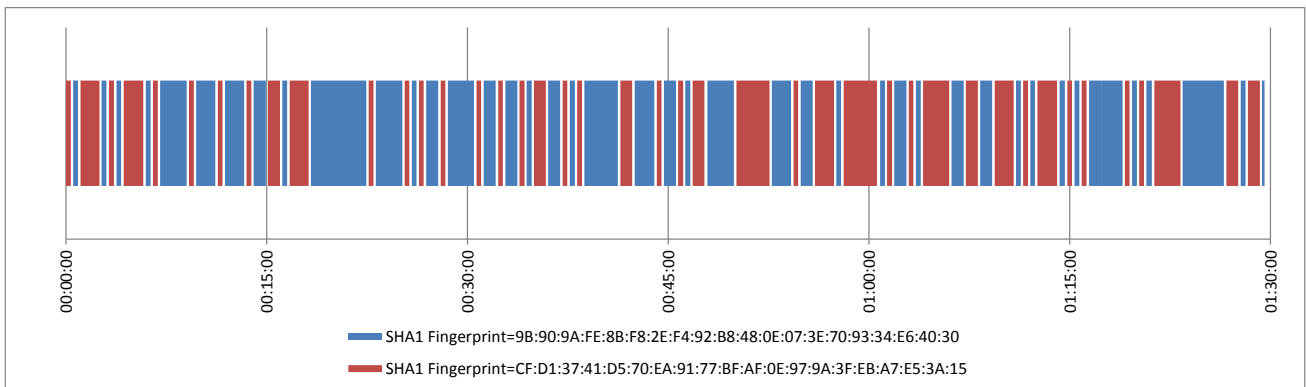


Figure 34 – Another 200-day Key History for bes-sec.bes.pt:443

Therefore, at this moment, we do not have a good explanation for the recent behavior of the Citibank site.



In any case, from the probing tests, and facing the collected results, we observed a consistent behavior with the following steps:

- ▶ the site randomly selects a certificate
- ▶ the same certificate is always returned during a specific period, around 30 seconds, a default value for most common web servers

From the observed results we can conclude that continuous probing does not improve certificate discovery (possibly being masquerade by caching mechanisms), suggesting a delay between probes larger than usual caching values, in a magnitude possibly larger than 5 minutes.

Chapter 5

Conclusions and Future work

5.1 Conclusions

As a first impact, we noticed that vulnerabilities due to improper setup on network components is still a very devastating flaw when attempting to attack a site, even when protected by SSL. From weak policies to misconfigured components, we encountered several situations where these flaws lead to vulnerabilities very easily exploited.

We concluded that the benefit of using multiple certificates is not a regular practice, even in sites with redundant servers. We could not find a handful of sites using more than two certificates. Even when two certificates were found, it seems that they are not consistently seen, which could indicate that they were either replacements or test certificates (only observed in a specific, small (couple of days) time frame).

From the observed results we can conclude that continuous probing does not improve certificate discovery (possibly being masquerade by caching mechanisms), suggesting a delay between probes larger than usual caching values, in a magnitude of at least 5 minutes. In any case, an increasing in the probing frequency not only allows the Notaries to become aware of multiple certificate sites, but also increases the accuracy for observed certificates.

We have proposed a comprehensive solution to address multiple certificate sites. This solution encompasses changes in several components of *Perspectives*, including the management of policies and the creation of quorums in the Notaries responses.

5.2 Future Work

Additional study on caching has to be done to validate the results of probing and the effect of caching. It seems that sites have not yet seen the benefit of using multiple certificates.

Further analysis should be done on the cross-validation protocol to study the impact of returning all history from multiple certificates, or just the certificate currently seen by the client.

The definition of function $f(N)$ in order to define the minimum number of probes require further study.

Some improvements to *Perspectives* may include:

- ▶ Upon receiving responses from the server with invalid signatures, the client can collect data instead of just discarding the message, since it may indicate an on-going attack, possibly reporting that to the Notary Authority;
- ▶ Handle airports and hotels. Since laptops cannot access the Internet at these locations, we could have laptops submit certificates they've learned about in those situations. If we obtain a sufficient number of observations, then we would add the certificate to a database, which clients can periodically download for off-line verification. Of course, we would need to study the case where an attacker submits observations for its own malicious access points;
- ▶ Determining how on-demand cloud computing resources might be used to power Notaries. One of the "big picture" ideas behind *Perspectives* was that computing resources (CPU, bandwidth, etc.) for automated probing keeps getting cheaper, while human time to manage PKIs does not. Using a cloud platform could let us put an exact \$ amount on the cost of monitoring X sites;

Annex I

The Coupon Collectors Problem applied to digital certificates

In probability theory, the Coupon Collectors Problem describes contests where a customer is required to collect all different coupons to win a prize. Imagine that a cereal brand includes 1 picture (out of 50 different possible pictures) inside each cereal box. The customer is required to collect all 50 different pictures to claim a free trip to Maldives. In our case we assume the following rules for our contest:

- ▶ The number of copies of each picture is similar (we only win if we are one of the first x customers to present all the pictures, where x is the number of trips available);
- ▶ Each time one box is taken from the shelf, a new one replaces it. This means (like in our site with multiple certificates) the probability of getting a particular picture is the same each time we buy a cereal box.

As mentioned in [8], if we have N certificates, the expected number of probes to get all certificates is $P = N \cdot H_N$ where H_N is the harmonic number $\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{N}$.

If a site has 50 certificates, how many probes do we need to perform to get all the certificates? The expected number of probes is $50H_{50}$, which equals 225. A sample of results for probing until 80 certificates is shown in Figure 36.

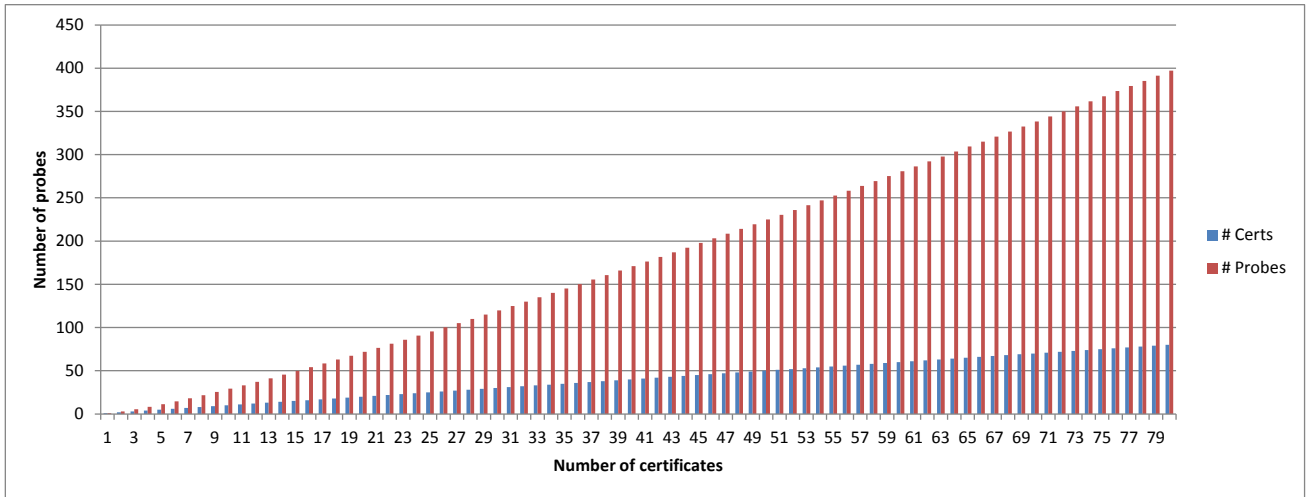


Figure 36 – Expected number of probes to retrieve all digital certificates

This probing method can generate a considerable amount of probes for sites with a large number of certificates. For example, 36 certificates would require 150 probes.

For that reason, we suggest the application of probabilities, assuming a population of the same size last observed. Any variation would be recalculated dynamically, during the probing process, considering the observed values. Let's recall that, for an estimated population of N certificates, the probability of observing O certificates after P probes is given by $(O/N)^P$. In the rest of this section we will present some examples, using the following convention:

- δ = Treshold
- N = Estimated number of certificates
- β = Current number of observed certificates
- P = Current number of probes performed so far
- Θ = Current Probability

The probing will be done according to the algorithm shown in Figure 37.

```

P = 0
Do Probe()
  P++
  if ( NewCertificate ) β ++
  Update (Θ)
While { Θ ≤ δ }
N = β

```

Figure 37 – Probing algorithm

Example 1:

Results are returned within the expected probability, so only after an increase in P will the probing stop.

$\delta = 0,003$
 $N = 5$
 $\beta = 4$
 $P = 27$, since $\Theta = 0,002417851639229 < \delta$

Example 2:

Now consider the following case:

$\delta = 0,003$
 $N = 15$
 $\beta = 1$
 $P = 3$
 $\Theta = 0,000296296296296 < \delta$

This case could be due to an existing cache. To minimize this effect, one could impose a minimum number of probes, as a function of N. If we consider m the minimum number of probes and f(N) the function determining the minimum number of probes, we would update the algorithm as shown in Figure 38.

```
P = 0
m = f(N)
do Probe()
    P++
    if ( NewCertificate )  $\beta$  ++
    Update( $\Theta$ )
While {  $\Theta \leq \delta$  &  $P < m$  }
N =  $\beta$ 
```

Figure 38 – Probing algorithm with minimum number of probes

Example 3:

Update on example 2 with minimum probes required:

$$\delta = 0,003$$

$$N = 15$$

$$F(N) = \frac{2}{3}N$$

$$m = 10$$

$$\beta = 10$$

$$P = 15$$

$$\Theta = 0,002283658260521 < \delta$$

Conventions

t = Time

I = Number of Initial Probes to determine is site is multi-cert

P = Number of Probes

PK_u = Public Key

K = Key

S = Service

n = Number of Notaries

q = Quorum of Notaries

q_s = Quorum of Notaries in a single certificate site

q_m = Quorum of Notaries in a multiple certificate site

z = Zone threshold

Z = Total number of defined zones

δ = Treshold

N = Estimated number of certificates

β = Current number of observed certificates

Θ = Current Probability

References

- [1] Dan Wendlandt, David G. Andersen, Adrian Perrig. *Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing*
- [2] Ed Schwartz. Contractual anonymity. [Technical report] / Carnegie Mellon University. Information Networking Institute; Technical report (Information Networking Institute), 2009
- [3] Bruce Schneier. *Secrets & Lies: Digital Security in a Networked World*, Wiley Publishing, Inc. ISBN 0-471-45380-3, 2000
- [4] W32.Arpiframe // http://www.symantec.com/business/security_response/writeup.jsp?docid=2007-061222-0609-99
- [5] Gunter Ollmann. *The pharming guide*. <http://www.ngssoftware.com/papers/ThePharmingGuide.pdf>
- [6] Zulfikar Ramzan, <http://www.symantec.com/connect/blogs/drive-pharming-follow>
- [7] Chris Karlof, J.D. Tygar, David Wagne, Umesh Shankar. *Dynamic Pharming Attacks and Locked Same-origin Policies for Web Browsers*
- [8] John Moriarty, Peter Neal. *The Generalized Coupon Collector Problem*, 2008
- [9] Tamar Gadrich, Rachel Ravid. *The Coupon Collector Problem in Statistical Quality Control*
- [10] Anna Pósfai. *Approximation Theorems Related to the Coupon Collector Problem*, 2010
- [11] S. Bellovin, *Security Problems in the TCP/IP Protocol Suite*, Computer Communication Review, Vol 19, 2, pp. 32-48, April 1989
- [12] Morris, R.T., *A Weakness in the 4.2BSD UNIX TCP/IP Software*. Computing Science Technical Report No. 117, AT&T Bell Laboratories, Murray Hill, New Jersey, 1985
- [13] S. Bellovin et al, *MD5 Considered Harmful Today - Creating a rogue CA certificate*, 25th Chaos Communication Congress, Berlin, December 2008
- [14] F.T. Grampp, R.H. Morris, *Unix Operating System Security*, AT&T Bell Laboratories Technical Journal, vol. 63, no 8, part 2, October 1984