Author: rupp,scd
Project:Device Fingerprinting/ Crypto
Date: 07/11/13

USING DEVICE FINGERPRINTING TO IMPROVE CRYPTOGRAPHIC PROTOCOLS

Index

Illustration Index

1. Introduction

This document describes how to use device fingerprinting to improve cryptographic protocols

2. Environmental data

2.1. Definition of an User's Environment

We define an User's Environment as a P.C or any computer machine equipped with a given operating system. This may or may not include the physical location of this machine.

Illustration 1: Example of an User's Environment

2.2. Using IP Addresses

Using IP addresses to check that a user is correctly identified is a well-known process. It is possible to perform geolocation from an IP address against a router database. Some of these database are more accurate than others. Some may identify the country of origin of an IP , some others the state or the town. For example GeoLite[1] is a *free* geolocation database provided by MaxMind which is known to be accurate enough to provide good checks.

- **Case where the Physical location is included in the User's Environment definition**

If the physical location of the machine is to be included in the definition of the user's environment then checking the IP geolocation is a good tool to improve the security of existing algorithms, Indeed it is enough to check that the connection originates *from the same country* it is supposed to come from.

It would be possible to check at the state or town level but this will gives room for more error, indeed the geolocation databases are rarely completely accurate at this level of precision.

1 http://www.maxmind.com/app/geolite

Self-tests[2] performed by MaxMind (which is one of the leader in IP-based geolocation) shows that MaxMind IP geolocation is 81% accurate for the U.S.A and in general around 50-70% for the other countries on a city level while they claim the product is 99.8% accurate worldwide on a country level[3]

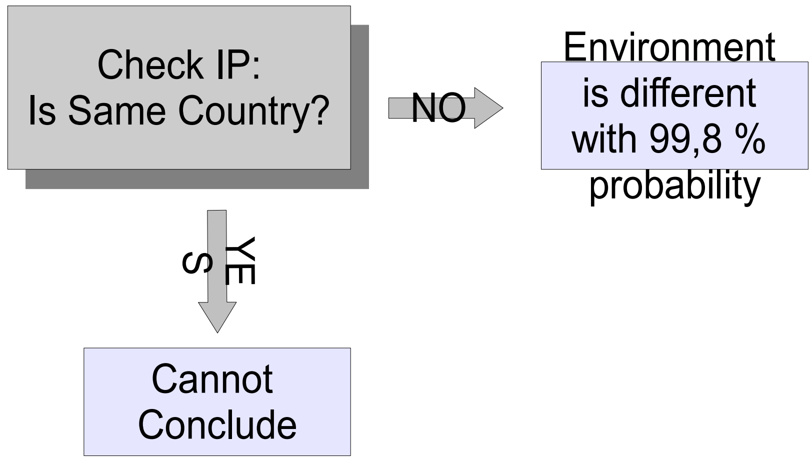


Illustration 2: IP Check (MaxMind Database)

In conclusion it is possible to reject with 99.8% accuracy (in the case of MaxMind products) a user connecting from a different country but is is hardly possible to ensure that Environment location is the same since Geolocation is only accurate at a country level. In other terms, IP-based tests may be used for rejection (if the country differs the environment is different) but not for acceptation (if it is the same country it doesn't mean it is the same environment) .

- **Case where the Physical location is not included in the User's Environment definition**

In that case (for example a laptop which may connect from any location), IP checks are even less interesting: if a country differs it does not necessarily mean that the user's environment is not the same, it may mean that the user is traveling with its laptop. It is however possible to take into account known average travel speed from airplanes (usually < mach 1) and compare it with the ratio distance between two consecutive log-ins locations /time between two consecutive log-ins.

2.3. Using Machine Environement for fingerprinting

a) Browser-based Fingerprints

These are 'non-intrusive' fingerprints in the sense that only commonly information retrieved by

2 http://www.maxmind.com/app/city_accuracy
3 http://www.maxmind.com/app/country#features

browser plug-ins are used.

Data	Level of characterization of the environment	Frequency of a given environment value among population of PCs
[redacted]		
Screen Parameters	Medium	~0,0001%
List of fonts installed	High	~0,000005%
Hardware information (full)	Very High	~ 0%

```
System Infos

avHardwareDisable: false
hasAccessibility: true
hasAudio: true
hasAudioEncoder: true
hasEmbeddedVideo: true
hasMP3: true
hasPrinting: true
hasScreenBroadcast: false
hasScreenPlayback: false
hasStreamingAudio: true
hasVideoEncoder: true
isDebugger: false
language: en
localFileReadDisable: false
manufacturer: Google Windows
os: Windows Server 2003 R2
pixelAspectRatio: 1
playerType: PlugIn
screenColor: color
screenDPI: 72
screenResolutionX: 1024
screenResolutionY: 600
serverString: A=t&SA=t&SV=t&EV=t&MP3=t&AE=t&VE=t&ACC=t&PR=t&SP=f&SB=f&D
version: WIN 11,3,300,257
```

Illustration 3: Flash Hardware Detection

It also required to compute how the known frequency (estimation) of a given parameters may influence an FAR or an FRR.

For example since the frequency of a given Screen Parameters s_0 is f_s =0,0001% this mean that probability This does *not* mean that the error rate that the environment is the referenced user environment is 0,0001%, this means, if the total population of different environments is P, that there are $(P-1)\times f_s+1$ different environment sharing the same signature and that therefore the probability that s_0 represent a given environment is $\frac{1}{(P-1)f_s+1}$ or equivalently $\frac{1}{N_s}$ where N_s is the (estimated) total number of environment in the given population sharing the same signature Then we get a (theoretical) FAR of :

$$FAR_s=1-\frac{1}{N_s}=\frac{N_s-1}{N_s}=\frac{1}{1+\frac{1}{(P-1)f_s}}$$

```
-------O.S INFO---------
OSVersion: Microsoft Windows NT 5.1.2600 Service Pack 3
CLR Version: 4.0.60831.0
Number of processors: 2
-------CLIENT INFOS---------
-------AUDIO CAPTURE DEVICES---------
---------------
name: Realtek HD Audio Input
-------VIDEO CAPTURE DEVICES---------
---------------
name: USB Video Device
-------GPU---------
---------------
Device ID=40977
Vendor ID=32902
Driver=6.14.10.5182
---- Fonts -----
```

Illustration 4: Silverlight Hardware Detection Extract

For example if we consider a population of 100 millions different environments then we get a *FAR* of *~99,2%*!-which is absolutely useless .But if we consider a population of only 100.000 different environments then we have an estimated FAR of ~11.8%.

Population	FAR_s
10^8	99.20%
10^7	93.00%
10^6	57.20%
10^5	11.80%

For the List of Fonts installed test with frequency f_f , we have in a population of 100 millions environments a FAR of 97,8% which is still useless it only makes sense for a population of 10^5.

Population	FAR_f
10^8	97.80%
10^7	81.70%
10^6	30.80%
10^5	4.27%

First thing we remark is that we may combine the different Environment tests to lower the

FAR.

For example if we combine the Screen Parameters test with frequency f_s and the List of Fonts installed test with frequency f_f, we get a FAR which is given by:

$$FAR_{fs}=(1-\frac{1}{N_s})\times(1-\frac{1}{N_f}) .$$

We get the following table:

Population	FAR_{fs}
10^8	97.02%
10^7	75.98%
10^6	17.62%
10^5	0.50%

We note that f_f may turn to be in reality far smaller that its actual estimation.

About the full hardware information provided by Silverlight there is no data available since this was only implemented by SCD, it may be considered from heuristic reasons that no two environment will possibly share the same hardware signature on this test but this is yet to be proven by a large scale experimentation.

The combinations of test may lower dramatically the full FAR for a combination of test but this will be always dependent of the full population of environments to be considered.

b) Hardware-based fingerprinting

This will require the installation and execution of a software

The following data can be retrieved:

-harddrive serial

-Disk Volume serial

-Devices ID

-MAC addresses of ethernet cards

The mac address of an ethernet card is in itself a unique identifier and as such a unique fingerprint.. Considering that almost all computer owns at least one ethernet card with a MAC address this allows us to consider that we will have always a unique hardware fingerprint.

The MAC address can be easly spoofed and most hardware-based Ids can be easily faked.

Conversely some device Ids of USB devices like webcams or removable harddrive could not be present and false the computation of the figerprint, also the disk volume serial could change when the computer is reformatted.

c) Profile-based fingerprinting

This approach consists in using the 'behaviour' of the device using statistical samplings and statistical models of, for example, data transmission,radiometrics, etc... This is much more tolerant to configuration changes and much more difficult to fake than a traditional hardware-based analysis.

2.4. Tolerance of the fingerprint to configurations updates

The device fingerprinting must be tolerant to small changes of the hardware configfuration.

This can be done by calculating a distance between two fingerprints.

For example if S_1 and S_2 are two different fingerprints, we define a distance d in the space of device fingerprints and we claim that if $d(S_1,S_2)<e$, where e is a very small parameter (for example e=1/1000) then S_1 and S_2 are the same device fingerprint signature.

Considering the fingerprints as strings, there exist several algorithms that can co,mpute the similarity beween them and be used as a distance:

- *Hamming distance*
- *Levenshtein distance*
- *Jaro-Winkler*

The Jaro-Winkler distance is defined as:

The Jaro distance d_j of two given strings s_1 and s_2 is

$$d_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \text{otherwise} \end{cases}$$

where:

- m is the number of *matching characters*;
- t is half the number of *transpositions*.

2.5. Bayesian Classification

A Bayesian Classificator system could be used to determine the acceptable variation of a fingerprint in the space of device fingerprints.

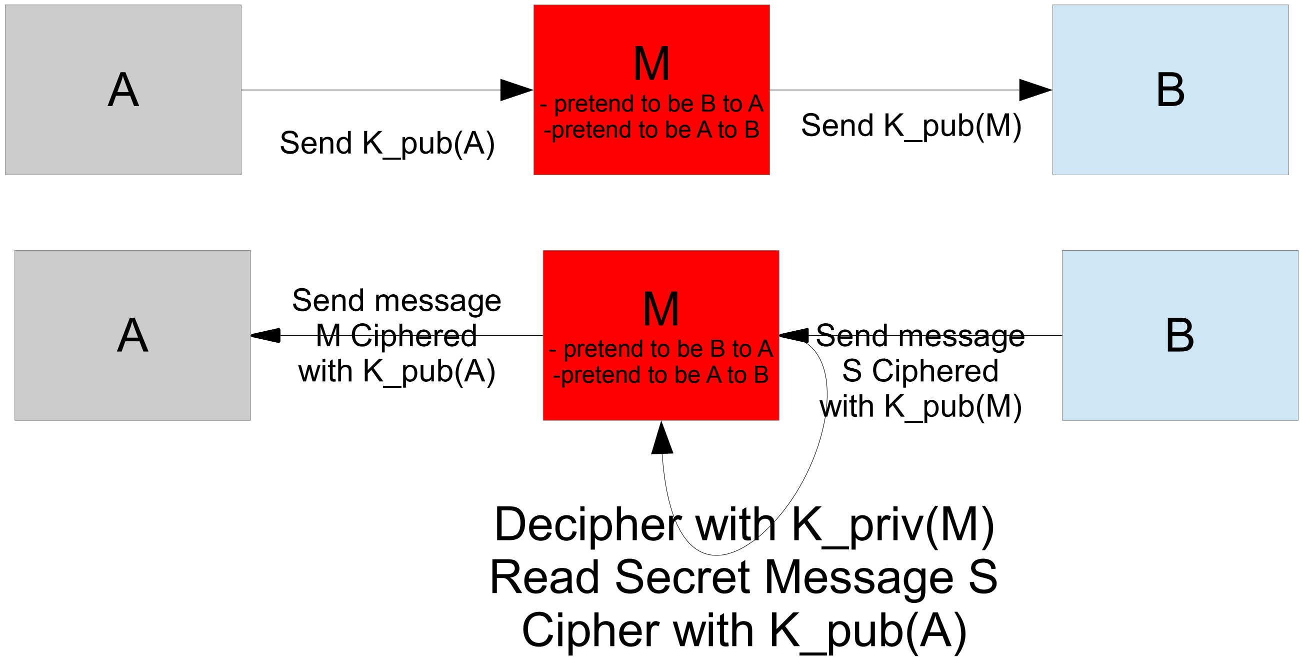
3. Improving Cryptographic protocols

A device fingerprinting system can improve existing cryptographic protocols by adding an authentication factor level defined as the environment authentication.

- secure channel
- ciphering

Explain why it will resists better to the following attacks:

a) Man—in-the-middle attack



lustration 5: Sample (Very) Basic MTM Attack

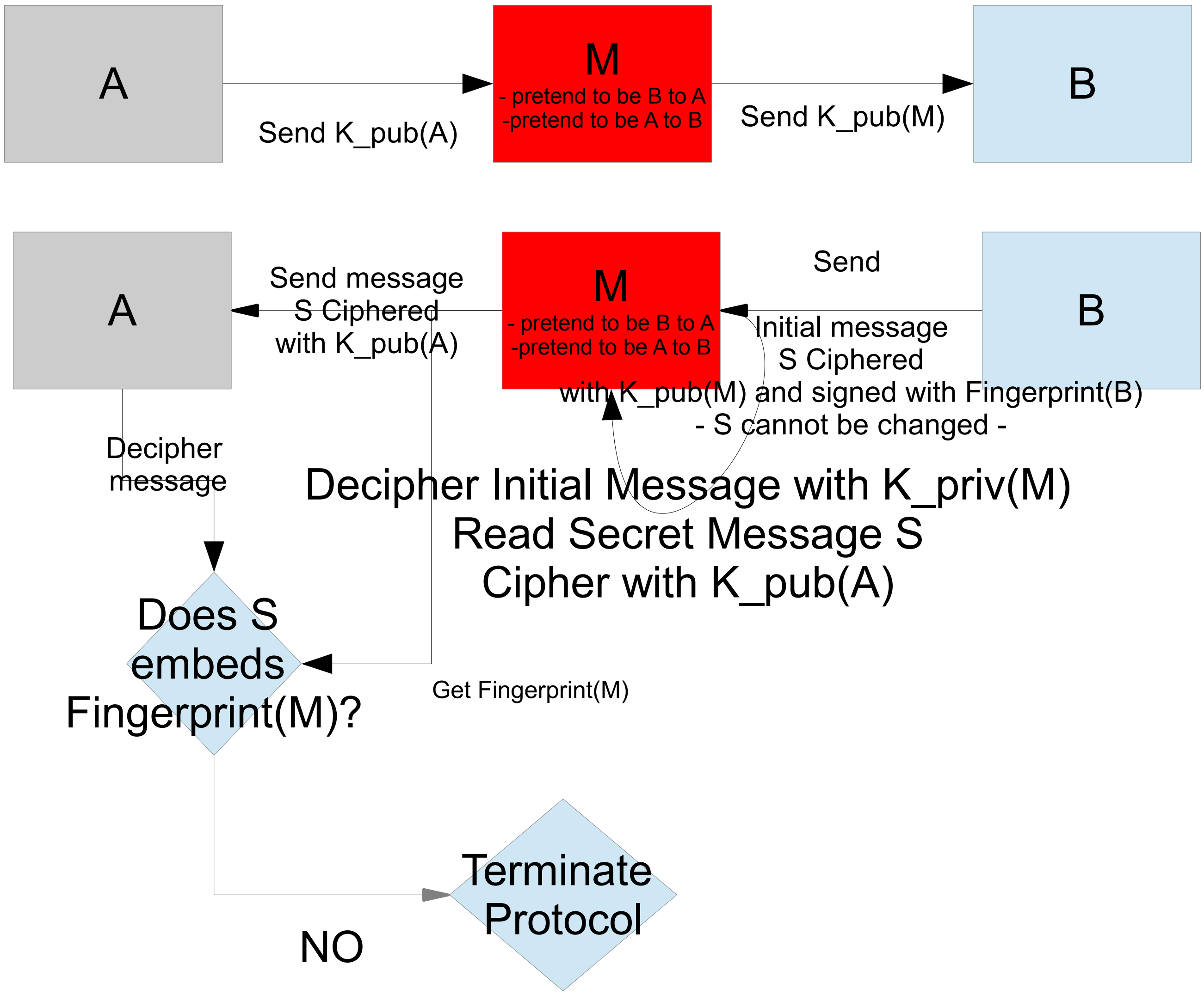


Illustration 6: Improved Basic One Way Authentication with device fingerprinting

In this method, B sends an initial message contining not sensitive information but embedding its device fingerpriting signature, something that M will not be able to counterfeit.

If M decipher the initial message, it is not of real interest, but M is unable to modify the message and embed its own signature so if M sends the initial messge that A is waiting for, A detects that the fingerprint signature embedded inside is different from the device fingerprinting it receives from M, the sender. Therefore A consider a MITM attack and stop the protocol.

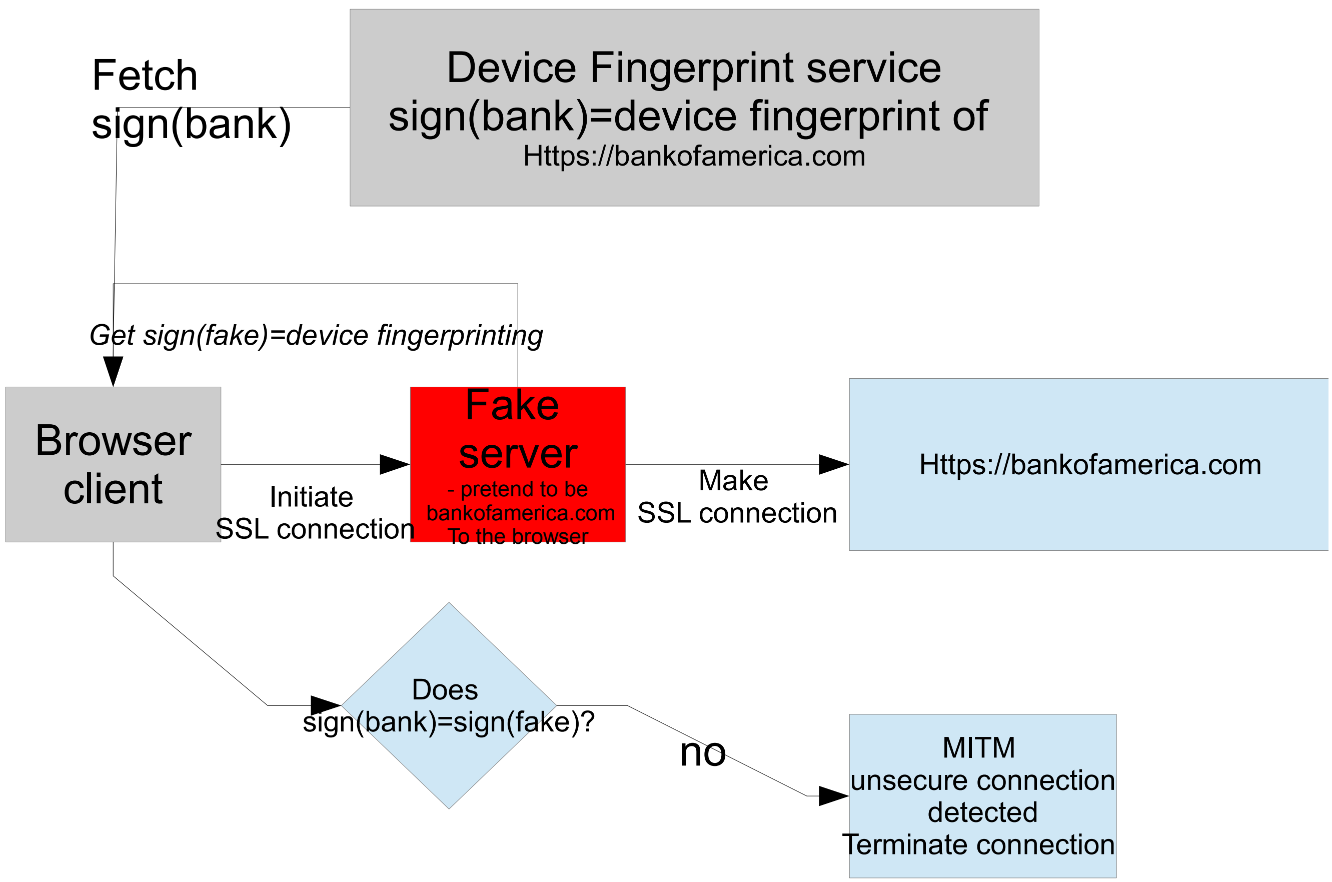


Illustration 7: Device Fingerprinting Service as prevention Against SSL MITM

4. Concrete use case: SSL man in the middle attack prevented with device fingerprinting

We focus on a concrete use case, which is the MITM attack on an SSL website.

We detail how an attacker could perform this attack and how a device fingerprint service could prevent this attack to take place.

4.1. Principle of SSL MITM

We suppose that the browser is inside a network (for example a corporation or an internet cafe or an internet WIFI access point) where has been installed a SSL proxy.

We suppose that somehow, the browser of the user computer has been installed with a fake certificate authority that will allow the browser to trust the SSL proxy server. This can be done by the IT departement of the corporation or the internet cafe employees to install this in the ,machine they own or this can be done by luring a wifi user to install the fake certificate by mean of a web page when connecting to the access point.

Once the SSL proxy certificate is installed, any time the user connects to an https server https://securesite.com the SSL proxy creates on-the-fly a fake self-signed certificate with the name of the website securesite.com etc... (that the user browser will trust since the fake CA has been installed). On the other hand the SSL proxy makes an SSL secure connection itself to securesite.com and forward the request to the user.

This attack is perfect in the sense that it would be impossible for a user to detect that it is not connected to securesite.com because the browser will show the SSL symbol in the address bar etc...

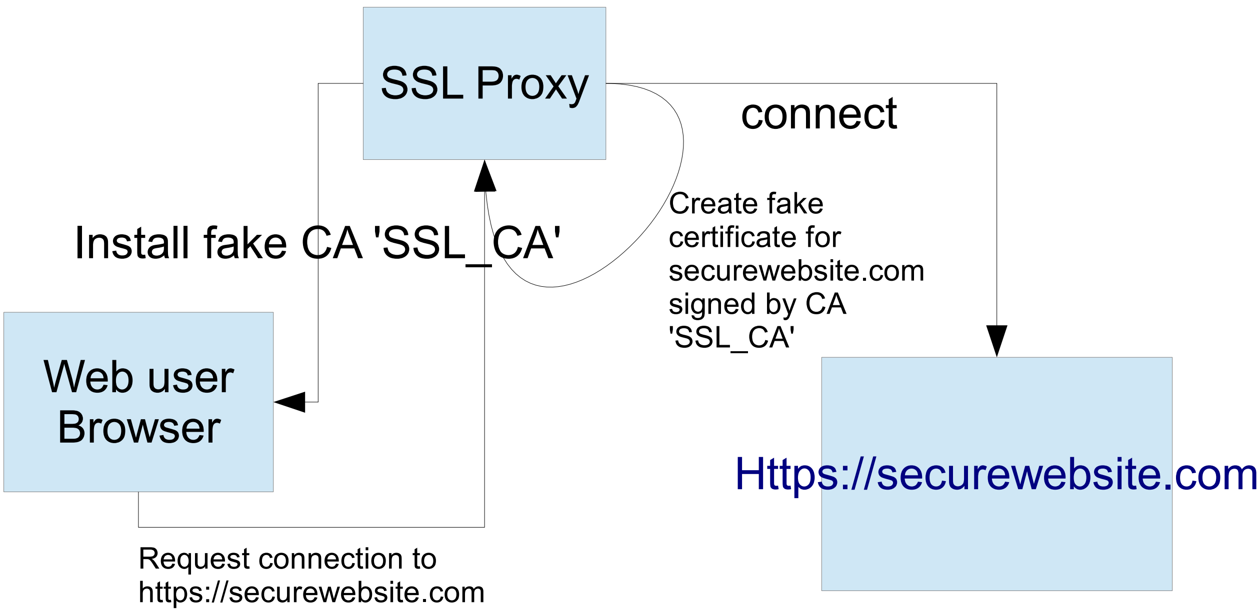


Illustration 8: Principle of SSL MITM

4.2. use of device fingerprinting

Here we present computation of a sample test device fingerprinting ID and we suppose that the browser is able to retrieve these informations directly from the server by mean of a service/applet etc...

This is fictive scenario in the sense that some of these informations may not be directly available to the browser, nevertheless when using statistical samplings trying to get a fingerprint of the server by using network probes, this is possible.

We compute the device fingerprint of the server we are dealing with (which is the SSL Proxy server) and we retrieve a hash fingerprint:

“**8A-97-44-DB-6D-BF-BB-E6-7E-34-95-FC-13-B6-B4-6C**”

but when accessing the service website for getting the device ID of the server that host securewebsite.com, we find the following values:

"EC-17-4D-92-73-42-46-54-B5-66-57-5C-14-91-32-A5"

Since these fingerprints are different, we reject the connection.

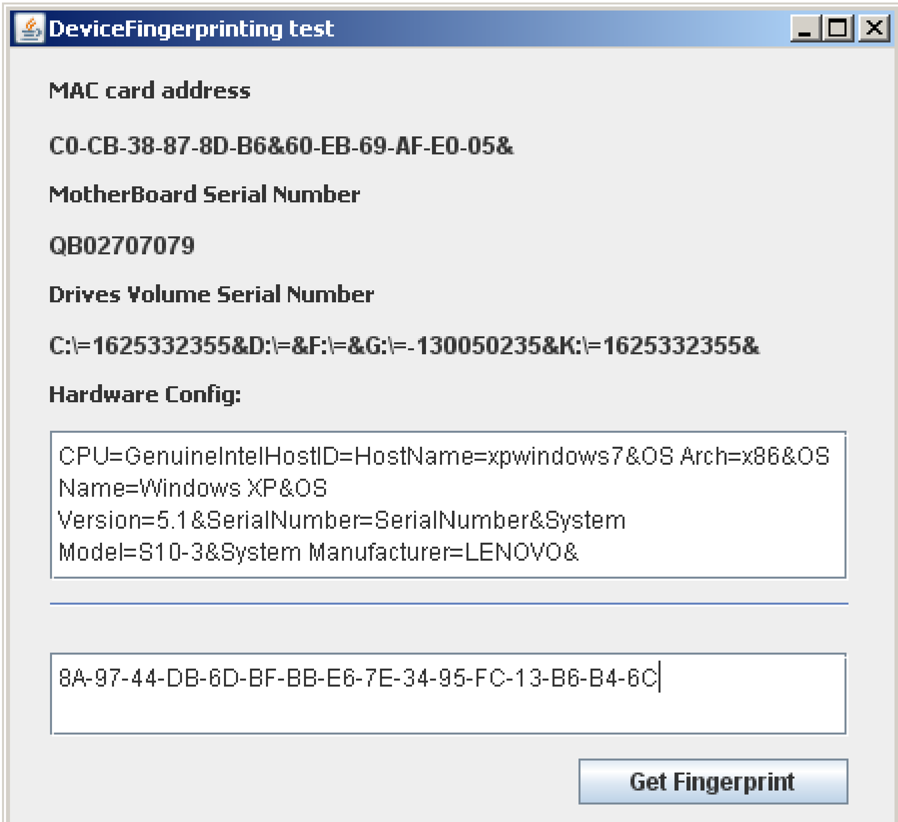


Illustration 9: computation of Device Fingerprinting for SSL proxy pretending to be securewebsite.com

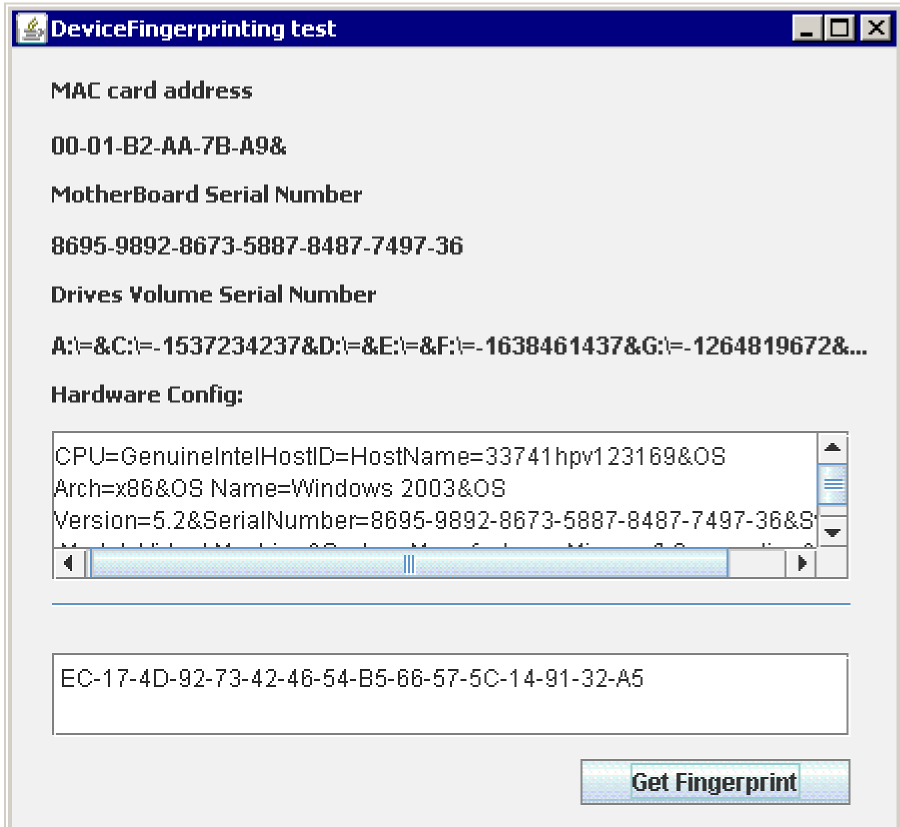


Illustration 10: Device Fingerprinting of the real server for securewebsite.com