Compromised Web Servers and Web Shells - Threat Awareness and Guidance | CISA

Published: 2017-08-09 · Archived: 2026-04-05 16:17:39 UTC

Systems Affected

Compromised web servers with malicious web shells installed

Overview

This alert describes the frequent use of web shells as an exploitation vector. Web shells can be used to obtain unauthorized access and can lead to wider network compromise. This alert outlines the threat and provides prevention, detection, and mitigation strategies.

Consistent use of web shells by Advanced Persistent Threat (APT) and criminal groups has led to significant cyber incidents.

This product was developed in collaboration with US-CERT partners in the United Kingdom, Australia, Canada, and New Zealand based on activity seen targeting organizations across these countries. The detection and mitigation measures outlined in this document represent the shared judgement of all participating agencies.

Web Shell Description

A web shell is a script that can be uploaded to a web server to enable remote administration of the machine. Infected web servers can be either Internet-facing or internal to the network, where the web shell is used to pivot further to internal hosts.

A web shell can be written in any language that the target web server supports. The most commonly observed web shells are written in languages that are widely supported, such as PHP and ASP. Perl, Ruby, Python, and Unix shell scripts are also used.

Using network reconnaissance tools, an adversary can identify vulnerabilities that can be exploited and result in the installation of a web shell. For example, these vulnerabilities can exist in content management systems (CMS) or web server software.

Once successfully uploaded, an adversary can use the web shell to leverage other exploitation techniques to escalate privileges and to issue commands remotely. These commands are directly linked to the privilege and functionality available to the web server and may include the ability to add, delete, and execute files as well as the ability to run shell commands, further executables, or scripts.

How and why are they used by malicious adversaries?

Web shells are frequently used in compromises due to the combination of remote access and functionality. Even simple web shells can have a considerable impact and often maintain minimal presence.

Web shells are utilized for the following purposes:

1. To harvest and exfiltrate sensitive data and credentials;
2. To upload additional malware for the potential of creating, for example, a watering hole for infection and scanning of further victims;
3. To use as a relay point to issue commands to hosts inside the network without direct Internet access;
4. To use as command-and-control infrastructure, potentially in the form of a bot in a botnet or in support of compromises to additional external networks. This could occur if the adversary intends to maintain long-term persistence.

While a web shell itself would not normally be used for denial of service (DoS) attacks, it can act as a platform for uploading further tools, including DoS capability.

Examples

Web shells such as China Chopper, WSO, C99 and B374K are frequently chosen by adversaries; however these are just a small number of known used web shells. (Further information linking to IOCs and SNORT rules can be found in the Additional Resources section).

- ***China Chopper*** – A small web shell packed with features. Has several command and control features including a password brute force capability.
- ***WSO*** – Stands for “web shell by orb” and has the ability to masquerade as an error page containing a hidden login form.
- ***C99*** – A version of the WSO shell with additional functionality. Can display the server’s security measures and contains a self-delete function.
- ***B374K*** – PHP based web shell with common functionality such as viewing processes and executing commands.

Delivery Tactics

Web shells can be delivered through a number of web application exploits or configuration weaknesses including:

- Cross-Site Scripting;
- SQL Injection;
- Vulnerabilities in applications/services (e.g., WordPress or other CMS applications);
- File processing vulnerabilities (e.g., upload filtering or assigned permissions);
- Remote File Include (RFI) and Local File Include (LFI) vulnerabilities;
- Exposed Admin Interfaces (possible areas to find vulnerabilities mentioned above).

The above tactics can be and are combined regularly. For example, an exposed admin interface also requires a file upload option, or another exploit method mentioned above, to deliver successfully.

Impact

A successfully uploaded shell script may allow a remote attacker to bypass security restrictions and gain unauthorized system access.

Solution

Prevention and Mitigation

Installation of a web shell is commonly accomplished through web application vulnerabilities or configuration weaknesses. Therefore, identification and closure of these vulnerabilities is crucial to avoiding potential compromise. The following suggestions specify good security and web shell specific practices:

- Employ regular updates to applications and the host operating system to ensure protection against known vulnerabilities.
- Implement a least-privileges policy on the web server to:
    - Reduce adversaries' ability to escalate privileges or pivot laterally to other hosts.
    - Control creation and execution of files in particular directories.
- If not already present, consider deploying a demilitarized zone (DMZ) between your webfacing systems and the corporate network. Limiting the interaction and logging traffic between the two provides a method to identify possible malicious activity.
- Ensure a secure configuration of web servers. All unnecessary services and ports should be disabled or blocked. All necessary services and ports should be restricted where feasible. This can include whitelisting or blocking external access to administration panels and not using default login credentials.
- Utilize a reverse proxy or alternative service, such as mod_security, to restrict accessible URL paths to known legitimate ones.
- Establish, and backup offline, a "known good" version of the relevant server and a regular change-management policy to enable monitoring for changes to servable content with a file integrity system.
- Employ user input validation to restrict local and remote file inclusion vulnerabilities.
- Conduct regular system and application vulnerability scans to establish areas of risk. While this method does not protect against zero day attacks it will highlight possible areas of concern.
- Deploy a web application firewall and conduct regular virus signature checks, application fuzzing, code reviews and server network analysis.

Detection

Due to the potential simplicity and ease of modification of web shells, they can be difficult to detect. For example, anti-virus products sometimes produce poor results in detecting web shells.

The following may be indicators that your system has been infected by a web shell. Note a number of these indicators are common to legitimate files. Any suspected malicious files should be considered in the context of other indicators and triaged to determine whether further inspection or validation is required.

- Abnormal periods of high site usage (due to potential uploading and downloading activity);

- Files with an unusual timestamp (e.g., more recent than the last update of the web applications installed);
- Suspicious files in Internet-accessible locations (web root);
- Files containing references to suspicious keywords such as cmd.exe or eval;
- Unexpected connections in logs. For example:
    - A file type generating unexpected or anomalous network traffic (e.g., a JPG file making requests with POST parameters);
    - Suspicious logins originating from internal subnets to DMZ servers and vice versa.
- Any evidence of suspicious shell commands, such as directory traversal, by the web server process.

For investigating many types of shells, a search engine can be very helpful. Often, web shells will be used to spread malware onto a server and the search engines are able to see it. But many web shells check the User-Agent and will display differently for a search engine spider (a program that crawls through links on the Internet, grabbing content from sites and adding it to search engine indexes) than for a regular user. To find a shell, you may need to change your User-Agent to one of the search engine bots. Some browsers have plugins that allow you to easily switch a User-Agent. Once the shell is detected, simply delete the file from the server.

Client characteristics can also indicate possible web shell activity. For example, the malicious actor will often visit only the URI where the web shell script was created, but a standard user usually loads the webpage from a linked page/referrer or loads additional content/resources. Thus, performing frequency analysis on the web access logs could indicate the location of a web shell. Most legitimate URI visits will contain varying user-agents, whereas a web shell is generally only visited by the creator, resulting in limited user-agent variants.

References

Australian Cyber Security Centre – Securing Content Management Systems (CMS)

FireEye China Chopper – The Little Malware That Could. Detecting and Defeating the China Chopper Web Shell

MANDIANT – Old Web Shells New Tricks

FireEye – Breaking Down the China Chopper Web Shell Part I

FireEye – Breaking Down the China Chopper Web Shell Part II

WSO Information

Exploit-db – China Chopper

C99

INFOSEC Institute – Web Shell Detection

Revisions

November 10, 2015: Initial Release|November 13, 2015: Changes to Title and Systems Affected sections|August 9, 2017: Updated c99 link

Source: https://www.us-cert.gov/ncas/alerts/TA15-314A