Risk Advisory: Web Shells

Web shells are software programs or scripts that are run on a web server to allow remote administration. They may be used for legitimate purposes, but they are often installed by cybercriminals and other adversaries to gain unauthorized access to systems and networks, including those at universities.

Web shells used for malicious purposes are delivered by exploiting server configuration weaknesses or web application vulnerabilities. They may be installed on network device management interfaces as well as content management systems and platforms, such as WordPress and Drupal.

Web shells may be as small and simple as one line of code, so malicious shells are easily concealed among website files. Cybercriminals can hide communications with web shells in encrypted HTTPS or encoded plaintext, confounding detection by firewalls, intrusion detection systems, and anti-virus and anti-malware software. Attackers may persist on the compromised network, using the unauthorized access for stealing data and a variety of other nefarious purposes. It was reported in February 2020 that Microsoft tracks an average of 77,000 active web shells spread across 46,000 servers.

How Web Shells Work

Adversaries take advantage of common web page vulnerabilities such as SQL injection, remote file inclusion (RFI), or cross-site scripting (XSS) in conjunction with social engineering tactics to attain file upload capabilities and transfer the malicious files. Once installed, web shells are used for remote administration of the affected systems. The attacker's functionality and privileges depend on the server's configuration, but adversaries may be able to:

1. Deface web pages.
2. Add, delete, and execute files.
3. Run operating system commands and shell scripts.
4. Harvest and steal confidential data and credentials.
5. Escalate access privileges, maintain persistence and gather information, and move across networks.
6. Upload additional malware to use the site for phishing, distributing malware, and other attacks.
7. Establish a relay point to issue commands to computers and systems inside the network.
8. Create a command-and-control (C2) infrastructure, potentially in the form of a bot in a botnet or to attack additional external networks.

The most commonly observed web shells are written in programming languages that are widely used and supported, such as PHP and ASP, Perl, Ruby, and Python. Unix shell scripts are also used. Adversaries often route malicious traffic across Internet-facing and internal networks by chaining web shells on compromised systems together.

Things to Do

Prevention

- Regularly update applications and the host operating system to ensure protection against known vulnerabilities.
- Configure web servers so that unauthorized users cannot access system utilities and directories in order to:
    - Reduce adversaries' abilities to escalate privileges or move laterally to other systems on the network.
    - Control creation and execution of files in particular directories.
- Ensure secure configuration of web servers.
- Change default login credentials.
- All unnecessary services and ports should be disabled or blocked.
- All necessary services and ports should be restricted where feasible.
- Employ user input validation to mitigate local and remote file inclusion vulnerabilities.
- Use process monitoring to detect web servers that perform suspicious actions such as running cmd.exe or accessing files that are not in the Web directory.
- Consider using file monitoring to detect changes to files in the web directory of a web server that do not match with updates to the server's content and may indicate implantation of a web shell script.
- Log authentication attempts to the server and any unusual traffic patterns to or from the server and internal network.
- Configure your server to send logs to a central log server so they cannot be modified or deleted by an attacker.
- Carefully evaluate the reputation and security practices of any plugins that you are considering using, and remove any unused application (e.g, WordPress) and server (e.g. Apache or IIS) modules and plugins.

Detection

Look for the following indicators that your system has been compromised with a malicious web shell. Note that some of these indicators are common to legitimate files, so should be considered within the context of additional signs of compromise. Further analysis should be done to determine whether or not a system is compromised.

- 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 or unusual web requests in logs. For example, a file type generating unexpected or anomalous network traffic, such as a JPG file making requests with POST parameters
- Any evidence of suspicious shell commands by the web server process, such as directory traversal

The above is not an exhaustive list of tactics for prevention and detection. Review the resources below for more information and best practices.

Resources

CISA: Compromised Web Servers and Web Shells – Threat Awareness and Guidance
Mitre ATT&CK: Server Software Component: Web Shell
Microsoft Security Blog: Ghost in the shell: Investigating web shell attacks
ZDNet: Microsoft says it detects 77,000 active web shells on a daily basis
NSA|CSS: Detect & Prevent Cyber Attackers from Exploiting Web Servers via Web Shell Malware
NSA Guidance: Detect and Prevent Web Shell Malware (pdf)
FireEye Blog: Breaking Down the China Chopper Web Shell – Part I
Bleeping Comuter: Feature-rich Ensiko malware can encrypt, targets Windows, macOS, Linux (July 2020)