SecurityMaverick.com
Radical and Ordinary Thoughts on InfoSec and the Grind

CVE-2013-4339: Two Exploits for WordPress 3.6 URL Redirect Restriction Bypass

Posted on September 21, 2013 | Leave a comment

According to WordPress, version 3.6 is affected by a URL Redirect Restriction Bypass issue that allows an attacker to craft a URL in such a way that, should it be clicked, would take the victim to a site of the attacker's choice via the Location: tag in a 302 Redirect. Current descriptions of the WordPress issue may be found at WordPress (1, 2), Mitre and OSVDB.

I can confirm that this issue exists in versions 3.1 and 3.6. Due to the range of versions, I assume that it exists in all releases in between but I have not confirmed it. If running an outdated version of WordPress please upgrade to the latest version, which is 3.6.1 at the time of this writing. The current 3.6.1 release fixes four additional issues (1, 2, 3, 4) as well.

There are two attack vectors: the first is a full URL redirect; the second is a partial redirect, as the victim is automatically taken to a WordPress error page with the attacker's link embedded in a prominently displayed <a href> tag.

In order for either exploit to work (assuming the victim clicks on the link) *two conditions must be true*: first, the victim must be logged into the site (wp-admin); and second, they must have permission to the editing page (edit-tags.php or edit-comments.php). If either of these two conditions are false the vulnerable code will not be reached.

Exploit 1: edit-tags.php Full URL Direct

```
[vulnerable site]/wp-admin/edit-tags.php?action=delete&_wp_http_referer=http://securitymaverick.com?edit-tags.php
```

The above exploit URL will do a full redirect when the link is clicked on by the victim. It is important to note that in order for the exploit to work the string 'edit-tags.php' must be present in the _wp_http_referer parameter. For example:

```
_wp_http_referer=http://securitymaverick.com?edit-tags.php
```

In this case I chose to put it after a '?' and by doing so I create a dummy parameter so that any valid URL before the '?' executes on the attacker's server. Under normal circumstances servers such as Apache and IIS will not view web application parameters (the stuff after the '?') as pointing to valid server files to execute. On the application side, WordPress should discard the extra parameter.

Alternatively an attacker may create a URL such as:

```
_wp_http_referer=http://securitymaverick.com/edit-tags.php
```

This URL has a '/' instead of a '?'. This means that the attacker would need to create a valid page named 'edit-tags.php' on the attacker's server in order to have their code run.

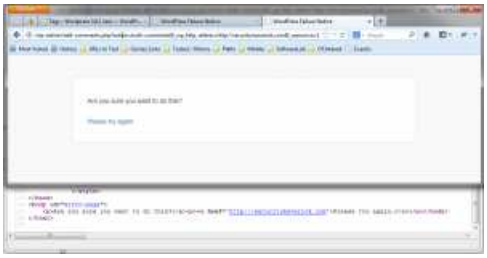
The normal WordPress login code prevents access to any page under wp-admin if the user has not authenticated. After authentication, the code in edit-tags.php below prevents users without permission from accessing the page:

```
if ( ! current_user_can( $tax->cap->manage_terms ) )
    wp_die( __( 'Cheatin' uh?' ) );
```

Exploit 2: edit-comments.php Embedded Link Partial Redirect

```
[vulnerable site]/wp-admin/edit-comments.php?action=bulk-comments&_wp_http_referer=http:
//securitymaverick.com&_wpnonce=1
```

The above exploit URL will embed the attackers link in the error page of WordPress with the text "Please try again" as seen in the image below.



Similar to Exploit 1, the normal WordPress login code prevents access to any page under wp-admin if the user has not authenticated. After authentication, the code in edit-comments.php below prevents users without permission from accessing the page:

```
1 if ( !current_user_can('edit_posts') )
2     wp_die(__('Cheatin' uh?'));
```

RELATED POSTS:

- Publicly Disclosed Vulnerabilities & Exploits between...
- Vulnerability: CVE-2013-1422 – WebCalendar 1.2.5 &
- Malformed HTML & XSS Character Filtering: A Few Lessons
- Vulnerability: OrangeHRM 2.7.1 Vacancy Name Persistent XSS
- PHP uniqid() and LCG non-randomness write-up

This entry was posted in Vulnerabilities and tagged articles, bugs, ethical hacking, hacking, software analysis, wordpress. Bookmark the permalink.
