Google XSS

Monday, April 14th, 2008

Now , normally w hen I find an XSS vulnerability on a popular domain I just report it to the appropriate security team and move on, but this one is interesting…

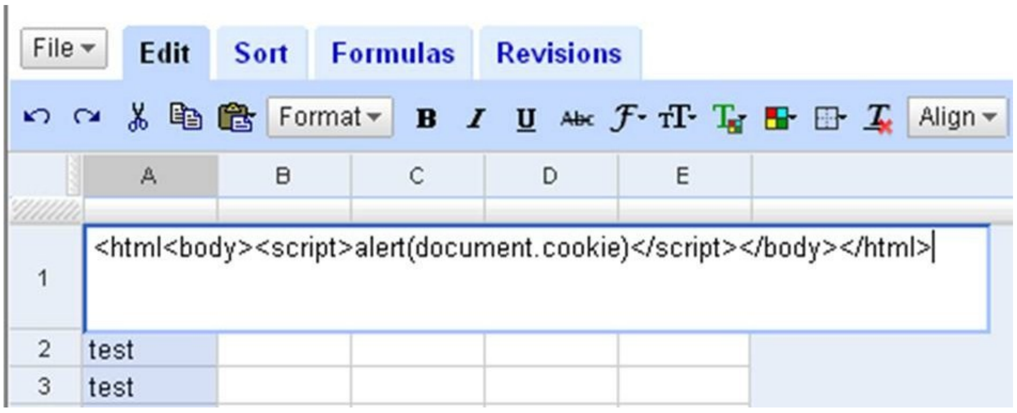
By taking advantage of the content-type returned by spreadsheets.google.com (and a caching flaw on the part of Google), I w as able to pull off a full blow n XSS against the google.com domain. For those of you w ho don't understand w hat this means, allow me to elaborate. When Google sets their cookie, it is valid for all of their sub domains. So, w hen you log into gmail (mail.google.com), your gmail cookie is actually valid for code.google.com, docs.google.com, spreadsheets.google.com…and so on. If someone (like me) finds an XSS vulnerability in any one of these sub domains, I'll be able to hijack your session and access any google service as if I w ere you.

So, in this instance, I have an XSS on spreadsheets.google.com. With this single XSS, I can read your Gmail, backdoor your source code (code.google.com), steal all your Google Docs, and basically do w hatever I w ant on Google as if I w ere you! Google's use of "document.domain=" also make things a little easier to jump from one domain to the next, but that's another story…

This particular XSS takes advantage of how Internet Explorer determines the content type of the HTTP response being returned by the server. Most w ould think that explicitly setting the content-type to something that isn't supposed to be rendered by the brow ser w ould easily solve this issue, but it does not. IE isn't the only brow ser that w ill ignore the content-type header in certain circumstances, Firefox, Opera, and Safari w ill ignore the content-type header as w ell (in certain circumstances). Security professionals and more importantly developers need to understand the nuances of how the popular w eb brow sers handle various content-type headers, otherw ise they may put their w eb application at risk of XSS. The most comprehensive paper I've seen on the subject w as w ritten by Blake Frantz of Leviathan. The paper can be found here. It's a "MUST HAVE" reference for w eb app security pros. Read it, understand it, protect yourself appropriately or expect others to exploit appropriately…

In this issue, Google set the content-type header for a response w hich I controlled the content to text/plain. If I can inject w hat looks like HTML into the first few bytes of the response, I'll be able to "trick" Internet Explorer into rendering the content as HTML. Luckily for me, I w as able to do just that.

I created a spreadsheet on spreadsheets.google.com and for the first cell (A1) I put the follow ing content: "<HTML><body><script>alert(document.cookie)</script></body></HTML>"

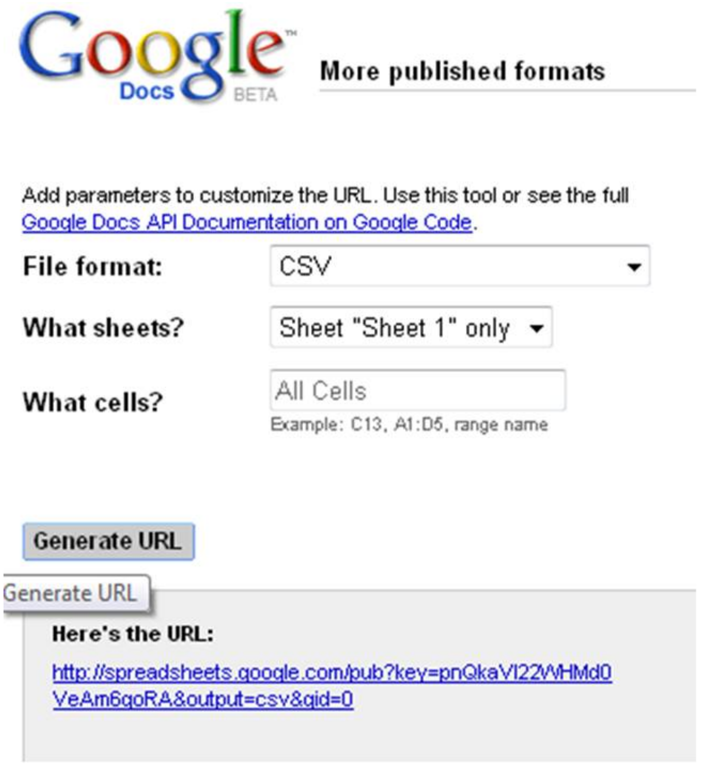


	A	B	C	D	E
1	<html<body><script>alert(document.cookie)</script></body></html>				
2	test				
3	test				

I then saved the spreadsheet and generated a link for the spreadsheet to be served as a CSV.

When this option is selected, the contents of the spreadsheet are displayed inline (the content-disposition header w as not explicitly set to "attachment"), IE ignores the content-type header, sniffs the content-type from the response, then proceeds to render the response as if it w ere HTML. At this point, I control the entire HTML being rendered under an xxx.google.com domain.

To be fair, Google included a subtle defense to protect against content-type sniffing (padding the response), but those protection measures failed (w ith a little prodding by me). The issue is fixed, but if you try to reproduce this issue, you'll see their defense in play. It a solid defense w hich show s they understand the nuances of content-type sniffing.



Add parameters to customize the URL. Use this tool or see the full Google Docs API Documentation on Google Code.

File format: CSV

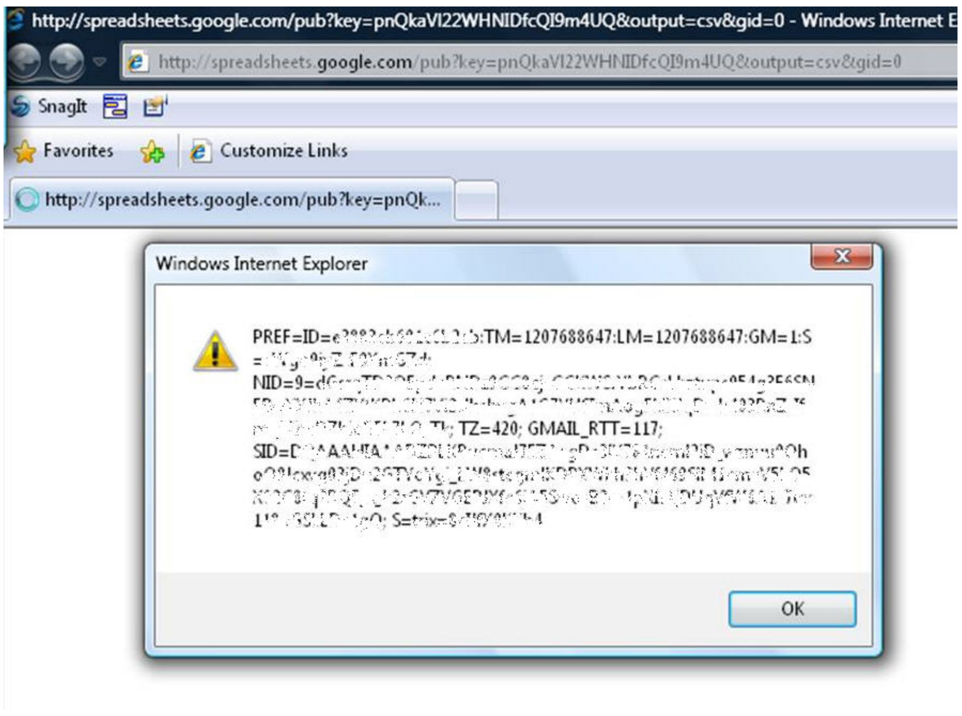
What sheets? Sheet "Sheet 1" only

What cells? All Cells

Example: C13, A1:D5, range name

Here's the URL:

http://spreadsheets.google.com/pub?key=pnQkaVI22WHMd0VeAm6qoRA&output=csv&gid=0


http://spreadsheets.google.com/pub?key=pnQkaVI22WHNIDfcQI9m4UQ&output=csv&gid=0 - Windows Internet E
http://spreadsheets.google.com/pub?key=pnQkaVI22WHNIDfcQI9m4UQ&output=csv&gid=0
SnagIt
Favorites
Customize Links
http://spreadsheets.google.com/pub?key=pnQk...
Windows Internet Explorer
OK

```
PREF=ID=[redacted]:TM=1207688647:LM=1207688647:GM=1:S
=[redacted]
NID=9=[redacted]
[redacted]; TZ=420; GMAIL_RTT=117;
SID=[redacted]
[redacted]; S=trix=[redacted]
```

I'll provide some tips on taking ownership of untrusted content and serving it from your server in a later post, but for now take a look at the paper written by Blake Frantz. I'm sure it will open some eyes…