XSS and Open Redirect on Telegraph.co.uk Authentication pages

11 August 2014 · security, vulnerability

I recently found a couple of security issues with the Telegraph.co.uk website. The site contained an Open redirect as well as an XSS vulnerability. These issues were in the authentication section of the website, https://auth.telegraph.co.uk/ . The flaws could provide an easy means to phish customer details and passwords from unsuspecting users.

I informed the telegraph's technical management, as part of a responsible disclosure process. The telegraph management forwarded the issue report and thanked me the same day. (12th May 2014)

The fix went live between the 11th and 14th of July, 2 months after the issue was reported.

The details:

The code served via auth.telegraph.co.uk appeared to have 2 vulnerabilities, an open redirect and a reflected Cross Site Scripting (XSS) vulnerability. Both types of vulnerabilty are in the OWASP Top 10 and can be used to manipulate and phish users of a website. As well has potentially hijack a user's session.

Compromised URLs, that exploit these flaws would have typically have been circulated to potential victims, in emails, via twitter or facebook. The fact the web-pages were served via HTTPS, provided no added protection for the user. HTTPS was encrypting an already compromised page.

The **Open Redirect** was on the reenterPassword.htm page, and allowed any URL to be entered via a URL argument and used to override the desired value.

Simply replacing the URL with another site is onesimple attack:

https://auth.telegraph.co.uk/sam-ui/reenterPassword.htm?redirectSuccess=**http://www.example.com**

In this example, the page included this HTML:

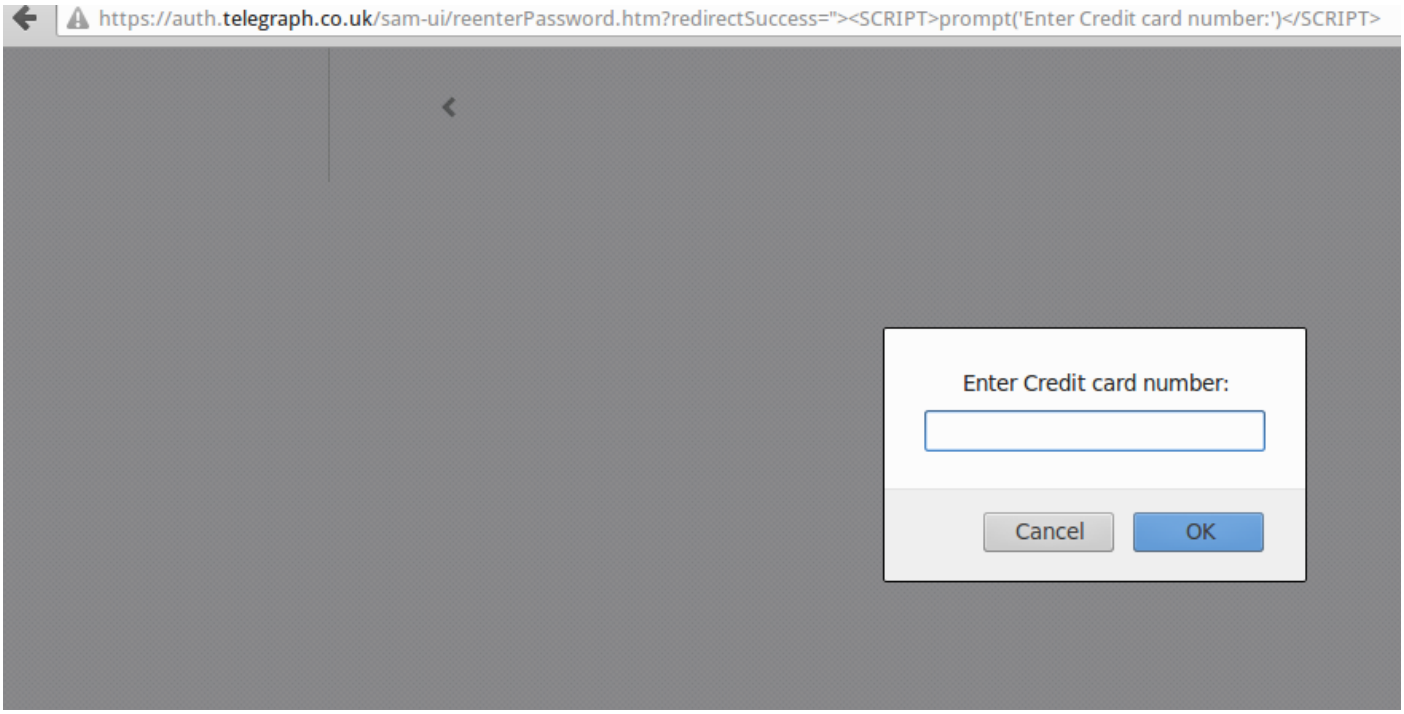
```
<input name="redirectSuccess" type="hidden" value="http://www.example.com" />
```

As the Open redirect was entirely unvalidated, an attacker could even incorporate javascript directly into the link:

https://auth.telegraph.co.uk/sam-ui/reenterPassword.htm?redirectSuccess=**javas-cript:prompt%28%27Enter%20Credit%20card%20number:%27%29**

Here the HTML returned includes our ‘dodgy’ example request for the customers credit card number:

```
<a href="javascript:prompt('Enter Credit card number:') " title="return to last page
visited">Back</a>
```



A screen capture of the affected page.

More details on this sort of vulnerability and how it can be mitigated can be found on the OWASP site (https://www.owasp.org/index.php/Top_10_2013-A10-Unvalidated_Redirects_and_Forwards).

The **Reflected XSS** issue was discovered on the login.htm page, and allowed a URL and arbitrary javascrpt code to be included in the plink URL argument.

An attack URL might look like this:

https://auth.telegraph.co.uk/sam-ui/login.htm?logintype=lite&plink=**http://www.example.com%22%3E%3CFORM%20on-click=%22alert%28%27HACKED%27%29%22%20name=%22**

And resulted in the following HTML being inserted into the page:

```
<a href="http://www.example.com"><FORM onclick="alert('HACKED')" name="?
command=slideUpLight" id="link_id" class='closeLink' title="close the login window"></a>
```

As you can see, clicking on the Form would have resulted in the alert message ‘HACKED’ being presented to the customer. In a real exploit, the attackers might choose to insert more subtle code or requests for information into the page to steal or phish a users details or session.

More details on this sort of vulnerability and how it can be mitigated can be found on the OWASP site (https://www.owasp.org/index.php/Top_10_2013-A3-Cross-Site_Scripting_%28XSS%29).

Details on a similar flaw in the Guardian’s web site, found last yeah can be found here (http://www.investigatingsoftware.co.uk/2013/07/web-application-security-testing.html).