Browser Plugins and Extensions Survey

Saarthik Tannan
Outline

• Background Information
  • 1.) Definition of plugins and extension
  • 2.) Examples
  • 3.) Statistics

• Section 1: Browser extensions description
  • 1.) Vulnerabilities of browser extensions and how extensions are used to find them
  • 2.) Description of the Firefox extensions architecture
  • 3.) Description of the Chrome extensions architecture

• Section 2: Description of some of the attacks that occur
  • 1.) Attacks from extensions
  • 2.) Attacks on extensions

• Section 3: Description of some solutions to attacks
  • 1.) General solutions
  • 2.) Specific solutions

• Case Study
• Conclusion
Background

• Extensions and plugins are available for many browsers
• They have been available since 1999, starting with Internet Explorer version 5

• Plugins
  • It is a third party library which is embedded in a web page
  • Many plugins use JavaScript
  • Plugins only affect web pages that refer and require specific plugins to show specific content
  • Plugins can be installed from an application (i.e. Adobe Reader) or downloaded from a website
  • Plugins can be enabled or disabled. To remove them the program has to be uninstalled from the operating system; the browser cannot uninstall them

• Extensions
  • Extensions only affect the web browser, by adding extra features and functionality. They process the page which the browser loads
  • Extensions can be installed from a store (i.e. Google Chrome store)
  • They can be enabled or disabled or uninstalled from the browser
Examples of Plugins and Extensions

• Some examples of plugins include:
  • Macromedia Flash
  • Microsoft Silverlight
  • Apple Quicktime
  • Adobe Reader

• Some examples of extensions include:
  • Adblock Plus
  • Google Mail Checker
  • Adobe Acrobat
Some Statistics for Extensions and Plugins

1) According to Nicholas Golubovic, “roughly 20 million users on addons.mozilla.org, extensions like Adblock Plus are used over the world. Google Chrome, currently leading the global browsers usage statistics, features a wide range of extensions in its associated Chrome Web Store, some of them amassing over 10 million active users.”

2) In the paper “Verified Security for Browser Extensions,” the researchers conducted an empirical study of over 1,000 Chrome extensions and found that a third of them have access to a user’s data on arbitrary websites and 60% have access to a user’s entire browsing history.

3) In the paper “Protecting Browsers from Extension Vulnerabilities,” the researchers examined 25 popular Firefox extensions and found that 3 out 25 extensions require full system access and the other 22 extensions were over privileged.

4) In the paper “Trends and Lessons from Three Years Fighting Malicious Extensions,” the researchers reviewed 100,000 extensions over a three-year period from January 2012 through 2015 and found that one in ten were malicious. WebEval (system that identifies malicious extensions) had a detection rate of 96.5%. The researchers removed 9,523 malicious extensions from the Chrome Web Store.
Some Statistics for Extensions and Plugins Cont.

- The following graph is taken from the Symantec 2016 “Internet Security Threat Report”\(^7\)

Note Adobe is Adobe Flash Player
Section 1: Browser Extensions Description
Extension Vulnerabilities and Solutions

• Many of the extensions are created by users who do not have a security background so there are vulnerabilities in them
  • In Firefox, extensions run with full privileges, which is a problem as attackers may exploit the vulnerabilities
• One vulnerability is the Cross-site scripting vulnerability, which can be exploited to install malware onto a user’s computer
• One solution to protecting against Firefox extensions vulnerabilities is to create new extension system in which extensions specify which privileges they require at install time and at run time they only have access to those privileges
How to Find Extensions Vulnerabilities

• VEX is a framework for finding vulnerabilities of Firefox extensions

• The following diagram comes from the paper “VEX: Vetting Browser Extensions For Security Vulnerabilities”

![Diagram](image-url)
Firefox Extensions Architecture

• In Firefox, extensions are known as add-ons
• They run with full privileges, can read and write arbitrary files, and can start new processes
• The same interface, XPCOM (Cross-platform Component Object Model) interface, is used for extensions and the internal browser components, which is a problem
• XPCOM is the Object system which Firefox uses
• Normally the XPCOM interface is implemented by the internal browser components, but the extensions can replace it
Chrome Extensions Architecture

• The Google Chrome Extension platform was created to protect against vulnerabilities of buggy extensions

• It focuses on three security methods:
  • Privilege separation
    • Chrome extensions consist of two types of components, the content script (includes zero or more of them) and the core extension (includes zero or one of them)
    • Content scripts read and modify websites and execute with no privileges. Core extensions do not directly interact with websites and use the extension’s full privileges
    • These two components are separated (i.e. run in different processes), and communicate to each other by sending structured clones through an authenticated channel
    • Each website receives its own instance of a content script
  • Isolated worlds
    • This is used to protect content scripts from attacks
  • Permissions
    • They are used to reduce core extension vulnerabilities
    • Extensions cannot access parts of the browser API which affect privacy or security unless it requests them
    • I.e. an extension has to request the bookmark permission in order for it to read or modify the bookmarks
Section 2: Description of Some Attack Types
Some Attacks Using Extensions

• Some attacks using extensions are the following:
  
  • Man-in-the-Browser Phishing
    • It is an attack that uses phishing and a Trojan horse to install an extension into the browser
    • The extension is used to modify, capture, or insert information on pages without the client and server’s knowledge

  • Malvertising
    • It is malicious activity that uses advertisements to distribute malware

Image Source: “A Review of Browser Extensions, a Man-In-the-Browser Phishing”
Some Attacks Using Extensions (Cont.)

• Attacks from Firefox Extensions:
  • Privileged Attacks
    • Attacks used to execute commands, steal passwords, and write a binary file to the file system
  • Privileged Escalation
    • Attacks that use browser bugs or design flaws to increase the privileges and access the API
    • Examples of attacks are Overriding Chrome URI, and DTD Content Injection
  • Misdirection
    • Attacks which cause the victim to execute privileges; some add-ons cannot execute privileges
    • Examples of attacks are Security Indicator Re-Skin and Clickjacking Internal Pages
  • DataLeaks
    • Attacks which use the privileges that the add-ons has in order to leak data
    • Examples of attacks are File System Inclusion and CSS Attribute Extraction

• Attacks from Chrome Extensions:
  • Privileged Attacks
    • Attacks used to inject scripts into current and future tabs, read all accessible cookies, and install a PAC script
  • Privileged Escalation
    • Attacks in which an extension with a low privilege gain access to a higher privilege
    • An Example of attack is using the filesystem API to create a file with a URL
Attacks on Extensions

- Some attacks on extensions include:
  - **Fingerprinting**
    - It is the process of determining the active extensions of a victim
  - **Cross-Context Scripting**
    - It is an attack that allows code to be executed when extensions have privileges
  - **SQL Injection**
    - It is used to insert commands into the query language and influence the results
    - SQL injection occurs differently in Firefox and Chrome
  - **Clickjacking**
    - One type of technique used is Bait and Switch. The following is an example of how this technique works:
      1) User visits a website controlled by an attacker. The page will load the resources of the extension in a background tab.
      2) Next, a distraction will occur. For instance, a game may be used as a user may have to click to play it
      3) After the instructions of the game have been explained, the page can be prepared for clicking, and then switched to the background tab at some moment
      4) When the game is positioned in a certain way, it will result in the victim clicking on an extension button
      5) If the website closes the tab quickly, then the victim will not know that the attack occurred
Section 3: Description of Some Solutions
Some Solutions to Attacks

• Limit the privileges for Firefox Extensions
• To limit Man-in-the-Browser attacks the following can be done:
  • Use anti-man in the browser Trojan technology
    • Monitor for anomalous web service access
    • Use a one-time password (OTP) scheme
  • Users should use different methods to protect their privacy
    • Install anti-virus software on computer
    • Install anti-browser toolbars
    • Disable or remove any suspicious extensions
    • Clear cookies
    • Clear private data
• Make users aware of security by training them
• Use TriCipher technology
  • TriCipher Armored Credential System (TACS) allows users to enhance their authentication infrastructure. When users make a transaction they must verify it by entering a password and clicking a mouse.
• Use Rapport protection technology
  • Rapport controls communication and protects websites by using API blocking between add-ons and the browser
• Use a Virtual Cryptogram
  • This is a virtual signing technology which uses a camera in a mobile phone or an optical token

• Examples of specific tools or systems
  • Hulk is a dynamic analysis system for identifying malicious Chrome extensions
  • Expector (Extension Inspector) is a tool that automatically finds ad-injecting extensions
  • IBEX is a framework for authoring, analyzing, and verifying, and deploying secure browser extensions
Case Study – Theola Malware Attack

- Win32/Theola is a malicious component of the bootkit, Win32/Mwbroot.FX

- In January, 2013 Theola was found in the Netherlands, Norway, Italy, Denmark, and Czech Republic

- Win32/Theola.F is a Chrome plugin based on the NPAPI interface (Netscape Plugin Application Programming Interface). It has a native module and uses the CRX format (CRX Package Format).

- The CRX container has the following permissions:
Case Study – Theola Malware Attack Cont.

- Win32/Theola loads into Chrome
- After deobfuscation the first JavaScript method loads the the native module as the default-plugin
- The JavaScript Module modifies all the POST tracking methods on the page. The password input field becomes visible.
Case Study – Theola Malware Attack Cont.

• The routine `NP_GetEntryPoint()` gets access to the pointers it needs, so that it work with the plugin

• Once the plugin is running, the `addListener()` loads the JavaScript code for tracking web activity to the computer

• When activity on the bank page is detected, Win32/Theola sends the information to pipe
• When activity on the bank page is detected, Win32/Theola sends the information to the special named pipe

```c
char *cdeci_generate_pipe_name()
{
    char *result; // eax
    unsigned int v1; // edx:esi
    unsigned int v2; // eax
    unsigned int v3; // ebx
    char v4; // [sp+0h] [bp-30h]
    int v5; // [sp+8h] [bp-ch]
    int v6; // [sp+0h] [bp-bh]
    int16_t v7; // [sp+20h] [bp-ah]
    
   Src = "\\\pipe\";
    v5 = "\\\pipe\";
    v7 = "\\\pipe\";
    if (byte_1008A900 )
    {
        result = &byte_1008A900;
    }
    else
    {
        v1 = get_system_root_fetime();
        memcpy(&v1, "\\\pipe\\\", 0x20); // ecx
        memcpy(&byte_1008A900, &Src, 9);
        v2 = 0;
        while
        {
            v3 = v1 & 0x27;
            v1 >>= 1;
            v2 = (byte_1008A900[v2] = *byte + v3);
        }
        while (v2 < 8 );
        byte_1008A900 = 0;
        result = &byte_1008A900;
    }
    return result;
}
```
Conclusion

• So as I have mentioned:
  • There are many users of extensions
  • Some plugin vulnerabilities have increased
  • Firefox and Chrome have different extension architectures
  • There are attacks from extensions and attacks to extensions
  • There are some solutions to these attacks
  • The case study shows that a malicious module can be provided as a plugin. This is hard to detect as the plugin only used documented API methods.
Sources

1) http://colonelpanic.net/2010/08/browser-plugins-vs/extensions-the-difference/
Sources Cont.


14) https://www.welivesecurity.com/2013/03/13/how-theola-malware-uses-a-chrome-plugin-for-banking-fraud/