BROWSER ATTACK POINTS STILL ABUSED BY BANKING TROJANS

Peter Kálnai
Malware Researcher
peter.kalnai@eset.cz

Michal Poslušný
Malware Analyst
michal.poslusny@eset.cz
Outline

Man-in-the-browser attacks

Banking Trojans

browserhooks for Volatility Framework
MitB Attacks

“Web browsers are not able to defend against the attacks by their own means.”
Web Browsers -

2005
MitB PoC

2010
Firefox attacked in-the-wild

2015
Opera/Chrome switch to BoringSSL

2015
HTTP/2 introduced

2007
The name by P. Gühring

2011
Chrome attacked in-the-wild
FSOCIETY

features

- Retained Startup (UAC Privilege Escalation Retention)
- HTTP / 2 Huffman Decoder
- SPDY Support
- QUIC Support (for google services)
- PE Injection
- Thread Safe

prices

$250 /bin
or $2500 for builder + reseller rights

Supported browsers
- Chrome (x86), x64 Coming next update!
- FireFox (x86), x64 Coming next update!
- Microsoft Edge coming next update! x64 & x86

Supported Protocols (All Browsers Specified Above)
- SPDY
- HTTP / 2
- SSL
- HTTP / 1.1
- QUIC

- HUFFMAN Decoder for HTTP / 2 (Plaintext headers!)

Neutrino v5.1 Builder [0x22]

[ root@neutrino v5.1 ~] Hello World!
[ root@neutrino v5.1 ~] mv /home/user/* /dev/null
[ root@neutrino v5.1 ~] Wut?
[ root@neutrino v5.1 ~] Greetings from 0x22
Attackers’ goals

1) Locating a browser’s process memory
2) Injecting a payload
3) Locating the attack points
   • Chromium-based projects only (SSL VMT)
4) Installing hooks
   • Inline hooks pointing to the payload
Attack points

HttpOpenRequest
InternetReadFile
HttpSendRequest
InternetWriteFile
...

Disable HTTP2:

![Internet Options settings](image)
Attack points

nspr4dll!PR_Read
nspr4dll!PR_Write
nss3!PR_Read
nss3!PR_Write

Disable HTTP2:

```bash
firefoxPref network.http.spdy.enabled false
firefoxPref network.http.spdy.enabled.http2 false
firefoxPref network.http.spdy.enabled.v3-1 false
```
SSL VMT:

```
static const SSL_PROTOCOL_METHOD kTLSProtocolMethod = {
  0 /* is_dtls */,
  ssl3_new,
  ssl3_free,
  ssl3_get_message,
  ssl3_read_message,
  ssl3_next_message,
  ssl3_read_app_data,
  ssl3_read_change_cipher_spec,
  ssl3_read_close_notify,
  ssl3_write_app_data,
  ssl3_dispatch_alert,
  ssl3_supports_cipher,
};
```
Banking Trojans

“Gangs behind banking bots are persistent in their implementations of MiTBs”
Win/PSW.Papras (2013)

```c
if ( result )
{
    switch ( browserType )
    {
        case INTERNET EXPLORER:
            return HookIE();
        case FIREFOX:
            return HookFirefox();
        case CHROME:
            WSACleanup();
            dword_1001C00C = CreateThread(0, 0, ChromeThreadProc, 0, 0, 0);
            result = dword_1001C00C != 0;
            break;
    }
    return result;
}
```

```c
void __stdcall __noretur
```
Win/Dridex

1) Pointer in the .text section (pattern-based search)

2) Left: patterns; Right: version checks

```c
if (chromeVersion >= 0x907005A) // 42.0.2311.90
    {
        if (chromeVersion >= 0x9350041) // 43.0.2357.65
            {
                if (chromeVersion >= 0x9960055) // 45.0.2454.05
                    {
                        if (chromeVersion >= 0x9DE0049) // 47.0.2526.73
                            {
                                if (chromeVersion >= 0xA040061) // 48.0.2564.97
                                    {
                                        sslClose = sslClass->vTable;
                                        if (chromeVersion >= 0xA9F0057) // 49.0.2623.87
                                            {
                                                if (chromeVersion >= 0xAE10074) // 53.0.2785.116
                                                    {
                                                        sslWrite = &sslClass->vTable[5];
                                                        sslRead = &sslClass->vTable[2];
                                                    }
                                            }
                                        else
                                            {
                                                sslWrite = &sslClass->vTable[7];
                                                sslRead = &sslClass->vTable[4];
                                            }
                                    }
                            }
                    }
            }
    }
```
<table>
<thead>
<tr>
<th>Chrome version</th>
<th>Release date</th>
<th>Dridex version</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>40.0.2214.115</td>
<td>19.2.2015</td>
<td>2.093</td>
<td>11.3.2015</td>
</tr>
<tr>
<td>42.0.2311.90</td>
<td>14.4.2015</td>
<td>2.108</td>
<td>17.4.2015</td>
</tr>
<tr>
<td>43.0.2357.65</td>
<td>19.5.2015</td>
<td>3.011</td>
<td>26.5.2015</td>
</tr>
<tr>
<td>44.0.2403.89</td>
<td>21.7.2015</td>
<td>3.073</td>
<td>6.8.2015</td>
</tr>
<tr>
<td>45.0.2454.85</td>
<td>1.9.2015</td>
<td>3.102</td>
<td>25.9.2015</td>
</tr>
<tr>
<td>47.0.2526.73</td>
<td>1.12.2015</td>
<td>3.154</td>
<td>7.12.2015</td>
</tr>
<tr>
<td>49.0.2623.87</td>
<td>8.3.2016</td>
<td>3.188</td>
<td>10.3.2016</td>
</tr>
<tr>
<td>58.0.3029.81</td>
<td>19.4.2017</td>
<td>4.048</td>
<td>16.5.2017</td>
</tr>
</tbody>
</table>
### Win/Spy.Ursnif

**Attack points lookup**
- “instrumentation” of the browser process

**Registry storage:**
- Checksum (chrome)
- Offset (SSL VMT)

```c
// Browser process instrumentation

if (flags & 0x1)
    goto LABEL_42;

if (flags & 0x2)
    goto LABEL_42;

w4 = sub_1007FCCD(x1);
_LABEL_42:
    w26 = (flags[0] & 8) >>= 16;
    goto LABEL_42;

w6 = strlen("\n\n\n\n");
w6 = w6 - w6 + 1;
char *w6;
if (w6 < 0)
    goto LABEL_42;
```
Win/Spy.Ursnif

- Attack points is SSL VMT replaced
Win/Spy.Ursnif.AX (St. Nicholas Case)

- Loose conditions to locate SSL VMT
- However, support for new Chrome releases lost easily
- Strict opcode condition left the recent 64-bit Chrome unsupported
Win/Spy.Ursnif.AX (St.Nicholas Case)

Locating attack points starting from legacy to recent variants 😊

```c
int HookBrowser()
{
    int result; // eax@4
    HMODULE v1; // eax@5

    switch ( browserType_ )
    {
        case INTERNET_EXPLORER:
            if ( LoadLibraryA("WININET.DLL") )
            {
                char* dllName[0] = "WININET.DLL";
                result = HookFunctions(&wininetHooks, 13);
            }
            else
            {
                result = 126;
            }
            break;
        case FIREFOX:
            result = HookFirefox();
            break;
        case CHROME:
            v1 = GetModuleHandleA("CHROME.DLL");
            if ( v1 )
                result = HookChrome(v1);
            else
                result = HookFunctions(&loadlibraryExHook, 1);
            break;
    }
    return result;
}
```
Win/Qbot

```c
BOOL __stdcall DllEntryPoint(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpReserved)
{
    CHAR OutputString; // [esp+10h] [ebp-80h]

    hModule = hinstDLL;
    if ( fdwReason == 1 )
        {
            Heap::Init();
            if ( DecryptStrings.ResolveAPIs(1) < 0 )
                return 0;
            sub_1000D31C(hinstDLL, lpReserved, 0, 1);
            sub_1000B826(&OutputString, 128, "%s %#x", dword_10022AD0);
            OutputDebugStringA(&OutputString);

            MH_Initialize();
            PrepareHooks_IE();
            PrepareHooks_Firefox();
            PrepareHooks_Chrome();
            if ( sub_1000645() )
                sub_10002C19();
            MH_EnableAllHooks();
```

Win/Tinukebot

- Original author — 18-year-old Augustin Inzirillo (from France)
- His project shared with his contacts
- These guys tried to profit off
- Augustin got sad → the sources released on Github for free for grabs

“I am very worried for him, because some technology company told him they wanted to fly him to the U.S. for a job interview as a result of him posting that online,” Daniel Inzirillo, Augustin’s father
Win/Tinukebot

- Code borrowed from the WebPageTest project supported by Google
  https://github.com/WPO-Foundation/webpagetest/

```c
static SSL_METHODS_SIGNATURE methods_signatures[] = {
    // August 2016 - hhlen is
    // Chrome 53
    {0, // switched for ssl max DWORD
     53,
     54,
     21,
     8,
     "\x0\x0\x0\x3\x4\x3\x0\x0",
     0,
     -1,
     2,
     3,
     -1,
     -1,
     7,
     10},

    // Nov 2015
    {52, // Ended in
     0, // No start version
     13, // count
     4, // Signature len
     "\x0\x0\x0\x0",
     (const void **)4,
     9,
     0,
     1,
     -1,
     -1,
     3,
     1,
     0},

    // Responsible for
    // SSL files
    6}
};
```
## Summary of Hooking types

<table>
<thead>
<tr>
<th>Hooking type</th>
<th>Banking Trojan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replacement of a function in SSL VMT</td>
<td>Win/Spy.Ursnif-based, Win/Qadars, Win/Trickbot, Win/Zbot-based</td>
</tr>
<tr>
<td>Inline hook in SSL VMT</td>
<td>Win/Dridex, Win/Tinukebot</td>
</tr>
<tr>
<td>Custom method</td>
<td>Win/Qbot</td>
</tr>
</tbody>
</table>
## Summary of Targets

<table>
<thead>
<tr>
<th>Banking trojan</th>
<th>Latest version</th>
<th>IE</th>
<th>Edge</th>
<th>Firefox</th>
<th>Chrome 32-bit</th>
<th>Chrome 64-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Win/Dridex</td>
<td>4.057</td>
<td>✔️</td>
<td>⚠️</td>
<td>✔️</td>
<td>48-59</td>
<td>48-59</td>
</tr>
<tr>
<td>Win/TrickBot</td>
<td>1025</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>54-59</td>
<td>54-59</td>
</tr>
<tr>
<td>Win/Spy.Ursnif * Gozi/ISFB</td>
<td>2.16 b. 943</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>44-59</td>
<td>44-59</td>
</tr>
<tr>
<td>Win/Spy.Ursnif.AX</td>
<td>26.05.2017</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>49-58</td>
<td>49-57</td>
</tr>
<tr>
<td>Win/Qbot</td>
<td>25.05.2017</td>
<td>✔️</td>
<td>⚠️</td>
<td>✔️</td>
<td>48-58</td>
<td>54-58</td>
</tr>
<tr>
<td>Win/Qadars</td>
<td>04.04.2017</td>
<td>✔️</td>
<td>⚠️</td>
<td>✔️</td>
<td>48-57</td>
<td>49-57</td>
</tr>
<tr>
<td>Win/Tinukebot</td>
<td>06.06.2017</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>52-59</td>
<td>52-59</td>
</tr>
</tbody>
</table>

*Attacks also Opera*
Other active banking trojans

**Panda Banker**
Win32/Spy.Zbot.{ACM,ACY,ACZ}

**Neutrino Bot**
Win/Kasidet

**GozNym**
Win32/TrojanDownloader.Nymaim

**FormBook**
Win32/Agent.YIJ

**Kronos**
Win32/Agent.QMH

**Rovnix**
Win/Rovnix
Remarks

• Malware authors do not copy from each other
• Banking modules usually separated from the distributed binaries
• SSL_VERSION is dropped Chromium 61 ➔ may affect many current implementations
• Versioning available ➔ good to track changes
• Support for browsers: good indicator if the family is active
browserhooks

“When the plugin prints some findings, then it’s a little bit too late.”
browserhooks

https://github.com/eset/volatility-browserhooks

☑ Extending apihooks with 3 new hooking types

☑ 32-bit modules in WoW64 processes supported

☑ Integration with VolUtility GUI (Kevin Breen, 2016)

⚠ Some limitations discovered
Protecting browsers

**Banking & Payment Protection**

This secured browser can protect your personal data while you use online banking or payment websites. It provides additional security for banking transactions, credit card numbers and other sensitive personal data.

Enter the URL of your online banking website in the address bar.

Important: this browser should only be used for online banking and payment websites but not for general browsing.
Questions & Answers
### VF – browserhooks

<table>
<thead>
<tr>
<th>SSL Hooks for Chrome implemented by Qbot</th>
<th>chrome.exe</th>
<th>2220</th>
<th>chrome.dll</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inline/Trampoline</td>
<td>chrome.exe</td>
<td>2220</td>
<td>WS2_32.dll</td>
</tr>
<tr>
<td>Inline/Trampoline</td>
<td>chrome.exe</td>
<td>2220</td>
<td>WS2_32.dll</td>
</tr>
<tr>
<td>Inline/Trampoline</td>
<td>chrome.exe</td>
<td>2220</td>
<td>WS2_32.dll</td>
</tr>
<tr>
<td>Inline/Trampoline</td>
<td>chrome.exe</td>
<td>2220</td>
<td>WS2_32.dll</td>
</tr>
<tr>
<td>Inline/Trampoline</td>
<td>chrome.exe</td>
<td>2220</td>
<td>WS2_32.dll</td>
</tr>
</tbody>
</table>

Showing 1 to 10 of 10 entries
**VF – browserhooks**

<table>
<thead>
<tr>
<th>Plugin Command</th>
<th>Plugin Type</th>
<th>Date Completed</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>dlldump</td>
<td>Processes and DLLs</td>
<td>09 Sep 17 20:45:16</td>
<td></td>
</tr>
<tr>
<td>browserhooks</td>
<td>Other</td>
<td>09 Sep 17 20:35:52</td>
<td></td>
</tr>
<tr>
<td>amcache</td>
<td>Registry</td>
<td></td>
<td></td>
</tr>
<tr>
<td>apihooks</td>
<td>Processes and DLLs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Dlldump**

<table>
<thead>
<tr>
<th>#</th>
<th>Process</th>
<th>ImageBase</th>
<th>Name</th>
<th>StoredFile</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0xfffffa80023d1380L</td>
<td>6442450944</td>
<td>chrome.exe</td>
<td>File Details</td>
</tr>
</tbody>
</table>

Showing 1 to 1 of 1 entries 0x18000000

PID 2220 = chrome.exe
### VirusTotalSearch

<table>
<thead>
<tr>
<th>Engine</th>
<th>Version</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESET-NOD32</td>
<td>16161</td>
<td>a variant of Win64/Qbot.B</td>
</tr>
</tbody>
</table>

**Details**

- **FileName**: module.2220.3e9d1380.180000000.dll
- **FileSize**: 137728 bytes
- **MD5**: c346f3d3082163927e2da9e834b52e3d
- **SHA256**: 1a62c4c0fd09a91ff47ec58411614b41c2ae9

**VirusTotal - complete**

- **PermaLink**: 2017-09-29 18:48:13
- **Scan Date**: 2017-09-29 18:48:13
- **Results**: 7 / 63