Loading GootLoader

dinohacks.blogspot.com/2022/06/loading-gootloader.html

Niranjan Hegde

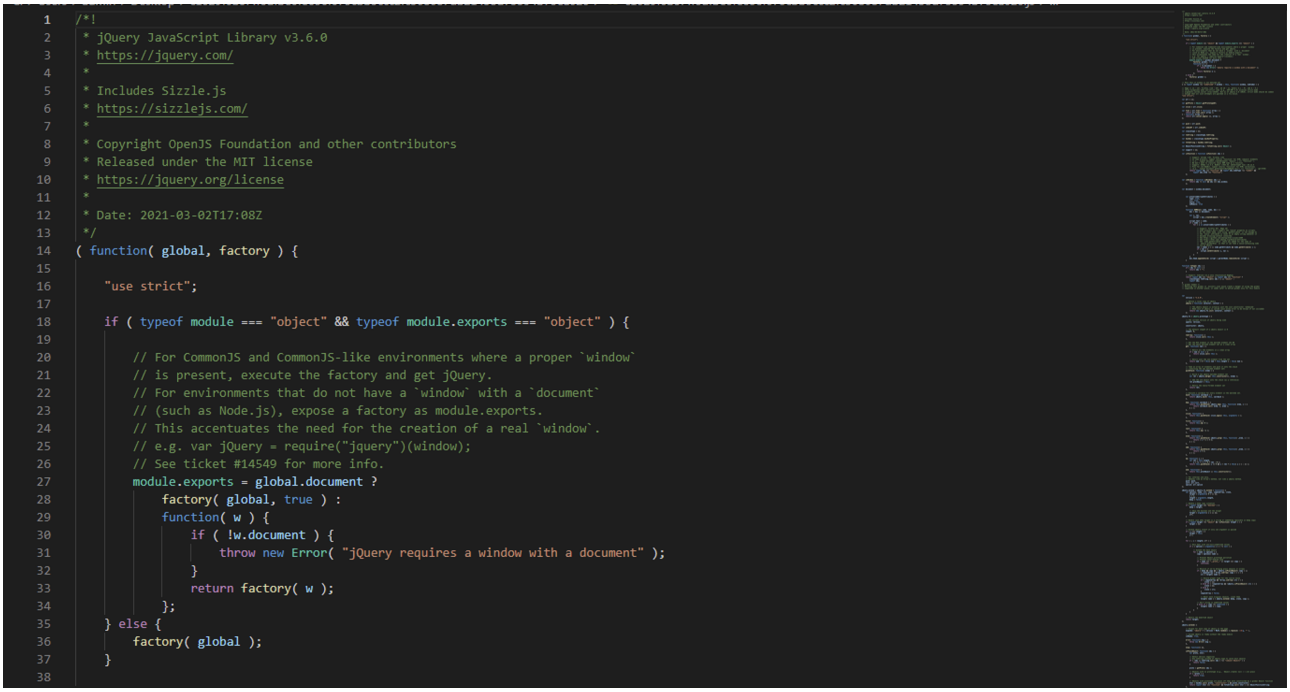
Disclaimer: Opinions expressed are solely my own. None of the ideas expressed in this blog post are shared, supported, or endorsed in any manner by my employer.

In this blog, I will be taking a look at the initial GootLoader sample (MD5: 4dd369b5e028beebe3aa5c980960c502 , Sha256: c1029f0b5f4f6dfbe0fe656f075cbb5ccc2fc308087db21438d73394b75ea020).

Available here: https://bazaar.abuse.ch/sample/c1029f0b5f4f6dfbe0fe656f075cbb5ccc2fc308087db21438d73394b75ea020/

The sample is a javascript program which is meant to be executed in windows using wscript.exe.

Opening the sample in a text editor, it appears to look like jquery library 3.6.0 (screenshot below).



```
/*!
 * 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 );
    }

```

Comparing the sample with jqeury 3.6.0 downloaded from https://code.jquery.com/jquery-3.6.0.js, differences can be observed.

The following lines shows the code inserted to create the sample:

One important thing to mention about javascript: If a variable is declared without keywords such as var, it will be treated as global variable.

After reading the code, following points can be observed:

- Function seat07() is incrementing a variable in a loop. The incremented variable is not referred in any part of the code. This is most likely done to increase the execution time and thus timeout sandbox detection. Commenting out the function would speed up the execution.
- Function gone068() contains the obfuscated string which is likely deobfuscated and executed. Generally, deobfuscated string would require one of the following to be executed:
    - eval()
    - function constructor
    - window object
- It would be interesting to see how deobfuscated string is executed in this sample.
- Function law1() and present4() are quite interesting because they are accessing array elements and making a call. They could be used to call deobfuscating routine and execute the deobfuscated string.

After adding breakpoints and running the program in a javascript debugger, first layer of the GootLoader becomes clear:

Following points can be observed:

- Deobfuscated strings are executed using function constructor.
- It contains obfuscated string which needs to be deobfuscated.
- It also seems to be accessing the registry: HKEY_CURRENT_USER

Deobfuscating the string in first layer reveals the second layer of GootLoader:
The following points can be noted:

- It checks if the environment variable userdnsdomain is set which would indicate that the system is part of AD.
- If active directory is set, then it would add the string "4173581" to parameter value in its requests.
- The C2 communication would look something like this:

- GET request to server with path: test.php?zgntopmtyplqx=
- The parameter will be <random value>4173581 if computer is part of AD otherwise only <random value>
- If the http response status code is anything other than 200, it would sleep for 12345 milliseconds and check for another domain.
- If the http response status code is 200, then it would look for the string : "@<parameter value sent>@" in the response text.
    - if it is not present, then it will sleep for 23232 milliseconds.
    - If it is present, then it will do the following:
        - Remove the string "@<parameter value sent>@" in the response text.
        - Response text is likely to be series of numbers which are parsed as integers then converted to ascii strings which are then deobfuscated.

If you are interested to read about this campaign, you can read the following blogs:

- https://news.sophos.com/en-us/2021/03/01/gootloader-expands-its-payload-delivery-options/
- https://thedfirreport.com/2022/05/09/seo-poisoning-a-gootloader-story/

You can also follow the twitter account: https://twitter.com/GootLoaderSites to get updates about the C2 server used by this campaign.

In this blog, I used a debugger to deobfuscate the sample. There is a decoder script for this campaign available
at: https://github.com/hpthreatresearch/tools/blob/main/gootloader/decode.py

You can also read the blog that explains how this decoder script was
written: https://threatresearch.ext.hp.com/tips-for-automating-ioc-extraction-from-gootloader-a-changing-javascript-malware/

Have a nice Day!

AUDI SQLi Labs Lesson 1 walkthrough

Kioptrix Level 1 walkthrough