

adfindBazarCallcobaltstrikecontiransomwaretrickbot

BazarCall to Conti Ransomware via Trickbot and Cobalt Strike

August 1, 2021

Intro

This report will go through an intrusion that went from an Excel file to domain wide ransomware. The threat actors used BazarCall to install Trickbot in the environment which downloaded and executed a Cobalt Strike Beacon. From there the threat actor discovered the internal network before moving laterally to a domain controller for additional discovery. A couple days later, the threat actors came back and executed Conti ransomware across the domain.

Unfamiliar with BazaCall/BazarCall? Read more [here](#) from [@MsftSecIntel](#), [@dreadphones](#), & [@JCearbhall](#) and [here](#) from [@Unit42_Intel](#) & [@malware_traffic](#).

Summary

In this intrusion, we observed a number of interesting techniques being leveraged by the threat actors. The threat actors were able to go from initial access to the deployment of Conti ransomware in a matter of hours. The Conti operators chose to wait a couple days before ransoming the environment. Even though most of the techniques aren’t new or advanced, they have proven to be effective. We have observed the same techniques in other intrusions and understanding these techniques will allow defenders to disrupt such intrusion activity and deny it in their own networks.

The Trickbot payload came from a [phishing campaign](#) associated with BazarCall, delivering weaponized XLSB files. Upon execution, certutil.exe was copied to %programdata% and renamed with random alphanumeric characters. Certutil was used to download and load the Trickbot DLL into memory. Trickbot was automatically tasked to inject into the wormgr.exe process and use its well-known “pwgrab” module to steal browser credentials. As part of further automated tasking, Trickbot performed an initial reconnaissance of the environment using native Windows tools such as nltest.exe and net.exe.

First hands-on activity was observed two hours after initial compromise, when Trickbot downloaded and executed Cobalt Strike Beacons. To guarantee execution on the beachhead host, multiple payloads were used. One of the Cobalt Strike Beacons was the same payload and command and control infrastructure as used in a [prior case](#). The initial access method for that case was IcedID, which shows that the threat actors utilize various initial access methods to get into environments and accomplish their goals.

Once access through Cobalt Strike was established, the threat actors immediately proceeded with domain enumeration via Nltest, AdFind, BloodHound, and PowerSploit. Presence was then expanded on the beachhead by using a PowerShell loader to execute additional Beacons.

SearchSearch

Sélectionner une langue ▼
Fourni par Google Traduction

Subscribe

Register For
Our Next
CTF

Reports

Threat
Intelligence

Detection
Rules

We observed the threat actors having technical issues. One example being with a Beacon unsuccessfully injecting into a process. It is unclear if this was an untrained actor, or there was a configuration issue.

Fifteen minutes after domain enumeration, we observed successful lateral movement to two endpoints on the network. Ten minutes after lateral movement, a PowerShell Cobalt Strike loader executed as a service on a server. Even though the execution was not successful, the threat actors kept trying, a total of eight times, until it finally worked. Windows Defender real-time monitoring was then disabled, the LSASS.exe process was dumped using [SysInternals ProcDump](#), and privilege was escalated to “SYSTEM” using [named pipe impersonation](#).

Almost four hours after initial execution, the threat actors pivoted to a domain controller using domain admin credentials and executed a Cobalt Strike Beacon. Once they had domain controller access, [ntdsutil](#) was used to take a snapshot of “ntds.dit”, saved under “C:\Perflogs\1”, for offline password hash extraction. This is a technique that we don’t see very often, but effective nevertheless.

The threat actors then reran many of the same discovery techniques that were previously executed on the beachhead, including AdFind and BloodHound. This was the last observed hands-on-keyboard activity for awhile.

Two days later, the Cobalt Strike Beacon on the domain controller was once again actively engaged by the threat actors. Psexec, with two separate batch files, were used to execute Conti ransomware on all domain-joined Windows hosts. This final deployment was executed around 6:45 UTC on a Monday morning.

From the point the threat actors returned, to ransom deployment, it was less than 30 minutes. This would give defenders little time to act if they had not identified and contained the activity from the first day of the Trickbot infection.

Services

We offer multiple services including a [Threat Feed](#) service which tracks Command and Control frameworks such as Cobalt Strike, Metasploit, Empire, PoshC2, etc. More information on this service and others can be found [here](#). Two of the Cobalt Strike servers used in this intrusion were added to our [Threat Feed](#) on 6/3/21 and the other one was added on 6/11/21

We also have artifacts available from this case such as pcaps, memory captures, files, event logs including Sysmon, Kape packages, and more, under our [Security Researcher and Organization](#) services.

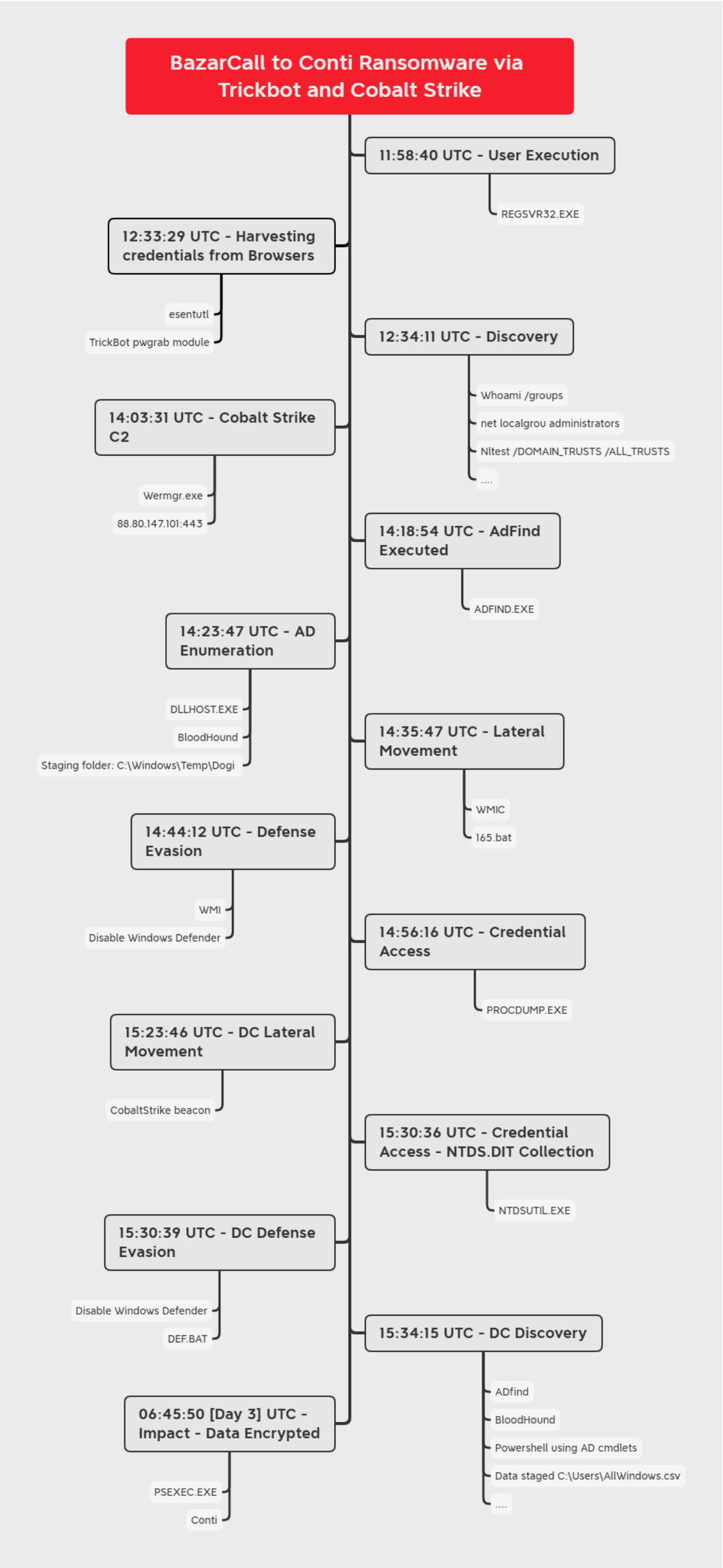
Timeline



DFIR Labs



Mentoring
and
Coaching



Analysis and reporting completed by [@_pete_0](#) and [@kostastsale](#).

Reviewed by [@RoxpinTeddy](#) and 1 unnamed contributor.

Initial Access


The initial access was achieved as a result of the user opening what appeared to be a benign workbook, a lure, requiring little user interaction.



The workbook contained hidden and password protected worksheets, these were malicious. Module functions also indicated code designed to obfuscate and hide true values and functions.

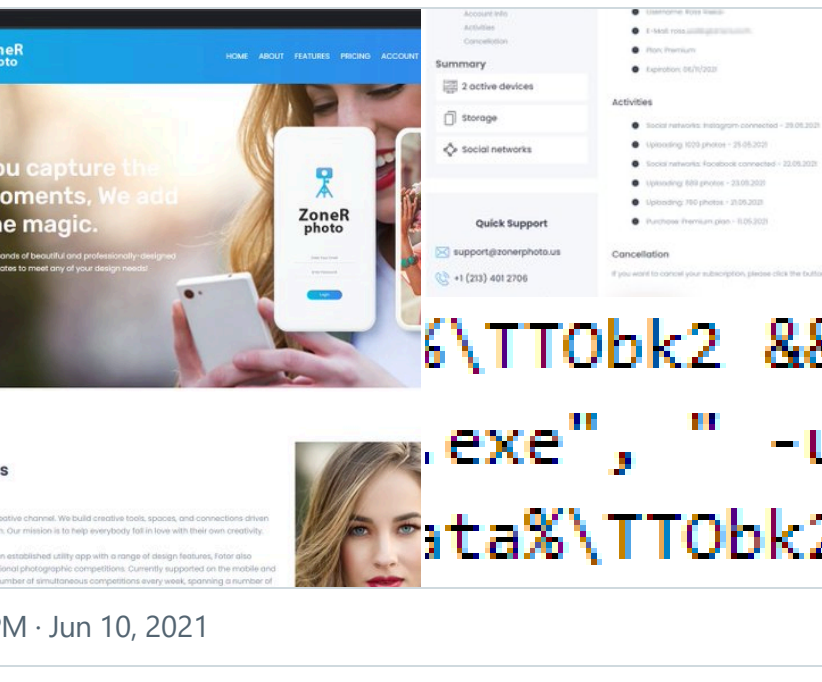
```
Public Function PuGvV(JOWETX6iQv As String) As String
    Set wubH5o = CreateObject("VBScript.RegExp")
    KfOV4I476 = Array(JOWETX6iQv)
    With wubH5o
        .Pattern = "B|j|v|D|q|P|X|M|z|L|U|Z|F|w|V|N|Q|K|I|G|H|Y"
        .Global = True
    End With
    PuGvV = wubH5o.Replace(KfOV4I476(0), "")
End Function
```

This document and the following DLL were noted as being associated to a BazarCall campaign by [@ffforward](#).






Tommy M (TheAnalyst)
 @fffoward · [Follow](#)

1/ Today [#BazarCall](#) dropped [#TrickBot](#) gtag mon311 from their brand new website /zonerphoto.us. I guess the gangs increased activity is to show that they are alive and well without that random programmer?



10:26 PM · Jun 10, 2021

 **44**
 **Reply**
 **Copy link**

Read 4 replies

Execution

From the [xlsb](#) document, the following execution chain occurs. Including copying the Windows CertUtil program and using that to collect further Trickbot payloads.

We observed a second stage execution using regsvr32 to load a DLL from the user's AppData\Local\Temp folder.

Almost immediately an outbound IPv4 address lookup was requested via HTTP. This is usually undertaken to identify the compromised environment, and to facilitate C2. The user agent refers to Curl – and used again for another stage of the intrusion.

On the beachhead, multiple executables were saved in a temp directory and then pushed into memory by TrickBot process “wormgr.exe”. The executables were identified as Cobalt Strike and communicated over port 443 to C2 88.80.147[.]101.

A PowerShell download cradle was then used to execute Cobalt Strike Beacon in memory:

Privilege Escalation

Named pipe impersonation was used to escalate to SYSTEM privileges – a common Cobalt Strike capability:

We observed several attempts by the threat actor trying to escalate to SYSTEM – ultimately succeeding, as evident in several new services running under the Local SYSTEM context:

Service creation events System Event ID 7045, coupled with unusual commands and service names are a strong indication of privilege escalation activity. RedCanary provided useful background on GetSystem capabilities of offensive security tools and [methods of detection](#).

Defense Evasion

Trickbot made extensive use of process injection to hide in benign operating system processes. It first injected into wermgr.exe and then later into svchost.exe.

Another defense evasion technique employed by Cobalt Strike, was to disable Windows Defender. WMIC was used to remotely execute ‘def.bat’. The contents of ‘def.bat’:

```
Set-MpPreference -DisableRealtimeMonitoring $true
```

Credential Access

Trickbot made use of esentutl to gather MSEdge history, webcache, and saved passwords using TrickBot’s “pwgrab” module.

LSASS was dumped remotely using ProcDump. The execution took place from the beachhead using WMIC.

“Ntdsutil” was used to take a snapshot of ntds.dit and save it under “C:\Perflogs\1”. This technique is useful for offline password hash extraction. This activity occurred twice. The same batch file, ‘12.bat’, was first executed in the context of SYSTEM; and secondly, in the context of a domain admin user. The contents of ‘12.bat’:

```
ntdsutil "ac in ntds" "ifm" "cr fu C:\Perflogs\1" q q
```

Discovery

Net and Nltest commands were used to gather network and domain reconnaissance. During the intrusion, this activity was seen multiple times, on multiple hosts.

Other discovery commands included:

```
systeminfo
nltest /dclist:<hidden>.local
nltest /domain_trusts /all_trusts
net localgroup Administrators
whoami.exe" /groups
```

AdFind.exe and adf.bat were uploaded to the beachhead. adf.bat was used to execute:

```
adfind.exe -f "(objectcategory=person)"
adfind.exe -f "(objectcategory=organizationalUnit)"
adfind.exe -f "objectcategory=computer"
adfind.exe -gcb -sc trustdmp
adfind.exe -f "(objectcategory=group)"
adfind.exe -subnets -f (objectCategory=subnet)
adfind.exe -sc trustdmp
```

AdFind results were written to the following locations:

```
C:\Windows\Temp\adf\ad_group.txt
C:\Windows\Temp\adf\trustdmp.txt
```

```
C:\Windows\Temp\adf\subnets.txt
C:\Windows\Temp\adf\ad_ous.txt
C:\Windows\Temp\adf\ad_computers.txt
C:\Windows\Temp\adf\ad_users.txt
```

On the beachhead, Cobalt Strike executed BloodHound in memory. The results were saved in:

```
"C:\Windows\Temp\Dogi"
```

BloodHound was later executed on the domain controller as well. Once again the results were stored in:

```
"C:\Windows\Temp\Dogi"
```

PowerSploit was loaded into memory on the DC and the following functions were used:

```
Get-NetSubnet
Get-NetComputer -ping
```

An encoded PowerShell command was executed on the domain controller to enumerate all AD joined hosts and save the results to:

```
"C:\Users\AllWindows.csv"
```

The decoded PowerShell command:

Lateral Movement

From the beachhead, WMIC was used to remotely execute ‘165.bat’ on two other hosts.

Multiple failed attempts were observed prior to the successful execution of a PowerShell Cobalt Strike loader via a service with “SYSTEM” privileges.

Decoded Cobalt Strike shellcode, using Cyber Chef recipe:
<https://github.com/mattnotmax/cyberchef-recipes#recipe-28—de-obfuscation-of-cobalt-strike-beacon-using-conditional-jumps-to-obtain-shellcode>

Command and Control

Multiple C2 channels were established, some were persistent whilst others appeared to be single purpose – used for payload retrieval or fallback C2. Persistent C2 activity was Cobalt Strike. The beachhead had multiple C2 channels, two of which were unique. We assess that the threat actors were ensuring a loss of a single source C2 wouldn’t result in losing all C2 to the compromised environment.

We observed a payload being retrieved from a unique IPv4 address. An indication that the threat actors were keeping C2 channels independent from payload delivery/retrieval.

Using the Curl 7.74.0 user agent:

Analysis of this binary, shows C2 activity to the following:

The binary has an unusual PDB string that indicates obfuscation:

The two persistent C2 channels were analyzed to determine the Cobalt Strike configuration. Each C2 channel was configured as follows:

- 149.248.52[.]187:443
- Onlineworkercz[.]com

(added to [Threat Feed](#) on 2021-06-11)

```
{
  "x86": {
    "sha1": "3f15a07cde64efda49670664af320603cf19e8a3",
    "sha256": "d4ab4ed720d674d4c8c35d48006724a9cf20396e020d5bd6c12fce8",
    "time": 1623422265288,
    "config": {
      "Method 1": "GET",
      "Spawn To x64": "%windir%\\sysnative\\WUAUCLT[.]exe",
      "Polling": 55490,
      "HTTP Method Path 2": "/media",
      "Port": 443,
      "Spawn To x86": "%windir%\\syswow64\\WUAUCLT[.]exe",
```

```
    "Jitter": 41,
    "C2 Server": "onlineworkercz[.]com,/kj",
    "Method 2": "POST",
    "Beacon Type": "8 (HTTPS)"
  },
  "md5": "7d9cdea210ed05a1ff96d7ff3e576c11"
},

"x64": {
  "sha1": "1d50772d506f1def4bd0659b38cf4cb41df7802c",
  "sha256": "4f009eb4252cf29daa24d1d018815aa228f0c58aba126bff3fec4cc",
  "time": 1623422268773.6,
  "config": {
    "Method 1": "GET",
    "Spawn To x64": "%windir%\\sysnative\\WUAUCLT[.]exe",
    "Polling": 55490,
    "HTTP Method Path 2": "/zh",
    "Port": 443,
    "Spawn To x86": "%windir%\\syswow64\\WUAUCLT[.]exe",
    "Jitter": 41,
    "C2 Server": "onlineworkercz[.]com,/kj",
    "Method 2": "POST",
    "Beacon Type": "8 (HTTPS)"
  },
  "md5": "23135b04a470db515db11e1364e3fcd9"
}
}
```

- 88.80.147[.]101:80
- gmbfrom[.]com

(added to [Threat Feed](#) on 2021-06-03)

```
{
  "x86": {
    "sha1": "b785cae596f7b68376464e3e300fe0aff5bea845",
    "config": {
      "Method 2": "POST",
      "Port": 80,
      "Method 1": "GET",
      "Polling": 5000,
      "Beacon Type": "0 (HTTP)",
      "Jitter": 10,
      "Spawn To x86": "%windir%\\syswow64\\dllhost[.]exe",
      "C2 Server": "88[.]80[.]147[.]101,/jquery-3[.]3[.]1[.]min[.]js",
      "HTTP Method Path 2": "/jquery-3[.]3[.]2[.]min[.]js",
      "Spawn To x64": "%windir%\\sysnative\\dllhost[.]exe"
```

```
    },
    "time": 1622753064031.5,
    "sha256": "dd0dd0b3e95ff62c45af048c0169e2631ac906da4a603cadbc7014c",
    "md5": "56830f9cc0fe712e22921a7a5a0f1a53"
  },
  "x64": {
    "sha1": "11724324f8ec1940be87553ae2bd5f96b979a5d6",
    "config": {
      "Method 2": "POST",
      "Port": 80,
      "Method 1": "GET",
      "Polling": 5000,
      "Beacon Type": "0 (HTTP)",
      "Jitter": 10,
      "Spawn To x86": "%windir%\\syswow64\\dllhost[.]exe",
      "C2 Server": "88[.]80[.]147[.]101,/jquery-3[.]3[.]1[.]min[.]js",
      "HTTP Method Path 2": "/jquery-3[.]3[.]2[.]min[.]js",
      "Spawn To x64": "%windir%\\sysnative\\dllhost[.]exe"
    }
  },
  "time": 1622753068830.2,
  "sha256": "36a5e68810f3823470fadd578efb75b5c2d1ffe9f4a16d5566f0722",
  "md5": "9dde7f14a076a5c3db8f4472b87fd11e"
}
}
```

Trickbot C2 Configuration:

<https://tria.ge/210610-vfygj4t1yn>

Exfiltration

As part of the discovery stage, we observed data being exfiltrated. The data ranged from host discovery, running processes, and user accounts:

Entire AD forest data – including usernames , DC configuration, and machine enumeration:

Impact

When, the threat actors returned two days later, the final payloads were staged by the threat actors on a domain controller in the following location:

```
C:\share$
```

Two batch scripts were executed on the domain controller to automate ransomware deployment via PSEXEC. The first was “_COPY.bat”, to stage the CONTI ransomware payload on all domain-joined computers. The second was “_EXE.bat”, to execute the staged CONTI payloads.

The batch scripts ran as expected a set of copy commands and then executed the Conti payload using psexec.

```
start PsExec.exe -accepteula @C:\share$\comps1.txt -u "domain\User" -p
```

```
start PsExec.exe -accepteula -d @C:\share$\comps5.txt -u "domain\User"
```

Files were then encrypted with the following extension [KCRAO]:

A readme.txt file was created in each folder:

The content of readme.txt:

IOCs

Network

Cobalt Strike
149.248.52.187 443
88.80.147.101 80
onlineworkercz.com
gmbfrom.com
Trickbot
116.0.6.110
123.231.149.123
146.196.121.219
177.221.39.161
180.178.106.50
85.248.1.126
94.142.179.179
94.142.179.77
88.150.240.129
46.209.140.220
85.175.171.246
89.37.1.2
94.183.237.101
103.101.104.229
103.124.145.98
114.7.240.222
131.0.112.122

123.231.149.122
45.5.152.39

File

netscan.exe
d1d579306a4ddf79a2e7827f1625581c
e141562aab9268faa4aba10f58052a16b471988a
bb574434925e26514b0daf56b45163e4c32b5fc52a1484854b315f40fd8ff8d2
12.bat
935fa508d2c41914f4549d3805456444
d40b5147e93204f03f0acfb3ad4cbb1b6f296a35
f88a59e0c1aa48aa46680f28c9e09781d3f678567f38e3b1b1ba7d2437cd9e0c
def.bat
abe4a11df74f6a2f07682174b5fb2876
e928fc3d74b976c539d55f75318b5ba89dab3f11
8a7399c37a27c46e1d61150cba71d76737233a971e0c15b07c47bcc97e710bbe
procdump.exe
6a09bc6c19c4236c0bd8a01953371a29
d1387f3c94464d81f1a64207315b13bf578fd10c
05732e84de58a3cc142535431b3aa04efbe034cc96e837f93c360a6387d8faad
tdr615.exe
a53f124fc4f07a26cc3497e665d0ec63
3f0a4ed4c0c1c5e156e4d29ac4adf109faa82cd9
12761d7a186ff14dc55dd4f59c4e3582423928f74d8741e7ec9f761f44f369e5
tdrE934.exe
d803ea86227c541c54b11bb583b3910f
f1b4faf4dfbf9ada3cc1496f9f9ad352314c2d59
48f2e2a428ec58147a4ad7cc0f06b3cf7d2587ccd47bad2ea1382a8b9c20731c
start.bat
4841c54b37729544fddcd014f09aa46e
f7d62cdca59fc09d19fa8a465ea3b2611cf797e1
f37b6c37e95f3fa27382f8b8e6256aa05e28703332bda54184e7223f82f02114
Get-DataInfo.ps1
16cde93b441e4363700dfbf34c687b08
092ac6f8d072c4cf045e35a839d5bb8f1360f1ae
a290ce75c6c6b37af077b72dc9c2c347a2eede4fafa6551387fa8469539409c7
62.dll
9e7756f47e57a03e6eb5fe7d2505b870
fb6339704bf11507038ddaf8f01324da5b71ee19
8b9d605b826258e07e63687d1cefb078008e1a9c48c34bc131d7781b142c84ab
cancel_sub_VCP1234567890123.xlsx
9e1ee4a42c381eabcf2cde38a1aae7c9
015bb306d9e54001d433b3ac2e7212b864f54ae2
fd71a2fcc0b5dd0fb0dbff257839b67749f2cadf30e2d3dae7f0e941d93d24d3

Detections

Network

ET POLICY OpenSSL Demo CA - Internet Widgits Pty (O)
ET CNC Feodo Tracker Reported CnC Server group 1
ET CNC Feodo Tracker Reported CnC Server group 2
ET CNC Feodo Tracker Reported CnC Server group 3
ET CNC Feodo Tracker Reported CnC Server group 5
ET CNC Feodo Tracker Reported CnC Server group 8
ET CNC Feodo Tracker Reported CnC Server group 9
ET CNC Feodo Tracker Reported CnC Server group 19

```
ET CNC Feodo Tracker Reported CnC Server group 22
ET CNC Feodo Tracker Reported CnC Server group 23
ET CNC Feodo Tracker Reported CnC Server group 24
ET POLICY HTTP traffic on port 443 (POST)
ET POLICY PE EXE or DLL Windows file download HTTP
ET POLICY curl User-Agent Outbound
ET HUNTING SUSPICIOUS Dotted Quad Host MZ Response
ET INFO Executable Download from dotted-quad Host
ET HUNTING GENERIC SUSPICIOUS POST to Dotted Quad with Fake Browser 1
ET MALWARE Trickbot Checkin Response
ET POLICY Observed Cloudflare DNS over HTTPS Domain (cloudflare-dns .c
ET HUNTING Suspicious POST with Common Windows Process Names - Possibl
ET MALWARE Win32/Trickbot Data Exfiltration
ET POLICY IP Check wtfismyip.com
GPL ATTACK_RESPONSE command completed
ET HUNTING Observed Suspicious SSL Cert (External IP Lookup - ident .m
ET INFO Dotted Quad Host DLL Request
ET MALWARE Cobalt Strike Malleable C2 JQuery Custom Profile M3
ET POLICY Possible External IP Lookup ipinfo.io
```

Sigma

- Abused Debug Privilege by Arbitrary Parent Processes –
https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/sysmon_abusing_debug_privilege.yml
- Accessing WinAPI in PowerShell. Code Injection –
https://github.com/SigmaHQ/sigma/blob/1ff5e226ad8bed34916c16ccc77ba281ca3203ae/rules/windows/powershell/powershell_code_injection.yml
- Bad Opsec Powershell Code Artifacts –
https://github.com/SigmaHQ/sigma/blob/5e35e387dd0dcdd564db7077da3470fbc070b975/rules/windows/powershell/powershell_bad_opsec_artifacts.yml
- CobaltStrike Service Installations –
https://github.com/SigmaHQ/sigma/blob/b26eece20d4c19b202185a6dce86aff147e92d0f/rules/windows/builtin/win_cobaltstrike_service_installs.yml
- CreateMiniDump Hacktool –
https://github.com/SigmaHQ/sigma/blob/1ff5e226ad8bed34916c16ccc77ba281ca3203ae/rules/windows/process_creation/win_hctl_createminidump.yml
- Domain Trust Discovery –
https://github.com/SigmaHQ/sigma/blob/99b0d32cec5746c8f9a79ddbbeb53391cef326ba/rules/windows/process_creation/win_trust_discovery.yml
- Dridex Process Pattern –
https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_malware_dridex.yml
- Empire PowerShell Launch Parameters –
https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_powershell_empire_launch.yml
- Execution from Suspicious Folder –
https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_execution_path.yml
- Invocation of Active Directory Diagnostic Tool (ntdsutil.exe) –
https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_ntdsutil.yml
- Local Accounts Discovery –
https://github.com/SigmaHQ/sigma/blob/ff0f1a0222b5100120ae3e43df18593f904c69c0/rules/windows/process_creation/win_local_system_owner_account_discovery.yml
- LSASS Memory Dump –

https://github.com/SigmaHQ/sigma/blob/b81839e3ce507df925d6e583e569e1ac3a3894ab/rules/windows/process_access/sysmon_lsass_memdump.yml

LSASS Memory Dump File Creation –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/file_event/sysmon_lsass_memory_dump_file_creation.yml

LSASS Memory Dumping –

https://github.com/SigmaHQ/sigma/blob/ff0f1a0222b5100120ae3e43df18593f904c69c0/rules/windows/process_creation/win_lsass_dump.yml

Malicious Base64 Encoded PowerShell Keywords in Command Lines –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_powershell_hidden_b64_cmd.yml

Malicious PowerShell Commandlets –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/powershell/powershell_malicious_commandlets.yml

Mimikatz Detection LSASS Access –

https://github.com/SigmaHQ/sigma/blob/b81839e3ce507df925d6e583e569e1ac3a3894ab/rules/windows/deprecated/sysmon_mimikatz_detection_lsass.yml

Net.exe Execution –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_net_execution.yml

Non Interactive PowerShell –

https://github.com/SigmaHQ/sigma/blob/1425ede905514b7dbf3c457561aaf2ff27274724/rules/windows/process_creation/win_non_interactive_powershell.yml

PowerShell as a Service in Registry –

https://github.com/SigmaHQ/sigma/blob/a80c29a7c2e2e500a1a532db2a2a8bd69bd4a63d/rules/windows/registry_event/sysmon_powershell_as_service.yml

PowerShell Download from URL –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_powershell_download.yml

PowerShell Execution –

https://github.com/SigmaHQ/sigma/blob/8aabb58eca06cc44ae21ae4d091793d8c5ca6a23/rules/windows/image_load/sysmon_powershell_execution_moduleload.yml

PowerShell Network Connections –

https://github.com/SigmaHQ/sigma/blob/c91eda766032b14eee60412a14875f91664e670f/rules/windows/network_connection/sysmon_powershell_network_connection.yml

PowerShell Scripts Installed as Services –

https://github.com/SigmaHQ/sigma/blob/a80c29a7c2e2e500a1a532db2a2a8bd69bd4a63d/rules/windows/builtin/win_powershell_script_installed_as_service.yml

Psexec Accepteula Condition –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_psexec_eula.yml

PsExec Tool Execution –

https://github.com/SigmaHQ/sigma/blob/ea430c8823803b9026a4e6e2ea7365dc5d96f385/rules/windows/other/win_tool_psexec.yml

Rare Service Installs –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/builtin/win_rare_service_installs.yml

Regsvr32 Anomaly –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_regsvr32_anomalies.yml

Rundll32 Internet Connection –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/network_connection/sysmon_rundll32_net_connections.yml

Suspicious AdFind Execution –

https://github.com/SigmaHQ/sigma/blob/30bee7204cc1b98a47635ed8e52f44fdf776c602/rules/windows/process_creation/win_susp_adfind.yml

Suspicious Encoded PowerShell Command Line –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_powershell_enc_cmd.yml

Suspicious In-Memory Module Execution –

https://github.com/SigmaHQ/sigma/blob/5cf7078fb3d61f2c15b01d9426f07f9197dd3db1/rules/windows/process_access/sysmon_in_memory_assembly_execution.yml

Suspicious PowerShell Parent Process –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_powershell_parent_process.yml

Suspicious Remote Thread Created –

https://github.com/SigmaHQ/sigma/blob/e7d9f1b4279a235406b61cc9c16fde9d7ab5e3ba/rules/windows/create_remote_thread/sysmon_suspicious_remote_thread.yml

Suspicious Use of Procdump –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_procdump.yml

Suspicious Use of Procdump on LSASS –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_procdump_lsass.yml

Suspicious WMI Execution –

https://github.com/SigmaHQ/sigma/blob/5e701a2bcb353338854c8ab47de616fe7e0e56ff/rules/windows/process_creation/win_susp_wmi_execution.yml

Trickbot Malware Recon Activity –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_malware_trickbot_recon_activity.yml

UNC2452 Process Creation Patterns –

https://github.com/SigmaHQ/sigma/blob/e7d9f1b4279a235406b61cc9c16fde9d7ab5e3ba/rules/windows/process_creation/win_apr_unc2452_cmds.yml

Usage of Sysinternals Tools –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/registry_event/sysmon_sysinternals_eula_accepted.yml

Whoami Execution –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/process_creation/win_susp_whoami.yml

Windows Network Enumeration –

https://github.com/SigmaHQ/sigma/blob/ff0f1a0222b5100120ae3e43df18593f904c69c0/rules/windows/process_creation/win_net_enum.yml

Windows PowerShell Web Request –

https://github.com/SigmaHQ/sigma/blob/08ca62cc8860f4660e945805d0dd615ce75258c1/rules/windows/powershell/win_powershell_web_request.yml

Yara Rules

```
/*
YARA Rule Set
Author: The DFIR Report
Date: 2021-08-02
Identifier: 4641
Reference: https://thedfirreport.com
*/

/* Rule Set -----

import "pe"
```

```
rule sig_4641_fQumH {
meta:
description = "4641 - file fQumH.exe"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-08-02"
hash1 = "3420a0f6f0f0cc06b537dc1395638be0bffa89d55d47ef716408309e65027
strings:
$s1 = "Usage: .system COMMAND" fullword ascii
$s2 = "Usage: .log FILENAME" fullword ascii
$s3 = "* If FILE begins with \"|\\" then it is a command that generates
$s4 = "AppPolicyGetProcessTerminationMethod" fullword ascii
$s5 = "Usage %s sub-command ?switches...?" fullword ascii
$s6 = "attach debugger to process %d and press any key to continue." f
$s7 = "%s:%d: expected %d columns but found %d - extras ignored" fullw
$s8 = "%s:%d: expected %d columns but found %d - filling the rest with
$s9 = "Unknown option \"%s\" on \".dump\"" fullword ascii
$s10 = "REPLACE INTO temp.sqlite_parameters(key,value)VALUES(%Q,%s);"
$s11 = "error in %s %s%s%s: %s" fullword ascii
$s12 = "UPDATE temp.sqlite_master SET sql = sqlite_rename_column(sql,
$s13 = "BBBBBBBBBBBBBBBBBBBBBB" wide /* reversed goodwill string 'BBBBBE
$s14 = "UPDATE temp.sqlite_master SET sql = sqlite_rename_column(sql,
$s15 = ");CREATE TEMP TABLE [_shell$self](op,cmd,ans);" fullword ascii
$s16 = "SqlExec" fullword ascii
$s17 = "* If neither --csv or --ascii are used, the input mode is deri
$s18 = "Where sub-commands are:" fullword ascii
$s19 = "max rootpage (%d) disagrees with header (%d)" fullword ascii
$s20 = "-- Query %d -----" fullword ascii
condition:
uint16(0) == 0x5a4d and filesize < 4000KB and
( pe.imphash() == "67f1f64a3db0d22bf48121a6cealda22" or 8 of them )
}

rule sig_4641_62 {
meta:
description = "4641 - file 62.dll"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-08-02"
hash1 = "8b9d605b826258e07e63687d1cefb078008e1a9c48c34bc131d7781b142c8
strings:
$s1 = "Common causes completion include incomplete download and damage
$s2 = "An error occurred writing to the file" fullword ascii
$s3 = "asks should be performed?" fullword ascii
$s4 = "The waiting time for the end of the launch was exceeded for an
$s5 = "Select the Start Menu folder in which you would like Setup to c
$s6 = "HcA<E3" fullword ascii /* Goodware String - occurred 1 times */
$s7 = "Select the Start Menu folder in which you would like Setup to c
$s8 = "D$(9D$@u" fullword ascii /* Goodware String - occurred 1 times *
$s9 = "Please verify that the correct path and file name are given" fu
$s10 = "Critical error" fullword ascii
$s11 = "Please read this information carefully" fullword ascii
$s12 = "Unknown error occurred for time: " fullword ascii
$s13 = "E 3y4i" fullword ascii
$s14 = "D$tOuo2" fullword ascii
$s15 = "D$PH9D$8tXH" fullword ascii
$s16 = "E$shik7" fullword ascii
$s17 = "D$p]mjk" fullword ascii
$s18 = "B):0~\"Z" fullword ascii
$s19 = "Richo/" fullword ascii
$s20 = "D$xJij" fullword ascii
condition:
```

```
uint16(0) == 0x5a4d and filesize < 70KB and
( pe.imphash() == "42205b145650671fa4469a6321ccf8bf" and pe.exports("S
}

rule sig_4641_tdrE934 {
meta:
description = "4641 - file tdrE934.exe"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-08-02"
hash1 = "48f2e2a428ec58147a4ad7cc0f06b3cf7d2587ccd47bad2ea1382a8b9c207
strings:
$s1 = "AppPolicyGetProcessTerminationMethod" fullword ascii
$s2 = "D:\\1W7w3cZ63gF\\wFIFSV\\YFU1GTi1\\i5G3cr\\Wb2f\\Cvezk3Oz\\2Zi9
$s3 = "https://sectigo.com/CPS0" fullword ascii
$s4 = "2http://crl.comodoca.com/AAACertificateServices.crl04" fullword
$s5 = "?http://crl.usertrust.com/USERTrustRSACertificationAuthority.cr
$s6 = "3http://crt.usertrust.com/USERTrustRSAAddTrustCA.crt0%" fullwor
$s7 = "ntdll.dllH" fullword ascii
$s8 = "http://ocsp.sectigo.com0" fullword ascii
$s9 = "2http://crl.sectigo.com/SectigoRSACodeSigningCA.crl0s" fullword
$s10 = "2http://crt.sectigo.com/SectigoRSACodeSigningCA.crt0#" fullwor
$s11 = "tmnEt6XElyFyz2dg5EP4TMpAvGdGtork5EZcpw3eBwJQFABWlUZa5slcF6hqfG
$s12 = "ealagi@aol.com0" fullword ascii
$s13 = "operator co_await" fullword ascii
$s14 = "ZGetModuleHandle" fullword ascii
$s15 = "api-ms-win-appmodel-runtime-l1-1-2" fullword wide
$s16 = "RtlExitUserThrea`NtFlushInstruct" fullword ascii
$s17 = "UAWAVAUATVWSH" fullword ascii
$s18 = "AWAVAUATVWUSH" fullword ascii
$s19 = "AWAVVWSH" fullword ascii
$s20 = "UAWAVATVWSH" fullword ascii
condition:
uint16(0) == 0x5a4d and filesize < 2000KB and
( pe.imphash() == "4f1ec786c25f2d49502ba19119ebfef6" or 8 of them )
}

rule sig_4641_netscan {
meta:
description = "4641 - file netscan.exe"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-08-02"
hash1 = "bb574434925e26514b0daf56b45163e4c32b5fc52a1484854b315f40fd8ff
strings:
$s1 = "netscan.exe" fullword ascii
$s2 = "TFMREMOTEPOWERSHELL" fullword wide
$s3 = "TFMREMOTEPOWERSHELLEDIT" fullword wide
$s4 = "TFMBASEDIALOGREMOTEEEDIT" fullword wide
$s5 = "*http://crl4.digicert.com/assured-cs-g1.crl0L" fullword ascii
$s6 = "*http://crl3.digicert.com/assured-cs-g1.crl00" fullword ascii
$s7 = "TFMIGNOREADDRESS" fullword wide
$s8 = "TREMOTECOMMONFORM" fullword wide
$s9 = "TFMSTOPSCANDIALOG" fullword wide
$s10 = "TFMBASEDIALOGSHUTDOWN" fullword wide
$s11 = "TFMBASEDIALOG" fullword wide
$s12 = "TFMOFFLINEDIALOG" fullword wide
$s13 = "TFMLIVEDISPLAYLOG" fullword wide
$s14 = "TFMHOSTPROPS" fullword wide
$s15 = "GGG`BBB" fullword ascii /* reversed goodwill string 'BBB`GGG'
$s16 = "SoftPerfect Network Scanner" fullword wide
$s17 = "TUSERPROMPTFORM" fullword wide
```

```
$s18 = "TFMREMOTESSH" fullword wide
$s19 = "TFMREMOTEGROUPSEdit" fullword wide
$s20 = "TFMREMOTEWMI" fullword wide
condition:
uint16(0) == 0x5a4d and filesize < 6000KB and
( pe.imphash() == "573e7039b3baff95751bded76795369e" and ( pe.exports(
}

rule sig_4641_tdr615 {
meta:
description = "4641 - file tdr615.exe"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-08-02"
hash1 = "12761d7a186ff14dc55dd4f59c4e3582423928f74d8741e7ec9f761f44f36
strings:
$s1 = "AppPolicyGetProcessTerminationMethod" fullword ascii
$s2 = "I:\\RoDcnyLYN\\k1GP\\ap0pivKfOF\\odudwtm30XMz\\UnWdqN\\01\\7aXg
$s3 = "https://sectigo.com/CPS0" fullword ascii
$s4 = "2http://crl.comodoca.com/AAACertificateServices.crl04" fullword
$s5 = "?http://crl.usertrust.com/USERTrustRSACertificationAuthority.cr
$s6 = "3http://crt.usertrust.com/USERTrustRSAAddTrustCA.crt0%" fullwor
$s7 = "http://ocsp.sectigo.com0" fullword ascii
$s8 = "2http://crl.sectigo.com/SectigoRSACodeSigningCA.crl0s" fullword
$s9 = "2http://crt.sectigo.com/SectigoRSACodeSigningCA.crt0#" fullword
$s10 = "ealagi@aol.com0" fullword ascii
$s11 = "operator co_await" fullword ascii
$s12 = "GetModuleHandleRNtUnmapViewOfSe" fullword ascii
$s13 = "+GetProcAddress" fullword ascii
$s14 = "api-ms-win-appmodel-runtime-l1-1-2" fullword wide
$s15 = "RtlExitUserThreBntFlushInstruct" fullword ascii
$s16 = "Sectigo Limitedl$0\\" fullword ascii
$s17 = "b<log10" fullword ascii
$s18 = "D*<W -" fullword ascii
$s19 = "WINDOWSPROJECT1" fullword wide
$s20 = "WindowsProject1" fullword wide
condition:
uint16(0) == 0x5a4d and filesize < 10000KB and
( pe.imphash() == "555560b7871e0ba802f2f6fbf05d9bfa" or 8 of them )
}

rule CS_DLL {
meta:
description = "62.dll"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-07-07"
hash1 = "8b9d605b826258e07e63687d1cefb078008e1a9c48c34bc131d7781b142c8
strings:
$s1 = "Common causes completion include incomplete download and damage
$s2 = "StartW" fullword ascii
$s4 = ".rdata$zzzdbg" fullword ascii
condition:
uint16(0) == 0x5a4d and filesize < 70KB and ( pe.imphash() == "42205b1
or (all of them)
}

rule tdr615_exe {
meta:
description = "Cobalt Strike on beachhead: tdr615.exe"
author = "The DFIR Report"
```

```
reference = "https://thedfirreport.com"
date = "2021-07-07"
hash1 = "12761d7a186ff14dc55dd4f59c4e3582423928f74d8741e7ec9f761f44f36
strings:
$a1 = "AppPolicyGetProcessTerminationMethod" fullword ascii
$a2 = "I:\\RoDcnyLYN\\k1GP\\ap0pivKfOF\\odudwtm30XMz\\UnWdqN\\01\\7aXg
$b1 = "ealagi@aol.com0" fullword ascii
$b2 = "operator co_await" fullword ascii
$b3 = "GetModuleHandleRNtUnmapViewOfSe" fullword ascii
$b4 = "RtlExitUserThrebNtFlushInstruct" fullword ascii
$c1 = "Jersey City1" fullword ascii
$c2 = "Mariborska cesta 971" fullword ascii
condition:
uint16(0) == 0x5a4d and filesize < 10000KB and
any of ($a* ) and 2 of ($b* ) and any of ($c* )
}
```

MITRE

- Phishing: Spearphishing Attachment – T1566.001
- Signed Binary Proxy Execution: Regsvr32 – T1218.010
- Impair Defenses: Disable or Modify Tools – T1562.001
- Domain Trust Discovery – T1482
- OS Credential Dumping: LSASS Memory – T1003.001
- System Owner/User Discovery – T1033
- Command and Scripting Interpreter: PowerShell – T1059.001
- Data Staged: Local Data Staging – T1074.001
- System Information Discovery – T1082
- Account Discovery: Local Account – T1087.001
- Account Discovery: Domain Account – T1087.002
- OS Credential Dumping: NTDS – T1003.003
- Windows Management Instrumentation – T1047
- Browser Bookmark Discovery – T1217
- Data Encrypted for Impact – T1486
- Remote Services: SMB/Windows Admin Shares – T1021.002

MITRE Software

- AdFind – S0552
- BloodHound – S0521
- Cobalt Strike – S0154
- Systeminfo – S0096
- Net – S0039
- Nltest – S0359
- Esentutl – S0404
- PsExec – S0029
- Cmd – S0106

References

TrickBot Malware Alert (AA21-076A), US CERT – <https://us-cert.cisa.gov/ncas/alerts/aa21-076a>

Advisory: Trickbot, NCSC – <https://www.ncsc.gov.uk/news/trickbot-advisory>


Trickbot Still Alive and Well, The DFIR Report – <https://thedfirreport.com/2021/01/11/trickbot-still-alive-and-well/>


Hunting for GetSystem in offensive security tools, RedCanary – <https://redcanary.com/blog/getsystem-offsec/>

TrickBot Banking Trojan, ThreatPost – <https://threatpost.com/trickbot-banking-trojan-module/167521/>


Internal case #4641


Share this:

 Twitter

 LinkedIn

 Reddit

 Facebook

 WhatsApp

Related

- 2021 Year In Review
- Stolen Images Campaign Ends in Conti Ransomware
- Trickbot Brief: Creds and Beacons

« ICEDID AND COBALT STRIKE VS ANTIVIRUS

TRICKBOT LEADS UP TO FAKE 1PASSWORD INSTALLATION »