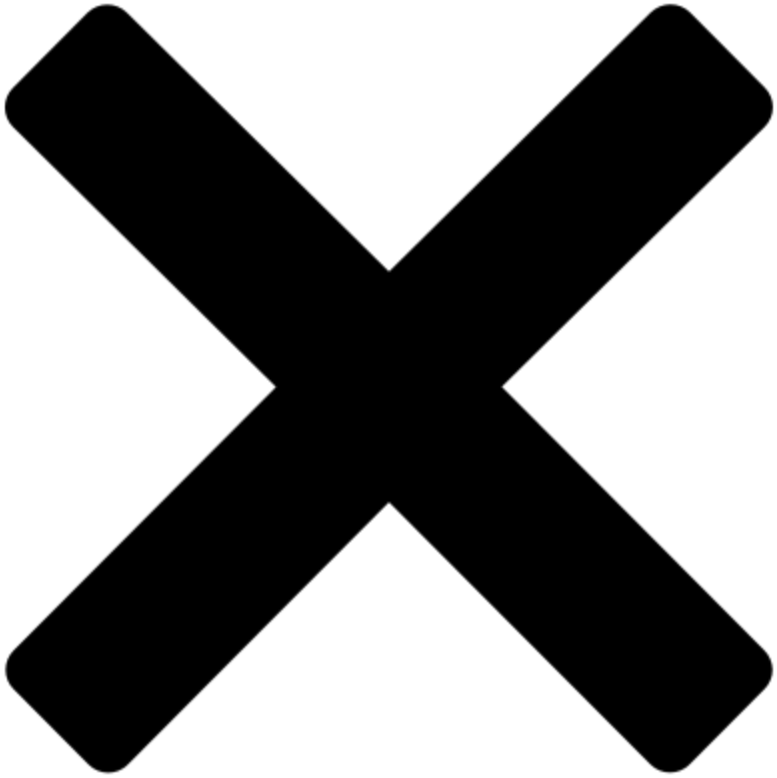
3CX Software Supply Chain Compromise Initiated by a Prior Software Supply Chain Compromise; Suspected North Korean Actor Responsible

mandiant.com/resources/blog/3cx-software-supply-chain-compromise



In March 2023, Mandiant Consulting responded to a supply chain compromise that affected 3CX Desktop App software. During this response, Mandiant identified that the initial compromise vector of 3CX's network was via malicious software downloaded from Trading Technologies website. This is the first time Mandiant has seen a software supply chain attack lead to another software supply chain attack.

Overview

3CX Desktop App is enterprise software that provides communications for its users including chat, video calls, and voice calls. In late March, 2023, a software supply chain compromise spread malware via a trojanized version of 3CX's legitimate software that was available to

download from their website. The affected software was 3CX DesktopApp 18.12.416 and earlier, which contained malicious code that ran a downloader, SUDDENICON, which in turn received additional command and control (C2) servers from encrypted icon files hosted on GitHub. The decrypted C2 server was used to download a third stage identified as ICONICSTEALER, a dataminer that steals browser information. Mandiant tracks this activity as UNC4736, a suspected North Korean nexus cluster of activity.

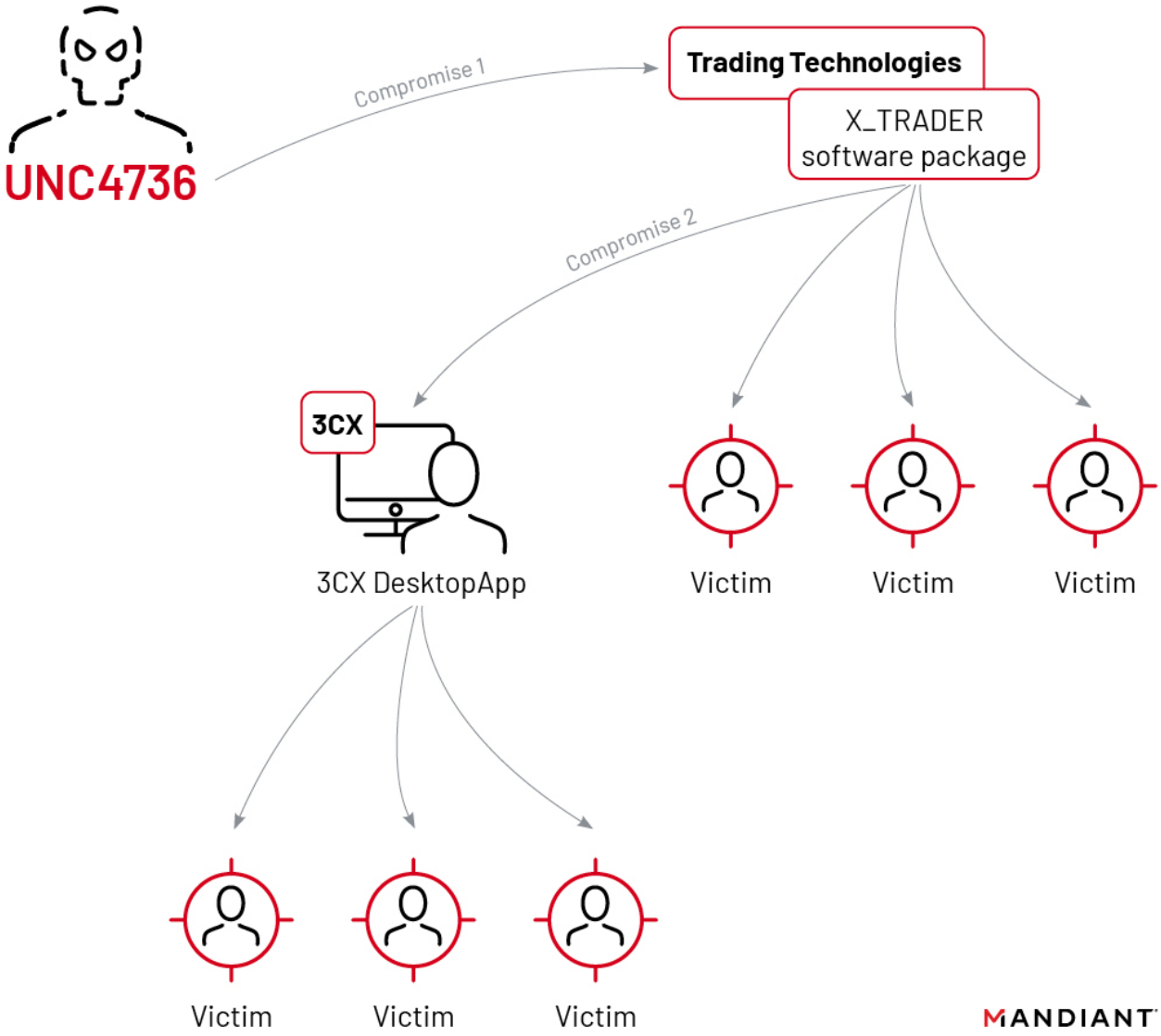


Figure 1: 3CX software supply chain compromise linked to Trading Technologies software supply chain compromise

Software Supply Chain Exploitation Explained

Mandiant Consulting's investigation of the 3CX supply chain compromise has uncovered the initial intrusion vector: a malware-laced software package distributed via an earlier software supply chain compromise that began with a tampered installer for X_TRADER, a software

package provided by Trading Technologies (Figure 1). Mandiant determined that a complex loading process led to the deployment of VEILEDSIGNAL, a multi-stage modular backdoor, and its modules.

VEILEDSIGNAL Backdoor Analysis

Mandiant Consulting identified an installer with the filename `X_TRADER_r7.17.90p608.exe` (MD5: ef4ab22e565684424b4142b1294f1f4d) which led to the deployment of a malicious modular backdoor: VEILEDSIGNAL.

Although the X_TRADER platform was reportedly discontinued in 2020, it was still available for download from the legitimate Trading Technologies website in 2022. This file was signed with the subject “Trading Technologies International, Inc” and contained the executable file Setup.exe that was also signed with the same digital certificate. The code signing certificate used to digitally sign the malicious software was set to expire in October 2022.

The installer contains and executes `Setup.exe` which drops two trojanized DLLs and a benign executable. Setup.exe uses the benign executable to side-load one of the malicious DLLs. Side-loading relies on legitimate Windows executables to load and execute a malicious file that has been disguised as a legitimate dependency. The loaded malicious DLLs contains and uses SIGFLIP and DAVESHELL to decrypt and load the payload into memory from the other dropped malicious executable. SIGFLIP relies on RC4 stream-cipher to decrypt the payload of choice and uses the byte sequence FEEDFACE to find the shellcode, in this case DAVESHELL, during the decryption stage.

SIGFLIP and DAVESHELL extract and execute a modular backdoor, VEILEDSIGNAL, and two corresponding modules. VEILEDSIGNAL relies on the two extracted modules for process injection and communications with the C2 server.

VEILEDSIGNAL and the accompanying two components provide the following functionality:

> The VEILEDSIGNAL backdoor supports three commands: send implant data, execute shellcode, and terminate itself.
>
> The process injection module injects the C2 module in the first found process instance of Chrome, Firefox, or Edge. It also monitors the named pipe and reinjects the communication module if necessary.
>
> The C2 module creates a Windows named pipe and listens for incoming communications, which it then sends to the C2 server encrypted with AES-256 in Galois Counter Mode (GCM).

The C2 configuration of the identified sample of VEILEDSIGNAL (MD5: c6441c961dcad0fe127514a918eaabd4) relied on the following hard-coded URL: `www.tradingtechnologies[.]com/trading/order-management`.

VEILEDSIGNAL Similarities and Code Comparison

The compromised X_TRADER and 3CXDesktopApp applications both contain, extract, and run a payload in the same way, although the final payload is different. Mandiant analyzed these samples and observed the following similarities:

- Usage of the same RC4 key `3jB(2bsG#@c7` in the SIGFLIP tool configuration to encrypt and decrypt the payload.
- Usage of SIGFLIP, a publicly available project on GitHub leveraging CVE-2013-3900 (MS13-098).
- Reliance on DAVESHELL, a publicly available open-source project that converts PE-COFF files to position-independent code or shellcode and that leverages reflective loading techniques to load the payload in memory.
- Use of the hardcoded cookie variable `__tutma` in the payloads.
- Both payloads encrypt data with AES-256 GCM cipher.

Compromise of the 3CX Build Environment

The attacker used a compiled version of the publicly available Fast Reverse Proxy project, to move laterally within the 3CX organization during the attack. The file `MsMpEng.exe` (MD5: 19dbffec4e359a198daf4ffca1ab9165), was dropped in C:\Windows\System32 by the threat actor.

Mandiant was able to reconstruct the attacker's steps throughout the environment as they harvested credentials and moved laterally. Eventually, the attacker was able to compromise both the Windows and macOS build environments. On the Windows build environment, the attacker deployed a TAXHAUL launcher and COLDCAT downloader that persisted by performing DLL search order hijacking through the IKEEXT service and ran with LocalSystem privileges. The macOS build server was compromised with POOLRAT backdoor using Launch Daemons as a persistence mechanism.

Previous reporting mentioned the macOS build server was compromised with SIMPLESEA. Mandiant Intelligence completed analysis of the sample and determined it to be the backdoor POOLRAT instead of a new malware family.

Threat Actor Spotlight: UNC4736

UNC4736 demonstrates varying degrees of overlap with multiple North Korean operators tracked by Mandiant Intelligence, especially with those involved in financially-motivated cybercrime operations. These clusters have demonstrated a sustained focus on cryptocurrency and fintech-related services over time.

Mandiant assesses with moderate confidence that UNC4736 is related to financially motivated North Korean “AppleJeus” activity as reported by CISA. This is further corroborated with findings from Google TAG who reported the compromise of `www.tradingtechnologies[.]com` in February 2022, preceding the distribution of compromised X_TRADER updates from the site.

- TAGreported on a cluster of North Korean activity exploiting a remote code execution vulnerability in Chrome, CVE-2022-0609, and identified it as overlapping with “AppleJeus” targeting cryptocurrency services.
- The site `www.tradingtechnologies[.]com` was compromised and hosting a hidden IFRAME to exploit visitors, just two months before the site was known to deliver a trojanized X_TRADER software package.
- Within the 3CX environment, Mandiant identified the POOLRAT backdoor using `journalide[.]org` as its configured C2 server.
- An older sample of POOLRAT (MD5: 451c23709ecd5a8461ad060f6346930c) was previouslyreported by CISA as part of the trojanized CoinGoTrade application used in the AppleJeus operation (Figure 2).

    The older sample’s infrastructure also has ties to another trojanized trading application, JMT Trading, also tracked under AppleJeus.

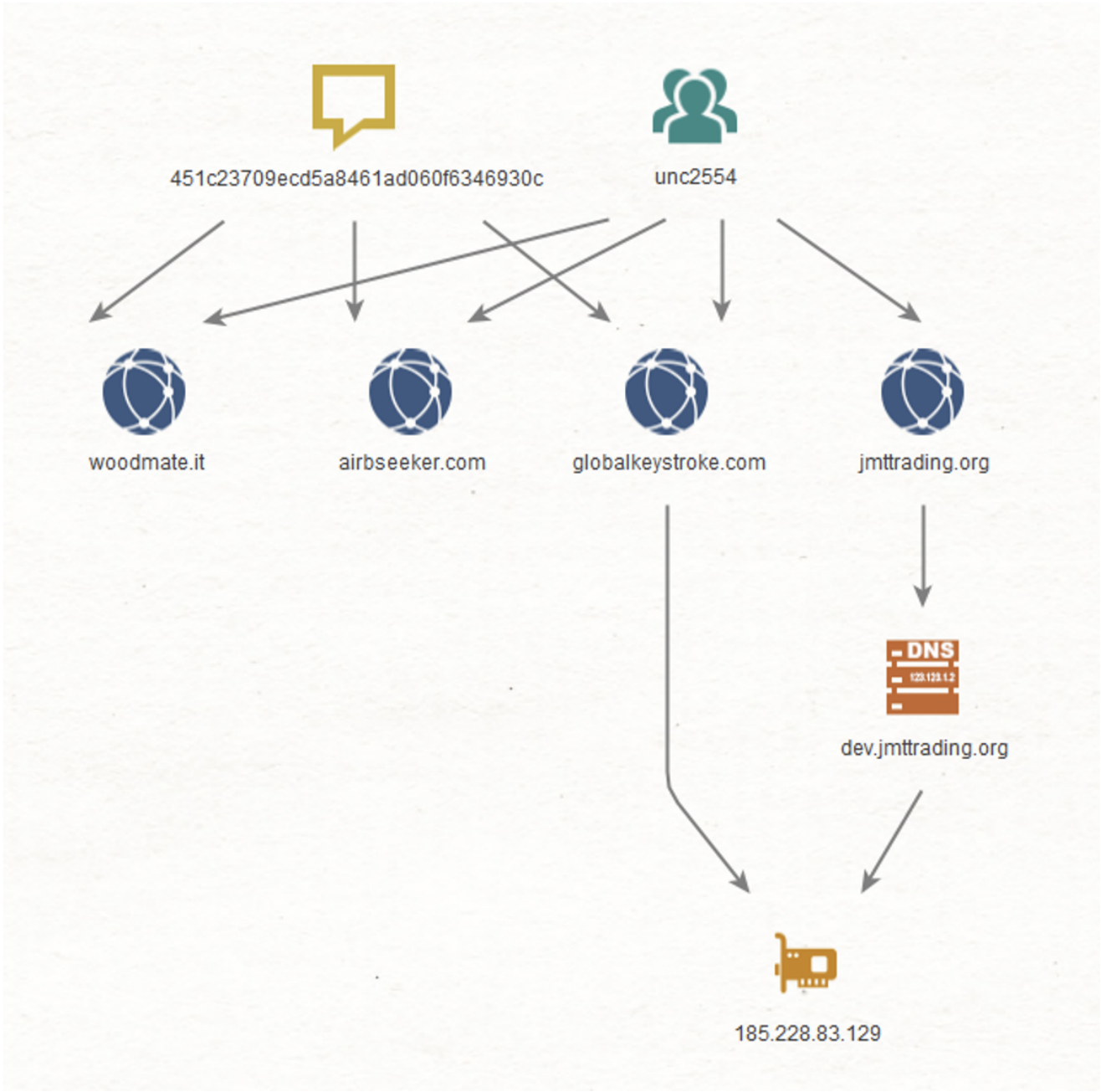


Figure 2: POOLRAT Link to CoinGoTrade and JMT Trading Activity

Weak infrastructure overlap was also identified between UNC4736 and two clusters of suspected APT43 activity, UNC3782 and UNC4469.

- DNS resolutions reveal infrastructure overlap between UNC4736 and activity linked to APT43 with moderate confidence (Tables 1 – 3)
- APT43 frequently targets cryptocurrency users and related services, highlighting such campaigns are widespread across North Korea-nexus cyber operators.

Table 1: Resolutions for IP 89.45.67.160

Date	Domain	UNC

2022-12-20	curvefinances[.]com	UNC4469
2022-12-29	pbxphonenetwork[.]com	UNC4736

Table 2: Resolutions for IP 172.93.201.88

Date	Domain	UNC
2022-04-08	journalide[.]org	UNC4736
2021-11-26	nxmnv[.]site	UNC3782

Table 3: Resolutions for IP 185.38.151[.]11

Date	Domain	UNC
2023-01-09	msedgepackageinfo[.]com	UNC4736
2023-03-22	apollo-crypto.org.shilaerc20[.]com	UNC4469

Outlook and Implications

The identified software supply chain compromise is the first we are aware of which has led to a cascading software supply chain compromise. It shows the potential reach of this type of compromise, particularly when a threat actor can chain intrusions as demonstrated in this investigation. Research on UNC4736 activity suggests that it is most likely linked to financially motivated North Korean threat actors. Cascading software supply chain compromises demonstrate that North Korean operators can exploit network access in creative ways to develop and distribute malware, and move between target networks while conducting operations aligned with North Korea's interests.

Malware Definitions

ICONICSTEALER

ICONICSTEALER is a C/C++ data miner that collects application configuration data as well as browser history.

DAVESHELL

DAVESHELL is shellcode that functions as an in-memory dropper. Its embedded payload is mapped into memory and executed.

SIGFLIP

SigFlip is a tool for patching authenticode signed PE-COFF files to inject arbitrary code without affecting or breaking the file's signature.

POOLRAT

POOLRAT is a C/C++ macOS backdoor capable of collecting basic system information and executing commands. The commands performed include running arbitrary commands, secure deleting files, reading and writing files, updating the configuration.

TAXHAUL

TAXHAUL is a DLL that, when executed, decrypts a shellcode payload expected at `C:\Windows\System32\config\TxR\<machine hardware profile GUID>.TxR.0.regtrans-ms`. Mandiant has seen TAXHAUL persist via DLL search order hijacking.

COLDCAT

COLDCAT is a complex downloader. COLDCAT generates unique host identifier information, and beacons it to a C2 that is specified in a separate file via POST request with the data in the cookie header. After a brief handshake, the malware expects base64 encoded shellcode to execute in response.

VEILEDSIGNAL

VEILEDSIGNAL is a backdoor written in C that is able to execute shellcode and terminate itself. Additionally, VEILEDSIGNAL relies on additional modules that connect via Windows named pipes to interact with the C2 infrastructure.

Acknowledgments

Special thanks to Michael Bailey, Willi Ballenthin, Michael Barnhart, and Jakub Jozwiak for their collaboration and review. Mandiant would also like to thank the Google Threat Analysis Group (TAG) and Microsoft Threat Intelligence Center (MSTIC) for their collaboration in this research.

Technical Annex: MITRE ATT&CK

Resource Development

- T1588 Obtain Capabilities

- T1588.004 Digital Certificates
- T1608 Stage Capabilities
- T1608.003 Install Digital Certificate

Initial Access

- T1190 Exploit Public-Facing Application
- T1195 Supply Chain Compromise
- T1195.002 Compromise Software Supply Chain

Persistence

- T1574 Hijack Execution Flow
- T1574.002 DLL Side-Loading

Privilege Escalation

- T1055 Process Injection
- T1574 Hijack Execution Flow
- T1574.002 DLL Side-Loading

Defense Evasion

- T1027 Obfuscated Files or Information
- T1036 Masquerading
- T1036.001 Invalid Code Signature
- T1055 Process Injection
- T1070 Indicator Removal
- T1070.001 Clear Windows Event Logs
- T1070.004 File Deletion
- T1112 Modify Registry
- T1140 Deobfuscate/Decode Files or Information
- T1497 Virtualization/Sandbox Evasion
- T1497.001 System Checks
- T1574 Hijack Execution Flow
- T1574.002 DLL Side-Loading
- T1620 Reflective Code Loading
- T1622 Debugger Evasion

Discovery

- T1012 Query Registry
- T1082 System Information Discovery
- T1083 File and Directory Discovery
- T1497 Virtualization/Sandbox Evasion

- T1497.001 System Checks
- T1614 System Location Discovery
- T1614.001 System Language Discovery
- T1622 Debugger Evasion

Command and Control

- T1071 Application Layer Protocol
- T1071.001 Web Protocols
- T1071.004 DNS
- T1105 Ingress Tool Transfer
- T1573 Encrypted Channel
- T1573.002 Asymmetric Cryptography

Impact

- T1565 Data Manipulation
- T1565.001 Stored Data Manipulation

Technical Annex: Detection Rules

YARA Rules

```
rule M_Hunting_3CXDesktopApp_Key {

  meta:

   disclaimer = "This rule is meant for hunting and is not tested to run in
a production environment"

   description = "Detects a key found in a malicious 3CXDesktopApp file"

   md5 = "74bc2d0b6680faa1a5a76b27e5479cbc"

   date = "2023/03/29"

   version = "1"

  strings:

   $key = "3jB(2bsG#@c7" wide ascii

  condition:

   $key

}
```

```
rule M_Hunting_3CXDesktopApp_Export {

  meta:

   disclaimer = "This rule is meant for hunting and is not tested to run in
a production environment"

   description = "Detects an export used in 3CXDesktopApp malware"

   md5 = "7faea2b01796b80d180399040bb69835"

   date = "2023/03/31"

   version = "1"

  strings:

   $str1 = "DllGetClassObject" wide ascii

   $str2 = "3CXDesktopApp" wide ascii

  condition:

   all of ($str*)

}



rule TAXHAUL
{
  meta:
  author = "Mandiant"
  created = "04/03/2023"
  modified = "04/03/2023"
  version = "1.0"
  strings:
   $p00_0 = {410f45fe4c8d3d[4]eb??4533f64c8d3d[4]eb??4533f64c8d3d[4]eb}
   $p00_1 = {4d3926488b01400f94c6ff90[4]41b9[4]eb??8bde4885c074}
  condition:
   uint16(0) == 0x5A4D and any of them
}
```

```
rule M_Hunting_MSI_Installer_3CX_1

{

meta:

author = "Mandiant"

md5 = "0eeb1c0133eb4d571178b2d9d14ce3e9, f3d4144860ca10ba60f7ef4d176cc736"

strings:

$ss1 = { 20 00 5F 64 33 64 63 6F 6D 70 69 6C 65 72 5F 34 37 2E 64 6C 6C 5F }

$ss2 = { 20 00 5F 33 43 58 44 65 73 6B 74 6F 70 41 70 70 2E }

$ss3 = { 20 00 5F 66 66 6D 70 65 67 2E 64 6C 6C 5F }

$ss4 = "3CX Ltd1" ascii

$sc1 = { 1B 66 11 DF 9C 9A 4D 6E CC 8E D5 0C 9B 91 78 73 }

$sc2 = "202303" ascii

condition:

(uint32(0) == 0xE011CFD0) and filesize > 90MB and filesize < 105MB and all
of them

}
```

```
rule M_Hunting_TAXHAUL_Hash_1

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

description = "Rule looks for hardcoded value used in string hashing
algorithm observed in instances of TAXHAUL."

md5 = "e424f4e52d21c3da1b08394b42bc0829"

strings:

$c_x64 = { 25 A3 87 DE [4-20] 25 A3 87 DE [4-20] 25 A3 87 DE }

condition:

filesize < 15MB and uint16(0) == 0x5a4d and uint32(uint32(0x3C)) ==
0x00004550 and any of them

}
```

```
rule M_Hunting_SigFlip_SigLoader_Native

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

description = "Rule looks for strings present in SigLoader (Native)"

md5 = "a3ccc48db9eabfed7245ad6e3a5b203f"

strings:

$s1 = "[*]: Basic Loader..." ascii wide

$s2 = "[!]: Missing PE path or Encryption Key..." ascii wide

$s3 = "[!]: Usage: %s <PE_PATH> <Encryption_Key>" ascii wide

$s4 = "[*]: Loading/Parsing PE File '%s'" ascii wide

$s5 = "[!]: Could not read file %s" ascii wide

$s6 = "[!]: '%s' is not a valid PE file" ascii wide

$s7 = "[+]: Certificate Table RVA %x" ascii wide

$s8 = "[+]: Certificate Table Size %d" ascii wide

$s9 = "[*]: Tag Found 0x%x%x%x%x" ascii wide

$s10 = "[!]: Could not locate data/shellcode" ascii wide

$s11 = "[+]: Encrypted/Decrypted Data Size %d" ascii wide

condition:

filesize < 15MB and uint16(0) == 0x5a4d and uint32(uint32(0x3C)) ==
0x00004550 and 4 of ($s*)

}
```

```
rule M_Hunting_Raw64_DAVESHELL_Bootstrap

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

description = "Rule looks for bootstrap shellcode (64 bit) present in
DAVESHELL"

md5 = "8a34adda5b981498234be921f86dfb27"

strings:

$b6ba50888f08e4f39b43ef67da27521dcfc61f1e = { E8 00 00 00 00 59 49 89 C8 48
81 C1 ?? ?? ?? ?? BA ?? ?? ?? ?? 49 81 C0 ?? ?? ?? ?? 41 B9 ?? ?? ?? ?? 56
48 89 E6 48 83 E4 F0 48 83 EC 30 C7 44 24 20 ?? ?? ?? ?? E8 ?? 00 00 00 48
89 F4 5E C3 }

$e32abbe82e1f957fb058c3770375da3bf71a8cab = { E8 00 00 00 00 59 49 89 C8 BA
?? ?? ?? ?? 49 81 C0 ?? ?? ?? ?? 41 B9 ?? ?? ?? ?? 56 48 89 E6 48 83 E4 F0
48 83 EC 30 48 89 4C 24 28 48 81 C1 ?? ?? ?? ?? C7 44 24 20 ?? ?? ?? ?? E8
?? 00 00 00 48 89 F4 5E C3 }

condition:

filesize < 15MB and any of them

}
```

```
rule M_Hunting_MSI_Installer_3CX_1

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

description = "This rule looks for hardcoded values within the MSI installer
observed in strings and signing certificate"

md5 = "0eeb1c0133eb4d571178b2d9d14ce3e9"

strings:

$ss1 = { 20 00 5F 64 33 64 63 6F 6D 70 69 6C 65 72 5F 34 37 2E 64 6C 6C 5F }

$ss2 = { 20 00 5F 33 43 58 44 65 73 6B 74 6F 70 41 70 70 2E }

$ss3 = { 20 00 5F 66 66 6D 70 65 67 2E 64 6C 6C 5F }

$ss4 = "3CX Ltd1" ascii

$sc1 = { 1B 66 11 DF 9C 9A 4D 6E CC 8E D5 0C 9B 91 78 73 }

$sc2 = "202303" ascii

condition:

(uint32(0) == 0xE011CFD0) and filesize > 90MB and filesize < 100MB and all
of them

}
```

```
rule M_Hunting_VEILEDSIGNAL_1

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

md5 = "404b09def6054a281b41d309d809a428, c6441c961dcad0fe127514a918eaabd4"

strings:

$rh1 = { 68 5D 7A D2 2C 3C 14 81 2C 3C 14 81 2C 3C 14 81 77 54 10 80 26 3C
14 81 77 54 17 80 29 3C 14 81 77 54 11 80 AB 3C 14 81 D4 4C 11 80 33 3C 14
81 D4 4C 10 80 22 3C 14 81 D4 4C 17 80 25 3C 14 81 77 54 15 80 27 3C 14 81
2C 3C 15 81 4B 3C 14 81 94 4D 1D 80 28 3C 14 81 94 4D 14 80 2D 3C 14 81 94
4D 16 80 2D 3C 14 81 }

$rh2 = { 00 E5 A0 2B 44 84 CE 78 44 84 CE 78 44 84 CE 78 1F EC CA 79 49 84
CE 78 1F EC CD 79 41 84 CE 78 1F EC CB 79 C8 84 CE 78 BC F4 CA 79 4A 84 CE
78 BC F4 CD 79 4D 84 CE 78 BC F4 CB 79 65 84 CE 78 1F EC CF 79 43 84 CE 78
44 84 CF 78 22 84 CE 78 FC F5 C7 79 42 84 CE 78 FC F5 CE 79 45 84 CE 78 FC
F5 CC 79 45 84 CE 78}

$rh3 = { DA D2 21 22 9E B3 4F 71 9E B3 4F 71 9E B3 4F 71 C5 DB 4C 70 94 B3
4F 71 C5 DB 4A 70 15 B3 4F 71 C5 DB 4B 70 8C B3 4F 71 66 C3 4B 70 8C B3 4F
71 66 C3 4C 70 8F B3 4F 71 C5 DB 49 70 9F B3 4F 71 66 C3 4A 70 B0 B3 4F 71
C5 DB 4E 70 97 B3 4F 71 9E B3 4E 71 F9 B3 4F 71 26 C2 46 70 9F B3 4F 71 26
C2 B0 71 9F B3 4F 71 9E B3 D8 71 9F B3 4F 71 26 C2 4D 70 9F B3 4F 71 }

$rh4 = { CB 8A 35 66 8F EB 5B 35 8F EB 5B 35 8F EB 5B 35 D4 83 5F 34 85 EB
5B 35 D4 83 58 34 8A EB 5B 35 D4 83 5E 34 09 EB 5B 35 77 9B 5E 34 92 EB 5B
35 77 9B 5F 34 81 EB 5B 35 77 9B 58 34 86 EB 5B 35 D4 83 5A 34 8C EB 5B 35
8F EB 5A 35 D3 EB 5B 35 37 9A 52 34 8C EB 5B 35 37 9A 58 34 8E EB 5B 35 37
9A 5B 34 8E EB 5B 35 37 9A 59 34 8E EB 5B 35 }

condition:

uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and 1 of ($rh*)

}
```

```
rule M_Hunting_VEILEDSIGNAL_2

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

md5 = "404b09def6054a281b41d309d809a428"

strings:

$sb1 = { C1 E0 05 4D 8? [2] 33 D0 45 69 C0 7D 50 BF 12 8B C2 41 FF C2 C1 E8
07 33 D0 8B C2 C1 E0 16 41 81 C0 87 D6 12 00 }

$si1 = "CryptBinaryToStringA" fullword

$si2 = "BCryptGenerateSymmetricKey" fullword

$si3 = "CreateThread" fullword

$ss1 = "ChainingModeGCM" wide

$ss2 = "__tutma" fullword

condition:

(uint16(0) == 0x5A4D) and (uint32(uint32(0x3C)) == 0x00004550) and
(uint16(uint32(0x3C)+0x18) == 0x020B) and all of them

}
```

```
rule M_Hunting_VEILEDSIGNAL_3

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

md5 = "c6441c961dcad0fe127514a918eaabd4"

strings:

$ss1 = { 61 70 70 6C 69 63 61 74 69 6F 6E 2F 6A 73 6F 6E 2C 20 74 65 78 74
2F 6A 61 76 61 73 63 72 69 70 74 2C 20 2A 2F 2A 3B 20 71 3D 30 2E 30 31 00
00 61 63 63 65 70 74 00 00 65 6E 2D 55 53 2C 65 6E 3B 71 3D 30 2E 39 00 00
61 63 63 65 70 74 2D 6C 61 6E 67 75 61 67 65 00 63 6F 6F 6B 69 65 00 00 }

$si1 = "HttpSendRequestW" fullword

$si2 = "CreateNamedPipeW" fullword

$si3 = "CreateThread" fullword

$se1 = "DllGetClassObject" fullword

condition:

(uint16(0) == 0x5A4D) and (uint32(uint32(0x3C)) == 0x00004550) and
(uint16(uint32(0x3C)+0x18) == 0x020B) and all of them

}
```

```
rule M_Hunting_VEILEDSIGNAL_4

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

md5 = "404b09def6054a281b41d309d809a428, c6441c961dcad0fe127514a918eaabd4"

strings:

$sb1 = { FF 15 FC 76 01 00 8B F0 85 C0 74 ?? 8D 50 01 [6-16] FF 15 [4] 48 8B
D8 48 85 C0 74 ?? 89 ?? 24 28 44 8B CD 4C 8B C? 48 89 44 24 20 }

$sb2 = { 33 D2 33 C9 FF 15 [4] 4C 8B CB 4C 89 74 24 28 4C 8D 05 [2] FF FF 44
89 74 24 20 33 D2 33 C9 FF 15 }

$si1 = "CreateThread" fullword

$si2 = "MultiByteToWideChar" fullword

$si3 = "LocalAlloc" fullword

$se1 = "DllGetClassObject" fullword

condition:

(uint16(0) == 0x5A4D) and (uint32(uint32(0x3C)) == 0x00004550) and
(uint16(uint32(0x3C)+0x18) == 0x020B) and all of them

}
```

```
rule M_Hunting_VEILEDSIGNAL_5

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

md5 = "6727284586ecf528240be21bb6e97f88"

strings:

$sb1 = { 48 8D 15 [4] 48 8D 4C 24 4C E8 [4] 85 C0 74 ?? 48 8D 15 [4] 48 8D
4C 24 4C E8 [4] 85 C0 74 ?? 48 8D 15 [4] 48 8D 4C 24 4C E8 [4] 85 C0 74 ??
48 8D [3] 48 8B CB FF 15 [4] EB }

$ss1 = "chrome.exe" wide fullword

$ss2 = "firefox.exe" wide fullword

$ss3 = "msedge.exe" wide fullword

$ss4 = "\\\\.\\pipe\\*" ascii fullword

$ss5 = "FindFirstFileA" ascii fullword

$ss6 = "Process32FirstW" ascii fullword

$ss7 = "RtlAdjustPrivilege" ascii fullword

$ss8 = "GetCurrentProcess" ascii fullword

$ss9 = "NtWaitForSingleObject" ascii fullword

condition:

(uint16(0) == 0x5A4D) and (uint32(uint32(0x3C)) == 0x00004550) and
(uint16(uint32(0x3C)+0x18) == 0x020B) and all of them

}
```

```
rule M_Hunting_VEILEDSIGNAL_6

{

meta:

author = "Mandiant"

disclaimer = "This rule is meant for hunting and is not tested to run in a
production environment"

md5 = "00a43d64f9b5187a1e1f922b99b09b77"

strings:

$ss1 = "C:\\Programdata\\" wide

$ss2 = "devobj.dll" wide fullword

$ss3 = "msvcr100.dll" wide fullword

$ss4 = "TpmVscMgrSvr.exe" wide fullword

$ss5 = "\\Microsoft\\Windows\\TPM" wide fullword

$ss6 = "CreateFileW" ascii fullword

condition:

(uint16(0) == 0x5A4D) and (uint32(uint32(0x3C)) == 0x00004550) and
(uint16(uint32(0x3C)+0x18) == 0x010B) and all of them

}



rule MTI_Hunting_POOLRAT {
  meta:
    author = "Mandiant"
    disclaimer = "This rule is meant for hunting and is not tested to run in
a production environment"
    description = "Detects strings found in POOLRAT. "
    md5 = "451c23709ecd5a8461ad060f6346930c"
    date = "10/28/2020"
    version = "1"
  strings:
    $str1 = "name=\"uid\"%s%s%u%s" wide ascii
    $str2 = "name=\"session\"%s%s%u%s" wide ascii
    $str3 = "name=\"action\"%s%s%s%s" wide ascii
    $str4 = "name=\"token\"%s%s%u%s" wide ascii
    $boundary = "--N9dLfqxHNUUw8qaUPqggVTpX-" wide ascii nocase
  condition:
    any of ($str*) or $boundary
}
```

```
rule M_Hunting_FASTREVERSEPROXY

{

      meta:

      author = "Mandiant"

      disclaimer = "This rule is meant for hunting and is not tested to run
in a production environment"

      md5 = "19dbffec4e359a198daf4ffca1ab9165"

      strings:

      $ss1 = "Go build ID:" fullword

      $ss2 = "Go buildinf:" fullword

      $ss3 = "net/http/httputil.(*ReverseProxy)." ascii

      $ss4 = "github.com/fatedier/frp/client" ascii

      $ss5 = "\"server_port\"" ascii

      $ss6 = "github.com/armon/go-socks5.proxy" ascii

      condition:

      uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and all of
them

}
```

Snort Rules

```
alert tcp any any -> any any (msg:"Possible malicious 3CXDesktopApp
Identified"; content:"raw.githubusercontent.com/IconStorages/images/main/";
threshold:type limit, track by_src, count 1, seconds 3600; sid: 99999999;)

alert tcp any any -> any any (msg:"Possible malicious 3CXDesktopApp
Identified";
content:"3cx_auth_id=%s\;3cx_auth_token_content=%s\;__tutma=true";
threshold:type limit, track by_src, count 1, seconds 3600; sid: 99999999;)

alert tcp any any -> any any (msg:"Possible malicious 3CXDesktopApp
Identified"; content:"__tutma"; threshold:type limit, track by_src, count 1,
seconds 3600; sid: 99999999;)

alert tcp any any -> any any (msg:"Possible malicious 3CXDesktopApp
Identified"; content:"__tutmc"; threshold:type limit, track by_src, count 1,
seconds 3600; sid: 99999999;)
```

Mandiant Security Validation

Organizations can validate their security controls using the following actions with Mandiant Security Validation.

VID	Name
A106-319	Command and Control - UNC4736, DNS Query, Variant #1
A106-321	Command and Control - UNC4736, DNS Query, Variant #2
A106-323	Command and Control - UNC4736, DNS Query, Variant #3
A106-324	Host CLI - UNC4736, 3CX Run Key, Registry Modification
A106-322	Malicious File Transfer - UNC4736, SUDDENICON, Download, Variant #1
S100-272	Evaluation: UNC4736 Conducting Supply Chain Attack Targeting 3CX Phone Management System