

MAR-10271944-3.v1 – North Korean Trojan: BUFFETLINE | CISA

Published: 2020-02-14 · Archived: 2026-04-05 14:08:46 UTC

Notification

This report is provided "as is" for informational purposes only. The Department of Homeland Security (DHS) does not provide any warranties of any kind regarding any information contained herein. The DHS does not endorse any commercial product or service referenced in this bulletin or otherwise.

This document is marked TLP:WHITE--Disclosure is not limited. Sources may use TLP:WHITE when information carries minimal or no foreseeable risk of misuse, in accordance with applicable rules and procedures for public release. Subject to standard copyright rules, TLP:WHITE information may be distributed without restriction. For more information on the Traffic Light Protocol (TLP), see <http://www.us-cert.gov/tlp>.

Summary

Description

This Malware Analysis Report (MAR) is the result of analytic efforts between Department of Homeland Security (DHS), the Federal Bureau of Investigation (FBI), and the Department of Defense (DoD). Working with U.S. Government partners, DHS, FBI, and DoD identified Trojan malware variants used by the North Korean government. This malware variant has been identified as BUFFETLINE. The U.S. Government refers to malicious cyber activity by the North Korean government as HIDDEN COBRA. For more information on HIDDEN COBRA activity, visit <https://www.us-cert.gov/hiddencobra>.

DHS, FBI, and DoD are distributing this MAR to enable network defense and reduce exposure to North Korean government malicious cyber activity.

This MAR includes malware descriptions related to HIDDEN COBRA, suggested response actions and recommended mitigation techniques. Users or administrators should flag activity associated with the malware and report the activity to the Cybersecurity and Infrastructure Security Agency (CISA) or the FBI Cyber Watch (CyWatch), and give the activity the highest priority for enhanced mitigation.

This report looks at a full-featured beaconing implant. This sample uses PolarSSL for session authentication, but then utilizes a FakeTLS scheme for network encoding using a modified RC4 algorithm. It has the capability to download, upload, delete, and execute files; enable Windows CLI access; create and terminate processes; and perform target system enumeration.

For a downloadable copy of IOCs, see [MAR-10271944-3.v1.stix](#).

Submitted Files (1)

52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695 (smss.exe)

IPs (2)

107.6.12.135

210.202.40.35

Findings

52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695

Tags

trojan

Details

Name	smss.exe
Size	139265 bytes

Type	PE32 executable (GUI) Intel 80386, for MS Windows
MD5	11cb4f1cdd9370162d67945059f70d0d
SHA1	f59c7ce763c4d5717f986e578e3bce8a43f721d2
SHA256	52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695
SHA512	53c308aa54eed5cf2979d519fc128fcebce8ce425566426086c88e9eb5ebf69c4e40361ebb5df50f98fd823b0ecf7f1a1736be189db67d56
ssdeep	3072:BqrWp5J6z3fNOo7R650dB+0l2puCERTVev7:4Wp5J6zP9di2Bt0J
Entropy	6.180760

Antivirus

Ahnlab	Trojan/Win32.Akdoor
Antiy	Trojan/Win32.Agent
Avira	TR/NukeSped.dtrpn
BitDefender	Trojan.GenericKD.5884300
ClamAV	Win.Trojan.HiddenCobra-7402602-0
ESET	a variant of Win32/NukeSped.AU trojan
Emsisoft	Trojan.GenericKD.5884300 (B)
Filseclab	Trojan.Agent.ikox.sjwn
Huorong	Trojan/Generic!6B2189F3963492CB
Ikarus	Trojan.Win32.NukeSped
K7	Trojan (004d07bc1)
McAfee	GenericRXDC-AJ!11CB4F1CDD93
NANOAV	Trojan.Win32.NukeSped.faxfdd
Symantec	Trojan.Hoplight
VirusBlokAda	BScope.Trojan.Tiggre
Zillya!	Trojan.Agent.Win32.817728

YARA Rules

- rule encodedHandshakeStrings


```
{
  meta:
    author = "CISA trusted 3rd party"
    incident = "10271944.r3.v1"
    date = "2019-12-25"
    category = "Hidden_Cobra"
    family = "BUFFETLINE"
  strings:
    $e1 = { dd 91 4a 1d cb 93 52 0a d0 cb 0a 4c ca d5 08 4b ca 92 4b 1d de 92 4b 1e d2 8b 5c 14 de 92 5c }
    $e2 = { 81 8c 4d 1d d1 8a 52 1d d7 8a 4c 0d 8b c8 01 4c cd 9c 5e 0b dc 97 5e 12 95 cb 4a 48 cf 9c 53 }
  condition:
    (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any of them
}
```
- rule polarsslClientHello


```
{
  meta:
    author = "CISA trusted 3rd party"
```

```

incident = "10271944.R3.V1"
date = "2019-12-25"
category = "Hidden_Cobra"
family = "BUFFETLINE"
strings:
  $polarSSL = "fjiejffndxklfsdkfjsaadiepwn"
  $cliHello = "!Q@W#E$R%T^Y&U*I(O)P"
condition:
  (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and all of them
}

```

ssdeep Matches

100	16c3a7f143e831dd0481d2d57aae885090e22ec55cc8282009f641755d423fcd
------------	--

PE Metadata

Compile Date	2016-10-03 02:34:09-04:00
Import Hash	6a3547c38d6806b7d5a8b2638621ca32

PE Sections

MD5	Name	Raw Size	Entropy
83eb1da0a8ab18f046922a558cb8ede6	header	4096	0.676716
b672be56b1bc345710663b196247c46c	.text	98304	6.661074
058bc0c9a6ef4120a61e2cb75b7e2825	.rdata	12288	6.220732
1b2e3c963ae327f7f74e13f15a31fa55	.data	20480	2.725473
02bb750555f1c2623effc3aa3d077a34	.rsrc	4096	0.897401

Packers/Compilers/Cryptors

Microsoft Visual C++ v6.0

Relationships

52f83cdaef...	Connected_To	107.6.12.135
52f83cdaef...	Connected_To	210.202.40.35

Description

The sample performs dynamic DLL importing and API lookups using LoadLibrary and GetProcAddress on obfuscated strings in an attempt to hide it's usage of network functions.

The sample obfuscates strings used for API lookups as well as the strings used during the network handshake using a modified RC4 algorithm. A Python 3 script to decrypt the obfuscated strings is given below. Note: The hardcoded command and control (C2) IP's are not obfuscated, but appear in plaintext within the executable.

--Begin Python 3 Decode Script--

```

def decode_string(enc, key=0x15b3):
    dec = b''
    sbox = b''

    tmp = ((key + len(enc)) * -0x52) & 0xff
    for i in range(0x100):
        sbox += bytes((((i + 1) * key * -0x78) & 0xff))

```

```

for b in enc:
dec += bytes([ord(b) ^ sbox[tmp]])
tmp = (tmp + (key + len(enc)) * 0x7c) & 0xff
return dec

```

--End Python 3 Decode Script--

--Begin C2 IP and Port--

107.6.12.135:443
210.202.40.35:443

--End C2 IP and Port--

The sample attempts to perform a PolarSSL handshake to initiate a connection to each of these hardcoded C2 IPs using TLS version 1.1. It uses the PolarSSL server_name extension with the Server Name set to "!Q@W#E\$R%^T^Y&U*(I)P". The PolarSSL certificate and private key are provided below.

--Begin PolarSSL Certificate--

-----BEGIN CERTIFICATE-----

```

MIIDhzCCAm+gAwIBAgIBADANBgkqhkiG9w0BAQUFADA7MQswCQYDVQQGEwJOTDER
MA8GA1UEChMIUG9sYXJlU0wzGTAxBgNVBAMTEFBvbGFyU1NMIFRlc3QgQ0EwHhcN
MTEwMjE0MjE0NDAwWhcNMjEwMjE0MjE0NDAwWjA7MQswCQYDVQQGEwJOTDERMA8G
A1UEChMIUG9sYXJlU0wzGTAxBgNVBAMTEFBvbGFyU1NMIFRlc3QgQ0EwggEiMA0G
CSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQA3zf8F7vglp0/ht6WmN1EpRagzSHx
mdTs6st8GFglIKXsm8WL3xoemTiZh57w1053zhdH057Zk+i5clHFzqMwUqny
50BwFMtEonILwuVA+T7lpg6z+exKY8C4KQB0nFc7qKUEkHHxvYPZP9al4jwqj+8n
YMPGn8u67GB9t+aEMr5P+1gmIgbN1LTV+/Xjli5wwOQuvfwu7uJBVCa0L0kcmnL
R7EUQIN9Z/Sg9jGr8XmksrUuEvmEF/Bibyc+E1ixVA0hmnM3oTDPb5Lc9un8rNsu
KNF+Aksj0BXYOGVkcCeoMbo4bF6BxyL0byavpw/LPh5aPgAIynplYb6LVAgMBAAGj
gZUwggZlDwAYDVR0TBAUwAwEB/zAdBgNVHQ4EFgQUitFrkpbPe0LL2udWmlQ/rPrzH
/f8wYwYDVR0jBFwwWoAUtFrkpbPe0LL2udWmlQ/rPrzH/f+hP6Q9MDSxCzAJBgNV
BAYTAk5MMREwDwYDVQQKEWhwQ2xhcmlNTTDEZMBcGA1UEAxMQUG9sYXJlU0wzGTAz
dCBDQYIBADANBgkqhkiG9w0BAQUFAAOCAQEAuP1U2ABUkIsIsCfdlc2i94QHHYeJ
SsR4EdgHtdciUI5I62J6Mom+Y0dT/7a+8S6MVMCZP6C5NyNyXw1GWY/YR82XTJ8H
DBJcT0k5DbZ6SzaONBzdWHXwWwmi5vg1dxn7YxrM9d0IjxM27WNKs4sDQhZBQkF
pjmf52cb40Pl4Y9T9meTx/lvdkRYEug61Jfn6cA+qHpyPYdTH+UshITnmp5/Ztkf
m/UTSLBNFNHesiTZeH31NcxYgDHSme9Nc/gfidRa0FLOCfWxRlFqAI47zG9jAQCZ
7Z2mCGDNMhjQc+BYcdnl0lPXjdDK6V0qCg1dVewhUBcW5gZKzV7e9+DpVA==

```

-----END CERTIFICATE-----

--End PolarSSL Certificate--

--Begin Private Key--

-----BEGIN RSA PRIVATE KEY-----

```

MIIEpAIBAAKCAQEAyHTEzLn5tXnpRdkUYLB9u5Pyax6fM60Nj4o8VmXl3ETZzGaF
B9X4J7BKNDbjngpuG7fa8H6r7gwQk4ZJGDTzqCrSV/Uu1C93KYRhtYJQj6eVSHD1
bk2y1RPD0hrt5kPqQhTrdOrA7R/UV06p86jt0uDBMHewMjDV0/Y10FZPRo7yX/k9
Z5GIMC5Cst99++UMd//sMcB4j7/Cf8qtbCHWjdmLao5v4Jv4EFbMs44TFeY0BGbH
7vk2DmqV9gmaBmf0ZXH4yqSxJeD+PIs1BGe64E92hfx//DZrtenNLQNiTrM9AM+v
dqBpVoNq0qjU51Bx5rU2BxcFbXvI5MT9TNUhXwIDAQABAoIBAGdNtfYDiap6bzt
yhCii8m9TtrhZw4MisaEaN/ll3XSjaOG2dvV6xMZCMV+5TeXDHOAZnY18Yi18vzz
4U2TnNFzizCECYNaA2fST3WgInnxUkV3YXAYp6CNxJaCmv2aA0yFr2kFVSeaKGt
ymvljNp2NVkv7Th8fBQBO7l7AXhz43k0mR7XmPgewe8ApZOG3hstK0aMvbWAvWA
zCZupdDjZyJ0JqLA4eEA4H8/w7F83r5CugeBE8LgEREjLPIyejrU5H1fubEY+h0d
l5HZBJ68ybTXfQ5U9o/QKA3dd0toBEhhdRUDGzWtjvwekEQfQf1reGWj/tod/gCpf
DFi6X0ECgYEA4wOv/pjSC3ty6TuOvKX2rOUiBrLXXv2JXzNmMoMiWI5ipLQt+RYT
VPaFl/m7Dn6MbwjayOkcZhBwk5CNz5A6Q4IJ64Mq/lqHznRCQ2Mc1G8eyDF/fYL
Ze2PlvwP9VD5jTc2miDfw+MnvJhywRRLcemDFP8k4hQVtm8PMp3ZmNECgYEA4gz7
wzObR4gn8ibe617uQPzjWzUj9dUHYd+in1gwBCrtnNna9I9U/Q6tegrYpii4ys
c176NmU+umy6XmuSKV5qD9bSpZWG2nLfnslrN15Lm3fhXoeMNHbaEDTnLT26yoi
33gp0mSSWy94ZEqipms+ULF6sY1ZtFW6tpGFoy8CgYAQHHnnvJflls2ky4q10B60

```

```
ZcxFp3rtDpkp0JxhFLhiizFrujMtZSjYNm5U7KkgPVHhLELEUvCmOnKTt4ap/vZ0
BxJNe1GZH3pW6SAvGDQpl9sG7uu/vTFP+ICxukmzxB0DrrDcvorEkKMom7ZCCRvW
KZsZ6YeH2Z81BauRj218kQKBgQCUV/DgKP2985xDTT79N08jUo3hTP5MVYCCuj/+
UeEw1TvZcx3Ljby7P6Xad6a1/BqveaGyFKIfEF1aBUBItk801sDDpDaYc4gL00Xc
7lFuBHOZkxJYlss5QrGpuOEI9ZwUt5lrFLBdYaKqNHZNVc1pCPfb/JyH6Dr2HUxq
gxUwAQKBgQcU6G2L8AG9d9c0UpOyL1tMvFe5Ttw0KjIqVdsh1MP6yigYo9DYuwu
bHFVW2r0dBTqegP2/KTOxKzaHfC1qf0RGDsUoJCNJrd1cwoCLG8P2EF4w3OBrKqv
8u4ytY0F+Vlanj5lm3TaoHSVF1+NWPpyOTiwevIECGKwSxvki4fDAA==
-----END RSA PRIVATE KEY-----
```

--End Private Key--

After the TLS authentication is completed this particular sample does NOT use the session key that is generated via TLS. Instead, it uses a FakeTLS scheme, where a 'fake' TLS packet header is prepended to the packet data which is encrypted with custom xor encryption scheme. The FakeTLS packet format and a Python 3 script to decrypt network traffic is given below.

--Begin FakeTLS Packet Structure--

```
17 03 02 <2 Byte data length> <4 Byte Key> <data>
```

--End Fake TLS Packet Structure--

Note: Each "Key" is generated by the sender rand().

--Begin Python 3 Network Communication Decode Script--

```
def decode_pkt(enc, key):
    dec = b''
    sbox = b''

    addVal = len(enc) * key & 0xff
    for i in range(0x100):
        sbox += bytes(((i + 1) * key) & 0xff)

        indexVal = addVal;
    for b in enc:
        dec += bytes([b ^ sbox[indexVal]])
        indexVal = indexVal + addVal & 0xff;
    return dec
```

--End Python 3 Network Communication Decode Script--

After the TLS authentication, the sample performs a handshake with the C2, where hardcoded 32 Byte strings are exchanged, as well as a Victim ID and the Victim Internal IP. After this exchange, the implant sends it's Victim Information (Figure 2), and then waits for tasking from the C2.

Screenshots

Figure 1 - Configuration Structure.

Figure 2 - Victim Information Structure.

Figure 3 - Implant Functionality.

Figure 4 - Session Structure.

107.6.12.135

Tags

command-and-control

Ports

- 443 TCP

Relationships

107.6.12.135	Connected_From	52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695
--------------	----------------	--

Description

Hardcoded C2 IP.

210.202.40.35

Tags

command-and-control

Ports

- 443 TCP

Relationships

210.202.40.35	Connected_From	52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695
---------------	----------------	--

Description

Hardcoded C2 IP.

Relationship Summary

52f83cdaef...	Connected_To	107.6.12.135
52f83cdaef...	Connected_To	210.202.40.35
107.6.12.135	Connected_From	52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695
210.202.40.35	Connected_From	52f83cdaefd194fff3d387631d5693a709cd7b3a20a072e7827c4d4218d57695

Mitigation

// The following Snort rule can be used to detect the FakeTLS handshake packets by targeting to a // logical inconsistency in the appdata packet sizes due to the inclusion of the 4 Byte decode key // before the data, but not being included in the data length.

```
alert tcp any any -> any any (msg:"Malware Detected"; content:"PolarSSL"; pcre:/"\x17\x03\x02\x00\x23.
{39}\x17\x03\x02"/; rev:1; sid:99999999;)
```

Recommendations

CISA recommends that users and administrators consider using the following best practices to strengthen the security posture of their organization's systems. Any configuration changes should be reviewed by system owners and administrators prior to implementation to avoid unwanted impacts.

- Maintain up-to-date antivirus signatures and engines.
- Keep operating system patches up-to-date.
- Disable File and Printer sharing services. If these services are required, use strong passwords or Active Directory authentication.
- Restrict users' ability (permissions) to install and run unwanted software applications. Do not add users to the local administrators group unless required.
- Enforce a strong password policy and implement regular password changes.
- Exercise caution when opening e-mail attachments even if the attachment is expected and the sender appears to be known.
- Enable a personal firewall on agency workstations, configured to deny unsolicited connection requests.
- Disable unnecessary services on agency workstations and servers.
- Scan for and remove suspicious e-mail attachments; ensure the scanned attachment is its "true file type" (i.e., the extension matches the file header).

- Monitor users' web browsing habits; restrict access to sites with unfavorable content.
- Exercise caution when using removable media (e.g., USB thumb drives, external drives, CDs, etc.).
- Scan all software downloaded from the Internet prior to executing.
- Maintain situational awareness of the latest threats and implement appropriate Access Control Lists (ACLs).

Additional information on malware incident prevention and handling can be found in National Institute of Standards and Technology (NIST) Special Publication 800-83, "**Guide to Malware Incident Prevention & Handling for Desktops and Laptops**".

Contact Information

Document FAQ

What is a MIFR? A Malware Initial Findings Report (MIFR) is intended to provide organizations with malware analysis in a timely manner. In most instances this report will provide initial indicators for computer and network defense. To request additional analysis, please contact CISA and provide information regarding the level of desired analysis.

What is a MAR? A Malware Analysis Report (MAR) is intended to provide organizations with more detailed malware analysis acquired via manual reverse engineering. To request additional analysis, please contact CISA and provide information regarding the level of desired analysis.

Can I edit this document? This document is not to be edited in any way by recipients. All comments or questions related to this document should be directed to the CISA at 1-844-Say-CISA or contact@mail.cisa.dhs.gov.

Can I submit malware to CISA? Malware samples can be submitted via three methods:

- Web: <https://malware.us-cert.gov>
- E-Mail: submit@malware.us-cert.gov
- FTP: <ftp://malware.us-cert.gov> (anonymous)

CISA encourages you to report any suspicious activity, including cybersecurity incidents, possible malicious code, software vulnerabilities, and phishing-related scams. Reporting forms can be found on CISA's homepage at www.us-cert.gov.

Source: <https://www.us-cert.gov/ncas/analysis-reports/ar20-045f>