TR-23 Analysis - NetWiredRC malware

circl.lu/pub/tr-23/

Overview

CIRCL analyzed a malware sample which was only sporadically detected by just a handful antivirus engines, based on heuristic detection. CIRCL analyzed the entire command structure of the malware and was able to attribute this specific malware to the malware NetWiredRC. The malware is a feature-rich Remote Access Tool, and compared to the identified predecessors, this specific version even implements more features.

Pre-Analysis

Sample A

Hashes:

Type of Hash	Hash
MD5	37e922093d8a837b250e72cc87a664cd
SHA1	c4d06a2fc80bffbc6a64f92f95ffee02f92c6bb9
SHA-256	3946d499d81e8506b8291dc0bd13475397bbcd7cb6e2c7ea504c079c92b99f62

VirusTotal results for sample A

Engine	Result
McAfee	Artemis!37E922093D8A
TrendMicro-HouseCall	TROJ_GEN.F47V0407
Comodo	TrojWare.Win32.Amtar.JEI
McAfee-GW-Edition	Artemis!37E922093D8A
ESET-NOD32	Win32/Spy.Agent.NYU
Ikarus	Backdoor:Signed.Agent

Scanned: 2014-04-07 - 49 scans - 7 detections

Engine	Result
AVG	BackDoor.Agent.AWYR

Scanned: 2014-04-07 - 49 scans - 7 detections

Signature check for sample A

Verified	Signed
Signers	Avira Operations GmbH & Co. KG
	VeriSign Class 3 Code Signing 2010 CA
	VeriSign Class 3 Public Primary Certification Authority - G5
Signing date	10:52 AM 6/25/2012
Publisher	Avira Operations GmbH & Co. KG
Description	Avira Notification Tool
Product	Avira Free Antivirus
Version	12.3.0.34
File version	12.3.0.34

Import table

- KERNEL32.dll
- USER32.dll
- GDI32.dll
- ADVAPI32.dll
- SHELL32.dll
- COMCTL32.dll
- SHLWAPI.dll
- ole32.dll
- OLEAUT32.dll
- VERSION.dll

Sections

Sections attributes in the file reveal a first hint on the maliciousness of the file: the .text section is writable and thus allows self-modifying code:

```
SECTION 1 (.text   ):
                virtual size                    : 000314DA ( 201946.)
                virtual address                 : 00001000
                section size                    : 00031600 ( 202240.)
                offset to raw data for section: 00000400
                offset to relocation            : 00000000
                offset to line numbers          : 00000000
                number of relocation entries  : 0
                number of line number entries : 0
                alignment                       : 0 byte(s)
                Flags E0000020:
                  text only
                  Executable
                  Readable
                  Writable
SECTION 2 (.rdata  ):
                virtual size                    : 0000E238 (  57912.)
                virtual address                 : 00033000
                section size                    : 0000E400 (  58368.)
                offset to raw data for section: 00031A00
                offset to relocation            : 00000000
                offset to line numbers          : 00000000
                number of relocation entries  : 0
                number of line number entries : 0
                alignment                       : 0 byte(s)
                Flags 40000040:
                  data only
                  Readable
SECTION 3 (.data   ):
                virtual size                    : 00003A5C (  14940.)
                virtual address                 : 00042000
                section size                    : 00002200 (   8704.)
                offset to raw data for section: 0003FE00
                offset to relocation            : 00000000
                offset to line numbers          : 00000000
                number of relocation entries  : 0
                number of line number entries : 0
                alignment                       : 0 byte(s)
                Flags C0000040:
                  data only
                  Readable
                  Writable
SECTION 4 (.rsrc   ):
                virtual size                    : 000064D0 (  25808.)
                virtual address                 : 00046000
                section size                    : 00006600 (  26112.)
                offset to raw data for section: 00042000
                offset to relocation            : 00000000
                offset to line numbers          : 00000000
                number of relocation entries  : 0
                number of line number entries : 0
                alignment                       : 0 byte(s)
                Flags 40000040:
                  data only
                  Readable
```

Debugging Sample A

We're not going into detail about all the obfuscation layers and extraction routines sample A is using, but briefly outline the concept. After an anti-emulation stage, stage 2 decrypts the final malware, using the key 0x5A4C4D4D4C4D, which in ASCII is ZLMMLM.

Stage 2 (xor):

```
.text:0040227A xor:
.text:0040227A                 lodsb
.text:0040227B                 xor     al, [ebx+edx]
.text:0040227E                 inc     edx
.text:0040227F                 jmp     short loc_40229B
.text:00402281 loc_402281:
.text:00402281                 stosb
.text:00402282                 mov     eax, edx
.text:00402284                 xor     edx, edx
.text:00402286                 mov     ebp, 6
.text:0040228B
.text:0040228B loc_40228B:
.text:0040228B                 div     ebp
.text:0040228D                 loop    xor
.text:0040228F                 mov     eax, ebx
.text:00402291                 add     esp, 6
.text:00402294                 pop     ebx
.text:00402295                 pop     esi
.text:00402296                 pop     edi
.text:00402297                 pop     ebp
.text:00402298                 push    eax
.text:00402299                 jmp     short loc_4022A8
.text:0040229B ; ---------------------------------------
.text:0040229B
.text:0040229B loc_40229B:
.text:0040229B                 test    edx, edx
.text:0040229D                 jnz     short loc_402281
...
.text:004022A8                 call    $+5
.text:004022AD                 pop     ebp
```

From the memory segment the code has been decrypted to, it is being written back to the .text section. Additional libraries are being loaded:

- C:\WINDOWS\system32\crypt32.dll
- C:\WINDOWS\system32\msasn1.dll
- C:\WINDOWS\system32\winmm.dll
- C:\WINDOWS\system32\ws2_32.dll
- C:\WINDOWS\system32\ws2help.dll

Finally, the instruction pointer is pointing back to the .text section at 0x00401FEC, which is the original entry point of this malware.

This binary has been isolated, extracted and named sample B:

Sample B

Hashes:

Type of Hash	Hash
MD5	759545ab2edad3149174e263d6c81dce
SHA1	2182ff6537f38a4e8c273316484c2c84872633d0
SHA-256	34d88b04956cbed54190823c94753b0dc6d8c19339d22153127293433b398cf1

VirusTotal results for sample B

VirusTotal result for hash: 759545ab2edad3149174e263d6c81dce -> Hash was not found on VirusTotal.

Signature check for sample B

File is not signed.

Analysis

Upon start, sample B, the actual malware, initializes memory, sets up Winsock by calling WSAStartup and decrypts the following strings:

String	Use
VM	Vmware check? Not used
37.252.120.122:3360	Communication channel
-	literally as “-“
Password	literally as this string
HostId-%Rand%	format string for identifier file
mJhcimNA	Name of mutex
%AppData%\Microsoft\Crypto\Office.exe	Filename when made persistent
Office	Registry key
-	literally as “-“

%AppData%\Microsoft\Crypto\Logs\	
105	?
001	?

Then it starts to communicate with the Command and Control server, waiting for commands.

The commands are listed in the following table.

All commands have return codes. In case of success, the return code corresponds to command code. If the command fails, usually the return code is the incremented command code.

Command switch:

The following table shows the commands of the malware. If there is an interesting return code, it is mentioned with (r):

Code	Command
1	(r) heartbeat (send back return code 1)
2	(r) socket created
3	(r) registered
4	(r) setting password failed
5	set password, identifier and fetch computer information (user, computername, windows version)
6	create process from local file or fetch from URL first and create process
7	create process from local file and exit (hMutex = CreateMutexA(0, 1, “mJhcimNA”))
8	(r) failed to create process
9	stop running threads, cleanup, exit
A	stop running threads, cleanup, sleep
B	stop running threads, delete autostart registry keys, cleanup, exit
C	add identifier (.Identifier) file
D	threaded: get file over HTTP and execute
E	fetch and send logical drives and types

Code	Command
10	locate and send file with time, attributes and size
12	find file
13	(r) file information
14	unset tid for 0x12
14	(r) file not found (?)
15	send file
16	write into file
17	close file (see 0x1F)
18	copy file
19	execute file
1A	move file
1B	delete file
1C	create directory
1D	file copy
1E	create directory or send file to server
1F	close file (see 0x17)
20	start remote shell
21	write into WritePipe
22	reset tid for remote shell
22	(r) terminated remote shell
23	(r) failed to start remote shell
24	collect client information and configuration
25	(r) failed to get client information and configuration
26	get logged on users
26	(r) send logged on users

Code	Command
27	(r) failed to send logged on users
28	get detailed process information
29	(r) failed to get detailed process information
2A	terminate process
2B	enumerate windows
2B	(r) send windows
2C	make window visible, invisible or show text
2D	get file over HTTP and execute
2E	(r) HTTP connect failed
2F	set keyboard event “keyup”
30	set keyboard event $event
31	set mouse button press
32	set cursor position
33	take screenshot and send
35	(r) failed to take screenshot
36	locate and send file from log directory with time, attributes and size
38	check if log file exists
39	delete logfile
3A	read key log file and send
3C	(r) failed to read key log file
3D	fetch and send stored credentials, history and certificates from common browsers
3E	fetch and send stored credentials, history and certificates from common browsers
3F	fetch and send chat (Windows Live and/or Pidgin) credentials
40	fetch and send chat (Windows Live and/or Pidgin) credentials
41	fetch and send mail (Outlook and/or Thunderbird) credentials and certificates

Code	Command
42	fetch and send mail (Outlook and/or Thunderbird) credentials and certificates
43	socks_proxy
44	get audio devices and formats
44	(r) audio devices and formats
45	(r) failed to get audio devices
46	start audio recording
47	(r) error during recording
48	stop audio recording
49	find file get md5
4C	unset tid for find file get md5 (0x49)

Network

Communication is performed via TCP/IP. First, the client registers itself at the server by sending

```
41 00 00 00 03 (...)
```

to the server, which in return replies with

```
41 00 00 00 05 (...)
```

There is a hearbeat communication going on by sending

```
01 00 00 00 02
```

to the remote site.

Outgoing communication can be detected by Network Intrusion Detection systems in order to detect compromised machines. Suricata rules are included in this report.

IOCs

- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
  - value:Office
  - data:%AppData%\Microsoft\Crypto\Office.exe

- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Active Setup\Installed Components
  - value:-
  - data:%AppData%\Microsoft\Crypto\Office.exe
- Mutex name “mJhcimNA”
- %AppData%\Microsoft\Crypto\Logs\
  logfile per day, format DD-MM-YYYY (without extension)
- %AppData%\Microsoft\Crypto\Office.exe
- %AppData%\Microsoft\Crypto\Office.exe.Identifier
- IP 37.252.120.122
- TCP port 3360

A MISP XML file is available if you want to import the indicators into MISP or any other threat indicators sharing platform.

NIDS

The following Suricata rule can be used to detect heartbeat and registration messages from a compromised client to the C&C server. The rules have only been tested mildly against live traffic and may produce a bunch of false positives. While keeping this fact in mind, you could limit the destination to the IP address and port given in this report. On the downside, you will lose the ability to track server/port changes the attacker may apply.

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any ( \
    msg:"NetWiredRC heartbeat"; \
    pkt_data; \
    content:"|01 00 00 00 02|"; \
    offset:0; \
    depth:10; \
    reference:url,https://www.circl.lu/pub/tr-23/; \
    sid:70023;\
    rev:1;)
alert tcp $HOME_NET any -> $EXTERNAL_NET any ( \
    msg:"NetWiredRC registration"; \
    pkt_data; content:"|41 00 00 00 03|"; \
    offset:0; \
    depth:10; \
    reference:url,https://www.circl.lu/pub/tr-23/; \
    sid:70123;\
    rev:1;)
```

Related samples

- Similarity by network connection (same IP:PORT), strings
  - MD5: 4af801e0de96814e9095bf78be790003
  - SHA1: b2beb80f0b1ed9b1ccbb9ae765b68d6db432a532
  - Attribution: Backdoor:Win32/NetWiredRC.B

- Similarity by network connection (same IP:PORT)
  - MD5: 1d2f110f37c43a05407e8295d75a1974
  - SHA1: d199349a3811c508ca620195327123600e1d9392
- By name NetWiredRC
  - http://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=Backdoor:Win32/NetWiredRC.B#tab=2
  - MD5: 1e279c58a4156ef2ae1ff55a4bc3aaf6
  - SHA1: 40e8e3b5fce0cd551106ccb86fc83a0ca03c9349
  - Quick analysis: previous version of this malware
    missing features: SOCKS, audio recording, find file by MD5

Decrypting NetWire C2 traffic

NetWire uses a proprietary protocol with encryption by default (AES-256-OFB). The Palto Alto Network threat intelligence team did a report on how to decrypt the traffic (as long as you know the key or you extracted it from the malware). The NetWiredDC Decoder is available on GitHub.

Recommendations

- CIRCL recommends to review the IOCs of this report and compare them with servers in the infrastructure of your organization which produce log files including proxies, A/V and system logs.
- In the case you have an infection, we recommend to capture the network traffic with the full payload as soon as possible. You might be able to decrypt the traffic later on.
- Isolate the machine infected. Acquire memory (especially to get a malware sample and a potential encryption key) and disk. Reinstall the system after the forensic acquisition.

Server intel

The server (37.252.120.122) used for this campaign is hosted at

```
inetnum:        37.252.120.0 - 37.252.120.255
netname:        TILAA
descr:          Tilaa
descr:          This space is statically assigned
country:        NL
admin-c:        TLRL-RIPE
tech-c:         TLRL-RIPE
status:         ASSIGNED PA
mnt-by:         TILAA-MNT
source:         RIPE # Filtered

role:           Tilaa admin role
address:        Februariplein 14
address:        1011MT Amsterdam
address:        The Netherlands
abuse-mailbox:  abuse@tilaa.net
admin-c:        TLDK-RIPE
admin-c:        TLGV-RIPE
admin-c:        TLRK-RIPE
tech-c:         TLDK-RIPE
tech-c:         TLGV-RIPE
tech-c:         TLRK-RIPE
nic-hdl:        TLRL-RIPE
mnt-by:         TILAA-MNT
source:         RIPE # Filtered

% Information related to '37.252.120.0/21AS196752'

route:          37.252.120.0/21
descr:          Routed by Tilaa
origin:         AS196752
mnt-by:         TILAA-MNT
source:         RIPE # Filtered
```

and reveals several open ports:

```
3360/tcp  open  unknown
3389/tcp  open  ms-wbt-server
5985/tcp  open  wsman
47001/tcp open  unknown
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49158/tcp open  unknown
49159/tcp open  unknown
49160/tcp open  unknown
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 2008 (92%)
OS CPE: cpe:/o:microsoft:windows_server_2008::sp1
OS fingerprint not ideal because: Host distance (11 network hops) is greater than
five
Aggressive OS guesses: Microsoft Windows Server 2008 SP1 (92%)
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SCAN(V=6.40%E=4%D=4/23%OT=3360%CT=1%CU=32387%PV=N%DS=11%DC=I%G=N%TM=5357A5F8%P=x86_64-
apple-darwin13.1.0)
SEQ(SP=104%GCD=1%ISR=10C%TI=I%TS=7)
OPS(O1=M5ACNW8ST11%O2=M5ACNW8ST11%O3=M5ACNW8NNT11%O4=M5ACNW8ST11%O5=M5ACNW8ST11%O6=M5A

WIN(W1=2000%W2=2000%W3=2000%W4=2000%W5=2000%W6=2000)
ECN(R=Y%DF=Y%T=80%W=2000%O=M5ACNW8NNS%CC=Y%Q=)
T1(R=Y%DF=Y%T=80%S=O%A=S+%F=AS%RD=0%Q=)
T2(R=N)
T3(R=N)
T4(R=N)
T5(R=Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=N)
T7(R=N)
U1(R=Y%DF=N%T=80%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=I%RUCK=0%RUD=G)
IE(R=N)

Uptime guess: 54.768 days (since Thu Feb 27 18:11:41 2014)
```

Ports might be used for several purposes/campaigns. Probing the ports gives the following result:

- 3360/tcp - C&C port for this campaign
- 3389/tcp - no reaction to crafted requests
- 5985/tcp - HTTP port
- 47001/tcp - HTTP port
- 49152/tcp - no reaction to crafted requests
- 49153/tcp - no reaction to crafted requests
- 49154/tcp - no reaction to crafted requests
- 49155/tcp - no reaction to crafted requests
- 49158/tcp - no reaction to crafted requests
- 49159/tcp - no reaction to crafted requests

- 49160/tcp - no reaction to crafted requests

The ports not reacting to crafted requests might be used for different campaigns for the same malware or for different versions of the malware family or even for other malware. We were not able to find a different sample of the malware that connects to a different port.

Starting of Friday 25 April, the C&C port is not active as the ISP took the appropriate action.

Classification of this document

TLP:WHITE information may be distributed without restriction, subject to copyright controls.

Acknowledgment

CIRCL thanks CERT Société Générale for sharing Sample A.

Revision

- Version 1.1 November 26, 2014 Decrypting NetWire C2 Traffic reference added
- Version 1.0 April 25, 2014 C&C (for the known TCP port) is no more active
- Version 0.9 April 23, 2014 Initial version (TLP:WHITE)