Botnets never die | APNIC Blog

Published: 2025-03-12 · Archived: 2026-04-05 20:15:31 UTC

The post was co-authored by Daji, Alex.Turing, and Acey9.

In August 2024, we at XLab observed a premeditated large-scale DDoS attack targeting the distribution platforms of the Chinese game Black Myth: Wukong — Steam and Perfect World. The attack was strategically launched during peak gaming hours across different time zones, lasting several hours each time, and impacted hundreds of servers across 13 global regions. The botnet used in the attack referred to itself as AISURU.

After AISURU was exposed, it temporarily ceased activities in September 2024 but soon resurfaced for profit-driven motives, evolving into new variants, kitty in October and AIRASHI in late November 2024. The botnet operators have repeatedly left messages in the samples to interact with us. After previously claiming to have learned to dance Macarena from XLab, they have now sent us an invitation to dance the Conga in the new variants. The following analysis will focus on the new variants, kitty and AIRASHI. Let's start 'dancing'!

The current AIRASHI botnet has the following main characteristics:

- Uses a zero-day (0day) vulnerability of cnPilot routers to spread samples.
- Sample strings are encrypted with RC4, while the CNC communication protocol has added HMAC-SHA256 verification and uses ChaCha20 encryption.
- CNC domain names include keywords such as xlabresearch, xlabsecurity, and foxthreatnointel, mocking XLab and security researchers.
- Stable T-level DDoS attack capabilities.
- Rich IP resources for the Command and Control (CNC) end, with nearly 60 IPs resolved from domains, distributed across different economies and service providers. This may be intended to accommodate more bot endpoints and increase the difficulty of dismantling the botnet. Figure 1 shows the Passive DNS records of AIRASHI CNC `xlabsecurity.ru`. It reveals that the CNC domain `xlabsecurity.ru` once resolved to 144 IPs distributed across 19 economies and 10 Autonomous System Numbers (ASNs).

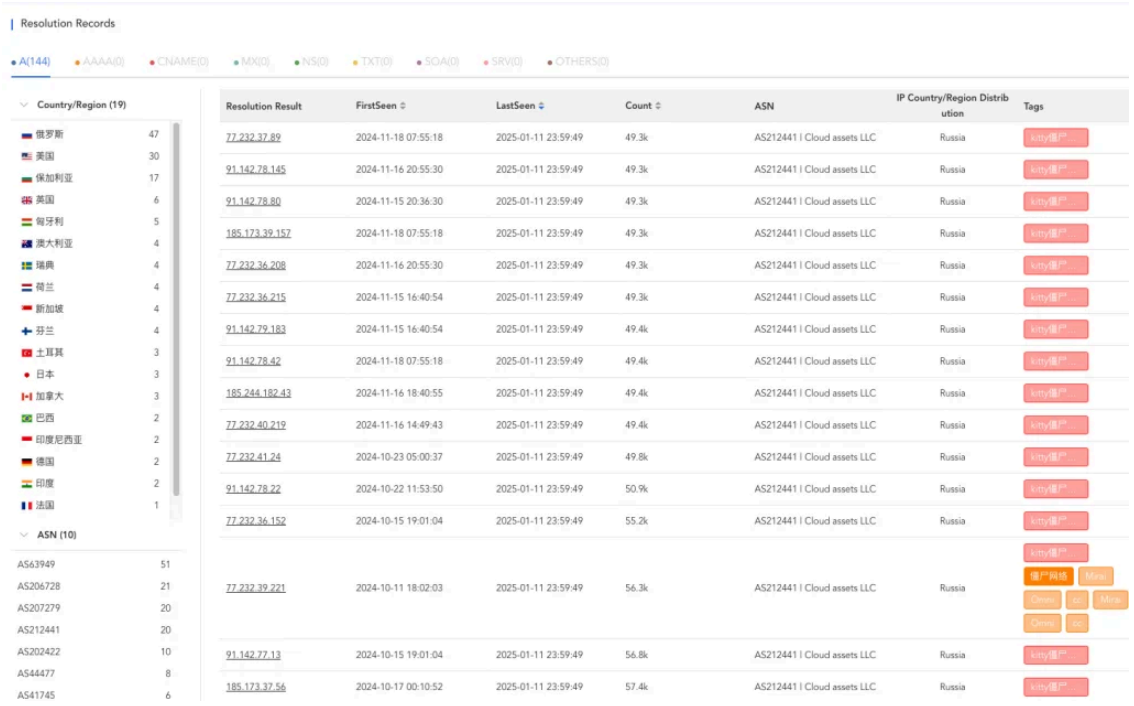

Figure 1 — `xlabsecurity.ru` Passive DNS records.

Exploitation details

Relying on the capabilities of XLab's large-scale threat awareness system, we observed that AIRASHI samples mainly spread through NDAY vulnerabilities and TELNET weak passwords, while also possessing the ability to exploit 0day vulnerabilities. Since June of last year, we have observed AIRASHI exploiting a 0day vulnerability in cnPilot routers to propagate its samples. Regarding this 0day vulnerability, we contacted the manufacturer in June of last year but received no response. To prevent its abuse, this article will withhold detailed information about the vulnerability.

Vulnerability
AMTK Camera cmd.cgi Remote Code Execution
Google Android ADB Debug Server – Remote Payload Execution
AVTECH IP Camera / NVR / DVR Devices
cve_2013_3307
cve_2016_20016
cve_2017_5259
cve_2018_14558
cve_2020_25499
cve_2020_8515

cve_2022_40005
cve_2022_44149
cve_2023_28771
Gargoyle Route run_commands.sh Remote Code Execution
LILIN Digital Video Recorder Multiple Remote Code Execution
CVE-2022-3573
cnPilot 0DAY
OptiLink ONT1GEW GPON 2.1.11_X101
Shenzhen TVT Digital Technology Co. Ltd & OEM {DVR/NVR/IPC} API RCE

Table 1 — The vulnerabilities exploited by AIRASHI.

DDoS capabilities and activity

DDoS capabilities

Botnet operators often showcase their attack capabilities through social media platforms such as Telegram, Discord, or forums, intending to attract potential customers or intimidating competitors. To prove the attack capabilities of their botnets, some operators use third-party botnet attack measurement services for validation. They direct their botnets to attack servers provided by these measurement services. The measurement services then collect and analyse information such as the size of attack traffic, packet rates, geographic locations of the attack sources, ASNs, and attack methods. After receiving these statistics, the botnet operators post them on their social media platforms to demonstrate the power of their botnets.

The AIRASHI botnet uses this exact method to prove its attack capabilities. Figure 2 shows one of their attack capability demonstrations：

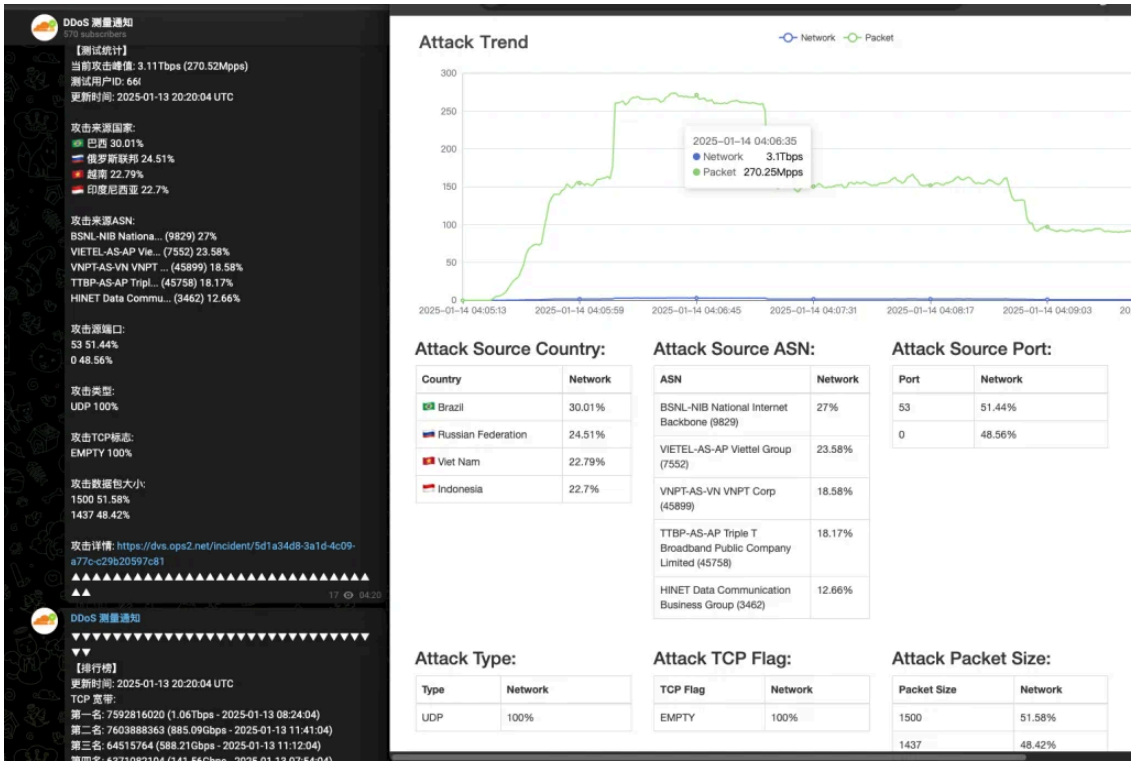

Figure 2 — AISURU attack capabilities.

The statistics displayed on the image are as follows:

- Current attack peak: 3.11Tbps (270.52Mpps).
- Test user ID: 66XXXXXXXX (This ID corresponds to the Telegram channel administrator of the AIRASHI botnet).
- Last updated: 2025-01-13 20:20:04 UTC.
- Attack source: Brazil — 30.01%, Russian Federation — 24.51%, Viet Nam — 22.79%, Indonesia — 22.7%.

The operator of AIRASHI has been posting their DDoS capability test results on Telegram. From historical data, it can be observed that the attack capacity of the AIRASHI botnet remains stable at around 1-3Tbps.

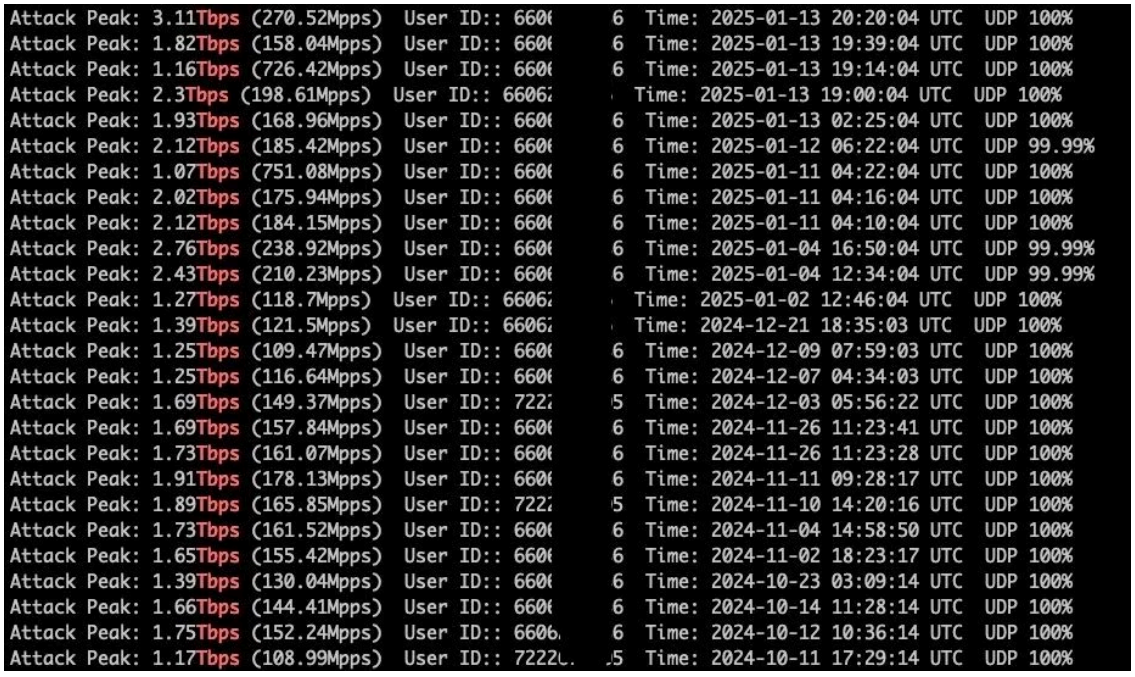

Figure 3 — AISURU attack capacity.

DDoS activities

The attack targets of the AIRASHI botnet are spread globally across various industries, with the primary targets located in regions such as China, the United States, Poland, and Russia. There is no clear, strong targeting strategy. The botnet typically attacks several hundred targets each day.

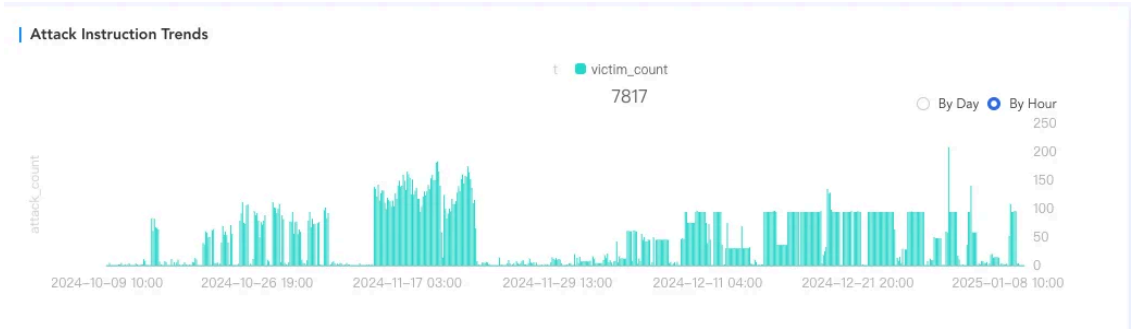

Figure 4 — Attack instruction trends.

Sample analysis

The AIRASHI botnet sample is frequently updated and has multiple versions. Some versions, in addition to supporting the main DDoS functionality and operating system command execution, also support proxy services. The following analysis focuses on kitty and AIRASHI, examining technical details of the botnet from aspects such as string decryption, C2 retrieval, communication protocols, and supported commands.

Part 1: kitty-socks5

The kitty sample began spreading in early October 2024. Unlike previous AISURU samples, it features a simplified network protocol. By the end of October, it started using SOCKS5 proxies to communicate with the C2

server, encoding 250 proxies and 55 C2 addresses in the string table.

0x1: Decryption of strings

There are no significant changes in the string decoding method; it still uses xor_bytes. However, the key has been modified to `DEADBEEFCAFEBABE1234567890ABCDEF`, and the number of entries in the string table has been reduced to 7.

```
int table_init()
{
  add_entry(1, &unk_1FE8C, 330);
  add_entry(2, &unk_1FFD8, 1500);
  add_entry(3, &unk_205B8, 8);
  add_entry(4, &unk_205C4, 12);
  add_entry(5, &unk_205D4, 3);
  add_entry(6, &unk_205D8, 5);
  return add_entry(7, &unk_205E0, 17);
}
```

Figure 5 — kitty init table.

0x2: How to get C2

In terms of C2 retrieval, the method of obtaining the C2 IP through HTTP was removed in early October 2024. The C2 string is still split using the | character, and as before, each domain is mapped to over 20 IP addresses. For example:

```
dvrhelpers.su|ipcamlover.ru|xlabresearch.ru|xlabsecurity.ru
```

However, after the addition of SOCKS5 at the end of October 2024, the string table was updated to include proxy entries. Both the C2 and proxy entries are now encoded using multiple sets of IP-PORT byte sequences. For example, `\x7f\x00\x00\x01\x00\x50` represents `127.0.0.1:80`.

0x3: Network protocol

In terms of the network protocol, it still uses a switch-case structure for handling different stages, similar to the Fodcha botnet.

```
switch ( connect_stage )
{
  case 0:
    main_make_connection();
    break;
  case 1:
    if ( !v4 )
    {
      sleep(2);
      if ( authenticate(a1) )
        v4 = 1;
    }
    break;
  case 2:
    main_check_connection();
    break;
  case 3:
    main_read_connection();
    break;
  case 4:
    main_disconnect_connection();
    v4 = 0;
    break;
  default:
    break;
}
```

Figure 6 — kitty net switch.

However, the communication process has been simplified. The latest sample uses a SOCKS5 proxy (with authentication) to access the C2 server.

```
 username: jjktkegl
password: 2bd463maabw5
```

The original key exchange process has been removed, and the communication traffic is no longer encrypted. The startup packet is replaced with `Kitty-Kitty-Kitty`, and every two minutes, a heartbeat packet `cat` is sent to the C2 server, which responds with `meow!`.

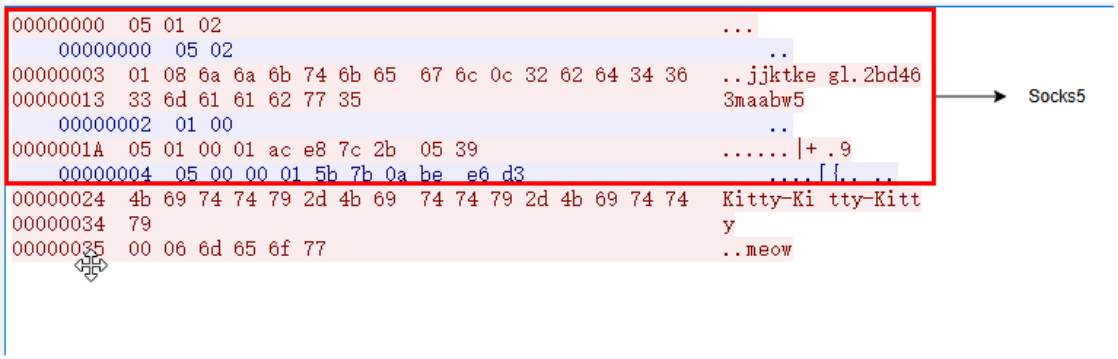

Figure 7 — kitty's communication protocol modification.

The command types still focus primarily on DDoS, with the addition of a reverse shell functionality. The command format hasn't changed significantly. It still follows the cmdtype+payload structure, but the value of Cmdtype has been updated. Additionally, DDoS-related commands now include a new AttckID field.

Cmdtype	Description
0x13	reverse shell
0x2c	stop attack
0x4b	start attack
0xaf	exit

Table 2 — Command type and description.

Part 2: AIRASHI

Currently, three types of AIRASHI samples have been discovered:

1. **AIRASHI-DDoS:** First identified in late October 2024, this sample primarily focuses on DDoS attacks but also allows arbitrary command execution and reverse shell access.
2. **Go-Proxisdk:** First discovered in late November 2024, this is a proxy tool based on muxado written in Go.
3. **AIRASHI-Proxy:** First identified in early December 2024, this is a heavily modified version of the AIRASHI-DDoS source code, using a private protocol to implement proxy functionality.

AIRASHI shares some similarities with AISURU. If kitty is a streamlined version of AISURU, then AIRASHI seems to be an upgraded version. Since October 2024, it has been continuously updated. After developing the simple Go-Proxisdk, the custom protocol proxy tool AIRASHI-Proxy was developed, indicating an attempt to surprise us with entirely new features.

0x1: RC4

AIRASHI and AISURU share some common characteristics in string decryption. Both continue to use a 16-byte key, and the decryption algorithm employed is RC4. The output string is snow slide, and special strings are

separated using the | character. The decryption method is the same for both the Proxy and DDoS versions, but the Proxy version contains significantly fewer strings.

Interestingly, some unused strings in the code seem to reference our previous blog post. One of them includes a YouTube link to a conga dance track along with an invitation to dance. There's also a message requesting XLab and foxnointel to name this variant "AIRASHI".

```
 0 'snow slide'
1 'telnetd|upnpc-static|udhcpc|/usr/bin/inetd|ntpclient|boa|lighttpd|httpd|goahead|mini_http|miniupnpd|dnsmasq|s
2 '/dvrEncoder|/dvrRecorder|/dvrDecoder|/rtspd|/ptzcontrol|/dvrUpdater'
3 'cve-2021-36260.ru'
4 'honeybooterz.cve-2021-36260.ru'
5 'stun.l.google.com:19302'
6 '/proc/'
7 '/proc/self/exe'
8 '/proc/net/tcp'
9 '/proc/mounts'
10 '/cmdline'
11 '/exe'
12 '/status'
13 '/fd/'
14 'PPid:'
15 '/bin/|/sbin/|/usr/|/snap/'
16 'wget|curl|tftp|ftpget|reboot|chmod'
17 '/bin/login'
18 '/usr/bin/cat'
19 'processor'
20 '/proc/cpuinfo'
21 '/bin/busybox echo AIRASHI > /proc/sys/kernel/hostname'
22 '/bin/busybox AIRASHI'
23 'AIRASHI: applet not found'
24 'abcdefghijklmnopqrstuvw012345678'
25 'come on, shake your body xlab, do the conga'
26 'i know you can't control yourself any longer'
27 'https://www.youtube.com/watch?v=ODKTITUPusM'
28 'dear researcher (xlab, foxnointel, ...), please refer to this malware as AIRASHI. thank you!'
```

0x2: How to get C2

AIRASHI uses three different methods to get C2:

1. **AIRASHI-DDoS (early development, late October 2024):** The most basic method, using DNS servers to resolve the C2's A record.
2. **AIRASHI-Proxy:** Retrieves the C2's TXT record from the DNS server and decodes the plaintext IP and port.

3. **AIRASHI-DDoS (late November 2024):** Uses DNS servers to retrieve the C2's TXT record, then base64-decrypts and decrypts 4 bytes of the IP using ChaCha20. The port is hardcoded in the sample.

Figure 8 — AIRASHI C2 TXT record.

- DNS_TXT_CHACHA20_KEY: `8E12DF8893A638354D851BCB46B5B7DC451C6F52066305AC641DE60C80D11850`
- DND_TXT_CHACHA20_NONCE: `941A247DDD53819F755FD59B`

It is worth noting that on 3 December 2024, both the A and TXT records for C2 resolution existed simultaneously for AIRASHI-DDoS, and there was a corresponding relationship after decryption. This may have been intended to maintain compatibility with previous versions, but it makes encryption and encoding pointless.

0x3: Network protocol

AIRASHI uses a completely new network protocol that involves HMAC-SHA256 and CHACHA20 algorithms. HMAC is used to verify the integrity of the message, while the negotiated CHACHA20_KEY is used to encrypt and decrypt the message. In the proxy version, HMAC is not used for message verification in the protocol part, but the rest of the protocol remains consistent with the DDoS version.

Communication with C2

Each message is divided into two parts — a 32-byte HMAC checksum of the message and the message itself. As shown in Figure 9, the header part of the message is sent first to confirm the message type and length. If the message length is not zero, the Payload part is then sent.

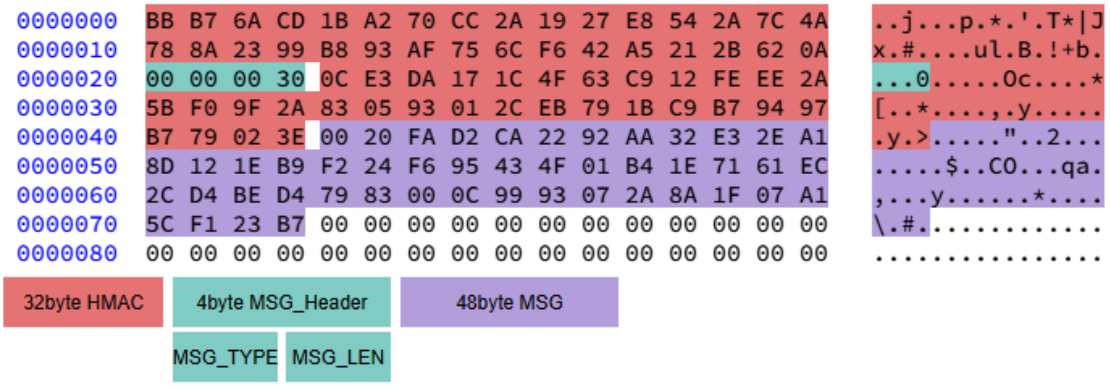

Figure 9 — AIRASHI message breakdown.

The communication process, like before, is controlled by a switch-case structure using status codes, and it is divided into four steps:

1. **Key negotiation** — Obtain a 32-byte CHACHA20_KEY and a nonce. Subsequent messages are encrypted using CHACHA20 and the CHACHA20_KEY is used as the key for HMAC-SHA256.
2. **Key confirmation** — A message with type 1 is encrypted using CHACHA20 and sent. The returned message type is verified to ensure it is also type 1.
3. **Send startup packet** — The architecture type is obtained by reading the ELF header. The structure of the startup packet is `c struct login{ uint8 uk1; uint8 uk2; uint8 uk3; uint32 stunIP; uint32 botid_len; char botid[botid_len]; uint16 cpu_core_num; uint16 arch_type; }`
4. **Check-in confirmation** — The C2 returns a message with type 2. The actual traffic generated is shown in Figure 10.

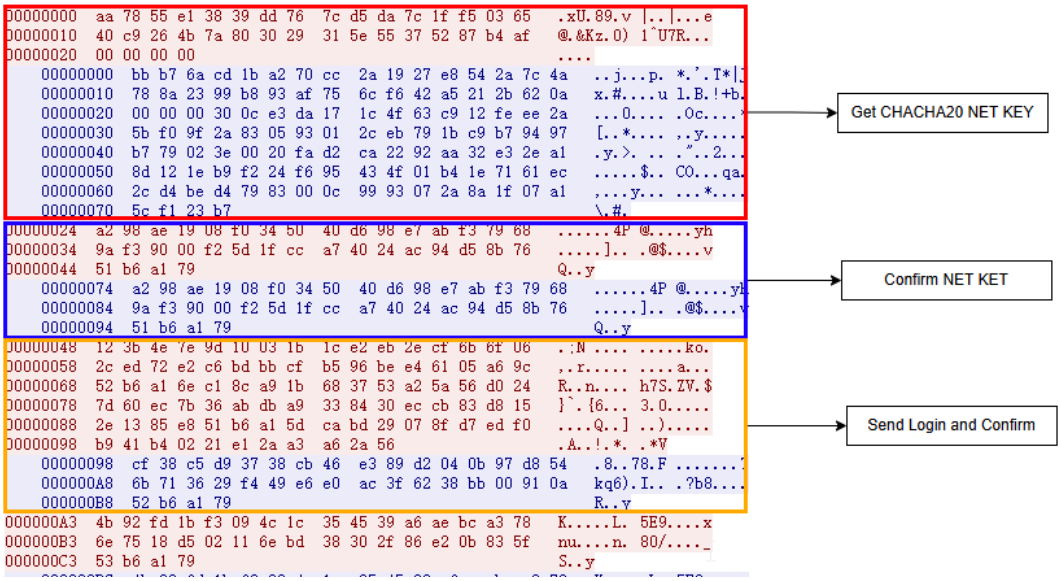

Figure 10 — AIRASHI network protocol.

Message type

AIRASHI-DDoS supports a total of 13 message types, and the corresponding handling functions are stored in an array within the bot's code. Some of the handling functions for certain message types are still incomplete, suggesting that they may still be under development.

```
:0805B140 ; int msg_handler[12]
:0805B140 msg_handler     dd 0                    ;
:0805B144                 dd 0
:0805B148                 dd 0
:0805B14C                 dd offset sub_80511F0
:0805B150                 dd offset sub_8051410
:0805B154                 dd offset sub_80513E0
:0805B158                 dd 0
:0805B15C                 dd 0
:0805B160                 dd offset sub_8051290
:0805B164                 dd offset sub_8051370
:0805B168                 dd offset sub_80512F0
:0805B16C                 dd offset sub_8051200
```

Figure 11 — AIRASHI message handler.

AIRASHI DDoS supports the following 13 message types, with some reserved for future development:

MSG_type	**Description**
0	Get Net Key
1	Confirm Net Key
2	Confirm Login
3	Heartbeat
4	Start Attack
5	Exit
6	Killer Report
7	unknown
8	unknown
9	Disable Killer
10	Enable killer
11	Exec Command
12	Reverse Shell

Table 3 — AIRASHI DDoS message types.

On the other hand, AIRASHI proxy supports only five message types, with the first four types being identical to those in AIRASHI DDoS.

MSG_type	Description
0	Get Net Key
1	Confirm Net Key
2	Confirm Login
3	Heartbeat
4	Unknown
5	Prxoy

Table 4 — AIRASHI proxy message types.

Detection

Due to the active exploitation of the cnPilot router 0day vulnerability, we are unable to provide further details. However, we are providing Snort rules to assist defenders in identifying vulnerability attempts and potential infections in their environment.

```
alert tcp any any -> any any (msg:"0DAY exploit #1 attempt"; content:"execute_script"; content:"sys_list"; cont
```

Readers are always welcome to reach us on X.

Wang Hao is a security researcher at XLab.

Adapted from the original post on XLab.

Indicators of Compromise (IOCs)

C2

```
 xlabresearch.ru
xlabsecurity.ru
foxthreatnointel.africa
```

SHA1

```
 3c33aa8d1b962ec6a107897d80d34a5d0b99899e
0339415f8f3e2b1eb6b24ed08c3a311210893a6e
95c8073cc4d8b80ceddb8384977ddc7bbcb30d8c
12fda6d480166d8e98294745de1cfdcf52dbfa41
08b30f5ffa490e15fb3735d69545c67392ea24e9
c8b8bd5384eff0fe3a3a0af82c378f620b7dc625
```

Downloader

```
 190.123.46.21    Panama|Panama|Panama       AS52284|Panamaserver.com
190.123.46.55     Panama|Panama|Panama       AS52284|Panamaserver.com
95.214.52.167     Poland|Mazowieckie|Warsaw  AS201814|MEVSPACE sp. z o.o.
162.220.163.14    United States|New Jersey|Secaucus   AS19318|Interserver, Inc
```
