Hive Pro

Threat Level

Amber

Hiveforce Labs

THREAT ADVISORY

ATTACK REPORT

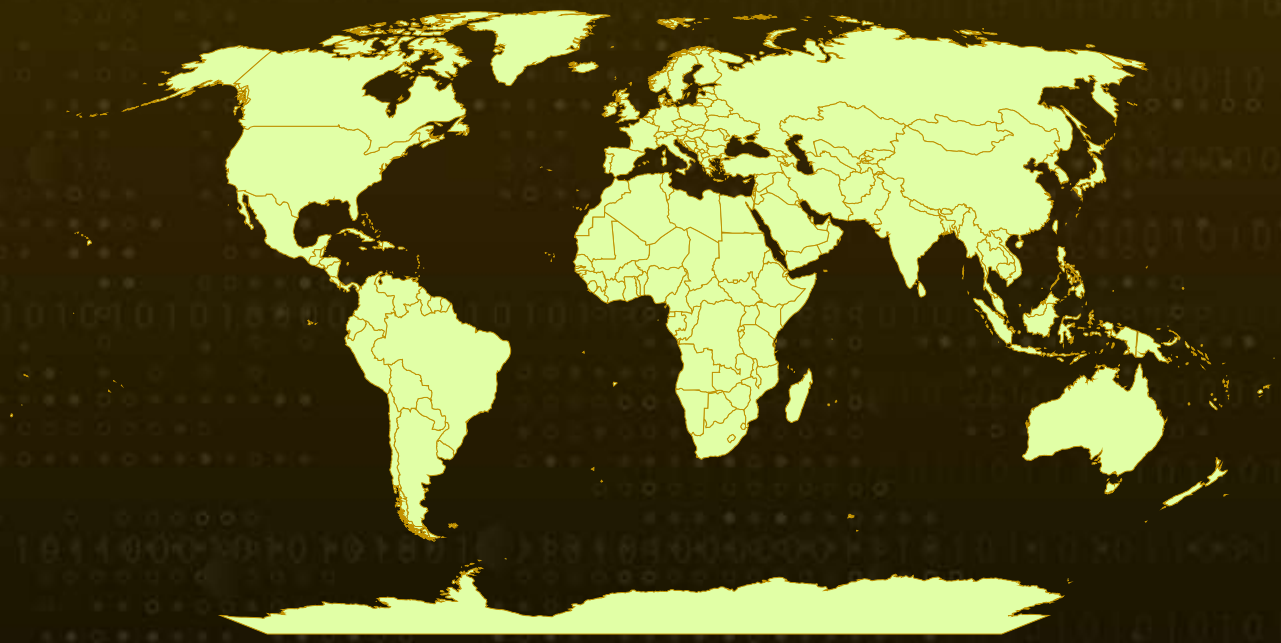
Tengu The Mirai Fork With a Resurrection Clause

Date of Publication	Admiralty Code	TA Number
July 31, 2026	A1	TA2026217

Summary

First Seen: July 2026
Targeted Regions: Worldwide
Targeted Platforms: Linux (embedded and IoT), Android; samples compiled for i386, amd64, MIPS, ARM, PowerPC, and m68k
Malware: Tengu Botnet
Attack: Tengu is a Linux botnet malware family built on Mirai's foundations but far more sophisticated in how it hides and defends itself. Infection in the observed case came from brute-forcing an exposed remote-management service with weak or default credentials, not from any software vulnerability, making poorly maintained IoT and embedded Linux devices the obvious target.

Attack Regions



Targeted Non-Targeted

Attack Details

#1

Tengu is a Linux botnet malware family that shares roots with Mirai but goes considerably further in stealth and self-defence. In the observed case, infection began with a brute-force attack on an exposed remote-management service using weak or default credentials no vulnerability, vendor, or device model was involved. A short shell script then pulled down builds matching the target's processor architecture, with versions available for common desktop, embedded, and legacy platforms. This makes exposed and poorly maintained IoT and embedded Linux devices the natural target.

#2

Once running, the malware decodes its command-and-control address in memory and immediately works to hide itself. It copies its own executable into an anonymous memory-backed file, relaunches from there, and disguises the process name so that monitoring tools display it as a legitimate system logging daemon. It then hardens itself by ignoring common termination signals, disabling core dumps to frustrate inspection, and instructing the kernel not to select it for termination under memory pressure. Anti-analysis checks run continuously: it exits if a debugger is attached or if library-injection variables are present, uses instruction-timing measurements to detect emulation or single-stepping, and periodically hashes its own executable code in memory to detect tampering. All sensitive strings are stored encrypted and decoded only at runtime.

#3

Persistence is where Tengu diverges most from ordinary Mirai variants. A detached guardian process checks on the main process every minute and relaunches it if it disappears. On instruction from the server, the malware can install itself as a fake system service, hook into legacy startup mechanisms, infect shell login scripts, and mark its own binary immutable so it cannot easily be deleted. Its most aggressive layer abuses the hardware watchdog timer: a background worker disguised as a kernel thread arms the watchdog with a short timeout and sends keepalive signals only while the malware is alive.

#4

If a defender kills the malware, the keepalives stop, and the device reboots, converting a hardware safety feature into an anti-remediation trap that gives the other persistence layers another chance to run. It compounds this by corrupting the reboot and shutdown utilities an administrator would normally use to restart or safely power down the device. Separately, it hunts competing malware, scanning running processes against known rival patterns and killing any matches along with their child processes, and it watches for new process creation so it can react within a fraction of a second rather than waiting for its next scan.

#5 Finally, the intelligence describes capability rather than reach. No operator, campaign, infection count, targeted sector, region, or confirmed victim is identified, no Android infections were documented despite the delivery path, and there is no evidence that the server ever issued commands or was even reachable. Independent telemetry confirms only that malicious Mirai-related files were hosted at the same address for several weeks before going offline.

Recommendations

Detect Fileless Execution Paths on Embedded Hosts: Alert on memfd_create usage producing anonymous executable mappings and on unlinked files under /dev/shm, particularly names resembling system daemons such as .journal. These paths are how the malware runs without a durable on-disk artifact to scan.

Plan Remediation Around a Forced Reboot: Assume that killing the malware process will cause the device to reboot within roughly 30 seconds via the hardware watchdog. Sequence responses so that persistence removal, forensic collection, and network containment are completed before process termination, or isolate the device at the network layer first and treat the reboot as expected rather than as evidence of a second intrusion.

Verify Reboot and Shutdown Utilities Before Relying on Them: Check the integrity of reboot- and shutdown-related binaries on suspected hosts, looking for ELF headers overwritten with the string ELFOOD. Recovery procedures that depend on those utilities may fail silently on an infected device, so prepare out-of-band power control or firmware reflash as the fallback.

Audit Every Persistence Location Before Returning a Device to Service: Review systemd unit files under /etc/systemd/system, init.d and rc .d scripts, procd and rc.local, shell startup files including .bashrc, .profile and /etc/profile, and cron-related paths. Also check for binaries marked immutable with chattr +i, which will resist deletion until the attribute is cleared. Tengu installs several of these independently, so removing one leaves the others live.

Treat Reinfection After Cleanup as Expected and Reimage Where Possible: Where firmware reflash or factory reset is feasible on a confirmed-infected device, prefer it over selective cleanup. The combination of a 60-second guardian process, multiple boot-time persistence mechanisms and watchdog-forced reboots makes partial remediation likely to fail.

Potential MITRE ATT&CK TTPs

Tactic	Technique	Sub-technique
Initial Access	T1133: External Remote Services	
Execution	T1059: Command and Scripting Interpreter	T1059.004: Unix Shell
	T1106: Native API	
	T1620: Reflective Code Loading	
Persistence	T1543: Create or Modify System Process	T1543.002: Systemd Service
	T1037: Boot or Logon Initialization Scripts	T1037.004: RC Scripts
	T1053: Scheduled Task/Job	T1053.003: Cron
	T1546: Event Triggered Execution	T1546.004: Unix Shell Configuration Modification
Credential Access	T1110: Brute Force	T1110.001: Password Guessing
Defense Evasion	T1222: File and Directory Permissions Modification	T1222.002: Linux and Mac File and Directory Permissions Modification
	T1036: Masquerading	T1036.004: Masquerade Task or Service
		T1036.005: Match Legitimate Name or Location
	T1497: Virtualization/Sandbox Evasion	
	T1622: Debugger Evasion	

Tactic	Technique	Sub-technique
Defense Evasion	T1027: Obfuscated Files or Information	
	T1562: Impair Defenses	T1562.001: Disable or Modify Tools
	T1070: Indicator Removal	T1070.004: File Deletion
	T1553: Subvert Trust Controls	
Discovery	T1082: System Information Discovery	
	T1016: System Network Configuration Discovery	
	T1057: Process Discovery	
Command and Control	T1573: Encrypted Channel	
	T1568: Dynamic Resolution	T1568.002: Domain Generation Algorithms
	T1090: Proxy	
	T1105: Ingress Tool Transfer	
Exfiltration	T1041: Exfiltration Over C2 Channel	
Impact	T1498: Network Denial of Service	T1498.001: Direct Network Flood
	T1529: System Shutdown/Reboot	
	T1485: Data Destruction	
	T1489: Service Stop	

Indicators of Compromise (IOCs)

TYPE	VALUE
IPv4	64[.]89[.]163[.]8
IPv4:Port	64[.]89[.]163[.]8[:]9931, 64[.]89[.]163[.]8[:]8080
SHA256	897226af37990fa60f25fea00b0509faa0e78d8bee10875c23b9b6ab0b8faed9
SHA1	caef4358921486cea54333bdcde3b61c845deb9c, 097522a52986982b9eefc29f95efdd9d3b6032e7, b6d406992411dad0ef80f2b7b61427448bd05540, 144521ad3baf1fc28ceb4ad26d6fe490ab1b31f4, 0243692ce6ca522bc1359a3d89d70a229cf76587, 01c3326ce5beb78a6c106960a3a0868682b97bfa, 197652174622cda52435249bc96a2d82b3613194, da040700e8c0852eaa848fb01138e5b820c6c765, b249800066433c0a8bf23edae719133465247abc, 0750489645d263efde56189be97c045cfc8ca890
File Path	/tmp/.proxy.pid, /tmp/.up, /dev/shm/.journal, /etc/init.d/tengu, /dev/watchdog0
Filename	systemd-journal
Encryption Key	c2a1b84d9f73512e8a6c15f0b4d9e73b114a66895dc328927e34a5b148ce6103

References

https://www.nozominetworks.com/blog/tengu-a-modernized-mirai-that-doesnt-want-to-leave

What Next?

At **Hive Pro**, it is our mission to detect the most likely threats to your organization and to help you prevent them from happening.

Book a Demo of **HivePro**.

REPORT GENERATED ON

July 31, 2026 • 3:00 AM

More at www.hivepro.com