

public-research/drifter/report.md at main · deepfield/public-research

By jpgmeyer

Archived: 2026-09-26 02:02:18 UTC

Drifter: C2 traffic dressed as camera management

Nokia Deepfield Emergency Response Team (ERT)

First published: 2026-03-28 | Last updated: 2026-04-26

Summary

Drifter is a previously undocumented DDoS botnet targeting Android TV devices via ADB. It competes for ADB-exposed devices with [MossadProxy](#), [Jackskid](#), and [Kimwolf](#), families we [documented previously](#) as part of a single ecosystem. Drifter shares no code, no infrastructure, and no cryptographic material with any of them, suggesting an independent operator on the same contested attack surface.

What distinguishes Drifter is how it hides. Its C2 domains are named after IP camera brands (`hikvision-cctv[.]su` , `nvms9000[.]su`), chosen to blend with the traffic of devices that share a VLAN with the Android TV boxes it infects. We have not seen this technique in any other family we track, though the bar for novelty on this attack surface is not high.

The binary is a 71 KB statically linked ARM ELF with 8 DDoS attack methods, a domain generation algorithm, a Telegram dead-drop resolver, and a custom stream cipher that resisted static analysis. For a new entrant still missing a kill list, the scale is already notable: network observations have attributed attacks up to 2.6 Tbps from approximately 80,000 sources. Within seconds of connecting to its C2, our analysis infrastructure received an attack command — a UDP flood against a Valve Source Engine game server — which was logged but not executed.

Sample

Field	Value
APK hash	d22d9a91...
Native binary hash	577a330a...
Binary size	71 KB
Package	io.nexus.drifter

Field	Value
Installed as	com.siliconworks.android.update
Architecture	ARM EABI4, statically linked, stripped
Min SDK	21 (Android 5.0)

Full hashes are in [iocs/hashes.csv](#) .

Delivery

The dropper installs the APK as `com.siliconworks.android.update` , grants runtime permissions, whitelists it from battery optimization, and launches the main activity. The disguise package name follows the time-honored social engineering principle that anything called "update" inherits the trust of whatever it claims to be updating.

The APK is a thin Java wrapper. `SdkNotifyService` extracts the native binary (`libcyn.so`) from assets, launches it via `ProcessBuilder` , and restarts it every 60 seconds if it exits. A `BootReceiver` (priority 999) relaunches the service on boot. The activity hides itself via `excludeFromRecents` and `noHistory` .

Drifter targets the same device population as MossadProxy v2.5.2, confirming both compete for the same pool of ADB-exposed Android TV devices. This delivery surface (residential proxy services exposing ADB on uncertified AOSP devices) was [first documented by Synthient](#) in the context of Kimwolf and has since attracted at least six independent operators that we track. The economics have not changed: a residential proxy subscription buys access to millions of devices with unauthenticated root shell access.

C2 infrastructure

The camera next door

The bot uses three C2 base domains across two TLD families:

Domain	TLD	Masquerade
daylightbomb[.]elite	Handshake (HNS)	—
hikvision-cctv[.]su	.su	Hikvision IP camera
nvms9000[.]su	.su	Hikvision NVMS9000 NVR

Resolution runs through a custom DNS resolver at `194.50.5[.]27` (located in Australia), bypassing the system resolver entirely. The geographic choice somewhat undermines the "blend with local camera traffic" strategy: DNS queries to an Australian resolver from a hotel lobby in São Paulo are not, strictly speaking, inconspicuous. The `.elite` TLD is a Handshake (HNS) name that standard resolvers (1.1.1.1, 8.8.8.8) return NXDOMAIN for, meaning these domains are absent from most passive DNS databases.

The CCTV brand names are not decorative. Android TV boxes (the cheap, unbranded AOSP devices this botnet targets) are frequently deployed on the same network segments as IP cameras. In a hotel, the lobby TV box and the hallway camera share a VLAN. In a small business, the break room streaming box sits next to the NVR. DNS queries from those subnets to `hikvision-cctv[.]su` or `nvms9000[.]su` blend with legitimate surveillance management traffic. The operator read the room. Or at least the subnet. The connectivity check domain, `connectivity.accesscam[.]org` (resolves to `1.3.3.7`, which tells you everything and nothing), completes the theme.

`daylightbomb[.]elite` suggests the operator's commitment to the CCTV cover story has limits.

DNS obfuscation

DNS A records for the C2 domains contain obfuscated IP addresses. Each 32-bit IP has its two 16-bit halves swapped before storage:

DNS A record	Decoded C2 IP
<code>253.229.172.232</code>	<code>172.232.253.229</code>
<code>35.106.172.232</code>	<code>172.232.35.106</code>
<code>74.253.172.105</code>	<code>172.105.74.253</code>

An analyst investigating these DNS responses will see IP addresses that match no known threat intelligence. It won't survive peer review, but it doesn't need to. It needs to survive the automated enrichment pipeline, and at time of writing, it does. The decoded IPs map to Linode/Akamai, DigitalOcean, GHOSTnet, and Onidel infrastructure (see [C2 IP pool](#)). Defenders should apply 16-bit half-swap to any A records returned for these domains before enrichment.

Domain generation algorithm

The bot prepends random 4–18 character base36 subdomains to the base domains:

```
<random>.daylightbomb.elite
<random>.hikvision-cctv.su
<random>.nvms9000.su
```

The DNS configuration differs by domain. The two `.su` domains use wildcard records: any subdomain returns the C2 IP pool. `hikvision-cctv[.]su` only resolves as a wildcard (the bare domain has no A record), while `nvms9000[.]su` resolves both bare and as a wildcard. `daylightbomb[.]elite` resolves only as a bare domain with no wildcard. For detection purposes, **any subdomain of the three base domains is an indicator**, including subdomains not yet observed. This makes subdomain-level indicators disposable; the bot generates a fresh one for every connection attempt.

Telegram dead-drop resolver

When primary DNS fails, the bot falls back to the Telegram account `t.me/disconnect` as a dead-drop resolver. At the time of writing, the account bio contains no obvious C2 address; the mechanism may not yet be active. The username suggests the operator processes things through naming.

NAT traversal

The bot contacts a public STUN server (`stun.sip.us:3478`) to discover its external IP, a practical necessity for bots behind CGNAT on residential networks.

C2 ports

The bot randomly selects from 8 non-standard high ports per connection:

```
23004, 25632, 32505, 34532, 36275, 36605, 37610, 44308
```

None are assigned by IANA, and none are interesting enough to appear in an `nmap` default scan. The operator generates port numbers the same way they generate subdomains: with a preference for the unmemorable.

C2 IP pool

At the time of writing, the C2 IP pool contains 25 unique IPs across four providers:

Provider	ASN	Count
Linode/Akamai	AS63949	16
GHOSTnet	AS12586	4
DigitalOcean	AS14061	4
Onidel	AS152900	1

The wildcard records return a rotating subset of 20 IPs per query; the remaining 5 appear only in bare domain resolution for `nvms9000[.]su` (3 IPs) and `daylightbomb[.]elite` (4 IPs, 2 overlapping with the wildcard pool). The GHOSTnet IPs are sequential (`5.230.170.237` – `.240`), suggesting a small dedicated allocation. Full IPs are in [iocs/ips.csv](#) .

C2 protocol

Registration

The bot sends a 62-byte encrypted registration containing a bot ID, the C2 domain used for connection, raw socket capability, and system information. Sixty-two bytes — the bot announces itself with less overhead than a TCP handshake. The C2 responds with binary keepalives (2-byte messages) and text-based attack commands.

Command format

Attack commands are plaintext, space-delimited with 8 fields:

```
[type] [target] [duration] [dst_port] [src_port] [payload_size] [pps_delay] [payload_data]
```

The C2 can also push domain and IP updates (`ud` command) and modify the process whitelist (`whitelist` command).

Encryption

The protocol uses a custom stream cipher. We can decrypt all observed traffic. The methodology — emulating the ARM cipher function directly with the [Unicorn CPU emulator](#) rather than reimplementing from decompilation — is a polite way of saying we stopped trying to understand the cipher and started letting it explain itself. Defenders interested in the decryption approach can contact us directly.

Attack methods

8 DDoS attack types, 7 using raw sockets with IP spoofing:

Type	Method	Spoofed	Notes
0	UDP flood	Yes	Most dispatched method; 1,400B or 1B payloads
1	TCP SYN flood	Yes	Rarely observed
2	TCP PSH+ACK flood	Yes	Second most common; often targets port 80
3	ICMP echo flood	Yes	Not observed in monitoring
4	UDP flood (socket)	No	Only unspoofed method; used alongside type 0
5	UDP flood (alt)	Yes	Rarely observed; used for DNS amplification
6	TCP ACK flood	Yes	Rarely observed
7	GRE flood	Yes	Not observed; 566 spoofed source ranges (~204M IPs)

The GRE flood draws spoofed source addresses from a hardcoded pool of 566 IP ranges covering approximately 204 million addresses, almost entirely Chinese allocations. The spoofed traffic appears to originate from Chinese ISPs regardless of where the bots actually sit.

Observed attacks

In recent C2 monitoring, we observed over 70 attack commands dispatched at a rate of roughly one every few minutes, each specifying a 60-second duration. The target profile is diverse: game servers (Valve Source Engine, Minecraft), cloud-hosted infrastructure, hosting providers, and ISP address space across Asia, Europe, and the Americas. Several commands target CIDR prefixes rather than individual IPs — including ranges as wide as /19 — indicating carpet-bombing capability designed to distribute attack traffic across an entire subnet.

The C2 uses two distinct payload strategies: 1,400-byte payloads (just under the 1,500-byte MTU) for volumetric throughput, and 1-byte payloads for maximum packets-per-second. It pairs raw-socket and standard-socket UDP floods against the same target simultaneously: spoofed and unspoofed traffic in parallel, and sustains campaigns against individual targets over hours through repeated commands. TCP floods (PSH+ACK, SYN, ACK) complement the UDP methods.

The diversity of targets, protocols, and payload strategies across short-duration bursts is consistent with a DDoS-for-hire service. The customer base, judging by the target spread, is not picky.

Host behavior

Anti-competition

Binds to local TCP port 2625 as a single-instance lock: if already bound, exits. It also scans `/proc/net/tcp` and enumerates `/proc/*/fd/` to match socket inodes to PIDs, killing competing network processes not on a C2-updatable whitelist. But where [MossadProxy](#) maintains ~30 named kill patterns, [Jackskid](#) runs rival uninstalls plus real-time process scanning, and [CECbot](#) performs explicit uninstalls, Drifter has no curated list of rival families. It kills the unfamiliar rather than the named. Competitively speaking, still in onboarding, but with a broader swing than a port lock alone would suggest.

Process masquerading

Renames its process to `ntpcclient` at runtime. The most aggressively boring cover identity available.

OOM protection

Writes `-1000` to `/proc/self/oom_score_adj`. The bot gets priority. The streaming gets best effort.

Self-destruct

Terminates after approximately 7 days (604,799 seconds — one second short of a clean week, suggesting either a fencepost error or a philosophical commitment to impermanence). This level of operational hygiene is not typically associated with 71 KB binaries, but it limits forensic exposure on devices where the dropper does not re-execute. [Katana](#) uses a similar dead-man's-switch (3 days without C2 contact); Drifter's timer is unconditional.

Lineage

Drifter shares architectural patterns with Mirai (port-based instance lock, OOM protection, process masquerading, raw socket attacks) but does not share code. No Mirai table structure, no credential list, no telnet scanner, no standard Mirai C2 protocol. These are proven design decisions adopted independently. At this point, Mirai patterns are less a lineage and more a building code. Claiming Mirai lineage for an IoT botnet is like claiming Beatles influence for a rock band. The question is no longer whether, but how much the developer is willing to admit.

The binary uses glibc LCG constants (`0x41C64E6D` , `0x3039`) in its PRNG, which also appear in [Jackskid](#). These are standard glibc `rand()` constants; shared usage indicates a textbook, not a relationship.

Detection

Network indicators

- DNS queries for `*.daylightbomb[.]elite` , `*.hikvision-cctv[.]su` , or `*.nvms9000[.]su` (any subdomain)
- DNS queries to `connectivity.accesscam[.]org`
- DNS resolution via `194.50.5[.]27` (custom resolver, non-standard for most networks)
- TCP connections to any of 8 ports: 23004, 25632, 32505, 34532, 36275, 36605, 37610, 44308
- Local TCP bind on port 2625

Host indicators

Indicator	Description
<code>io.nexus.drifter</code>	Real package name
<code>com.siliconworks.android.update</code>	Installer disguise
<code>libcyn.so</code>	Native binary filename
Port 2625 TCP bind	Anti-competition lock
Process name <code>ntpclient</code>	Runtime masquerading
<code>/proc/self/oom_score_adj</code> = -1000	OOM protection

Conclusion

Six operators on the same ADB delivery surface in three months, each with a different evasion strategy — CCTV-themed domains, custom ciphers, process whitelists — but all exploiting the same economics: cheap devices shipped with open shells and no update path. The competition between operators is intensifying faster than the defenses around the devices they share.

Indicators of compromise

Machine-readable IoC files are in [iocs/](#) :

File	Contents
domains.csv	C2 domains, connectivity check, STUN, Telegram DDR
ips.csv	C2 IP pool (25 IPs), custom DNS resolver, STUN
hashes.csv	APK and native binary SHA-256 hashes

References

- [Aisuru ecosystem report](#) — documents the broader ADB TV box battlefield and the families competing on this attack surface (Nokia Deepfield ERT, March 2026)
- [CECbot report](#) — the most recent family from the same delivery vector, for architectural contrast (Nokia Deepfield ERT, March 2026)
- Synthient, "[A Broken System Fueling Botnets](#)" (Jan 2026) — Kimwolf infection mechanism via residential proxy ADB exploitation

Edit history

Date	Change
2026-04-26	Corrected <code>.elite</code> TLD attribution from OpenNIC to Handshake (HNS). <code>.elite</code> is not in the OpenNIC TLD namespace; it is a Handshake name resolved via HNS-compatible resolvers.

Source: <https://github.com/deepfield/public-research/blob/main/drifter/report.md>