ODIN

TLP:CLEAR

ODIN RESEARCH · MONOGRAPH 41 / 30 · C2 INFRASTRUCTURE

Dual DNS and HTTP Beacons
ARM64 Agents and RustDesk Staging

Static analysis of four staging directories and a separate cloud C2 feed cohort

Prepared November 2026
Full Report · Public edition · Version 1.1

4 Hosts
Open-directory staging servers in the US, Hong Kong, China, and South Korea (addresses withheld; 0 shared hosts)

116 / 106
Exposed directory files reviewed (82, 12, 21, and 1) and 106 global unique SHA-256 digests

21 + 27
21 Windows DNS/HTTP beacon builds (14 root + 7 equal-length secondary copies) and 27 cross-OS ELF, Go, and Mach-O builds on Workbench A

12 / 8
Feed-labeled Cobalt Strike C2 ports (four dual-port 80/443 pairs plus four 443 listeners) on eight withheld AWS and Azure hosts

What Is in This Report. 11 chapters and two appendices.

Static analysis of four staging directories and a separate cloud C2 feed cohort

01	Executive Summary	04
02	Four-Workbench and Cloud C2 Analytical Model	07
03	Workbench A: Dual-Channel DNS and HTTP Beacons With Cross-Platform Agents	11
04	Workbenches B, C, and D: Root-Pwn Persistence, RustDesk Staging, and Lateral Kits	14
05	Post-Exploitation Cloud C2 Cohort: Feed-Labeled Cobalt Strike vs Other Feed Matches	18
06	Why Most Feed and Directory Entries Stay Off Blocklists	21
07	What Defenders Should Do	23
08	Detection Rules	25
09	Indicators	28
10	Scope and Limits	31
11	Sources	33
A	Indicators of compromise (9)	35
B	Detection rules (4 files)	36

IN SHORT

Across records dated 14–20 November 2026, Odin by Brandefense analyzed 116 exposed files across four open HTTP directories alongside 153 command-and-control endpoints from 97,103 feed records. The directories stage DNS and HTTP beacons, Linux and macOS implants, privilege-escalation scripts, and a legitimate RustDesk package. Twelve feed-labeled Cobalt Strike ports on eight cloud hosts lack independent protocol confirmation. Hunt the five custom implant digests and four TLS JARM fingerprints, inspect recursive DNS traffic, and restrict unauthorized remote desktop installers.

01

CHAPTER 01

Executive Summary

IN SHORT

Across records dated 14–20 November 2026, Odin by Brandefense analyzed 116 exposed files across four open HTTP directories alongside 153 command-and-control endpoints from 97,103 feed records. The directories stage DNS and HTTP beacons, Linux and macOS implants, privilege-escalation scripts, and a legitimate RustDesk package. Twelve feed-labeled Cobalt Strike ports on eight cloud hosts lack independent protocol confirmation. Hunt the five custom implant digests and four TLS JARM fingerprints, inspect recursive DNS traffic, and restrict unauthorized remote desktop installers.

INSIDE THIS CHAPTER

Three Decisions for Leaders 04

The Study in Numbers 05

01 Executive Summary

Enterprise networks rarely run one operating system or permit direct outbound HTTPS traffic from every segment. Windows workstations sit behind authenticating web proxies, cloud workloads run on x86_64 and ARM64 Linux instances, and developer laptops run Apple Silicon macOS. Post-exploitation toolkits mirror that split.

The staged DNS beacon and RustDesk package could provide fallback paths if deployed. These records do not show a blocked callback, an installation, or target use.

We reviewed file paths, byte sizes, dates, and SHA-256 digests across four open HTTP directories discovered between 14 and 20 November 2026. Defenders can use cyber threat intelligence to assess protocol and certificate records across command-and-control feed families. No binary was run and no external target was contacted. All four open-directory addresses and all 137 C2 host addresses (153 reviewed endpoints) remain withheld.

1. **Dual-channel DNS and HTTP beacons sit beside cross-platform Go and ELF implants.**
   Observed from 19 to 20 November 2026 in the United States, an 82-file directory stages fourteen root Windows DNS and HTTP beacons alongside seven secondary copies of identical byte sizes. Next to them sit a custom Linux ELF implant and twenty-six cross-platform Go binaries compiled for Linux AMD64, Linux ARM64, Windows AMD64, Windows ARM64, and macOS Intel and Apple Silicon.

2. **Linux privilege-escalation kits pair ARM64 WebSocket stagers with a RustDesk installer.**
   Observed on 20 November 2026 in Hong Kong, a twelve-file directory combines an eight-kilobyte Linux root escalation script with custom Linux x86_64 and ARM64 WebSocket implants. The folder also stages a legitimate twenty-five-megabyte RustDesk Debian package. Two separate directories in China and South Korea stage internal scanners, Mimikatz builds, and a 32-bit Linux TCP stager.

3. **Upstream C2 feed labels require independent protocol verification before firewall blocking.**
   Across 153 recently observed endpoints on 137 hosts spanning Cobalt Strike, Sliver, and Metasploit feeds, twelve Cobalt Strike ports on eight AWS and Azure hosts carry confirmed feed labels. The supplied inventory records do not establish independent beacon or protocol confirmation, so all eight cloud addresses and the other 141 endpoints remain withheld from blocklists.

Three Decisions for Leaders

1. **Require independent protocol confirmation before blocking feed C2 addresses.** Keep uncorroborated cloud and dynamic-ISP feed addresses off automated firewall blocklists, and use the four Cobalt Strike TLS JARM fingerprints for multi-signal hunting. Inspect recursive DNS logs for high-entropy TXT, A, and AAAA tunneling queries.
2. **Treat unauthorized remote desktop packages on Linux servers as persistence events.** Enforce package-repository allowlists and systemd policies across Linux fleets. Debian or RPM installers for RustDesk and similar tools must never install outside approved administration hosts.
3. **Separate custom implant hashes from public dual-use tools and shared TLS fingerprints.** Load the five custom beacon, ELF, ARM64, and escalation SHA-256 digests into EDR hunting while keeping stock Mimikatz, fscan, RustDesk, and shared JARM fingerprints off automated blocklists.

The Study in Numbers

Number	What it is
4 Hosts	Open-directory staging servers in the US, Hong Kong, China, and South Korea (addresses withheld; 0 shared hosts)
116 / 106	Exposed directory files reviewed (82, 12, 21, and 1) and 106 global unique SHA-256 digests
21 + 27	21 Windows DNS/HTTP beacon builds (14 root + 7 equal-length secondary copies) and 27 cross-OS ELF, Go, and Mach-O builds on Workbench A
97,103 / 153	Feed records across four C2 families and 153 recently observed endpoints reviewed across 137 hosts
12 / 8	Feed-labeled Cobalt Strike C2 ports (four dual-port 80/443 pairs plus four 443 listeners) on eight withheld AWS and Azure hosts
153 + 41	All 153 reviewed C2 endpoints (12 feed-labeled plus 141 other rows) and 41 dual-use, remote-desktop, and log files kept off blocklists
8 + 4 IOCs	Published custom DNS/HTTP beacon, Linux ELF, ARM64 WebSocket, and root-pwn SHA-256 hashes (8) plus Cobalt Strike TLS JARM fingerprints (4) for hunting
6 + 3 Rules	Published Suricata 7 rules (6), bounded YARA rules (3), Sigma rules (2), and a Sentinel KQL hunting query

How to use this report.

Security leaders should read Chapters 1, 6, and 7. Endpoint, Linux, and cloud architects should review Chapters 3, 4, and 7. SOC analysts and detection engineers should focus on Chapters 5, 8, and 9.

CHAPTER 02

Four-Workbench and Cloud C2 Analytical Model

IN SHORT

In records dated 14–20 November 2026, Odin by Brandefense reviewed 116 exposed files (106 unique SHA-256 digests) on four open HTTP directories alongside 153 C2 endpoints on 137 hosts. Zero shared IP addresses connect the four directories to each other or to the C2 hosts. Treat this synthesis as an analytical model separating five custom hashes and four TLS JARM fingerprints from unverified C2 and dual-use records.

INSIDE THIS CHAPTER

Four Open HTTP Staging Directories (116 Files) 08

Five Artifact Groups Across the 116 Staged Files 08

02 Four-Workbench and Cloud C2 Analytical Model

Intrusions rarely stay confined to one binary format or one network protocol. Windows endpoints receive DLLs, service executables, or raw shellcode over HTTP, HTTPS, or DNS, while Linux servers and containers receive x86_64, ARM64, or 32-bit i386 ELF binaries. Once inside, operators add internal scanners, LSASS dumpers, or legitimate remote desktop packages to maintain backup access.

Comparing attack surface records for the four open directories and 137 C2 hosts shows **zero shared IPv4 addresses**. Between Workbench A and Workbench C, the only hash overlap is three unmodified public `Mimikatz` files (`mimikatz.exe`, `mimilib.dll`, and `mimidrv.sys`), with **zero shared custom hashes**. We therefore present them explicitly as an **analytical model** rather than one threat group. Figure 1 maps each slice to its defensive action.

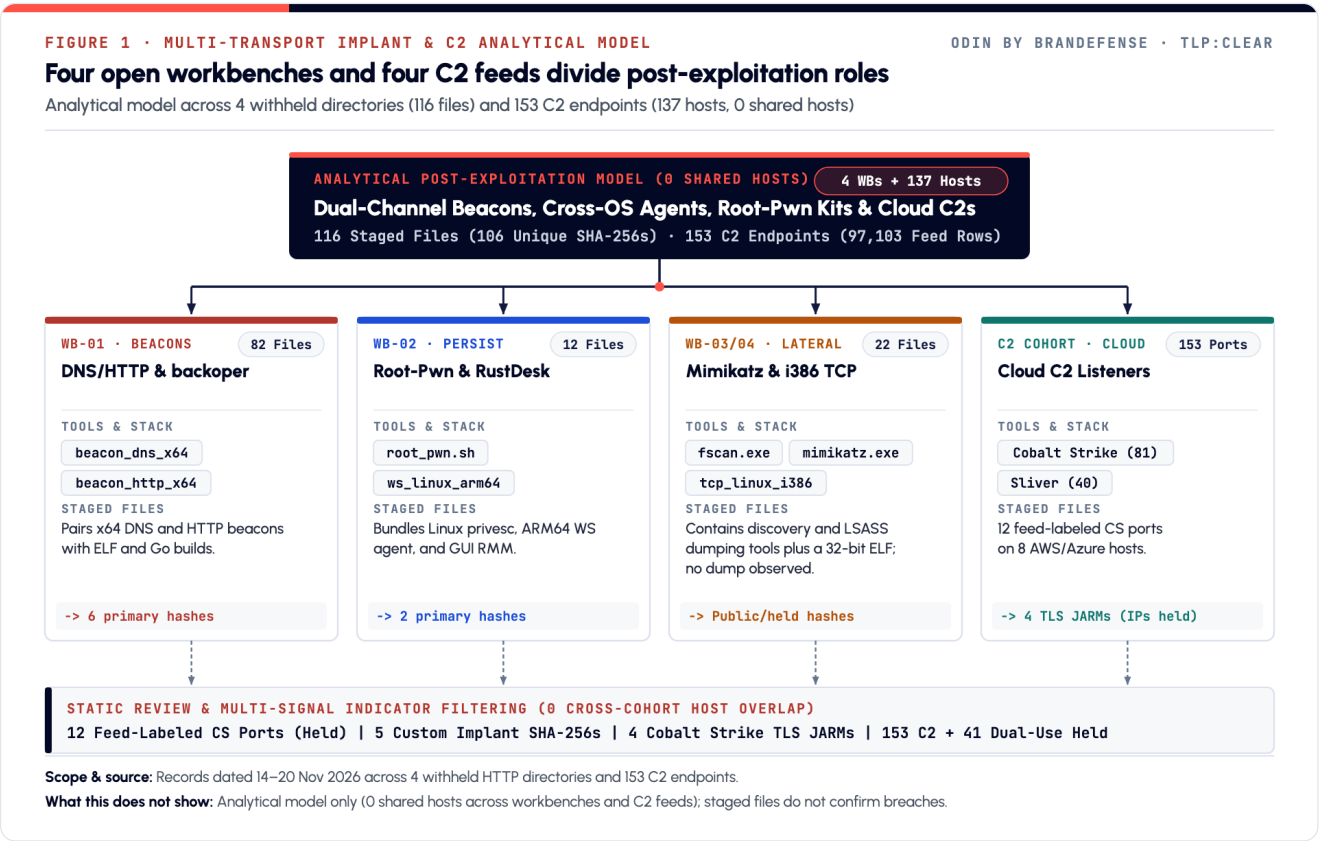


Figure 1: Four open workbenches and four C2 feeds divide post-exploitation roles

Four Open HTTP Staging Directories (116 Files)

Operators on rented virtual private servers often run an ad-hoc web server, such as `python3 -m http.server`, to pull payloads with `curl` or PowerShell. Leaving that server bound to `0.0.0.0` exposes the entire folder, which defenders can find through external attack surface management. Table 2.1 summarizes all four directories (addresses withheld).

Directory	Geo & port	Files	Hashed	Unique SHA-256	Unhashed	Observed dates	Primary staged contents
Workbench A (`WB-01`)	US · :8000	82	81	81	1	19 to 20 Nov 2026	Dual DNS/HTTP beacons, `backoper_agent.elf`, Go builds & scripts
Workbench B (`WB-02`)	HK · :8000	12	11	10	1	20 Nov 2026	`root_pwn.sh`, `ws_linux_arm64`, `c_agent`, `RR.exe` & `rustdesk-1.5.1-x86_64.deb`
Workbench C (`WB-03`)	CN · :80	21	21	17	0	14 to 20 Nov 2026	`fscan.exe`, `Win32` / `x64` `Mimikatz`, `kiwi_passwords.yar` & archives
Workbench D (`WB-04`)	KR · :80	1	1	1	0	20 Nov 2026	Single 32-bit Linux TCP stager (`/tcp_linux_i386`, 2,607,220 B)

Across the 116 entries (78 executables/DLLs/shellcode, 19 text/log files, 14 scripts, and 5 archives), 114 carry SHA-256 digests. Two entries record no digest at source: `/RR.exe` (75,678,656 B) on Workbench B and one unindexed text entry on Workbench A. Within-server deduplication reduces 114 hashed rows to 109 per-server unique digests.

Workbench B holds one duplicate 9,848 B pair (`/system_sa` and `/wws`), while Workbench C holds four duplicate pairs (`Mimikatz` copies and `/tools.dat` matching `/tools.zip`). Removing the 3 shared public `Mimikatz` digests leaves **106 global unique SHA-256 digests**.

Five Artifact Groups Across the 116 Staged Files

Static review partitions all 116 entries into five mutually exclusive groups (`34 + 25 + 17 + 16 + 14 = 106` unique SHA-256 digests):

Artifact group	Files	Share	Unique SHA-256	Unhashed	Servers	Size range (`B`)	Defensive handling
Cross-platform ELF, ARM64 & Go agents	35	30.2%	34	0	3	9,848 to 6,430,904	Hunt `backoper_agent.elf` & `ws_linux_arm64`
Windows DNS/HTTP beacons & shellcode	26	22.4%	25	1	2	212 to 75,678,656	Hunt 5 DNS/HTTP beacon & shellcode hashes
Credential harvest & lateral tooling	24	20.7%	17	0	2	2,834 to 9,258,745	Hold public `Mimikatz` / `fscan` hashes; detect LSASS/SMB
Legitimate RMM package & log files	17	14.7%	16	1	3	27 to 25,337,156	Hold `RustDesk` `.deb` & log hashes; enforce RMM policy
Privilege-escalation & callback scripts	14	12.1%	14	0	2	102 to 8,030	Hunt `root_pwn.sh` hash & HTTP script pulls

Figure 2 connects these five groups and the feed-labeled Cobalt Strike cohort to our published detection and hunting rules.

FIGURE 2 · ARTIFACT TAXONOMY & DETECTION CAPABILITY MATRIX

ODIN BY BRANDEFENSE · TLP:CLEAR

Detection rules separate custom beacons and ELF agents from dual-use kits

Five static workbench classes (116 files, 106 unique SHA-256s) plus 12 feed-labeled Cobalt Strike C2 ports

1. TOOL / BINARY	2. HOW USED (CLI / CONFIG)	3. TARGET SURFACE	4. DEFENDER SIGNAL
Dual DNS/HTTP Beacons 26 files (25 SHA-256s)	DNS/HTTP DLL, EXE & shellcode `beacon_dns_x64 + beacon_http_x64`	Windows endpoints & DNS/HTTP	`sid:9541001 + Sigma 2`
ELF, ARM64 & Go Agents 35 files (34 SHA-256s)	Linux/macOS/ARM64 C2 implants `backoper_agent.elf + ws_linux_arm64`	Linux x86_64/ARM64 & macOS	`sid:9541002-003 + Sigma 1`
Privesc & OOB Scripts 14 files (14 SHA-256s)	Linux root script & HTTP staging `root_pwn.sh + oob_listener.py`	Linux privilege & staging ports	`sid:9541004, 006 + YARA 2`
Feed-Labeled Cloud CS C2 12 ports on 8 cloud hosts	Paired :80 HTTP & :443 HTTPS C2 `7 AWS + 1 Azure Cobalt Strike hosts`	Outbound enterprise HTTP/TLS	`4 TLS JARMs (hunt-only, IPs held)`
Mimikatz & Lateral Kits 24 files (17 SHA-256s)	Internal host discovery & LSASS `fscan.exe + kiwi_passwords.yar`	Active Directory & SMB subnets	`YARA 3 (public hashes held)`
RustDesk .deb & Logs 17 files (16 SHA-256s)	Legitimate GUI RMM & OOB logs `rustdesk-1.5.1-x86_64.deb`	Linux GUI/systemd & staging logs	`sid:9541005 + KQL RMM hunt`

Scope & source: Covers all 116 exposed entries across the four HTTP directories and 12 feed-labeled C2 endpoints.

What this does not show: Public Mimikatz, fscan, and RustDesk package hashes are excluded from automated malware hash blocklists.

Figure 2: Detection rules separate custom beacons and ELF agents from dual-use kits

CHAPTER 03

Workbench A: Dual-Channel DNS and HTTP Beacons With Cross-Platform Agents

IN SHORT

Observed in the United States on port 8000 between 19 and 20 November 2026, Workbench A exposes 82 files (81 unique SHA-256 digests). It pairs 21 Windows DNS and HTTP beacon builds (14 root, 7 equal-length secondary) with `/backoper_agent.elf` (6,430,904 B) and 26 cross-platform Go implants across Linux, Windows, and macOS. Inspect recursive DNS queries alongside web proxy logs and extend EDR coverage to Linux ARM64 and macOS workloads.

INSIDE THIS CHAPTER

Equal-Length Secondary Builds Across 21 DNS and HTTP Beacons 11

`/backoper_agent.elf` and the 27-Binary Cross-OS Go (`gopher`) Fleet 12

Out-of-Band Callback Scripts and Late Credential Staging 12

03 Workbench A: Dual-Channel DNS and HTTP Beacons With Cross-Platform Agents

Workbench A (`WB-01`, United States, port 8000) operates as a multi-transport build directory. Its 82 files include 23 Windows beacon and shellcode binaries alongside 27 cross-platform ELF, Mach-O, and Go implants. The folder also stages 13 delivery scripts, 15 log and manifest files, and 4 Windows credential utilities added on 20 November 2026.

Equal-Length Secondary Builds Across 21 DNS and HTTP Beacons

Workbench A exposes 14 Windows beacon binaries in its root folder alongside 7 companion beacons in a task-numbered subfolder (name generalized). Comparing byte sizes against SHA-256 digests shows equal-length secondary files:

- **Seven equal-length pairs with distinct SHA-256 digests.** Every secondary beacon has the same byte length as its root counterpart but a different SHA-256. The seven pairs cover `/beacon_dns_x64.dll` (166,400 B), `/beacon_dns_x64.exe` (167,424 B), `/beacon_http_x64.dll` (103,936 B), and `/beacon_http_x64.exe` (104,448 B). The remaining three pairs cover `/beacon_http_x64_shellcode.bin` (103,935 B), `/beacon_http_x64_svc.exe` (104,448 B), and `/beacon_http_x86.exe` (95,232 B). The inventory establishes equal lengths and different hashes, but does not identify the changed bytes.
- **DNS beacons add roughly 62.5 KB over HTTP builds.** `/beacon_dns_x64.dll` (166,400 B) and `/beacon_dns_x64.exe` (167,424 B) measure 62,464 B and 62,976 B larger than their HTTP counterparts (103,936 B and 104,448 B). The file inventory establishes the size difference but does not identify the code responsible. DNS is a possible C2 transport (`T1071.004`).
- **Service wrappers, IAT-evasion builds, and shellcode.** Additional root binaries include service wrappers (`/beacon_win_x64_svc.exe` at 104,448 B, `/beacon_win_x64_svc2.exe` at 94,720 B, and `/beacon_win_x64_svc_v1.exe` at 169,984 B). The folder also stages an import-table evasion build (`/beacon_win_x64_iat.exe` at 94,208 B), versioned executables (104,448 B to 169,984 B), `/agent.x64.exe` (104,448 B), and a 212-byte `second_stage.bin` payload.

Staged file(s) on Workbench A (:8000)	Byte size (`B`)	Copies	Architecture & role
`/beacon_dns_x64.dll` & `/beacon_dns_x64.exe`	166,400 / 167,424	2 each	64-bit Windows DNS C2 Beacon DLL and executable
`/beacon_http_x64.dll`, `.exe` & `_svc.exe`	103,936 / 104,448	2 each	64-bit Windows HTTP Beacon DLL, EXE, and service wrapper
`/beacon_http_x64_shellcode.bin`	103,935	2	Position-independent x64 HTTP Beacon raw shellcode
`/beacon_http_x86.exe` & `/beacon_win_x64_iat.exe`	95,232 / 94,208	2 / 1	32-bit HTTP Beacon and x64 import-table evasion build
`/beacon_win_x64_svc2.exe`, `_v1.exe`, `_v2.exe`	94,720 to 169,984	1 each	Service and versioned 64-bit Windows Beacon variants

`/backoper_agent.elf` and the 27-Binary Cross-OS Go (`gopher`) Fleet

Beside the Windows beacons sits a 27-binary cross-platform Go implant fleet targeting six OS and CPU combinations:

- **Eighteen 6,430,904-byte Linux AMD64 ELF builds.** Eighteen files share the exact length 6,430,904 B (6.13 MB) with 18 distinct SHA-256 digests. Generic builds include `/backoper_agent.elf`, `/agent.bin`, `/agent_custom_tag.bin`, `/gopher_linux_amd64.elf`, `/gopher_kali_linux_amd64.` and `/gopher_linux_amd64_c2`. They also include `/gopher224.elf`, `/gopher_201_new.elf`, `/gopher_relay201.elf`, `/g_4tomcat.elf`, `/g42.bin`, and seven task-tagged builds (target-specific prefixes withheld). The filenames match Mythic's `gopher` agent conventions or a custom fork, while the inventory does not identify which bytes differ across the 18 builds.
- **ARM64 Linux, ARM64 Windows, and macOS Mach-O coverage (9 builds).** Two equal-length `linux/arm64` ELF files (`/gopher_linux_arm64.elf` and its subfolder copy) measure 5,963,960 B each with distinct hashes. Three `windows/amd64` builds measure 6,048,256 B each, and two `windows/arm64` builds measure 5,523,456 B each. Two macOS Mach-O binaries cover Intel (`/gopher_macos_amd64.macho` at 6,209,664 B) and Apple Silicon (`/gopher_macos_arm64.macho` at 5,854,658 B).

Out-of-Band Callback Scripts and Late Credential Staging

Thirteen scripts and fifteen log files support delivery and callback verification. Python scripts include `/oob_listener.py` (1,485 B). Five callback logs (73 B to 599 B) reference ports 8890, 9001, 9123, and 10143.

Companion scripts include `/execws.py` (1,929 B), `/conn.py` (269 B), and `/conn2.py` (399 B). Additional helpers include `/serve_raw.py` (553 B), `/serve_small.py` (499 B), and a 485 B HTTP server helper.

The directory also holds an IMAP canary (2,204 B), a 102 B batch file, and a 3,010 B PowerShell sweep script. An appliance-themed subfolder holds `c2_server.py` (5,198 B), `agent.js` (744 B), `start.sh` (296 B), and three C2 logs. All 78 beacon, Go, script, and log files were indexed on 19 November 2026. Four Windows credential dumpers (`/mimikatz.exe`, `/mimilib.dll`, `/mimidrv.sys`, and `/ps64.exe`) were added on 20 November 2026.

CHAPTER 04

Workbenches B, C, and D: Root-Pwn Persistence, RustDesk Staging, and Lateral Kits

IN SHORT

Workbenches B (Hong Kong, port 8000, 12 files), C (China, port 80, 21 files), and D (South Korea, port 80, 1 file) show the post-foothold side of the model. They combine `/root_pwn.sh`, ARM64 and i386 Linux implants, a legitimate `RustDesk` Debian installer, and `Mimikatz` plus `fscan` lateral kits. Block unauthorized Linux RMM packages while hunting `/root_pwn.sh` and `/ws_linux_arm64`.

INSIDE THIS CHAPTER

Workbench B (:8000, Hong Kong): `/root_pwn.sh`, `/ws_linux_arm64`, and `RustDesk` Abuse 14

Workbenches C (:80, China) and D (:80, South Korea): `fscan`, `Mimikatz`, and `/tcp_linux_i386` 15

04 Workbenches B, C, and D: Root-Pwn Persistence, RustDesk Staging, and Lateral Kits

Workbenches B, C, and D illustrate privilege escalation, interactive remote-access persistence, and internal network discovery. Figure 3 models a hypothetical fallback sequence, and Figure 4 compares the four directory layouts.

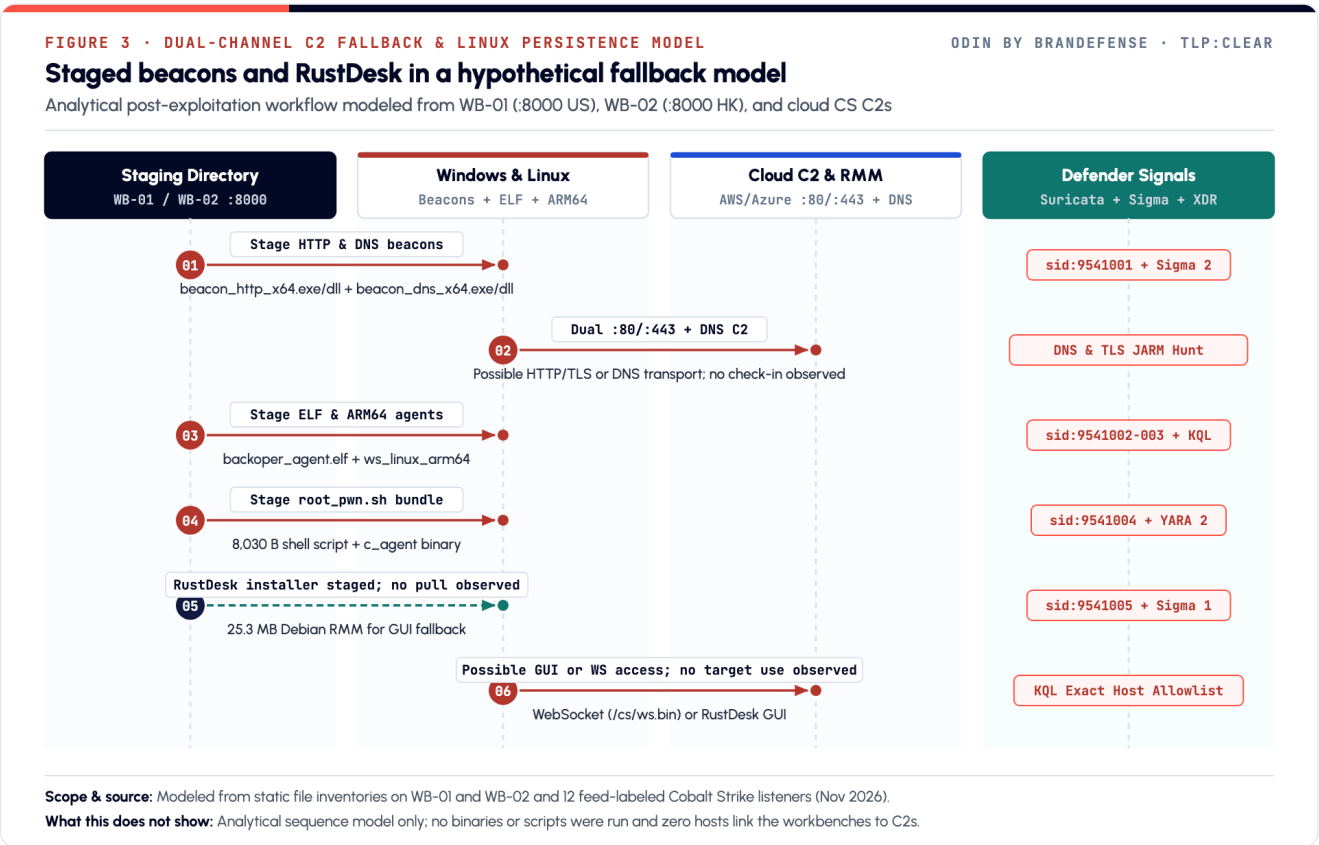


Figure 3: Staged beacons and RustDesk in a hypothetical fallback model

Workbench B (:8000, Hong Kong): `/root_pwn.sh`, `/ws_linux_arm64`, and `RustDesk` Abuse

Observed on 20 November 2026 in Hong Kong on port 8000, Workbench B (`WB-02`) exposes 12 files (10 unique SHA-256 digests plus 1 unhashed executable):

- **Linux root escalation and WebSocket C2 stagers (9 files).** `/root_pwn.sh` (8,030 B) sits alongside three 9,848 B Linux binaries: `/system`, `/system_sa`, and byte-identical `/wws`. Persistent C2 binaries span `/ws` (2,922,708 B), `/ws_linux_arm64` (10,816 B), `/cs/ws.bin` (2,100 B), and two equal-length 534,256 B Linux agents (`/c_agent` and `/c_a`) with distinct SHA-256 digests.
- **Windows companions and the legitimate `RustDesk 1.5.1` Debian package (3 files).** Beside `/system.exe` (393,728 B) and `/RR.exe` (75,678,656 B, unhashed), Workbench B stages `/rustdesk-1.5.1-x86_64.deb` (25,337,156 B). `RustDesk` is a legitimate open-source remote desktop client (`T1219`). If installed, its Linux package can register `rustdesk.service` and use relay connections. The staged file alone does not establish persistence, GUI access, or EDR fallback. Because `RustDesk` is legitimate third-party software, **its SHA-256 is withheld**; defenders should detect unauthorized HTTP `.deb` pulls (`SID 9541005`), `dpkg`/`apt` installs (`Sigma Rule 1`), and `.deb` command lines or URLs (`Sentinel KQL`).

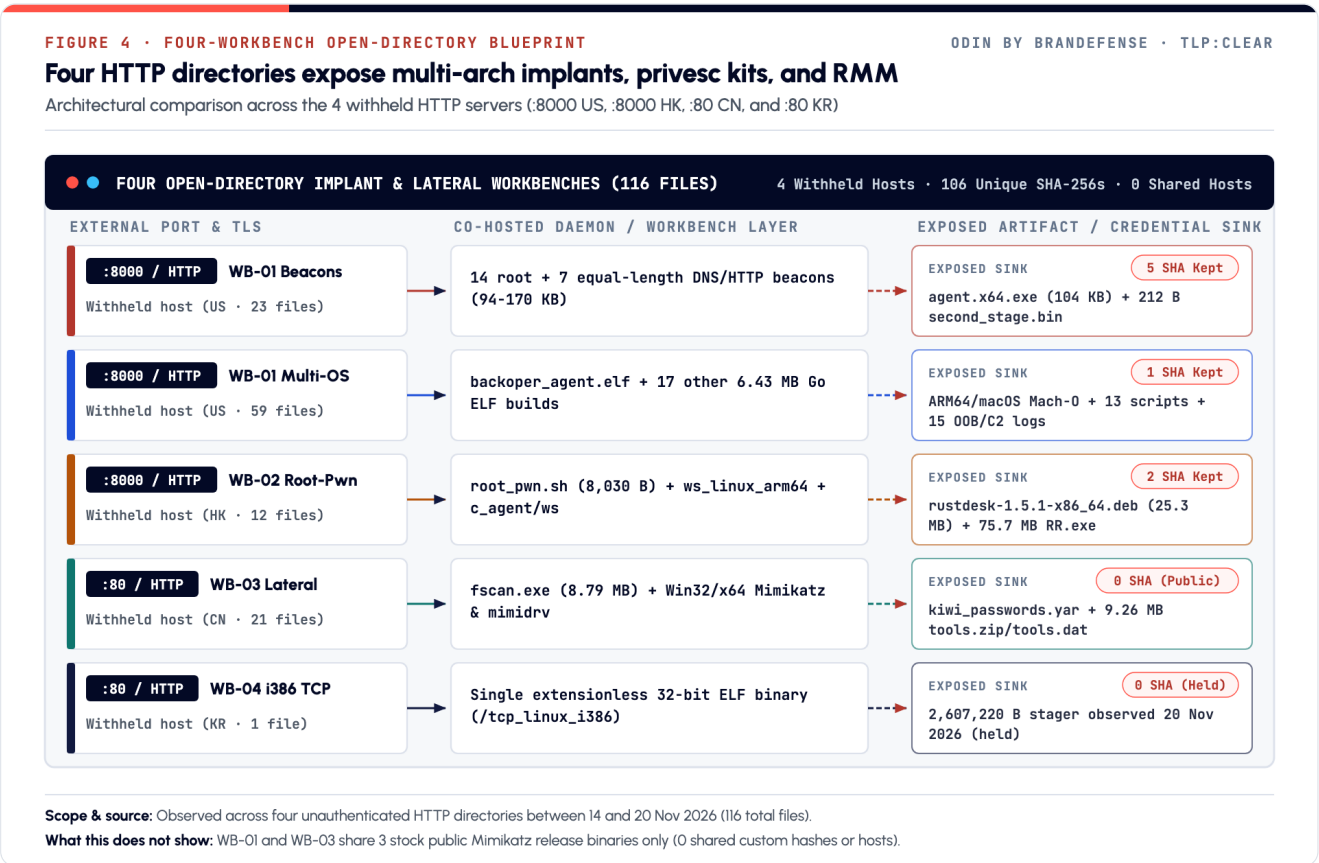


Figure 4: Four HTTP directories expose multi-arch implants, privesc kits, and RMM

Workbenches C (:80, China) and D (:80, South Korea): `fscan`, `Mimikatz`, and `/tcp_linux_i386`

- **Workbench C (`WB-03`, China, port 80, 21 files, 17 unique SHA-256s).** Observed from 14 to 20 November 2026, Workbench C stages `/fscan.exe` (8,785,920 B), `/mimikatz.exe` (1,355,264 B), `/mimilib.dll` (37,376 B), `/mimidrv.sys` (37,208 B), `/mimikatz.zip` (1,206,166 B), and `/test.txt` (27 B). It also hosts `/tools.zip`, byte-identical `/tools.dat` (9,258,745 B), and `/tools.tar.gz` (9,252,130 B). Unpacked `Win32` (`mimikatz.exe` at 1,084,416 B, `mimilib.dll`, `mimilove.exe`, `mimispool.dll`, `mimidrv.sys`) and `x64` folders sit alongside `/mimikatz/kiwi_passwords.yar` (2,834 B). All Workbench C hashes represent public dual-use releases or generic archives and stay withheld.
- **Workbench D (`WB-04`, South Korea, port 80, 1 file).** Observed on 20 November 2026, Workbench D serves a single extensionless 32-bit Linux executable at `/tcp_linux_i386` (2,607,220 B), suited to legacy 32-bit Linux appliances. Because this staging directory holds a single stager binary without companion kit files, its hash stays withheld.

Directory & port	File path	Byte size (B)	Role & indicator handling
Workbench B (HK : 8000)	/root_pwn.sh & /ws_linux_arm64	8,030 / 10,816	Linux root script & ARM64 WebSocket stager · **Published SHA-256s**
Workbench B (HK : 8000)	/c_agent, /c_a, /ws, /cs/ws.bin	2,100 to 2,922,708	Equal-length Linux agents & WebSocket stagers (/c_agent hash withheld)
Workbench B (HK : 8000)	/rustdesk-1.5.1-x86_64.deb	25,337,156	Legitimate RustDesk Debian package · **Hash withheld** (RMM policy)
Workbench C (CN : 80)	/fscan.exe, Mimikatz, kiwi_passwords.yar	2,834 to 9,258,745	Public internal scanner, Win32 / x64 Mimikatz & archives · **Held**
Workbench D (KR : 80)	/tcp_linux_i386	2,607,220	Standalone 32-bit Linux TCP stager · **Hash withheld**

05

CHAPTER 05

Post-Exploitation Cloud C2 Cohort: Feed-Labeled Cobalt Strike vs Other Feed Matches

IN SHORT

We reviewed 153 endpoints on 137 hosts from 97,103 Cobalt Strike, Sliver, and Metasploit feed records. Twelve ports on eight AWS and Azure hosts carry confirmed Cobalt Strike feed labels. The supplied records do not establish independent beacon or protocol confirmation. Withhold all eight cloud addresses from firewall blocklists and route their four TLS JARM fingerprints and 141 other endpoints to hunting.

INSIDE THIS CHAPTER

Twelve Feed-Labeled Cobalt Strike Ports Across Eight Withheld Cloud Hosts 18

Four Patterns Behind the Remaining 141 Withheld C2 Endpoints 19

05 Post-Exploitation Cloud C2 Cohort: Feed-Labeled Cobalt Strike vs Other Feed Matches

To evaluate upstream C2 feed records, we examined 153 recently observed endpoints from 19 to 20 November 2026. Those endpoints span 137 distinct hosts drawn from 97,103 cumulative feed records. Table 5.1 summarizes all four categories.

Feed category	Total feed rows	Feed-verified	Countries	Reviewed endpoints	Distinct hosts	Feed-confirmed (`Hunt`, withheld)	Other reviewed (`Hunt`, withheld)
`cobaltstrike`	80,375	39	105	71	60	**12 (8 hosts withheld)**	59 (cloud, cert collision & telecom)
`cobaltstrikeunverified`	7,037	0	62	10	9	**0**	10 (unverified Azure/AWS TLS)
`sliver`	5,592	23	83	40	36	**0**	40 (ports 443, 8443, 31337 & ISP)
`metasploit`	4,099	3	85	32	32	**0**	32 (shared JARM, :3790 & ISP)
Total	**97,103**	**65**	**-**	**153**	**137**	**12 (8 hosts withheld)**	**141 (129 hosts withheld)**

Twelve Feed-Labeled Cobalt Strike Ports Across Eight Withheld Cloud Hosts

The source inventory labels 12 of 153 reviewed endpoints as confirmed Cobalt Strike, but the supplied records do not establish independent beacon or protocol confirmation. Those 12 endpoints (7.8% of the sample, or 0.012% of 97,103 feed rows) were observed on 19 November 2026 across eight commercial cloud hosts (addresses withheld):

- **AWS and Azure concentration (7 AWS hosts, 1 Azure host, addresses withheld).** Seven hosts (11 ports) sit on `Amazon.com, Inc.` in Australia (3 hosts), Germany (2 hosts), the United States (1 host), and Japan (1 host). The eighth host (:443) sits on `Microsoft Corporation` (Azure) in the United States. Because the inventory does not include independent beacon or protocol confirmation, all eight IPv4 addresses are withheld and excluded from firewall blocklists.
- **Dual-port HTTP (80) and HTTPS (443) pairing (4 hosts / 8 ports).** Four AWS hosts (one in the United States, two in Germany, and one in Australia) expose paired Cobalt Strike listeners on TCP 80 and 443. Two Australian AWS hosts also share an identical custom certificate Common Name (withheld) and identical JARM (`29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96`).

Withheld cloud host & port(s)	Geo & cloud provider	TLS JARM fingerprint (on :443)	Observed date
Cloud Host 1 (:443, address withheld)	US · Microsoft Corporation	`27d40d40d00040d00042d43d0000007d20e9b9a3e87aa1ec17cb4d646ce9a5`	19 Nov 2026
Cloud Host 2 (:443 & :80, address withheld)	US · Amazon.com, Inc.	`1dd40d40d00040d0001dd40d1dd40dc7fe523f2d40ec58fa09eb4367e88094`	19 Nov 2026
Cloud Host 3 (:443 & :80, address withheld)	DE · Amazon.com, Inc.	`29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96`	19 Nov 2026
Cloud Host 4 (:443 & :80, address withheld)	DE · Amazon.com, Inc.	(no JARM recorded on capture)	19 Nov 2026
Cloud Host 5 (:443 & :80, address withheld)	AU · Amazon.com, Inc.	`29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96`	19 Nov 2026
Cloud Host 6 (:443, address withheld)	AU · Amazon.com, Inc.	`29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96`	19 Nov 2026
Cloud Host 7 (:443, address withheld)	AU · Amazon.com, Inc.	`1dd40d40d00040d00042d43d000000e1ea2a807a629b496b664cf07ad7c08d`	19 Nov 2026
Cloud Host 8 (:443, address withheld)	JP · Amazon.com, Inc.	`29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96`	19 Nov 2026

Four Patterns Behind the Remaining 141 Withheld C2 Endpoints

1. **Dynamic telecom broadband churn (10 ISP endpoints).** Two Cobalt Strike rows tagged `confirmed` upstream sit on dynamic `Chinanet` subscriber space. Eight more Cobalt Strike, Sliver, and Metasploit rows sit on Chinanet, China Telecom, China Mobile, and Moroccan carrier networks.
2. **Cross-family JARM and CDN certificate collisions (22 endpoints).** JARM `29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96` appears on 4 feed-confirmed Cobalt Strike AWS hosts, 2 other Cobalt Strike hosts, and 12 Metasploit :443 AWS hosts (18 total). Three more Cobalt Strike rows present shared CDN/appliance certificates (`CloudFlare Origin Certificate`), and one shares an open directory.
3. **Unverified Azure TLS templates (10 endpoints).** Eight of ten `cobaltstrikeunverified` rows sit on Microsoft Azure (`US` and `NL`) presenting generic `CN=ARM` (7 endpoints on 6 hosts) or `CN=CA` (1 endpoint) certificates without beacon confirmation.
4. **Sliver (40) and Metasploit (32) red-team defaults.** Six Sliver rows expose default multiplayer port 31337, three use `CN=localhost`, and two Metasploit rows expose default UI port 3790, matching everyday red-team setups.

CHAPTER 06

Why Most Feed and Directory Entries Stay Off Blocklists

IN SHORT

We reviewed 97,103 C2 feed rows (153 endpoints) and 116 exposed directory files (97,219 combined records). Our public indicator set retains five custom SHA-256 digests and four Cobalt Strike TLS JARM fingerprints for hunting. All 153 C2 endpoints, including twelve feed-labeled Cobalt Strike ports on eight cloud hosts, remain withheld from blocklists. Apply four analytical boundaries before acting on open-directory or feed indicators.

06 Why Most Feed and Directory Entries Stay Off Blocklists

Figure 5 compares `log10`-scaled feed and directory counts against the 12 feed-labeled Cobalt Strike ports (addresses withheld) and the 5 custom SHA-256 digests published in Chapter 9.

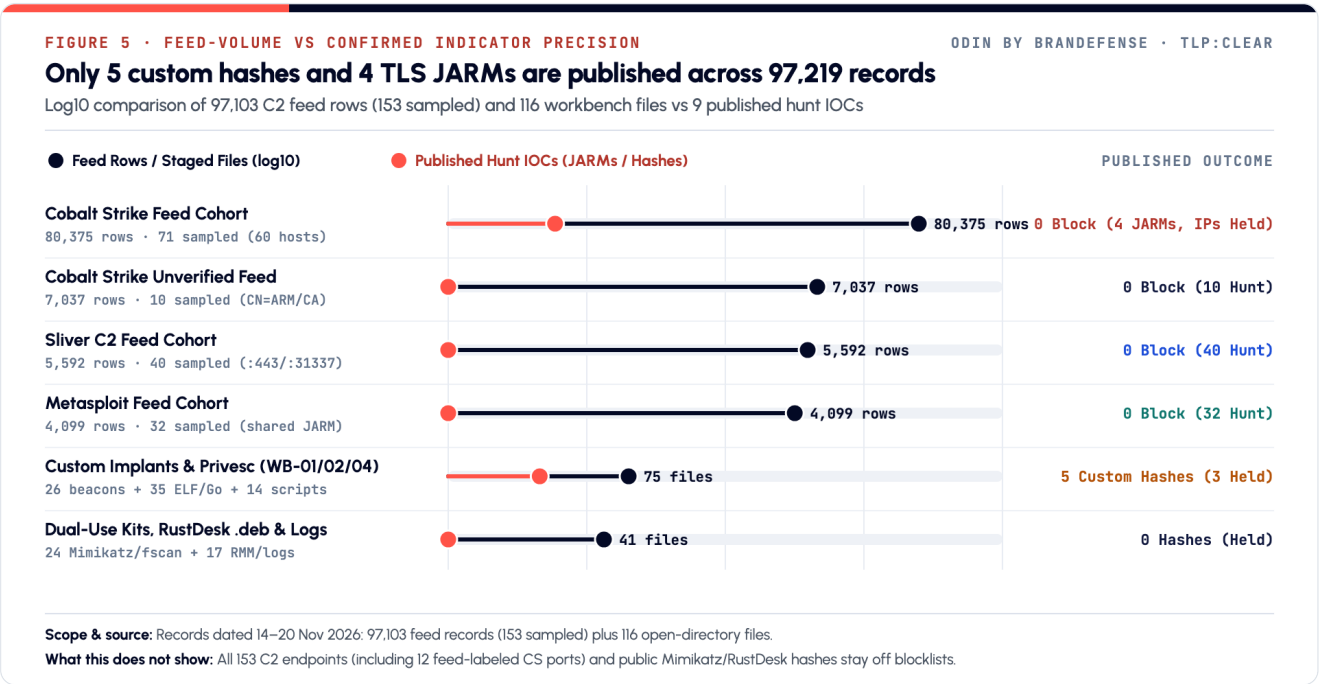


Figure 5: Only 5 custom hashes and 4 TLS JARMs are published across 97,219 records

Four practical limits govern our conclusions:

1. **Analytical model across zero shared hosts.** Zero shared IPv4 addresses or custom implant hashes link the four directories to each other or to the 137 C2 hosts. They illustrate complementary layers of post-exploitation operations, not one actor.
2. **Staged files do not establish downstream breaches.** Observing `/beacon_dns_x64.dll`, `/backoper_agent.elf`, `/root_pwn.sh`, or `/rustdesk-1.5.1-x86_64.deb` in an open directory shows what an operator built or staged, not whether any external target was breached.
3. **Withheld workbench addresses and target-tagged filenames.** All four open directories are assessed as adversary payload-staging workbenches, so generic implant filenames, staging paths, and five primary implant SHA-256 digests are published for hunting. Three more digests stay withheld because an earlier Odin report withheld the same files. We withhold the four workbench IPv4 addresses and generalize task-numbered or target-tagged filenames on Workbench A to avoid exposing target-specific strings or ephemeral VPS addresses.
4. **Why 41 dual-use, `RustDesk`, and log files stay off hash lists.** Blocking the SHA-256 of `/rustdesk-1.5.1-x86_64.deb` misclassifies an open-source Debian package, and blocking `/test.txt` (27 B) or short logs risks noisy collisions. Enforce Linux RMM policies (`T1219`) and LSASS protections (`T1003.001`) instead.

07

CHAPTER 07

What Defenders Should Do

IN SHORT

This chapter lists the actions defenders should take, in priority order.

07 What Defenders Should Do

Network and DNS security

- [ ] Require independent beacon or protocol confirmation before placing upstream C2 feed addresses on firewall blocklists, and keep uncorroborated cloud and dynamic-ISP endpoints in hunting workflows.
- [ ] Route endpoint DNS through internal resolvers, block direct outbound port 53 and unapproved DoH/DoT, and alert on high-entropy `TXT` or `AAAA` query bursts (`T1071.004`).

Linux platform and remote-access governance

- [ ] Lock `dpkg` and `apt` on Linux servers to signed internal mirrors, block ad-hoc `RustDesk` `.deb` installs (`rustdesk.service`), and restrict outbound ports 21115 to 21119 (`T1219`).
- [ ] Confirm EDR coverage across Linux `ARM64`, Linux `x86_64` / `i386`, Windows `ARM64`, and macOS `.macho` hosts, and close unauthenticated `python3 -m http.server` listeners on testing VPS assets.

Windows endpoint hardening

- [ ] Enable Credential Guard and LSA Protection (`RunAsPPL=1`), block `mimidrv.sys`, and alert when `rundll32.exe` or `regsvr32.exe` loads unsigned DLLs from writable staging folders.

SOC and threat hunting

- [ ] Deploy the Suricata, Sentinel KQL, Sigma, and YARA rules in Chapter 8 alongside the five custom SHA-256 digests in Chapter 9, keeping JARMs and public tool hashes hunt-only.

CHAPTER 08

Detection Rules

IN SHORT

This chapter describes each published rule and where to deploy it.

INSIDE THIS CHAPTER

How the Suricata Signatures Work 25

How the KQL, Sigma, and YARA Rules Work 26

MITRE ATT&CK Mapping 26

08 Detection Rules

The full deployable files sit in Appendix B. On 21 November 2026, all four files passed static checks: Suricata lint (0 problems), Kusto type-checking (0 diagnostics), `pySigma` validation (2 rules), and `YARA-X` compilation (3 rules). Because the rules were not replayed against live traffic on this build host, test and tune them in staging before deployment.

Rule file	What it catches
`odin_dual_dns_http_beacons_backoper_and_rustdesk.rules`	Suricata 7: six signatures for staged DNS/HTTP beacons, `backoper_agent.elf`, Go `gopher_*` builds, `/root_pwn.sh`, `RustDesk .deb` pulls, and open directories
`sentinel_dual_beacons_backoper_rustdesk_and_c2_hunt.kql`	Microsoft Sentinel & Defender XDR: seven-day `fullouter` join across custom hashes, staged beacon/ELF/`root_pwn.sh` processes, command lines mentioning a `RustDesk .deb` file, and staged payload URLs (returns process-only, network-only, and combined rows)
`sigma_dual_beacons_backoper_root_pwn_and_rustdesk.yml`	Sigma (2 rules): Linux `backoper` / `gopher` /ARM64/ `root_pwn.sh` and `RustDesk .deb` execution, plus Windows DNS/HTTP beacon execution
`odin_dual_beacons_backoper_and_root_pwn_artifacts.yar`	YARA (3 bounded rules): dual-beacon and `backoper` manifests, Linux `root_pwn.sh` and ARM64 WebSocket bundles, and `Mimikatz` / `kiwi_passwords.yar` / `fscan` kits

How the Suricata Signatures Work

SID	Looks for	Operational intent
9541001	HTTP `GET` for `/beacon_dns_x64*`, `/beacon_http_x64*`, `/beacon_http_x86.exe`, or `/beacon_win_x64*`	Catch HTTP pulls of staged DNS/HTTP beacons, service wrappers, IAT builds, and shellcode
9541002	HTTP `GET` for `/backoper_agent.elf`, `/agent_custom_tag.bin`, `/agent.bin`, `/agent.x64.exe`, or `/c_agent`	Catch HTTP staging of custom Linux ELF and Windows agent payloads
9541003	HTTP `GET` for `/gopher_linux_*`, `/gopher_kali_linux_amd64.elf`, `/gopher_windows_*`, `/gopher_win_x64.exe`, or `/gopher_macos_*.macho`	Catch cross-platform Go implant pulls across Linux, Windows, and macOS
9541004	HTTP `GET` for `/root_pwn.sh`	Catch retrieval of the staged Linux root privilege-escalation script
9541005	Outbound HTTP `GET` matching `/rustdesk-` and `x86_64.deb`	Policy/hunt alert for direct HTTP pulls of `RustDesk` Debian packages
9541006	Inbound HTTP 200 `Directory listing for /` naming `beacon_dns_x64`, `backoper_agent.elf`, `root_pwn.sh`, `ws_linux_arm64`, `tcp_linux_i386`, or `kiwi_passwords.yar`	Spot exposed implant staging directories browsed from internal hosts

Keep `SID 9541005` in alert-only mode where Linux administrators legitimately fetch `RustDesk` over HTTP. Pair `SID 9541001` through `SID 9541006` with internal DNS resolver logging to spot DNS beacon tunneling (`T1071.004`).

How the KQL, Sigma, and YARA Rules Work

The Sentinel KQL query looks back seven days across `DeviceProcessEvents` and `DeviceNetworkEvents`. On the process side, it matches the five custom SHA-256 digests, staged `beacon_*`, `backoper_agent.elf`, `gopher_*`, `ws_linux_arm64`, or `root_pwn.sh` command lines, and command lines mentioning a RustDesk `.deb` file. The network side adds staged-payload URL matches; the fullouter join also returns process-only and network-only rows.

Host allowlists in KQL (`ApprovedRedTeamHosts`, `ApprovedRmmPilotHosts`) and Sigma (`filter_validation_hosts`, `filter_validation_win`) match **exact hostnames only** (`REDTEAM-BUILD-VPS-01`, `SEC-VALIDATION-LNX-01`, `SEC-VALIDATION-WIN-01`, `IT-RUSTDESK-PILOT-01`). Those exclusions never hide a match on the five custom implant hashes.

Engine & rule	Severity	Fires when
Sigma 1: Linux Multi-Arch `backoper` / `gopher` / `root_pwn` or `RustDesk` `.deb`	High	Linux runs `backoper_agent.elf`, `ws_linux_arm64`, `tcp_linux_i386`, `gopher_linux_*`, `root_pwn.sh`, or `dpkg` / `apt` with `rustdesk-1.5.1-x86_64.deb`
Sigma 2: Windows Dual-Channel DNS or HTTP Beacon Execution	High	Windows runs staged `beacon_*` executables or `rundll32` / `regsvr32` loading `beacon_dns_x64.dll` or `beacon_http_x64.dll`
YARA 1–3: Bounded Manifest & Bundle Suite (< 256 KB)	Hunt / High	Matches dual-beacon/ `backoper` manifests, `root_pwn.sh` + ARM64/RMM bundles, and `kiwi_passwords.yar` + `fscan` / `Mimikatz` kits

MITRE ATT&CK Mapping

Tactic	Technique	Observed evidence	Detection
Command and Control	`T1071.004` Application Layer Protocol: DNS	`beacon_dns_x64.dll` (166,400 B) & `beacon_dns_x64.exe` (167,424 B)	`SID 9541001`; KQL; Sigma 2; YARA 1
Command and Control	`T1071.001` Application Layer Protocol: Web Protocols	`beacon_http_x64*`, `/ws_linux_arm64`, `/cs/ws.bin`, and 12 feed-labeled :80/:443 Cobalt Strike ports	`SID 9541001`; KQL; Sigma 1–2
Command and Control	`T1219` Remote Access Software	`/rustdesk-1.5.1-x86_64.deb` (25.34 MB) beside `root_pwn.sh` and `ws_linux_arm64`	`SID 9541005`; KQL; Sigma 1; YARA 2
Privilege Escalation	`T1068` Exploitation for Privilege Escalation	`/root_pwn.sh` (8,030 B) staged alongside `/system` and `/system_sa` (9,848 B)	`SID 9541004`; KQL; Sigma 1; YARA 2
Credential Access	`T1003.001` OS Credential Dumping: LSASS Memory	`Win32` / `x64` `Mimikatz`, `mimidrv.sys`, and `kiwi_passwords.yar` on Workbenches A & C	`SID 9541006`; YARA 3

09

CHAPTER 09

Indicators

IN SHORT

This chapter lists the published indicators and explains what stays withheld.

INSIDE THIS CHAPTER

Custom Implant and Script SHA-256 Digests to Hunt (5 Hashes) 28

TLS JARM Fingerprints to Hunt (4 Fingerprints) 28

Text-Only Staging Path and Filename Hunt Patterns 29

09 Indicators

Appendix A holds **9 published indicators**: **5 custom SHA-256 digests** and **4 TLS JARM fingerprints**. All 12 feed-labeled Cobalt Strike address-and-port entries across eight cloud hosts are withheld pending independent protocol confirmation. Eight text-only file and URI patterns appear below for proxy and EDR hunting.

Indicator type	Published count	Recommended action
Custom Windows beacon, shellcode, ELF, ARM64 & script SHA-256s	8	Hunt across Windows, Linux `x86_64`, and Linux `ARM64` EDR
Feed-labeled Cobalt Strike TLS JARM fingerprints	4	TLS hunting only; never block on a JARM fingerprint alone
Staged URI and filename patterns (text-only below)	8	Proxy, NDR, and command-line hunting
Feed-labeled Cobalt Strike cloud addresses (12 ports on 8 hosts)	0 (withheld)	Withheld pending independent beacon or protocol confirmation

Custom Implant and Script SHA-256 Digests to Hunt (5 Hashes)

SHA-256 digest	Staged filename	Size (B)	Platform & role
`8eecab05f564d2e23c876a7dea636b9a3581737ff21d648043d5d2a6a60add97`	`/beacon_dns_x64.exe`	167,424	Windows x64 DNS C2 Beacon executable (`WB-01`)
`4d207c3508e92547c093d81dd41e97bc1123dc3b4e4de8d8bdc0b1846b8bd7bf`	`/beacon_http_x64.dll`	103,936	Windows x64 HTTP C2 Beacon DLL (`WB-01`)
`4d2eedd82563b15f93ff9e931e8d3caa68e04f14d95495c439e951d9eeab7086`	`/beacon_http_x64.exe`	104,448	Windows x64 HTTP C2 Beacon executable (`WB-01`)
`2773e2b5c49544f6c1e33c4c75e9b1320e78d8874ee0f4a046ce79ceb034de2b`	`/root_pwn.sh`	8,030	Linux root privilege-escalation shell script (`WB-02`)
`9c2ef0205fe3a98208b44879a67e14007ac4c956acfe6a784c400d0928b8ff3b`	`/ws_linux_arm64`	10,816	Linux `ARM64` WebSocket C2 stager (`WB-02`)

TLS JARM Fingerprints to Hunt (4 Fingerprints)

JARM fingerprint	Observed on	Collision warning
`27d40d40d00040d00042d43d0000007d20e9b9a3e87aa1ec17cb4d646ce9a5`	Feed-labeled Cobalt Strike :443 host on Azure (`US`, address withheld)	Hunt only; combine with SNI, URI, or process activity
`1dd40d40d00040d0001dd40d1dd40dc7fe523f2d40ec58fa09eb4367e88094`	Feed-labeled Cobalt Strike :443 host on AWS (`US`, address withheld)	Hunt only; never block on JARM alone
`1dd40d40d00040d00042d43d000000e1ea2a807a629b496b664cf07ad7c08d`	Feed-labeled Cobalt Strike :443 host on AWS (`AU`, address withheld)	Hunt only; never block on JARM alone
`29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96`	Four feed-labeled AWS Cobalt Strike :443 servers (addresses withheld)	**High collision:** also seen on 2 other CS and 12 Metasploit AWS hosts

Text-Only Staging Path and Filename Hunt Patterns

Pattern	Context	Where to hunt
`/beacon_dns_x64.` & `/beacon_http_x64`	Dual-channel Windows Beacon DLL, EXE, service, and shellcode pulls	Suricata `SID 9541001`, proxy logs, Windows EDR
`/backoper_agent.elf` & `/agent_custom_tag.bin`	Custom 6,430,904 B Linux `x86_64` ELF implant names	Suricata `SID 9541002`, Linux `auditd` / EDR
`/gopher_linux_arm64.elf` & `/gopher_macos_arm64.macho`	Cross-compiled Go ARM64 server and Apple Silicon macOS implants	Suricata `SID 9541003`, cloud & macOS EDR
`/root_pwn.sh`	8,030 B Linux privilege-escalation shell script	Suricata `SID 9541004`, Linux command-line logs
`/ws_linux_arm64` & `/cs/ws.bin`	Linux `ARM64` WebSocket binary (10,816 B) and 2,100 B WebSocket stager	Linux EDR & outbound WebSocket proxy logs
`/rustdesk-1.5.1-x86_64.deb`	Legitimate `RustDesk` Debian installer staged alongside escalation scripts	Suricata `SID 9541005` (hunt), `dpkg` / `apt` logs
`/tcp_linux_i386`	2,607,220 B standalone 32-bit Linux TCP stager	Linux appliance & legacy server process logs
`/mimikatz/kiwi_passwords.yar`	Staged `Mimikatz` YARA file beside `fscan.exe` and `mimidrv.sys`	Suricata `SID 9541006`, YARA 3

Withheld: all four open-directory host addresses, all 137 C2 host addresses (153 endpoints, including the eight cloud hosts with 12 feed-labeled Cobalt Strike ports), custom certificate Common Names, and target-tagged filenames on Workbench A. Also withheld are all 98 remaining file SHA-256 digests from the 106-digest workbench corpus, including `/rustdesk-1.5.1-x86_64.deb`, public `Mimikatz` and `fscan.exe` releases, and single-binary watchlist files.

10

CHAPTER 10

Scope and Limits

IN SHORT

This chapter sets out what the evidence can show and what it cannot.

10 Scope and Limits

This edition covers **records dated 14–20 November 2026**. It examines four open HTTP directories (116 files, 106 unique SHA-256 digests) and 153 C2 endpoints (137 hosts) from 97,103 feed records. Open-directory findings rely strictly on static file metadata (paths, byte lengths, dates, and cryptographic digests). No binary, DLL, shellcode blob, or shell script was run or tested against external systems.

Zero shared IPv4 addresses or custom hashes link the four directories to each other or to the 137 C2 hosts. Readers should treat the cross-directory and C2 synthesis strictly as an analytical model rather than one intrusion campaign. Because the inventory records for the twelve feed-labeled Cobalt Strike ports do not establish independent beacon or protocol confirmation, all eight cloud host addresses remain withheld.

11

CHAPTER 11

Sources

IN SHORT

This chapter lists the public references used in the report.

11 Sources

1. **MITRE ATT&CK**, Software S0154: Cobalt Strike (HTTP/HTTPS and DNS Beacon Architecture): https://attack.mitre.org/software/S0154/ (verified 21 November 2026).
2. **MITRE ATT&CK**, Enterprise Sub-Technique T1071.004: Application Layer Protocol - DNS: https://attack.mitre.org/techniques/T1071/004/ (verified 21 November 2026).
3. **MITRE ATT&CK**, Enterprise Sub-Technique T1071.001: Application Layer Protocol - Web Protocols: https://attack.mitre.org/techniques/T1071/001/ (verified 21 November 2026).
4. **MITRE ATT&CK**, Enterprise Technique T1219: Remote Access Software: https://attack.mitre.org/techniques/T1219/ (verified 21 November 2026).
5. **RustDesk Open-Source Remote Desktop Project**, Official Client Source and x86_64 Debian Package Architecture: https://github.com/rustdesk/rustdesk (verified 21 November 2026).
6. **MITRE ATT&CK**, Software S0002: Mimikatz Credential Dumping Suite: https://attack.mitre.org/software/S0002/ (verified 21 November 2026).
7. **MITRE ATT&CK**, Software S0633: Sliver Cross-Platform Go Command-and-Control Framework: https://attack.mitre.org/software/S0633/ (verified 21 November 2026).
8. **Salesforce Engineering**, JARM: Active Transport Layer Security Server Fingerprinting Specification: https://github.com/salesforce/jarm (verified 21 November 2026).

APPENDICES

Indicators and Detection Rules

FOR DEFENDERS

Appendix A lists the indicators published for this study. Appendix B carries every published detection rule in full.

INSIDE

A. Indicators of compromise (9) 35

B. Detection rules (4 files) 36

A Indicators of Compromise

Values are defanged for display. The CSV, JSON and STIX files on the online edition carry the raw values. Values the report marks as withheld are not listed.

ID	Type	Indicator	Port	Context	Action
R41-IOC-01	SHA-256	8eecab05f564d2e23c876a7dea636b9a3581737ff21d648043d5d2a6a60add97		beacon_dns_x64.exe · Windows x64 DNS C2 Beacon executable	HUNT
R41-IOC-02	SHA-256	4d207c3508e92547c093d81dd41e97bc1123dc3b4e4de8d8bdc0b1846b8bd7bf		beacon_http_x64.dll · Windows x64 HTTP C2 Beacon DLL	HUNT
R41-IOC-03	SHA-256	4d2eedd82563b15f93ff9e931e8d3caa68e04f14d95495c439e951d9eeab7086		beacon_http_x64.exe · Windows x64 HTTP C2 Beacon executable	HUNT
R41-IOC-04	SHA-256	2773e2b5c49544f6c1e33c4c75e9b1320e78d8874ee0f4a046ce79ceb034de2b		root_pwn.sh · Linux privilege-escalation shell script	HUNT
R41-IOC-05	SHA-256	9c2ef0205fe3a98208b44879a67e14007ac4c956acfe6a784c400d0928b8ff3b		ws_linux_arm64 · Linux ARM64 WebSocket C2 stager	HUNT
R41-IOC-06	JARM	27d40d40d00040d00042d43d0000007d20e9b9a3e87aa1ec17cb4d646ce9a5		Cobalt Strike TLS JARM (a withheld address:443, Microsoft Azure US)	HUNT
R41-IOC-07	JARM	1dd40d40d00040d0001dd40d1dd40dc7fe523f2d40ec58fa09eb4367e88094		Cobalt Strike TLS JARM (a withheld address:443, AWS US)	HUNT
R41-IOC-08	JARM	1dd40d40d00040d00042d43d000000e1ea2a807a629b496b664cf07ad7c08d		Cobalt Strike TLS JARM (a withheld address:443, AWS AU)	HUNT
R41-IOC-09	JARM	29d29d00029d29d00041d41d00041dcb3f8752a2d70d1dbf446fc46ab5df96		Cobalt Strike TLS JARM (4 confirmed AWS listeners across AU, DE, and JP)	HUNT

B Detection Rules

Each rule file loads in its own engine (YARA-X, Suricata 7, Kusto, pySigma). Tune thresholds and allow-lists to your environment before alerting or blocking. The same files can be downloaded from the online edition.

odin_dual_beacons_backoper_and_root_pwn_artifacts.yar

YARA

```
/*
   Odin by Brandefense - Monograph 41 / 46 Bounded YARA Detection Suite
   Covers dual-channel DNS/HTTP Beacon and backoper_agent staging manifests,
   Linux root_pwn.sh and ARM64 WebSocket persistence bundles, and Mimikatz/fscan
   lateral-movement staging kits.
   Classification: TLP:CLEAR
*/

rule ODIN_Dual_Channel_DNS_HTTP_Beacon_Staging_Manifest {
    meta:
        author = "Odin by Brandefense"
        monograph = "41 / 46"
        date = "2026-11-21"
        tlp = "CLEAR"
        description = "Detects staging manifests or directory listings pairing DNS and HTTP x64 Beacons with
backoper_agent.elf or custom agent builds"
    strings:
        $dns1 = "beacon_dns_x64.dll" ascii wide
        $dns2 = "beacon_dns_x64.exe" ascii wide
        $http1 = "beacon_http_x64.dll" ascii wide
        $http2 = "beacon_http_x64.exe" ascii wide
        $http3 = "beacon_http_x64_shellcode.bin" ascii wide
        $ag1 = "backoper_agent.elf" ascii wide
        $ag2 = "agent_custom_tag.bin" ascii wide
        $ag3 = "gopher_linux_arm64.elf" ascii wide
    condition:
        filesize < 256KB and (1 of ($dns*)) and (1 of ($http*)) and (1 of ($ag*))
}

rule ODIN_Linux_Root_Pwn_and_ARM64_WebSocket_Bundle {
    meta:
        author = "Odin by Brandefense"
        monograph = "41 / 46"
        date = "2026-11-21"
        tlp = "CLEAR"
        description = "Detects Linux privilege-escalation helper scripts or manifests pairing root_pwn.sh with
ARM64 WebSocket agents or RustDesk .deb staging"
    strings:
        $pwn = "root_pwn.sh" ascii wide
        $ws1 = "ws_linux_arm64" ascii wide
        $ws2 = "/cs/ws.bin" ascii wide
        $ag = "c_agent" ascii wide
        $rd = "rustdesk-1.5.1-x86_64.deb" ascii wide
        $sh1 = "#!/bin/" ascii
        $sh2 = "chmod +x" ascii
    condition:
        filesize < 256KB and ($pwn or ($sh1 and $sh2)) and (2 of ($ws1, $ws2, $ag, $rd))
}

rule ODIN_Mimikatz_Kiwi_Passwords_and_Lateral_Kit_Manifest {
    meta:
        author = "Odin by Brandefense"
        monograph = "41 / 46"
        date = "2026-11-21"
        tlp = "CLEAR"
        description = "Detects lateral-movement staging inventories pairing kiwi_passwords.yar with multi-arch
Mimikatz binaries and fscan.exe"
    strings:
        $yar = "kiwi_passwords.yar" ascii wide
        $fsc = "fscan.exe" ascii wide
```

```
        $mk1 = "mimikatz.exe" ascii wide
        $mk2 = "mimilib.dll" ascii wide
        $mk3 = "mimispool.dll" ascii wide
        $mk4 = "mimidrv.sys" ascii wide
    condition:
        filesize < 256KB and $yar and $fsc and (2 of ($mk*))
}
```

odin_dual_dns_http_beacons_backoper_and_rustdesk.rules

SURICATA

```
# Odin by Brandefense - Monograph 41 / 46 Suricata 7.x Detection Suite
# Coverage: Dual-channel DNS and HTTP Beacon downloads, backoper_agent.elf and custom
#           agent payloads, cross-platform Go gopher implants, Linux root_pwn.sh and
#           RustDesk .deb HTTP staging, and open-directory workbench listings
# Classification: TLP:CLEAR

alert http any any -> any any (msg:"ODIN [R41] HTTP Retrieval of Staged Dual-Channel DNS or HTTP Beacon Payload";
flow:established,to_server; http.method; bsize:3; content:"GET"; http.uri; bsize:>17; content:"/beacon_";
fast_pattern; pcre:"/\/beacon_(dns_x64\.(dll|exe)|http_(x64(_shellcode\.bin|_svc\.exe|\.(dll|exe))|x86\.exe)|
win_x64(_iat|_svc|_svc2|_svc_v1|_v1|_v2)?\.exe)(\?|$)/"; classtype:trojan-activity; sid:9541001; rev:1;
metadata:created_at 2026_11_21, deployment perimeter_and_east_west, tactic command_and_control;)

alert http any any -> any any (msg:"ODIN [R41] HTTP Retrieval of Staged backoper_agent ELF or Custom Agent
Binary"; flow:established,to_server; http.method; bsize:3; content:"GET"; http.uri; bsize:>7; content:"agent";
fast_pattern; pcre:"/\/(backoper_agent\.elf|agent_custom_tag\.bin|agent\.bin|agent\.x64\.exe|c_agent)(\?|$)/";
classtype:trojan-activity; sid:9541002; rev:1; metadata:created_at 2026_11_21, deployment
perimeter_and_east_west, tactic execution;)

alert http any any -> any any (msg:"ODIN [R41] HTTP Retrieval of Cross-Platform Go Gopher Implant Binary";
flow:established,to_server; http.method; bsize:3; content:"GET"; http.uri; bsize:>17; content:"/gopher_";
fast_pattern; pcre:"/\/gopher_(linux_(amd64|arm64)(\.elf|_c2)|kali_linux_amd64\.elf|(windows_(amd64|arm64)|
win_x64)\.exe|macos_(amd64|arm64)\.macho)(\?|$)/"; classtype:trojan-activity; sid:9541003; rev:1;
metadata:created_at 2026_11_21, deployment perimeter_and_east_west, tactic command_and_control;)

alert http any any -> any any (msg:"ODIN [R41] HTTP Retrieval of Linux root_pwn.sh Privilege-Escalation Script";
flow:established,to_server; http.method; bsize:3; content:"GET"; http.uri; bsize:>11; content:"/root_pwn.sh";
fast_pattern; classtype:attempted-admin; sid:9541004; rev:1; metadata:created_at 2026_11_21, deployment
perimeter_and_east_west, tactic privilege_escalation;)

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R41] HUNT Direct HTTP Retrieval of RustDesk x86_64
Debian Package"; flow:established,to_server; http.method; bsize:3; content:"GET"; http.uri; bsize:>25; content:"/
rustdesk-"; fast_pattern; content:"x86_64.deb"; classtype:policy-violation; sid:9541005; rev:1;
metadata:created_at 2026_11_21, deployment perimeter_egress, policy behavioral_hunt;)

alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ODIN [R41] Open HTTP Directory Listing Exposing Dual-Beacon
backoper or Root-Pwn Kit"; flow:established,to_client; http.stat_code; bsize:3; content:"200";
http.response_body; bsize:>48; content:"Directory listing for /"; fast_pattern; pcre:"/(beacon_dns_x64|
backoper_agent\.elf|root_pwn\.sh|ws_linux_arm64|tcp_linux_i386|kiwi_passwords\.yar)/"; classtype:attempted-recon;
sid:9541006; rev:1; metadata:created_at 2026_11_21, deployment perimeter, tactic resource_development;)
```

sentinel_dual_beacons_backoper_rustdesk_and_c2_hunt.kql

KQL (SENTINEL / DEFENDER XDR)

```
// Odin by Brandefense - Monograph 41 / 46 Microsoft Sentinel & Defender XDR Hunt
// Joins (fullouter) execution of the five published custom Beacon/ELF/ARM64/privesc SHA-256 hashes,
// command lines mentioning staged beacon/ELF/root_pwn.sh markers or a RustDesk .deb file,
// and outbound requests for staged payload URLs (returning process-only, network-only, and combined rows).
// Host allowlists hold exact device names only and never suppress a custom hash match.
// Classification: TLP:CLEAR

let Lookback = 7d;
let ApprovedRedTeamHosts = dynamic(["REDTEAM-BUILD-VPS-01", "SEC-VALIDATION-LNX-01", "SEC-VALIDATION-WIN-01"]);
let ApprovedRmmPilotHosts = dynamic(["IT-RUSTDESK-PILOT-01"]);
let PrimaryImplantHashes = dynamic([
    "8eecab05f564d2e23c876a7dea636b9a3581737ff21d648043d5d2a6a60add97",
```

```
    "4d207c3508e92547c093d81dd41e97bc1123dc3b4e4de8d8bdc0b1846b8bd7bf",
    "4d2eedd82563b15f93ff9e931e8d3caa68e04f14d95495c439e951d9eeab7086",
    "2773e2b5c49544f6c1e33c4c75e9b1320e78d8874ee0f4a046ce79ceb034de2b",
    "9c2ef0205fe3a98208b44879a67e14007ac4c956acfe6a784c400d0928b8ff3b"
]);
let StagedToolMarkers = dynamic([
    "beacon_dns_x64", "beacon_http_x64", "backoper_agent.elf", "agent_custom_tag.bin",
    "gopher_linux_amd64", "gopher_linux_arm64", "root_pwn.sh", "ws_linux_arm64", "tcp_linux_i386"
]);
let ProcSignals =
    DeviceProcessEvents
    | where Timestamp >= ago(Lookback)
    | where SHA256 in~ (PrimaryImplantHashes)
        or (not(DeviceName in~ (ApprovedRedTeamHosts))
            and (ProcessCommandLine has_any (StagedToolMarkers)
                 or (ProcessCommandLine has "rustdesk-" and ProcessCommandLine has ".deb" and not(DeviceName in~
(ApprovedRmmPilotHosts)))))
    | project ProcTime = Timestamp, DeviceId, DeviceName, AccountName, InitiatingProcessFileName, FileName,
ProcessCommandLine, SHA256;
let NetSignals =
    DeviceNetworkEvents
    | where Timestamp >= ago(Lookback)
    | where not(DeviceName in~ (ApprovedRedTeamHosts))
        and RemoteUrl has_any ("/beacon_dns_x64", "/beacon_http_x64", "/backoper_agent.elf", "/root_pwn.sh", "/
ws_linux_arm64", "/rustdesk-1.5.1-x86_64.deb")
    | summarize
        NetFirstSeen = min(Timestamp),
        NetLastSeen = max(Timestamp),
        SampleEndpoints = make_set(strcat(RemoteIP, ":", tostring(RemotePort)), 12),
        SampleUrls = make_set(RemoteUrl, 10),
        NetHits = count()
        by DeviceId, DeviceName;
ProcSignals
| join kind=fullouter (NetSignals) on DeviceId, DeviceName
| extend ResolvedDevice = coalesce(DeviceName, DeviceName1)
| extend ResponsePriority = case(
    SHA256 in~ (PrimaryImplantHashes), "ISOLATE_KNOWN_BEACON_OR_BACKOPER_IMPLANT_HASH",
    ProcessCommandLine has "rustdesk-" or tostring(SampleUrls) has "rustdesk-1.5.1-x86_64.deb",
"HUNT_UNAUTHORIZED_RUSTDESK_DEB_PERSISTENCE",
    "HUNT_STAGED_BEACON_ELF_OR_ROOT_PWN_ACTIVITY"
)
| project ProcTime, NetFirstSeen, NetLastSeen, ResolvedDevice, AccountName, ResponsePriority,
InitiatingProcessFileName, FileName, ProcessCommandLine, SHA256, SampleEndpoints, SampleUrls, NetHits
| sort by coalesce(ProcTime, NetLastSeen) desc
```

sigma_dual_beacons_backoper_root_pwn_and_rustdesk.yml

SIGMA

```
title: ODIN Linux backoper ELF ARM64 WebSocket root_pwn or Unauthorized RustDesk Staging
id: 6d41c92a-8b1e-4b70-9a41-01f4e8c94101
status: experimental
description: Detects Linux execution of staged backoper_agent.elf, ws_linux_arm64, Go gopher ELF agents,
root_pwn.sh privilege-escalation scripts, or ad-hoc Debian package installation of rustdesk-1.5.1-x86_64.deb
outside approved pilot or validation hosts.
author: Odin by Brandefense
date: 2026-11-21
tags:
  - attack.execution
  - attack.privilege-escalation
  - attack.command-and-control
  - attack.t1068
  - attack.t1219
logsource:
  category: process_creation
  product: linux
detection:
  selection_implant_exec:
    Image|endswith:
      - '/backoper_agent.elf'
```

```
      - '/ws_linux_arm64'
      - '/tcp_linux_i386'
      - '/gopher_linux_amd64.elf'
      - '/gopher_linux_arm64.elf'
  selection_privesc_cli:
    CommandLine|contains:
      - 'root_pwn.sh'
      - 'backoper_agent.elf'
      - 'ws_linux_arm64'
  selection_rustdesk_deb:
    Image|endswith:
      - '/dpkg'
      - '/apt'
      - '/apt-get'
    CommandLine|contains: 'rustdesk-1.5.1-x86_64.deb'
  filter_validation_hosts:
    ComputerName:
      - 'REDTEAM-BUILD-VPS-01'
      - 'SEC-VALIDATION-LNX-01'
      - 'IT-RUSTDESK-PILOT-01'
  condition: (selection_implant_exec or selection_privesc_cli or selection_rustdesk_deb) and not
filter_validation_hosts
falsepositives:
  - Approved Linux remote-desktop pilot installations on IT-RUSTDESK-PILOT-01
level: high
---
title: ODIN Windows Dual-Channel DNS or HTTP Beacon Executable and DLL Loader Execution
id: 6d41c92a-8b1e-4b70-9a41-01f4e8c94102
status: experimental
description: Detects execution of staged Windows DNS and HTTP Beacon executables or rundll32 and regsvr32 loading
beacon_dns_x64.dll or beacon_http_x64.dll outside approved security validation hosts.
author: Odin by Brandefense
date: 2026-11-21
tags:
  - attack.execution
  - attack.defense-evasion
  - attack.command-and-control
  - attack.t1071.001
  - attack.t1071.004
  - attack.t1218.011
logsource:
  category: process_creation
  product: windows
detection:
  selection_beacon_exe:
    Image|endswith:
      - '\beacon_dns_x64.exe'
      - '\beacon_http_x64.exe'
      - '\beacon_http_x64_svc.exe'
      - '\beacon_http_x86.exe'
      - '\beacon_win_x64_iat.exe'
      - '\gopher_windows_amd64.exe'
      - '\gopher_windows_arm64.exe'
  selection_beacon_dll:
    Image|endswith:
      - '\rundll32.exe'
      - '\regsvr32.exe'
    CommandLine|contains:
      - 'beacon_dns_x64.dll'
      - 'beacon_http_x64.dll'
  filter_validation_win:
    ComputerName:
      - 'SEC-VALIDATION-WIN-01'
  condition: (selection_beacon_exe or selection_beacon_dll) and not filter_validation_win
falsepositives:
  - Isolated Windows security validation runs on SEC-VALIDATION-WIN-01
level: high
```

ODIN

About Odin Research

Odin Research is the public research series of Odin by Brandefense. It documents adversary infrastructure seen on the internet and turns each finding into indicators and detection rules that defenders can use. Victims and third parties stay anonymous, addresses appear only for confirmed adversary infrastructure, and no working attack code is published.

Odin is the agentic analyst inside the Brandefense threat intelligence platform, which unifies Cyber Threat Intelligence, Digital Risk Protection and External Attack Surface Management.

Online edition, indicators and rules · brandefense.io/odin-research/reports/dual-dns-and-http-beacons/
Short Read · brandefense.io/odin-research/blog/inside-multi-transport-beacons-linux-arm64-agents-and-rustdesk-persistence/
Defense Kit · brandefense.io/odin-research/defense-kit/