ODIN

TLP:CLEAR

ODIN RESEARCH · MONOGRAPH 03 / 30 · INTRUSION WORKFLOWS

Fileless GitHub-C2 and Hybrid Lateral Movement

Inside a 1,824-Record Workbench

How one exposed staging server pairs memory-only Linux agents, Oracle database-link pivots, rival eviction and search-bot cloaking

Prepared November 2026
Full Report · Public edition · Version 1.1

1,824
Records exposed across five staging ports (1,171 unique paths)

99 / 94
Script and JSON configuration instances retrieved (94 unique SHA-256 digests)

3
Linux persistence hooks (systemd, cron, profile) gated by one advisory file lock

26
Core Oracle Database, RAC and E-Business Suite lateral scripts (23 Bash, 3 PHP)

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

How one exposed staging server pairs memory-only Linux agents, Oracle database-link pivots, rival eviction and search-bot cloaking

01	Executive Summary	04
02	One Server, Five Ports, Many Jobs	07
03	Memory-Only Linux Agents and Cloudflare R2 Delivery	11
04	Pivoting Through Oracle, Zimbra and Windows	15
05	Evicting Rivals and Cloaking for Search Bots	19
06	What the Evidence Can and Cannot Show	22
07	What Defenders Should Do	25
08	Detection Rules	27
09	Indicators	33
10	Scope and Limits	37
11	Sources	39
A	Indicators of compromise (34)	41
B	Detection rules (5 files)	43

IN SHORT

Between 13 and 16 November 2026, Odin by Brandefense observed an exposed staging server, 38.60.227[.]210, holding 1,824 records across five open ports. The scripts combine memory-only Linux agents, Cloudflare R2 delivery, Oracle database-link pivots, Windows SMB tooling, rival-malware eviction and search-bot cloaking. They show intent and tooling, not confirmed activity on targets. Hunt executable memory regions paired with temporary file locks, audit Oracle database links, and block the listed staging hosts.

01

CHAPTER 01

Executive Summary

IN SHORT

Between 13 and 16 November 2026, Odin by Brandefense observed an exposed staging server, 38.60.227[.]210, holding 1,824 records across five open ports. The scripts combine memory-only Linux agents, Cloudflare R2 delivery, Oracle database-link pivots, Windows SMB tooling, rival-malware eviction and search-bot cloaking. They show intent and tooling, not confirmed activity on targets. Hunt executable memory regions paired with temporary file locks, audit Oracle database links, and block the listed staging hosts.

INSIDE THIS CHAPTER

Three Decisions for Leaders 04

The Study in Numbers 05

01 Executive Summary

Most toolkits do one job. This one does several. The same server holds enterprise lateral-movement scripts for Oracle, Zimbra and Windows, and the tools that turn a breached server into income: cryptocurrency mining and search-engine spam. This report explains how each part works and what defenders should look for.

All findings come from scripts and configuration files on the attacker's own server. Victim domains, database schemas, internal addresses and secrets are withheld.

1. **Memory-only agents and kernel file locks are designed to evade Linux process scans.**
   Comments in the root bootstrapper describe a GitHub-named agent that deletes its own file, blanks its command line and poses as the system init binary. Three persistence hooks check whether it is alive through a file lock, not a process name. A separate launcher serves unprivileged accounts from a Cloudflare R2 bucket.

2. **Lateral scripts target Oracle database links, Zimbra servers and Windows SMB shares.**
   Thirty middleware scripts, 26 of them for Oracle Database, RAC and E-Business Suite, are written to run remote DDL over database links. Zimbra scripts keep or lock a configuration re-entry channel. Windows scripts copy files over SMB, add a hidden administrator and overwrite AnyDesk password hashes.

3. **Rival eviction and cloaking scripts are designed to keep hosts quiet and profitable.**
   A watchdog is configured to kill 13 rival process patterns and quarantine 11 rival file names every two minutes. It never edits web server configuration, so sites stay up. Three PHP implants serve spam only to search bots and selected mobile visitors.

Three Decisions for Leaders

1. **Correlate Linux executable memory with kernel file locks.** Process names are not enough. Alert when an anonymous executable memory region and a temporary advisory lock appear together, and exclude kernel workers, user service managers and non-executable shared memory.
2. **Audit Oracle database links and restrict middleware egress.** Inventory privileged database links. Alert on remote directory or Java creation in Unified Audit logs, and block unapproved cloud-bucket downloads from database and middleware servers.
3. **Inspect PHP prepend hooks, backup JVMs and night-time Windows tasks.** Replay web checks with search-bot headers. Hunt encoded PowerShell started by Java, hidden local accounts, and SYSTEM tasks running from a temporary folder.

The Study in Numbers

Number	What it is
1,824	Records exposed across five staging ports (1,171 unique paths)
99 / 94	Script and JSON configuration instances retrieved (94 unique SHA-256 digests)
3	Linux persistence hooks (systemd, cron, profile) gated by one advisory file lock
26	Core Oracle Database, RAC and E-Business Suite lateral scripts (23 Bash, 3 PHP)
11 / 13	Rival file name patterns quarantined, and rival process patterns killed
18:00 to 09:00	Local-time window in which a Windows script kills Task Manager
47	Published indicators: 30 script digests and 17 network, cloud, binary and pool indicators

As of 16 November 2026, none of six primary indicators appeared in five public threat-intelligence sources. Private commercial feeds and later indexing are outside that check.

> **How to use this report.**
>
> Security leaders can read Chapter 1, Chapter 6 and Chapter 9. Detection engineers should start with Chapters 7 and 8. CTI analysts and hunters will find the tooling detail in Chapters 2 to 5.

02

CHAPTER 02

One Server, Five Ports, Many Jobs

IN SHORT

The operator put lateral-movement scripts, callback endpoints, cloaking modules and mining configurations on one Linux VPS. Five HTTP ports serve files, and the scripts reference four more callback ports, a second check-in host and two cloud buckets.

INSIDE THIS CHAPTER

What the server held 07

What each port does 08

Why this toolkit does not fit one label 08

Binaries listed but not retrieved 08

02 One Server, Five Ports, Many Jobs

What the server held

The workbench at 38.60.227[.]210 exposed 1,824 directory records across five open HTTP ports. Port 9002 mirrors port 8000, accounting for 645 paths. After removing that mirror and 8 cross-port overlaps, 1,171 unique paths remain.

From ports 8000, 8082 and 8083, 99 script and JSON configuration instances were retrieved. They map to 94 unique SHA-256 digests. By type, they break down as 47 `.sh`, 16 `.php`, 13 `.json`, 13 `.ps1`, 8 `.py`, 1 `.bat` and 1 `.js`.

Some files appear on more than one port. The table lists them.

File	Copies	Ports	Notes
`/install_user.sh`	3	8000, 8082, 8083	425-line R2 variants on 8000 and 8082; 327-line variant on 8083
`/config.json`	3	8000, 8082, 8083	Port-specific pool and thread settings
`/javas`	3	8000, 8082, 8083	ELF payload, listed but not retrieved
`/inst_nb.ps1`	2	8000, 8082	Differ only in `max-threads-hint` (50 vs. 60)
`/miner_restart_verify.sh`	2	8000, 8083	Identical copies

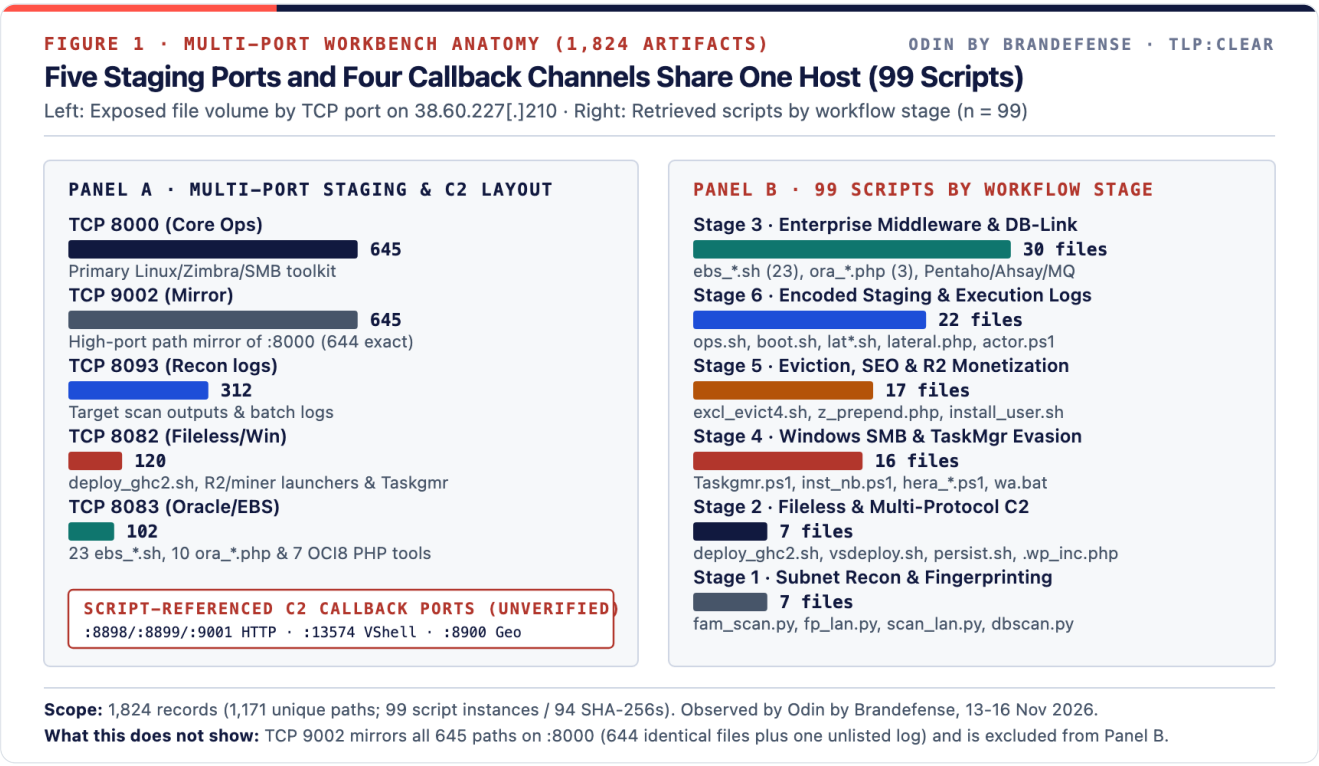


Figure 1: Multi-port staging layout, script-referenced callback ports and retrieved script breakdown

What each port does

Matching script download URLs against directory contents shows a clear layout. Some ports serve files. Others appear only inside scripts as callback endpoints. Whether those callback ports were listening was not verified.

Port	Role	Files listed	What it holds
8000	Main staging root	645	Zimbra loaders (`vsdeploy.sh` , `ops.sh`), rival eviction (`excl_evict4.sh`), PHP cloakers (`z_prepend.php` , `wbi_v10.php`), Ahsay SMB pivot scripts
9002	Mirror of 8000	645	644 identical files plus one unlisted log; likely a fallback if port 8000 is blocked
8093	Recon and log store	312	Scan outputs, target liveness lists, encoded execution logs
8082	Fileless and Windows staging	120	`deploy_ghc2.sh` , R2 and `/miner/` launchers, `.wp_inc.php` , `tj.js` , Windows SMB and AnyDesk tools (`Taskgmr.ps1` , `hera_*.ps1`)
8083	Oracle and PHP relays	102	23 `ebs_*.sh` scripts, 10 `ora_*.php` files, 7 more OCI8 PHP tools, Pentaho loaders, HTTP PUT servers (`put_srv*.py`)
8898, 8899, 9001, 13574	Callbacks named in scripts	Not verified	45-second HTTP polling (8898), eviction and recon uploads (8899), pull-test callbacks (9001), VShell sessions (13574)

A second host, 38.60.252[.]110, receives `/GEO_TOP_<host>` check-ins on port 8900. Its reachability was also not verified.

Why this toolkit does not fit one label

Threat models often separate enterprise intrusion groups from opportunistic miners. This server holds both kinds of tooling. The scripts are written to reach high-CPU database and middleware servers through Oracle database links, Zimbra configuration changes, SMB and MSSQL execution, and lock-gated Linux persistence.

Companion modules are designed to earn money from the same hosts. They run mining at night and serve spam to search bots. The stage labels in Figure 1 reflect how files were grouped by port and campaign on the server.

Binaries listed but not retrieved

The directory listings also named 122 binary and archive files that were not retrieved. Their names show how the operator prepares privilege escalation and tunnelling.

Port	Staged files	Platform	Purpose
8082	`/github-c2-agent-dual` , `/vs.agent` , `/agent_l64`	Linux ELF	Main C2 and VShell agent binaries used by `deploy_ghc2.sh` , `vsdeploy.sh` and `boot.sh`
8082	`/cve3156/pwnkit.tar.gz` , `pwnkit2.tar.gz` , `pwnkit3.tar.gz` , `hax.tar.gz` , `/dirtyfrag`	Linux local root	Privilege-escalation archives for `sudo` and `polkit` , plus a kernel exploit binary
8082	`/.nd/NtApiDotNet.dll` , `/win/edge.exe`	Windows	.NET token and RPC library for the `PrintSpoofer` loaders, and a disguised Windows payload
8000	`/chisel_linux` , `/chisel_orig.exe` , `/chisel_upx.exe` , `/plink.exe` , `/vagent_win.exe`	Linux and Windows	Reverse SOCKS and port-forwarding tools to reach internal Oracle and SMB tiers

The `cve3156` folder name points at CVE-2021-3156 in `sudo`. The `pwnkit` archives point at CVE-2021-4034 in `polkit`. Both are long-patched, so unpatched hosts are the real risk.

CHAPTER 03

Memory-Only Linux Agents and Cloudflare R2 Delivery

IN SHORT

The root bootstrapper `deploy_ghc2.sh` carries the author's own notes on how its agent hides. The agent deletes its file, blanks its command line and fakes its executable path, and three hooks restart it only when a kernel file lock is free.

INSIDE THIS CHAPTER

The bootstrapper and its author's notes 11

How the agent hides 12

Three hooks gated by one file lock 12

Rootless launchers for service accounts 13

03 Memory-Only Linux Agents and Cloudflare R2 Delivery

The bootstrapper and its author's notes

The most carefully built Linux file is `deploy_ghc2.sh` on port 8082. It is 156 lines and 5,796 bytes, with SHA-256 `549d39ceceb57866c0ade97fa02897d29f3141034ff6a02732118324c4ca1ce7`. It runs as root, downloads a binary named `github-c2-agent-dual`, and guards it against process sweeps and broken downloads.

The binary itself was not retrieved. Its GitHub C2 channel rests on the deployment banner (`GitHub-C2 agent`) and the file name. No repository URL or token was found.

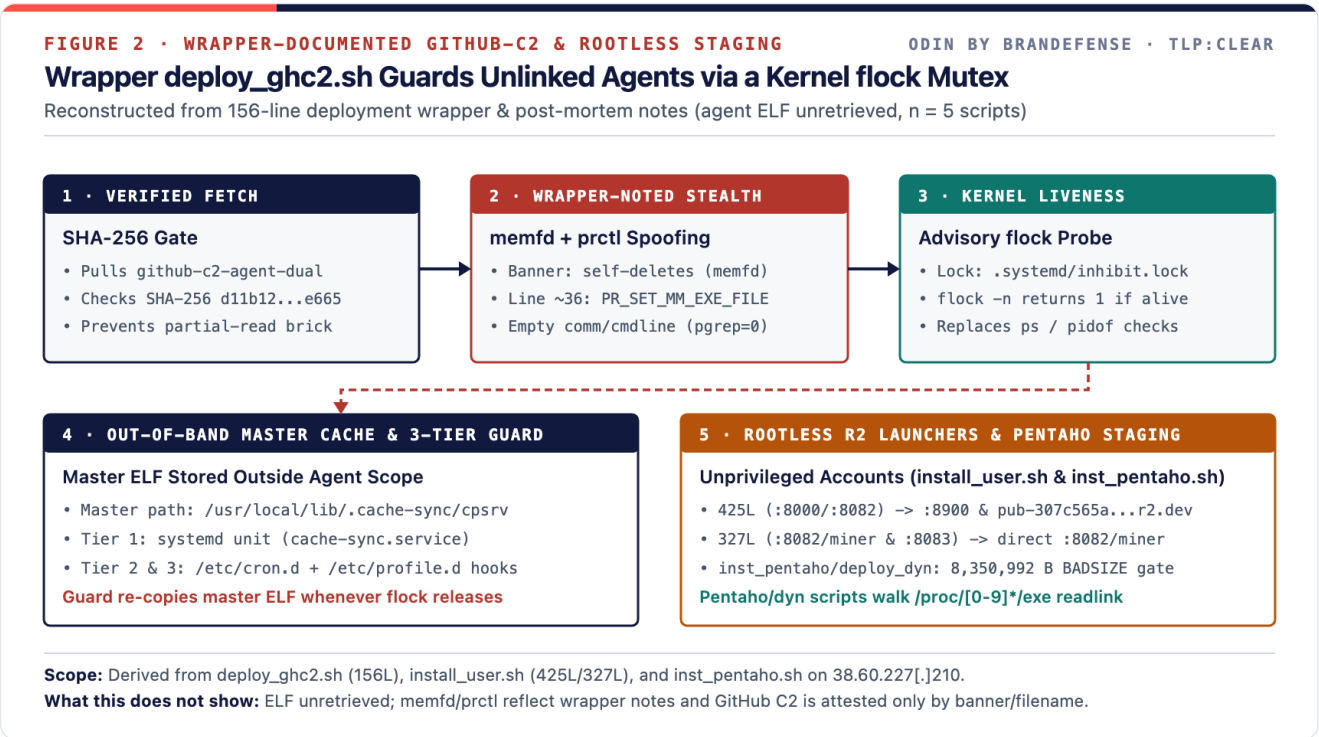


Figure 2: Fileless GitHub-C2 execution, kernel flock liveness and Cloudflare R2 fallback

The author left four numbered lessons in Chinese at the top of the script, headed "pitfalls already stepped in, do not alter". A fifth note sits around line 36. Translated, they say the following.

Note	What it says	Why it matters to defenders
1. Self-deleting agent	The agent deletes its binary at startup (`fileless/memfd`). A master copy must sit outside its scope so the guard can restore it.	The running process has no file on disk, but a master copy does.
2. Invisible to `pgrep`	The process has no `comm` name and an empty command line. The only reliable liveness check is `flock -n` on the lock file.	Process-name hunts miss it. Lock files do not.
3. Checksum before install	Downloads must pass a SHA-256 check with retries. A partial download once left a CentOS 7 host "permanently dark".	The enforced hash identifies the agent binary.
4. System-style names	All names pose as cache or system parts: `cache-sync`, `cpsrv`, `Cache sync helper`.	Search for these names in units, cron and profile scripts.
Line ~36	The agent sets `/proc/<pid>/exe` to `/usr/lib/systemd/systemd` with `PR_SET_MM_EXE_FILE`, so the script skips system folders when finding its working directory.	A `systemd` process that is not PID 1 is a red flag.

The script also records these defaults.

Default	Value
Payload URL	`http://38.60.227[.]210:8082/github-c2-agent-dual`
Enforced SHA-256 of the agent	`d11b12d628615d5eff982e3e312f4c406fde38373eee76bc056be780a9e2e665`
Master cache folder	`/usr/local/lib/.cache-sync/`
Kernel lock file	`/var/tmp/.systemd/inhibit.lock`

How the agent hides

According to the script, the agent removes its own ELF file and keeps running from an anonymous memory file (`memfd`). It clears its `comm` and `cmdline` strings, so `pgrep` finds nothing. It then calls `prctl` with `PR_SET_MM_EXE_FILE` to point `/proc/<pid>/exe` at `/usr/lib/systemd/systemd`.

The script's own filter skips `/memfd:*` and `/usr/*` paths when it works out the agent's folder. That confirms the author expected both tricks to be in place.

For hunters, this means `/proc/<pid>/exe -> memfd:` alone is not a reliable check. Correlate three signals instead:

- executable (`r-xp`) anonymous `memfd:` regions in `/proc/<pid>/maps`
- a `systemd` process that is not PID 1 and has no `--user` argument
- a lock inode in `/proc/locks` that matches a file under `/var/tmp`

This maps to MITRE ATT&CK T1620 (Reflective Code Loading). The operator-claimed GitHub channel would map to T1102.002 (Web Service: Bidirectional Communication).

Three hooks gated by one file lock

The running agent holds an exclusive advisory lock on `/var/tmp/.systemd/inhibit.lock`. Three persistence hooks are written to test that lock with `flock -n`:

Hook	Path
systemd unit	`/etc/systemd/system/cache-sync.service`
cron job	`/etc/cron.d/cache-sync`
login profile	`/etc/profile.d/cache-sync.sh`

If a hook can take the lock, the agent is dead. The hook then checks the master copy in `/usr/local/lib/.cache-sync/` against the hardcoded SHA-256 and restores it. This replaces `ps` and `pidof` checks, which the agent defeats.

Rootless launchers for service accounts

Initial access often lands in an unprivileged service account such as `oracle`, `zimbra`, `pentaho` or `apache`, with no `sudo` rights. For that case the toolkit ships rootless launchers that install a process named `java_update`.

Port and path	Lines / bytes	Delivery and checks
`8000:/install_user.sh`	425 / 16,351	Sends a check-in to `38.60.252[.]110:8900/GEO_TOP_<host>`; pulls `/javas` and `/config.json` from the R2 bucket `pub-307c565a487b43c5bee73d21edc4d83b.r2[.]dev/linux`
`8082:/install_user.sh` and `/linux/install_user.sh`	425 / 16,351	Port 8082 revision; same check-in; stages into `$HOME/.cache/java_update` with `WALLET=krxYMRN97D`
`8082:/miner/install_user.sh` and `8083:/install_user.sh`	327 / 12,157	Direct-IP variant; no :8900 check-in; pulls payloads from `38.60.227[.]210:8082/miner`
`8083:/inst_pentaho.sh` and `8083:/watch_pentaho.sh`	21 / 1,177 and 5 / 262	Stages `:8082/miner/javas` into `/opt/pentaho/.cache/java_update`; enforces an 8,350,992-byte size check
`8083:/deploy_dyn.sh`	17 / 1,359	Stages into `/tmp/.cache/java_update`; enforces the same size check plus ELF magic `7f454c46`

The 425-line launchers download from Cloudflare R2 by default. That makes payload downloads look like ordinary Cloudflare TLS traffic. This maps to T1105 (Ingress Tool Transfer).

The Pentaho and dynamic stagers check that the payload is exactly 8,350,992 bytes (8.35 MB). They then walk `/proc/[0-9]*/exe` links with `readlink` to avoid starting a second `java_update`. They do this without calling `ps`.

CHAPTER 04

Pivoting Through Oracle, Zimbra and Windows

IN SHORT

Thirty middleware scripts and companion Zimbra and Windows tools are written to move between Linux and Windows tiers. They target Oracle database links, Zimbra `swatchrc` files, Pentaho folders and Ahsay Backup web shells.

INSIDE THIS CHAPTER

Oracle E-Business Suite over database links 15

Zimbra `swatchrc` handling and `noexec` probing 16

From backup servers to Windows: Ahsay, SMB, AnyDesk and Task Manager 17

04 Pivoting Through Oracle, Zimbra and Windows

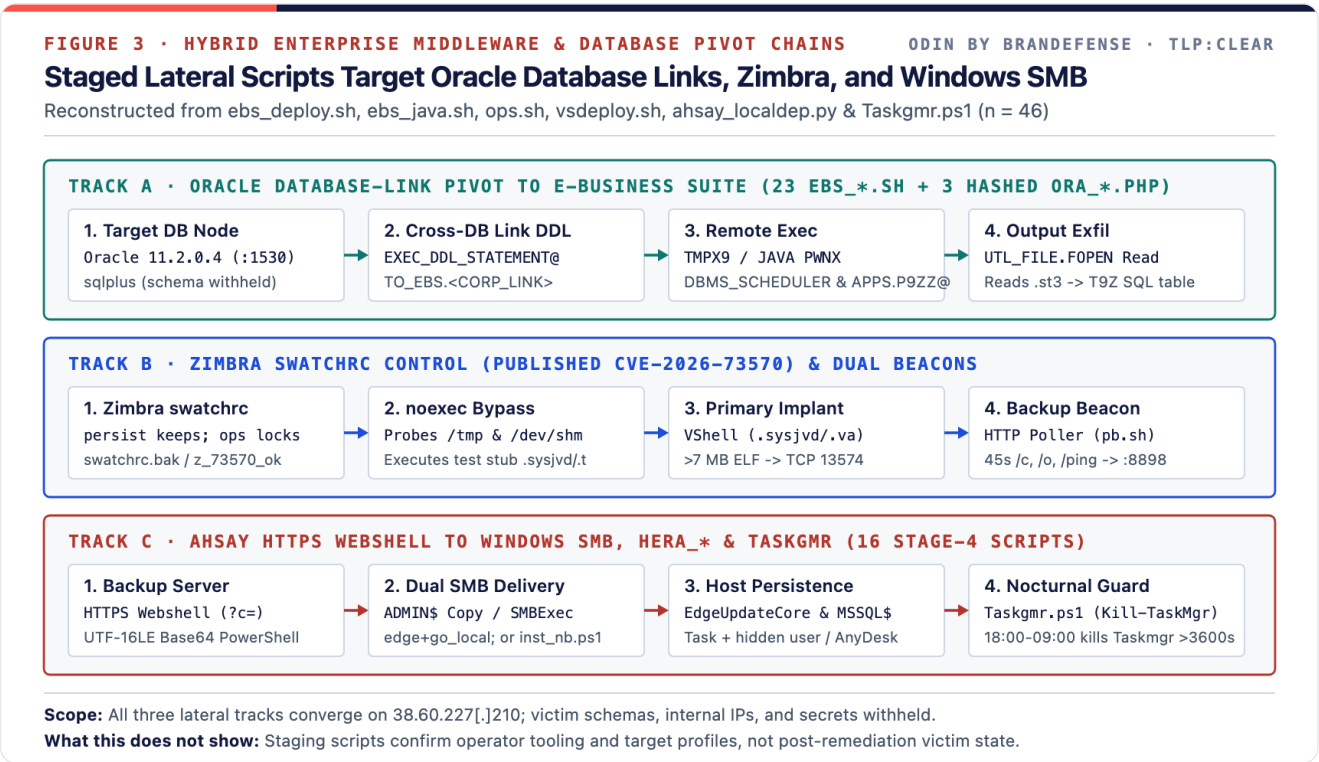


Figure 3: Cross-tier lateral movement chains across Oracle database links, Zimbra and Windows SMB

Oracle E-Business Suite over database links

Port 8083 holds 26 core Oracle lateral scripts and 7 supporting OCI8 PHP tools. They are written for Oracle Database `11.2.0.4`, Real Application Clusters (`ebs_rac.sh`) and Oracle E-Business Suite (EBS).

Script group	Listed / hashed	Key files	What they are written to do
`ebs_*.sh` database-link wrappers	23 / 23	`ebs_deploy.sh`, `ebs_rac.sh`, `ebs_java.sh`, `ebs_verify.sh`, `ebs_zz.sh`	Connect with `sqlplus` to an internal database on port 1530, then call `SYS.DBMS_UTILITY.EXEC_DDL_STATEMENT@` or `APPS.P9ZZ@` across the `TO_EBS` link
`ora_*.php` OCI8 deployers	10 / 3	`ora_fleet.php`, `ora_deploy.php`, `ora_rce.php`	Create directory `TMPX9`, schedule `/bin/sh` jobs with `DBMS_SCHEDULER.CREATE_JOB`, and read output with `UTL_FILE`
Supporting OCI8 and MySQL PHP tools	7 / 7	`dl_probe.php`, `dl_fleet.php`, `dump2.php`, `dump_users.php`, `implant.php`, `jobchk.php`, `lateral.php`	List database links and `session_privs@`, dump application user tables, schedule `WPOLL` command pollers, and pivot to internal web apps

The script `ebs_deploy.sh` (63 lines, 4,199 bytes) lays out a four-step remote sequence.

Step	Oracle call across `@TO_EBS`	Intended effect on the remote EBS database
1. Map a folder	`SYS.DBMS_UTILITY.EXEC_DDL_STATEMENT@TO_EBS`	Create directory `TMPX9` pointing at `/tmp`
2. Output table	`SYS.DBMS_UTILITY.EXEC_DDL_STATEMENT@TO_EBS`	Drop and recreate table `T9Z` to hold command output (schema withheld)
3. Read-back procedure	`SYS.DBMS_UTILITY.EXEC_DDL_STATEMENT@TO_EBS`	Compile procedure `P9Z`, which reads `/tmp/.st3` with `UTL_FILE.FOPEN` into `T9Z`
4. OS job	`SYS.DBMS_SCHEDULER.CREATE_JOB@TO_EBS`	Create `/bin/sh` job `JZEBS1` to fetch `:8082/miner/javas` into `/tmp/.cache/java_update` and write output to `/tmp/.st3`

> **Why database links bypass network segmentation.**
>
> Firewalls often isolate EBS servers but still allow SQL*Net (`TCP 1521/1530`) from neighbouring databases over pre-configured links. A call to `SYS.DBMS_UTILITY.EXEC_DDL_STATEMENT@<link>` runs privileged DDL on the remote instance. No direct socket to the EBS host is needed.

Two more scripts, `ebs_java.sh` and `ebs_zz.sh`, are written to compile Java classes `PWNX` and `PWNZ` on the remote side. PL/SQL wrappers `P9RUN` and `APPS.P9ZZ@` expose them for inline command execution. This maps to T1053 (Scheduled Task/Job).

Zimbra `swatchrc` handling and `noexec` probing

Seven Zimbra scripts on port 8000, including `persist.sh`, `vsdeploy.sh` and `ops.sh`, mention CVE-2026-73570 in comments or backup file names. NVD lists this as a Zimbra Collaboration Suite SNMP monitoring (`swatchrc` / `dosnmp`) command-injection flaw, published 13 August 2026 and fixed in Zimbra 10.1.20.

These are post-exploitation scripts, not exploit traffic. The CVE label alone does not show which initial access path was used on any target.

The scripts treat `/opt/zimbra/conf/swatchrc` in two opposite ways:

- `persist.sh` tells the operator to leave `swatchrc` alone, so the CVE-2026-73570 re-entry channel stays open.
- `ops.sh` is written to back up the file as `swatchrc.bak_cve73570` and rewrite the `dosnmp` entry with marker `z_73570_ok`, closing the channel to rival operators.

> **Testing for `noexec` mounts.**
>
> Hardened servers often mount `/tmp` with `noexec`. Before downloading anything, `vsdeploy.sh` loops through `/tmp`, `$HOME/.cache`, `/dev/shm` and `$HOME`. In each it creates a small test stub (`.sysjvd/.t`), marks it executable and tries to run it.

Once a folder passes, the script fetches a VShell binary of more than 7 MB (`.sysjvd/.va`), which calls back to `38.60.227[.]210:13574`. The companion `persist.sh` installs a 45-second `curl` poller (`pb.sh` in `.jvbd`). It checks `/c/`, `/o/` and `/ping/` on `38.60.227[.]210:8898`.

From backup servers to Windows: Ahsay, SMB, AnyDesk and Task Manager

To move from backup appliances into Windows subnets, the server stages 16 Windows scripts alongside Python and MSSQL wrappers. Four modules stand out.

Module	Port	How it is delivered	What it is written to do on Windows
`ahsay_localdep.py` and `go_local.ps1`	8000	HTTPS requests with certificate checks off (`ssl.CERT_NONE`) to a JSP web shell on an Ahsay Backup server, passing `?c=powershell -nop -enc ...`	Stage `edge.exe`, `go_local.ps1` and `config_template.json` in `C:\Windows\Temp\msedge`, mount neighbouring `IPC$` shares, and copy the files to `\\<ip>\ADMIN$\Temp\msedge\`
`inst_nb.ps1`	8000, 8082	`Invoke-SMBExec` sessions running as `SYSTEM`	Download `:8082/win/edge.exe`, write a plaintext Stratum `config.json` (`tls: false`) and register task `MicrosoftEdgeUpdateCore`
`actor.ps1`, `dec_hera*.ps1`, `hera_*.ps1`	8082	MSSQL `xp_cmdshell` into reflective .NET `PrintSpoofer` loaders (`dec_hera2.ps1`, `dec_hera3.ps1`)	`hera_adduser.ps1` creates a hidden local administrator `MSSQL$`; `hera_setpwd.ps1` overwrites `ad.anynet.pwd_hash` in `C:\ProgramData\AnyDesk\service.conf`
`win/Taskgmr.ps1` and `wa.bat`	8082, 8000	Pulled from port 8082 or an Alibaba Cloud OSS bucket (`imagefiles-backup.oss-ap-southeast-7.aliyuncs[.]com/javas/win/`)	Watch Task Manager and hide the miner from it, as described below

The hidden account works by setting `Winlogon\SpecialAccounts\UserList` to 0 for `MSSQL$`. That keeps it off the Windows logon screen. The AnyDesk change sets an operator-chosen password hash, which is withheld here. These map to T1136.001 (Create Local Account) and T1219 (Remote Access Software). The fake Edge update task maps to T1053.005 and T1036.005.

> **Task Manager watch (`Taskgmr.ps1`).**
>
> This 151-line script polls `Taskmgr` and `TaskmgrApp` every 3 seconds with `Get-Process` and CIM `Win32_Process`. During the day, it stops `MicrosoftEdgeUpdateSvc` whenever Task Manager opens. That hides the miner's CPU use from anyone who looks.

Between 18:00 and 09:00 local time, the script's "automatic Kill Window" applies. If Task Manager stays open longer than 3,600 seconds, a function named `Kill-TaskManager` closes it. The aim is long, quiet night-time mining runs.

05

CHAPTER 05

Evicting Rivals and Cloaking for Search Bots

IN SHORT

Internet-facing servers often host several competing intruders. A watchdog script is written to kill rival processes and quarantine their files every two minutes, while three PHP implants show spam only to search bots.

INSIDE THIS CHAPTER

A watchdog that avoids breaking the site 19

PHP prepend implants that cloak for search bots 20

05 Evicting Rivals and Cloaking for Search Bots



Figure 4: Rival-implant quarantine watchdog and conditional PHP search-bot cloaking

A watchdog that avoids breaking the site

Opportunistic botnets such as Kinsing (`kdevtmpfsi`) often overwrite crontabs and web configuration. The resulting outages bring in incident responders. The script `excl_evict4.sh` ("Exclusive Eviction v4", 7,153 bytes, 131 lines) takes the opposite approach. It is written around five rules.

Rule	What the script does
1. No web configuration edits	The header forbids writing to `.htaccess` or `nginx.conf`, so site routing keeps working
2. Quarantine, not delete	Foreign binaries and web shells are moved with `mv` into `/tmp/.excl_q`, so a wrongly matched web file can be recovered
3. Protect its own files	The sweep skips the operator's files by name (`.sysjvd`, `.va`, `java_update`, `z_73570`) and by content (`38.60.227[.]210`, :8899)
4. Match rival names and processes	11 file name patterns are quarantined; a 13-branch `grep -iE` sweep runs `kill -9` on matching processes
5. Repeat and report	Every 120 seconds it re-scans, checks its own implants are running (`selfok`) and posts a log to `38.60.227[.]210:8899`

The match lists are below.

Match type	Count	Patterns
File quarantine (`FOREIGNNAMES`)	11	`.khp`, `.rguard`, `.exectest`, `.java_cache`, `.app_args`, `.d`, `phpupdate`, `kinsing`, `kdevtmpfsi`, `.ksmd_`, `.UUID_NODEID`
Process kill (`kill -9`)	13	5 shared with the file list (`kinsing`, `kdevtmpfsi`, `.ksmd_`, `.khp`, `.rguard`) plus 8 miner and daemon names (`xmrig`, `minerd`, `kryptex`, `nicehash`, `xmr-stak`, `sysupdates`, `kworkerds`, `crond64`)

For defenders, a folder named `/tmp/.excl_q` full of other malware is a sign that a second intruder is on the host.

PHP prepend implants that cloak for search bots

On PHP servers, three implants hook the `auto_prepend_file` setting: `z_prepend.php`, `wbi_v10.php` and `.wp_inc.php`. A JavaScript redirector, `tj.js`, pairs with them. PHP runs the prepended file before the application on every dynamic request. No Nginx change is needed.

Each implant decides who sees spam. Administrators and direct visitors get the normal page.

Implant and spam feed	Who triggers it	What it is written to serve
`z_prepend.php` (`th8.jssnjc[.]com/feed.php`)	Search-bot user agents (`googlebot`, `bingbot`, `yandex`, `baiduspider`), or a mobile visitor with a target `Accept-Language` and a search-engine referrer	Spam HTML inline to bots (302 fallback); a `feed.php?js=1` redirect script to mobile search visitors
`wbi_v10.php` (`th12.jssnjc[.]com/feed.php`)	Bot user agents; debug switch `?xxe_debug=test123`	Checks an `X-Mode` header from the feed and renders the spam payload, skipping static files. It uses only anonymous closures and `$GLOBALS['_XP']='V10'`, so no named functions show up in PHP-FPM function lists
`.wp_inc.php` (`t1.sysflowus[.]com`, `th10.jssnjc[.]com`)	`googlebot` and `bingbot` only; the mobile and language branches were removed; kill switch `?xxe_off=off9x`	When the feed returns HTTP 200 and `X-Mode: inject`, buffers the page with `ob_start` and inserts spam links before `</body>`

Uptime monitors and homepage checks see the clean page. Only a request that looks like a search bot shows the spam. This maps to T1505.003 (Server Software Component).

CHAPTER 06

What the Evidence Can and Cannot Show

IN SHORT

The server's own scripts show what the operator built and intended. They do not show which targets were reached, whether the agent binary works as described, or who runs the server.

INSIDE THIS CHAPTER

Scripts versus binaries and victims 22

Four explanations for the server 22

Shared mining pools are not operator infrastructure 22

06 What the Evidence Can and Cannot Show

Scripts versus binaries and victims

The `memfd`, blank command line and `PR_SET_MM_EXE_FILE` behaviour come from the bootstrapper's comments. The agent binary was not retrieved, so these were not confirmed by disassembly. The GitHub C2 channel rests only on the banner and file name.

Lateral scripts such as `ebs_deploy.sh` and `ahsay_localdep.py` describe intended workflows. They do not show that the steps ran on any target, or what state any victim is in now. The SHA-256 values in Chapter 9 were recorded when the files were acquired.

Four explanations for the server

Because the evidence comes from a staging host and not from victims, we weighed four explanations. Without target-side logs or client authorization records, none can be fully excluded. This is an analytical model, not attribution.

Hypothesis	Fits	Does not fit	Cannot be excluded without target records
1. Unauthorized intrusion and monetization workbench	Oracle, Zimbra and Ahsay pivots paired with live pool accounts (`krxYMRN97D`), spam feeds (`*.jssnjc[.]com`) and rival eviction	Nothing in the staging evidence	Whether specific scripts ran on targets or stayed drafts
2. Authorized red-team exercise	Standard lateral tools (`DBMS_UTILITY.EXEC_DDL_STATEMENT@`, `Invoke-SMBExec`, `PrintSpoofer`, `chisel`, `plink`) and target-scoped scripts	Third-party spam feeds, rival quarantine, night-time mining configs, and a note about bricking a CentOS 7 host	Whether a tester copied real monetization scripts as realistic lures
3. Research lab or honeypot	1,824 records on five open directory listeners on one public VPS	Target-specific Oracle schema and link names, callback ports (8898, 8899, 13574) and custom checksum guards	Whether the server mirrors or sinkholes a captured toolkit
4. Open-source test fixture	Known privilege-escalation archives (`pwnkit`, `dirtyfrag`, `NtApiDotNet.dll`) and standard SQL and PowerShell templates	Custom markers (`z_73570_ok`, `TMPX9`, `P9ZZ`, `WALLET=krxYMRN97D`) and two-server callback wiring	Whether sub-modules came from unpublished training or shared toolkits

Hypothesis 1 fits all the staging evidence. The others each meet at least one conflict.

Shared mining pools are not operator infrastructure

The scripts name three bare IP addresses on port 8029, labelled `# pool egress (Kryptex naked IPs)` in `diag.sh`. They also name `xmr.kryptex[.]network:8029` and `gulf.moneroocean[.]stream:443`. These are shared public mining-pool front ends used by many unrelated miners. Treat them as policy signals, not as C2.

The operator's own infrastructure is different. It is the workbench (38.60.227[.]210), the check-in host (38.60.252[.]110:8900) and a custom TLS pool proxy (`wa.theshyc[.]xyz:5556`, which resolves to 38.54.14[.]162).

Of the 24 files that reference `krxYMRN97D`, most use encrypted Stratum. `8083:/config_t31.json` sets `tls: true`, and six Linux Oracle and Pentaho stagers use TLS Stratum on port 8029. Plaintext Stratum (`tls: false`) appears only in `8000:/config_template.json` and the two `inst_nb.ps1` copies. The other 14 files pass the account without a TLS flag.

Direct-IP TLS on port 8029 encrypts the login and sends no server name. Network rules alone will miss it, so pair them with a host-side search for `krxYMRN97D`. Likewise, blocking all `*.r2[.]dev` traffic would break developer tools. Tie R2 alerts to the process that made the request.

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

These steps cover the Linux agents, the Oracle, Zimbra and Windows pivots, and the rival-eviction scripts in Chapters 3 to 5, ordered by urgency.

Within 24 hours

- [ ] Block traffic to 38.60.227[.]210 (all listed ports), 38.60.252[.]110:8900 and `wa.theshyc[.]xyz:5556` (38.54.14[.]162).
- [ ] Block the R2 bucket `pub-307c565a487b43c5bee73d21edc4d83b.r2[.]dev`, the OSS bucket `imagefiles-backup.oss-ap-southeast-7.aliyuncs[.]com`, `*.jssnjc[.]com`, `t1.sysflowus[.]com` and `vjdghqegthea[.]cn[.]com`.
- [ ] Search Linux hosts for `/var/tmp/.systemd/inhibit.lock`, `/usr/local/lib/.cache-sync/`, `cache-sync.service`, `/tmp/.sysjvd/`, `/tmp/.jvbd/`, `/tmp/.excl_q/` and any `java_update` folder.

Within 72 hours

- [ ] On Linux, correlate executable `memfd:` mappings, `/proc/<pid>/exe -> /memfd:*` and non-PID-1 `systemd` processes with lock inodes in `/proc/locks`. Exclude `kthreadd` workers, `systemd --user` and non-executable shared memory.
- [ ] Turn on Oracle Unified Auditing of `SQL_TEXT`. Alert on `DBMS_UTILITY.EXEC_DDL_STATEMENT@`, `DBMS_SCHEDULER.CREATE_JOB@`, `TMPX9` and `COMPILE JAVA SOURCE`, and review `DBA_DB_LINKS`.
- [ ] Update Zimbra to 10.1.20 or later (CVE-2026-73570) and watch `/opt/zimbra/conf/swatchrc` for the `z_73570_ok` marker.
- [ ] Check `php.ini` and `.user.ini` for unexpected `auto_prepend_file` entries. Replay key pages with a Googlebot user agent and with mobile search headers.

Within 7 to 30 days

- [ ] Patch `sudo` (CVE-2021-3156) and `polkit` (CVE-2021-4034), and limit `SeImpersonatePrivilege` on Windows service accounts.
- [ ] Hunt `chisel` and `plink` reverse tunnels (`R:` arguments), encoded PowerShell started by Java, files in `C:\Windows\Temp\msedge\`, a hidden `MSSQL$` account, AnyDesk `service.conf` edits and `Taskgmr.ps1`.
- [ ] Limit egress from database and middleware servers to approved destinations. Pair port-8029 alerts with host searches for `krxYMRN97D`.
- [ ] Before keeping permanent IP blocks, re-check who owns the VPS addresses. Cloud addresses get reassigned.

CHAPTER 08

Detection Rules

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

INSIDE THIS CHAPTER

Linux auditd 28

Suricata 28

Microsoft Sentinel KQL 29

Sigma 29

YARA 30

Hunting in `/proc` 30

MITRE ATT&CK mapping 31

08 Detection Rules

The full rules are in Appendix B. Together they cover five layers: the Linux kernel, the Oracle database, the network perimeter, web and mail servers, and Windows endpoints.

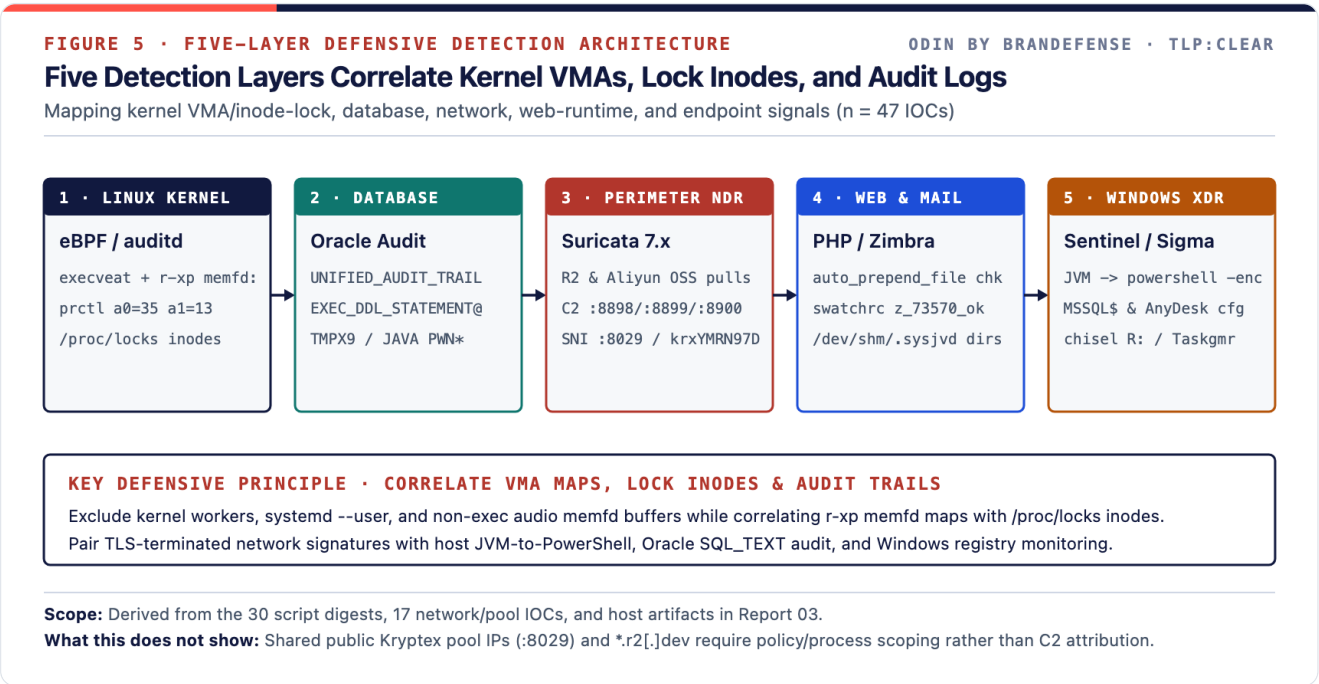


Figure 5: Five-layer detection architecture across Linux, Oracle, network, web and Windows

Rule file	What it catches
`linux_auditd_memfd_and_flock.rules`	Linux auditd: `memfd_create` with `execveat`, `prctl` executable spoofing, scoped `flock` calls, writes under `/var/tmp` and `/dev/shm`, and watches on the campaign's persistence paths
`suricata_r2_staging_and_c2_beacons.rules`	Suricata 7: 17 signatures for staging downloads, R2 and OSS buckets, C2 polling, cloaking feeds, web-shell PowerShell and Stratum traffic
`sentinel_cross_platform_fileless_and_smb_hunt.kql`	Microsoft Sentinel and Defender XDR: Linux and Windows process, registry, file and network hunt across the campaign's artifacts
`sigma_hybrid_fileless_dblink_and_taskgmr.yml`	Sigma (6 rules): `flock` liveness checks, Oracle database-link DDL, Java-spawned encoded PowerShell, fake Edge tasks, hidden `MSSQL$` and tunnels, AnyDesk and Task Manager script blocks
`yara_ghc2_eviction_and_prepend_implants.yar`	YARA (5 rules): the bootstrapper, the eviction watchdog, the PHP cloakers, Oracle pivot scripts and mining configs

Linux auditd

The auditd rules need kernel 3.17 or later, for `memfd_create` and `execveat`. They load paired 64-bit and 32-bit syscall rules before the file watches, and use `-i` so a missing optional folder does not stop loading.

Rule key	Looks for	Notes
`odin_r3_execveat`, `odin_r3_memfd_create`	Execution from an anonymous file descriptor	Correlate both on the same PID within 5 seconds; `pulseaudio`, `pipewire` and `systemd-journald` are excluded from `memfd_create`
`odin_r3_prctl_exe_masquerade`	`prctl` with `PR_SET_MM` and `PR_SET_MM_EXE_FILE`	Catches the `/proc/<pid>/exe` spoof
`odin_r3_flock_syscall`	`flock` calls outside systemd, cron, logrotate and dpkg	Scope alerts to lock files under `/var/tmp`, `/tmp` or `/dev/shm`
`odin_r3_vartmp_staging`, `odin_r3_shm_staging`	File creation and renames under `/var/tmp` and `/dev/shm`	On ARM64, drop legacy `creat` and `rename`
Path watches	`cache-sync.service`, `.cache-sync/`, `inhibit.lock`, `.sysjvd/`, `.jvbd/`, `.excl_q/`, `swatchrc`, the Pentaho `java_update` path, `/etc/cron.d/`, `/etc/profile.d/`	Deep paths arm only if the parent folder exists at load time

The rules deliberately do not filter out unset login IDs. That keeps service daemons such as `zimbra`, `oracle` and `tomcat` in scope. Container runtimes such as `runc` and `crun` may need an allowlist if they call `memfd_create`.

Suricata

Run these at perimeter egress and DMZ boundaries with the HTTP and TLS parsers on. SID 9302607 needs TLS termination in front of backup or middleware web servers, because `ahsay_localdep.py` uses HTTPS.

SIDs	Looks for
9302601, 9302602	Downloads of `/deploy_ghc2.sh` and `/github-c2-agent-dual`
9302603	TLS server name of the R2 payload bucket
9302604, 9302612, 9302613	`pb.sh` polling on port 8898: `GET /ping/<host>`, `GET /c/<host>`, `POST /o/<host>`
9302605	Any outbound connection to 38.60.227[.]210 on the nine workbench ports
9302614	`/GEO_TOP_` check-ins to 38.60.252[.]110:8900
9302606, 9302608, 9302609	TLS server names of the cloaking feeds and the `tj.js` redirector
9302607	Inbound `c=powershell` with `-enc` in a URL query (web-shell pivot)
9302610	Plain HTTP fetches of `/javas/win/` from the Alibaba OSS bucket (`wa.bat`)
9302611	TLS to `wa.theshyc[.]xyz` on port 5556
9302615	`krxYMRN97D` in plaintext Stratum on port 8029
9302616	Any connection to the three shared Kryptex pool IPs on port 8029 (policy signal)
9302617	TLS server name ending `kryptex[.]network` on port 8029 (policy signal)

Scope SIDs 9302616 and 9302617 to database, middleware and DMZ subnets. They fire on any Kryptex miner, so confirm with a host-side `krxYMRN97D` check before linking an alert to this operator.

Microsoft Sentinel KQL

The query looks back 14 days and joins `DeviceInfo` to label each host's platform. It has four branches.

Branch	Looks for
Process and persistence	Command lines and paths with `deploy_ghc2.sh`, `java_update`, `.cache-sync`, `.sysjvd`, `.jvbd`, `swatchrc.bak_cve73570`, `MicrosoftEdgeUpdateCore`, `Temp\msedge`, `krxYMRN97D`, the wallet prefix, `net user MSSQL$`, and `chisel` or `plink` with `R:`
Hidden account registry	Writes under `Winlogon\SpecialAccounts\UserList` naming `MSSQL$` or setting 0
AnyDesk tampering	Writes to `ProgramData\AnyDesk\service.conf` by any process other than `AnyDesk.exe`
Network	Connections to the workbench IPs, the shared pool IPs on 8029, exact campaign host names, the `jssnjc[.]com` and `vjdghqegthea[.]cn[.]com` suffixes, `kryptex[.]network`, and `*.r2[.]dev` requests from shells, `curl`, Python or PowerShell

Each network hit is labelled. Exact campaign hosts are direct hits. Related domain suffixes and R2 traffic from scripting tools are hunt leads, and pool traffic is a policy check. Allowlist approved developer R2 bucket names, not whole processes. If a management tool deploys AnyDesk configuration, allowlist that signed binary by path and hash.

Sigma

The six Sigma rules need Linux process logs, Oracle Unified Auditing with `SQL_TEXT`, Windows process creation (Security 4688 or Sysmon 1) and PowerShell Script Block Logging (4104).

Sigma rule	Fires when
Linux flock liveness and master cache	A `flock` command, or any command line, names `/var/tmp/.systemd/inhibit`, `.cache-sync`, `github-c2-agent-dual` or `deploy_ghc2.sh`
Oracle database-link DDL and scheduler pivot	`SQL_TEXT` contains `DBMS_UTILITY.EXEC_DDL_STATEMENT@`, `DBMS_SCHEDULER.CREATE_JOB@`, `CREATE OR REPLACE DIRECTORY TMPX9`, `COMPILE JAVA SOURCE NAMED "PWN`, `APPS.P9ZZ@` or `P9RUN@`
Backup or Tomcat JVM spawning encoded PowerShell	`java.exe`, `javaw.exe`, `bJW.exe`, `cbssvc.exe` or a Tomcat service starts PowerShell with an encoded-command flag
Fake Edge update in Temp	`schtasks /create` with `MicrosoftEdgeUpdateCore` and `\Windows\Temp\msedge\`, or `edge.exe` running from that folder
Hidden `MSSQL$` and reverse tunnels	`net user MSSQL$ /add`, a `SpecialAccounts\UserList` change naming `MSSQL$`, or `chisel` in client mode with `R:`
AnyDesk hash or Task Manager loop in PowerShell	A script block writing `ad.anynet.pwd_hash=` to AnyDesk `service.conf`, or containing `MicrosoftEdgeUpdateSvc`, `TaskMgrMaxOpenDuration` and `KillStartTime` together

Routine `SELECT ... FROM dual@<link>` queries are common in Oracle. Keep the database rule scoped to the DDL, scheduler, directory and Java calls listed above. On backup appliances, allowlist signed vendor scripts by full path.

YARA

The five YARA rules scan web roots, temporary folders, `/usr/local/lib` and service-account homes. Size limits keep them off large database files.

YARA rule	Fires when
`ODIN_R3_Linux_Deploy_GitHubC2_Memfd_Guard`	At least three bootstrapper strings appear in a file under 50 KB
`ODIN_R3_Linux_Exclusive_Turf_Eviction_Watchdog`	At least three watchdog strings appear in a file under 60 KB
PHP prepend cloaker rule (`ODIN_R3_PHP_AutoPrepend_*`)	At least two cloaker strings appear in a PHP file under 30 KB
`ODIN_R3_Oracle_DBLink_DDL_Lateral_Pivot`	A high-confidence DDL or Java string appears, or enough supporting and staging strings appear together, in a file under 40 KB, so routine DBA checks do not fire
`ODIN_R3_Campaign_Monetization_Config_And_Stager`	A mining config under 200 KB names `krxYMRN97D` , the Monero wallet or the custom pool proxy

Hunting in `/proc`

Live checks need root or `CAP_SYS_PTRACE` . Link three signals per process: executable `memfd` mappings, `PR_SET_MM_EXE_FILE` spoofing and lock inodes in `/proc/locks` .

Treat as normal the known non-executable shared-memory users (`pulseaudio` , `pipewire` , `wayland` , `php-fpm` opcache) and post-upgrade daemons. Do not allowlist `/proc/<pid>/exe -> /memfd:*` by process name alone. An operator can pass a fake name such as `pulseaudio` to `memfd_create` .

MITRE ATT&CK mapping

Tactic	Technique	How the toolkit uses it	Detection
Defense Evasion	T1620 Reflective Code Loading	Agent runs from `memfd` after deleting its file	auditd `memfd_create` and `execveat`; Sigma flock rule
Command and Control	T1102.002 Web Service: Bidirectional Communication	GitHub C2 channel, claimed by the banner and file name only	YARA bootstrapper rule; SIDs 9302601 and 9302602
Command and Control	T1105 Ingress Tool Transfer	Payloads from Cloudflare R2, Alibaba OSS and the workbench	SIDs 9302603, 9302605, 9302610; KQL network branch
Execution	T1053 Scheduled Task/Job	`DBMS_SCHEDULER.CREATE_JOB@` over database links	Sigma Oracle rule; YARA Oracle rule
Persistence	T1053.005 Scheduled Task	`MicrosoftEdgeUpdateCore` task from `Temp\msedge`	Sigma fake Edge rule; KQL process branch
Execution	T1059.001 PowerShell	Encoded PowerShell through the Ahsay web shell	SID 9302607; Sigma JVM rule
Persistence	T1505.003 Server Software Component	Ahsay JSP web shell and PHP prepend implants	Sigma JVM rule; YARA cloaker rule
Persistence	T1136.001 Create Account: Local Account	Hidden `MSSQL$` administrator	Sigma hidden account rule; KQL registry branch
Persistence	T1219 Remote Access Software	AnyDesk password hash overwrite	Sigma script block rule; KQL AnyDesk branch
Command and Control	T1572 Protocol Tunneling	`chisel` and `plink` reverse tunnels	Sigma tunnel selection; KQL process branch
Defense Evasion	T1036.005 Match Legitimate Name or Location	`MicrosoftEdgeUpdateCore`, `cache-sync`, `java_update`, fake `systemd` exe	Sigma rules; auditd path watches

09

CHAPTER 09

Indicators

IN SHORT

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

INSIDE THIS CHAPTER

Network, cloud and pool indicators 33

Key script hashes 34

Artifacts on target hosts 34

09 Indicators

The report publishes 47 indicators. Appendix A holds the 34 that suit automated feeds: the three operator addresses, the agent binary digest and the 30 script digests. The shared pool addresses, domains, buckets, pool account and wallet appear only in the table below.

Type	Count
Operator IPv4 addresses (workbench, check-in host, pool proxy)	3
Shared public mining-pool IPv4 addresses (policy signal, not C2)	3
Domains and cloud buckets	8
Agent binary SHA-256	1
Mining pool account and wallet	2
Script SHA-256 digests	30

Network, cloud and pool indicators

Indicator	Ports	Role
38.60.227[.]210	8000, 8082, 8083, 8093, 8898, 8899, 9001, 9002, 13574	Main staging server and callback host
38.60.252[.]110	8900	`/GEO_TOP_<host>` check-in host for the 425-line launchers
38.54.14[.]162	5556	TLS mining pool proxy behind `wa.theshyc[.]xyz`
135.125.145[.]1, 51.195.127[.]124, 40.160.29[.]14	8029	Shared public Kryptex pool front ends; policy signal only
`pub-307c565a487b43c5bee73d21edc4d83b.r2[.]dev`	443	Cloudflare R2 bucket serving Linux `java_update` payloads
`imagefiles-backup.oss-ap-southeast-7.aliyuncs[.]com`	80	Alibaba OSS bucket serving `Taskgmr.ps1`, `edge.exe` and `WinRing0x64.sys`
`th8.jssnjc[.]com`, `th10.jssnjc[.]com`, `th12.jssnjc[.]com`	443	Spam and cloaking feeds for the PHP implants
`t1.sysflowus[.]com`	443	Link-injection feed for `.wp_inc.php`
`ifwetrtjyudfwtuytxdfarxtyt.vjdghqegthea[.]cn[.]com`	443	JavaScript redirector in `tj.js`
`wa.theshyc[.]xyz`	5556	TLS mining pool proxy host name
`d11b12d628615d5eff982e3e312f4c406fde38373eee76bc056be780a9e2e665`	n/a	SHA-256 enforced for `github-c2-agent-dual`
`krxYMRN97D`	n/a	Kryptex pool account in launchers, `ebs_*.sh` and `inst_nb.ps1`

Indicator	Ports	Role
`89v9DbfLx49f6kuziDnExs6BHGyVV2UanFvMhcAU386sDKtT8eBqAwfh sSmxPwYSJT1r5bQyRgbpGHY741w5gMnk4McHFL3.creativejs`	n/a	Monero wallet and worker name in `config.json` and `cfg511.json`

Key script hashes

Appendix A lists all 30 script digests with path, port, size and line count. The most useful for hunting on hosts are below.

File	Port	Bytes / lines	SHA-256
`/deploy_ghc2.sh`	8082	5,796 / 156	`549d39ceceb57866c0ade97fa02897d29f3141034ff6a02732118324c4ca1ce7`
`/install_user.sh`	8082	16,351 / 425	`73fe311d3323b6bbc9f97854e43eaf33ed45d3122b236675f68350b820640b01`
`/excl_evict4.sh`	8000	7,153 / 131	`2a271c5cbf9bcc000b914c6f1f23cabc168fb98ed1628c555a4e4c416a456cde`
`/vsdeploy.sh`	8000	1,756 / 46	`d441ac91d579ee400e188e36c678e7a543c24a69a7ce6a4aa290bb9c4098aeba`
`/persist.sh`	8000	1,617 / 47	`c9269bd3c242a96a9bbe4c61df6d8f3473a343e9bbf3a5a91159f0f6baa9bc91`
`/z_prepend.php`	8000	3,148 / 58	`dbf856873b8142dac2d29118d60532f6d7bdafd8eb8763e9e149b9ef46ec832d`
`/wbi_v10.php`	8000	3,655 / 74	`a7423702c637bd84d96d69d14b1bbefacebb3bfcbecac44b2990c5f59ab4f720`
`/.wp_inc.php`	8082	5,889 / 137	`dceb0d07a78ccecd98d7e3b3660a9aef82b580d9ffcbc44d5c0c40e45bdc0ca2`
`/win/Taskgmr.ps1`	8082	5,443 / 151	`481728a7c9c4c02be07051d9c1958d902ea6397ebb8952ab83944818e3d25d21`
`/ops.sh`	8000	3,308 / 72	`525b7b194bede5a194204bdb78816b0db64b75ff9593e6d2ab3a736da79f44f6`

Lateral orchestrators such as `ebs_deploy.sh` and `ahsay_localdep.py` run from the staging server or a foothold, not on the target. Hunt them through network traffic, database audit logs and the target artifacts below, rather than by file hash.

Artifacts on target hosts

Artifact	Platform	Source script	Meaning
`/var/tmp/.systemd/inhibit.lock` and `/usr/local/lib/.cache-sync/`	Linux	`deploy_ghc2.sh`	Liveness lock and master ELF cache, guarded by `cache-sync.service`
`/opt/zimbra/conf/swatchrc.bak_cve73570` and `z_73570_ok`	Linux (Zimbra)	`ops.sh`	Backup and lockout marker written when `swatchrc` is changed
Directory `TMPX9`, Java sources `PWNX` and `PWNZ`, function `APPS.P9ZZ`	Oracle DB / EBS	`ebs_deploy.sh`, `ebs_zz.sh`	Remote `/tmp` mapping and Java command wrappers
`C:\Windows\Temp\msedge\edge.exe` and task `MicrosoftEdgeUpdateCore`	Windows	`ahsay_localdep.py`, `inst_nb.ps1`	Disguised `SYSTEM` task and staging folder, watched by `Taskgmr.ps1`
`MSSQL$` with `SpecialAccounts\UserList` = 0	Windows	`hera_adduser.ps1`	Hidden local administrator
`ad.anynet.pwd_hash` in `C:\ProgramData\AnyDesk\service.conf`	Windows	`hera_setpwd.ps1`	Operator password hash written before AnyDesk restarts (value withheld)

Withheld: victim domains, database schema names, internal database addresses, application user names, the AnyDesk password hash and the agent's per-user install path.

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

- **Window.** The workbench was observed from 13 to 16 November 2026. The findings describe its contents in that window.
- **Staging is not execution.** The scripts show what the operator built and configured. We did not see any step run on a target system.
- **The agent binary was not retrieved.** Its hiding behaviour comes from the bootstrapper's comments. Its GitHub C2 channel rests on a banner and a file name.
- **Callback ports are unverified.** Ports 8898, 8899, 9001, 13574 and 8900 are named in scripts. Whether they were listening was not checked.
- **Who runs it is open.** An unauthorized intrusion workbench fits the evidence best, but a red team, a research mirror or a test fixture cannot be fully excluded.
- **Shared pools are not C2.** The three port-8029 pool addresses serve many unrelated miners.
- **Public sources are a snapshot.** The zero-hit result covers six indicators in five public sources as of 16 November 2026, not private feeds or later reports.
- **Rule testing.** Every published rule file loads without errors in its engine (Suricata 7.0.12, YARA-X, pySigma and Microsoft's Kusto.Language analyzer, checked on 20 November 2026). The rules were not replayed against live traffic, so test and tune them in your own environment before production use. The auditd rules were checked against auditctl syntax.

11

CHAPTER 11

Sources

IN SHORT

This chapter lists the public references used in the report.

11 Sources

1. Linux man-pages. `memfd_create(2)`. https://man7.org/linux/man-pages/man2/memfd_create.2.html (verified 18 November 2026)
2. Linux man-pages. `flock(2)`. https://man7.org/linux/man-pages/man2/flock.2.html (verified 18 November 2026)
3. Linux man-pages. `prctl(2)`, `PR_SET_MM_EXE_FILE`. https://man7.org/linux/man-pages/man2/prctl.2.html (verified 18 November 2026)
4. Oracle. Database PL/SQL Packages and Types Reference, `DBMS_UTILITY`. https://docs.oracle.com/en/database/oracle/oracle-database/19/arpls/DBMS_UTILITY.html (verified 18 November 2026)
5. NIST NVD. CVE-2026-73570 Detail. https://nvd.nist.gov/vuln/detail/CVE-2026-73570 (verified 18 November 2026)
6. Zimbra. Security Advisories (CVE-2026-73570, fixed in 10.1.20). https://wiki.zimbra.com/wiki/Zimbra_Security_Advisories (verified 18 November 2026)
7. PHP Manual. Core `php.ini` directives, `auto_prepend_file`. https://www.php.net/manual/en/ini.core.php#ini.auto-prepend-file (verified 18 November 2026)
8. MITRE ATT&CK. T1620 Reflective Code Loading. https://attack.mitre.org/techniques/T1620/ (verified 18 November 2026)
9. MITRE ATT&CK. T1102.002 Web Service: Bidirectional Communication. https://attack.mitre.org/techniques/T1102/002/ (verified 18 November 2026)
10. MITRE ATT&CK. T1105 Ingress Tool Transfer. https://attack.mitre.org/techniques/T1105/ (verified 18 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 (34) 41

B. Detection rules (5 files) 43

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
R03-IOC-01	SHA-256	549d39ceceb57866c0ade97fa02897d29f3141034ff6a02732118324c4ca1ce7	8082	deploy_ghc2.sh · Stage 2: Fileless & Multi-Protocol C2 Persistence	HUNT
R03-IOC-02	IPv4	38.60.227[.]210	8082	Main staging server and callback host (ports 8000, 8082, 8083, 8093, 8898, 8899, 9001, 9002, 13574)	BLOCK
R03-IOC-03	SHA-256	d11b12d628615d5eff982e3e312f4c406fde38373eee76bc056be780a9e2e665	8082	github-c2-agent-dual binary (digest enforced by deploy_ghc2.sh)	HUNT
R03-IOC-04	SHA-256	73fe311d3323b6bbc9f97854e43eaf33ed45d3122b236675f68350b820640b01	8082	install_user.sh · Stage 5: Rival Eviction, SEO Cloaking & R2 Monetization	HUNT
R03-IOC-05	IPv4	38.60.252[.]110	8900	Check-in host for the 425-line launchers	BLOCK
R03-IOC-06	SHA-256	2a271c5cbf9bcc000b914c6f1f23cabc168fb98ed1628c555a4e4c416a456cde	8000	excl_evict4.sh · Stage 5: Rival Eviction, SEO Cloaking & R2 Monetization	HUNT
R03-IOC-07	IPv4	38.54.14[.]162	5556	TLS mining pool proxy behind the operator's pool host name	BLOCK
R03-IOC-08	SHA-256	d441ac91d579ee400e188e36c678e7a543c24a69a7ce6a4aa290bb9c4098aeba	8000	vsdeploy.sh · Stage 2: Fileless & Multi-Protocol C2 Persistence	HUNT
R03-IOC-09	SHA-256	c9269bd3c242a96a9bbe4c61df6d8f3473a343e9bbf3a5a91159f0f6baa9bc91	8000	persist.sh · Stage 2: Fileless & Multi-Protocol C2 Persistence	HUNT
R03-IOC-10	SHA-256	dbf856873b8142dac2d29118d60532f6d7bdafd8eb8763e9e149b9ef46ec832d	8000	z_prepend.php · Stage 5: Rival Eviction, SEO Cloaking & R2 Monetization	HUNT
R03-IOC-11	SHA-256	af3a3f9702eda99e3fdb8a2bf63f77c52b3180f423d798d674420f18448f6dca	8000	ahsay_localdep.py · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-12	SHA-256	a44e898fa7796b8e86393398c088c78516a94fb2ee823a3c8ac3fc6ec96eaf95	8083	ora_fleet.php · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-13	SHA-256	dadd08b682d1d95b6a6c53494a179c0e34779ecdfb3aaa72bb357c87766f2501	8083	ora_deploy.php · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-14	SHA-256	36696147fc80a0657aa9259dd0bfce17fb1b0de42af4e0422b57cee9894504c3	8083	ora_rce.php · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-15	SHA-256	9bb5852cb65e3f9eb102f41f04062cc9c8a0d645e48735918208a08d5554eeb0	8083	ebs_deploy.sh · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-16	SHA-256	8d0823378cf7d6e78020dfe4fd6b9a69c1096be48368a7f91713b74e1a350028	8083	ebs_rac.sh · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-17	SHA-256	815ff1eff88646fa9a24700bdec32cf5142cce7b3419157dd9f56bee8f6041a4	8083	ebs_java.sh · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-18	SHA-256	109f2960e27fa901c1d1d28fb050fb0b0c575f8f6b740d792c1de8fb63c799e8	8083	ebs_verify.sh · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-19	SHA-256	860939d4eda603d22790507e27c085dd598a1d1f21f35dd06b54e0fe6b4cbc8f	8083	inst_pentaho.sh · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT
R03-IOC-20	SHA-256	867069a658f7f2efbeca4690918c7c830529b893221c66078e28c8c9d7311f72	8083	mq_probe.php · Stage 3: Enterprise Middleware & DB-Link Pivoting	HUNT

ID	Type	Indicator	Port	Context	Action
R03-IOC-21	SHA-256	0f890a3d7102f5a950627f44924e71456f14b9bb5c44616fe20ece7d3b17357e	8083	lateral.php · Stage 6: Encoded Staging & Execution Logs	HUNT
R03-IOC-22	SHA-256	0d927ad3490596db7f7a6d4883586ee26cd3726f23aa398ec59eff4b8afee6c5	8083	implant.php · Stage 2: Fileless & Multi-Protocol C2 Persistence	HUNT
R03-IOC-23	SHA-256	a7423702c637bd84d96d69d14b1bbefacebb3bfcbecac44b2990c5f59ab4f720	8000	wbi_v10.php · Stage 2: Fileless & Multi-Protocol C2 Persistence	HUNT
R03-IOC-24	SHA-256	dceb0d07a78ccecd98d7e3b3660a9aef82b580d9ffcbc44d5c0c40e45bdc0ca2	8082	.wp_inc.php · Stage 2: Fileless & Multi-Protocol C2 Persistence	HUNT
R03-IOC-25	SHA-256	481728a7c9c4c02be07051d9c1958d902ea6397ebb8952ab83944818e3d25d21	8082	Taskgmr.ps1 · Stage 4: Windows SMB Lateral & TaskMgr Evasion	HUNT
R03-IOC-26	SHA-256	f3d34077915d76b43b497d136a200a6aa705c5a6f9546d4ede7634c6551aeeab	8082	inst_nb.ps1 · Stage 4: Windows SMB Lateral & TaskMgr Evasion	HUNT
R03-IOC-27	SHA-256	97036e688d4932109229cce4e4832990184fcbc0135c2c4a404af826990fdebc	8082	actor.ps1 · Stage 6: Encoded Staging & Execution Logs	HUNT
R03-IOC-28	SHA-256	eb160e307d20a7586138902c5693cc69a6a03e7370e890310451a562540ab94f	8082	hera_adduser.ps1 · Stage 4: Windows SMB Lateral & TaskMgr Evasion	HUNT
R03-IOC-29	SHA-256	85f7f6cbd9db334213be08a3da2b76d15b1b5a8c86f8df62b4061ef4f0387854	8082	dec_hera2.ps1 · Stage 4: Windows SMB Lateral & TaskMgr Evasion	HUNT
R03-IOC-30	SHA-256	caaa8acedea1d9fbcc6802c03da6e50c93d9f8b6b9dbf56193a0a36312e5af82	8083	fam_scan.py · Stage 1: Subnet Recon & Target Fingerprinting	HUNT
R03-IOC-31	SHA-256	0a76b965f59b05458a64b75130fb0384f4b4faacea11e0338bf53087f576c3f9	8083	fp_lan.py · Stage 1: Subnet Recon & Target Fingerprinting	HUNT
R03-IOC-32	SHA-256	d143740b871dd26fd5cc0a8d8c4385f48d4a658eaafa1a35330638d77b12d462	8083	scan_lan.py · Stage 1: Subnet Recon & Target Fingerprinting	HUNT
R03-IOC-33	SHA-256	4d86e984d2a1a9079657171fd8b797c898ccc5b2f4ca709e4ddb72b38b5fb727	8083	dbscan.py · Stage 1: Subnet Recon & Target Fingerprinting	HUNT
R03-IOC-34	SHA-256	525b7b194bede5a194204bdb78816b0db64b75ff9593e6d2ab3a736da79f44f6	8000	ops.sh · Stage 6: Encoded Staging & Execution Logs	HUNT

B Detection Rules

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

linux_auditd_memfd_and_flock.rules

LINUX AUDITD

```
## Odin by Brandefense - Report #03 Linux auditd Ruleset (TLP:CLEAR)
## Targets kernel syscalls and staging paths of deploy_ghc2.sh, install_user.sh, vsdeploy.sh, ops.sh, and
excl_evict4.sh
## Note 1: Never filter auid!=4294967295 so service daemons (zimbra/oracle/tomcat) and systemd units are
captured.
## Note 2: Ignore missing optional role-specific paths (-i) and load architecture-paired syscall rules BEFORE
optional file watches.
## Note 3: Linux auditctl supports syscall argument filters ONLY on a0-a3; execveat's 5th argument
(flags=AT_EMPTY_PATH) cannot be filtered in auditctl, so capture all execveat calls (rare on servers) and
correlate with memfd_create (<5s, same PID).
## Note 4: On ARM64 (aarch64), omit legacy creat/rename and use -S openat,renameat,renameat2 on the staging
directory rules.
-i

# 1. Fileless execution via anonymous FD (execveat correlated with memfd_create within 5s on same PID)
-a always,exit -F arch=b64 -S execveat -k odin_r3_execveat
-a always,exit -F arch=b32 -S execveat -k odin_r3_execveat
-a always,exit -F arch=b64 -S memfd_create -F exe!=/usr/bin/pulseaudio -F exe!=/usr/bin/pipewire -F exe!=/usr/
lib/systemd/systemd-journald -k odin_r3_memfd_create
-a always,exit -F arch=b32 -S memfd_create -F exe!=/usr/bin/pulseaudio -F exe!=/usr/bin/pipewire -F exe!=/usr/
lib/systemd/systemd-journald -k odin_r3_memfd_create

# 2. Executable symlink spoofing via prctl(PR_SET_MM=35, PR_SET_MM_EXE_FILE=13)
#    (Optional high-volume hunt omitted by default: prctl a0=15 PR_SET_NAME fires on glibc pthread_setname_np;
argv wipe is userland memory write)
-a always,exit -F arch=b64 -S prctl -F a0=35 -F a1=13 -k odin_r3_prctl_exe_masquerade
-a always,exit -F arch=b32 -S prctl -F a0=35 -F a1=13 -k odin_r3_prctl_exe_masquerade

# 3. Scoped advisory flock(2) syscall (excluding high-volume package/init daemons) + tmpfs/var staging directory
creation/open
-a always,exit -F arch=b64 -S flock -F exe!=/usr/lib/systemd/systemd -F exe!=/usr/sbin/cron -F exe!=/usr/sbin/
logrotate -F exe!=/usr/bin/dpkg -k odin_r3_flock_syscall
-a always,exit -F arch=b32 -S flock -F exe!=/usr/lib/systemd/systemd -F exe!=/usr/sbin/cron -F exe!=/usr/sbin/
logrotate -F exe!=/usr/bin/dpkg -k odin_r3_flock_syscall
-a always,exit -F arch=b64 -S openat,creat,rename,renameat,renameat2 -F dir=/var/tmp -F perm=wa -k
odin_r3_vartmp_staging
-a always,exit -F arch=b32 -S openat,creat,rename,renameat,renameat2 -F dir=/var/tmp -F perm=wa -k
odin_r3_vartmp_staging
-a always,exit -F arch=b64 -S openat,creat,rename,renameat,renameat2 -F dir=/dev/shm -F perm=wa -k
odin_r3_shm_staging
-a always,exit -F arch=b32 -S openat,creat,rename,renameat,renameat2 -F dir=/dev/shm -F perm=wa -k
odin_r3_shm_staging

# 4. Persistence & optional role-specific watches (deep paths arm only if parent dir exists at load time; dir=/
var/tmp & dir=/dev/shm above catch initial creation)
-w /etc/cron.d/ -p wa -k odin_r3_cron_persistence
-w /etc/profile.d/ -p wa -k odin_r3_profile_persistence
-w /etc/systemd/system/cache-sync.service -p rwxa -k odin_r3_ghc2_systemd_unit
-w /usr/local/lib/.cache-sync/ -p rwxa -k odin_r3_ghc2_master_cache
-w /var/tmp/.systemd/inhibit.lock -p rwxa -k odin_r3_ghc2_flock_mutex
-w /tmp/.sysjvd/ -p rwxa -k odin_r3_vshell_staging
-w /tmp/.jvbd/ -p rwxa -k odin_r3_http_beacon
-w /tmp/.excl_q/ -p rwxa -k odin_r3_rival_eviction_quarantine
-w /opt/zimbra/conf/swatchrc -p wa -k odin_r3_zimbra_swatchrc_tamper
-w /opt/pentaho/.cache/java_update -p rwxa -k odin_r3_pentaho_java_update
```

sentinel_cross_platform_fileless_and_smb_hunt.kql

KQL (SENTINEL / DEFENDER XDR)

```
// Odin by Brandefense - Report #03 Microsoft Sentinel & Defender XDR Hunting Suite (TLP:CLEAR)
// Uses KQL substring `contains` on punctuated paths/URLs (avoiding word-token `has` misses) plus
DeviceFileEvents for AnyDesk tampering.

let Lookback = 14d;
let WorkbenchIPs = dynamic(["38.60.227.210", "38.60.252.110", "38.54.14.162"]);
let SharedKryptexPoolIPs = dynamic(["135.125.145.1", "51.195.127.124", "40.160.29.14"]);
let ExactCampaignHosts = dynamic([
    "pub-307c565a487b43c5bee73d21edc4d83b.r2.dev",
    "imagefiles-backup.oss-ap-southeast-7.aliyuncs.com",
    "th8.jssnjc.com", "th10.jssnjc.com", "th12.jssnjc.com",
    "t1.sysflowus.com", "ifwetrtjyudfwtuytxdfarxtyt.vjdghqegthea.cn.com", "wa.theshyc.xyz"
]);
let HostPlatforms = DeviceInfo
    | where Timestamp > ago(Lookback)
    | summarize arg_max(Timestamp, OSPlatform) by DeviceId;
union
(
    DeviceProcessEvents
    | where Timestamp > ago(Lookback)
    | where ProcessCommandLine contains "/var/tmp/.systemd/inhibit"
        or ProcessCommandLine contains "/usr/local/lib/.cache-sync"
        or ProcessCommandLine contains "deploy_ghc2"
        or ProcessCommandLine contains "github-c2-agent-dual"
        or ProcessCommandLine contains ".sysjvd/.va"
        or ProcessCommandLine contains "/tmp/.excl_q"
        or ProcessCommandLine contains "swatchrc.bak_cve73570"
        or ProcessCommandLine contains "MicrosoftEdgeUpdateCore"
        or ProcessCommandLine contains @"\Windows\Temp\msedge\edge.exe"
        or ProcessCommandLine contains @"SpecialAccounts\UserList"
        or ProcessCommandLine contains "ad.anynet.pwd_hash"
        or ProcessCommandLine contains "krxYMRN97D"
        or ProcessCommandLine contains "89v9DbfLx49f6kuz"
        or ProcessCommandLine matches regex @"(?i)\bnet1?(\.exe)?\s+(user|localgroup\s+\S+)\s+MSSQL\$(\s|$)"
        or (FileName in~ ("chisel", "chisel_linux", "chisel.exe", "chisel_upx.exe", "plink.exe") and
ProcessCommandLine contains " R:")
        or FolderPath contains "/memfd:"
        or FolderPath contains "/.cache/java_update"
        or FolderPath contains "/.sysjvd/"
        or FolderPath contains "/.jvbd/"
        or FolderPath contains "/.cache-sync/"
        or FolderPath contains @"\Windows\Temp\msedge"
    | join kind=leftouter HostPlatforms on DeviceId
    | extend DetectionSurface = "ProcessOrPersistenceAnomaly", RemoteHost = ""
    | project Timestamp, DeviceName, OSPlatform, DetectionSurface, RemoteHost, AccountName, FileName, FolderPath,
ProcessCommandLine, SHA256
),
(
    DeviceRegistryEvents
    | where Timestamp > ago(Lookback)
    | where RegistryKey contains @"Winlogon\SpecialAccounts\UserList"
        and (RegistryValueName =~ "MSSQL$" or RegistryValueData == "0")
    | join kind=leftouter HostPlatforms on DeviceId
    | extend DetectionSurface = "HiddenLocalAdminRegistry",
             RemoteHost = "",
             FolderPath = RegistryKey,
             ProcessCommandLine = strcat(RegistryValueName, "=", RegistryValueData)
    | project Timestamp, DeviceName, OSPlatform, DetectionSurface, RemoteHost, AccountName =
InitiatingProcessAccountName,
              FileName = InitiatingProcessFileName, FolderPath, ProcessCommandLine, SHA256 =
InitiatingProcessSHA256
),
(
    DeviceFileEvents
    | where Timestamp > ago(Lookback)
    | where FileName =~ "service.conf"
        and FolderPath contains @"ProgramData\AnyDesk"
    | join kind=leftouter HostPlatforms on DeviceId
```

```
    | extend DetectionSurface = "AnyDeskServiceConfTamper",
             RemoteHost = "",
             ProcessCommandLine = InitiatingProcessCommandLine
    | project Timestamp, DeviceName, OSPlatform, DetectionSurface, RemoteHost, AccountName =
InitiatingProcessAccountName,
             FileName = InitiatingProcessFileName, FolderPath, ProcessCommandLine, SHA256 =
InitiatingProcessSHA256
),
(
    DeviceNetworkEvents
    | where Timestamp > ago(Lookback)
    | extend RawUrlWithScheme = iff(RemoteUrl contains "://", RemoteUrl, strcat("https://", trim_start("/",
RemoteUrl)))
    | extend NormalizedHost = tolower(trim_end(@"\.$", tostring(parse_url(RawUrlWithScheme).Host)))
    | where RemoteIP in (WorkbenchIPs)
        or (RemoteIP in (SharedKryptexPoolIPs) and RemotePort == 8029)
        or NormalizedHost in (ExactCampaignHosts)
        or NormalizedHost endswith ".jssnjc.com"
        or NormalizedHost endswith ".vjdghqegthea.cn.com"
        or NormalizedHost endswith "kryptex.network"
        or (NormalizedHost endswith ".r2.dev" and InitiatingProcessFileName in~ ("curl", "wget", "bash", "sh",
"python3", "powershell.exe", "pwsh.exe"))
    | join kind=leftouter HostPlatforms on DeviceId
    | extend RemoteHost = iff(isnotempty(NormalizedHost), NormalizedHost, RemoteIP),
             DetectionSurface = case(
                 RemoteIP in (WorkbenchIPs) or NormalizedHost in (ExactCampaignHosts),
"VerifiedCampaignStagingOrC2",
                 NormalizedHost endswith ".jssnjc.com" or NormalizedHost endswith ".vjdghqegthea.cn.com",
"RelatedDomainSuffix_HuntReview",
                 RemoteIP in (SharedKryptexPoolIPs) or NormalizedHost endswith "kryptex.network",
"SharedMiningPoolEgress_PolicyCheck",
                 "CloudflareR2ScriptEgress_PolicyReview"
             )
    | project Timestamp, DeviceName, OSPlatform, DetectionSurface, RemoteHost, AccountName =
InitiatingProcessAccountName,
              FileName = InitiatingProcessFileName, FolderPath = InitiatingProcessFolderPath,
              ProcessCommandLine = InitiatingProcessCommandLine, SHA256 = InitiatingProcessSHA256
)
| order by Timestamp desc
```

sigma_hybrid_fileless_dblink_and_taskgmr.yml

SIGMA

```
title: Linux Fileless GitHub-C2 Advisory Flock Liveness Check and Master Cache
id: 7a319c14-5c4b-4e82-9a11-386022721001
status: experimental
description: Detects advisory flock checks or shell references against /var/tmp/.systemd/inhibit.lock or /usr/
local/lib/.cache-sync used by deploy_ghc2.sh three-tier resurrection guards.
author: Odin by Brandefense
date: 2026-11-16
logsource:
    product: linux
    category: process_creation
detection:
    selection_flock:
        Image|endswith: '/flock'
        CommandLine|contains:
            - '/var/tmp/.systemd/inhibit'
            - '.cache-sync'
    selection_dirs:
        CommandLine|contains:
            - '/var/tmp/.systemd/inhibit'
            - '/usr/local/lib/.cache-sync'
            - 'github-c2-agent-dual'
            - 'deploy_ghc2.sh'
    condition: selection_flock or selection_dirs
falsepositives:
    - Unlikely; standard systemd-inhibit uses D-Bus and /run/systemd/inhibit rather than /var/tmp/.systemd/
inhibit.lock
```

```
level: critical
tags:
    - attack.defense-evasion
    - attack.t1620
    - attack.persistence
---
title: Oracle Database-Link Dynamic DDL and Scheduler Pivot Execution
id: 7a319c14-5c4b-4e82-9a11-386022721004
status: experimental
description: Detects cross-database-link DBMS_UTILITY.EXEC_DDL_STATEMENT, remote DBMS_SCHEDULER OS job creation,
TMPX9 directory mapping, and Java OS wrappers in Oracle Unified Auditing (requires audit policy capturing
UNIFIED_AUDIT_TRAIL.SQL_TEXT).
author: Odin by Brandefense
date: 2026-11-16
logsource:
    category: database
    product: oracle
detection:
    selection_dblink_ddl:
        SQL_TEXT|contains:
            - 'DBMS_UTILITY.EXEC_DDL_STATEMENT@'
            - 'DBMS_SCHEDULER.CREATE_JOB@'
            - 'CREATE OR REPLACE DIRECTORY TMPX9'
            - 'COMPILE JAVA SOURCE NAMED "PWN'
            - 'APPS.P9ZZ@'
            - 'P9RUN@'
    condition: selection_dblink_ddl
fields:
    - DBUSERNAME
    - SQL_TEXT
    - ACTION_NAME
falsepositives:
    - Administrative DDL executed across database links via DBMS_UTILITY is exceedingly rare in production EBS
environments
level: critical
tags:
    - attack.lateral-movement
    - attack.execution
    - attack.t1053
---
title: Ahsay CBS or Middleware Tomcat JVM Spawning Encoded PowerShell
id: 7a319c14-5c4b-4e82-9a11-386022721005
status: experimental
description: Host-side detection for encrypted HTTPS JSP webshell commands (ahsay_localdep.py ?c=powershell -nop
-enc) covering bundled Ahsay CBS JVM (bJW.exe/java.exe) and PowerShell encoded flag variants (-e, -ec, -enc, -
encodedcommand).
author: Odin by Brandefense
date: 2026-11-16
logsource:
    product: windows
    category: process_creation
detection:
    selection_parent:
        ParentImage|endswith:
            - '\java.exe'
            - '\javaw.exe'
            - '\bJW.exe'
            - '\cbssvc.exe'
            - '\tomcat8.exe'
            - '\tomcat9.exe'
        Image|endswith:
            - '\powershell.exe'
            - '\pwsh.exe'
        CommandLine|re: '(?i)\s[-/]e(c|nc|ncodedcommand)?\s'
    condition: selection_parent
falsepositives:
    - Rare administrative backup post-scripts invoking Base64-encoded PowerShell directly from Java
level: critical
tags:
    - attack.persistence
    - attack.execution
    - attack.t1505.003
```

```
    - attack.t1059.001
---
title: Windows SMB Lateral Staging Masquerading as MicrosoftEdgeUpdateCore in Temp
id: 7a319c14-5c4b-4e82-9a11-386022721002
status: experimental
description: Detects SYSTEM scheduled task creation for MicrosoftEdgeUpdateCore pointing to C:
\Windows\Temp\msedge\edge.exe observed in inst_nb.ps1, go_local.ps1, and ahsay_localdep.py lateral movement.
author: Odin by Brandefense
date: 2026-11-16
logsource:
    product: windows
    category: process_creation
detection:
    selection_schtasks:
        Image|endswith: '\schtasks.exe'
        CommandLine|contains|all:
            - '/create'
            - 'MicrosoftEdgeUpdateCore'
            - '\Windows\Temp\msedge\'
    selection_exec:
        Image|endswith: '\Windows\Temp\msedge\edge.exe'
    condition: selection_schtasks or selection_exec
falsepositives:
    - Legitimate Microsoft Edge Update runs from Program Files (x86)\Microsoft\EdgeUpdate, never C:
\Windows\Temp\msedge
level: critical
tags:
    - attack.lateral-movement
    - attack.persistence
    - attack.t1053.005
---
title: Hidden MSSQL$ Local Administrator Creation and Reverse Tunnel Tooling
id: 7a319c14-5c4b-4e82-9a11-386022721006
status: experimental
description: Detects hera_adduser.ps1 creating hidden local account MSSQL$ and hiding it under Winlogon
SpecialAccounts UserList, or chisel/plink reverse-tunnel execution staged from workbench port 8000.
author: Odin by Brandefense
date: 2026-11-16
logsource:
    product: windows
    category: process_creation
detection:
    selection_net_user:
        Image|endswith:
            - '\net.exe'
            - '\net1.exe'
        CommandLine|contains|all:
            - 'MSSQL$'
            - '/add'
    selection_reg_hide:
        CommandLine|contains|all:
            - 'Winlogon\SpecialAccounts\UserList'
            - 'MSSQL$'
    selection_tunnel:
        CommandLine|contains|all:
            - 'client'
            - ' R:'
        Image|endswith:
            - '\chisel.exe'
            - '\chisel_upx.exe'
            - '\chisel_orig.exe'
    condition: selection_net_user or selection_reg_hide or selection_tunnel
falsepositives:
    - Legitimate SQL Server named-instance service administration uses sc.exe/net start MSSQL$INSTANCE, never net
user MSSQL$ /add
level: critical
tags:
    - attack.persistence
    - attack.defense-evasion
    - attack.command-and-control
    - attack.t1136.001
    - attack.t1572
```

```
---
title: PowerShell Script-Block Tampering with AnyDesk Password Hash or Nocturnal TaskMgr Loop
id: 7a319c14-5c4b-4e82-9a11-386022721003
status: experimental
description: Detects PowerShell EventID 4104 execution of hera_setpwd.ps1 (in-script Get-Content/Set-Content
overwriting ad.anynet.pwd_hash in AnyDesk service.conf) or Taskgmr.ps1 (nocturnal 18:00-09:00 Taskmgr.exe kill
loop).
author: Odin by Brandefense
date: 2026-11-16
logsource:
    product: windows
    category: ps_script
detection:
    selection_anydesk_setpwd:
        ScriptBlockText|contains|all:
            - 'AnyDesk\service.conf'
            - 'ad.anynet.pwd_hash='
    selection_taskgmr_loop:
        ScriptBlockText|contains|all:
            - 'MicrosoftEdgeUpdateSvc'
            - 'TaskMgrMaxOpenDuration'
            - 'KillStartTime'
    condition: selection_anydesk_setpwd or selection_taskgmr_loop
falsepositives:
    - None expected; AnyDesk manages service.conf natively inside AnyDesk.exe rather than PowerShell Set-Content
level: critical
tags:
    - attack.persistence
    - attack.defense-evasion
    - attack.t1219
    - attack.t1036.005
```

suricata_r2_staging_and_c2_beacons.rules

SURICATA

```
# Odin by Brandefense - Report #03 Suricata 7.x Network Detection Suite (TLP:CLEAR)

# 1 & 2. Fileless GitHub-C2 bootstrapper and dual-agent ELF fetch
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN-R3 Fileless GitHub-C2 Bootstrapper Fetch
(deploy_ghc2.sh)"; flow:established,to_server; http.uri; content:"/deploy_ghc2.sh"; fast_pattern;
classtype:trojan-activity; sid:9302601; rev:3;)
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN-R3 Fileless github-c2-agent-dual ELF Download";
flow:established,to_server; http.uri; content:"/github-c2-agent-dual"; fast_pattern; classtype:command-and-
control; sid:9302602; rev:3;)

# 3. Cloudflare R2 object-storage bucket observed hosting rootless Linux payloads
alert tls $HOME_NET any -> $EXTERNAL_NET 443 (msg:"ODIN-R3 TLS SNI to Malicious Cloudflare R2 Payload Bucket";
flow:established,to_server; tls.sni; content:"pub-307c565a487b43c5bee73d21edc4d83b.r2.dev"; fast_pattern; nocase;
classtype:trojan-activity; sid:9302603; rev:3;)

# 4, 12 & 13. persist.sh (pb.sh) 45-second HTTP C2 beacon channels (/ping/<host>, GET /c/<host>, POST /o/<host>
on TCP 8898)
alert http $HOME_NET any -> $EXTERNAL_NET 8898 (msg:"ODIN-R3 Lightweight Linux HTTP C2 Idle Poll (pb.sh /ping/)";
flow:established,to_server; http.method; content:"GET"; http.uri; content:"/ping/"; startswith; fast_pattern;
pcre:"/^\/ping\/[A-Za-z0-9._-]{1,40}$/"; classtype:command-and-control; sid:9302604; rev:3;)
alert http $HOME_NET any -> $EXTERNAL_NET 8898 (msg:"ODIN-R3 Lightweight Linux HTTP C2 Task Fetch (pb.sh /c/)";
flow:established,to_server; http.method; content:"GET"; http.uri; content:"/c/"; startswith; fast_pattern;
pcre:"/^\/c\/[A-Za-z0-9._-]{1,40}$/"; classtype:command-and-control; sid:9302612; rev:1;)
alert http $HOME_NET any -> $EXTERNAL_NET 8898 (msg:"ODIN-R3 Lightweight Linux HTTP C2 Output Exfil (pb.sh POST /
o/)"; flow:established,to_server; http.method; content:"POST"; http.uri; content:"/o/"; startswith; fast_pattern;
pcre:"/^\/o\/[A-Za-z0-9._-]{1,40}$/"; classtype:command-and-control; sid:9302613; rev:1;)

# 5 & 14. Exact observed IP/port pairings on primary workbench (38.60.227.210) and secondary GEO_TOP node
(38.60.252.110:8900)
alert tcp $HOME_NET any -> 38.60.227.210 [8000,8082,8083,8093,8898,8899,9001,9002,13574] (msg:"ODIN-R3 Outbound
Connection to Primary Multi-Port Intrusion Workbench"; flow:established,to_server; threshold:type limit, track
by_src, count 1, seconds 300; classtype:command-and-control; sid:9302605; rev:3;)
alert http $HOME_NET any -> 38.60.252.110 8900 (msg:"ODIN-R3 Rootless Launcher GEO_TOP Check-In Beacon
(38.60.252.110:8900)"; flow:established,to_server; http.uri; content:"/GEO_TOP_"; startswith; fast_pattern;
```

```
classtype:command-and-control; sid:9302614; rev:1;)

# 6, 8 & 9. PHP auto_prepend_file SEO cloaking feeds (.jssnjc.com, t1.sysflowus.com) and tj.js redirector
alert tls $HOME_NET any -> $EXTERNAL_NET 443 (msg:"ODIN-R3 Outbound TLS SNI to jssnjc.com SEO Cloaking Feed";
flow:established,to_server; tls.sni; content:".jssnjc.com"; endswith; fast_pattern; nocase; classtype:trojan-
activity; sid:9302606; rev:3;)
alert tls $HOME_NET any -> $EXTERNAL_NET 443 (msg:"ODIN-R3 Outbound TLS SNI to t1.sysflowus.com SEO Cloaking Feed
(.wp_inc.php)"; flow:established,to_server; tls.sni; content:"t1.sysflowus.com"; fast_pattern; nocase;
classtype:trojan-activity; sid:9302608; rev:2;)
alert tls $HOME_NET any -> $EXTERNAL_NET 443 (msg:"ODIN-R3 TLS SNI to tj.js Redirector Domain
(vjdghqegthea.cn.com)"; flow:established,to_server; tls.sni; content:".vjdghqegthea.cn.com"; endswith;
fast_pattern; nocase; classtype:trojan-activity; sid:9302609; rev:2;)

# 7. Ahsay webshell encoded PowerShell pivot (requires reverse-proxy/NDR TLS termination when ahsay_localdep.py
uses HTTPS)
alert http any any -> $HOME_NET any (msg:"ODIN-R3 Encoded PowerShell via Webshell Query Parameter (TLS-Terminated
HTTP)"; flow:established,to_server; http.uri; content:"c=powershell"; nocase; fast_pattern; content:"-enc";
nocase; distance:0; classtype:web-application-attack; sid:9302607; rev:3;)

# 10. Alibaba OSS Windows stager fetch (wa.bat over plaintext HTTP; Suricata 7 normalizes http.host lowercase so
omit nocase)
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN-R3 Windows Stager Fetching Payloads from Observed
Alibaba OSS Bucket (wa.bat)"; flow:established,to_server; http.host; content:"imagefiles-backup.oss-ap-
southeast-7.aliyuncs.com"; bsize:49; fast_pattern; http.uri; content:"/javas/win/"; startswith; classtype:trojan-
activity; sid:9302610; rev:2;)

# 11, 15, 16 & 17. Custom TLS proxy (wa.theshyc.xyz:5556), plaintext Stratum (tls:false inst_nb.ps1), TLS SNI
Stratum (tls:true EBS/Pentaho), and naked pool IP egress
#    Note: Direct-IP TLS connections to 135.125.145.1 / 51.195.127.124 / 40.160.29.14 omit SNI and are covered by
sid:9302616 plus host config YARA/KQL hunts.
alert tls $HOME_NET any -> $EXTERNAL_NET 5556 (msg:"ODIN-R3 TLS SNI to Operator Monetization Proxy Domain
(wa.theshyc.xyz)"; flow:established,to_server; tls.sni; content:"wa.theshyc.xyz"; fast_pattern; nocase;
classtype:policy-violation; sid:9302611; rev:2;)
alert tcp $HOME_NET any -> $EXTERNAL_NET 8029 (msg:"ODIN-R3 Plaintext Stratum Login krxYMRN97D (tls:false Windows
inst_nb.ps1 Variant)"; flow:established,to_server; content:"krxYMRN97D"; fast_pattern; classtype:trojan-activity;
sid:9302615; rev:2;)
alert tcp $HOME_NET any -> [135.125.145.1,51.195.127.124,40.160.29.14] 8029 (msg:"ODIN-R3 Server Egress to Shared
Public Kryptex Pool IP (Covers No-SNI TLS & Plaintext)"; flow:established,to_server; threshold:type limit, track
by_src, count 1, seconds 600; classtype:policy-violation; sid:9302616; rev:2;)
alert tls $HOME_NET any -> $EXTERNAL_NET 8029 (msg:"ODIN-R3 TLS Stratum SNI to Kryptex Pool (tls:true Enterprise
EBS/Pentaho Fleet)"; flow:established,to_server; tls.sni; content:"kryptex.network"; endswith; nocase;
fast_pattern; classtype:policy-violation; sid:9302617; rev:1;)
```

yara_ghc2_eviction_and_prepend_implants.yar

YARA

```
// Odin by Brandefense - Report #03 YARA Detection Ruleset (TLP:CLEAR)

rule ODIN_R3_Linux_Deploy_GitHubC2_Memfd_Guard {
    meta:
        author = "Odin by Brandefense"
        description = "Detects deploy_ghc2.sh fileless memfd GitHub-C2 bootstrapper and 3-tier guard script"
        classification = "TLP:CLEAR"
    strings:
        $s1 = "/var/tmp/.systemd/inhibit" ascii
        $s2 = "/usr/local/lib/.cache-sync" ascii
        $s3 = "github-c2-agent-dual" ascii
        $s4 = "flock -n" ascii
        $s5 = "PR_SET_MM_EXE_FILE" ascii
    condition:
        filesize < 50KB and 3 of ($s*)
}

rule ODIN_R3_Linux_Exclusive_Turf_Eviction_Watchdog {
    meta:
        author = "Odin by Brandefense"
        description = "Detects excl_evict4.sh rival-implant quarantine watchdog"
        classification = "TLP:CLEAR"
    strings:
```

```
        $w1 = "/tmp/.excl_q" ascii
        $w2 = "z_73570" ascii
        $w3 = "kinsing|kdevtmpfsi|.ksmd_" ascii
        $w4 = ".sysjvd/.va" ascii
    condition:
        filesize < 60KB and 3 of ($w*)
}

rule ODIN_R3_PHP_AutoPrepend_Crawler_SEO_Cloaker {
    meta:
        author = "Odin by Brandefense"
        description = "Detects z_prepend.php, wbi_v10.php, and .wp_inc.php rootless auto_prepend_file crawler SEO
cloakers"
        classification = "TLP:CLEAR"
    strings:
        $p1 = "define('Z_73570_SEO', 1);" ascii
        $p2 = "HTTP_ACCEPT_LANGUAGE" ascii
        $p3 = "/feed.php?js=1" ascii
        $p4 = "/feed.php?domain=%s&path=%s" ascii
        $p5 = "xxe_debug" ascii
        $p6 = "wl-main/_xxe_inject/jssnjc" ascii
        $p7 = "xxe_off" ascii
    condition:
        filesize < 30KB and 2 of ($p*)
}

rule ODIN_R3_Oracle_DBLink_DDL_Lateral_Pivot {
    meta:
        author = "Odin by Brandefense"
        description = "Detects Oracle DBMS_UTILITY.EXEC_DDL_STATEMENT, Java wrapper, and DBMS_SCHEDULER lateral
pivot scripts without firing on routine DBA queries"
        classification = "TLP:CLEAR"
    strings:
        $hi1 = "DBMS_UTILITY.EXEC_DDL_STATEMENT@" ascii nocase
        $hi2 = "CREATE OR REPLACE DIRECTORY TMPX9" ascii nocase
        $hi3 = "COMPILE JAVA SOURCE NAMED \"PWN" ascii nocase
        $hi4 = "APPS.P9ZZ@" ascii nocase
        $hi5 = "P9RUN" ascii nocase
        $hi6 = "JZEBS1" ascii
        $ax1 = "'TMPX9'" ascii nocase
        $ax2 = "WPOLL" ascii
        $ax3 = "session_privs@" ascii nocase
        $ax4 = "user_scheduler_job_run_details" ascii nocase
        $ax5 = "java_update" ascii nocase
        $ax6 = "krxYMRN97D" ascii nocase
        $st1 = "utl_file.fopen" ascii nocase
        $st2 = "DBMS_SCHEDULER.CREATE_JOB" ascii nocase
        $st3 = "gv$instance@" ascii nocase
        $st4 = "dual@TO_" ascii nocase
        $st5 = "dba_db_links" ascii nocase
    condition:
        filesize < 40KB and (1 of ($hi*) or (1 of ($ax*) and 2 of ($ax*, $st*)) or 3 of ($st*))
}

rule ODIN_R3_Campaign_Monetization_Config_And_Stager {
    meta:
        author = "Odin by Brandefense"
        description = "Host-side file hunt for TLS-wrapped Kryptex account krxYMRN97D, Monero wallet, and custom
pool proxy configs"
        classification = "TLP:CLEAR"
    strings:
        $m1 = "krxYMRN97D" ascii nocase
        $m2 = "89v9DbfLx49f6kuziDnExs6BHGyVV2UanFvMhcAU386sDKtT8eBqAwfhsSmxPwYSJT1r5bQyRgbpGHY741w5gMnk4McHFL3"
ascii
        $m3 = "wa.theshyc.xyz:5556" ascii nocase
    condition:
        filesize < 200KB and 1 of ($m*)
}
```

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/fileless-github-c2-and-hybrid-lateral-movement/
Short Read · brandefense.io/odin-research/blog/one-staging-server-two-businesses/
Defense Kit · brandefense.io/odin-research/defense-kit/