ODIN

TLP:CLEAR

ODIN RESEARCH · MONOGRAPH 31 / 30 · POST-EXPLOITATION

Numbered Task Folders, Potato Binaries and Chisel Tunnels

Inside Two Exposed Workbenches

How numbered task folders stage Windows privilege escalation, reverse SOCKS tunnels and .NET single-sign-on cipher analysis

Prepared November 2026
Full Report · Public edition · Version 1.1

438 / 415
Newly observed autonomous-offense endpoints, on 415 distinct hosts, across seven framework families

11 / 10
Endpoints with an upstream C2 flag on 10 dedicated IPv4 addresses (unvalidated; addresses withheld)

427
Suspected or candidate endpoints on 405 hosts; addresses withheld

6,203
Files exposed on Workbench A (26 numbered task folders observed in a 270-entry sample)

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

How numbered task folders stage Windows privilege escalation, reverse SOCKS tunnels and .NET single-sign-on cipher analysis

01	Executive Summary	04
02	Autonomous Offense Endpoints: 438 New, 11 Flagged as C2	07
03	Workbench A: Numbered Task Folders and Windows Impersonation Staging	11
04	Tunnels, Pivot Scripts and .NET Cipher Files	16
05	Workbench B: SQLMap Batch Runs and HTTP Listener Logs	19
06	Noise, False Positives and Other Explanations	22
07	What Defenders Should Do	25
08	Detection Rules	27
09	Indicators	30
10	Scope and Limits	32
11	Sources	34
A	Indicators of compromise (0)	36
B	Detection rules (4 files)	37

IN SHORT

From 18 to 20 November 2026, Odin by Brandefense recorded 438 newly observed endpoints of seven autonomous-offense frameworks. Eleven, on 10 addresses, carry an upstream command-and-control flag without independent per-address validation; all 10 addresses are withheld. Two exposed web directories held 8,173 files. A 270-entry sample from one includes 26 numbered task folders, a layout that fits subagent tooling, staging GodPotato, PrintSpoofer, Chisel tunnels and .NET single-sign-on cipher files. Audit `SeImpersonatePrivilege`, retire SMBv1, and inspect outbound `chisel-v3` WebSocket upgrades.

01

CHAPTER 01

Executive Summary

IN SHORT

From 18 to 20 November 2026, Odin by Brandefense recorded 438 newly observed endpoints of seven autonomous-offense frameworks. Eleven, on 10 addresses, carry an upstream command-and-control flag without independent per-address validation; all 10 addresses are withheld. Two exposed web directories held 8,173 files. A 270-entry sample from one includes 26 numbered task folders, a layout that fits subagent tooling, staging GodPotato, PrintSpoofer, Chisel tunnels and .NET single-sign-on cipher files. Audit `SeImpersonatePrivilege`, retire SMBv1, and inspect outbound `chisel-v3` WebSocket upgrades.

INSIDE THIS CHAPTER

Three Decisions for Leaders 04

The Study in Numbers 05

01 Executive Summary

Autonomous penetration-testing frameworks split a job into many small worker tasks. Each worker, or subagent, gets its own folder for scan output, captured tokens and the tools it needs next. When an operator serves that folder tree over plain HTTP, anyone can read how the swarm moves from a web bug to a Windows server. This report covers the wider framework population and two exposed workbenches whose layout fits this pattern.

1. **Eleven of 438 new autonomous-offense endpoints carry an upstream C2 flag on 10 addresses.** ARTEX accounts for 57.1% of the new endpoints, followed by HexStrikeAI and CyberStrikeAI. Eleven endpoints on 10 dedicated addresses carry an upstream C2 flag that the study records do not independently validate, and the other 427 are suspected or candidate endpoints. The addresses of all 438 endpoints are withheld.

2. **Byte-identical Windows impersonation tools appear at the root and in one sampled task folder.** In a 270-entry sample of Workbench A, GodPotato and PrintSpoofer binaries at the root match files inside one numbered task folder. Sibling task folders in the sample stage Chisel tunnels, Socat-named logs, an MS17-010 script and .NET single-sign-on (SSO) assemblies with a disassembly and password-cracking logs.

3. **A second host pairs SQL injection batch logs with large HTTP listener logs.** Workbench B holds five `sqlmap` logs (3.49 MB) next to 34.94 MB of HTTP listener logs across two web listeners. Nothing links the two workbenches, so we treat them as separate examples of the same staging habit.

Three Decisions for Leaders

1. **Require multi-signal proof before blocking framework addresses.** All 10 flagged C2 addresses are withheld pending per-address signal validation. Treat autonomous-agent hits as hunting leads until a certificate, JARM value and endpoint process agree.
2. **Harden Windows service accounts and watch WebSocket egress.** Limit which accounts hold the impersonation right, run IIS and SQL Server pools as low-privilege managed accounts, retire SMBv1 and inspect outbound WebSocket upgrades on unusual ports.
3. **Remove static application keys and parameterize queries.** Move hard-coded keys out of .NET SSO components into a managed secret store, enforce parameterized SQL and stop servers from pulling binaries over plain HTTP.

The Study in Numbers

Number	What it is
438 / 415	Newly observed autonomous-offense endpoints, on 415 distinct hosts, across seven framework families
11 / 10	Endpoints with an upstream C2 flag on 10 dedicated IPv4 addresses (unvalidated; addresses withheld)
427	Suspected or candidate endpoints on 405 hosts; addresses withheld
6,203	Files exposed on Workbench A (26 numbered task folders observed in a 270-entry sample)
1,970	Files exposed on Workbench B across three HTTP listeners (984 on the runner listener)
38.43 MB	HTTP listener logs (34.94 MB) plus `sqlmap` batch logs (3.49 MB) on Workbench B
0 / 4	Published IP or hash indicators in Appendix A (all withheld); 4 text-only hunting selectors in Chapter 9

How to use this report.

Security leaders can read Chapters 1, 6 and 7. Windows, Active Directory and application teams should read Chapters 3 to 5 and the checklist in Chapter 7. SOC analysts and detection engineers should start with Chapters 8 and 9. CTI analysts will find the framework population in Chapter 2.

02

CHAPTER 02

Autonomous Offense Endpoints: 438 New, 11 Flagged as C2

IN SHORT

Seven autonomous-offense frameworks account for 438 newly observed endpoints on 415 hosts. Eleven endpoints on 10 dedicated addresses carry an upstream C2 flag that the study records do not independently validate, so all 10 addresses are withheld. Hunt with the JARM and certificate selectors instead of blocking single banners.

INSIDE THIS CHAPTER

Seven families, one dominant framework 08

Where the endpoints sit 08

The 11 endpoints with an upstream C2 flag 08

02 Autonomous Offense Endpoints: 438 New, 11 Flagged as C2

In threat intelligence, a family label such as "ARTEX" or "CyberStrikeAI" describes a product fingerprint: a banner, an API route or a TLS certificate. It says nothing about who runs the server. Developers, consultants and intruders all deploy the same containers on the same cloud ranges, where cyber threat intelligence helps defenders separate hostile use from testing. So a single matching banner is not a reason to block an address.

We pair the 438 endpoints with two exposed HTTP directories holding 8,173 files. No shared IP, TLS certificate or file hash links Workbench A to Workbench B, so we analyse them as separate examples.

Nothing in either listing names an AI agent framework. No IP, certificate or hash links the workbenches to the 438 endpoints. We treat the 26 sampled numbered folders as subagent tasks only as an analytical model. Figure 1 shows how the parts fit together.

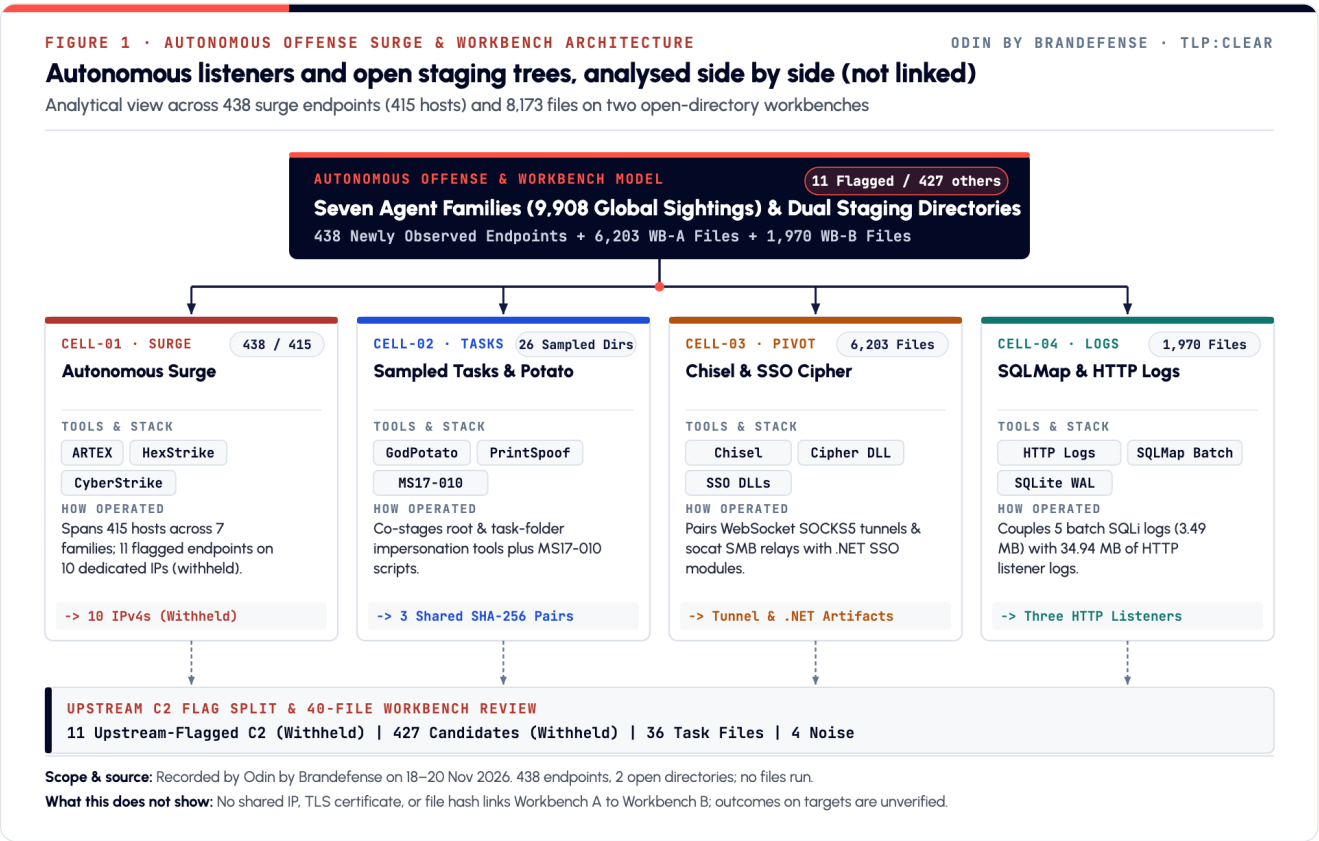


Figure 1: Autonomous listeners and open staging trees, analysed side by side (not linked)

Seven families, one dominant framework

Odin holds 9,908 endpoint records for the same seven families. ARTEX (4,597), CyberStrikeAI (3,425) and PentAGI (1,297) make up 94.1% of them. Among the 438 newly observed endpoints, ARTEX leads with 250 (57.1%).

Family	All records	New endpoints	Distinct hosts	Flagged C2	Other endpoints	Main ports
ARTEX	4,597 (46.4%)	250 (57.1%)	238	6 (5 IPs)	244	8787 (137), 8790 (28), 80 (21)
HexStrikeAI	268 (2.7%)	69 (15.8%)	66	0	69	80 (38), 5000 (11), 443 (8)
CyberStrikeAI	3,425 (34.6%)	62 (14.2%)	58	3 (3 IPs)	59	8080 (36), 443 (5), 8081 (3)
PentAGI	1,297 (13.1%)	39 (8.9%)	39	1 (1 IP)	38	8443 (31), 8666 (3), 8444 (1)
AutoHunter	101 (1.0%)	12 (2.7%)	12	0	12	18800 (9), 80 (1), 18801 (1)
Cairn	170 (1.7%)	5 (1.1%)	5	1 (1 IP)	4	8000 (4), 443 (1)
Strix	50 (0.5%)	1 (0.2%)	1	0	1	8080 (1)

The per-family host counts add up to 419, not 415. Four hosts run two families at once. Three pair ARTEX on port 8787 with CyberStrikeAI on port 8080. One pairs ARTEX on 8787 with AutoHunter on 18800.

Where the endpoints sit

The 438 endpoints concentrate in a few countries and hosting networks.

Country	Endpoints	Share
United States	138	31.5%
Hong Kong	94	21.5%
China	62	14.2%
Singapore	26	5.9%
Japan	19	4.3%

Hosting network	Endpoints
PEG TECH INC	63
Amazon.com, Inc.	55 (including 50 of 69 HexStrikeAI endpoints)
Hangzhou Alibaba Advertising Co.,Ltd.	23
NetLab Global	22
CTG Server Limited	18

The 11 endpoints with an upstream C2 flag

Eleven endpoints (2.5% of 438) on 10 dedicated IPv4 addresses carry an upstream C2 flag; the study records do not independently validate that classification. Because independent per-address corroboration is absent, all 10 addresses are withheld. None of the 10 addresses overlaps with the 405 hosts that hold the other endpoints.

All three flagged CyberStrikeAI controllers present the certificate subject `CN=CyberStrikeAI` and the same TLS JARM value, `40d40d40d00000000043d43d00043d6aff5ab0f4fc31897186312c9e639319`.

The 11 flagged endpoints comprise six ARTEX, three CyberStrikeAI, one PentAGI and one Cairn endpoint. Their host-level details are withheld pending independent validation.

Cloud leases change hands. Require a matching certificate subject, JARM value or endpoint process before blocking any framework address.

CHAPTER 03

Workbench A: Numbered Task Folders and Windows Impersonation Staging

IN SHORT

Workbench A serves 6,203 files; a 270-entry sample includes a root tool store and 26 numbered task folders. Byte-identical GodPotato and PrintSpoofer files appear at the root and in one task folder. Hunt for GodPotato and PrintSpoofer execution on IIS and SQL Server hosts.

INSIDE THIS CHAPTER

Matching files at the root and in task folders 12

Re-used files across tasks 13

03 Workbench A: Numbered Task Folders and Windows Impersonation Staging

Multi-agent offense frameworks hand each worker a dedicated directory. It holds that worker's scan output, session caches and the post-exploitation tools it needs. If the operator serves the parent directory over HTTP, the whole task tree becomes readable; external attack surface management can surface such exposed directories.

A folder listing shows what was staged. It does not show what happened on any target. Nothing in either listing names an AI agent framework. No IP, certificate or hash links the workbenches to the 438 endpoints. We treat the 26 sampled numbered folders as subagent tasks only as an analytical model.

Workbench A serves its directory tree over an unauthenticated HTTP listener. In a sample of 270 catalog entries, Python scripts lead with 127 files (47.0%), ahead of 16 logs, 11 JavaScript files and 9 PHP files. The sample also holds seven `.dll`, seven `.sql`, seven `.json`, seven `.ps1` and six `.exe` files.

In that 270-entry sample, 26 numbered folders split the work by assignment (folder IDs withheld). Several folders add a letter suffix to the task number, which looks like re-runs of the same task. Figure 2 groups the 40 files we selected from both workbenches into six classes.

FIGURE 2 · WORKBENCH CAPABILITY & DETECTION MATRIX — ODIN BY BRANDEFENSE · TLP:CLEAR

Six artifact classes separate privesc binaries from SPA fallback noise

40 selected workbench files (37 unique SHA-256s) on two open-directory hosts (addresses withheld)

1. TOOL / BINARY	2. HOW USED (CLI / CONFIG)	3. TARGET SURFACE	4. DEFENDER SIGNAL
Potato & SMB Exploit 6 files (3 unique SHA-256s)	SeImpersonate & MS17-010 staging `GodPotato, PrintSpoofer, MS17-010`	Windows IIS/SQL & SMBv1 hosts	sid:9531001-002 + Sigma 1
Chisel & Pivot Tools 9 files (9 unique SHA-256s)	WS SOCKS5, socat & LDAP/AXFR `Chisel, Socat & AD recon`	Internal SMB 445, AD 389 & DNS	sid:9531003-004 + Sigma 2
.NET SSO & Cipher 8 files (8 unique SHA-256s)	CIL disassembly & MD5 cracking `.NET cipher DLL, CIL & MD5 logs`	.NET SSO portals & config XML	SSO code & key audit
SQLMap & HTTP Logs 8 files (8 unique SHA-256s)	Batch SQLi & HTTP pull listeners `HTTP listener + SQLMap logs`	Web parameters & HTTP listeners	sid:9531007-008 + Sigma 3
Subagent Recon & Auth 5 files (5 unique SHA-256s)	Task API probes & Graph tokens `API probes + token files`	Admin APIs & OAuth2 tenants	Hunt only
SPA & Library Noise 4 files (4 unique SHA-256s)	Filters 676 B SPA & sqlmap libs `SPA fallback copies + sqlmap libs`	SPA 200 OK shells & Python libs	Not indicators

Scope & source: Covers 40 selected files on two open-directory workbenches. Recorded by Odin by Brandefense, 18–20 Nov 2026.
What this does not show: Zero runnable exploit bodies or victim identifiers are published; open-directory host addresses are withheld.

Figure 2: Six artifact classes separate privesc binaries from SPA fallback noise

Matching files at the root and in task folders

Matching Windows binaries appear at the web root and in a numbered task folder. The listing does not establish who placed them there or which copy came first. Each pair has identical byte sizes and matching SHA-256 values.

Tool	Root store	Task folder file	Size	What it does
GodPotato	Name withheld	Name withheld	57,344 bytes	Abuses DCOM/RPC OXID resolution to impersonate `NT AUTHORITY\SYSTEM` from a service account
PrintSpoofer	Name withheld	Name withheld	27,136 bytes	Abuses Print Spooler named pipes for the same elevation
MS17-010 script	Name withheld	Name withheld	43,626 bytes	Public SMBv1 exploit script for CVE-2017-0144

GodPotato and PrintSpoofer both need a Windows service account that holds `SeImpersonatePrivilege`. IIS worker processes (`w3wp.exe`) and SQL Server (`sqlservr.exe`) often run under such accounts. An attacker who lands a web shell or a SQL command shell under that account can use either tool to reach SYSTEM (T1134.001).

Staging both in the same task folder gives a fallback. If the Print Spooler service (`spoolsv.exe`) is off, GodPotato still has the RPC path. The same folder holds a 69,632-byte binary file and a PKCS#12 certificate bundle whose names are withheld.

The MS17-010 script in another sampled task folder sits with a 16,974-byte SMB transport helper script (name withheld) from the same public toolkit. Together they target unpatched SMBv1 servers. Microsoft fixed CVE-2017-0144 in security bulletin MS17-010 on 14 March 2017.

NVD rates it 8.8 (High) under CVSS 3.1. Affected desktop editions are Windows Vista SP2, 7 SP1, 8.1, RT 8.1, and 10 original, 1511 and 1607. Affected server editions are Windows Server 2008 SP2, 2008 R2 SP1, 2012, 2012 R2 and 2016 (NVD; MS17-010).

Re-used files across tasks

Other sampled folders show workers reusing the same staging data. Four copies of one 71,165-byte admin JavaScript bundle sit at the root and in three task folders, all with one SHA-256 value. Three SQL archives across three other numbered task folders also share digests. That pattern fits repeated task runs over shared staging data.

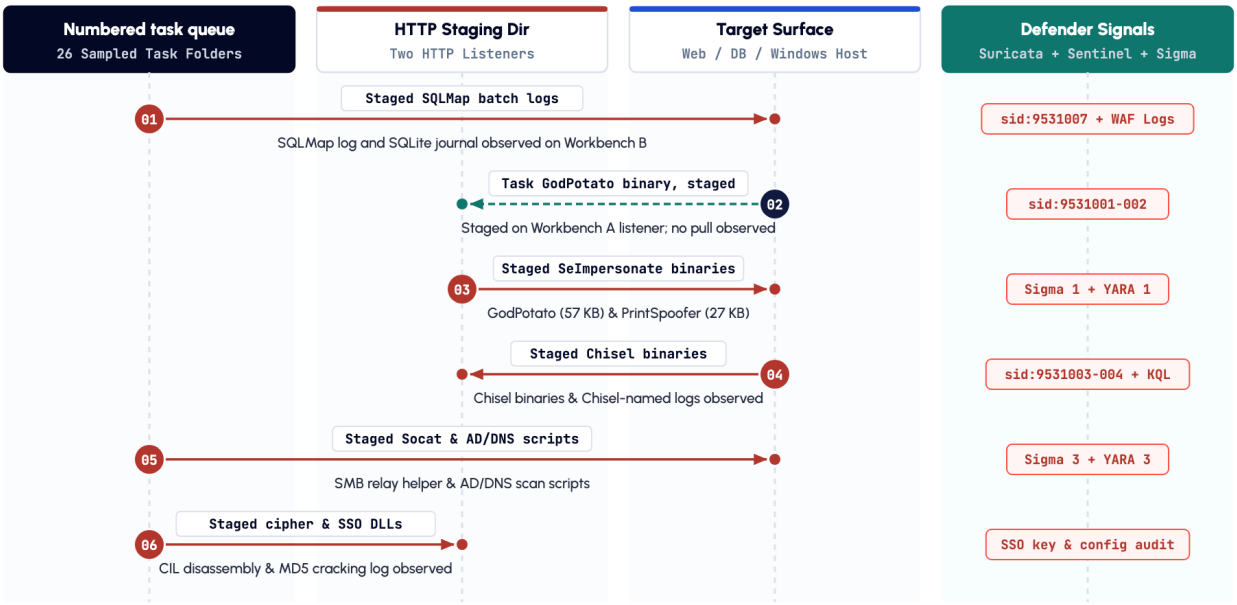
Sampled location	What it holds (paths and owner-specific names withheld)	Size range (bytes)	Role (not confirmed on any target)
Root and privesc task folder	GodPotato, PrintSpoofer and a 69,632-byte binary payload	27,136 to 69,632	Windows impersonation elevation
Root and SMB relay task folder	MS17-010 exploit script, SMB transport helper, Socat relay logs and a Chisel tunnel log	4,811 to 43,626	MS17-010 SMB relay and Chisel tunnel logs
Directory-discovery task folder	Three short Python scripts and RDP/SMB banner logs	135 to 6,565	Active Directory LDAP, DNS zone transfer, SMB and RDP discovery
SSO analysis task folder	SSO assembly copies, a CIL disassembly file, an MD5 cracking log and a potfile	2,121 to 1,975,808	.NET SSO disassembly and offline MD5 cracking
SQLi session task folder	`sqlmap` SQLite session journals	8,272 to 61,832	SQL injection session state
Three token task folders	JSON files named after OAuth and Microsoft Graph tokens	6,548 to 8,276	Files named as OAuth and Microsoft Graph tokens
Two API-test task folders	Authorization-matrix and admin API test scripts	804 to 71,165	Admin API and access-control testing

Figure 3 lays out the six-stage sequence these folders suggest, from web injection on Workbench B to tunnelling on Workbench A. It is a reconstruction from file names and sizes, not a log of events. The two hosts are not linked; the figure combines them only to illustrate the pattern.

FIGURE 3 · NUMBERED TASK TO PRIVESC & TUNNEL SEQUENCE (MODEL) ODIN BY BRANDEFENSE · TLP:CLEAR

Six-stage model: web injection to token impersonation staging on two unlinked hosts

Six-stage structural workflow reconstructed from Workbench A and Workbench B artifacts



Scope & source: Built from file paths, byte sizes and log names on two unlinked hosts.
What this does not show: Staged binaries and scripts indicate intended capability only; downstream execution on targets is unverified.

Figure 3: Six-stage model: web injection to token impersonation staging on two unlinked hosts

04

CHAPTER 04

Tunnels, Pivot Scripts and .NET Cipher Files

IN SHORT

Beyond elevation, Workbench A stages two Chisel builds, Socat SMB relay logs, directory discovery scripts and a .NET SSO cipher analysis kit. Each points to a different defender control. Inspect egress for `chisel-v3` and audit SSO components for static keys.

INSIDE THIS CHAPTER

Chisel reverse SOCKS tunnels and Socat relays 16

.NET single-sign-on cipher analysis 16

04 Tunnels, Pivot Scripts and .NET Cipher Files

After local elevation, an operator needs a way into segmented internal networks and a way to read application secrets. Workbench A stages both.

Chisel reverse SOCKS tunnels and Socat relays

Two Windows Chisel builds sit in the root store (names and paths withheld). One measures 11,954,176 bytes, and the version 1.8.1 build measures 8,676,864 bytes.

Chisel is a public TCP/UDP tunnel carried over HTTP and secured with SSH. Its reverse SOCKS mode (`R:socks`) lets a client inside a network open a SOCKS5 proxy on the operator's server. The client starts with an HTTP-to-WebSocket upgrade that carries the header `Sec-WebSocket-Protocol: chisel-v3` (T1572).

The listing includes two Chisel-named logs: one at the root (152,308 bytes) and one in a task folder (4,811 bytes). Their names reference high-numbered listener ports (withheld), but the listing does not confirm that a tunnel ran.

The same task folder holds two Socat-named logs (7,797 bytes and 6,417 bytes; names withheld). Their file names pair local relay ports (withheld) with the SMB helper script, suggesting a planned relay to SMB port 445. The listing does not confirm that a relay connection ran or fed the MS17-010 script.

A sibling task folder holds three short Python scripts (1,319 to 1,561 bytes). From their file names, they cover an LDAP check on port 389, a directory attribute scan and a DNS zone transfer check. Two companion logs carry names pointing to RDP NTLM and legacy SMB checks, though their contents are not confirmed. All script and log names, paths and hashes are withheld because the workbench owner is not established.

Artifact	Sampled location	Size (bytes)	Role
Chisel Windows binary (name withheld)	Root store	11,954,176	Go WebSocket SOCKS5 tunnel
Chisel 1.8.1 build (name withheld)	Root store	8,676,864	Chisel 1.8.1 Windows build
Two Chisel-named logs (names withheld)	Root and task folder	4,811 to 152,308	Files named after high-port Chisel listeners
Two Socat-named logs and SMB helper script (names withheld)	SMB relay task folder	6,417 to 16,974	Files named after local Socat relay listeners, plus SMB helper
Three discovery scripts (names withheld)	Discovery task folder	1,319 to 1,561	AD LDAP and DNS zone transfer checks

.NET single-sign-on cipher analysis

A third cluster targets custom .NET single-sign-on and portal login libraries. All vendor, campus, payment-SDK and gateway names in these files are withheld.

In the root store sits a .NET symmetric cipher DLL (40,960 bytes; name and path withheld). Next to it are a 1.18 MB archive and its extracted folder (names withheld). That folder holds a 46,310-byte Common Intermediate Language (CIL) disassembly file (name withheld) and seven SSO module assemblies (19,456 to 1,975,808 bytes). A 2,121-byte XML configuration file sits with them, and matching binary copies appear in a numbered task folder.

Many intranet and campus SSO agents written in C# wrap AES (Rijndael) or 3DES in a helper DLL. They use it to encrypt cookies, database connection strings and SSO tickets. Disassembling such a library to CIL, with `ildasm` or `ilspycmd`, can expose hard-coded keys, IVs or unsalted MD5 routines (T1552.001).

The SSO analysis task folder fits that reading. It holds an offline MD5 cracking log (10,266 bytes) and a potfile (13,484 bytes; names withheld). Another sampled task folder holds two Base64-encoded application and database configuration files (names and contents withheld).

Artifact	Sampled location	Size (bytes)	Role
.NET symmetric cipher DLL (name withheld)	Root store	40,960	.NET symmetric cipher helper
CIL disassembly file (name withheld)	Extracted archive	46,310	CIL disassembly
SSO module assemblies (7; names withheld)	Extracted archive	19,456 to 1,975,808	Third-party .NET SSO libraries
MD5 cracking log (name withheld)	SSO task folder	10,266	Offline MD5 cracking session

Three other sampled task folders hold files named after OAuth and Microsoft Graph tokens (folder IDs, file names and contents withheld). The listing entries suggest interest in application access tokens (T1528).

05

CHAPTER 05

Workbench B: SQLMap Batch Runs and HTTP Listener Logs

IN SHORT

Workbench B serves 1,970 files across three HTTP listeners. Five `sqlmap` logs sit beside 34.94 MB of HTTP listener logs, the web-injection side of the same staging habit. Block the `sqlmap/` user agent and parameterize queries.

INSIDE THIS CHAPTER

What the runner tree holds 19

05 Workbench B: SQLMap Batch Runs and HTTP Listener Logs

SQL injection rarely stops at reading a database banner. An operator testing stacked queries, `xp_cmdshell` or file writes often runs a plain Python HTTP listener on the scanner host. The target database can then pull a second-stage file from it, or call back to confirm a blind injection (T1190).

Workbench B shows that pattern across three unauthenticated HTTP listeners (ports withheld), holding 1,970 files in total (984 on the runner listener). Like Workbench A, it exposes `sqlmap` SQLite session files next to an open HTTP tree. No shared IP, certificate or file hash links it to Workbench A. Figure 4 compares the layers of both hosts.

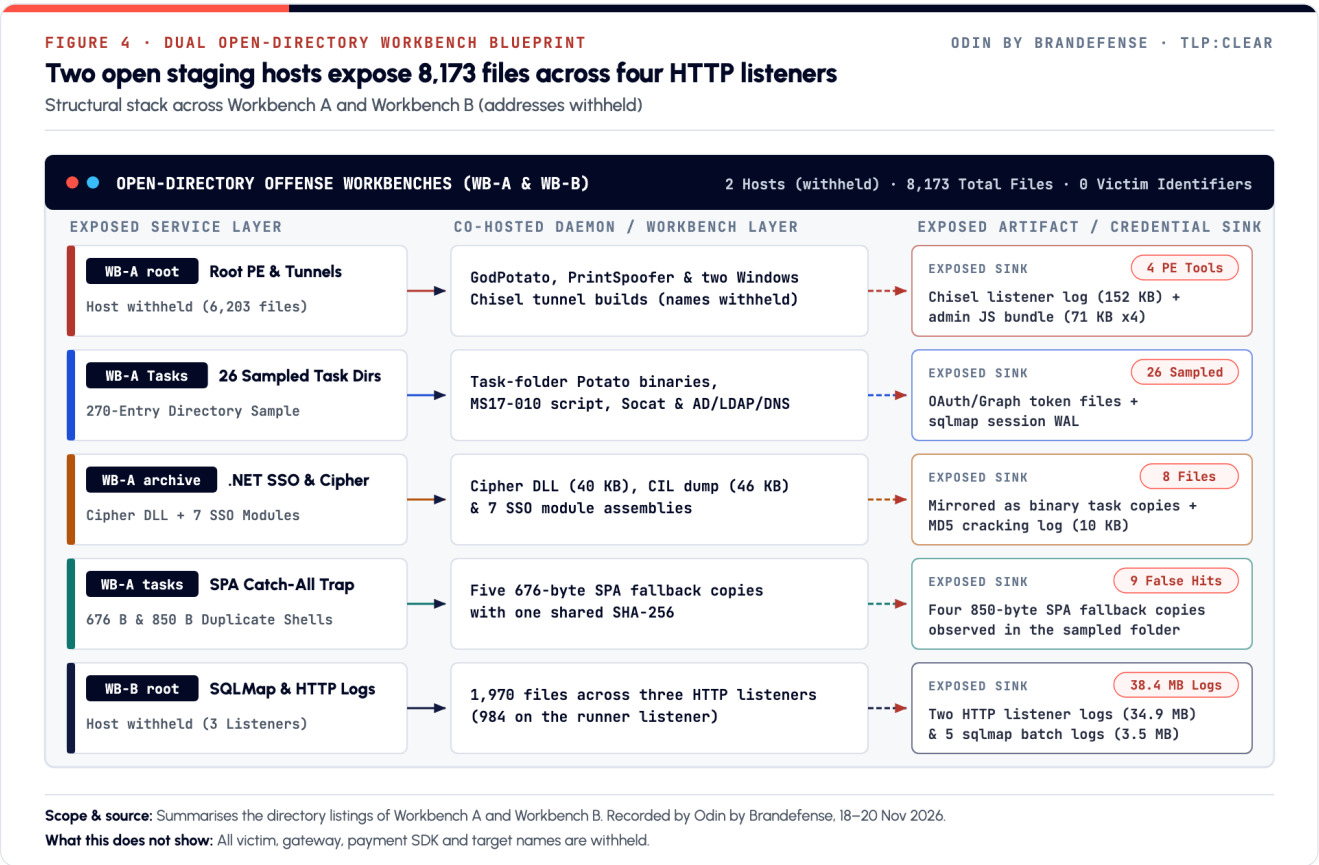


Figure 4: Two open staging hosts expose 8,173 files across four HTTP listeners

What the runner tree holds

In a sample of 362 entries from the runner listener, 284 `.py` files and 18 `.xml` files belong to a full Git checkout of upstream `sqlmap`. The tree also holds a compiled Dart web bundle (3.76 MB) and a 35,849-byte word list (file names and paths withheld).

The notable files are seven logs at the root, 38,429,854 bytes in total. Two carry HTTP listener names for two web listeners (ports withheld). Five carry `sqlmap` names and range from 40,930 to 1,743,251 bytes, though the listing does not establish their run order or contents.

Log artifact (name and path withheld)	Size (bytes)	What the listing shows
HTTP-named listener log 1	17,833,637	HTTP-named log file at the root; contents not confirmed
HTTP-named listener log 2	17,110,690	Second HTTP-named log file at the root; contents not confirmed
`sqlmap`-named log 1	1,743,251	`sqlmap`-named log file; run order and contents not confirmed
`sqlmap`-named log 2	1,550,847	`sqlmap`-named log file; run order and contents not confirmed
`sqlmap`-named log 3	103,244	`sqlmap`-named log file; run order and contents not confirmed
`sqlmap`-named log 4	47,255	`sqlmap`-named log file; run order and contents not confirmed
`sqlmap`-named log 5	40,930	`sqlmap`-named log file; run order and contents not confirmed

A `sqlmap` output folder also holds a 32,768-byte SQLite shared-memory file for a target host. The target name is withheld.

The two HTTP-named logs account for most of that volume. Their names and sizes suggest HTTP listener output across two web listeners, though the listing does not include log contents or confirm callback activity.

CHAPTER 06

Noise, False Positives and Other Explanations

IN SHORT

Some files that look like leaked secrets are blank web-app pages, and none of the new framework endpoints has independent C2 validation. Hash before you trust a scraped file, and never auto-block a candidate address.

INSIDE THIS CHAPTER

Single-page app pages saved as secrets 22

Library code is not operator code 22

Upstream flags and banners are not C2 proof 23

A contracted red team is possible 23

06 Noise, False Positives and Other Explanations

This study has three sources of noise and one open alternative explanation. Figure 5 sets the 9,908 records against the 438 new endpoints and the 11 endpoints that carry an upstream C2 flag.

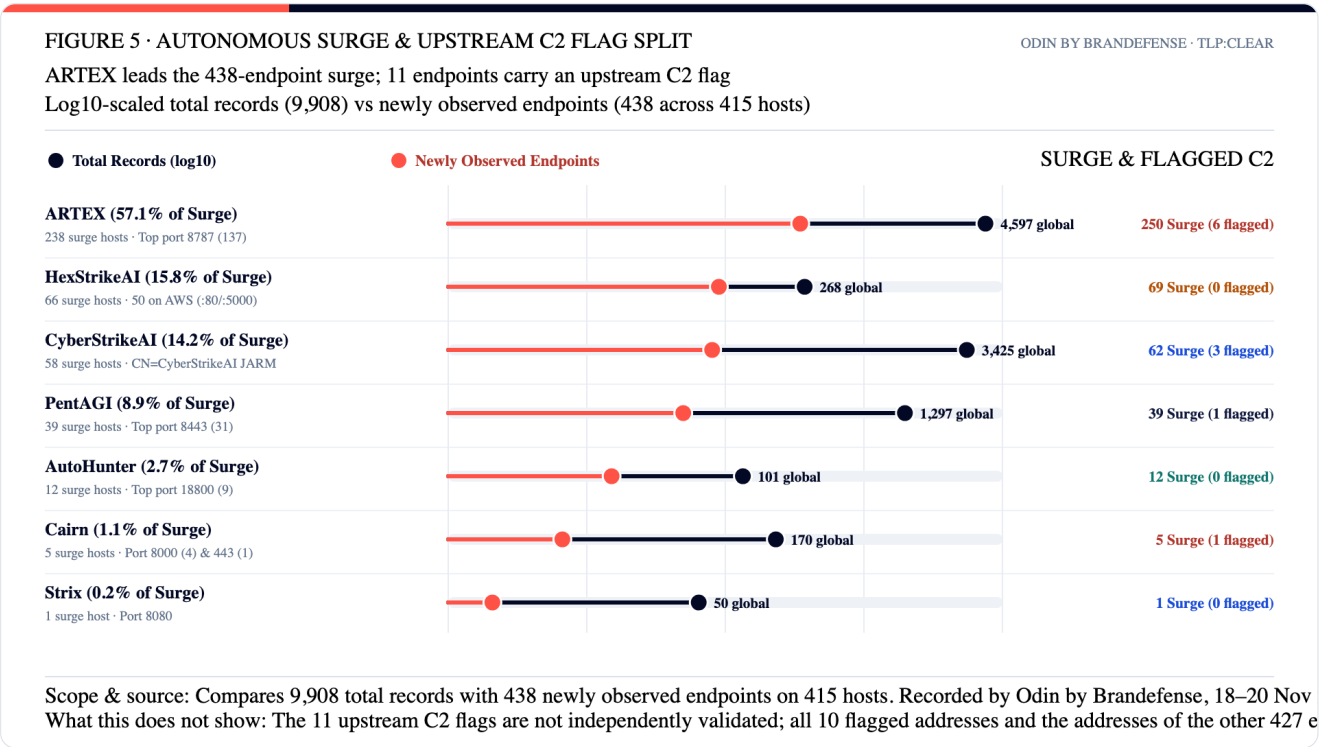


Figure 5: ARTEX leads the 438-endpoint surge; 11 endpoints carry an upstream C2 flag

Single-page app pages saved as secrets

On Workbench A, five differently named files all measure 676 bytes and share one SHA-256 value. Their names suggest environment-variable files, Git configuration and an app script. Four more output files in one sampled task folder share one 850-byte value (names, paths and folder IDs withheld).

A single-page application (SPA) returns the same HTML start page, with status 200, for any path it does not know. A scraper or worker that checks only for status 200 saves that page under whatever name it asked for. So nine "credential leaks" are really nine copies of two blank pages. Compare file hashes against a known page before treating a scraped configuration file as a secret.

Library code is not operator code

The `sqlmap` checkout on Workbench B adds 284 ordinary Python files from the scanner's core library and bundled web helpers (file names and paths withheld). They are public library code. Hunt for the operator's logs, not the scanner's source.

Upstream flags and banners are not C2 proof

Of the 438 new endpoints, 11 carry an upstream C2 flag without independent per-address validation, and 427 (97.5%) are suspected or candidates. In HexStrikeAI, 50 of 69 endpoints (72.5%) sit on short-lived AWS leases. Some candidate certificates are default Kubernetes (`CN=k3s`) certificates on DigitalOcean. Others carry test strings planted by scanners. These are reasons not to block on a single banner.

Artifact class	Files	Unique SHA-256	Workbench	Defender action
Windows privilege escalation and exploit	6	3	A	Hunt on endpoints; restrict `SeImpersonatePrivilege`
Tunnel and pivot tools	9	9	A	Hunt network and process activity
.NET SSO and cipher files	8	8	A	Audit SSO components; hunt for the cipher DLL
SQL injection and HTTP runner files	8	8	A and B	Hunt web and staging logs
Subagent discovery and token files	5	5	A	Hunt only
SPA pages and library code	4	4	A and B	Not indicators

The 40 selected files hash to 37 unique SHA-256 values. Thirty-six are offense or task files; four are noise.

A contracted red team is possible

Authorized consultancies and intruders use the same tools: numbered task queues, ARTEX, `sqlmap` and Chisel. A paid penetration-testing service could have left these trees open. Even so, serving 6,203 task files, .NET SSO libraries and OAuth token files through unauthenticated services exposes third parties to real risk. The workbench owners are not established, so both host addresses are withheld.

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 threat intelligence

- [ ] Do not block dual-use framework addresses on a single banner; the 10 flagged C2 addresses are withheld pending per-address validation.
- [ ] Use the two JARM values and `CN=CyberStrikeAI` as hunting selectors only. Pair them with a certificate subject, URI or endpoint process before acting.

Windows and Active Directory

- [ ] Review which accounts hold "Impersonate a client after authentication" (`SeImpersonatePrivilege`). Run IIS and SQL Server pools as group managed service accounts or virtual accounts with least privilege.
- [ ] Disable the Print Spooler service on servers that do not print. Block unapproved GodPotato, PrintSpoofer and Chisel binaries with application control or EDR.
- [ ] Remove SMBv1 everywhere, confirm the MS17-010 update for CVE-2017-0144 is installed, and restrict DNS zone transfers to secondary authoritative servers.

Network security

- [ ] Inspect outbound HTTP-to-WebSocket upgrades at egress proxies for `Sec-WebSocket-Protocol: chisel-v3`.
- [ ] Stop database and web tiers from opening direct HTTP connections to raw IP addresses or unapproved high ports.

Application security and IAM

- [ ] Audit custom .NET SSO and portal components for hard-coded symmetric keys and plain MD5 password hashes. Move keys into DPAPI- or HSM-backed vaults, and hash passwords with Argon2id or PBKDF2.
- [ ] Enforce parameterized SQL queries in web portals, and block the `sqlmap/` user agent at the WAF.
- [ ] Deploy the rules in Chapter 8, and test them on your own traffic first.

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 27
How the other rules work 28
MITRE ATT&CK mapping 28

08 Detection Rules

The full rules are in Appendix B. The Suricata file passes a syntax lint. Suricata, YARA, Sigma and KQL engines were not available to load the files, and the rules were not replayed against live traffic. Test and tune them in your own environment before production use.

Rule file	What it catches
`odin_subagent_workspaces_and_potato_chisel.rules`	Suricata 7: seven signatures for GodPotato, PrintSpoofer and Chisel downloads, the `chisel-v3` WebSocket handshake, the CyberStrikeAI certificate, the `sqlmap` user agent and exposed workbench listings
`sentinel_subagent_task_privesc_and_chisel_hunt.kql`	Microsoft Sentinel and Defender XDR: impersonation tools, Chisel SOCKS, service-process shells and connections to locally validated C2 addresses
`sigma_potato_spoofer_chisel_and_task_staging.yml`	Sigma (3 rules): GodPotato and PrintSpoofer use on Windows, Chisel reverse SOCKS clients, and Linux staging listeners, Socat SMB relays and `sqlmap` batch runs
`odin_potato_privesc_and_task_runners.yar`	YARA (3 rules): the impersonation tools, Chisel binaries, and MS17-010 or `sqlmap` task files

How the Suricata signatures work

SID	Looks for	Why
9531001	Outbound `GET` for a GodPotato `.exe` binary	A server pulling an impersonation tool over plain HTTP
9531002	Outbound `GET` for a PrintSpoofer `.exe` binary	The same, for the Print Spooler tool
9531003	Outbound `GET` for a Chisel `.exe` binary	A host fetching a tunnel binary
9531004	Outbound request headers with `Sec-WebSocket-Protocol: chisel-v` and `Upgrade: websocket`	The Chisel tunnel handshake itself
9531006	A server certificate subject with `CN=CyberStrikeAI`	Hunt only: pair with an address or process before acting
9531007	An inbound request whose user agent contains `sqlmap/`	Automated SQL injection against your web tier
9531008	An outbound HTTP 200 directory listing from a protected host that names GodPotato, PrintSpoofer, Chisel or `sqlmap`	An exposed staging directory on a protected server

Start all SIDs in alert mode. After tuning, convert only SID 9531004 to drop on production Windows server subnets. Keep SID 9531006 as a hunting signal.

How the other rules work

The KQL query looks back 7 days and joins two views of each device. The process view finds locally supplied tool hashes and binary names, command lines naming the tools or `R:socks`, and service processes such as `w3wp.exe` starting a shell. The network view finds connections to locally validated C2 addresses, URLs naming the tools, and outbound connections from Chisel processes. Exact hostnames go in two named allowlists, which never hide a locally validated C2 address match.

Sigma rule	Fires when
Rule 1 (critical, Windows)	GodPotato or PrintSpoofer image names appear, or IIS, SQL Server, Apache or PHP-CGI starts a process with `-cmd`, `-i -c` or a named pipe
Rule 2 (high, Windows)	A Chisel Windows binary runs, or a `client` command line ends in `R:socks` or `:socks`
Rule 3 (high, Linux)	Python runs `http.server`, Socat relays SMB port 445, or a command line runs `sqlmap` in batch mode

Each Sigma rule excludes only exact named hosts or service accounts. There are no wildcard or prefix exclusions.

YARA rule	Fires when
`ODIN_Windows_GodPotato_PrintSpoofer_Impersonation_Tool`	A PE under 256 KB holds `SeImpersonatePrivilege`, a named-pipe API and a tool name or `ncacn_ip_tcp`
`ODIN_Go_Chisel_WebSocket_Reverse_SOCKS_Binary`	A PE under 16 MB holds `chisel-v3` and `Sec-WebSocket-Protocol` with `jpillora/chisel` or `R:socks`
`ODIN_Subagent_MultiTask_SMB_AD_And_SQLMap_Runner`	A file holds two MS17-010 toolkit markers, or both `sqlmap` injection-report strings

For custom .NET single-sign-on helpers whose file names are withheld, audit portal assemblies for static Rijndael or 3DES keys rather than matching one DLL name.

MITRE ATT&CK mapping

Tactic	Technique	How the workbenches use it	Detection
Initial Access	T1190 Exploit Public-Facing Application	`sqlmap` batch runs and HTTP callback listeners	SID 9531007; Sigma rule 3
Privilege Escalation	T1134.001 Token Impersonation/Theft	GodPotato and PrintSpoofer staged in task folders	SIDs 9531001, 9531002; KQL; Sigma rule 1; YARA
Command and Control	T1572 Protocol Tunneling	Chisel reverse SOCKS over WebSocket	SIDs 9531003, 9531004; KQL; Sigma rule 2; YARA
Credential Access	T1552.001 Credentials In Files	.NET SSO assembly disassembly and cipher helper	Hunting and code review
Credential Access	T1528 Steal Application Access Token	OAuth and Graph token files in task folders	Hunting only

09

CHAPTER 09

Indicators

IN SHORT

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

INSIDE THIS CHAPTER

Hunting selectors 30

09 Indicators

The workbench file hashes are withheld because the host owner is unknown. All 10 flagged C2 addresses and all workbench download paths are also withheld, leaving 0 published IP or hash indicators in Appendix A. Four text-only hunting selectors appear in this report: two JARM values, one certificate subject and one WebSocket header.

Type	Count	Where to use it
Flagged C2 addresses (IPv4, withheld)	0 published (10 withheld)	Withheld pending per-address validation
Workbench file hashes (SHA-256, withheld)	0 published (12 withheld)	Withheld because the host owner is unknown
TLS JARM values and certificate subject	3	Hunting, paired with another signal
WebSocket header	1	Egress proxy inspection

Hunting selectors

Indicator	Type	Use
`40d40d40d00000000043d43d00043d6aff5ab0f4fc31897186312c9e639319`	JARM of the three flagged CyberStrikeAI controllers	Hunt with `CN=CyberStrikeAI`
`27d40d40d00040d00042d43d000000d2e61cae37a985f75ecafb81b33ca523`	JARM observed on a flagged ARTEX endpoint	Hunt with another signal
`CN=CyberStrikeAI`	TLS certificate subject	Hunt (SID 9531006)
`Sec-WebSocket-Protocol: chisel-v3`	WebSocket header	Hunt at egress proxies

Withheld: the 10 flagged C2 addresses (pending per-address corroborating signals), their host-level details and the addresses of the 427 suspected and candidate endpoints. Both workbench addresses and their service ports are also withheld. All workbench file hashes, task-folder IDs, file and log names, and download paths are withheld because the host owners are not established. So are all victim and target names.

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 findings were recorded on 18–20 November 2026, covering the 438 newly observed endpoints and both exposed workbench directory listings.
- **Files, not outcomes.** The workbench files were read as listings and file metadata. We ran no binary and contacted no system named in them. Nothing we saw confirms use against any target.
- **No independently validated C2 addresses.** Eleven endpoints carry an upstream C2 flag, but the study records do not independently corroborate each address. A family label on the other 427 endpoints is a product fingerprint, not proof of hostile use.
- **No link between workbenches.** No shared IP, certificate or file hash ties Workbench A to Workbench B.
- **An authorized test is possible.** The tooling fits a contracted red team as well as an intruder. The open directories still expose third-party data.
- **Rule testing.** The Suricata file passes a syntax lint. The rule engines were not available to load the files, and no rule was replayed against live traffic. Test and tune them before production use.

11

CHAPTER 11

Sources

IN SHORT

This chapter lists the public references used in the report.

11 Sources

1. Microsoft Learn. Impersonate a client after authentication (security policy setting). https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/impersonate-a-client-after-authentication (verified 20 November 2026)
2. Microsoft Security Bulletin MS17-010, Critical: Security Update for Microsoft Windows SMB Server (published 14 March 2017). https://learn.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010 (verified 20 November 2026)
3. NIST National Vulnerability Database. CVE-2017-0144 (CVSS 3.1 base score 8.8, High). https://nvd.nist.gov/vuln/detail/CVE-2017-0144 (verified 20 November 2026)
4. CVE Program. CVE-2017-0144 record. https://www.cve.org/CVERecord?id=CVE-2017-0144 (verified 20 November 2026)
5. jpillora. Chisel: a fast TCP/UDP tunnel over HTTP. https://github.com/jpillora/chisel (verified 20 November 2026)
6. MITRE ATT&CK. T1134.001 Access Token Manipulation: Token Impersonation/Theft. https://attack.mitre.org/techniques/T1134/001/ (verified 20 November 2026)
7. MITRE ATT&CK. T1572 Protocol Tunneling. https://attack.mitre.org/techniques/T1572/ (verified 20 November 2026)
8. MITRE ATT&CK. T1190 Exploit Public-Facing Application. https://attack.mitre.org/techniques/T1190/ (verified 20 November 2026)
9. MITRE ATT&CK. T1552.001 Unsecured Credentials: Credentials In Files. https://attack.mitre.org/techniques/T1552/001/ (verified 20 November 2026)
10. MITRE ATT&CK. T1528 Steal Application Access Token. https://attack.mitre.org/techniques/T1528/ (verified 20 November 2026)
11. OWASP Cheat Sheet Series. SQL Injection Prevention Cheat Sheet. https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html (verified 20 November 2026)

APPENDICES

Indicators and Detection Rules

FOR DEFENDERS

No individual indicators are published for this study; Appendix A explains why. Appendix B carries every published detection rule in full.

INSIDE

A. Indicators of compromise (0) 36

B. Detection rules (4 files) 37

A Indicators of Compromise

No individual addresses or file hashes are published for this study. The hosts in this report are exposed or unattributed systems rather than confirmed adversary infrastructure, so they stay anonymous. Use the detection rules in Appendix B.

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_potato_privesc_and_task_runners.yar

YARA

```
/*
 * Odin by Brandefense - Monograph 31 / 35 Bounded YARA Structural Suite
 * Coverage: GodPotato/PrintSpoofer PE/CLR markers, Go Chisel reverse SOCKS binaries,
 *           and subagent SMB/SQLMap task runners
 * Classification: TLP:CLEAR
 */

rule ODIN_Windows_GodPotato_PrintSpoofer_Impersonation_Tool
{
    meta:
        author = "Odin by Brandefense"
        date = "2026-11-20"
        monograph = "31 / 35"
        tlp = "CLEAR"
        description = "Matches structural PE/CLR strings of GodPotato and PrintSpoofer SeImpersonatePrivilege
binaries"
    strings:
        $priv = "SeImpersonatePrivilege" ascii wide
        $pipe1 = "CreateNamedPipe" ascii wide
        $pipe2 = "ImpersonateNamedPipeClient" ascii wide
        $tool1 = "GodPotato" ascii wide nocase
        $tool2 = "PrintSpoofer" ascii wide nocase
        $rpc = "ncacn_ip_tcp" ascii wide
    condition:
        uint16(0) == 0x5A4D and filesize < 256KB and $priv and ($pipe1 or $pipe2) and ($tool1 or $tool2 or $rpc)
}

rule ODIN_Go_Chisel_WebSocket_Reverse_SOCKS_Binary
{
    meta:
        author = "Odin by Brandefense"
        date = "2026-11-20"
        monograph = "31 / 35"
        tlp = "CLEAR"
        description = "Matches Go-compiled Chisel HTTP/WebSocket reverse SOCKS5 tunnel binaries"
    strings:
        $ws_proto = "chisel-v3" ascii
        $pkg = "jpillora/chisel" ascii
        $hdr = "Sec-WebSocket-Protocol" ascii
        $socks = "R:socks" ascii
    condition:
        uint16(0) == 0x5A4D and filesize < 16MB and $ws_proto and $hdr and ($pkg or $socks)
}

rule ODIN_Subagent_MultiTask_SMB_AD_And_SQLMap_Runner
{
    meta:
        author = "Odin by Brandefense"
        date = "2026-11-20"
        monograph = "31 / 35"
        tlp = "CLEAR"
        description = "Matches subagent task scripts for SMBv1 MS17-010 relays or SQLMap batch logs"
    strings:
        $smb1 = "SMB_COM_NT_TRANSACT" ascii
        $smb2 = "TRANS2_SESSION_SETUP" ascii
        $smb3 = "PeekNamedPipe" ascii
```

```
        $sqli1 = "sqlmap identified the following injection point" ascii
        $sqli2 = "back-end DBMS:" ascii
    condition:
        filesize < 2MB and ((2 of ($smb*)) or ($sqli1 and $sqli2))
}
```

odin_subagent_workspaces_and_potato_chisel.rules

SURICATA

```
# Odin by Brandefense - Monograph 31 / 35 Suricata 7.x Detection Suite
# Coverage: GodPotato/PrintSpoofer HTTP staging, Chisel binaries & WebSocket chisel-v3 tunnels,
#           CyberStrikeAI TLS certificates, SQLMap UA, and open workbenches
# Classification: TLP:CLEAR

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R31] Out-of-Band GodPotato Windows Privesc Binary
Retrieval"; flow:established,to_server; http.method; content:"GET"; http.uri; bsize:>13; content:"GodPotato";
nocase; fast_pattern; pcre:"/\/GodPotato[^\/\r\n]*\.exe$/i"; classtype:trojan-activity; sid:9531001; rev:3;
metadata:created_at 2026_11_20, updated_at 2026_11_22, deployment perimeter, tactic privilege_escalation;)

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R31] Out-of-Band PrintSpoofer Named-Pipe Privesc Binary
Retrieval"; flow:established,to_server; http.method; content:"GET"; http.uri; bsize:>16; content:"PrintSpoofer";
nocase; fast_pattern; pcre:"/\/PrintSpoofer[^\/\r\n]*\.exe$/i"; classtype:trojan-activity; sid:9531002; rev:3;
metadata:created_at 2026_11_20, updated_at 2026_11_22, deployment perimeter, tactic privilege_escalation;)

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R31] Out-of-Band Chisel Reverse Tunnel Binary
Retrieval"; flow:established,to_server; http.method; content:"GET"; http.uri; bsize:>10; content:"/chisel";
nocase; fast_pattern; pcre:"/\/chisel([\d._-]+)?\.exe$/i"; classtype:trojan-activity; sid:9531003; rev:2;
metadata:created_at 2026_11_20, updated_at 2026_11_22, deployment perimeter, tactic command_and_control;)

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R31] Chisel HTTP-to-WebSocket Reverse SOCKS Tunnel
Handshake (chisel-v3)"; flow:established,to_server; http.header; bsize:>30; content:"Sec-WebSocket-Protocol:
chisel-v"; nocase; fast_pattern; pcre:"/Upgrade:\s*websocket/i"; classtype:trojan-activity; sid:9531004; rev:1;
metadata:created_at 2026_11_20, deployment perimeter, tactic command_and_control;)

alert tls $EXTERNAL_NET any -> $HOME_NET any (msg:"ODIN [R31] HUNT Autonomous Offense CyberStrikeAI Server TLS
Certificate"; flow:established,from_server; tls.cert_subject; bsize:>15; content:"CN=CyberStrikeAI";
fast_pattern; classtype:trojan-activity; sid:9531006; rev:2; metadata:created_at 2026_11_20, updated_at
2026_11_20, deployment perimeter, policy behavioral_hunt;)

alert http any any -> $HOME_NET any (msg:"ODIN [R31] Automated SQLMap Injection Scanner User-Agent Probe";
flow:established,to_server; http.user_agent; bsize:>6; content:"sqlmap/"; fast_pattern; classtype:web-
application-attack; sid:9531007; rev:1; metadata:created_at 2026_11_20, deployment perimeter_and_waf, tactic
initial_access;)

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R31] Open-Directory Subagent Privesc or SQLMap Runner
Listing"; flow:established,from_server; http.stat_code; content:"200"; http.response_body; bsize:>40;
content:"Directory listing for /"; fast_pattern; pcre:"/(GodPotato|PrintSpoofer|chisel|sqlmap)/i";
classtype:attempted-recon; sid:9531008; rev:3; metadata:created_at 2026_11_20, updated_at 2026_11_22, deployment
perimeter, tactic reconnaissance;)
```

sentinel_subagent_task_privesc_and_chisel_hunt.kql

KQL (SENTINEL / DEFENDER XDR)

```
// Odin by Brandefense - Monograph 31 / 35 Microsoft Sentinel & Defender XDR Hunt
// Correlates Windows SeImpersonatePrivilege binaries (GodPotato/PrintSpoofer), Chisel reverse
// SOCKS tunnels, service-process shells, and connections to locally validated C2s
// Classification: TLP:CLEAR
// Host allowlists apply only to the tool and URL branches; a confirmed C2 IP match is never excluded.

let Lookback = 7d;
let ApprovedAdminHosts = dynamic(["WIN-JUMPBOX-ADM-01", "NET-BASTION-OPS-01"]);
let ApprovedSecValidationHosts = dynamic(["REDTEAM-LAB-WIN-01", "APPSEC-SCANNER-LNX-01"]);
let ConfirmedC2Ips = dynamic([]); // Populate with locally validated multi-signal C2 IPv4s; study addresses are
withheld pending per-address validation.
let ServiceParentBinaries = dynamic(["w3wp.exe", "sqlservr.exe", "httpd.exe", "php-cgi.exe", "tomcat9.exe"]);
let PrivescAndTunnelHashes = dynamic([]); // Populate with locally validated tool SHA-256s; workbench file hashes
```

```
are withheld because host owners are unknown.
let SuspiciousHostProcs = DeviceProcessEvents
    | where Timestamp >= ago(Lookback)
    | where not(DeviceName in~ (ApprovedAdminHosts)) and not(DeviceName in~ (ApprovedSecValidationHosts))
    | where SHA256 in~ (PrivescAndTunnelHashes)
        or FileName has_any ("GodPotato", "PrintSpoofer", "chisel")
        or ProcessCommandLine has_any ("GodPotato", "PrintSpoofer", "R:socks", "chisel-v3", "sqlmap")
        or (InitiatingProcessFileName in~ (ServiceParentBinaries) and FileName in~ ("cmd.exe", "powershell.exe",
"pwsh.exe", "whoami.exe") and ProcessCommandLine has_any (" -cmd ", "-i -c", "pipe\\", "R:socks"))
    | project ProcTime = Timestamp, DeviceId, DeviceName, InitiatingProcessFileName, FileName, SHA256,
ProcessCommandLine;
DeviceNetworkEvents
| where Timestamp >= ago(Lookback)
| where RemoteIP in (ConfirmedC2Ips)
    or (not(DeviceName in~ (ApprovedAdminHosts)) and not(DeviceName in~ (ApprovedSecValidationHosts))
        and (RemoteUrl has_any ("GodPotato", "PrintSpoofer", "chisel")
            or InitiatingProcessFileName has "chisel"))
| summarize
    NetFirstSeen = min(Timestamp),
    NetLastSeen = max(Timestamp),
    DistinctRemoteIPs = dcount(RemoteIP),
    SampleRemoteEndpoints = make_set(strcat(RemoteIP, ":", tostring(RemotePort)), 12),
    SampleUrls = make_set(RemoteUrl, 10),
    ConnectionCount = count(), ConfirmedC2Hit = max(toint(RemoteIP in (ConfirmedC2Ips)))
    by DeviceId, DeviceName, InitiatingProcessFileName
| join kind=fullouter (SuspiciousHostProcs) on DeviceId, DeviceName
| extend ResolvedDevice = coalesce(DeviceName, DeviceName1)
| where isnotempty(ProcessCommandLine) or ConnectionCount > 0
| extend AlertAction = case(
    ConfirmedC2Hit == 1, "BLOCK_AND_TRIAGE_CONFIRMED_C2_CONNECTION",
    SHA256 in~ (PrivescAndTunnelHashes) or ProcessCommandLine has_any ("GodPotato", "PrintSpoofer", "R:socks"),
"ISOLATE_PRIVESC_OR_CHISEL_TUNNEL_HOST",
    "HUNT_TOOL_URL_OR_SERVICE_SHELL_ACTIVITY"
)
| extend SortTime = coalesce(ProcTime, NetLastSeen)
| project SortTime, NetFirstSeen, NetLastSeen, ProcTime, ResolvedDevice, AlertAction, InitiatingProcessFileName,
InitiatingProcessFileName1, FileName, SHA256, ProcessCommandLine, SampleRemoteEndpoints, SampleUrls,
ConnectionCount
| sort by SortTime desc
```

sigma_potato_spoofer_chisel_and_task_staging.yml

SIGMA

```
title: Odin Windows SeImpersonatePrivilege Token Impersonation via GodPotato or PrintSpoofer
id: 7a31b4e2-9015-4c8d-a431-953100000001
status: experimental
description: Detects Windows execution of GodPotato or PrintSpoofer binaries and characteristic RPC/named-pipe
privilege escalation command-line switches observed in numbered subagent task workspaces.
author: Odin by Brandefense
date: 2026-11-20
references:
  - https://attack.mitre.org/techniques/T1134/001/
  - https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/impersonate-a-
client-after-authentication
logsource:
  category: process_creation
  product: windows
detection:
  selection_binary:
    Image|contains:
      - '\GodPotato'
      - '\PrintSpoofer'
    Image|endswith: '.exe'
  selection_cli_potato:
    ParentImage|endswith:
      - '\w3wp.exe'
      - '\sqlservr.exe'
      - '\httpd.exe'
      - '\php-cgi.exe'
```

```
    CommandLine|contains:
      - ' -cmd '
      - ' -i -c '
      - '\pipe\'
  filter_approved_redteam_host:
    ComputerName:
      - 'REDTEAM-LAB-WIN-01'
      - 'SEC-VALIDATION-WIN-02'
  filter_approved_redteam_user:
    User:
      - 'CORP\svc_redteam_audit'
  condition: (selection_binary or selection_cli_potato) and not 1 of filter_*
falsepositives:
  - Authorized red-team exercises on explicitly named validation hosts or service accounts
level: critical
tags:
  - attack.privilege-escalation
  - attack.t1134.001
---
title: Odin Windows Chisel Reverse SOCKS5 Tunnel Client Invocation
id: 7a31b4e2-9015-4c8d-a431-953100000002
status: experimental
description: Detects Windows Chisel tunnel binary execution or reverse SOCKS5 client command-line invocations.
author: Odin by Brandefense
date: 2026-11-20
references:
  - https://attack.mitre.org/techniques/T1572/
logsource:
  category: process_creation
  product: windows
detection:
  selection_chisel_img:
    Image|contains: '\chisel'
    Image|endswith: '.exe'
  selection_chisel_cli:
    CommandLine|contains:
      - ' client '
    CommandLine|endswith:
      - 'R:socks'
      - ':socks'
  filter_approved_admin:
    ComputerName:
      - 'WIN-JUMPBOX-ADM-01'
      - 'APPSEC-BUILD-WIN-01'
  condition: (selection_chisel_img or selection_chisel_cli) and not 1 of filter_*
falsepositives:
  - Approved network tunnel testing on explicitly named admin jumpboxes
level: high
tags:
  - attack.command-and-control
  - attack.t1572
---
title: Odin Linux Subagent Open-Directory Staging Socat SMB Relay and SQLMap Batch Runner
id: 7a31b4e2-9015-4c8d-a431-953100000003
status: experimental
description: Detects unauthenticated Python HTTP staging listeners, Socat TCP port relays targeting SMB port 445,
or automated SQLMap batch runners.
author: Odin by Brandefense
date: 2026-11-20
references:
  - https://attack.mitre.org/techniques/T1190/
  - https://attack.mitre.org/techniques/T1572/
logsource:
  category: process_creation
  product: linux
detection:
  selection_http_stage:
    Image|endswith:
      - '/python3'
      - '/python'
    CommandLine|contains:
      - 'http.server'
```

```
    selection_socat_smb:
      Image|endswith:
        - '/socat'
      CommandLine|contains:
        - ':445'
    selection_sqlmap_batch:
      CommandLine|contains|all:
        - 'sqlmap'
        - '--batch'
    filter_approved_scanner_host:
      HostName:
        - 'APPSEC-SCANNER-LNX-01'
        - 'PT-RUNNER-LNX-02'
    filter_approved_scanner_user:
      User:
        - 'svc_dast_runner'
    condition: (selection_http_stage or selection_socat_smb or selection_sqlmap_batch) and not 1 of filter_*
falsepositives:
    - Authorized DAST or internal security validation hosts or accounts listed in filter_approved_scanner_*
level: high
tags:
    - attack.initial-access
    - attack.t1190
    - attack.command-and-control
```

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/numbered-task-folders-potato-binaries-and-chisel-tunnels/
Short Read · brandefense.io/odin-research/blog/inside-numbered-task-folders/
Defense Kit · brandefense.io/odin-research/defense-kit/