**Advanced Log Analysis – SSH Brute Force & Sudo Caching (Wazuh + Elastic Security)**

**Scope:** Description of my environment, the precise commands I executed, rationale for each step, and the method I used to correlate, enrich, and detect suspicious events.  
**Stack used:** Debian 22.x VM (Kali-like workload with Wazuh Agent) at 192.168.204.137 + Windows 11 VM (agent) at 192.168.204.138 running WSL Ubuntu with Wazuh Manager/Indexer/Dashboard, Filebeat, Elastic Security (optional), and Google Sheets for documentation.

**0) Summary Outcomes**

* **Observed** SSH brute-force activity and **captured** sudo timestamp caching behavior from the Debian VM.
* **Validated** the full ingestion chain (Agent → Manager → Filebeat → Indexer → Dashboard).
* **Compiled** a correlation spreadsheet linking failed logons to outbound network events.
* **Implemented** an Elastic Security detection for excessive egress and created a GeoIP enrichment pipeline.
* **50-word summary:** *Repeated SSH authentication failures followed by a successful login and a flurry of sudo commands suggest brute-force followed by privilege use. Subsequent DNS and HTTP outbound activity from the same host/time window increases suspicion. GeoIP enrichment located external hosts. A 1-minute egress spike (>1MB) produced an alert for possible data exfiltration.*

**1) System Topology & Event Flow**

[Debian 22.x VM (192.168.204.137) Wazuh Agent] → [WSL Ubuntu on Win11 (192.168.204.138) Wazuh Manager] → Filebeat → Wazuh Indexer → Wazuh Dashboard

**Why:** Knowing this path helps locate where data may be dropped or delayed.

**2) Requirements**

* **Debian 22.x VM** with Wazuh Agent installed (IP: 192.168.204.137).
* **Windows 11 VM** with WSL (Ubuntu) hosting:
  + wazuh-manager
  + wazuh-indexer
  + wazuh-dashboard
  + filebeat
* (Optional) Elastic Stack / Elastic Security for extended tasks.
* Local testing tools: hydra, nmap, and SSH client.

If reinstalling WSL from scratch, remove the distro via Windows Settings → Apps → Ubuntu → Uninstall, then re-install and re-deploy Wazuh before running the service commands below.

**3) Start & Check Services (WSL Ubuntu)**

From Windows PowerShell:

wsl -d Ubuntu

**Rationale:** Switch into the Linux environment where Wazuh runs.

Inside WSL (Ubuntu), enable and bring up services:

sudo systemctl daemon-reload # Why: refresh unit files

sudo systemctl enable --now wazuh-manager # Why: core event handling

sudo systemctl enable --now wazuh-indexer # Why: stores and indexes events

sudo systemctl enable --now wazuh-dashboard # Why: web interface

sudo systemctl enable --now filebeat # Why: ships alerts to the indexer

If anything looks wrong, check service status:

systemctl status wazuh-manager wazuh-indexer wazuh-dashboard filebeat

Useful logs to tail for diagnostics:

sudo journalctl -u wazuh-manager -f

sudo journalctl -u filebeat -f

sudo journalctl -u wazuh-indexer -f

sudo journalctl -u wazuh-dashboard -f

**Why:** These commands verify the data flow and reveal errors (for example, Filebeat lockfile issues).

If Filebeat reports a stale lock:

sudo rm -f /var/lib/filebeat/filebeat.lock # Why: remove abandoned lock

sudo systemctl restart filebeat

**4) Open the Dashboard**

* On Windows, open your browser and point to the Wazuh Dashboard address (use the IP/port your dashboard is bound to).
* Log in with the dashboard account you configured.

**Why:** Confirm events appear and use Discover/Security to filter and validate.

**5) Produce SSH Brute Force Events (on Debian VM 192.168.204.137)**

**Option A – hydra (recommended):**

sudo apt update && sudo apt install -y hydra

# Replace the target host accordingly. This rapidly fires incorrect passwords to produce auth failures.

hydra -l pi -P /usr/share/wordlists/rockyou.txt -t 4 ssh://192.168.204.138 -s 22

**Why:** Rapidly generates many sshd failures for Wazuh detection.

**Option B – basic loop (slower):**

for i in {1..15}; do ssh -o PreferredAuthentications=password -o PubkeyAuthentication=no pi@192.168.204.138 || true; done

**Why:** Produces repeated authentication failures without extra tooling.

**Evidence to capture:**

* /var/log/auth.log on 192.168.204.137 containing Failed password or Invalid user.
* Wazuh alerts in the Dashboard (search for sshd or authentication failure keywords).

**6) Show Sudo Timestamp Caching (on Debian VM 192.168.204.137)**

1. **Clear the sudo timestamp and authenticate once:**

sudo -k # Why: clear cached credentials

sudo ls # Why: prompt for password once

1. **Run multiple sudo commands without being prompted:**

for i in {1..10}; do sudo -n date; done

**Why:** -n succeeds while the timestamp is active; demonstrates multiple privileged actions after a single authentication.

**Evidence to collect:**

* /var/log/auth.log lines showing the initial sudo authentication and subsequent sudo calls without password.
* Wazuh alerts or records tied to sudo usage.

**7) Key Searches in Wazuh**

* **SSH failures (from Debian agent):** search Discover for sshd and filter by agent.name or host.
* **Sudo bursts (from Debian agent):** filter by program: sudo or check rules mentioning sudo.
* **Temporal analysis:** use the timeframe selector to zoom around your test activity.

**Why:** These checks validate that the system detected and logged the relevant behavior.

**8) Optional: Elastic Security Enhancements**

If you also have Elastic Security, follow these to complete the enhanced portion of the exercise.

**8.1 Ingest Boss of the SOC & Correlate 4625 → Outbound**

**Objective:** Link Windows Event ID 4625 (failed login) with outbound network activity during the same interval.

**Sample KQL for Discover/Security:**

# Failed logon

index:\* AND event.code:4625

# Outbound network events

index:\* AND network.direction:outbound AND destination.ip:\* AND (event.category:network or network.transport:\*)

**Correlation method:**

1. Filter event.code:4625, note host.name, user.name, source.ip, and @timestamp.
2. Look ±5–10 minutes for outbound events where host.name or source.ip matches that host.
3. Export and join records into a spreadsheet (CSV template below) to document associations.

**CSV template:**

Timestamp,Event ID,Source IP,Destination IP,Notes

2025-08-18 12:00:00,4625,192.168.204.137,8.8.8.8,Suspicious DNS request

**8.2 Anomaly Detection – High Egress (1 min)**

**Detection rule (KQL) – bytes\_out > 1MB within 1 minute:**

Index patterns: packetbeat-\*, firewall-\*, logs-\*

Custom query (KQL): network.bytes\_out >= 1048576 and event.duration <= 60000000

# Scheduling

Runs every: 1 minute

Look-back: 1 minute

**Why:** Surges in outbound bytes can indicate exfiltration. Adjust the field names to what your data source uses (e.g., destination.bytes, network.bytes).

**8.3 GeoIP Enrichment Pipeline**

Create an ingest pipeline to append GeoIP for destination.ip:

PUT \_ingest/pipeline/geoip-dest

{

"processors": [

{ "geoip": { "field": "destination.ip", "target\_field": "destination.geo" } }

]

}

Attach pipeline as default for an index:

PUT packetbeat-\*/\_settings

{

"index": { "default\_pipeline": "geoip-dest" }

}

**Why:** Adding country/city/ASN metadata makes triage and attribution faster.

**9) Example Correlation Table**

Use this for your README or Google Sheets and expand with your captured rows.

| **Timestamp** | **Event ID** | **Source IP** | **Destination IP** | **Notes** |
| --- | --- | --- | --- | --- |
| 2025-09-04 21:56:00 | 4625 | 192.168.204.137 | 192.168.204.138 | sudo and sudo caching |
| 2025-09-04 21:45:16 | 4625 | 192.168.204.137 | 192.168.204.138 | Brute Force |

**10) Evidence to Collect**

* **Screenshots:**
  + Wazuh Dashboard showing SSH failure timeline.
  + Wazuh Dashboard showing the sudo burst.
  + Elastic Discover for event.code:4625 and outbound flows.
  + Elastic Security alert showing the 1-minute egress spike.
* **Log snippets:** Selected /var/log/auth.log entries from the Debian agent (sanitize sensitive details).
* **Rule & pipeline configs:** JSON/YAML for the detection rule and ingest pipeline.

**11) Troubleshooting Notes**

* **No alerts for nmap scans:** Base Wazuh rules may not flag simple port scans — pair with IDS like Suricata or tune decoders/rules.
* **Filebeat lockfile issues:** delete the lock (/var/lib/filebeat/filebeat.lock) and restart Filebeat.
* **Missing Wazuh alerts:** check wazuh-alerts-\* index in the Dashboard/Discover and ensure time picker covers the test window.
* **No data visible:** confirm the agent shows connected in the manager and that the indexer is healthy.

**12) If Using Security Onion Instead**

If you swap to Security Onion, capture SSH brute force via Zeek and sudo via syslog/OSQuery; correlate in Kibana/OpenSearch using host and time. Suricata catches scans; Zeek conn.log helps measure egress. GeoIP enrichment is typically available via ingest pipelines.

**13) Reproduction Quickstart (Copy/Paste)**

# On Windows PowerShell

wsl -d Ubuntu

# On WSL Ubuntu – start services

sudo systemctl enable --now wazuh-manager wazuh-indexer wazuh-dashboard filebeat

# On Debian VM (192.168.204.137) – generate SSH failures

sudo apt update && sudo apt install -y hydra

hydra -l pi -P /usr/share/wordlists/rockyou.txt -t 4 ssh://192.168.204.138 -s 22

# On Debian VM (192.168.204.137) – demonstrate sudo caching

sudo -k; sudo ls # enter password once

for i in {1..10}; do sudo -n date; done

# On Windows browser – open dashboard

https://localhost:5601

**14) Repo Organization Suggestions**

* README.md → this document.
* documents/ → detection rule JSON, pipeline JSON, and scheduling notes.
* screenshots/ → PNG/JPG and sanitized logs.
* data/ → correlation CSV and any exported evidence.

**15) Future Enhancements**

* Integrate Suricata/Zeek (or Security Onion) to improve network telemetry.
* Tailor Wazuh rules for sshd and sudo with custom thresholds for noisy hosts.
* Automate evidence extraction with a Python script using the Kibana/API to pull search results and export artifacts.