Jelly Concepcion

Aspiring Cyber Defense Incident Responder | Future Chief Information Security Officer (CISO) | Passionate About Cybersecurity & Threat Detection.

Email: jellydizonconcepcion@gmail.com

GitHub: https://github.com/jellyconcepcion/blue-team-homelab

LinkedIn: https://www.linkedin.com/in/jellyconcepcion/

Splunk Project 01 – SOC Home Lab

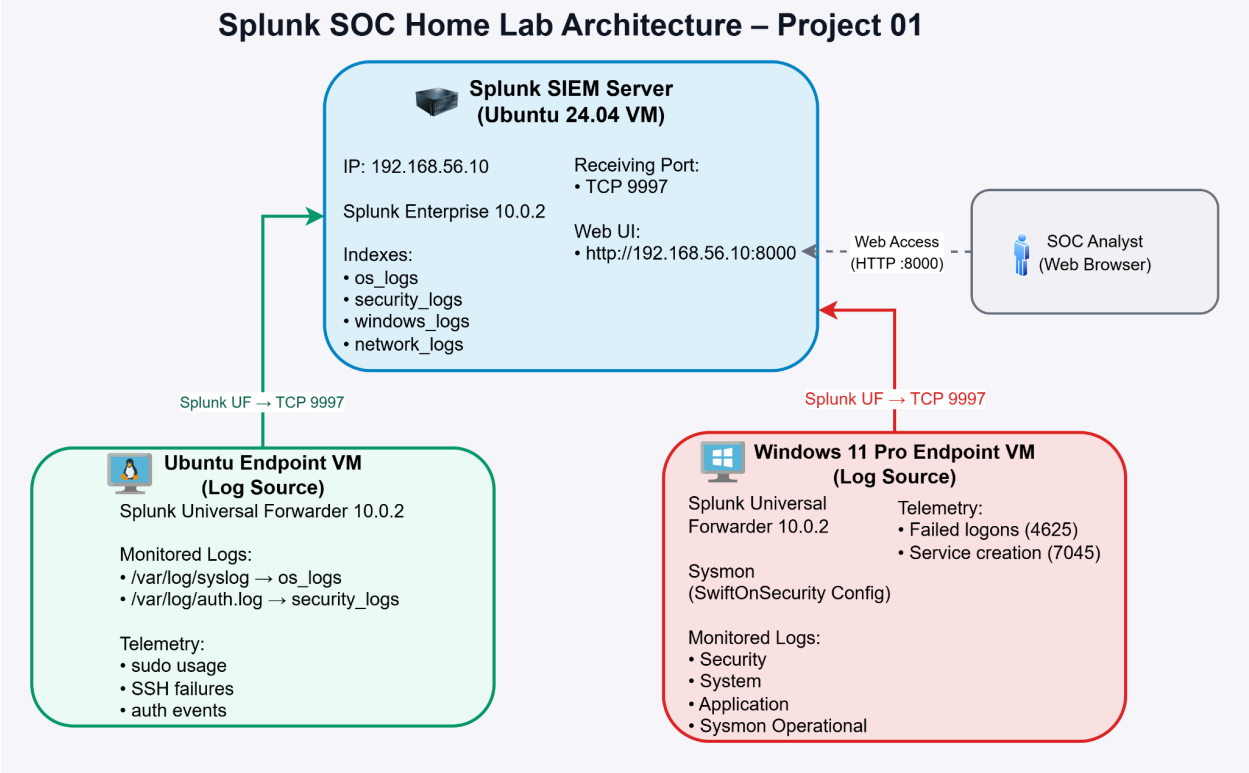
Log Collection, Forwarding, and First Detections

The purpose of this lab is to simulate a professional SOC environment for log collection, analysis, and basic detection using Splunk Enterprise. It demonstrates:

- Installation and configuration of Splunk Enterprise on Ubuntu 24.04
- Deployment of Splunk Universal Forwarders on both Ubuntu and Windows endpoints
- Log ingestion from multiple sources (OS logs, Security logs, Sysmon)
- Basic detections: failed logins, sudo usage, new Windows service creation
- Professional documentation, snapshots, and SHA verification for supply-chain integrity

---

This lab consists of a centralized Splunk SIEM server deployed on Ubuntu 24.04, receiving logs from both Ubuntu and Windows 11 endpoints via Splunk Universal Forwarders 10.0.2. Windows telemetry is enhanced using Sysmon with the SwiftOnSecurity configuration. All log sources forward data over TCP 9997, while SOC analysts access Splunk via the web interface on port 8000.



Key Terms & Concepts (Splunk Project 01)

This section provides essential terminology to understand how logs flow, how Splunk processes data, and how detections are validated in this SOC lab.

1️⃣ SIEM & SOC Fundamentals

SIEM (Security Information and Event Management)
A platform that collects, indexes, correlates, and searches security logs from multiple sources to support threat detection, investigation, and incident response.
Example: Splunk Enterprise

SOC (Security Operations Center)
The function responsible for monitoring security events, triaging alerts, investigating incidents, and responding to threats using SIEM data.

Telemetry
Detailed behavioral data generated by systems (processes, logins, network connections) that enables detection and investigation beyond basic logs.

2️⃣ Log Sources & Endpoints

Endpoint
A system that generates logs, such as Windows, Linux, or servers running applications.

Log Source
The origin of log data (e.g., Windows Event Logs, Linux auth logs, Sysmon events).

Syslog
A standardized logging protocol and message format commonly used by Unix/Linux systems.

Sysmon (System Monitor)
A Windows telemetry tool that logs detailed security-relevant events (process creation, network connections, file creation) that Windows does not log by default.

3️⃣ Splunk Architecture Components

Splunk Enterprise (Indexer / SIEM Server)
The core Splunk component that receives, parses, indexes, and stores log data for searching and analysis.

Splunk Universal Forwarder (UF)
A lightweight agent installed on endpoints that reads logs and securely forwards them to Splunk Enterprise.

Indexer
The Splunk component responsible for storing indexed data and responding to search queries.

4️⃣ Log Processing Concepts (How Splunk Understands Data)

Source
The origin path or input of the data (e.g., `/var/log/auth.log`, `WinEventLog:Security`).

Sourcetype
Defines the format and structure of the log data, allowing Splunk to correctly parse fields.

Parsing
The process by which Splunk interprets raw data using its source and sourcetype to extract searchable fields.

Index
A logical storage location in Splunk where events are stored.
Examples used in this lab:

- `os_logs` — Linux system logs
- `security_logs` — Authentication, sudo, login failures
- `windows_logs` — Windows Event Logs & Sysmon events
- `network_logs` — Reserved for future firewall / IDS data

5️⃣ Linux Logging Components

Daemon
A background service that runs continuously without user interaction.

rsyslog
A Linux daemon that implements the syslog protocol and forwards logs reliably (TCP-capable) to external systems like SIEMs.

rsyslogd
The running rsyslog daemon process responsible for collecting and forwarding logs.

6️⃣ Splunk Data Flow & Networking (Planes & Ports)

Data Plane
Used for log ingestion and telemetry flow.

- **Port 9997** — Splunk's standard receiving port for Universal Forwarders

- **Port 514 (TCP)** — Standard syslog port used by rsyslog for log forwarding

Control Plane
Used for management, configuration, and API communication.

- **Port 8089** — Splunk management API (Splunk-to-Splunk and UF control)

User Plane
Used for analyst access and visualization.

- **Port 8000** — Splunk Web UI (Search & Reporting interface)

7️⃣ Detection & Validation Concepts

Baseline Validation
Confirming that logs are successfully ingested and searchable before building detections.

Detection Logic
Search queries designed to identify suspicious or security-relevant behavior (e.g., failed logins, privilege escalation).

Event
A single log record stored in Splunk.

Search & Reporting
Splunk's primary interface for querying, validating, and analyzing indexed data.

Steps / Workflow Summary

Phase	VM	Task	Index	Screenshot / Snapshot

Phase 1	Ubuntu	Install Splunk Enterprise	n/a	Splunk-dpkg-Install.png
Phase 4	Ubuntu	Create Indexes	os_logs, windows_logs, security_logs, network_logs	Splunk-Index-*.png
Phase 6	Windows	Install UF + Sysmon	windows_logs	UF-Forwarding-to-Splunk.png, Sysmon-Config-Loaded.png
Phase 7	Ubuntu	Install UF & configure inputs/outputs	os_logs, security_logs	Ubuntu-UF-*.png
Phase 8	Both	First detections	security_logs, windows_logs	Windows-*.png
Final Snapshot	Both	Full functional lab	All	Splunk-Project-01-Final

---

Follow the same step-by-step process on how to create Ubuntu VM using the ubuntu-24.04.3-live-server-amd64.iso on VirtualBox 7.1.12 in my LAB-Ubuntu-01-Setup.pdf uploaded on my GitHub https://github.com/jellyconcepcion/blue-team-homelab/blob/main/01-lab-setup/01-ubuntu-siem/LAB-Ubuntu-01-Setup.pdf

Note: Remember to change all the words “Wazuh” into “Splunk” as the following steps will use Splunk 10.0.2 instead of Wazuh.

Snapshot Name: Clean-Ubuntu-Install (When to Take It: After Ubuntu setup)

Phase 1:

✅ How to Download & Install Splunk Enterprise (Free) on Ubuntu 24.04

STEP 1 — Create a Free Splunk Account

Splunk requires a free account to download the installer.

👉 Go to: **splunk.com → Free Trials → Splunk Enterprise. Link: https://www.splunk.com/en_us/download/splunk-enterprise.html**
Choose **Linux (.deb)** when selecting package type.

You'll get a `.deb` link like:

wget -O splunk-10.0.2-e2d18b4767e9-linux-amd64.deb "https://download.splunk.com/products/splunk/releases/10.0.2/linux/splunk-10.0.2-e2d18b4767e9-linux-amd64.deb"

STEP 2 — Download Splunk Into Your Ubuntu VM

Use **wget** and paste the official link you copied.

If you **don't have `wget` installed yet**, you need to install it first.

Step by step:

STEP 3: Install `wget` on your Ubuntu VM

```
sudo apt update
```

```
sudo apt install -y wget
```

Screenshot: (Terminal showing:

- `sudo apt update`
- `sudo apt install -y wget`

Filename: `Ubuntu-Install-wget.png`)

Snapshot Name: Before-Splunk-Install (When to Take It: After updating & installing wget)

Note: (**Do NOT use `/tmp` for installers if you need to keep them after reboot.**

Professionally, installers are stored in a **persistent directory**, *not* `/tmp`.

Recommended for labs and documentation:
Use:

```
/opt/installers
```

Why?

- `/opt` is for “optional software”
- Organizing installers in `/opt/installers` looks professional in screenshots
- Files stored here **do not delete** after reboot)

Create the professional installer directory

Run:

```
sudo mkdir -p /opt/installers
```

```
sudo chmod 777 /opt/installers
```

This ensures the folder is permanent and easy to write to.

Move into the new directory

```
cd /opt/installers
```

Download Splunk (get this on https://www.splunk.com/en_us/download/splunk-enterprise.html)

```
wget -O splunk-10.0.2-e2d18b4767e9-linux-amd64.deb "https://download.splunk.com/products/splunk/releases/10.0.2/linux/splunk-10.0.2-e2d18b4767e9-linux-amd64.deb"
```

Screenshot: (Terminal showing:

- `cd /opt/installers`
- `wget -O splunk-10.0.2-xxx.deb "<official link>"`

Filename:
`Splunk-wget-Download.png`)

STEP 4: Install Splunk

```
sudo dpkg -i splunk-10.0.2-e2d18b4767e9-linux-amd64.deb
```

It will install to `/opt/splunk`.

Screenshot: (Terminal showing:

- `sudo dpkg -i splunk-10.0.2-xxx.deb`

Filename:
`Splunk-dpkg-Install.png`)

Snapshot Name: After-Splunk-Install (When to Take It: After dpkg install succeeds)

Start Splunk and set admin credentials

`sudo /opt/splunk/bin/splunk start --accept-license`

- Set **username**: `splunkadmin`
- Set **password**: Admin123!

Screenshot: (License acceptance (terminal)

Filename: `Splunk-License-Accept.png`)

STEP 5: Enable auto-start on boot

`sudo /opt/splunk/bin/splunk enable boot-start`

Screenshot: (Terminal showing:

- `sudo /opt/splunk/bin/splunk enable boot-start`

Filename:
`Splunk-Boot-Start.png`)

Open Splunk Web UI

From your host machine (or VM browser if you installed a GUI):

`http://192.168.56.10:8000`

Log in with the **admin username/password** you set.

Take **2 screenshots**:

1. Browser loading: `http://192.168.56.10:8000`
   Filename: `Splunk-Web-Login-Page.png`

2. Splunk Dashboard after login
   Filename: `Splunk-Home-Dashboard.png`

Snapshot Name: Splunk-Fully-Set-Up (When to Take It: After successful login to dashboard)

Phase 2:

STEP 6: Splunk Installer File Stored in /opt/installers

After running:

```
ls -lh /opt/installers
```

Shows:

- The Splunk `.deb` file
- Size
- Timestamp

You can confirm the file (.deb) is there:

```
ls -lh
```

SHA512 Verification

Download the SHA512 on https://www.splunk.com/en_us/download/splunk-enterprise.html

This is the most important screenshot.

Take when:

You run this inside `/opt/installers`:

```
sha512sum splunk-10.0.2-e2d18b4767e9-linux-amd64.deb
```

Screenshot must show:

1. **The terminal output** of the SHA512 hash
2. **The Splunk official SHA512 hash** from the Splunk website

You can do this in two ways:

- Side-by-side windows

Filename:

`Splunk-SHA512-Verification.png`

🔐 Phase 3 — Secure & Prepare the Splunk Server (NO log ingestion yet)

This phase proves you understand **production hygiene**, not just “it runs”.

✅ STEP 7 — Create a Dedicated Splunk Service User (Optional but Professional)

> ⚠️ Splunk already created a `splunk` user automatically.
> **Do NOT recreate it.**
> Just **verify it exists**.

Command

```
id splunk
```

Screenshot

- **TAKE**
  - Terminal output showing `uid`, `gid`
  - Filename:
    Splunk-Service-User-Verified.png

🔐 STEP 8 — Verify Splunk Directory Permissions

This shows you understand **least privilege**.

Commands

```
ls -ld /opt/splunk

ls -ld /opt/splunk/var
```

Expected

- Owner: `splunk`

- Group: `splunk`

Screenshot

- **TAKE**
    - Terminal showing both commands + output
    - Filename:
      Splunk-Directory-Permissions.png

🔐 STEP 9 — Confirm Splunk Service Status

Command

```
sudo systemctl status splunk
```

Screenshot

- **TAKE**
    - Service running
    - Filename:
      Splunk-Service-Running.png

To return to:

```
labadmin@splunk-lab-01:~$
```

👉 **Press:**

```
q
```

That's it.

🧱 Phase 4 — Create SOC Indexes (STILL Project 1)

You're now shaping Splunk for **security data**, not random logs.

✅ STEP 10 — Create Security Indexes

On http://192.168.56.10:8000/

Go to:

`Settings → Indexes → New Index`

Create **ONE AT A TIME**:

Index Name	Purpose
`os_logs`	Linux system logs
`windows_logs`	Windows Event Logs
`security_logs`	Auth, sudo, failures
`network_logs`	Firewall / IDS later

Screenshots

TAKE for each index creation screen

- Show:
    - Index name
    - Default paths
- Filenames:
    - `Splunk-Index-os_logs.png`
    - `Splunk-Index-windows_logs.png`
    - `Splunk-Index-security_logs.png`
    - `Splunk-Index-network_logs.png`

🔍 Phase 5 — Prepare for Log Ingestion (NO forwarding yet)

✅ STEP 11 — Enable Receiving Port (9997)

Location

`Settings → Forwarding and receiving → Configure receiving → New`

Port

`9997`

Screenshot

- **TAKE**
    - Receiving port configured
    - Filename:
      Splunk-Receiving-Port-9997.png

Phase 6 — Windows 11 Endpoint (Splunk Forwarding)

Objective:

- Prepare your Windows VM to send logs to your Splunk server.
- Professional approach: everything documented and rollback-ready.

Step 12 — Install Splunk Universal Forwarder (UF)

1️⃣ Download the Splunk UF on Windows VM

- Go to Splunk website → Downloads → Universal Forwarder → Windows (.msi). Link: https://www.splunk.com/en_us/download/universal-forwarder.html
- Save to a **persistent folder**, e.g., `C:\Installers`
- **Professional note:** Do **not** use Desktop or Downloads.

2️⃣ Verify SHA256/512 (critical!)

- Use `CertUtil` on Windows:

```
CertUtil -hashfile
C:\Installers\splunkforwarder-10.0.2-e2d18b4767e9-windows-x64.msi
SHA512
```

- Download the SHA512 on splunk website (https://www.splunk.com/en_us/download/universal-forwarder.html) then compare the two.

✅ **Screenshot**

- SHA output **side-by-side with official SHA**

- Filename: `UF-SHA512-Verification.png`

3️⃣ **Install UF**

- Run `.msi` with default installation path: `C:\Program Files\SplunkUniversalForwarder`

- Set service to run **Local System** or **splunkuser** (professional labs often create a dedicated user, optional)
- On Deployment Server set the IP to ubuntu VM static IP 192.168.56.10 and follow the default port 8089.
- On receiving the indexer, set the IP to ubuntu VM static IP 192.168.56.10 and follow the default port 9997.

- During install, do **not** configure forwarder yet. That will be done later.

✅ **Screenshot**

- UF installed (Windows Services showing SplunkForwarder running)

Windows Services

1. Press `Win + R` → type `services.msc` → press Enter.

2. In the Services window, look for:

   `SplunkForwarder`
3. Confirm **Status = Running** and **Startup Type = Automatic** (or `Manual` if you left default).
- Filename: `UF-Installed-Service-Verified.png`

EXACT NEXT STEPS

Download the Sysmon64.exe on https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon

🔐 STEP 13A — Verify Sysmon Binary Integrity (SHA Evidence)

Even though it's Microsoft, **SOC discipline = verify binaries**.

Command (PowerShell):

```
CertUtil -hashfile "C:\Installers\Sysmon\Sysmon64.exe" SHA512
```

Screenshot

✅ **TAKE**

- SHA512 output
- Filename:
  Sysmon-SHA512-Verification.png

📄 STEP 13B — Download Sysmon Config (NO INSTALL YET)

- Download **SwiftOnSecurity sysmon config**

📥 How to Download It

Direct Download

1. Open this **direct raw file link**:

```
https://raw.githubusercontent.com/SwiftOnSecurity/sysmon-config/master/sysmonconfig-export.xml
```

2. When the XML file opens in your browser:

- Right-click → **Save As**
- Save it as:

  `C:\Installers\Sysmon\sysmon-config.xml`

❌ Do NOT edit yet

🧠 STEP 13C — Understand What You're Enabling (Critical Knowledge)

You should know this **before install**:

Event	Why SOCs Care
Process Create	Detect malware execution
Network Connect	Detect C2 traffic
File Create	Detect payload drops
Registry Change	Detect persistence
Image Load	Detect DLL hijacking

📌 You are enabling **telemetry**, not detection yet.

🛠️ STEP 13D — Install Sysmon WITH Config (Admin PowerShell)

Make sure the sysmon-config.xml and Sysmon64.exe are in the same folder.

Once the files are in the same folder:

```
cd C:\Installers\Sysmon
.\sysmon64.exe -accepteula -i sysmon-config.xml -h md5,sha512 -n
```

This:

- Installs Sysmon
- Loads config immediately
- Enables SHA512 logging
- Prevents default noisy logging

2️⃣ Sysmon Service Running

Command:

```
Get-Service Sysmon64
```

📁 Filename:
Sysmon-Service-Running.png

3️⃣ Config Loaded Verification

Command:

```
sysmon64.exe -c
```

Screenshot should show:

Example:

```
Configuration file: C:\Installers\sysmon-config.xml
```

✔️ Proves:

- Sysmon is NOT running with default config
- A custom config is actively loaded

📁 Filename:
Sysmon-Config-Loaded.png

Step 14 — Configure Forwarder to Splunk

- Connect Windows UF to your Splunk server IP: `192.168.56.10:9997`
- Command line (PowerShell):

```
& 'C:\Program Files\SplunkUniversalForwarder\bin\splunk.exe' add forward-server 192.168.56.10:9997 -auth splunkadmin:Admin123!

& 'C:\Program Files\SplunkUniversalForwarder\bin\splunk.exe' enable boot-start
```

Optional verification command (recommended to run after the above):

```
& 'C:\Program Files\SplunkUniversalForwarder\bin\splunk.exe' list forward-server
```

- Confirms the forwarder is connected

✅ **Screenshot**

- Terminal showing successful connection
- Filename: `UF-Forwarding-to-Splunk.png`

Phase 7 — Ubuntu Log Forwarding

Objective:

- Send Linux logs (auth, sudo, syslog) to Splunk.

Step 15: Enable `ufw`

```
sudo ufw enable
```

- You'll get a confirmation like: `Firewall is active and enabled on system startup.`

Step 16: Allow firewall traffic

Allow port 8000:

```
sudo ufw allow 8000/tcp
```

```
sudo ufw reload
```

```
sudo ufw status
```

Step 17: Restart Splunk input (optional but recommended)

Either restart Splunk or just confirm the input is active:

```
sudo /opt/splunk/bin/splunk restart
```

```
sudo /opt/splunk/bin/splunk status
```

Note: Make sure splunk is running after restart.

🔐 STEP 18 — Install Splunk Universal Forwarder (Ubuntu)

🎯 Goal

- Install Splunk UF properly
- Prove installer integrity with SHA512
- Keep evidence clean for GitHub / SOC review

🔹 STEP 18A — Download UF into `/opt/installers`

Where to run this

➡️ Ubuntu forwarding VM terminal

Commands (run exactly in this order)

```
cd /opt/installers
```

```
wget -O splunkforwarder-10.0.2-e2d18b4767e9-linux-amd64.deb "https://download.splunk.com/products/universalforwarder/releases/10.0.2/linux/splunkforwarder-10.0.2-e2d18b4767e9-linux-amd64.deb"
```

📸 Screenshot #1 — UF Downloaded

TAKE

Must show:

- `cd /opt/installers`
- `wget` command completed
- Filename visible

Filename

`Ubuntu-UF-Download.png`

🔹 STEP 18B — SHA512 Verification (CRITICAL)

Download the SHA512 on https://www.splunk.com/en_us/download/universal-forwarder.html then open it on notepad.

Command (On Ubuntu VM)

```
sha512sum splunkforwarder-10.0.2-e2d18b4767e9-linux-amd64.deb
```

What to do

- Open Splunk website **SHA512 value**
- Compare visually
- Confirm they match

📸 Screenshot #2 — SHA512 Evidence

TAKE (MANDATORY)

Must show:

- Terminal SHA512 output
- Official Splunk SHA512 (side-by-side browser or text)

Filename

`Ubuntu-UF-SHA512-Verification.png`

📌 This is **professional supply-chain evidence** — excellent for SOC review.

🔹 STEP 18C — Install the Universal Forwarder

Command

```
sudo dpkg -i splunkforwarder-10.0.2-e2d18b4767e9-linux-amd64.deb
```

📌 Installs to:

`/opt/splunkforwarder`

📸 Screenshot #3 — UF Installed

TAKE

Must show:

- `dpkg -i` command
- Successful install (no fatal errors)

Filename

`Ubuntu-UF-dpkg-Install.png`

🔗 STEP 19 — Configure UF to Send Logs to Splunk (9997)

🎯 Goal

- Start UF
- Accept license
- Point it to Splunk receiver
- Enable persistence

🔹 STEP 19A — Start UF & Accept License

Command

```
sudo /opt/splunkforwarder/bin/splunk start --accept-license
```

- Set admin credentials (use something simple but document it)
- Example:
    - User: `splunkuf`
    - Password: `Admin123!`

📸 Screenshot #4 — UF License Accepted

TAKE

Must show:

- License accepted
- UF started successfully

Filename

`Ubuntu-UF-License-Accepted.png`

If you encounter:

```
ERROR: mgmt port [8089] port is already bound.

Splunk needs to use this port.

Would you like to change ports? [y/n]:
```

Why this happened

- **Splunk Enterprise** already uses **management port 8089**
- **Splunk Universal Forwarder (UF)** also tries to use **8089 by default**
- Two Splunk components **cannot share the same management port**

STEP 19B - EXACTLY WHAT TO DO (DO NOT SKIP)

When prompted:

```
Would you like to change ports? [y/n]:
```

👉 Type:

```
y
```

When asked for the new port:

```
Please enter new mgmt port:
```

👉 Type:

```
18089
```

Splunk UF will then:

- Finish startup
- Bind to **mgmt port 18089**
- Run independently from Splunk Enterprise

STEP 19C - AFTER UF SUCCESSFULLY STARTS — DO THIS NEXT

🔹 Verify UF is Running

Run:

```
sudo /opt/splunkforwarder/bin/splunk status
```

Expected:

```
splunkd is running
```

📸 Screenshot

- Filename:

`Ubuntu-UF-Status-Running.png`

STEP 19D — Stop the Universal Forwarder

Run this on the Ubuntu forwarding VM:

```
sudo /opt/splunkforwarder/bin/splunk stop
```

Expected:

```
Stopping splunkd...
Done.
```

📸 Screenshot

TAKE

Filename:

`Ubuntu-UF-Stopped.png`

1️⃣ Disable the old init.d boot-start

Run:

```
sudo /opt/splunkforwarder/bin/splunk disable boot-start
```

Expected output:

```
Stopping boot-start for /opt/splunkforwarder
```

```
Done.
```

📸 Screenshot:
Filename: `Ubuntu-UF-Boot-Start-Disabled.png`

2️⃣ Verify no old init.d service exists (optional but professional)

```
ls /etc/init.d/ | grep splunk
```

You should see either nothing or just confirmation that the service is gone.

3️⃣ Enable systemd boot-start properly

Now run:

```
sudo /opt/splunkforwarder/bin/splunk enable boot-start
```

This time, it should **create a proper systemd service** without complaining.

📸 Screenshot:
Filename: `Ubuntu-UF-Boot-Start-Enabled.png`

4️⃣ Start the Splunk UF service again

```
sudo /opt/splunkforwarder/bin/splunk start
```

📸 Screenshot:
Filename: `Ubuntu-UF-Started.png`

5️⃣ Verify systemd service

Before verifying the service, check the splunk forwarder name:

```
ls -l /etc/systemd/system/ | grep -i splunk
```

You should see something like:

```
-rw-r---- 1 root root 830 Dec 14 04:21 SplunkForwarder.service
```

Correct command

You need to use **exact capitalization**:

```
sudo systemctl status SplunkForwarder.service
```

📸 Screenshot:

Filename: `Ubuntu-UF-Systemd-Service-Running.png`

🔹 STEP 20 — Configure UF Inputs (THIS IS THE MISSING STEP)

20A — Create inputs.conf for OS logs

Run on the **Ubuntu forwarding VM**:

```
sudo mkdir -p /opt/splunkforwarder/etc/system/local
```

```
sudo vim /opt/splunkforwarder/etc/system/local/inputs.conf
```

Press `i` and paste **exactly this**:

```
[monitor:///var/log/syslog]

index = os_logs

sourcetype = syslog


[monitor:///var/log/auth.log]

index = os_logs

sourcetype = linux_secure
```

Save and exit:

```
Esc → :wq → Enter
```

📸 **Screenshot (TAKE)**

- Show the `inputs.conf` content
- Filename:

```
Ubuntu-UF-Inputs-Configured.png
```

20B — Restart UF to apply inputs

```
sudo systemctl restart SplunkForwarder.service
```

Verify:

```
sudo systemctl status SplunkForwarder.service
```

📸 **Screenshot (TAKE)**

- Must show `active (running)`
- Filename:

`Ubuntu-UF-Restarted-With-Inputs.png`

🔷 STEP 21 — Configure UF Forwarding Destination (MISSING STEP)

Create outputs.conf

Run on the **Ubuntu forwarding VM**:

```
sudo vim /opt/splunkforwarder/etc/system/local/outputs.conf
```

Press `i` and paste **exactly this**:

```
[tcpout]
defaultGroup = splunk_indexer


[tcpout:splunk_indexer]
server = 192.168.56.10:9997


[tcpout-server://192.168.56.10:9997]
```

✅ This tells UF:

- Use TCP
- Send data to Splunk Enterprise
- Use receiving port 9997

Save and exit:

`Esc → :wq → Enter`

📸 Screenshot (TAKE)

Must show:

- outputs.conf content

Filename:

`Ubuntu-UF-Outputs-Configured.png`

🔹 STEP 22 — Restart UF (CRITICAL)

```
sudo systemctl restart SplunkForwarder.service
```

Verify:

```
sudo systemctl status SplunkForwarder.service
```

📸 Screenshot:

`Ubuntu-UF-Restarted-With-Outputs.png`

🔹 STEP 23 — Generate Test Log on Ubuntu VM

```
logger "Test log for Splunk"
```

Wait **10–20 seconds**.

🔹 STEP 24 — CONFIRM IN SPLUNK

Search in Splunk:

```
index=os_logs "Test log for Splunk"
```

Time range:

```
Last 15 minutes
```

✅ You should now see the event.

📸 Screenshot (MANDATORY):

`Ubuntu-UF-Test-Log-Visible.png`

✅ Phase 8 — (Starting STEP 25)

🔢 STEP 25 — Configure Windows Inputs

Steps to Create `inputs.conf` in `local` Folder (only if it doesn't exist yet)

1. **Open Notepad as Administrator**
    - Press **Win**, type `Notepad` → Right-click → **Run as administrator**
2. **Create a New File**
    - Since `inputs.conf` doesn't exist, just type the **config contents** in the Notepad window
    - Example minimal config for Windows event logs:

```
[WinEventLog://Security]

disabled = 0

index = windows_logs


[WinEventLog://System]

disabled = 0

index = windows_logs


[WinEventLog://Application]

disabled = 0

index = windows_logs


[WinEventLog://Microsoft-Windows-Sysmon/Operational]

disabled = 0

index = windows_logs
```

3. **Save as `inputs.conf`**

- File → Save As
- Make sure the **Save as type** is **All Files (*.*)**
- File name: `inputs.conf`
- Folder: `C:\Program Files\SplunkUniversalForwarder\etc\system\local\`

Screenshots to TAKE

inputs.conf content (Notepad or PowerShell `type`)

- **Filename:**
  `Windows-inputs.conf.png`

4. Restart Splunk Forwarder Service (required to apply changes)

Use Splunk CLI (Most Reliable for UF)

1. Open **Admin PowerShell**
2. Navigate to UF bin folder:

```
cd "C:\Program Files\SplunkUniversalForwarder\bin"
```

3. Restart Splunk Forwarder using its own command:

```
.\splunk.exe stop

.\splunk.exe start
```

Screenshots to TAKE

Forwarder restarted

- PowerShell showing restart success

- Filename:
  `Win-UF-Restarted.png`

Back on **Windows VM (Admin PowerShell)**:

```
cd "C:\Program Files\SplunkUniversalForwarder\bin"
```

Then:

```
.\splunk.exe list forward-server
```

Expected:

```
Active forwards:

    192.168.56.10:9997
```

If it showed:

Active forwards:

    None

Then do the next one.

Check via Network Stack (MOST RELIABLE)

On **Ubuntu VM**:

```
sudo ss -tulnp | grep 9997
```

✅ EXPECTED OUTPUT:

You should see something like:

```
LISTEN 0 128 0.0.0.0:9997 users:(("splunkd",pid=XXXX,fd=YY))
```

Verify via btool (Splunk-Authoritative)

This confirms Splunk **configuration**, not just sockets.

```
sudo /opt/splunk/bin/splunk btool inputs list tcp
```

Expected:

```
[tcp://9997]

disabled = 0
```

If it didn't show "[tcp://9997]"

We will **manually and explicitly** define the receiving port.

🔢 1 — Edit the Correct File (Ubuntu VM)

Open the file **as root**:

```
sudo nano /opt/splunk/etc/system/local/inputs.conf
```

If the file does not exist, nano will create it — that's expected.

🔢 2 — Add THIS EXACT Stanza (Nothing Else)

Paste **exactly** this at the bottom:

```
[tcp://9997]

disabled = 0
```

Rules:

- No extra spaces
- No duplicate `[tcp]` stanzas
- No comments needed

Save and exit:

- `CTRL + O` → Enter
- `CTRL + X`

📸 Screenshot (TAKE):
Splunk-inputs-tcp-9997-Manual.png

This screenshot is **gold** in interviews — it shows you understand Splunk config hierarchy.

🔢 3 — Restart Splunk (Mandatory)

```
sudo /opt/splunk/bin/splunk restart
```

📸 Screenshot:
Splunk-Restart-After-Inputs-Change.png

🔢 4 — Verify with btool (THE SOURCE OF TRUTH)

```
sudo /opt/splunk/bin/splunk btool inputs list tcp
```

✅ NOW you MUST see:

```
[tcp://9997]
```

```
disabled = 0
```

📸 Screenshot (MANDATORY):
Splunk-btool-TCP-9997-Confirmed.png

Confirm connectivity from Windows VM → Ubuntu VM (Splunk Server)

1. Open **PowerShell** on Windows VM and run:

```
Test-NetConnection -ComputerName 192.168.56.10 -Port 9997
```

If the result fails like this “WARNING: TCP connect to (192.168.56.10 : 9997) failed”, do the steps below.

1️⃣ Check Ubuntu VM firewall (ufw)

On your **Ubuntu (Splunk) VM**, run:

```
sudo ufw status
```

- If it’s active, allow TCP 9997:

```
sudo ufw allow 9997/tcp
```

```
sudo ufw reload
```

- Optional: check rules applied:

```
sudo ufw status numbered
```

Screenshot: `9997-TCP-Allowed.png`

2️⃣ Check Splunk is listening on all interfaces

Run:

```
sudo ss -tulpn | grep 9997
```

- You should see something like:

```
0.0.0.0:9997 0.0.0.0:* users:(("splunkd",pid=xxxx,fd=x))
```

- **0.0.0.0** = listening on all interfaces (correct)
- Then restart Splunk:

```
sudo /opt/splunk/bin/splunk restart
```

3️⃣ Check Windows VM firewall

On **Windows 11 VM**:

1. Open **Windows Defender Firewall → Advanced Settings**
2. Go to **Outbound Rules**
3. Add a **new rule**:
    - Type: **Port**
    - TCP, Specific Port: `9997`
    - Action: **Allow**

- Profile: Domain/Private/Public
- Name: `Allow Splunk TCP 9997 Outbound`

`Screenshot: Win-Firewall-Rule-TCP9997.png`

4. Then retry:

```
Test-NetConnection -ComputerName 192.168.56.10 -Port 9997
```

- ✅ Should return `TcpTestSucceeded : True`

Screenshot: `Win-Test-NetConnection-TCP9997.png`

🔍 STEP 26 — Validate Windows Log Ingestion

Now go to **Splunk Web → Search & Reporting**:

```
index=windows_logs
```

What You SHOULD See

- Events now flowing
- Host = Windows 11 VM
- Sourcetypes like:
  - `WinEventLog:Security`
  - `XmlWinEventLog:Microsoft-Windows-Sysmon/Operational`

📸 Screenshot

✅ TAKE
Filename:
Windows-Baseline-Logs.png

STEP 27 — Detection #1: Failed Windows Logins

Generate Event (Windows VM)

- Lock screen
- Enter wrong password 3–5 times
- Log in correctly

Splunk Search

```
index=windows_logs EventCode=4625
```

Screenshot

✅ TAKE
Filename:
Phase8-Windows-Failed-Logons.png

Note: If you ever see “Error: Failed to read compression bits from bucket=windows_logs... Results may be incomplete!”, that is a **classic Splunk single-node lab issue**, not a failure of your detection. But if you want the error to go away, just generate events again and make sure to change the time range to only show those new events.

STEP 28 — Detection #2: New Windows Service Creation

Generate Event

Admin CMD:

```
sc create SOC_Test_Service binPath= "cmd.exe /c exit"
```

(Optional cleanup)

```
sc delete SOC_Test_Service
```

Splunk Search

```
index=windows_logs EventCode=7045
```

Screenshot

✅ TAKE
Filename:
Windows-New-Service-Creation.png

Step 29 — Validate Ubuntu Security Logs

Update `inputs.conf` for Security Logs

Professional approach: separate indexes by log purpose.

Edit `/opt/splunkforwarder/etc/system/local/inputs.conf` (or equivalent on your Ubuntu UF):

```
sudo vim /opt/splunkforwarder/etc/system/local/inputs.conf
```

Press `i` and paste **exactly this:**

```
# Security logs
```

```
[monitor:///var/log/auth.log]

disabled = 0

index = security_logs

sourcetype = linux_secure


# Optional general logs

[monitor:///var/log/syslog]

disabled = 0

index = os_logs

sourcetype = syslog
```

Save and exit:

`Esc → :wq → Enter`

Restart Splunk UF on Ubuntu VM

```
sudo /opt/splunkforwarder/bin/splunk restart
```

Screenshot to take:

- Terminal output showing restart success
- Filename: **Phase8-Ubuntu-UF-Restarted.png**

Generate Test Security Logs

For Step 29, you want **auth/security events**:

```
# Failed sudo

sudo -k

sudo ls /root


# Failed login attempt (simulate)

ssh wronguser@localhost
```

Or just type “sudo reboot” then enter the wrong password once when you log in.

> Professional tip: simulate **typical SOC events**, don’t spam system logs.

Validate Logs in Splunk Web

1. Open **Splunk → Search & Reporting**
2. Run:

```
index=security_logs
```

3. Confirm:

- Events exist
- Host = Ubuntu VM
- Sourcetype = `linux_secure`
- Shows failed logins / sudo attempts (your test events)

Screenshot to take:

- Search results showing multiple events
- Filename: **Phase8-Ubuntu-Security-Baseline.png**

STEP 30 — Detection #3: Failed SSH Login (Ubuntu)

Generate from Ubuntu VM:

```
ssh wronguser@192.168.56.10
```

Splunk Search:

```
index=security_logs "Failed password" OR "authentication failure"
```

Screenshot

✅ TAKE
Filename:
Ubuntu-Failed-SSH.png

STEP 31 — Detection #4: sudo Usage (Ubuntu)

Generate:

```
sudo ls /root
```

Splunk Search:

```
index=security_logs sudo
```

Screenshot

✅ TAKE
Filename:
Ubuntu-Sudo-Usage.png

Notes:

- All SHA512 hashes verified for supply-chain integrity (Splunk, UF, Sysmon).
- All VMs have snapshots at major milestones.
- Inputs and outputs configured according to SOC best practices.
- Firewall rules verified (TCP 9997, 8000) and forwarding confirmed.
- Lab fully functional for initial SOC detections and can be extended for network/firewall/IDS logs.

Create 2 “Final Verification Snapshot” on both Ubuntu and Windows VM on VBox

Snapshot Name:

`Splunk-Project-01-Final`

When to Take:

- All Windows & Ubuntu detections validated
- All Splunk searches producing expected events
- Forwarders running
- Sysmon running on Windows
- Inputs on Ubuntu & Windows configured

Splunk Dashboard Screenshot — `Splunk-Dashboard-Overview.png`

Goal: Show a single Splunk search/dashboard that demonstrates your lab is actively ingesting and correlating logs from **both Ubuntu and Windows endpoints**.

Step-by-Step:

1. **Open Splunk Web UI**
   URL: `http://192.168.56.10:8000` → Login as `splunkadmin`.

2. **Go to Search & Reporting**.

3. **Run a Combined Search Query:**
   This query shows logs from all indexes and both hosts for the last 24 hours (adjust time as needed):

Correct Host Names in Query

Try this query:

```
index=* (host="splunk-lab-01" OR host="WIN11-LAB-01")
| table _time, host, index, sourcetype, EventCode, message
| sort -_time
```

✅ Explanation:

- We use the **actual hostnames**.
- `index=*` includes all indexes (os_logs, security_logs, windows_logs).
- `table _time, host, index, sourcetype, EventCode, message` → keeps the table clean.

- `sort -_time` → newest events on top.

Optional: Include All Linux & Windows Logs Without Exact Host Match

If you want this dashboard to work even if the hostnames change in the future:

```
index=* (index=os_logs OR index=security_logs OR index=windows_logs)
| table _time, host, index, sourcetype, EventCode, message
| sort -_time
```

✅ This works because it filters by **index** instead of relying on host pattern matching.

Detection Logic Table

Detection	Source	Event Type / Code	How Tested	Screenshot
Failed Windows logins	Windows Security	EventCode 4625	Enter wrong password 3x	Windows-Failed-Logons.png
New Windows Service	Windows System	EventCode 7045	`sc create SOC_Test_Service`	Windows-New-Service-Creation.png
Failed SSH Ubuntu	Linux auth	ssh failed login	`ssh wronguser@localhost`	Ubuntu-Failed-SSH.png
sudo Usage	Linux auth	sudo command	`sudo ls /root`	Ubuntu-Sudo-Usage.png