SISEN Labs Student Guide

1 Purpose

This document provides the shared setup, prerequisites, and common working practices for the SISEN lab scenarios.

The SISEN labs examine how security failures in wireless, IoT, simulated BLE, MQTT, and low-power networked systems can become **safety-relevant system behaviours**. The focus is not only on observing an attack, but on understanding how a technical security event can affect system behaviour, operator interpretation, and safety outcomes.

Each scenario follows the same general pattern: students start the scenario, observe normal behaviour, capture traffic, run a controlled attack or failure demonstration, observe the system effect, and then map the observed behaviour to a possible hazard and mitigation strategy.

2 How to Use This Lab Set

- Start with this document to prepare the SISEN environment.
- Use the relevant scenario guide to start, observe, and stop the Medical IoT, Smart Building, or 6LoWPAN scenario.
- Use the attack, observation, and disruption guide for the controlled activities and evidence-capture commands.
- Use the hazard analysis worksheet to connect the observed security event to a possible safety consequence.
- You may also use the `README` files provided in the GitHub repository for additional setup notes and command examples.

3 Repository Access

- GitHub repository: `https://github.com/azelhajjar/SISEN.git`
- Recommended local path: `$HOME/Desktop/SISEN`

Install Git and clone the repository

```
sudo apt update
sudo apt install -y git
cd $HOME/Desktop
git clone https://github.com/azelhajjar/SISEN.git
cd SISEN
```

Repository Notes

The repository includes a main `README` file and scenario-specific documentation under `docs/`. Use these files alongside the lab handouts when you need current command examples, capture guidance, attack references, or troubleshooting notes.

The main manual-attack guidance is stored in `docs/manual-attacks.md`. Separate hardware access point activities are documented under `ap/hw-ap/docs/`.

All commands in the SISEN documentation assume that you are working from the root folder of the SISEN repository unless stated otherwise.

4 Lab Environment

SISEN should be run on a Debian-based Linux system. The system can be installed directly on a physical machine or run inside a virtual machine. The important requirement is the Linux environment, not whether the machine is native or virtualised.

SISEN was developed and validated on Ubuntu Linux running inside a virtual machine. Ubuntu 24.04 or later is recommended.

- Operating system: Debian-based Linux, such as Debian or Ubuntu.
- Installation type: native installation or virtual machine.
- Required privileges: some networking, namespace, AP, and capture commands require `sudo`.
- Network mode: NAT networking is sufficient when SISEN is run in a virtual environment.

4.1 Core Dependencies

SISEN uses a Debian-based Linux environment with Python, MQTT tooling, AP infrastructure tools, attack-support tools, and packet capture tools. The dependencies are grouped below so that each set has a clear purpose.

Install SISEN system dependencies

```
cd $HOME/Desktop/SISEN
sudo ./setup/setup-dependencies.sh
```

This script installs the operating-system, networking, wireless, MQTT, packet-capture, analysis, Python, and virtual-environment dependencies required by SISEN. Wireshark is installed as part of this process.

4.2 Wi-Fi Infrastructure Attack Tools

Some SISEN activities use the separate HW-AP materials and require a physical USB wireless adaptor that supports monitor mode and, where required, packet injection. The system dependency script installs the relevant wireless analysis and attack-support tools used by these activities.

Use these tools only against the controlled lab access point and only when instructed by the relevant activity document under `ap/hw-ap/docs/`.

5 Initial Setup

After installing the system dependencies, run the project setup script as the normal user. The script creates the local `.env` file when required, creates the project-local Python virtual environment under `.venv`, installs the Python packages listed in `requirements.txt`, and verifies the required modules.

Run the SISEN project setup

```
cd $HOME/Desktop/SISEN
./setup.sh
```

Students do not need to activate `.venv` manually. SISEN uses the project-local Python interpreter directly.

6 General Working Practice

- Record the scenario, AP mode, node or patient count, and lab activity being used.
- Observe the read-only dashboard, terminal output, MQTT messages, logs, and captures before and after the security event.
- Keep the lab running long enough to establish a normal baseline before triggering the attack or failure demonstration.
- Continue observing the system after the attack to determine whether the effect is temporary, persistent, repeated, or recoverable.
- Record the exact command, target, capture filename, interface, topic, packet number or time range, and recovery evidence where relevant.
- Separate direct observation from inference and from assumptions used in the safety analysis.
- Use your notes and evidence to complete the hazard analysis worksheet.

Common Lab Workflow

Each SISEN scenario follows the same workflow:

Stage	Student activity
Start scenario	Launch the assigned scenario and confirm that the required services are running.
Observe normal behaviour	Watch the dashboard, telemetry stream, logs, and network traffic before any security event is triggered.
Capture traffic	Start packet capture on the relevant interface or subscribe to the relevant MQTT topics.
Run controlled activity	Trigger the bounded observation, attack, or disruption activity described in the activity guide.
Observe system effect	Compare dashboard state, telemetry, logs, and traffic before and after the activity.
Map to hazard	Identify the safety-relevant consequence of the observed system behaviour.
Discuss mitigation	Propose controls, resilience mechanisms, and residual risk.

6.1 Capture-Friendly Execution

The labs are designed to be capture-friendly. Launcher-managed scenarios remain running until stopped by the user. Helper-based telemetry attacks are bounded demonstrations: where an injected value must remain visible, the helper refreshes it for approximately ten seconds and then exits while normal scenario telemetry continues.

Capture practice

Do not start and stop scenarios too quickly. You need time to observe normal behaviour, start packet capture, trigger the controlled activity, and compare the resulting evidence.

Typical evidence sources include:

- packet captures from wireless, namespace, low-power, or MQTT-facing interfaces;
- MQTT topic output from `mosquitto_sub`;

- dashboard state before, during, and after the security event;
- terminal logs from scenario runners, gateways, controllers, and attack helpers;
- recovery evidence showing whether normal operation resumed automatically or required cleanup;
- hazard analysis notes completed during the scenario.

7 Security-Informed Safety Reasoning

The labs do not stop at identifying a security issue. You should reason from the security event to the system effect and then to the possible safety consequence.

Reasoning step	Example question
Security event	What attack, failure, spoofing, replay, suppression, or disruption occurred?
Affected component	Which sensor, gateway, broker, dashboard, AP, namespace, or trust boundary was affected?
Observed system effect	What changed in the dashboard, telemetry, logs, captured traffic, or service state?
Unsafe state and consequence	What unsafe or misleading system behaviour could follow, and what harm may result?
Assumptions	Which additional conditions are required for the observed cyber event to become a real safety issue?
Mitigation and assurance	What cyber and safety controls are proposed, and what evidence would show that they work?
Residual risk	What risk remains even after the proposed mitigation?

8 Hazard Analysis Notes

You should complete a hazard analysis worksheet for each scenario. The worksheet should be completed during the lab, not only after the lab has finished. The hazard analysis worksheet is available **online** and in the SISEN repository under `teaching-materials/labs/`. The hazard analysis should capture:

- the scenario or lab name and normal baseline;
- the evidence source and precise evidence reference;
- the security event and weakened security property;
- the affected component, interface, topic, or trust boundary;
- the observed system effect and recovery evidence;
- the unsafe state, decision or automation impact, and possible consequence;
- whether each statement is a direct observation, inference, or assumption;
- cyber controls, safety-impact controls, detection, and assurance evidence;
- whether a control is proposed, implemented, tested, or partially effective;
- residual risk and, where used, the qualitative risk before and after mitigation.

Cleanup Stop launcher-managed scenarios using `python3 launch_sisen.py --stop`. Some manual activities, such as wireless monitor-mode capture, require additional cleanup. For example, remove a temporary monitor interface using `sudo iw dev mon-sisen del`. Follow the activity-specific stopping instructions where provided.