

Qwalora Ranger Manual v4.0

Cyber Operations and Offensive Security Academy

Candidate Execution Manual and Examination Standard, Version 4.0

Qwalora / Intelligence before infrastructure

26 weeks | 130 workdays | minimum 3 focused hours per workday | 26 weekly integrations | 6 independent monthly boards | Windows 11 + Ubuntu 24.04 (WSL2) + Docker + GitHub + Vercel

Qualify as a **Qwalora Ranger** by scoring **>= 95 percent** across all boards, passing every monthly board at **>= 95**, passing every mandatory safety gate, and completing the entire program **without AI assistance**.

1. Program contract: one lab, incrementally extended

This is not a list of independent learning activities. Every assignment occurs in the same private `qwalora-cyber-range` GitHub repository and builds directly on the prior passing submission. You must at all times distinguish six execution venues: the Windows host, the Linux WSL distro, a Docker container, an optional Windows test VM, hosted GitHub automation, and the public Vercel portfolio. An assignment is incomplete if it does not name its execution venue and provide executable proof that reproduces on a clean checkout.

Expected result: you independently demonstrate junior-to-mid-level SOC analysis, detection engineering, secure software and pipeline engineering, security architecture, and ethically scoped web and API assessment against owned local targets, in a repeatable environment, and you can defend every decision live and uncoached.

Honest limits: no six-month program guarantees employment, independent senior architecture authority, or any specific external vacancy. This program produces assessable, reproducible evidence of readiness and a Ranger who can be trusted on a live defensive team.

2. How to use this manual (you will not need a tutor)

1. Read this manual, then `docs/integrity-policy.md`, `docs/grading-rubric.md`, and `docs/submission-format.md`. Complete day-zero using `docs/SELF_SERVICE.md`.
2. Work strictly in order. Each day states its dependency gate, venue, required tools, what to perform, what to submit, the machine-verifiable PASS, the required negative control, and the standards mapping. That is a complete specification.
3. **Treat the machine-verifiable PASS as the authority.** If your output satisfies it on a clean checkout, with the required negative control behaving correctly, you are correct. You do not need confirmation from a person.
4. **Resolving ambiguity yourself:** when two readings are possible, choose the one that is safer (more isolated, loopback-only, least privilege), implement it, and record the decision and its justification in that day's evidence file. Primary documentation (Section 15) is the only tie-breaker. An AI is never a tie-breaker and is prohibited.
5. One pull request per day; one weekly integration PR per week; one board per month. Nothing merges to `main` without green CI and review.

3. Integrity: no AI, ever (summary)

You may not use any AI assistant, AI code completion, or AI-backed answer engine to produce, debug, or explain any part of any submission. You may use primary documentation, man pages, and tool source. Every evidence file and PR carries the attestation. Detection is by clean-room reproduction, blind scenarios, and live uncoached oral defense, not by a text classifier. Fabrication, scope violation, public exposure of a vulnerable service or secret, or a false attestation is automatic, non-recoverable failure. Full policy: `docs/integrity-policy.md`.

4. Canonical hardware, hypervisor, and tool choices

Layer	Exactly where it runs and why
Host (QW-HOST)	Candidate-owned Windows 11 laptop/desktop, virtualization enabled. 16 GB RAM minimum for the lightweight track, 32 GB recommended once the Wazuh SIEM is running. 80 to 150 GB free disk.
Linux (QW-LNX)	Ubuntu 24.04 in WSL2. All command-line work, Linux users, scripting, packet analysis, and local git run here. <code>wsl --install -d Ubuntu-24.04</code> from an administrator PowerShell.
Containers	Docker Desktop with WSL integration; issue <code>docker compose</code> from the Ubuntu terminal. qw-api and qw-juice bind to 127.0.0.1 only. Wazuh later uses its official version-matched compose assets on a distinct local-only profile.

SSH client	Native Windows Terminal / OpenSSH, or Ubuntu <code>ssh</code> . MobaXterm is optional convenience, not a Linux install or hypervisor, and is never required.
VM (optional)	A Windows 11 evaluation VM is optional, for authentic Windows telemetry (W03, W12). It needs a licensed/evaluation image and compatible virtualization. Otherwise use the supplied sanitized EVTX fixtures. Do not nest a VM inside WSL.
Source of truth	One private GitHub repo <code>qwalora-cyber-range</code> with <code>.github/workflows/ci.yml</code> ; protected <code>main</code> , reviewed and version-gated PRs.
Public presentation	A dedicated Vercel project pointed at <code>portfolio/site</code> , static site only. It must NEVER host the vulnerable app, API, SIEM, credentials, or any attack interface.

5. Day-zero setup

Follow `docs/SELF_SERVICE.md` Part A (steps 0.1 to 0.8) exactly. It ends with `scripts/verify-ownership.sh` printing `RESULT: PASS` , `pytest green`, and a green `test-and-validate` CI run. Record the baseline in `evidence/W01/D001/` and `architecture/host-baseline.md` . If anything fails, Part B of that document is your troubleshooting runbook.

6. Named systems and test identities

Name	Location	Purpose / authority
QW-HOST	Windows 11 personal machine	Virtualization, local browser
QW-LNX	Ubuntu 24.04 WSL2	Linux users, scripts, curl, git, automation
qw-api	Docker service <code>api</code> , <code>http://127.0.0.1:8000</code>	FastAPI training app
qw-juice	Docker service <code>juice-shop</code> , <code>http://127.0.0.1:3000</code>	OWASP Juice Shop
qw-siem	Wazuh official single-node compose (from W05)	Local SIEM; access restricted to the owned host
qw-sensor	Suricata/Zeek container or lab VM (from W06)	Observe traffic on a configured test sensor
alice	Bearer <code>alice-demo-token</code>	Owns record id 1
bob	Bearer <code>bob-demo-token</code>	Owns record id 2

7. The exact owner vs non-owner API laboratory

`app/api/main.py` implements a deliberately vulnerable route `GET /lab/vulnerable/notes/{note_id}` and a correctly protected route `GET /api/v1/notes/{note_id}` . This is a toy in-memory training API with predictable identities; it never processes real accounts or real sensitive data and is bound to loopback only.

Run from QW-LNX at the repo root with the lab up:

```
# Owner positive -> HTTP 200, owner alice
curl -i -H "Authorization: Bearer alice-demo-token" http://127.0.0.1:8000/api/v1/notes/1
# Non-owner negative -> HTTP 403, no Bob record body
curl -i -H "Authorization: Bearer alice-demo-token" http://127.0.0.1:8000/api/v1/notes/2
# Missing auth -> HTTP 401
curl -i http://127.0.0.1:8000/api/v1/notes/1
# Intentional baseline vulnerability -> HTTP 200 (broken object-level authorization)
curl -i -H "Authorization: Bearer alice-demo-token" http://127.0.0.1:8000/lab/vulnerable/notes/2
```

`app/api/tests/test_authorization.py` encodes positive (200), negative (403), missing-auth (401), not-found (404), and the known vulnerable baseline (200). Run exactly as CI does: `PYTHONPATH=app/api pytest -q app/api/tests` . The Week 10 fix exercise changes the vulnerable handler so non-owners receive 403 and flips the baseline test from 200 to 403, in the same PR, so CI shows before and after. The examiner can revert that branch to reset the demonstration.

8. Mandatory daily evidence contract

For each of the 130 days create `evidence/Wxx/Dyyy/README.md` from `evidence/TEMPLATE.md` (scaffold with `bash scripts/new-evidence.sh xx yyy`), where `Dyyy` is the overall day number (so Week 10 Day 3 is `W10/D048`). Include prerequisites, execution venue, exact commands, expected vs actual, test data, links to runnable source, the CI run URL, the commit SHA, failure analysis, and follow-up. Open a PR referencing that evidence path. A screenshot never replaces executable verification. Full format: `docs/submission-format.md` .

9. Grading and the 95 percent bar (summary)

Scored on a 100-point rubric: Functional implementation 30, Security reasoning 25, Detection and verification 20, Documentation and evidence 15, Live defense 10. Qualification requires a weighted **>= 95 percent**, every monthly board **>= 95**, and every mandatory safety gate passed. Full rubric and automatic-failure conditions: `docs/grading-rubric.md`.

10. Standards and version lock

Map findings and controls to the following, and pin exact releases and image digests at cohort kickoff in `architecture/host-baseline.md`.

Standard / tool	Version this program targets (Oct 2026)
OWASP Top 10 (web)	OWASP Top 10:2025 (A01 Broken Access Control ... A03 Software Supply Chain Failures ... A09 Logging and Alerting Failures, A10 Mishandling of Exceptional Conditions)
OWASP API Security Top 10	2023 (API1:2023 BOLA ... API10:2023), separate from the web list
OWASP WSTG / ASVS	WSTG current; ASVS for verification requirements
MITRE ATT&CK (Enterprise)	v19.0 current; v18 introduced Detection Strategies and Analytics (used in W15 purple team)
NIST CSF	2.0 functions GOVERN, IDENTIFY, PROTECT, DETECT, RESPOND, RECOVER
NIST SP 800-115 / 800-61	Technical assessment; incident handling
SIEM	Wazuh 4.x official single-node <code>wazuh-docker</code> (pin the exact tag at kickoff; set <code>vm.max_map_count=262144</code>)
OS / runtime	Ubuntu 24.04 LTS (WSL2); Python 3.12
Targets	qw-api (owned), OWASP Juice Shop (pin the image digest at kickoff)

11. Versioned dependency gates and submission destinations

Weeks	Prerequisite before start	Persistent destination
1-4	Day-zero baseline; local API bootstraps; repo exists	<code>infra/</code> , <code>app/api/</code> , <code>scripts/</code> , <code>.github/</code> , <code>evidence/W###/</code>
5-8	Board 1 pass; reproducible local lab	<code>telemetry/</code> , <code>detections/</code> , <code>cases/</code> , <code>evidence/W###/</code>
9-12	Board 2 pass; signed RoE; two local targets	<code>offensive/</code> , <code>reports/</code> , <code>app/api/tests/</code> , <code>evidence/W###/</code>
13-16	Board 3 pass; verified ownership regression and CI	<code>policy/</code> , <code>app/</code> , <code>purple/</code> , <code>.github/workflows/</code> , <code>evidence/W###/</code>
17-20	Board 4 pass; stable telemetry and tests	<code>architecture/</code> , <code>infra/iac/</code> , <code>governance/</code> , <code>evidence/W###/</code>
21-26	Board 5 pass; all prior gates accepted	<code>capstone/</code> , <code>portfolio/site/</code> , <code>assessment/</code> , <code>evidence/W###/</code>

See `diagrams/convergence-map.svg` for how each month's artifacts stack into the final capstone.

12. Daily execution manual: 130 linked assignments

Each day card: **Gate** (what accepted state it builds on), **Venue**, **Perform**, **Submit** (implementation path plus the day's evidence folder), **PASS** (machine-verifiable), **Negative control**, **Map** (standards). Every day also submits `results.txt` (or sanitized JSON), test output, a reviewed PR URL, and the source commit SHA. Reproduce on a clean checkout.

Week 01 - Environment engineering and source control

- Dependency gate: day-zero completed (`verify-ownership.sh` PASS, CI green).
- Active environment: QW-LNX (Ubuntu WSL2) + GitHub. Canonical paths: `infra/`, `scripts/`, `docs/`, `evidence/`.
- Weekly outcome: rebuild a secure lab from versioned instructions alone.

D001 - Establish scope and baseline

- Gate: day-zero verified baseline. Venue: QW-LNX + GitHub.
- Perform: write the standing RoE (`scope/roe.md`), an asset inventory, local host resource and network threat assumptions. Bring up the existing range (`docker compose -f infra/compose.yaml up -d --build`) and bootstrap `architecture/host-baseline.md` with OS, image tags/digests, network bindings, and pinned versions.

- Submit: `scope/roe.md` , `architecture/host-baseline.md` ; `evidence/W01/D001/` .
- PASS: examiner reads in-scope targets and stop rules; baseline reproduces on a clean checkout; `curl -f http://127.0.0.1:8000/health` returns `200 status ok` .
- Negative control: `host-baseline.md` records that no service answers on a non-loopback address (`verify-ownership.sh` loopback check PASS).
- Map: NIST CSF 2.0 GOVERN/IDENTIFY; NIST SP 800-115 scoping.

D002 - Build the engineering repository

- Gate: accepted D001. Venue: QW-LNX + GitHub.
- Perform: initialize protected `main` , issue and PR templates, `CODEOWNERS` , commit conventions, branch protection requiring review plus `test-and-validate` .
- Submit: `.github/ config` + first reviewed PR; `evidence/W01/D002/` .
- PASS: a PR cannot merge without a passing check and a review; direct push to `main` is rejected.
- Negative control: attempt a direct push to `main` and show it is blocked.
- Map: OWASP Top 10:2025 A08 (integrity of the change process); NIST CSF 2.0 PROTECT.

D003 - Linux operations foundation

- Gate: accepted D002. Venue: QW-LNX (or disposable `ubuntu:24.04` container for privileged tests).
- Perform: script users, permissions, services, journal and process inspection using `scripts/host-audit.sh` ; inspect `id` , `/etc/passwd` , `stat` , `sudo/audit`. Never edit Windows host accounts.
- Submit: `scripts/host-audit.sh` + `host-report.json` ; `evidence/W01/D003/` .
- PASS: the script exits non-zero when an intentionally broken control (world-writable file or loose-permission script) is present, zero otherwise.
- Negative control: introduce one world-writable file, show exit code 1; remove it, show exit code 0.
- Map: CWE-732 (incorrect permission assignment); NIST CSF 2.0 PROTECT.

D004 - Isolate Docker networks

- Gate: accepted D003. Venue: QW-LNX + GitHub.
- Perform: build a segmented compose network with separate app and sensor segments; identify the real Linux namespace/interface before any capture using `ip route` , `ss` , `dig` , and a short timed `tcpdump` ; redact packets before committing.
- Submit: updated `infra/compose.yaml` + a network diagram (`architecture/network.svg` or Mermaid); `evidence/W01/D004/` .
- PASS: the vulnerable target is unreachable from any non-loopback interface; authorized intra-segment flow works.
- Negative control: a probe from outside the app segment to `qw-api` fails.
- Map: ATT&CK T1046 (network service discovery) observability; NIST CSF 2.0 PROTECT (PR.AA/PR.IR).

D005 - Integrate and rebuild

- Gate: accepted D004. Venue: QW-LNX + GitHub.
- Perform: delete and redeploy the environment from the documented bootstrap only (`scripts/reset.sh` , `docs/REBUILD.md`).
- Submit: `docs/REBUILD.md` + `demo`; `evidence/W01/D005/` .
- PASS: examiner rebuilds from a clean clone with zero manual edits; health and tests green.
- Negative control: a deliberately missing step in a scratch copy of REBUILD causes a reproducible failure, proving the documented path is necessary and sufficient.
- Map: NIST CSF 2.0 RECOVER; reproducibility.

Friday W01 integration: create PR `week-01-integration` ; check source, run the negative control, run CI, demonstrate lab continuity; submit `assessments/week01/review.md` with blockers and reviewer signoff.

Week 02 - Packets, services, and troubleshooting

- Gate: W01 Friday PR accepted, all blocking tests green. Venue: QW-LNX + local compose networks. Paths: `infra/` , `captures/` , `networking/` , `evidence/` .
- Weekly outcome: explain paths and failure domains at the packet level.
- Standing tools: `ip addr` , `ip route` , `dig` , `ss -lnt` , `docker network ls` , controlled `tcpdump` (record interface and filter; redact before commit).

D006 - TCP/IP tracing

- Perform: capture DNS, ARP, TCP handshake, and TLS negotiation inside the private lab.
- Submit: `captures/sanitized/trace.pcapng` + packet notes; `evidence/W02/D006/` .
- PASS: you identify the handshake and the DNS answer from your own capture.
- Negative control: a filter that should capture nothing produces an empty capture, proving the filter is correct.
- Map: fundamentals; ATT&CK data source: network traffic.

D007 - DNS investigation

- Perform: deploy a private DNS record and intentionally misconfigure resolution.
- Submit: `networking/dns/zone.conf` + fault transcript; `evidence/W02/D007/` .
- PASS: you correctly diagnose NXDOMAIN vs timeout from evidence.
- Negative control: the working record resolves; the broken record reproduces the exact failure class you claim.
- Map: fundamentals; detection of resolution failures.

D008 - Firewall segmentation

- Perform: apply an allow-list between application, proxy, and monitoring networks; pin image digests referenced by `host-baseline.md` .
- Submit: `infra/firewall.rules` + tests; `evidence/W02/D008/` .
- PASS: authorized flows pass and forbidden flows fail, each shown by command.
- Negative control: a forbidden flow is denied and logged.
- Map: NIST CSF 2.0 PROTECT (PR.IR); CWE-284.

D009 - TLS fundamentals

- Perform: issue a local CA certificate and compare valid vs invalid TLS behavior.
- Submit: `networking/certs/README.md` + handshake log; `evidence/W02/D009/` .
- PASS: you explain the chain-of-trust failure without bypassing validation.
- Negative control: an untrusted cert is rejected; you never disable verification to "pass".
- Map: OWASP Top 10:2025 A04 (cryptographic failures); CWE-295.

D010 - Network fault challenge

- Perform: repair a randomized route/port/DNS failure introduced into the lab.
- Submit: `cases/week02.md` + short recording; `evidence/W02/D010/` .
- PASS: service recovers and you identify root cause with evidence.
- Negative control: pre-fix probe fails, post-fix probe succeeds, both captured.
- Map: NIST CSF 2.0 RESPOND/RECOVER.

Friday W02 integration: PR `week-02-integration` ; `assessments/week02/review.md` .

Week 03 - Endpoints, identity, and hardening

- Gate: W02 accepted. Venue: QW-LNX + disposable local Linux container. Paths: `scripts/` , `endpoints/` , `telemetry/` , `evidence/` .
- Weekly outcome: demonstrate system trust and audit visibility.
- Standing tool: `docker run --rm -it --name qw-linux-admin ubuntu:24.04 bash ; inside, apt update && apt install -y sudo procs ; create test users with useradd -m trainee ; inspect id trainee` .

D011 - Linux auth logs

- Perform: configure audit logging for sudo, SSH, and auth events; emit to `telemetry/linux-events.jsonl` with a stable schema.
- Submit: `telemetry/linux-events.jsonl` + generator; `evidence/W03/D011/` .
- PASS: you generate and locate three known auth events by field and time.
- Negative control: a benign action does not produce the auth event you are keying on.
- Map: ATT&CK T1078 (valid accounts), auth logging; NIST CSF 2.0 DETECT (DE.CM).

D012 - Windows telemetry

- Perform: stand up a Windows evaluation VM where licensing allows, or ingest the supplied curated EVTX fixtures (`telemetry/windows/fixtures/`).
- Submit: `telemetry/windows/` + source manifest; `evidence/W03/D012/` .
- PASS: you explain three Windows Event IDs and their provenance (for example 4624, 4625, 4688).
- Negative control: a fabricated-looking event is rejected by your manifest check.
- Map: ATT&CK data source: Windows event logs; NIST CSF 2.0 DETECT.

D013 - Least privilege

- Perform: apply user/group permissions, service accounts, and file ACLs.
- Submit: `endpoints/access-matrix.md` + enforcing script; `evidence/W03/D013/` .
- PASS: an unauthorized account cannot access a protected resource (shown).
- Negative control: the authorized account can; the unauthorized account is denied.
- Map: CWE-269/CWE-732; NIST CSF 2.0 PROTECT (PR.AA).

D014 - Vulnerability hygiene

- Perform: record baseline packages, patch versions, and misconfiguration checks.

- Submit: `scans/host-baseline.json` + risk list; `evidence/W03/D014/`.
- PASS: you show one real finding and a validated fix.
- Negative control: post-fix re-scan no longer reports the finding.
- Map: OWASP Top 10:2025 A02 (security misconfiguration); NIST CSF 2.0 IDENTIFY (ID.RA).

D015 - Hardening review

- Perform: rebuild a hardened host and present evidence.
- Submit: `endpoints/hardening-week03.md` + reproducible test; `evidence/W03/D015/`.
- PASS: the examiner introduces one misconfiguration and your check detects it.
- Negative control: the clean host passes the same check.
- Map: NIST CSF 2.0 PROTECT; CIS-style hardening.

Friday W03 integration: PR `week-03-integration` ; `assessments/week03/review.md`.

Week 04 - Automation and the first infrastructure board

- Gate: W03 accepted. Venue: GitHub Actions runner + QW-LNX. Paths: `.github/workflows/`, `infra/`, `assessments/`, `evidence/`.
- Weekly outcome: ship reproducible infrastructure under CI.
- Standing tool: push a PR, inspect the `test-and-validate` workflow, verify required pytest and compose-config checks; report the full run URL and SHA.

D016 - GitHub Actions build

- Perform: run lint, unit tests, YAML validation, and artifact upload in `.github/workflows/ci.yml`.
- Submit: `.github/workflows/ci.yml` + badge evidence; `evidence/W04/D016/`.
- PASS: a deliberately bad configuration fails CI.
- Negative control: restore the config, show the green run.
- Map: OWASP Top 10:2025 A08; NIST CSF 2.0 PROTECT.

D017 - Secrets hygiene

- Perform: set encrypted CI secrets, minimal token permissions, and a local `.env.template` ; add a secret scanner (gitleaks) step.
- Submit: `SECURITY.md` + secret-scan results; `evidence/W04/D017/`.
- PASS: an injected fake secret causes a failing check.
- Negative control: a clean branch passes the scan.
- Map: OWASP Top 10:2025 A02/A08; CWE-798 (hard-coded credentials).

D018 - Provisioning scripts

- Perform: create idempotent `scripts/up.sh`, `scripts/reset.sh`, `scripts/down.sh`.
- Submit: the three scripts; `evidence/W04/D018/`.
- PASS: running `up` twice makes no unintended changes (idempotent).
- Negative control: `down -v` followed by `up` reaches the same healthy state.
- Map: reproducibility; NIST CSF 2.0 RECOVER.

D019 - Recovery drill

- Perform: restore a corrupted configuration from git and evidence backup; measure RTO.
- Submit: `recovery/drill01.md` + measured RTO; `evidence/W04/D019/`.
- PASS: restored services pass health probes within the stated objective.
- Negative control: the corrupted state fails the probe before restore.
- Map: NIST CSF 2.0 RECOVER (RC.RP).

D020 - BOARD 1

- Perform: defend the infrastructure after the examiner changes one unknown dependency; restore from git.
- Submit: `assessment/month01/` + tag `board1` (release v1.0); `evidence/W04/D020/`.
- PASS: all essential services restored, no public vulnerable port, clean rebuild.
- Negative control: the injected break is shown failing, then fixed and passing.
- Map: see Board 1 in Section 13.

Friday W04 integration: PR `week-04-integration` ; `assessments/week04/review.md`. **EXAMINATION GATE: Board 1 must pass (>= 95) before Week 05 opens.**

Week 05 - SIEM design and telemetry ingestion

- Gate: W04 accepted and Board 1 pass. Venue: QW-LNX + Wazuh official single-node. Paths: `telemetry/wazuh/`, `docs/`,

- `evidence/`.
- Weekly outcome: build measurable log ingestion and monitoring coverage.
- Standing tool: clone official `wazuh-docker` at a pinned supported release; single-node deployment; local-only port binding; set `vm.max_map_count=262144`. For SIEM days show actual ingested event metadata and a rule id; if resource-constrained, replay documented fixtures and label it a fallback, not live ingestion.

D021 - Wazuh architecture

- Perform: draw the manager/indexer/dashboard trust and data flows.
- Submit: `telemetry/siem-design.md` + diagram; `evidence/W05/D021/`.
- PASS: you explain each component's failure effects.
- Negative control: you identify what breaks if the indexer is down.
- Map: NIST CSF 2.0 DETECT (DE.CM); architecture.

D022 - Deploy SIEM

- Perform: deploy the supported Wazuh stack with correct host sizing; reuse the existing `infra/compose.yaml` app; record `docker compose -f infra/compose.yaml ps` before and after.
- Submit: `telemetry/wazuh/` + health probes; `evidence/W05/D022/`.
- PASS: all required components healthy after a reboot.
- Negative control: a component health probe fails when you stop it, recovers on restart.
- Map: NIST CSF 2.0 DETECT.

D023 - Ingest Linux data

- Perform: install the agent or forward `journald/syslog` to the manager.
- Submit: `telemetry/source-linux.md` + event ids; `evidence/W05/D023/`.
- PASS: an expected event appears with host and time.
- Negative control: when the source is stopped, the event stops arriving.
- Map: ATT&CK data source: process/auth logs.

D024 - Ingest application events

- Perform: emit `qw-api` JSON auth/access logs (the structured schema already in `app/api/logging_config.py`) with a stable schema and ingest them.
- Submit: `app/logging/` config + sample dataset; `evidence/W05/D024/`.
- PASS: you can search by event type and correlation id (`request_id`).
- Negative control: a malformed line is handled without breaking the pipeline.
- Map: OWASP Top 10:2025 A09 (logging and alerting failures).

D025 - Source coverage review

- Perform: add uptime and log-freshness checks and a pipeline map.
- Submit: `telemetry/coverage.csv` + alert proof; `evidence/W05/D025/`.
- PASS: simulate an offline source and the freshness alarm fires.
- Negative control: a healthy source does not raise the freshness alarm.
- Map: NIST CSF 2.0 DETECT (DE.CM-3).

Friday W05 integration: PR `week-05-integration` ; `assessments/week05/review.md` .

Week 06 - Network telemetry and detection

- Gate: W05 accepted. Venue: QW-LNX + local sensor/container. Paths: `telemetry/suricata/` , `telemetry/zeek/` , `detections/` , `evidence/` .
- Weekly outcome: detect and explain controlled network activity.
- Standing tool: generate test traffic ONLY inside owned compose services; confirm the capture interface and parsed `eve.json` before claiming a detection.

D026 - Suricata sensor

- Perform: deploy Suricata in the private capture path.
- Submit: `telemetry/suricata/` + `eve.json` ; `evidence/W06/D026/` .
- PASS: you observe benign and test-signature events.
- Negative control: benign traffic does not trigger the test signature.
- Map: ATT&CK detection; NIST CSF 2.0 DETECT.

D027 - Network metadata

- Perform: add Zeek or a documented lightweight flow parser.
- Submit: `telemetry/network-flows.jsonl` ; `evidence/W06/D027/` .

- PASS: you reconstruct a connection tuple and duration.
- Negative control: a flow you did not generate does not appear.
- Map: ATT&CK data source: network connection.

D028 - Normalization

- Perform: normalize timestamp, source IP, destination, outcome, and actor.
- Submit: `telemetry/schema.md` + parser tests; `evidence/W06/D028/`.
- PASS: malformed events are rejected gracefully.
- Negative control: a malformed fixture is dropped and counted, not silently accepted.
- Map: data quality; A09.

D029 - First detection

- Perform: implement a safe synthetic brute-force-like auth threshold rule.
- Submit: `detections/auth-burst.yaml` + fixtures; `evidence/W06/D029/`.
- PASS: the positive fixture alerts; the negative fixture does not.
- Negative control: the benign fixture produces no alert.
- Map: ATT&CK T1110 (brute force); detection engineering.

D030 - Noise tuning

- Perform: quantify false alarms and apply suppression with documented reasoning.
- Submit: `detections/tuning01.md` + replay logs; `evidence/W06/D030/`.
- PASS: false positives drop without losing the known positive.
- Negative control: the known-positive fixture still alerts after tuning.
- Map: precision/recall; A09.

Friday W06 integration: `PR week-06-integration` ; `assessments/week06/review.md` .

Week 07 - SOC cases and response

- Gate: W06 accepted. Venue: QW-LNX + Wazuh UI (local). Paths: `soc/` , `cases/` , `detections/` , `evidence/` .
- Weekly outcome: investigate hypotheses and preserve evidence.
- Standing tool: generate controlled authz events via `curl` , locate correlated logs by UTC timestamp and fields, export sanitized JSON evidence. Produce a timestamped case (source event, hypothesis, scope, decision, containment test, rollback).
Receiving an alert is not a completed investigation.

D031 - Alert triage

- Perform: create a severity/confidence/triage decision tree.
- Submit: `soc/triage-sop.md` ; `evidence/W07/D031/` .
- PASS: you correctly classify supplied sample alerts.
- Negative control: a benign sample is classified as benign, not escalated.
- Map: NIST SP 800-61 triage; A09.

D032 - Correlate events

- Perform: link login, web, and firewall events via timestamps and entities.
- Submit: `cases/case01/timeline.csv` ; `evidence/W07/D032/` .
- PASS: the chronology matches the underlying evidence.
- Negative control: an unrelated event is correctly excluded from the timeline.
- Map: ATT&CK correlation; DETECT.

D033 - Evidence preservation

- Perform: compute SHA-256 over the exported event package and document handling.
- Submit: `cases/case01/manifest.sha256` ; `evidence/W07/D033/` .
- PASS: hashes verify after transfer.
- Negative control: a tampered copy fails hash verification.
- Map: chain of custody; integrity.

D034 - Containment simulation

- Perform: disable a compromised demo identity or isolate a test container.
- Submit: `response/containment.sh` + rollback; `evidence/W07/D034/` .
- PASS: containment works and recovery reverses it.
- Negative control: post-containment access by the identity is denied; post-rollback it is restored.
- Map: NIST CSF 2.0 RESPOND (RS.MI); ATT&CK mitigation.

D035 - Incident briefing

- Perform: present a five-minute executive and a ten-minute analyst version.
- Submit: `cases/case01/report.md` + recording; `evidence/W07/D035/`.
- PASS: no unsupported attribution; you explain uncertainty.
- Negative control: you explicitly state what the evidence does NOT support.
- Map: NIST SP 800-61 reporting.

Friday W07 integration: PR `week-07-integration`; `assessments/week07/review.md`.

Week 08 - SOC milestone board

- Gate: W07 accepted. Venue: QW-LNX + GitHub Actions + Wazuh. Paths: `tests/detections/`, `assessments/board02/`, `evidence/`.
- Weekly outcome: defend an end-to-end incident investigation.
- Standing tool: replay saved benign/malicious fixtures through a documented decoder/test harness; fail the regression when the expected rule id does not appear.

D036 - Detection QA

- Perform: create automated regression tests for at least three rules; run locally then via `.github/workflows/ci.yml`.
- Submit: `tests/detections/` + report; `evidence/W08/D036/`.
- PASS: three positive and three benign scenarios are distinguished.
- Negative control: each benign scenario yields no alert.
- Map: detection CI; A09.

D037 - Triage speed

- Perform: measure detection and triage latency for generated events.
- Submit: `metrics/latency.csv`; `evidence/W08/D037/`.
- PASS: you report clock assumptions and the calculation.
- Negative control: you disclose the sample size and its limits.
- Map: MTTD definition.

D038 - Simulated incident

- Perform: run a scoped, preapproved identity-misuse scenario.
- Submit: `exercises/incident02.md` + logs; `evidence/W08/D038/`.
- PASS: you correlate sources and isolate the affected account.
- Negative control: an unaffected account is shown not isolated.
- Map: ATT&CK T1078; RESPOND.

D039 - Blind case prep

- Perform: rebuild the monitoring pipeline without manual dashboard click-paths.
- Submit: `scripts/rebuild-monitoring.sh`; `evidence/W08/D039/`.
- PASS: a fresh test event is visible within the documented SLA.
- Negative control: before rebuild, the event is absent; after, present.
- Map: reproducibility; DETECT.

D040 - BOARD 2

- Perform: investigate an unknown injected event set live.
- Submit: `assessment/month02/` + tag `board2` (release v2.0); `evidence/W08/D040/`.
- PASS: detect, reason, contain, preserve, brief.
- Negative control: you separate evidence from hypothesis explicitly.
- Map: see Board 2 in Section 13.

Friday W08 integration: PR `week-08-integration`; `assessments/week08/review.md`. **EXAMINATION GATE: Board 2 must pass (>= 95) before Week 09 opens.**

Week 09 - Ethical hacking process and reconnaissance

- Gate: W08 accepted and Board 2 pass. Venue: QW-LNX + private Juice Shop/API. Paths: `offensive/`, `scope/`, `evidence/`.
- Weekly outcome: plan and execute a bounded, authorized assessment.
- Standing tool: verify `curl http://127.0.0.1:8000/health` and a signed `scope/ROE.md` before any local enumeration.

D041 - Assessment RoE

- Perform: write purpose, scope, exclusions, contacts, test windows, and stop rules into `offensive/roe-assessment01.md`

(from `scope/RoE.template.md`).

- Submit: signed RoE; `evidence/W09/D041/`.
- PASS: RoE is complete and signed before any test; only loopback targets are in scope.
- Negative control: the RoE explicitly lists out-of-scope targets you will not touch.
- Map: NIST SP 800-115; PTES pre-engagement.

D042 - Asset discovery

- Perform: inventory only the local lab services; compare known vs discovered.
- Submit: `offensive/assets.csv` + provenance; `evidence/W09/D042/`.
- PASS: no out-of-scope address is scanned.
- Negative control: a scan attempt outside loopback is refused by your own guardrail.
- Map: ATT&CK TA0043 (reconnaissance), scoped; API inventory (OWASP API9:2023 improper inventory).

D043 - Threat modeling

- Perform: construct a dataflow and STRIDE threats for the demo application.
- Submit: `offensive/threat-model.md` + diagram; `evidence/W09/D043/`.
- PASS: each critical trust boundary has a mitigation.
- Negative control: you identify at least one accepted risk and justify it.
- Map: OWASP Top 10:2025 A06 (insecure design); STRIDE.

D044 - Safe web mapping

- Perform: enumerate owned app routes, auth states, and exposed headers.
- Submit: `offensive/web-map.md`; `evidence/W09/D044/`.
- PASS: the inventory matches the actual application.
- Negative control: a route that does not exist is not listed as present.
- Map: OWASP WSTG information gathering.

D045 - Recon briefing

- Perform: explain the highest-risk attack surfaces and a prioritized test plan.
- Submit: `offensive/test-plan.md` + recording; `evidence/W09/D045/`.
- PASS: each test ties to a scoped asset and expected evidence.
- Negative control: a test with no scoped asset is removed from the plan.
- Map: PTES; prioritization.

Friday W09 integration: PR `week-09-integration`; `assessments/week09/review.md`.

Week 10 - OWASP authentication and access control

- Gate: W09 accepted. Venue: QW-LNX + qw-api + browser. Paths: `app/api/`, `app/api/tests/`, `offensive/`, `evidence/`.
- Weekly outcome: demonstrate concrete flaws and fixes.
- Standing tool: exact alice/bob `GET /api/v1/notes/1`, `/2`, and `/lab/vulnerable/notes/2`; compare expected 200/403/200; include a non-owner 403 negative control.

D046 - Application setup

- Perform: confirm the owned targets (qw-api, Juice Shop) run in the disconnected local environment; `docker compose -f infra/compose.yaml ps` before and after.
- Submit: `offensive/targets/` notes + teardown; `evidence/W10/D046/`.
- PASS: targets are not reachable from the public internet.
- Negative control: `verify-ownership.sh` loopback check PASS.
- Map: safety invariant; A01.

D047 - Broken access control (the core exercise)

- Perform: reproduce unauthorized cross-user object access via `/lab/vulnerable/notes/2` with alice's token; capture the 200; then on a `fix/bola-403` branch add the owner check to `get_note_vulnerable()` in `main.py` and flip `test_vulnerable_baseline` from 200 to 403 in the same PR.
- Submit: `offensive/findings/BAC-01.md` + safe proof + the fix PR; `evidence/W10/D047/`.
- PASS: reproduce 200 before, deny with 403 after the fix; CI shows the before/after; the regression test passes post-fix.
- Negative control: `test_non_owner_is_forbidden` already enforces 403 on the protected route and must stay green throughout.
- Map: OWASP API Security Top 10 2023 API1:2023 (BOLA); OWASP Top 10:2025 A01; CWE-639, CWE-284.

D048 - Authentication control

- Perform: analyze local weak session handling and rotate demo tokens.

- Submit: `offensive/findings/AUTH-01.md` ; `evidence/W10/D048/` .
- PASS: a test demonstrates session invalidation.
- Negative control: an invalidated token is rejected with 401.
- Map: OWASP Top 10:2025 A07 (authentication failures); API2:2023.

D049 - Input handling

- Perform: exercise parameterized queries vs an insecure demo query locally.
- Submit: `offensive/findings/INJ-01.md` + tests; `evidence/W10/D049/` .
- PASS: an injection regression fixture fails the insecure version and passes the fixed one.
- Negative control: a benign input is handled identically by both.
- Map: OWASP Top 10:2025 A05 (injection); CWE-89.

D050 - OWASP review

- Perform: map observations to OWASP Top 10:2025 and CWE; use signed local RoE against 127.0.0.1 only.
- Submit: `offensive/owasp-matrix.csv` ; `evidence/W10/D050/` .
- PASS: each finding has evidence and remediation.
- Negative control: a non-finding is not claimed as a finding.
- Map: OWASP Top 10:2025; CWE.

Friday W10 integration: `PR week-10-integration` ; `assessments/week10/review.md` .

Week 11 - Web testing and reporting

- Gate: W10 accepted. Venue: QW-LNX + local OWASP ZAP (Docker/Desktop). Paths: `offensive/zap/` , `reports/` , `evidence/` .
- Weekly outcome: deliver an auditable penetration-testing report.
- Standing tool: restrict ZAP context to `localhost:3000` or `:8000` ; run the passive baseline first; save scope + HTML/JSON report; manually review findings.

D051 - ZAP configuration

- Perform: run a ZAP baseline against the owned demo target with exclusions.
- Submit: `offensive/zap/zap-config.yaml` + report; `evidence/W11/D051/` .
- PASS: scope excludes all non-target services.
- Negative control: a request to an out-of-scope host is not issued by ZAP.
- Map: OWASP WSTG; safe tooling.

D052 - Manual verification

- Perform: manually validate a scanner finding and mark false positives.
- Submit: `offensive/validation.md` ; `evidence/W11/D052/` .
- PASS: independent reproduction matches your disposition.
- Negative control: a false positive is shown not reproducible.
- Map: triage of scanner output.

D053 - API authorization

- Perform: test owner vs non-owner access against qw-api with alice/bob tokens.
- Submit: `offensive/findings/API-01.md` ; `evidence/W11/D053/` .
- PASS: the negative authorization test passes post-fix.
- Negative control: non-owner access is denied (403).
- Map: API1:2023 (BOLA); A01.

D054 - Fix validation

- Perform: apply and retest two vulnerabilities with CI regression.
- Submit: `tests/security/` + patch PRs; `evidence/W11/D054/` .
- PASS: a finding is closed only with passing proof.
- Negative control: the pre-fix test fails; the post-fix test passes.
- Map: remediation verification.

D055 - Pentest report

- Perform: write an executive summary, technical steps, risk, and retest results; put automated tests in `app/api/tests/` and run via CI.
- Submit: `reports/report01.md` ; `evidence/W11/D055/` .
- PASS: the examiner reproduces each finding safely.
- Negative control: every claim maps to a reproducible step.

- Map: NIST SP 800-115 reporting; PTES.

Friday W11 integration: PR `week-11-integration` ; `assessments/week11/review.md` .

Week 12 - Penetration testing board

- Gate: W11 accepted. Venue: QW-LNX + local apps + GitHub PR. Paths: `offensive/` , `reports/` , `assessments/board03/` , `evidence/` .
- Weekly outcome: explain a black-box and gray-box assessment.
- Standing tool: re-run named positive and negative tests, link a fix PR, capture before/after HTTP response codes.

D056 - Black-box phase

- Perform: evaluate the lab-hosted app with only the published interface and RoE.
- Submit: `offensive/blackbox-notes.md` ; `evidence/W12/D056/` .
- PASS: findings are justified without source assumptions.
- Negative control: a guess unsupported by observed behavior is excluded.
- Map: PTES black-box.

D057 - Gray-box phase

- Perform: review source, the identity model, and test-account entitlements.
- Submit: `offensive/graybox-notes.md` ; `evidence/W12/D057/` .
- PASS: you show how partial context improves coverage.
- Negative control: a finding only visible with source is labeled as such.
- Map: PTES gray-box.

D058 - Defender comparison

- Perform: verify offensive event observability in the SOC.
- Submit: `purple/visibility-matrix.csv` ; `evidence/W12/D058/` .
- PASS: correlate at least three test events to detections.
- Negative control: an action with no detection is recorded as a visibility gap.
- Map: purple-team visibility; ATT&CK mapping.

D059 - Retest

- Perform: demonstrate fixes, reopen unresolved findings, assess residual risk; put automated tests in `app/api/tests/` and trigger via CI.
- Submit: `offensive/retest.md` ; `evidence/W12/D059/` .
- PASS: no unsupported claims of remediation.
- Negative control: an unfixed finding stays open with proof.
- Map: remediation verification.

D060 - BOARD 3

- Perform: present a professional pen test and live patch validation.
- Submit: `assessment/month03/` + tag `board3` (release v3.0); `evidence/W12/D060/` .
- PASS: evidence, RoE, risk, and regression all pass.
- Negative control: pre-fix reproduces, post-fix denies.
- Map: see Board 3 in Section 13.

Friday W12 integration: PR `week-12-integration` ; `assessments/week12/review.md` . **EXAMINATION GATE: Board 3 must pass (>= 95) before Week 13 opens.**

Week 13 - Secure software development

- Gate: W12 accepted and Board 3 pass. Venue: QW-LNX + FastAPI code and Python venv. Paths: `app/api/` , `app/api/tests/` , `evidence/` .
- Weekly outcome: build secure services with verifiable controls.
- Standing tool: `PYTHONPATH=app/api pytest -q app/api/tests` ; capture test names, status, and a failing injection scenario.

D061 - Service design

- Perform: create a minimal security-events API with auth and structured logging.
- Submit: `app/api/` + OpenAPI spec; `evidence/W13/D061/` .
- PASS: tests cover accepted and rejected requests.
- Negative control: an unauthenticated request is rejected (401).
- Map: OWASP ASVS; A07.

D062 - Unit/integration tests

- Perform: add positive, negative, and adversarial-input tests; run via CI.
- Submit: `app/api/tests/` + coverage report; `evidence/W13/D062/`.
- PASS: CI blocks failing tests.
- Negative control: a deliberately failing test blocks merge.
- Map: test rigor.

D063 - Dependency audit

- Perform: generate an SBOM, vulnerability inventory, and exception rationale.
- Submit: `security/sbom.json` + `dependencies.md`; `evidence/W13/D063/`.
- PASS: critical issues are fixed or formally risk-accepted.
- Negative control: an injected vulnerable dependency is flagged.
- Map: OWASP Top 10:2025 A03 (software supply chain failures); SBOM.

D064 - SAST and secrets

- Perform: set lint, SAST, and secret-scanning gates.
- Submit: `.github/workflows/security.yml`; `evidence/W13/D064/`.
- PASS: synthetic bad code triggers a failure.
- Negative control: clean code passes all gates.
- Map: A08; CWE-798.

D065 - Peer code review

- Perform: submit a security-review PR with a threat-oriented checklist.
- Submit: `reviews/secure-pr.md`; `evidence/W13/D065/`.
- PASS: no direct protected-branch bypass.
- Negative control: an attempted bypass is blocked by branch protection.
- Map: secure SDLC.

Friday W13 integration: PR `week-13-integration`; `assessments/week13/review.md`.

Week 14 - CI/CD, supply chain, and identity

- Gate: W13 accepted. Venue: GitHub Actions hosted runner + local build. Paths: `.github/workflows/`, `policy/`, `evidence/`.
- Weekly outcome: protect the build and deployment pipeline.
- Standing tool: push a deliberately failing test on a branch, examine the required-check result, restore, verify a green merge.

D066 - Build provenance

- Perform: pin reusable action versions and least token permissions.
- Submit: `.github/workflows/build.yml` + ADR; `evidence/W14/D066/`.
- PASS: the workflow uses least required token permissions.
- Negative control: an over-privileged token change is rejected in review.
- Map: A03/A08; SLSA-style provenance.

D067 - Image assurance

- Perform: scan the built image and document base-image updates.
- Submit: `reports/container-scan.json`; `evidence/W14/D067/`.
- PASS: a known high-risk package triggers the gate.
- Negative control: a clean image passes.
- Map: A03; container hygiene.

D068 - Signed artifacts

- Perform: generate artifact checksums and a build-provenance record.
- Submit: `releases/manifest.sha256`; `evidence/W14/D068/`.
- PASS: a downloaded artifact hash verifies.
- Negative control: a tampered artifact fails verification.
- Map: A08 (data integrity); supply chain.

D069 - Environment secrets

- Perform: separate development and production-like mock config; reuse `infra/compose.yaml`.
- Submit: `infra/env-policy.md`; `evidence/W14/D069/`.
- PASS: secrets are absent from log output and repo history.
- Negative control: a secret-scan of history is clean.

- Map: A02/A08; CWE-532 (log of sensitive info).

D070 - Release rollback

- Perform: deploy a tagged version, inject a failure, and roll back.
- Submit: `release/drill02.md` ; `evidence/W14/D070/` .
- PASS: service health is restored on the prior tag.
- Negative control: the failed version is shown unhealthy before rollback.
- Map: RECOVER; release safety.

Friday W14 integration: PR `week-14-integration` ; `assessments/week14/review.md` .

Week 15 - Purple-team measurement

- Gate: W14 accepted. Venue: QW-LNX + Wazuh + safe lab emulation. Paths: `purple/` , `detections/` , `evidence/` .
- Weekly outcome: connect attack simulations to detection outcomes.
- Standing tool: generate a pre-approved synthetic security event; link an ATT&CK hypothesis, the raw event, and a detection-rule test. Use ATT&CK v18+ Detection Strategies and Analytics objects.

D071 - ATT&CK mapping

- Perform: map lab actions to relevant ATT&CK techniques accurately.
- Submit: `purple/attack-map.csv` ; `evidence/W15/D071/` .
- PASS: each technique mapping is supported by observable steps.
- Negative control: an unobserved technique is not claimed as covered.
- Map: ATT&CK v19 technique ids.

D072 - Safe emulation

- Perform: execute a benign authorized emulation test with expected events.
- Submit: `purple/scenarios/01/` ; `evidence/W15/D072/` .
- PASS: RoE and cleanup are verified.
- Negative control: post-run cleanup leaves no residual state.
- Map: adversary emulation (safe).

D073 - Detection validation

- Perform: reproduce the signal in Wazuh and capture event linkage.
- Submit: `detections/purple01.yaml` + event ids; `evidence/W15/D073/` .
- PASS: the known behavior triggers the expected alert.
- Negative control: benign activity does not trigger it.
- Map: ATT&CK Detection Strategy/Analytic; DETECT.

D074 - Gap closure

- Perform: improve missing visibility or an overly broad detection.
- Submit: `purple/gap-remediation.md` + PR; `evidence/W15/D074/` .
- PASS: a before/after replay demonstrates the gain.
- Negative control: the previously missed event is now detected; benign still clean.
- Map: detection engineering.

D075 - Purple review

- Perform: report coverage vs gaps with precision metrics.
- Submit: `purple/metrics.csv` + briefing; `evidence/W15/D075/` .
- PASS: includes limitations and no inflated coverage.
- Negative control: you state which techniques remain uncovered.
- Map: honest measurement.

Friday W15 integration: PR `week-15-integration` ; `assessments/week15/review.md` .

Week 16 - DevSecOps and purple-team board

- Gate: W15 accepted. Venue: GitHub Actions + local SIEM/API. Paths: `tests/` , `policy/` , `purple/` , `evidence/` .
- Weekly outcome: prove a policy-based security feedback loop.
- Standing tool: execute benign and suspicious fixtures before/after remediation; archive the workflow URL and detector output.

D076 - Policy checks

- Perform: implement a pipeline policy for images, dependencies, and deploy rules.

- Submit: `policy/ + tests; evidence/W16/D076/` .
- PASS: an unauthorized change blocks the merge.
- Negative control: a compliant change merges.
- Map: `policy-as-code; A08`.

D077 - Regression suite

- Perform: replay benign + attack scenarios after code changes; run via CI.
- Submit: `tests/end-to-end/ ; evidence/W16/D077/` .
- PASS: all required assertions pass.
- Negative control: a regression reintroduced on a branch is caught.
- Map: `regression safety`.

D078 - Fault scenario

- Perform: kill a log source or alter a schema and detect ingestion failure.
- Submit: `soc/pipeline-failure.md ; evidence/W16/D078/` .
- PASS: a health alert triggers within the stated limit.
- Negative control: a healthy pipeline does not alert.
- Map: `DETECT (DE.CM-3); A09`.

D079 - Joint exercise

- Perform: run a red action, blue investigation, engineering fix, and retest.
- Submit: `purple/engagement02.md ; evidence/W16/D079/` .
- PASS: cross-linked events and commits.
- Negative control: the fix's regression test fails on the vulnerable commit.
- Map: `purple loop`.

D080 - BOARD 4

- Perform: perform a challenge-driven secure release under observation.
- Submit: `assessment/month04/ + tag board4 (release v4.0); evidence/W16/D080/` .
- PASS: a broken build and an attacker signal are resolved end-to-end.
- Negative control: the injected vulnerable sample blocks the pipeline; the good sample passes.
- Map: see Board 4 in Section 13.

Friday W16 integration: `PR week-16-integration ; assessments/week16/review.md` . **EXAMINATION GATE: Board 4 must pass (>= 95) before Week 17 opens.**

Week 17 - Security architecture foundations

- Gate: W16 accepted and Board 4 pass. Venue: QW-LNX + Mermaid/diagrams + Docker topology. Paths: `architecture/ , evidence/` .
- Weekly outcome: translate risks into enforceable architecture.
- Standing tool: derive real trust boundaries from `docker inspect` and compose files; validate each diagram against configured listeners (`ss -lnt`).

D081 - Assets and trust

- Perform: produce context/container/network diagrams with actors and data.
- Submit: `architecture/context.md + diagram; evidence/W17/D081/` .
- PASS: the examiner can identify trust boundaries from your diagram.
- Negative control: the diagram matches the running topology, not an idealized one.
- Map: `C4 model; threat modeling`.

D082 - Identity architecture

- Perform: specify service identities, token scope, and a revocation model.
- Submit: `architecture/identity.md ; evidence/W17/D082/` .
- PASS: least privilege shown by a denied test.
- Negative control: an over-scoped token request is rejected.
- Map: `A01/A07; PR.AA`.

D083 - Network architecture

- Perform: implement segmentation and controlled egress.
- Submit: `infra/netpol/ + probes; evidence/W17/D083/` .
- PASS: denied traffic is evidenced by packet/flow.
- Negative control: an allowed flow passes; a denied flow is dropped and logged.

- Map: PR:IR; CWE-284.

D084 - Cryptography architecture

- Perform: document TLS and encryption choices, key lifecycles, and rotation.
- Submit: `architecture/crypto.md` ; `evidence/W17/D084/` .
- PASS: you explain key-compromise recovery.
- Negative control: an expired/rotated key is shown rejected.
- Map: A04; CWE-320 (key management).

D085 - Architecture review

- Perform: defend security/usability tradeoffs via an ADR.
- Submit: `architecture/ADR-001.md` + recording; `evidence/W17/D085/` .
- PASS: alternatives and residual risks are documented.
- Negative control: a rejected alternative is recorded with the reason.
- Map: ADR practice.

Friday W17 integration: PR `week-17-integration` ; `assessments/week17/review.md` .

Week 18 - Infrastructure as code and resilience

- Gate: W17 accepted. Venue: QW-LNX + local IaC scripts/OpenTofu. Paths: `infra/iac/` , `resilience/` , `evidence/` .
- Weekly outcome: provision, verify, recover, and destroy predictably.
- Standing tool: implement and verify a disposable local resource only; capture `plan` /dry-run, apply, verification, and destroy.

D086 - IaC foundation

- Perform: define test infrastructure with OpenTofu/Terraform or scripted local provisioning.
- Submit: `infra/iac/` + README; `evidence/W18/D086/` .
- PASS: a fresh setup runs without undeclared state.
- Negative control: a second apply is a no-op (no drift).
- Map: IaC; reproducibility.

D087 - Configuration drift

- Perform: detect deviation between declared config and the running system.
- Submit: `tests/drift/` + evidence; `evidence/W18/D087/` .
- PASS: injected drift is detected.
- Negative control: a clean system reports no drift.
- Map: NIST CSF 2.0 DETECT.

D088 - Backup design

- Perform: back up configuration and sanitized test data securely.
- Submit: `recovery/backup.sh` + restore notes; `evidence/W18/D088/` .
- PASS: a checksum validates the restored copy.
- Negative control: a corrupted backup fails the checksum.
- Map: RECOVER (RC.RP).

D089 - Failure recovery

- Perform: simulate a component failure and measure recovery time.
- Submit: `recovery/scenario03.md` ; `evidence/W18/D089/` .
- PASS: RTO/RPO are measured, not guessed.
- Negative control: the failed state fails the probe before recovery.
- Map: resilience; RECOVER.

D090 - Resilience review

- Perform: evaluate logging loss, credential loss, and single-point failure.
- Submit: `architecture/failure-modes.csv` ; `evidence/W18/D090/` .
- PASS: mitigation plans for the top five risks.
- Negative control: each listed failure mode has a tested mitigation.
- Map: A09; RESPOND/RECOVER.

Friday W18 integration: PR `week-18-integration` ; `assessments/week18/review.md` .

Week 19 - Security operations management

- Gate: W18 accepted. Venue: QW-LNX + lab operating logs. Paths: `governance/`, `runbooks/`, `evidence/`.
- Weekly outcome: create operational practices and governance artifacts.
- Standing tool: tie a named control and an operational owner to an observed test event, an issue, and a remediation PR, not generic policy prose.

D091 - Risk register

- Perform: score risks with probability, impact, owner, and treatment.
- Submit: `governance/risk-register.csv`; `evidence/W19/D091/`.
- PASS: every high risk has an accountable treatment.
- Negative control: a risk with no owner is flagged as a gap.
- Map: NIST CSF 2.0 GOVERN (GV.RM).

D092 - Control mapping

- Perform: link evidence to selected NIST CSF 2.0 categories and relevant controls.
- Submit: `governance/control-map.csv`; `evidence/W19/D092/`.
- PASS: claims trace to configuration or a test.
- Negative control: an unsupported control claim is removed.
- Map: NIST CSF 2.0; traceability.

D093 - SLA and runbooks

- Perform: write an incident severity matrix and escalation procedures.
- Submit: `runbooks/` + `soc/severity-sla.md`; `evidence/W19/D093/`.
- PASS: the examiner executes one runbook accurately.
- Negative control: a runbook step that is wrong is caught during execution.
- Map: NIST SP 800-61; RESPOND.

D094 - Metrics and quality

- Perform: define MTTD/MTTR with defensible calculations.
- Submit: `metrics/operations.csv`; `evidence/W19/D094/`.
- PASS: metric formula and sample period are disclosed.
- Negative control: you state the metric's limitations.
- Map: operational metrics.

D095 - Stakeholder briefing

- Perform: explain the threat posture to technical and executive audiences.
- Submit: `reports/security-review.md` + recording; `evidence/W19/D095/`.
- PASS: findings are prioritized without unsupported assurance.
- Negative control: you avoid any unverifiable guarantee.
- Map: communication.

Friday W19 integration: PR `week-19-integration`; `assessments/week19/review.md`.

Week 20 - Architecture qualification board

- Gate: W19 accepted. Venue: QW-LNX + live local lab + board. Paths: `architecture/`, `assessments/board05/`, `evidence/`.
- Weekly outcome: defend a secure operational design.
- Standing tool: show a denial test, fault injection, a restore command, and UTC recovery measurements against a declared objective.

D096 - Design package

- Perform: freeze a coherent reference architecture and assets.
- Submit: `architecture/review-package/`; `evidence/W20/D096/`.
- PASS: diagrams match the live configuration.
- Negative control: a diagram-vs-reality mismatch would be detectable (and is absent).
- Map: architecture integrity.

D097 - Security tests

- Perform: test ACLs, TLS, authentication, and logs against requirements; run via CI.
- Submit: `tests/architecture/` + evidence; `evidence/W20/D097/`.
- PASS: each security control has a negative test.
- Negative control: each control's negative test fails on a weakened config.
- Map: ASVS; verification.

D098 - Fault injection

- Perform: perform a service-outage or credential-rotation exercise.
- Submit: `recovery/fault04.md` ; `evidence/W20/D098/` .
- PASS: recovery without exposing a secret.
- Negative control: the rotated credential is rejected after rotation.
- Map: RECOVER; A04.

D099 - Threat model revision

- Perform: update trust boundaries and risk decisions based on the tests.
- Submit: `architecture/threat-model-v2.md` ; `evidence/W20/D099/` .
- PASS: new risks are tracked with an owner.
- Negative control: a closed risk is shown closed with evidence.
- Map: iterative threat modeling.

D100 - BOARD 5

- Perform: live architecture defense and break/fix/recovery.
- Submit: `assessment/month05/ + tag board5 (release v5.0)`; `evidence/W20/D100/` .
- PASS: independent rebuild and justified decisions.
- Negative control: access denied as designed; restoration within the declared objective.
- Map: see Board 5 in Section 13.

Friday W20 integration: PR `week-20-integration` ; `assessments/week20/review.md` . **EXAMINATION GATE: Board 5 must pass (>= 95) before Week 21 opens.**

Week 21 - Integrated capstone planning

- Gate: W20 accepted and Board 5 pass. Venue: QW-LNX + scoped integrated range. Paths: `capstone/` , `scope/` , `evidence/` .
- Weekly outcome: define the final adversary/defender exercise.
- Standing tool: freeze in-scope hostname/ports, role identities, and scenario assumptions in a board-approved capstone charter.

D101 - Engagement charter

- Perform: write the capstone scenario, roles, scope, exclusions, and safety signoff.
- Submit: `capstone/charter.md` ; `evidence/W21/D101/` .
- PASS: the examiner approves the defined safety boundary.
- Negative control: the charter explicitly forbids all non-loopback targets.
- Map: RoE; GOVERN.

D102 - Detection acceptance

- Perform: state testable telemetry and coverage criteria.
- Submit: `capstone/acceptance.yaml` ; `evidence/W21/D102/` .
- PASS: criteria have reproducible test ids.
- Negative control: an untestable criterion is rewritten or removed.
- Map: acceptance testing.

D103 - Threat hypothesis

- Perform: build attack/defend hypotheses using real trust paths.
- Submit: `capstone/threat-hypotheses.md` ; `evidence/W21/D103/` .
- PASS: each hypothesis ties to a control and evidence.
- Negative control: a hypothesis with no control is removed.
- Map: ATT&CK-driven planning.

D104 - Readiness audit

- Perform: conduct a clean-room rebuild from the repo and record defects.
- Submit: `capstone/rebuild-log.md` ; `evidence/W21/D104/` .
- PASS: another reviewer completes the documented build.
- Negative control: an undocumented step surfaces as a defect to fix.
- Map: `docs/REBUILD.md` ; reproducibility.

D105 - Go/no-go checkpoint

- Perform: freeze a candidate release and identify blockers.
- Submit: `capstone/readiness.md` ; `evidence/W21/D105/` .

- PASS: no critical unsolved safety/control failures.
- Negative control: any open critical blocker halts go.
- Map: release readiness.

Friday W21 integration: PR `week-21-integration` ; `assessments/week21/review.md` .

Week 22 - Integrated capstone implementation

- Gate: W21 accepted. Venue: QW-LNX + red/blue local tools. Paths: `capstone/` , `cases/` , `reports/` , `evidence/` .
- Weekly outcome: run controlled red-blue-purple workflows.
- Standing tool: collect a matched red request id, a blue timestamp, and a rule/case reference; show a safe fix and a negative control.

D106 - Red engagement

- Perform: perform an approved gray-box or black-box lab validation.
- Submit: `capstone/red-report.md` ; `evidence/W22/D106/` .
- PASS: findings show scoped reproducibility.
- Negative control: no out-of-scope action occurs.
- Map: PTES; A01.

D107 - Blue response

- Perform: detect the scenario, create an incident case, contain, and recover.
- Submit: `capstone/blue-case.md` ; `evidence/W22/D107/` .
- PASS: an end-to-end timestamped event chain.
- Negative control: a benign baseline produces no incident.
- Map: NIST SP 800-61; RESPOND.

D108 - Purple feedback

- Perform: improve a rule or control based on missed or noisy activity.
- Submit: `capstone/purple-improvement.md` ; `evidence/W22/D108/` .
- PASS: a before/after replay supports the change.
- Negative control: benign activity still produces no alert after the change.
- Map: detection engineering.

D109 - Engineering patch

- Perform: fix a finding and ship it through blocked/approved CI gates.
- Submit: `capstone/remediation-pr.md` ; `evidence/W22/D109/` .
- PASS: the fix is linked to a passing security test.
- Negative control: the test fails on the pre-fix commit.
- Map: remediation verification.

D110 - Mid-capstone demo

- Perform: present the connected workflow without hidden manual steps.
- Submit: `capstone/demo-week22.md` ; `evidence/W22/D110/` .
- PASS: a reviewer traces an event to a commit and an alert.
- Negative control: there is no undocumented manual step.
- Map: integration.

Friday W22 integration: PR `week-22-integration` ; `assessments/week22/review.md` .

Week 23 - Portfolio delivery and public proof

- Gate: W22 accepted. Venue: local `portfolio/site` + Vercel. Paths: `portfolio/site/` , `public-evidence/` , `evidence/` .
- Weekly outcome: communicate real outcomes without publishing insecure infrastructure.
- Standing tool: `cd portfolio/site && npm install && npm run build` ; Vercel project root `portfolio/site` , output `dist` , publish only sanitized files. Never publish the API or vulnerable targets.

D111 - Portfolio architecture

- Perform: design an accessible Vercel static or server-rendered portfolio.
- Submit: `portfolio/site/` + `sitemap`; `evidence/W23/D111/` .
- PASS: no test-exploit target is exposed.
- Negative control: a scan of the deployed site finds no app/API/SIEM endpoints.
- Map: safety invariant; A01.

D112 - Projects and evidence

- Perform: publish sanitized case studies and architecture diagrams.
- Submit: `portfolio/content/projects/` ; `evidence/W23/D112/` .
- PASS: each case links an evidence release and states limitations.
- Negative control: no secret or internal endpoint appears in published content.
- Map: responsible disclosure.

D113 - Deployment pipeline

- Perform: connect GitHub to Vercel with preview/production checks; report the workflow URL and SHA.
- Submit: `portfolio/.github/workflows/` ; `evidence/W23/D113/` .
- PASS: a bad build cannot become a published release.
- Negative control: a failing build is blocked from production.
- Map: release safety.

D114 - Security review

- Perform: run headers, dependency, and secret checks against the portfolio.
- Submit: `portfolio/security-review.md` ; `evidence/W23/D114/` .
- PASS: no embedded tokens, credentials, or internal endpoints.
- Negative control: a secret-scan of the site build is clean.
- Map: A02/A08.

D115 - Hiring narrative

- Perform: build a role-specific story and interview answers from evidence.
- Submit: `career/STAR-stories.md` + resume; `evidence/W23/D115/` .
- PASS: every claimed skill is backed by a repository artifact.
- Negative control: an unbacked claim is removed.
- Map: evidence-based narrative.

Friday W23 integration: PR `week-23-integration` ; `assessments/week23/review.md` .

Week 24 - Blind qualification simulations

- Gate: W23 accepted. Venue: QW-LNX + examiner-supplied local fixtures. Paths: `assessments/mock/` , `evidence/` .
- Weekly outcome: rehearse unannounced troubleshooting and incident panels.
- Standing tool: solve a surprise but scoped break/fix; record commands and proof before the examiner reveals the expected behavior.

D116 - Blind SOC scenario

- Perform: investigate unseen sanitized events with an unknown initial label.
- Submit: `assessments/mock-soc.md` ; `evidence/W24/D116/` .
- PASS: reasoning distinguishes evidence from hypothesis.
- Negative control: you state what the evidence does not support.
- Map: NIST SP 800-61.

D117 - Blind network scenario

- Perform: diagnose a broken TLS/DNS/firewall private service.
- Submit: `assessments/mock-network.md` ; `evidence/W24/D117/` .
- PASS: restore without disabling security controls.
- Negative control: you never bypass TLS validation to "fix" it.
- Map: fundamentals; A04.

D118 - Blind secure-build scenario

- Perform: remediate a surprise pipeline failure and assess the supply chain.
- Submit: `assessments/mock-devsecops.md` ; `evidence/W24/D118/` .
- PASS: the fix is verified with tests and a PR.
- Negative control: the pre-fix pipeline fails; the post-fix passes.
- Map: A03/A08.

D119 - Blind design defense

- Perform: explain identity, segmentation, and recovery under examiner questions.
- Submit: `assessments/mock-architecture.md` ; `evidence/W24/D119/` .
- PASS: no reliance on memorized tool UI; you reason from the system.

- Negative control: you correct a deliberately flawed premise if asked.
- Map: architecture reasoning.

D120 - Portfolio audit

- Perform: an independent reviewer validates public links and release provenance.
- Submit: `career/portfolio-audit.md` ; `evidence/W24/D120/` .
- PASS: all links live; no vulnerable target is public.
- Negative control: a scan confirms no exposed app/API/SIEM.
- Map: safety invariant.

Friday W24 integration: PR `week-24-integration` ; `assessments/week24/review.md` .

Week 25 - Final board and remediation

- Gate: W24 accepted. Venue: fresh QW-LNX clone + live board. Paths: `assessment/final/` , `capstone/` , `evidence/` .
- Weekly outcome: earn the capstone qualification through independent performance.
- Standing tool: rebuild from a clone, run full checks, perform the approved exercise and remediation, publish tagged release v6.0.

D121 - Final rebuild

- Perform: the examiner begins from the tagged repo on a clean machine.
- Submit: `assessment/final/rebuild.md` ; `evidence/W25/D121/` .
- PASS: documented provision, tests, and teardown all pass.
- Negative control: a clean clone requires zero manual edits.
- Map: `docs/REBUILD.md` .

D122 - Red/Blue exercise

- Perform: scoped discovery, validate a weakness, detect and investigate.
- Submit: `assessment/final/engagement.md` ; `evidence/W25/D122/` .
- PASS: red evidence links to blue events.
- Negative control: a benign baseline produces no incident.
- Map: purple loop.

D123 - Fix/retest

- Perform: demonstrate root-cause correction, detection tuning, and a CI guardrail; run via CI.
- Submit: `assessment/final/fix-verification.md` ; `evidence/W25/D123/` .
- PASS: attack reproduction blocked; regression passes.
- Negative control: the regression fails on the pre-fix commit.
- Map: remediation verification.

D124 - Executive and architect panels

- Perform: a SOC case brief, a security-design defense, and a risk Q&A.
- Submit: `assessment/final/panels.md` ; `evidence/W25/D124/` .
- PASS: engineering decisions are defensible with evidence.
- Negative control: you acknowledge any residual risk honestly.
- Map: communication; live defense.

D125 - BOARD 6

- Perform: complete a blind final scenario, scorecard, feedback, and handover.
- Submit: `assessment/final/` + tag `v6.0` ; `evidence/W25/D125/` .
- PASS: ≥ 95 , all mandatory gates, verified portfolio, safety pass.
- Negative control: no public vulnerable services or leaked secrets; peer rebuild succeeds.
- Map: see Board 6 in Section 13.

Friday W25 integration: PR `week-25-integration` ; `assessments/week25/review.md` . **EXAMINATION GATE: Board 6 must pass (≥ 95) before Week 26 closeout.**

Week 26 - Remediation, hiring panels, and operational handoff

- Gate: W25 accepted and Board 6 pass. Venue: QW-LNX + GitHub + Vercel. Paths: `career/` , `handover/` , `evidence/` .
- Weekly outcome: close uncovered competency gaps and demonstrate independent readiness.
- Standing tool: an independent reviewer follows `docs/REBUILD.md` , opens the public portfolio, and checks artifact links/teardown.

D126 - Deficit closure

- Perform: resolve board-identified deficiencies through tracked pull requests.
- Submit: `assessment/remediation-tracker.md` ; `evidence/W26/D126/` .
- PASS: each blocker is independently retested.
- Negative control: a closed blocker fails its old failing test and passes the new one.
- Map: remediation.

D127 - Reverse teaching

- Perform: teach a junior engineer a lab concept and field questions.
- Submit: `career/teaching-session.md` ; `evidence/W26/D127/` .
- PASS: the reviewer reproduces your explained example.
- Negative control: a wrong explanation would fail reproduction (and does not).
- Map: mastery via teaching.

D128 - Timed role interview

- Perform: SOC/security-engineer technical interviews with evidence.
- Submit: `career/interview-results.md` ; `evidence/W26/D128/` .
- PASS: the role-specific rubric threshold is met.
- Negative control: unsupported claims are challenged and withdrawn.
- Map: readiness.

D129 - Systems handover

- Perform: transfer runbooks and hand off to a blinded peer.
- Submit: `handover/operations-pack.md` ; `evidence/W26/D129/` .
- PASS: the peer restores and operates the lab independently.
- Negative control: a missing runbook step blocks the peer (and is then fixed).
- Map: operational handoff.

D130 - Exit readiness

- Perform: sign the release checklist, publish sanitized evidence, archive the private lab.
- Submit: `career/exit-package.md` ; `evidence/W26/D130/` .
- PASS: complete repo, working portfolio, safe teardown.
- Negative control: a final scan confirms no vulnerable target is public and no secret leaked.
- Map: close-out; safety invariant.

Friday W26 integration: PR `week-26-integration` ; `assessments/week26/review.md` .

13. Independent monthly boards: concrete scenarios and outputs

Each board is an independent assessment conducted by an examiner who did not coach the work. Procedure for every board: the assessor checks your last passing release; independently spins up the environment or replays stored fixtures; introduces one controlled failure; records a 60 to 90 minute uncoached demonstration and a 20 minute oral explanation. No credit for a screenshot that cannot be recreated. You must distinguish black-box/gray-box knowledge, Red (simulated attacker), Blue (SOC), and Purple (joint validation). A diagram for each board is in `diagrams/` .

Board 1 - Infrastructure and networking (Week 4) | `diagrams/board1-infrastructure.svg`

- Examiner executes: rebuild a segmented lab from a fresh clone, demonstrate TCP/DNS/TLS reasoning, break/fix an injected misconfiguration, restore from git.
- Required outputs: running environment; `infra/` source; network evidence; CI runs; recovery record; examiner notes; release tag `board1` .
- Hard pass (≥ 95): one isolated target; negative network reachability proven; automated deployment; reproducible break/fix; secret-free repository.

Board 2 - SOC operations and investigation (Week 8) | `diagrams/board2-soc.svg`

- Examiner executes: onboard at least three logically distinct telemetry sources, fire at least three detection rules, investigate an unknown incident, preserve and present evidence.
- Required outputs: Wazuh stack; source map; detection fixtures; case file; containment/rollback record; measured latency; release tag `board2` .
- Hard pass (≥ 95): all sources show fresh events; positives and negatives demonstrated; timeline consistent; safe containment and rollback.

Board 3 - Ethical hacking and OWASP (Week 12) | `diagrams/board3-ethical-hacking.svg`

- Examiner executes: conduct approved black-box and gray-box tests against the owned local app, demonstrate two findings,

verify one scanner false positive, fix and retest.

- Required outputs: signed RoE; asset list; web test plan; report; reproducible PoCs confined to the lab; fix PR; retest; release tag `board3`.
- Hard pass (≥ 95): no scope violation; accurate OWASP Top 10:2025 / API 2023 / CWE mapping; findings reproduce before the fix and fail after.

Board 4 - Secure pipeline and purple team (Week 16) | [diagrams/board4-devsecops.svg](#)

- Examiner executes: deploy a secure build gate, emulate a safe action, show detection and a case, merge an engineered fix, replay the scenario through CI.
- Required outputs: GitHub Actions logs; SBOM; image scan; ATT&CK-mapped exercise; detection regression; release tag `board4`.
- Hard pass (≥ 95): the injected vulnerable sample blocks the pipeline; the good sample passes; red/blue evidence correlation documented.

Board 5 - Architecture and resilience (Week 20) | [diagrams/board5-architecture.svg](#)

- Examiner executes: explain and defend a security architecture, reject unauthorized access, detect drift, restore from failure, justify residual risk.
- Required outputs: ADRs; diagrams; negative controls; IaC; recovery measurements; risk register; release tag `board5`.
- Hard pass (≥ 95): model matches the production-like lab; access denied as designed; restoration within the declared objective.

Board 6 - Integrated professional qualification (Week 25, closeout Week 26) | [diagrams/board6-capstone.svg](#)

- Examiner executes: a blind scenario: deploy, discover an authorized weakness, investigate detection, contain, remediate, prove the CI gate, retest, brief leaders, and hand off.
- Required outputs: tagged `v6.0` release; public safe Vercel showcase; comprehensive capstone evidence; independent scorecard.
- Hard pass (≥ 95): all critical controls pass; peer rebuild succeeds; no public vulnerable services or leaked secrets.

14. Hosting and release sequence

- **Source:** private `qwalora-cyber-range` repo. Release tags `board1` through `board6` and `v6.0` for the completed capstone. Evidence is sanitized before any public release.
- **Security lab:** Docker Compose inside Windows WSL2 for the API and Juice Shop. Wazuh deployed locally with the official single-node configuration when capacity permits; the dashboard is restricted to the private machine. No public Vercel host for any attack surface.
- **Portfolio:** Vercel, Import Git Repository, root directory `portfolio/site`, build `npm run build`, output `dist`, deploy and verify the live production URL. Add a custom Qwalora domain only after ownership/approval and a link review.
- **Operational proof:** record the public portfolio URL, the Vercel deployment id, the reviewed GitHub commit SHA, and proof that no vulnerable endpoint is publicly reachable.

15. Authoritative upstream references

- OWASP Top 10:2025 - <https://owasp.org/Top10/>
- OWASP API Security Top 10 2023 - <https://owasp.org/API-Security/>
- OWASP WSTG - <https://owasp.org/www-project-web-security-testing-guide/>
- OWASP ASVS - <https://owasp.org/www-project-application-security-verification-standard/>
- OWASP Juice Shop - <https://owasp.org/www-project-juice-shop/>
- MITRE ATT&CK (Enterprise, current v19) - <https://attack.mitre.org/>
- NIST CSF 2.0 - <https://www.nist.gov/cyberframework>
- NIST SP 800-115 (technical assessment) - <https://csrc.nist.gov/pubs/sp/800/115/final>
- NIST SP 800-61 (incident handling) - <https://csrc.nist.gov/pubs/sp/800/61/r3/final>
- Microsoft Learn: Install WSL - <https://learn.microsoft.com/windows/wsl/install>
- Docker: WSL 2 backend - <https://docs.docker.com/desktop/features/wsl/>
- Docker Compose - <https://docs.docker.com/compose/>
- Wazuh Docker deployment - <https://documentation.wazuh.com/current/deployment-options/docker/>
- GitHub Actions - <https://docs.github.com/en/actions>
- Vercel projects / root directory - <https://vercel.com/docs/projects/overview>

Review owner: Qwalora Security Engineering Board. Version lock per cohort: document exact software releases and image digests at kickoff in `architecture/host-baseline.md`; pin the starter image and GitHub Action version tags before the official assessment.

16. Glossary

- **QW-HOST / QW-LNX:** the Windows host and the Ubuntu WSL2 Linux instance.
- **qw-api / qw-juice / qw-siem / qw-sensor:** the training API, Juice Shop, Wazuh SIEM, and the network sensor.
- **BOLA / IDOR:** broken object-level authorization; a user reads another user's object (API1:2023, CWE-639).

- **Negative control:** the test that must fail on the broken/insecure state and pass after the fix, or the forbidden flow that must be denied.
- **Machine-verifiable PASS:** a condition an examiner can reproduce by command on a clean checkout, not a screenshot.
- **Dependency gate:** the prior accepted state a day builds on; work is strictly sequential.
- **Board:** a monthly independent, uncoached examination; six total; each requires ≥ 95 .
- **Red / Blue / Purple:** simulated attacker, SOC defender, and joint validation.
- **RoE:** Rules of Engagement; the signed scope and stop rules for any assessment.

Qwalora Cyber Operations Academy | Candidate Execution Manual v4.0 | Qualify as a Ranger: ≥ 95 percent, all six boards, no AI, real work.