Reproduce before repair

DevOps & infrastructure | Advanced

Session M4-03 | 3 hours

Agenda

Time	Topic
0:00 to 0:08	Observation, hypothesis, check results
0:08 to 0:18	Local baseline
0:18 to 0:34	Pipeline investigation
0:34 to 0:44	Multi-stage failure
0:44 to 0:54	Security remediation
0:54 to 1:00	Lab handoff

Record logs and local results

Record:

```
Observation:
Local proof:
Hypothesis:
Smallest repair:
Proof after repair:
```

Do not put a root-cause answer in the observation field.

Check the application first

```
npm ci
npm test
npm run lint
npm run build
test -f dist/index.js
```

If these commands pass locally, the workflow may still use the wrong runtime, directory, environment, or job order.

Give Copilot the workflow and failed-step output

```
Read this workflow and failed-step output. Separate observations from hypotheses.
Use the local command as additional evidence. Propose the smallest repair and one
proof command. Do not change unrelated jobs.
```

Check every claim against the files.

Do not reveal the answer early

1. Read the failed step.
2. Record the exact log output.
3. Reproduce the relevant command.
4. State one hypothesis.
5. Repair one cause.
6. Check the rerun.
7. Open the reference fix.

One failure can hide another

```
first failed stage
        ↓ repair
next workflow run
        ↓
new failed stage
```

Use the new log to diagnose the next failure. Keep the repair you already checked and avoid speculative changes.

Check the affected code path

For each finding, identify:

- input source;
- dangerous sink;
- expected protection;
- focused test results;
- approval decision.

An absence of scanner findings is not enough.

Access policy

GitHub Copilot and Node.js 20.x are required.

Actions, code scanning, and remote remediation are optional. The local path uses the supplied application, logs, workflows, and syntax checks.

Lab

Keep solutions closed until each checkpoint. Case 3 has two investigation stages.