Lab 1 – Risk Path Practice Scenarios – 202610

Each scenario contains more detail than needed
Most details are irrelevant
Your job is to produce one defensible risk path per scenario

Use these scenarios exactly as written. They are intentionally noisy. Many details are irrelevant. Some details feel important but aren't. Your task is to extract the key facts and express a single, disciplined risk path.

For each scenario, students should produce:

Asset → Entry Point → Weak Spot → Damage

Then apply the Acid Test and classify CIA based only on Damage.

Scenario 1: The Shared Balance Report

A small operations team generates a daily CSV report containing customer account balances. The report is written to a shared directory on an internal file server so multiple team members can download it and answer support questions.

Initially, only two analysts had access to the directory. Over time, more staff were added to the same group to help with coverage during vacations and busy periods.
The report generation job still runs on schedule and completes successfully. No alerts fire, and customers are not notified of any issues.

Last week, a staff member mentioned seeing customer balance information while helping with an unrelated task.

Scenario 2: The Helpful Permission Change

A Linux service reads its configuration from a file that controls both logging behaviour and feature flags. During an incident, an engineer temporarily widens permissions on the configuration directory so another team can inspect settings without using elevated access. The service continues to run normally after the incident, and no restart is required. The permissions change is not tracked anywhere except in the host's shell history.

Weeks later, engineers notice that certain features appear enabled in some environments but not others, even though deployment artifacts are identical.

No one can point to a specific change request or deployment that explains the difference.

Scenario 3: The Undocumented Reprocessing Path

A data processing system runs a primary pipeline, which a service manager manages. The pipeline reads input files from a staging directory, transforms them, and writes the results to an output directory for downstream systems.

During a previous migration, an older version of the pipeline was left on the host to allow for output comparison. That older version is no longer referenced in documentation, but it still exists on disk.

An engineer recently added a small scheduled task to reprocess specific inputs for validation purposes. The task uses the older pipeline binary because it was convenient and already present.

All outputs continue to appear in the correct location, and timestamps look normal. Downstream systems continue operating without errors.

During a quarterly review, auditors find discrepancies between expected and actual transformed values but cannot determine when or how the differences were introduced.