Introduction to Cybersecurity

Job Readiness Companion

Turn vague security concerns into a defensible, reviewable risk explanation that supports decisions.

Common artifacts you would see

- Risk register entries
- Security review comments in design docs
- Audit findings written in plain, technical prose
- Incident follow-ups that explain what happened and what must change

Junior expectations

- Identify the correct asset (the thing stakeholders rely on)
- Write one complete cause → effect path (Asset → Entry Point → Weak Spot → Damage)
- Separate observed facts from assumptions
- Make damage observable (something someone would notice)

Junior guardrails

- Do not assign severity labels without explaining the path
- Do not speculate about attackers, exploits, or tools
- Do not use vague outcomes (“compromised,” “hacked”) in place of damage
- Do not widen the scope to “the whole system” when a specific asset is at stake

How issues are discussed professionally

In reviews and incidents, security concerns are framed as assets, reachability, present weaknesses, and observable harm—not labels.

A credible explanation sounds like: “Customer billing records are accessible through an internal admin endpoint that is reachable from the support subnet. The endpoint accepts unvalidated identifiers, so a user with basic access can retrieve records outside their case scope. The observable damage is unauthorized disclosure of customer PII in application logs and case notes.”

Not: “This is a critical vulnerability.”

Your job is to make the first statement possible.

Translation patterns you must be able to use

Confidentiality risk

Structure:

- Asset (what information or capability matters)
- Entry point (where access starts)
- Weak spot (present condition that makes the wrong action easy)
- Unauthorized action (what becomes possible)
- Observable damage (what someone would notice)

Example: “Customer support transcripts are stored in an S3 bucket used by the case system. The bucket policy allows read access from a broader internal role than intended. That weak spot makes it easy for any user with that role to fetch transcripts outside their assigned region. The observable damage is transcripts appearing in teams that should not see them and access logs showing unexpected principals reading objects.”

Integrity risk

Structure:

- Asset (what must remain correct)
- Entry point (where changes can be made)
- Weak spot (missing validation, missing authorization, unsafe default)
- Incorrect action (what change becomes possible)
- Observable damage (what breaks, what reports diverge, what users notice)

Example: “Pricing rules are managed through an internal configuration API used by the storefront. The API trusts a client-supplied ‘isAdmin’ flag and does not verify it on the server. That weak spot enables unauthorized rule edits. The observable damage is inconsistent prices in the storefront, audit logs showing unexpected config changes, and customer support tickets reporting mismatched totals.”

Availability risk

Structure:

- Asset (service or workflow relied on)
- Entry point (resource, endpoint, dependency)
- Weak spot (unbounded work, missing rate limiting, fragile dependency)
- Failure mode (how the service degrades or stops)
- Observable damage (alerts, error rates, user-visible symptoms)

Example: “The authentication service is relied on for all user logins. The password reset endpoint triggers an expensive lookup and email workflow without rate limiting. That weak spot allows repeated requests to saturate worker capacity. The observable damage is elevated login latency, queued reset emails arriving hours late, and on-call alerts for worker backlog and elevated 5xx rates.”

How this skill is used on the job

Security reviews

- Question this helps answer: “What could go wrong, through what path, and what would we see?”
- Inputs you work from: design docs, proposed workflows, trust boundaries, data flows
- What your explanation enables: targeted mitigations tied to specific assets and present weaknesses

Incident analysis (post-incident learning)

- Question this helps answer: “What happened, what was possible, and why did impact occur?”
- Inputs you work from: logs, tickets, timelines, access records, system behaviour
- What your explanation enables: remediation that addresses the actual weak spot and prevents recurrence

Audit and compliance responses

- Question this helps answer: “What is the control, where is it enforced, and what evidence supports it?”
- Inputs you work from: configurations, access reviews, control descriptions, evidence requests
- What your explanation enables: precise mapping from requirement → control → evidence without hand-waving.

Interview translation

Question: “How do you decide what matters most in a system?”

Answer: “I start by naming the asset the business actually relies on. I explain, using a standard workflow, how that asset becomes reachable and what present condition makes the wrong action too easy. I end with the specific, observable harm someone would notice. Once the harm is concrete, I compare issues by impact and reversibility and choose what to address first.”

Answer pattern:

- Name the asset
- Explain one believable path to harm (entry point → weak spot → damage)
- Separate observed facts from assumptions
- Make damage observable
- Justify priority using impact and reversibility

Common early-career mistakes

- Jumping to severity labels (“critical/high”) before explaining the asset and path
- Using attacker vocabulary prematurely (implying exploits or intent without evidence)
- Treating CIA labels (confidentiality/integrity/availability) as the explanation instead of a category
- Widening scope (“the whole system is vulnerable”) instead of naming the specific asset at stake
- Describing outcomes as “compromised” rather than stating what would be observable and reviewable

What good looks like / If you forget everything else

- You can name the asset and the stakeholders who rely on it
- You can state one complete, believable path (entry point → weak spot → observable damage)
- You separate what you saw from what you inferred
- Your damage statement is something a non-security teammate could verify
- Your explanation is tight enough that a reviewer could propose a targeted fix without guessing

If a teammate can repeat your explanation and reach the same conclusion, the work is job-ready.