AI Security Engineering

SEC410

Build the AI components security work now runs on, a hunting agent, an IR agent, MCP servers, a fuzzing harness and an intel pipeline, and prove each one works before you trust it: tested against known answers, hardened against the attacks aimed at it.

Modules	21
Lessons	156
Phases	4
Free preview	Foundations of AI Security Engineering (13 lessons)

The method

Build it, then the lab says whether it works. Every module has you build a component and test it against evidence where the answer is known. The check reports what it found, missed and invented, what it did without authority, and what it cost, and it fails until your work holds.

Controls that hold whatever the model decides. A model's decisions are tendencies. Each module shows a failure in a trace, shows the obvious fix failing where it does, and builds the control in code that holds regardless, so you know which of your fixes are guarantees and which only change the odds.

Your own repository. Every module adds to one repository you build across the course: agents, tools, guards and their tests. At the end it is working, tested evidence of what you can build.

Full syllabus

Phase 1: Agents That Find

Foundations of AI Security Engineering FREE PREVIEW

The foundations of every module in the course, one lesson each: how an agent works, fuzzing, LLM application failures, MCP, authority in response, machine learning, software identity, the supply chain, privacy, testing AI, threat intelligence and disclosure.

1. 0.1 How an Agent Works
2. 0.2 Fuzzing Fundamentals
3. 0.3 How LLM Applications Fail
4. 0.4 Tool Protocols and MCP
5. 0.5 Autonomy and Authority in Response
6. 0.6 Machine Learning for Defenders
7. 0.7 Identity for Software
8. 0.8 Software Supply Chain Basics
9. 0.9 Privacy Foundations
10. 0.10 Testing AI Systems
11. 0.11 Threat Intelligence Foundations
12. 0.12 Vulnerability Research and Disclosure
13. Check My Knowledge

Building a Hunting Agent

A hunting agent built and measured against planted answers: the agent loop and its cost, the schema and query validation, truncation, pivoting and stacking, read-only authority, accounting for every entity, grounding, and self-vouching evidence.

1. 1.1 A Hunt an Agent Can Run
2. 1.2 The Agent Loop: Steps, Stopping and Cost
3. 1.3 What the Model Needs to Know About the Data
4. 1.4 Validating Model-Written Queries
5. 1.5 Results That Say What They Left Out
6. 1.6 Pivoting and Stacking
7. 1.7 Read-Only Authority and Enrichment
8. 1.8 Accounting for Every Entity
9. 1.9 Invention, and Measuring Across Trials
10. 1.10 Evidence That Vouches for Itself, and the Report
11. Check My Knowledge

AI-Guided Fuzzing and Crash Triage

A fuzzer for a DNS parser with eight planted bugs: coverage, feedback and oracles, seeds and a generator configuration from a model, a measured comparison on equal budgets, triage by root cause, fixes with regression tests, and an audit of the model.

1. 2.1 The Target and Its Known Bugs
2. 2.2 A Baseline Fuzzer
3. 2.3 Coverage as Feedback
4. 2.4 Oracles Beyond Crashes
5. 2.5 Seeds a Model Proposes
6. 2.6 A Grammar the Model Configures
7. 2.7 Measuring Against the Baseline
8. 2.8 Crash Triage by Root Cause
9. 2.9 Fixing and Regression-Testing
10. 2.10 What the Model Got Wrong About the Target
11. Check My Knowledge

Phase 2: Securing What You Build

Securing an LLM Application

An alert assistant for a SOC, built and secured: attacker text in every alert, injected instructions, output that acts in a browser, a ticket or a query, case data filtered by access, and every guard measured across trials.

1. 3.1 The Alert Assistant You Will Secure
2. 3.2 Where Attacker Text Enters a Security Tool
3. 3.3 Injection Through the Evidence
4. 3.4 Output That Reaches a Ticket, a Query or a Browser
5. 3.5 Case Data Only for the Analyst Who May See It
6. 3.6 Guards That Hold and Guards That Lower the Rate
7. 3.7 The Attack Suite, Before and After
8. Check My Knowledge

Building and Securing MCP Servers

An MCP server for the hunt tools and EDR isolation, built with the Python SDK and secured: descriptions as instructions, authorization on every call, token audience, the confused deputy, a hostile server in the host, and a gate.

1. 4.1 Exposing Security Tools Over MCP
2. 4.2 Your Tool Descriptions Are Instructions
3. 4.3 Authorization on Every Call
4. 4.4 Tokens: Audience, Scope and No Passthrough
5. 4.5 The Confused Deputy
6. 4.6 A Malicious Server in the Agent’s List
7. 4.7 The Hardened Server, Tested
8. Check My Knowledge

Building an IR Agent

An incident response agent that contains attacks through an EDR, a directory and a firewall, bounded by a policy, approvals bound to the exact action, a check against injected evidence and an action ledger that can undo.

1. 5.1 What an IR Agent May Do
2. 5.2 Triage Tools and the Evidence Bundle
3. 5.3 Containment Tools
4. 5.4 Approving a Containment Action
5. 5.5 Acting on Injected Evidence
6. 5.6 Rollback and the Action Ledger
7. 5.7 The IR Agent Against a Full Incident
8. Check My Knowledge

Phase 3: Models, Identities and Data

Adversarial ML Against Your Own Model

A detection classifier of your own, attacked by evasion, by poisoning its retrain and by extraction through its scoring service, with each defense measured against what it costs.

1. 6.1 A Detection Classifier of Your Own
2. 6.2 Evasion: Moving a Sample Across the Line
3. 6.3 Evasion an Attacker Could Actually Do
4. 6.4 Poisoning the Training Data
5. 6.5 Extraction Through the Model’s Answers
6. 6.6 Defenses and What They Cost
7. 6.7 The Robustness Report
8. Check My Knowledge

Non-Human Identity and Agent Authentication

The identities AI agents act with: an inventory, a local identity provider, short-lived scoped tokens, token exchange for delegation, secrets kept out of the model, revocation mid-run and identity tests for every agent.

1. 7.1 The Identities Your Agents Already Have
2. 7.2 A Local Identity Provider
3. 7.3 Short-Lived, Scoped Credentials
4. 7.4 Acting for a User: Token Exchange
5. 7.5 Secrets the Model Must Never Read
6. 7.6 Revocation in the Middle of a Run
7. 7.7 The Identity Tests
8. Check My Knowledge

AI-Written Code and the AI Supply Chain

What an assistant’s work brings into a repository: its code measured, imports and dependencies gated, models pinned and loaded only as data, detection rules tested, and one gate on every commit.

1. 8.1 Measuring Your Own Assistant's Code
2. 8.2 Packages That Do Not Exist
3. 8.3 A Dependency Gate
4. 8.4 Models as Dependencies
5. 8.5 Model Formats That Run Code
6. 8.6 Reviewing AI-Written Detection Logic
7. 8.7 The Gate in the Pipeline
8. Check My Knowledge

Privacy-Preserving AI

Personal data in a security agent, measured and controlled: what reaches the model, redaction and what it misses, keyed pseudonyms, differential privacy on released counts, logs without content, and privacy tests for any agent.

1. 9.1 What Personal Data Reaches the Model
2. 9.2 Redaction Before the Model
3. 9.3 What Redaction Misses
4. 9.4 Pseudonymization That Still Lets You Hunt
5. 9.5 Differential Privacy on Aggregates
6. 9.6 Logging Without Keeping the Prompt
7. 9.7 The Privacy Tests
8. Check My Knowledge

Phase 4: Red Teaming, Intelligence and the Capstone

Automated AI Red Teaming

garak and PyRIT against your own alert assistant and IR agent: what a scan proves, findings reproduced and traced, fixes rerun, and a release decision from thresholds set first.

1. 10.1 What an Automated Scan Can Prove
2. 10.2 garak Against Your Alert Assistant
3. 10.3 PyRIT: Targets, Converters and Scorers
4. 10.4 Multi-Turn Attacks
5. 10.5 Triaging Scanner Findings
6. 10.6 Fixing and Rerunning
7. 10.7 Red Teaming as a Release Gate
8. Check My Knowledge

Building an AI Threat Intelligence Pipeline

Model extraction from threat reports made safe to publish: a labeled corpus, a pattern baseline, every value checked against its source, techniques against ATT&CK, roles and confidence, STIX 2.1, and a release gate.

1. 11.1 The Corpus and Its Ground Truth
2. 11.2 Extracting Indicators
3. 11.3 Indicators That Are Not in the Source
4. 11.4 Mapping to ATT&CK
5. 11.5 Enrichment and Confidence
6. 11.6 Output as STIX 2.1
7. 11.7 The Pipeline Scorecard
8. Check My Knowledge

Capstone: Find, Fix, Disclose

The whole loop on a codebase you have not seen: an assistant’s review and bandit, a test per finding, fixes chosen inside a budget, a disclosure per bug, a generated portfolio and a defended release recommendation.

1. 12.1 The Codebase and the Brief
2. 12.2 AI-Assisted Code Review
3. 12.3 Confirming a Finding
4. 12.4 Choosing What to Fix
5. 12.5 Fixing With Tests
6. 12.6 Writing the Disclosure
7. 12.7 The Repository as Portfolio
8. 12.8 The Release Recommendation, Graded
9. Check My Knowledge

Phase 0: Course Resources

AI Security Engineering Cheatsheets

One sheet per component the course builds: the question, the check that settles it as the lab implements it, how to use it and what it does not prove.

1. The Hunting Agent
2. AI-Assisted Fuzzing
3. The Alert Assistant
4. MCP Servers
5. The IR Agent
6. Adversarial ML
7. Non-Human Identity
8. The AI Supply Chain
9. Privacy
10. Automated Red Teaming
11. Threat Intelligence
12. The Capstone
13. Lookup Values

AI Security Engineering Cookbooks

Eight runbooks for building an AI security component and proving it works, each built in the lab and checked by its harness.

1. A Hunting Agent That Reports Only What It Saw
2. An Assistant That Reads Attacker-Written Text
3. An Agent That Contains Behind Approvals
4. An MCP Server That Checks Every Call
5. An Agent Identity That Expires
6. A Gate for AI-Written Code and Models
7. A Scanner Run to a Release Decision
8. Model-Extracted Intelligence to a Validated Bundle

Lab Setup

Installing the SEC410 lab, the packages some lessons add, an optional local model, and checking the lab works before the first exercise.

1. Installing the Lab
2. Module Packages
3. An Optional Local Model
4. Verify the Lab

Walkthroughs

Three cases worked end to end across the modules, including the wrong turns, each ending in a scorecard, a record and the point that transfers.

1. The Hunting Agent Before It Runs Unattended
2. An Approved Isolation of the Wrong Host
3. An Assistant-Drafted Repository Before It Merges

Playbooks

Eight playbooks, one per thing that lands on a team running AI security components: what to check first, how to contain it, and what the record must say.

1. An Agent Reported Something Not in the Data
2. An Agent Took an Action Nobody Asked For
3. An Agent Credential Is Exposed
4. An MCP Server Changed or Appeared
5. A Dependency or Model Is Suspect
6. Personal Data Found in an Agent Log
7. A Vulnerability Reported in a Component
8. An Extracted Indicator Blocked Something Legitimate

Playground

The lab in free mode: every command that runs a course component on its own, outside a check, by module.

1. The Lab in Free Mode

References

The standards, protocols and tools the course uses, with the version the lab pins and where each is defined.

1. Standards and Protocols
2. Tools and Versions

Course Completion

Course Exam

AI Security Engineering end-of-course exam: 40 questions in four parts, from the hunting agent to the capstone, ending on a case study.

1. Course Completion. AI Security Engineering

Generated 2026-11-19 from the published course. The current syllabus is always at ridgelinecyber.com/training/courses/ai-security-engineering/