Security Automation Engineering

SEC202

The engineering behind security automation that stays under control. One playbook, built lesson by lesson in Azure Logic Apps against the Microsoft Sentinel, Defender for Endpoint and Graph APIs: specification and cases, least-privilege identities, evidence with completeness labels, matched approval, verified containment, duplicate and failure handling, and a tested deployment package. The foundation for SEC412 Microsoft Security Automation.

Modules	21
Lessons	134
Phases	4
Free preview	Setup and First Run (5 lessons)

The method

Built and run, every lesson. Each of the 96 lessons changes the playbook and runs it: in the real Logic Apps runtime on your machine, against a harness that plays the Microsoft APIs from their documented shapes and lists every choice it had to make. Every run shown in a lesson is a run that happened.

Specified as cases. The playbook's decisions are written as cases that send an incident and check the reply, the run and the comment. A change that fails a case is reverted or decided, and the capstone's cases check your playbook the way they checked the reference.

Six investigations. A suspicious sign-in, endpoint execution, a reported phish, a malicious indicator, repeated incident updates and a failed containment, each with the control that makes automating it safe, and each worked end to end in the course resources. Two end with no action, because the evidence did not support one.

Full syllabus

Phase 1: Building the Workflow

Setup and First Run FREE PREVIEW

What the course builds and how it is built, the tools and their versions, installing them with the lab pack, and your first workflow run in the local runtime against the harness.

1. 0.1 What This Course Builds
2. 0.2 How the Course Is Built
3. 0.3 The Tools and Their Versions
4. 0.4 Installing the Tools and the Lab Pack
5. 0.5 Start Here: Your First Run

Your First Security Playbook

Build a Logic Apps Standard security playbook and run it locally: read an incident safely, decide, look up devices in Defender, comment on the incident and fail with a record.

1. 1.1 The Incident Arrives
2. 1.2 Reading the Incident
3. 1.3 Deciding
4. 1.4 Entities
5. 1.5 Calling Out
6. 1.6 Writing Back
7. 1.7 When a Step Fails
8. 1.8 Exercise: Your First Playbook

Designing the Workflow

Specify a security playbook as cases and run them on every change: decisions, inputs, outputs, components, stop conditions and regression checks.

1. 2.1 The Procedure Behind the Playbook
2. 2.2 The Decision Table
3. 2.3 Inputs as a Contract
4. 2.4 Outputs: What the Analyst Gets
5. 2.5 Choosing the Component
6. 2.6 Where Automation Stops
7. 2.7 Change Without Regression
8. 2.8 Exercise: Specify a Playbook

Identity and Secrets

Give a security playbook least-privilege identities: one token per API, a reader and a writer, a permission matrix run as tests, secrets out of run history and the repository, and a rotation with no failed run.

1. 3.1 Who the Playbook Is
2. 3.2 Client Credentials, Step by Step
3. 3.3 One Token per API
4. 3.4 Separate Read and Write Identities
5. 3.5 The Permission Matrix as Tests
6. 3.6 Secrets Out of the Definition
7. 3.7 Rotating a Secret
8. 3.8 Exercise: Least Privilege

Phase 2: Evidence, APIs and Code

Querying Evidence

Add query evidence to a security playbook: the Log Analytics query API, saved functions with guarded parameters, a tested window, and a sentence for every way a query can fall short.

1. 4.1 The Evidence Question
2. 4.2 Calling the Query API
3. 4.3 A Saved Function with Parameters
4. 4.4 The Window
5. 4.5 Results That Report Completeness
6. 4.6 Evidence in the Comment
7. 4.7 When the Query Fails
8. 4.8 Exercise: An Evidence Step

Calling the Security APIs

Call the Microsoft security APIs from a playbook: Graph alerts paged to the end, a shared child workflow with one vocabulary of failure, Defender rate limits, and an antivirus scan polled to its end.

1. 5.1 The Alerts Behind the Account
2. 5.2 Paging Graph Results
3. 5.3 A Child Workflow for API Calls
4. 5.4 Throttling at the Rate Limit
5. 5.5 An Asynchronous Machine Action
6. 5.6 Polling with Until
7. 5.7 Predictable Errors
8. 5.8 Exercise: Alert Evidence

Enrichment and Evidence

Enrich the triage comment with the account, device, addresses and threat intelligence through saved workspace functions, audit the watchlists behind them, and keep a traceable evidence record of every fact.

1. 6.1 What Enrichment Is For
2. 6.2 The Account
3. 6.3 The Device
4. 6.4 The Addresses
5. 6.5 Watchlists as Data
6. 6.6 Threat Intelligence
7. 6.7 A Traceable Evidence Record
8. 6.8 Exercise: Enrichment

Code in the Workflow

Put code where the workflow needs it: inline JavaScript, inline PowerShell, and a tested PowerShell Azure Function that reads Microsoft Graph with its own identity and answers the playbook in one contract.

1. 7.1 When the Workflow Needs Code
2. 7.2 Inline JavaScript
3. 7.3 Inline PowerShell
4. 7.4 A PowerShell Azure Function
5. 7.5 Calling Microsoft Graph from the Function
6. 7.6 Calling the Function from the Playbook
7. 7.7 Testing the Function
8. 7.8 Exercise: A Graph Function for the Escalation Check

Phase 3: Approval and Response

Analyst Approval

Ask an analyst before acting: an approval record bound to one device and one action, sent through the HTTP Webhook action with an expiry, a decision that must match its request, and a recheck before anything is done.

1. 8.1 What an Approval Must Bind
2. 8.2 The HTTP Webhook Action
3. 8.3 The Request the Analyst Sees
4. 8.4 Expiry
5. 8.5 Three Answers
6. 8.6 A Decision That Matches Its Request
7. 8.7 Revalidate Before Acting
8. 8.8 Exercise: Approval for Isolating NE-BENNETT-LT

Response and Verification

Act on an approval and verify it: isolate a device once, follow the machine action to its end, verify against the record, release with its own approval, disable an account through Graph, and package both as templates.

1. 9.1 Accepted Is Not Contained
2. 9.2 Isolating the Device
3. 9.3 Following the Machine Action
4. 9.4 Verifying the Outcome
5. 9.5 Release
6. 9.6 Disabling an Account
7. 9.7 The Containment Templates
8. 9.8 Exercise: Contain NE-BENNETT-LT

Surviving Failure

Make the triage playbook safe to run twice: guard each action against earlier work, key comments so reruns update them, close the race between concurrent runs, handle malformed entities, and label a failed containment.

1. 10.1 Why the Same Incident Arrives Twice
2. 10.2 An Idempotency Key
3. 10.3 Comments That Update, Not Multiply
4. 10.4 Two Runs at Once
5. 10.5 A Malformed Entity
6. 10.6 Failed Containment
7. 10.7 Every Fault, Once
8. 10.8 Exercise: Survive a Duplicate Trigger

Phase 4: Deployment and Operations

Deploying to a Tenant

Put the triage playbook into a tenant: managed identities, Bicep infrastructure, the Sentinel trigger, automation rules as code, a pipeline that checks before it deploys, and a rollback that switches the rules off first.

1. 11.1 What Changes in a Tenant
2. 11.2 The Infrastructure in Bicep
3. 11.3 The Sentinel Trigger
4. 11.4 Identities and Roles
5. 11.5 The Automation Rule
6. 11.6 Shipping the Workflows
7. 11.7 Versions and Rollback
8. 11.8 Exercise: Deploy the Triage Playbook

Operating It, and Capstone

Keep the deployed playbook working: the health log, the run log, four alert rules, a cost model, retention and a generated handover; then the capstone, one incident through every stage in seven cases.

1. 12.1 What Operating Means
2. 12.2 Inside the Run
3. 12.3 Alerts That Fire
4. 12.4 What It Costs
5. 12.5 Retention and Evidence
6. 12.6 Handover
7. 12.7 The Capstone Brief
8. 12.8 Capstone: Contain NE-THOMPSON-LT

Phase 0: Course Resources

Cheatsheets

One sheet per subject of the course: the question, the workflow fragment, query or command that answers it, what to read in the result and what it does not prove.

1. Workflow Definitions and Expressions
2. Identity, Tokens and Secrets
3. Querying Evidence and Enrichment
4. Security API Calls and Code
5. Approval and Containment
6. Failure and Idempotency
7. Deployment and Operations
8. Lookup Values

Cookbooks

Seven runbooks, one per stage of building a security playbook: the lab, the specification, identities, evidence, containment, running twice safely, and deployment with its handover.

1. Running the Lab and a First Case
2. Specifying a Playbook and Changing It
3. Identities and Secrets for a Playbook
4. Adding an Evidence or Enrichment Step
5. Adding an Approved Containment Action
6. Making a Playbook Safe to Run Twice
7. Deploying to a Tenant and Handing Over

Lab Setup

The local lab every lesson runs in: what each part plays, what replaces it in a tenant, the settings and faults it can switch on, and how to prove it works and how far back its data reaches.

1. What the Lab Plays
2. Settings, Faults and Activity
3. Verifying the Lab and How Far Back It Reaches

Walkthroughs

The six investigations the course automates, each worked end to end with the wrong turns: a suspicious sign-in, endpoint execution, a reported phish, an indicator, repeated incident updates and a failed containment.

1. A Suspicious Sign-In
2. Suspicious Endpoint Execution
3. A Reported Phish
4. A Malicious Indicator
5. Repeated Incident Updates
6. Failed Containment

Playbooks

The template pack, and one playbook for each thing that fires once the triage playbook is deployed: a rule run that did not start it, a failed run, a failed containment, an incident nothing touched, and a release that did not verify.

1. The Template Pack
2. Triage Not Started
3. Triage Run Failed
4. Triage Containment Failed
5. High Incident Untouched
6. Release Did Not Verify

Practice Playground

The exercise that closes each module, with what each asks you to build, the lab it runs in, and how it is checked.

1. Exercises and How They Are Checked

References and Reading

Every Microsoft Learn page the lessons were built from, grouped by product, each with the lessons that rely on it.

1. Microsoft Learn Pages

Course Completion

Course Exam

Security Automation Engineering end-of-course exam: 29 questions in four parts, from building the workflow to operating it, ending on a case study.

1. Course Completion. Security Automation Engineering

Generated 2026-11-14 from the published course. The current syllabus is always at ridgelinecyber.com/training/courses/security-automation-engineering/