GETU ACADEMY

THE CYBERSECURITY ROADMAP 2027

The exact certs and projects I'd pursue if I was starting all over, broken down step by step.

WHAT'S INSIDE	9 certs in the order I'd take them, what each costs and how to study for it
PROJECTS	5 build-it-yourself projects with step-by-step instructions, including the full project from my BUILD video: SOC 2-ready AWS infrastructure in Terraform, scanned with Prowler
ENDS WITH	How to turn it all into a portfolio, resume bullets and a first-week plan

By Aman Getu, cybersecurity manager · @getuacademy

Prices and exam versions checked November 2026. They change, so confirm on the official page before you pay.

START HERE

The whole roadmap on one page

Certs get you past the HR filter. Projects get you hired. Every stage below pairs one with the other, so by the time you apply you have the cert **and** proof you can do the work.

STAGE	CERT	PROJECT
1. Foundations	Google Cybersecurity Certificate	P1: Home lab
2. Get past HR	Network+, then Security+	P2: SIEM + catch a brute-force attack
3. Prove the job	TryHackMe SAL1	P3: Phishing investigation + incident report
4. Cloud + GRC	(project-led)	P4: AWS + Terraform + SOC 2 scan with Prowler
5. Level up	CySA+	P5: Detection rules mapped to MITRE ATT&CK
Red team branch	eJPT, then OSCP+	Box write-ups
Later	CISM, CISSP	Needs real work experience

How to use this guide

- Go in order. Each stage builds on the one before it.
- Do the project **while** you study the cert, not after. The project is what makes the cert stick.
- Put every project on GitHub as you finish it. The Make It Count page near the end shows you how.
- Don't buy everything at once. Pay for the next cert only when you're ready to sit it.

STAGE 1 · FOUNDATIONS

Google Cybersecurity Certificate

This is where I'd start if I had zero background. It walks you through the language of security, Linux, SQL, networking, SIEM tools and Python in a way a complete beginner can follow. It won't get you hired on its own, and that's fine. Its job is to give you the base so Network+ and Security+ make sense.

COST	US$49 a month on Coursera after a 7-day free trial. The faster you finish, the less you pay.
TIME	Coursera estimates about 6 months at 7 hours a week. Many people go faster.
PREREQS	None
LINK	grow.google/certificates/cybersecurity

How I'd do it

- Block the same time every day, even 45 minutes. Consistency beats weekend marathons.
- Do every lab. Skip the videos you already understand, never the hands-on parts.
- Start Project 1 (home lab) in week one, so you practise the Linux commands on your own machine.
- When you hit the Python course, rewrite each exercise from memory the next day.

You're ready to move on when

- You can explain the CIA triad, what a SIEM does and what a firewall rule is, without notes.
- You can move around a Linux terminal: files, permissions, users, grep.

PROJECT 1 · GOES WITH STAGE 1

Build your home lab

Every other project runs on this lab, so build it first. You'll end up with three virtual machines on your own computer: a Windows machine to defend, an Ubuntu server, and Kali Linux to attack from.

COST	Free
YOU NEED	A computer with at least 16 GB of RAM if you can. 8 GB works for two VMs at a time.
TIME	One weekend

Steps

1. Install VirtualBox (free): www.virtualbox.org.
2. Download three images: a Windows 11 or Windows Server **evaluation** from Microsoft Evaluation Center (free trial), Ubuntu Server from ubuntu.com, and Kali Linux from kali.org (Kali has a ready-made VirtualBox image).
3. Create a **NAT Network** in VirtualBox (Tools, Network) called **lab-net**. Put all three VMs on it. Now they can talk to each other and the internet, but not to the rest of your home network.
4. Give each VM 2 CPUs. Windows 4 GB RAM, Ubuntu 2 GB, Kali 2 to 4 GB.
5. Install everything, then take a **snapshot** of each VM named "clean". If you break something later, you roll back in seconds.
6. Prove they can talk: from Kali, run `ip a` on each box to find its IP, then `ping` the other two.
7. On Ubuntu, install SSH: `sudo apt install openssh-server`. You'll attack it in Project 2.
8. Draw a simple diagram of your lab (draw.io is free): three boxes, their IPs and the network. This goes in your GitHub README.

Show it: a GitHub repo called **home-lab** with the diagram, the specs of each VM and three screenshots. That's your first portfolio piece.

STAGE 2 · GET PAST THE HR FILTER

Network+, then Security+

I put Security+ in **Elite** for a reason. It's the cert that shows up in job postings again and again, and it's what gets your resume past the filter. Network+ comes first because you can't secure a network you don't understand: ports, protocols, subnetting, routing, DNS. Most of the Security+ questions people struggle with are really networking questions.

Network+ (N10-009)

COST	About US$390 for the exam voucher in the US (CompTIA store, 2026)
FORMAT	Up to 90 questions, multiple choice plus performance-based
FOCUS	Subnetting, the OSI model, common ports, routing and switching, wireless, troubleshooting

Security+ (SY0-801)

COST	US$439 for the exam voucher from CompTIA in the US
FORMAT	Max 90 questions in 90 minutes. Pass mark 750 on a 100 to 900 scale.
VERSION	SY0-801 launched November 17, 2026. If you're starting now, study for 801.
LINK	comptia.org: Security+ V8

How I'd study for both

- One video course, one book, and practice questions. That's it. More resources isn't more learning.
- Write out a subnetting table by hand every day for two weeks. It's the single biggest point-scorer on Network+.
- Do practice questions every day, and read the explanation for every one you get wrong. The order I'd follow is Network+, then Security+, then CySA+, and there's a set for each at getuprep.com/paths/cybersecurity with free questions on every step, so you can see where you stand before you pay for anything.
- Book the exam when you're scoring 85% or more on practice tests, on questions you haven't seen before.
- Do Project 2 between the two exams. Seeing real logs makes the Security+ ops domain click.

PROJECT 2 · GOES WITH STAGE 2

Set up a SIEM and catch a brute-force attack

A SOC analyst's day job in miniature: logs come in, an alert fires, you investigate. You'll run Wazuh (a free, open-source SIEM), attack your own Ubuntu server from Kali and catch it.

Cost: free · **Tools:** Wazuh, Hydra (already on Kali) · **Time:** 1 to 2 weekends

Steps

1. Download the Wazuh OVA (a ready-made VM) from the Wazuh docs and import it into VirtualBox on **lab-net**. Check the docs for the RAM it needs. It's the heaviest VM in your lab.
2. Start it, find its IP with `ip a`, and open `https://<wazuh-ip>` in a browser. Log in with the default credentials from the docs, then **change the password straight away**.
3. In the Wazuh dashboard, choose **Deploy new agent**, paste the command it gives you into your Ubuntu VM and start the agent. Repeat for Windows. Both should show as **active**.
4. Make a small wordlist on Kali with 20 or so passwords, and put the Ubuntu user's real password last.
5. Attack SSH from Kali:

```
hydra -l <ubuntu-user> -P list.txt ssh://<ubuntu-ip>
```

6. In Wazuh, open **Threat Hunting** (or Security Events, depending on version). You'll see a burst of SSH authentication failures, then a brute-force alert, then a **successful login**. That last one is the one that matters.
7. Now do it on Windows: try to log in with the wrong password 15 times, then the right one. Find Windows **Event ID 4625** (failed logon) and **4624** (successful logon) in Wazuh.
8. Write it up like an analyst: what happened, when, from which IP, which account, whether it worked, and what you'd do next.
9. Fix it for real: on Ubuntu set `PasswordAuthentication no` in `/etc/ssh/sshd_config`, set up SSH keys, restart SSH and run the attack again. Screenshot that it fails.

Show it: a repo called **siem-brute-force-detection** with your lab diagram, alert screenshots, the timeline of the attack and the before and after of the fix.

STAGE 3 · PROVE YOU CAN DO THE JOB

TryHackMe SAL1

SAL1 (Security Analyst Level 1) is **Legit** because it doesn't just test what you memorised. The exam puts you in a simulated SOC: real alerts, real tools, and you have to triage, escalate and write an incident report. That's the closest thing to job experience you can get before you have a job, and it's exactly what a hiring manager wants to see next to Security+.

COST	US$349, including 3 months of TryHackMe Premium and one free retake (as reported in 2026)
FORMAT	A practical exam in a SOC simulator, plus multiple-choice questions and a graded report
PREP	TryHackMe's SOC Level 1 path
LINK	tryhackme.com: SAL1

How I'd prepare

- Work through the whole SOC Level 1 path. Don't skip the SIEM rooms. You've already used one in Project 2, so they'll go faster.
- Practise writing reports. Every alert you investigate, write five lines: what, when, who, impact, next step. The report is graded, so a weak write-up can fail you even if you found everything.
- Do Project 3 before you book. It's the same skill the exam's report tests.

PROJECT 3 · GOES WITH STAGE 3

Investigate a phishing email end to end

Phishing is the alert you'll handle most in a SOC. This project shows you can take a suspicious email from "reported by a user" to "closed with a report".

COST	Free
TOOLS	MXToolbox header analyzer, urlscan.io, VirusTotal

Steps

1. Get a sample. Open a real phishing email from your own spam folder (don't click anything), or use a sample from a TryHackMe phishing room.
2. Save the full headers (in Gmail: the three dots, then **Show original**). Paste them into the MXToolbox header analyzer.
3. Check the basics: does the **From** domain match the **Return-Path**? Did **SPF**, **DKIM** and **DMARC** pass or fail? Which IP actually sent it?
4. Pull out the indicators (IOCs): sender address, sending IP, every URL, any attachment's file hash.
5. Check each one **without opening it**: put the URLs into urlscan.io and the hash and IP into VirusTotal. Screenshot what they say.
6. Decide: malicious, suspicious or safe. Write down why in one sentence.
7. Write the incident report using the template below. Keep it to one page.

Incident report template

```
TITLE: Phishing - fake Microsoft login
SEVERITY: Medium STATUS: Closed
SUMMARY: 2-3 sentences, plain English
TIMELINE: received / reported / analysed
IOCs: sender, IP, URLs, hashes
ANALYSIS: SPF/DKIM/DMARC, URL + hash results
IMPACT: did anyone click? any logins?
ACTIONS: block sender + domain, purge from
          inboxes, reset creds if clicked
LESSONS: what would stop the next one
```

Show it: a repo called **phishing-investigations** with three reports. Three is a pattern. One is a one-off.

STAGE 4 · CLOUD + GRC · THE BIG ONE

Project 4: SOC 2-ready AWS infrastructure, built with Terraform

This is the project from my BUILD video, step by step. If you only do one project from this guide, make it this one. Almost every company runs in the cloud, and almost every company that sells software gets asked for a SOC 2 report. This project proves three things in one repo: you can **build** cloud infrastructure, you can build it **as code**, and you can **prove it's compliant** the way a GRC team does.

STEP	WHAT YOU DO
01 AWS account	Free plan account, locked down properly before you build anything
02 Terraform	Write the VPC, EC2 app server, S3 logs bucket and IAM role as code. One command builds it.
03 Compliance scan	Scan it with Prowler (free, open source) against SOC 2. Fix the gaps in code. Scan again.
04 GitHub + resume	Push it all to a repo called aws-soc2-terraform, with the SOC 2 report, and put it on your resume

COST	New AWS accounts on the Free plan get US$100 in credits at sign-up and can earn up to US$100 more. The Free plan lasts up to 6 months. Destroy everything when you're done.
TOOLS	AWS, AWS CLI, Terraform, Prowler, GitHub. All free.
TIME	Two to three weekends

Get this right in interviews: no tool makes you "SOC 2 compliant". SOC 2 is a report an auditor writes after testing a company's controls. Companies pay for platforms like Vanta to check their controls continuously and collect the evidence. Prowler is the free, open-source way to run the same kind of check: it tests your AWS setup against its SOC 2 framework and shows you the gaps. Say "SOC 2-ready", not "SOC 2 compliant".

PROJECT 4 · STEP 01

Set up the AWS account safely

1. Create a new AWS account at aws.amazon.com/free and choose the **Free plan**.
2. Turn on **MFA for the root user** straight away. Then stop using root. (Prowler checks this. It's one of the first things an auditor looks at.)
3. Go to **Billing, Budgets** and create a budget of US$5 with an email alert. Do this before you build anything.
4. Create an admin user for yourself (IAM Identity Center, or an IAM user with MFA). Terraform will run as this user.
5. Create a second, **read-only** identity just for scanning, called **prowler-audit**. Attach the AWS managed policies **SecurityAudit** and **ViewOnlyAccess**. An auditor never needs write access, and that's a point worth making in an interview.
6. On your computer, install the AWS CLI and Terraform, then set up two profiles:

```
aws configure --profile builder
aws configure --profile prowler-audit
terraform -version
```

Never paste access keys into your Terraform files or commit them to GitHub. They live in your AWS CLI profile and nowhere else.

Create the repo

Make a folder called **aws-soc2-terraform** and start it as a Git repo. By the end it will look like this:

```
aws-soc2-terraform/
  main.tf the infrastructure
  variables.tf your region, IP, names
  compliance.tf the SOC 2 fixes (step 03)
  modules/ optional: reusable pieces
  soc2-report.html Prowler's report
  docs/ control mapping, risk
                    register, policy, summary
  README.md
  .gitignore
```

Create **.gitignore** first, so you never push secrets or state by accident:

```
.terraform/
*.tfstate
*.tfstate.*
*.tfvars
```

PROJECT 4 · STEP 02

Build the infrastructure in Terraform

You're writing a tiny version of what a startup runs: a network, an app server, a bucket for logs and an IAM role. Writing it as code (IaC) means anyone can rebuild it with one command, and every change is reviewable in Git. That's how real cloud teams work.

variables.tf

```
variable "region" { default = "us-east-1" }
variable "prefix" { default = "acme-yourname" }
variable "my_ip" { description = "your IP/32" }
```

main.tf: provider and network

```
terraform {
  required_providers {
    aws = { source = "hashicorp/aws"
            version = "~> 5.0" }
  }
}
provider "aws" {
  region = var.region
  profile = "builder"
}

resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  tags = { Name = "${var.prefix}-vpc" }
}
resource "aws_subnet" "public" {
  vpc_id = aws_vpc.main.id
  cidr_block = "10.0.1.0/24"
  map_public_ip_on_launch = true
}
resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main.id
}
resource "aws_route_table" "public" {
  vpc_id = aws_vpc.main.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
}
resource "aws_route_table_association" "public" {
  subnet_id = aws_subnet.public.id
  route_table_id = aws_route_table.public.id
}
```

PROJECT 4 · STEP 02 (CONTINUED)

main.tf: app server, logs bucket, IAM role

Two gaps are left in on purpose: SSH is open to the whole internet, and the bucket has none of the protections SOC 2 expects. Prowler will catch both.

```
resource "aws_security_group" "app" {
  vpc_id = aws_vpc.main.id
  ingress { # GAP: open to world
    from_port = 22
    to_port = 22
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port = 0
    to_port = 0
    protocol = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

data "aws_ami" "al2023" {
  most_recent = true
  owners = ["amazon"]
  filter {
    name = "name"
    values = ["al2023-ami-2023*-x86_64"]
  }
}
resource "aws_instance" "app" {
  ami = data.aws_ami.al2023.id
  instance_type = "t3.micro"
  subnet_id = aws_subnet.public.id
  vpc_security_group_ids = [aws_security_group.app.id]
  iam_instance_profile = aws_iam_instance_profile.app.name
}

resource "aws_s3_bucket" "logs" { # GAP: no
  bucket = "${var.prefix}-logs" # protections
  force_destroy = true
}

resource "aws_iam_role" "app" {
  name = "${var.prefix}-app-role"
  assume_role_policy = jsonencode({
    Version = "2012-10-17"
    Statement = [{ Effect = "Allow",
      Action = "sts:AssumeRole",
      Principal = { Service = "ec2.amazonaws.com" } }]
  })
}
resource "aws_iam_instance_profile" "app" {
  role = aws_iam_role.app.name
}
```

Use a Free Tier eligible instance type. t3.micro is a common one, but check the label in the EC2 console for your account and region. Bucket names are global, so make your prefix unique.

PROJECT 4 · STEP 02 (CONTINUED)

Build it

```
terraform init
terraform plan -var my_ip=YOUR.IP.ADDRESS/32
terraform apply -var my_ip=YOUR.IP.ADDRESS/32
```

- **plan** shows you exactly what will be created before anything happens. Read it every time. In a real job, this output is what gets reviewed before a change goes live.
- **apply** builds it. Type **yes** when it asks.
- Open the AWS console and find your VPC, instance, bucket and role. Screenshot them.
- Commit: `git add . && git commit -m "baseline infra"`

Level up: modules

Once it works, move the network into **modules/network** and the app server into **modules/app**, then call them from main.tf. That's the "Terraform module" part of the video, and it shows you can write reusable infrastructure, not one-off scripts.

PROJECT 4 · STEP 03

Run the baseline SOC 2 scan

1. Install Prowler. It needs Python 3.10, 3.11 or 3.12 (it doesn't support 3.13 yet):

```
python3 -m venv prowler-env
source prowler-env/bin/activate
pip install prowler
prowler -v
```

2. Scan with the **read-only** profile against SOC 2, saving to a baseline folder:

```
prowler aws --profile prowler-audit \
  --compliance soc2_aws \
  -M csv html -o ./baseline
```

3. Open the HTML report. The compliance folder inside **baseline** has a CSV listing each SOC 2 requirement and the checks under it. Prowler's soc2_aws framework has 23 requirements and 134 checks.
4. Write down your starting numbers: checks passed, checks failed, and how many failures are critical or high. Screenshot the summary.
5. Find your two planted gaps, then the ones you didn't plant (no audit logging, for example). There will be some. That's great interview material.

PROJECT 4 · STEP 03 (CONTINUED)

Fix the gaps in code

Don't click fixes in the console. Fix them in Terraform, so the fix is reviewable, repeatable and permanent. These are the four checks from the video, plus the rest:

CHECK	FIX	SOC 2 AREA
Audit logging on	CloudTrail trail for all regions with log file validation (code on the next page)	CC7.2 monitoring
No public buckets	S3 Block Public Access on the logs bucket	CC6.1 logical access
Encryption at rest	Default encryption on the bucket. Encrypted root volume on the instance.	CC6.1 / CC6.7 data protection
MFA on root	Turn on root MFA in the console (Terraform can't). You did this in step 01.	CC6.1 logical access
SSH open to world	Change the ingress rule to `[var.my_ip]`, or use SSM Session Manager and remove port 22	CC6.6 boundary protection
Versioning	Turn on bucket versioning so deleted logs can be recovered	CC6.1 / A1.2
IMDSv2	Require IMDSv2 on the instance (stops a classic credential-theft path)	CC6.1
Password policy	Account password policy: length, complexity, no reuse	CC6.1

The SOC 2 areas are Trust Services Criteria codes. Use the exact requirement IDs from your own Prowler compliance CSV in your write-up.

Change these in main.tf

```
# in aws_security_group.app, ingress:
    cidr_blocks = [var.my_ip]

# in aws_instance.app, add:
  metadata_options {
    http_tokens = "required" # IMDSv2
  }
  root_block_device {
    encrypted = true
  }
```

PROJECT 4 · STEP 03 (CONTINUED)

compliance.tf

```
resource "aws_s3_bucket_public_access_block" "logs" {
  bucket = aws_s3_bucket.logs.id
  block_public_acls = true
  block_public_policy = true
  ignore_public_acls = true
  restrict_public_buckets = true
}
resource "aws_s3_bucket_versioning" "logs" {
  bucket = aws_s3_bucket.logs.id
  versioning_configuration { status = "Enabled" }
}
resource "aws_s3_bucket_server_side_encryption_configuration" "logs" {
  bucket = aws_s3_bucket.logs.id
  rule {
    apply_server_side_encryption_by_default {
      sse_algorithm = "AES256"
    }
  }
}
resource "aws_iam_account_password_policy" "strict" {
  minimum_password_length = 14
  require_uppercase_characters = true
  require_lowercase_characters = true
  require_numbers = true
  require_symbols = true
  password_reuse_prevention = 24
}

# Audit logging: CloudTrail writes to its own bucket
data "aws_caller_identity" "me" {}
locals { acct = data.aws_caller_identity.me.account_id }
resource "aws_s3_bucket" "trail" {
  bucket = "${var.prefix}-trail"
  force_destroy = true
}
resource "aws_s3_bucket_policy" "trail" {
  bucket = aws_s3_bucket.trail.id
  policy = jsonencode({
    Version = "2012-10-17"
    Statement = [
      { Effect = "Allow", Action = "s3:GetBucketAcl",
        Principal = { Service = "cloudtrail.amazonaws.com" },
        Resource = aws_s3_bucket.trail.arn },
      { Effect = "Allow", Action = "s3:PutObject",
        Principal = { Service = "cloudtrail.amazonaws.com" },
        Resource = "${aws_s3_bucket.trail.arn}/AWSLogs/${local.acct}/*",
        Condition = { StringEquals = {
          "s3:x-amz-acl" = "bucket-owner-full-control" } } }
    ]
  })
}
resource "aws_cloudtrail" "main" {
  name = "${var.prefix}-trail"
  s3_bucket_name = aws_s3_bucket.trail.id
  is_multi_region_trail = true
  include_global_service_events = true
  enable_log_file_validation = true
  depends_on = [aws_s3_bucket_policy.trail]
}
```

PROJECT 4 · STEP 03 (CONTINUED)

Apply the fixes and scan again

```
terraform apply -var my_ip=YOUR.IP.ADDRESS/32

prowler aws --profile prowler-audit \
  --compliance soc2_aws \
  -M csv html -o ./after
```

- Put the before and after numbers side by side. "Passing checks went from X to Y, critical findings from A to zero" is the line that goes on your resume.
- Copy the final HTML report into the repo root as **soc2-report.html**. Open it first and remove your AWS account ID.
- Commit: `git commit -am "SOC 2 fixes"`. Now your Git history shows the before, the fix and the after. That's an audit trail in itself.
- Want a visual? `prowler dashboard` opens Prowler's local dashboard in your browser.

Document it like a GRC analyst

This is what separates you from everyone else who "ran a scanner". Put these four files in **docs/**:

- **control-mapping.md:** one row per fix: the check, what you changed (link the commit), the SOC 2 criterion, and the evidence (the Prowler check that now passes).
- **risk-register.md:** one row for each thing you **didn't** fix: the risk, likelihood, impact, owner and decision (fix, accept, transfer). For example, "No AWS Config recording. Medium. Accepted for the lab because of cost. Fix in production." Accepting a risk with a written reason is a real GRC decision.
- **access-control-policy.md:** one page in plain English: who gets AWS access, how it's approved, MFA for everyone, least privilege, quarterly access reviews, what happens when someone leaves. Your Terraform should match what the policy says. Auditors check that.
- **executive-summary.md:** half a page for a manager who isn't technical: what you assessed, the top three risks, what you fixed, before and after numbers, what you'd do next.

Level up: continuous compliance

Add a GitHub Action that runs `terraform plan` on every pull request and Prowler once a week. That's automated compliance, the same thing companies pay platforms like Vanta for, and you built it for free.

PROJECT 4 · STEP 04

GitHub + resume

1. Write the README (template on the Make It Count page): the problem, an architecture diagram (VPC, subnet, EC2, S3, IAM, CloudTrail), how to run it, and before and after numbers.
2. Check one last time that no **.tfstate**, **.tfvars** or keys are in the repo. Then push:

```
git remote add origin https://github.com/YOU/aws-soc2-terraform.git
git push -u origin main
```

3. Tear it down so it stops using your credits:

```
terraform destroy -var my_ip=YOUR.IP.ADDRESS/32
```

4. Open **Billing** and confirm nothing is still running. Leave the budget alert on.

On your resume

SOC 2-ready AWS infrastructure · Terraform, AWS, Prowler, GitHub
Built AWS infrastructure (VPC, EC2, S3, IAM, CloudTrail) as code in Terraform and audited it against SOC 2 with Prowler; fixed X findings in code, raising passing checks from A to B and clearing all critical findings; documented the control mapping, risk register and access control policy.

This is what puts you ahead of someone who only has Security+. Same cert, but you have proof you can build, secure and audit real cloud infrastructure.

In the interview, walk through one finding from start to finish: what Prowler flagged, why it matters for SOC 2, the Terraform change that fixed it, and the rescan that proved it. That one story beats listing ten tools.

STAGE 5 · LEVEL UP AS AN ANALYST

CySA+ (CS0-004)

CySA+ is **Elite** on my list because it's the step from "I know the concepts" to "I can run detection and response": SIEM and EDR, threat intel, vulnerability management, incident response and reporting. The new version also covers cloud and hybrid environments, so Project 4 sets you up well for it.

COST	US$425 for the exam voucher
FORMAT	Max 85 questions, multiple choice and performance-based. Pass mark 750.
VERSION	CS0-004 (V4) launched June 23, 2026
WHEN	CompTIA recommends about four years of hands-on experience. I'd take it once you're in your first SOC role, or after SAL1 plus Projects 2 to 5.
LINK	comptia.org: CySA+ V4

PROJECT 5 · GOES WITH STAGE 5

Write detection rules and test them

Analysts respond to alerts. Strong analysts write the alerts. This project shows you can do both.

1. Pick three attack techniques from MITRE ATT&CK: for example, brute force (T1110), a new local admin account (T1136) and scheduled-task persistence (T1053).
2. For each one, write down what it would look like in the logs (which Windows Event IDs, which Linux log lines).
3. Write a detection rule for each: as a custom Wazuh rule, or in Sigma format, which converts to most SIEMs.
4. Test it: run the technique in your lab (by hand, or with Atomic Red Team tests) and confirm the alert fires in Wazuh.
5. Tune it: run normal activity and make sure it **doesn't** fire. Write down any false positives and how you fixed them.

Show it: a repo called **detection-rules**. One folder per rule: the rule file, the ATT&CK ID, the test you ran, a screenshot of the alert and your tuning notes.

OPTIONAL BRANCH

Red team: eJPT, then OSCP+

If breaking in excites you more than defending, this is your branch. Do it after Security+, alongside or instead of Stages 3 and 5.

eJPT (Legit)

COST	US$249, which includes 3 months of INE's Fundamentals training
FORMAT	A 48-hour, fully hands-on lab exam
WHY	The best first hands-on pentest cert. It proves you can actually run the tools, not just name them.
LINK	ine.com: eJPT

OSCP+ (Elite)

COST	PEN-200 course and exam bundle from US$1,749 (90 days of labs, one attempt), or Learn One at US$2,749 a year (one year, two attempts). OffSec's Black Friday deal had Learn One at US$2,199 until January 4, 2027.
FORMAT	A hands-on exam where you have to compromise machines and write a professional report
NOTE	OSCP+ expires after 3 years. The original OSCP title is for life.
WHY	The pentest cert hiring managers respect most. Don't rush it. Go in with eJPT and plenty of box practice behind you.
LINK	offsec.com: PEN-200 / OSCP+

Project: box write-ups

Root 10 machines on TryHackMe or Hack The Box, and write each one up: how you found the way in, how you escalated, and how a defender would have stopped you. That last part makes your write-ups stand out from everyone else's.

LATER, WITH EXPERIENCE

CISM and CISSP

These are the leadership certs. You need real work experience to hold them, so they aren't where you start. They're where you're going. I'm including them so you can see the whole road.

CISM (Legit)

COST	US$760, or US$575 for ISACA members
EXPERIENCE	5 years in information security, including 3 years in security management. You can pass the exam first and then have 5 years to apply.
FOR	People moving into security management and GRC leadership
LINK	isaca.org: CISM

CISSP (Elite)

COST	US$749 exam fee in the Americas
EXPERIENCE	5 years of full-time paid work in at least two of the eight CISSP domains. A degree or an approved cert can cover one year.
EARLY?	You can pass the exam before you have the experience and become an **Associate of ISC2**, then earn the experience within six years.
LINK	isc2.org: CISSP

Tip: start tracking your experience now. Keep a simple log of projects, incidents you've handled and what you owned. When you apply for CISM or CISSP years from now, you'll have it all written down.

MAKE IT COUNT

Turn this into interviews

Five projects on GitHub plus Security+ beats a stack of certs with nothing to show. Here's how to package it.

Every README, same structure

```
# Project name
## The problem (1-2 lines, business terms)
## What I built (diagram + tools)
## What I found (screenshots, numbers)
## What I fixed (before / after)
## What I'd do next
## How to reproduce
```

Resume

- Put a **Projects** section above Education. Each project gets one line that starts with a verb and ends with a result.
- Use the words from the job posting: SIEM, incident response, SOC 2, AWS, IAM, vulnerability management.
- Link your GitHub at the top, next to your email.

LinkedIn

- Post each project when you finish it: one screenshot, three lines on what you learned. Recruiters search for the tools you name.
- Headline: **Aspiring SOC Analyst | Security+ | AWS + SOC 2 projects**, updated as you add certs.

In the interview

Pick one project and tell it as a story: the problem, what you did, what went wrong, how you fixed it and the result. Project 4 is usually the best one to tell.

START TODAY

Your first week

DAY	DO THIS
1	Install VirtualBox and download Kali, Ubuntu Server and a Windows evaluation image
2	Build the Kali VM and the NAT network. Take a snapshot.
3	Start the Google Cybersecurity Certificate free trial and finish the first module
4	Build the Ubuntu and Windows VMs and get all three pinging each other
5	Make a GitHub account and create the **home-lab** repo with your diagram
6	One more module of the Google cert, and practise the Linux commands in your lab
7	Rest, or read Project 2 so you know what's coming

You don't need to be ready to start. You need to start to get ready. Do day 1 today.

If this helped, please consider following @getuacademy. I post a new roadmap, project or cert breakdown every week.