National College of Ireland

Configuration Manual

MSc Research Project
Master of Science in Cyber Security

Ajay Mohan
Student ID: 23389796

School of Computing
National College of Ireland

Supervisor: Khadija Hafeez

National College of Ireland
MSc Project Submission Sheet
School of Computing

Student Name:	Ajay Mohan	
Student ID:	23389796	
Programme:	Master of Science in Cyber Security	**Year:** 2026
Module:	MSc (Research) Practicum Part - 2	
Lecturer:	Khadija Hafeez	
Submission Due Date:	28/01/2026	
Project Title:	Integrating Zero Trust Security with Network Penetration testing in Multi-cloud Environments	
Word Count:	1634	
Page Count:	13	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Ajay Mohan

Date: 27/01/2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Ajay Mohan
Student ID: 23389796

Introduction

This guide gives you thorough, easy-to-follow directions on how to create the required cloud environments (AWS and Azure) and use the 'Integrating Zero Trust Security with Network Penetration Testing' method. By using these directions, you can do a full, repeatable run of your entire security process, from setting up your cloud environments to analysing your results.

1 Section 1: System Prerequisites

Before beginning, ensure the following software is installed on your local machine (e.g., laptop or workstation) where you will run the framework.

1. **Git:** For cloning the project repository.
2. **Python:** Version 3.9 or higher is required.
3. **Python venv module:** Typically included with Python installations. Used for creating isolated Python environments.
4. **AWS CLI:** (Optional but recommended) For verifying AWS credentials and settings.
5. **Azure CLI: Required** for authenticating with your Azure account.
6. **SSH Client:** For generating SSH keys. Standard on Linux/macOS, available on Windows (e.g., via Git Bash or OpenSSH).
7. **A Code Editor:** Such as Visual Studio Code, for editing configuration files.

2 Section 2: Project and Environment Setup

This section covers cloning the project and setting up the Python environment.

2.1 Step 2.1: Clone the Project Repository

Open a terminal or command prompt and run/reach to the folder.

Navigate into the project directory
cd project name
Explanation: This command downloads the entire project structure, including all Python scripts and configuration templates, to your local machine.

2.2 Step 2.2: Create and Activate a Python Virtual Environment

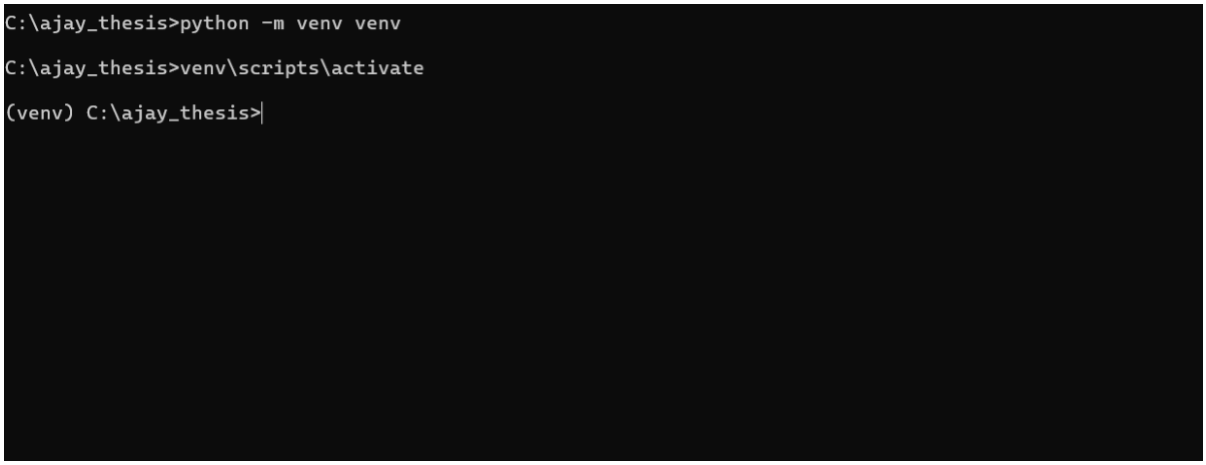
It is a best practice to use a virtual environment to manage project-specific dependencies.

Create a virtual environment named 'venv'

python -m venv

Activate the virtual environment

On Windows:



.\venv\Scripts\activate

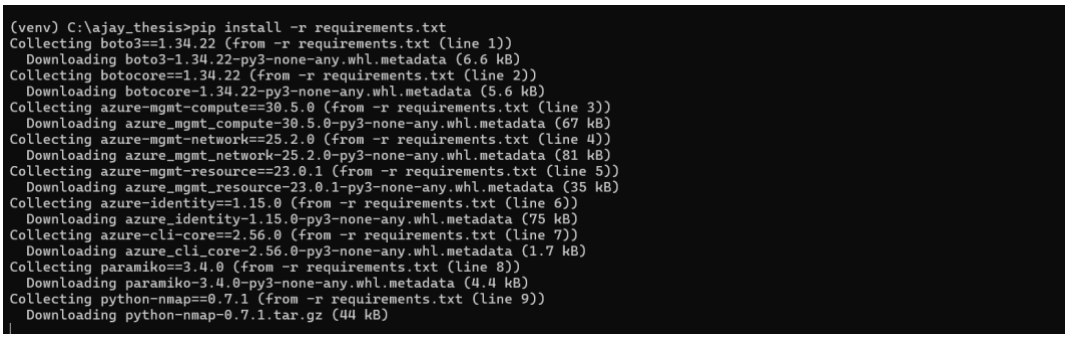
On macOS/Linux:

source venv/bin/activate

Explanation: This creates an isolated Python environment. When activated, your terminal prompt will typically change to show (venv), indicating that any packages installed will be local to this project and will not interfere with your global Python installation.

2.3 Step 2.3: Install Required Python Packages

With the virtual environment active, install all the necessary libraries.



Install all packages listed in the requirements.txt file

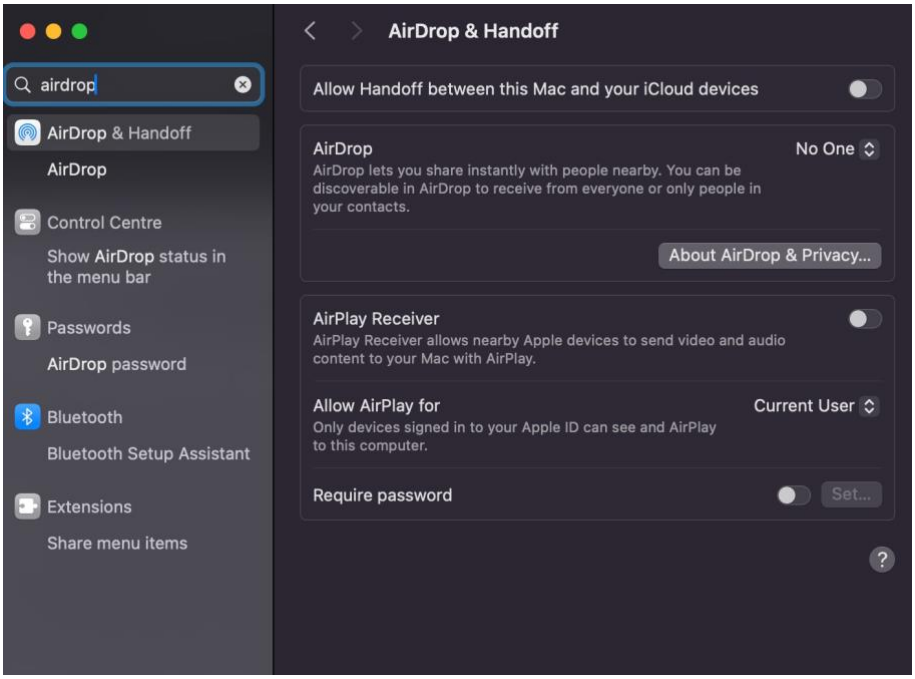
pip install -r requirements.txt

Explanation: This command reads the requirements.txt file and automatically downloads and installs all the specified Python libraries, such as boto3 (for AWS), azure-sdk-for-python, pandas, and matplotlib.

2.4 Disabling AirDrop and Handoff for Mac Os Users

Before doing the further steps, Mac users should disable the AirDrop and Handoff.

1. Go to system preferences in settings and search for AirDrop & Handoff and disable AirPlay Receiver and for confirmation enter the system password.

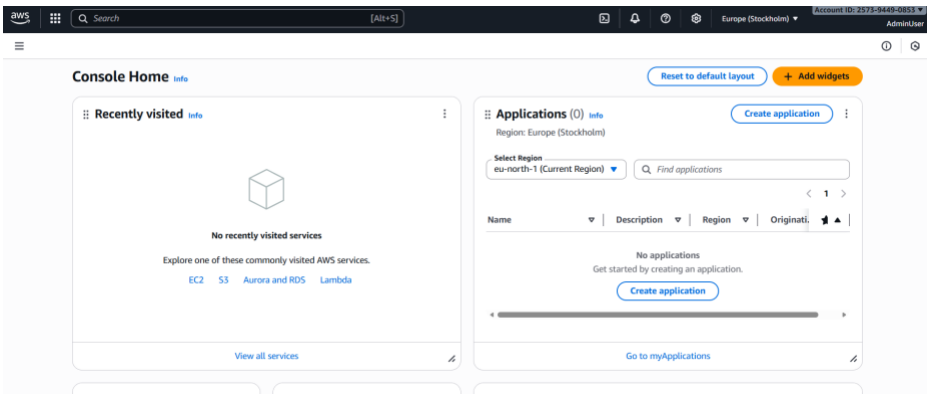


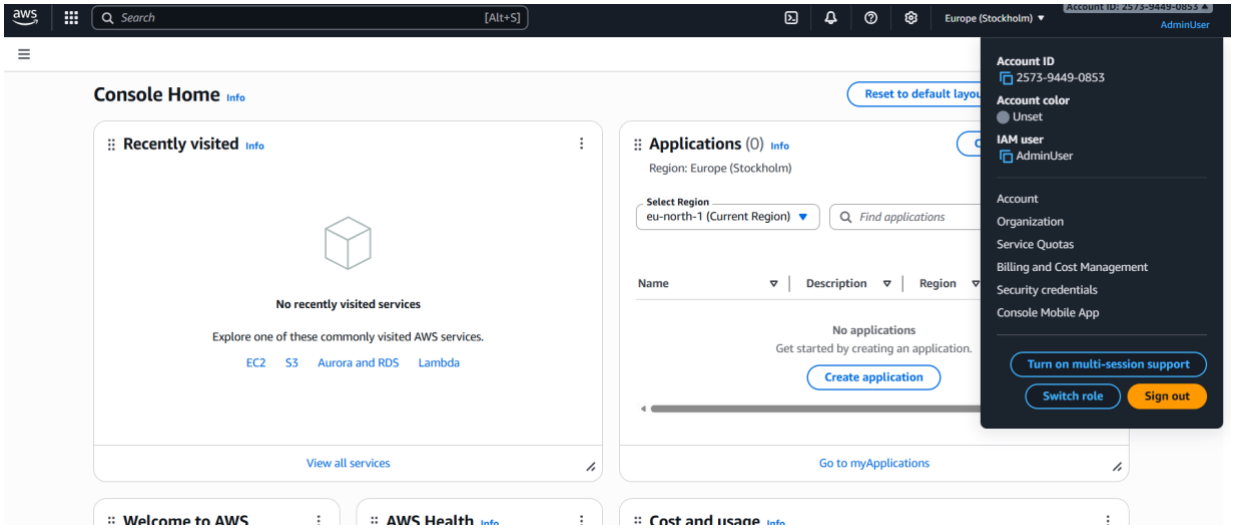
3 Section 3: AWS Account Configuration

This section details the steps to prepare your AWS account.

3.1 Step 3.1: Create an IAM User

For security, do not use your root AWS account. Create a dedicated IAM user.





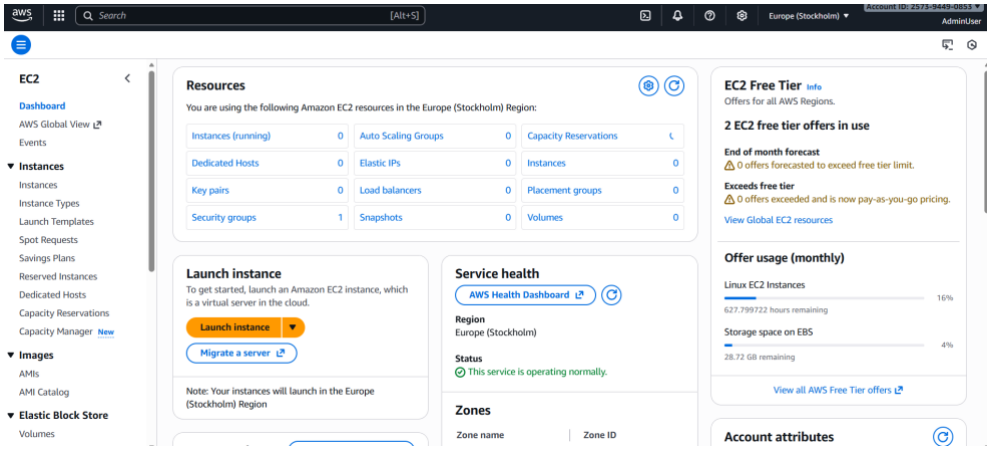
1. Navigate to the **IAM console** in your AWS account.
2. Go to **Users** and click **"Add users"**.
3. Enter a **Username**, for this project use AdminUser.
4. Select **"Access key - Programmatic access"** for the credential type.
5. Click **"Next: Permissions"**.
6. Select **"Attach existing policies directly"**.
7. Search for and select the AdministratorAccess policy. **Note:** For this academic project, AdministratorAccess is used for simplicity. In a production environment, you must create a custom policy with the minimum required permissions (Principle of Least Privilege).
8. Proceed through the tags section and click **"Create user"**.

3.2 Step 3.2: Obtain AWS Access Keys

On the final screen after creating the user, AWS will display the **Access key ID** and the **Secret access key**.

- **This is the only time the Secret access key will be shown.**
- Copy both values immediately and store them securely. They will be needed for the aws_config.json file.

3.3 Step 3.3: Create an EC2 Key Pair



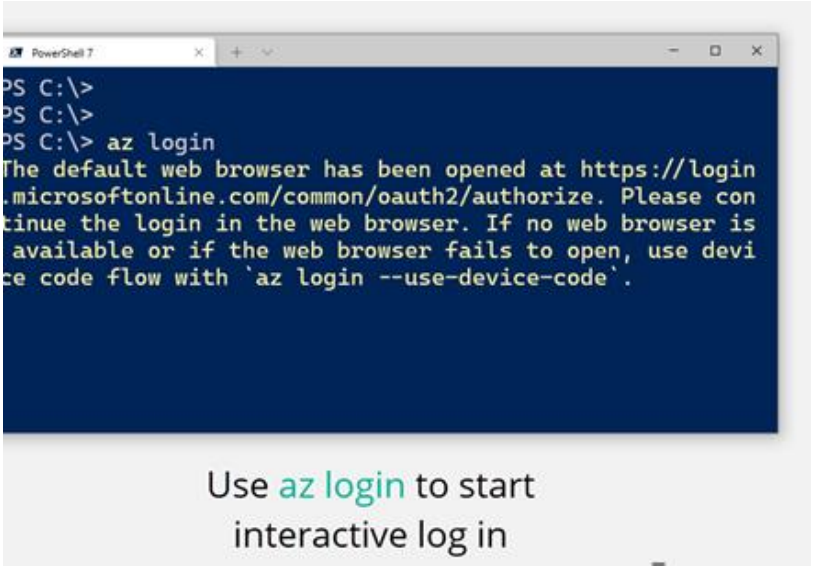
This key pair is required to allow the framework to interact with EC2 instances.

1. Navigate to the **EC2 console** in your AWS account.
2. Ensure you are in the correct region: **us-east-1 (N. Virginia)**.
3. In the left navigation pane, under **Network & Security**, click **"Key Pairs"**.
4. Click **"Create key pair"**.
5. Enter a **Name**: zero-trust-key.
6. Keep the defaults (**RSA** and **.pem**).
7. Click **"Create key pair"**. The file zero-trust-key.pem will be downloaded.
8. Move this .pem file into the root directory of the project.

4 Section 4: Azure Account Configuration

This section details the steps to prepare your Azure account.

4.1 Step 4.1: Install and Log in with Azure CLI



The framework uses the Azure CLI for authentication.

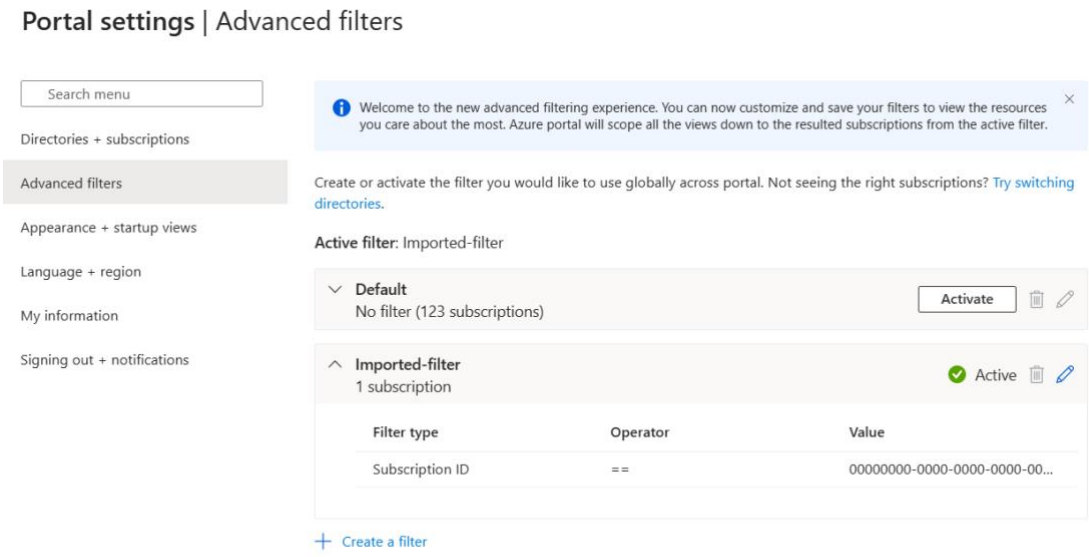
1. If not already installed, How to install the Azure CLI | Microsoft Learn.

2. Open a terminal and log in to your Azure account:

az login

Explanation: This command will open a web browser, prompting you to sign in to your Microsoft Azure account. After successful authentication, your terminal will display your subscription information.

4.2 Step 4.2: Set the Active Subscription



If you have multiple subscriptions, ensure the correct one is active.

List all available subscriptions

az account list --output table
Set the active subscription using its ID

az account set --subscription "211b33f0-03b0-4e97-aa5b-2093538142e8"

Explanation: This ensures that all subsequent commands and script executions target the correct Azure subscription where resources will be created and managed.

4.3 Step 4.3: Generate an SSH Key Pair

```
The key fingerprint is:
SHA256:GnWfYWRZuAUFkarFY9pN3AW33PgBThX+Yaiuer2KIME zero-trust-multicloud
The key's randomart image is:
+---[RSA 4096]----+
|           *%=+o|
|          o* *o+|
|        ...oo*.O.|
|  .    . .*o=oo.+|
|   E  . S* =o   o|
|    .  oo o .    |
|   . ..   ..     |
|    . . ....     |
|       ooo...    |
+----[SHA256]-----+
```

The scripts will use this key to create the Azure VM with secure access.

1. If you do not have an SSH key, generate one:

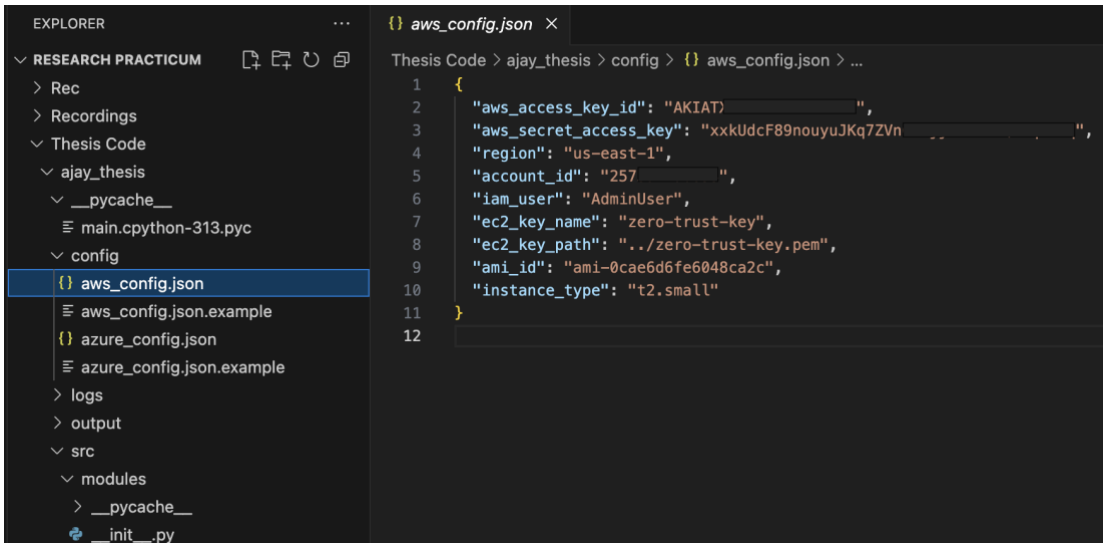
ssh-keygen -t rsa -b 2048 -f ~/.ssh/id_rsa

Explanation: This command, command creates a new SSH key pair in the users .ssh directory. When prompted for a password, press Enter to leave it blank for project, project purposes. The VM creation script automatically finds and uses the public key (~/.ssh/id_rsa.pub).

5 Section 5: Application Configuration

The framework is controlled by two JSON configuration files located in the config/ directory.

5.1 Step 5.1: Configure aws_config.json



Note: Hidden few fields for security reasons in the above figure.

Create a file named aws_config.json inside the config/ directory and populate it with the following content, replacing the placeholder values with your own from the previous steps.

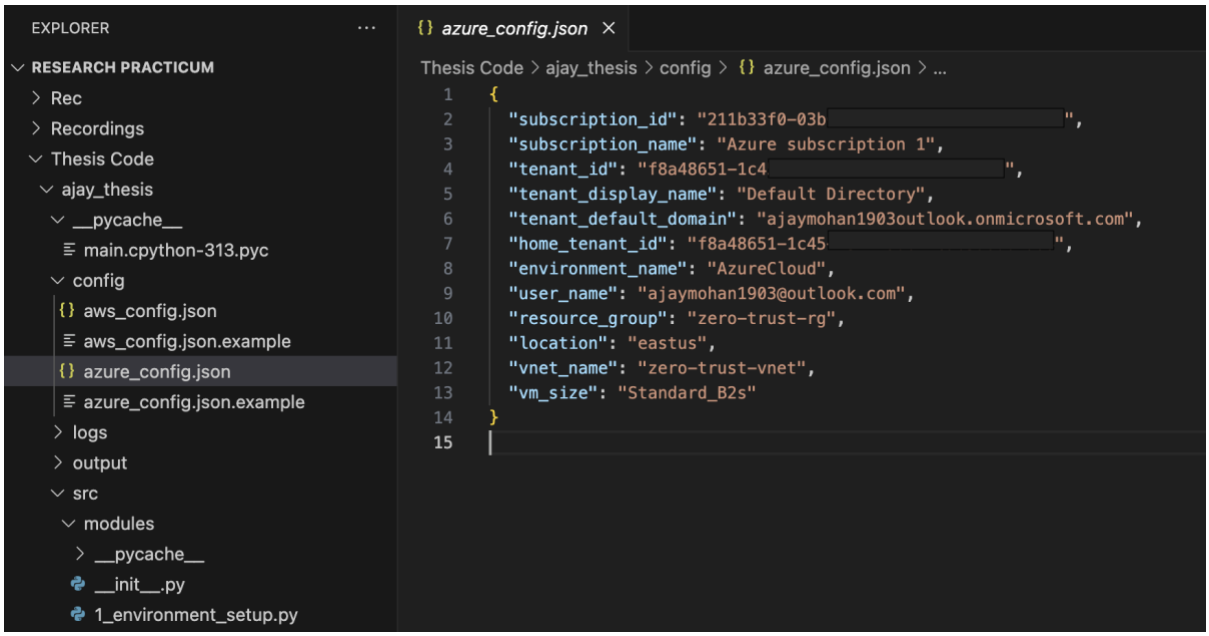
```
{
 "aws_access_key_id": "YOUR_AWS_ACCESS_KEY_ID",
 "aws_secret_access_key": "YOUR_AWS_SECRET_ACCESS_KEY",
 "region": "us-east-1",
 "account_id": "257394490853",
 "iam_user": "AdminUser",
 "ec2_key_name": "zero-trust-key",
 "ec2_key_path": "zero-trust-key.pem",
 "ami_id": "ami-0cae6d6fe6048ca2c",
 "instance_type": "t2.small"
}
```

Field Explanations:

- aws_access_key_id: Your IAM user's access key (from Step 3.2).

- aws_secret_access_key: Your IAM user's secret key (from Step 3.2).
- region: The AWS region for deployment (us-east-1).
- account_id: Your 12-digit AWS Account ID.
- ec2_key_name: The name of the EC2 key pair created in AWS (zero-trust-key).
- ec2_key_path: The relative path to the downloaded .pem file.
- ami_id: The ID of the Amazon Machine Image to use for the EC2 instance.
- instance_type: The size of the EC2 instance to launch.

5.2 Step 5.2: Configure azure_config.json



Note: Hidden the subscription_id, tenant _id and home_tenant_id for security reasons in the above figure.

Create a file named azure_config.json inside the config/ directory and populate it. You can get most of these values from the output of the az login command.

```
{
 "subscription_id": "211b33f0-03b0-4e97-aa5b-2093538142e8",
 "tenant_id": "YOUR_AZURE_TENANT_ID",
 "resource_group": "zero-trust-rg",
 "location": "eastus",
 "vnet_name": "zero-trust-vnet",
 "vm_size": "Standard_B1ms"
}
```

Field Explanations:

- subscription_id: Your Azure Subscription ID.
- tenant_id: Your Azure Tenant ID (shown in az login output).
- resource_group: The name for the resource group that will hold all Azure assets.
- location: The Azure region for deployment (eastus).
- vnet_name: The name for the virtual network to be created.
- vm_size: The size of the Azure VM to launch.

6 Section 6: Workflow Execution

Execute the scripts in the following order from the root directory of the project, ensuring your Python virtual environment is active.

6.1 Step 6.1: Run Prerequisite Scripts (Optional but Recommended)

These scripts prepare the environment for the main test.

1. **Create the Azure VM:**

python scripts/create_azure_vm.py

Explanation: This script uses the settings in azure_config.json and your active Azure CLI session to provision all the necessary Azure resources, including the resource group, VNet, NSG, public IP, and the virtual machine itself.

2. **Introduce Simulated Vulnerabilities:**

- python scripts/add_azure_vulnerabilities.py

Explanation: This script deliberately opens several high-risk ports on the Azure VM's Network Security Group. **This is a crucial step for the experiment**, as it simulates a security misconfiguration that the penetration test is designed to find.

6.2 Step 6.2: Run the Main Orchestrator

This command executes the entire 7-module workflow.

- python main.py

Explanation: This is the main entry point. The script runs sequentially through each module: environment setup, device , device discovery, basic security, penetration testing, zero , zero trust validation, monitoring, and finally analyzing the results. You will , will see detailed log output in the console for each stage.

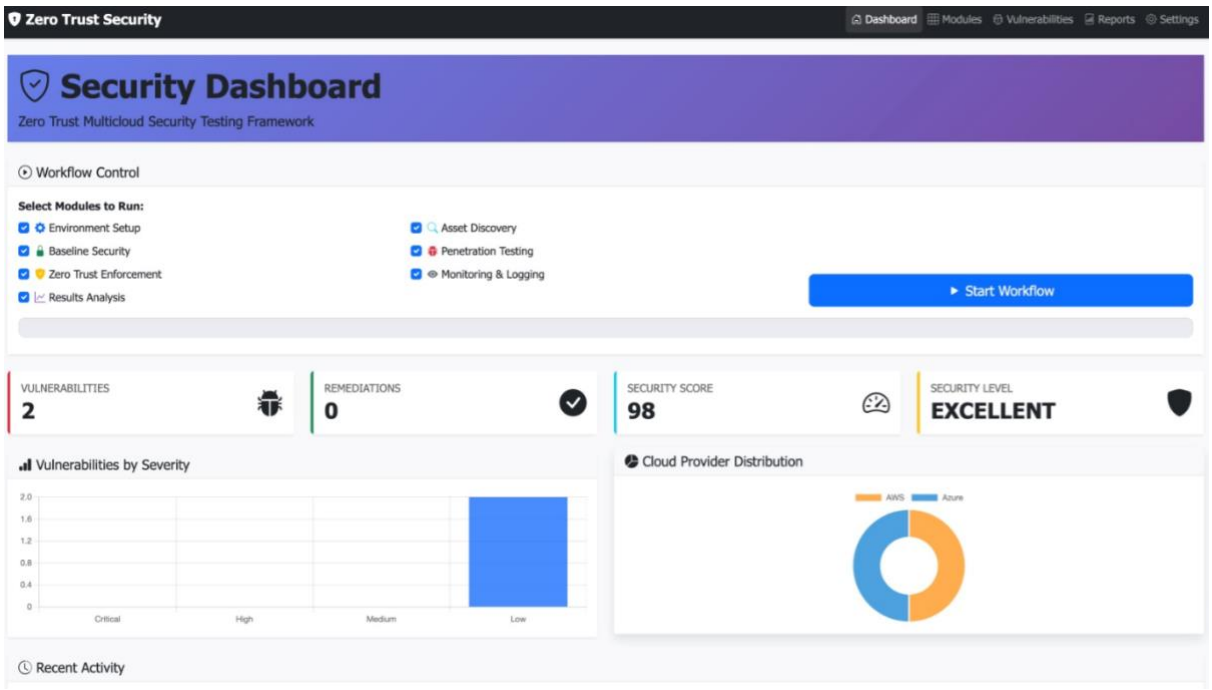
6.3 Step 6.3: (Alternative) Run a Partial Workflow

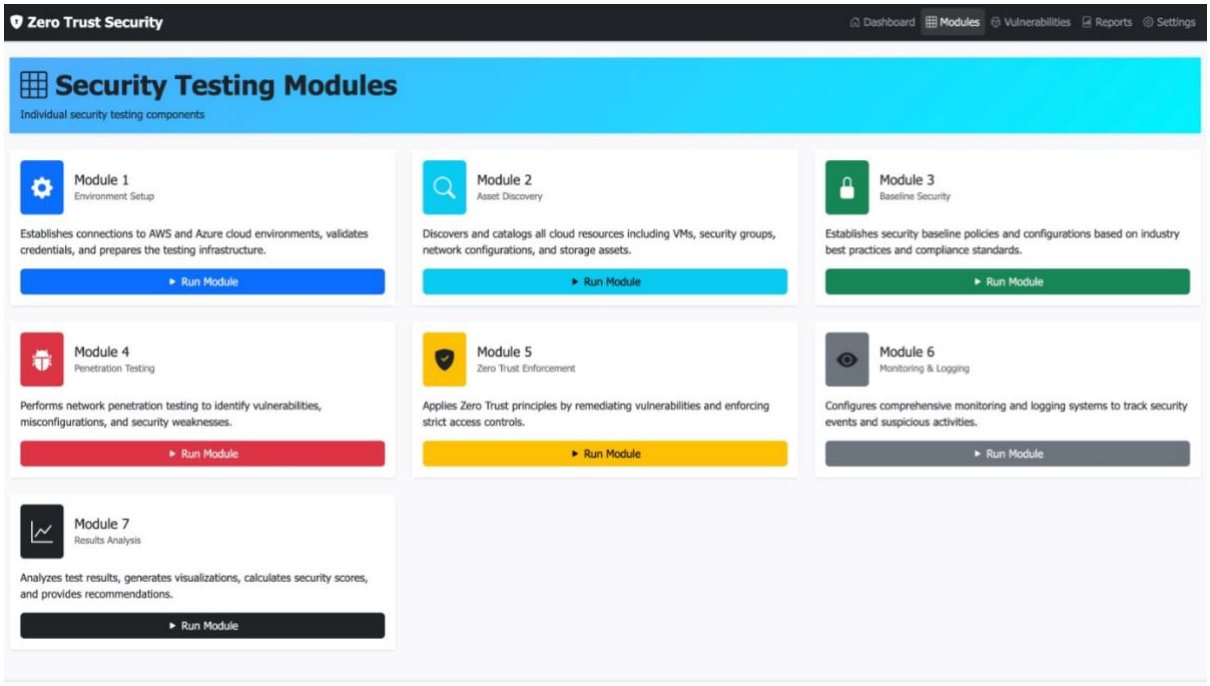
You can also run specific modules. For example, to run only the Penetration Testing (4), Enforcement (5), and Analysis (7) modules:

- python main.py --modules 4,5,7

Explanation: The --modules flag allows you to execute a subset of the workflow, which is useful for re-running tests without having to re-provision the entire environment.

7 Section 7: Verifying the Output





```
2025-12-08 03:55:10 - main_orchestrator - INFO -
2025-12-08 03:55:10 - main_orchestrator - INFO - Running full workflow...
2025-12-08 03:55:10 - main_orchestrator - INFO -
================================================================================
2025-12-08 03:55:10 - main_orchestrator - INFO -   MODULE: 1. Environment Setup
2025-12-08 03:55:10 - main_orchestrator - INFO - ================================================================================
=========

2025-12-08 03:55:10 - environment_setup - INFO - ============================================================
2025-12-08 03:55:10 - environment_setup - INFO - ZERO TRUST MULTICLOUD ENVIRONMENT SETUP
2025-12-08 03:55:10 - environment_setup - INFO - ============================================================
2025-12-08 03:55:10 - environment_setup - INFO - ============================================================
2025-12-08 03:55:10 - environment_setup - INFO - Verifying AWS Connection
2025-12-08 03:55:10 - environment_setup - INFO - ============================================================
2025-12-08 03:55:13 - environment_setup - INFO - [OK] AWS Account ID: 257394490853
2025-12-08 03:55:13 - environment_setup - INFO - [OK] AWS User ARN: arn:aws:iam::257394490853:user/AdminUser
2025-12-08 03:55:13 - environment_setup - INFO - [OK] AWS Region: us-east-1
2025-12-08 03:55:15 - environment_setup - INFO - [OK] Found 2 VPC(s)
2025-12-08 03:55:15 - environment_setup - INFO - ============================================================
2025-12-08 03:55:15 - environment_setup - INFO - Verifying Azure Connection
2025-12-08 03:55:15 - environment_setup - INFO - ============================================================
AzureCliCredential.get_token failed: Azure CLI not found on path
```

```
============================================================
2025-12-10 17:08:03 - results_analysis - INFO - Calculating Security Score
2025-12-10 17:08:03 - results_analysis - INFO - ============================================================
2025-12-10 17:08:03 - results_analysis - INFO - Security Score: 98/100
2025-12-10 17:08:03 - results_analysis - INFO - Security Level: EXCELLENT
2025-12-10 17:08:03 - results_analysis - INFO -
============================================================
2025-12-10 17:08:03 - results_analysis - INFO - Generating Visualizations
2025-12-10 17:08:03 - results_analysis - INFO - ============================================================
2025-12-10 17:08:03 - results_analysis - INFO - [OK] Saved: /Users/ajay/Documents/NCI/Research Practicum/Thesis Code/ajay_thesis/output/visualizations/vulnerabilities_by_severity.png
2025-12-10 17:08:03 - results_analysis - INFO - [OK] Saved: /Users/ajay/Documents/NCI/Research Practicum/Thesis Code/ajay_thesis/output/visualizations/vulnerabilities_by_cloud.png
2025-12-10 17:08:03 - results_analysis - INFO - [OK] Saved: /Users/ajay/Documents/NCI/Research Practicum/Thesis Code/ajay_thesis/output/visualizations/security_score.png
2025-12-10 17:08:04 - results_analysis - INFO - [OK] Saved: /Users/ajay/Documents/NCI/Research Practicum/Thesis Code/ajay_thesis/output/visualizations/remediation_effectiveness.png
2025-12-10 17:08:04 - results_analysis - INFO -
[OK] Visualizations saved to: /Users/ajay/Documents/NCI/Research Practicum/Thesis Code/ajay_thesis/output/visualizations
2025-12-10 17:08:04 - results_analysis - INFO -
============================================================
2025-12-10 17:08:04 - results_analysis - INFO - Security Recommendations
2025-12-10 17:08:04 - results_analysis - INFO - ============================================================
2025-12-10 17:08:04 - results_analysis - INFO -
1. [HIGH] Zero Trust Architecture
2025-12-10 17:08:04 - results_analysis - INFO -    Implement micro-segmentation across all resources
2025-12-10 17:08:04 - results_analysis - INFO -    -> Isolate workloads and apply principle of least privilege
2025-12-10 17:08:04 - results_analysis - INFO -
2. [MEDIUM] Identity & Access
2025-12-10 17:08:04 - results_analysis - INFO -    Enforce MFA for all administrative access
2025-12-10 17:08:04 - results_analysis - INFO -    -> Require multi-factor authentication for console and API access
2025-12-10 17:08:04 - results_analysis - INFO -
3. [MEDIUM] Monitoring & Logging
2025-12-10 17:08:04 - results_analysis - INFO -    Enable comprehensive logging across all cloud services
2025-12-10 17:08:04 - results_analysis - INFO -    -> Configure CloudTrail, GuardDuty, and Azure Security Center
2025-12-10 17:08:04 - results_analysis - INFO -
4. [LOW] Compliance
2025-12-10 17:08:04 - results_analysis - INFO -    Conduct regular security audits and compliance checks
2025-12-10 17:08:04 - results_analysis - INFO -    -> Implement automated compliance scanning and reporting
2025-12-10 17:08:04 - results_analysis - INFO -
============================================================
2025-12-10 17:08:04 - results_analysis - INFO - COMPREHENSIVE ANALYSIS SUMMARY
2025-12-10 17:08:04 - results_analysis - INFO - ============================================================
2025-12-10 17:08:04 - results_analysis - INFO -
+-------------------------+-----------+
| Total Vulnerabilities | 2         |
+-------------------------+-----------+
| - Critical            | 0         |
+-------------------------+-----------+
| - High                | 0         |
+-------------------------+-----------+
| - Medium              | 0         |
+-------------------------+-----------+
| - Low                 | 2         |
+-------------------------+-----------+
|                         |           |
+-------------------------+-----------+
| Total Remediations    | 0         |
+-------------------------+-----------+
| Remediation Rate      | 0.0%      |
+-------------------------+-----------+
|                         |           |
+-------------------------+-----------+
| Security Score        | 98/100    |
+-------------------------+-----------+
| Security Level        | EXCELLENT |
+-------------------------+-----------+
2025-12-10 17:08:04 - results_analysis - INFO -
[OK] Analysis results saved to: /Users/ajay/Documents/NCI/Research Practicum/Thesis Code/ajay_thesis/output/analysis_results_20251210_170804.json
2025-12-10 17:08:04 - main_orchestrator - INFO -
[OK] 7. Results Analysis completed successfully
2025-12-10 17:08:04 - main_orchestrator - INFO -   Execution time: 0.75 seconds
```

After the main.py script completes, you can verify the results.

1. **Console Output:** The terminal will display a final **EXECUTION SUMMARY** table showing the status (SUCCESS or FAILED) and execution time for each module.
2. **Output Files:** Locate the /output directory in your project. You'll find a bunch of time-stamped JSON files containing the raw data , data from each stage:
    - asset_inventory_*.json: A complete list of all discovered AWS and Azure resources.
    - pentest_results_*.json: Detailed results of the port scan, including open ports and identified vulnerabilities.
    - enforcement_actions_*.json: A log of any automated remediation actions taken.

    - analysis_results_*.json: The final calculated metrics, including the security score and recommendations.
3. **Visualizations:** Navigate to the output/visualizations/ directory. You will find several PNG image files, such as:
    - vulnerabilities_by_severity.png
    - security_score.png
    - remediation_effectiveness.png

These charts provide a graphical summary of the experiment's findings. You have now successfully configured and executed the entire framework.