HoneyPot Hosted in Cloud

Project Title: Deployment of Honeypot on Microsoft Azure Cloud

Author: Bhargav Pavan Sai Surisetti

Date: 07 October 2025

Objective:

The objective of this step is to set up a Microsoft Azure account, which will serve as the cloud platform for deploying and managing the honeypot infrastructure. Azure provides scalable computing, storage, and network services that are essential for hosting and monitoring cyber-attack activities in a controlled environment.

Description:

In cybersecurity research, a honeypot is a **deceptive system** designed to attract attackers and record their actions. Deploying it on **Microsoft Azure Cloud** provides several advantages, including global accessibility, scalability, secure isolation, and integrated monitoring services.

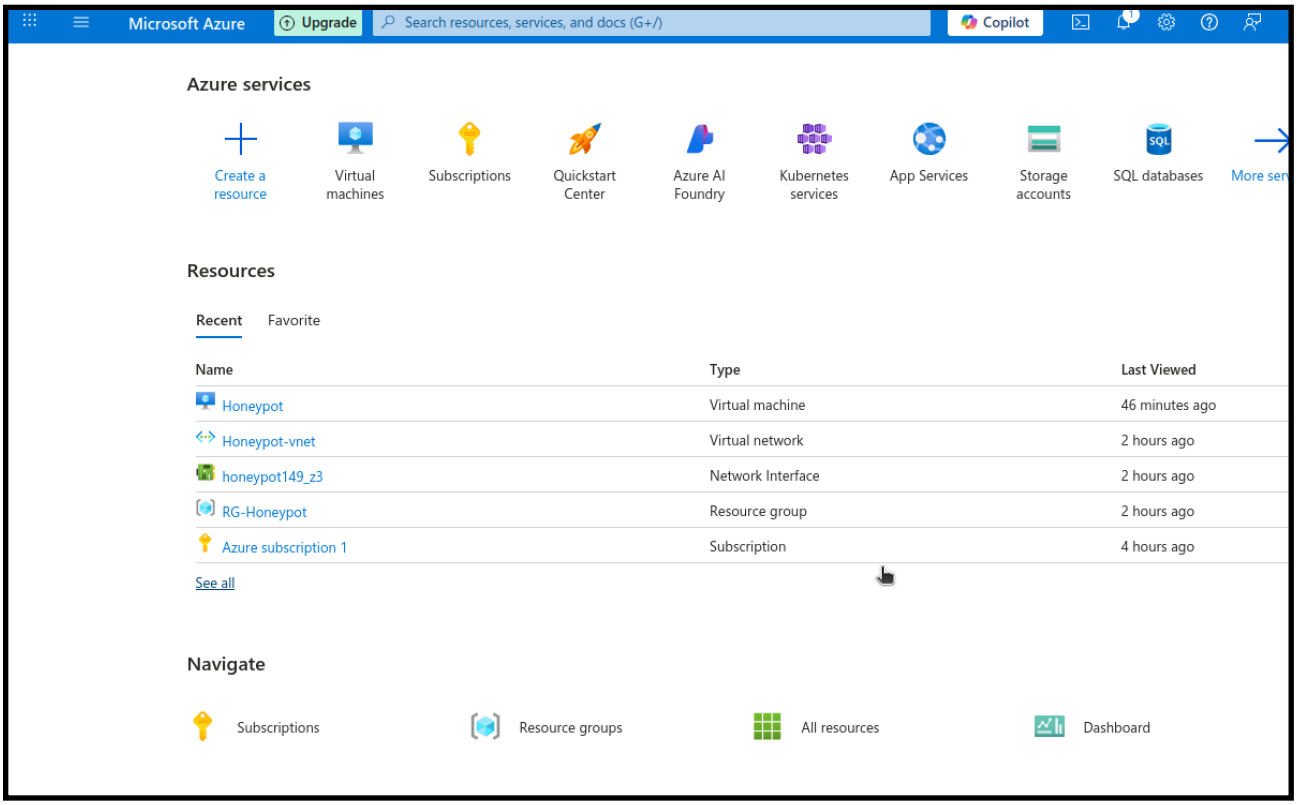
STEP-1 : Creating a Microsoft Azure account

Creating an Azure account is the first and most essential step, as it enables access to cloud resources such as virtual machines, virtual networks, firewalls, and security monitoring tools. These components will later be used to build and observe the honeypot environment.

Procedure

1. Visit the official **Microsoft Azure website** (https://azure.microsoft.com/).
2. Click on **“Start Free”** or **“Azure for Students”**, depending on eligibility.
3. Sign in with an existing Microsoft account or create a new one.
4. Provide personal information such as name, email, and region.
5. Complete the **identity verification** process using a valid phone number and credit/debit card (not required for student subscriptions).

6. Once verification is complete, the Azure account will be activated, and the **Azure Portal Dashboard** will become accessible for managing resources.



STEP-2 : Creating a Virtual Machine in Microsoft Azure

Description

A **Virtual Machine (VM)** is the fundamental building block of a honeypot setup. It provides an isolated computing environment where simulated vulnerable services can be hosted safely.

By using Azure's VM service, it becomes possible to quickly create, configure, and manage virtualized systems without the need for physical hardware.

For the honeypot project, the VM will be configured with:

- A lightweight operating system (commonly **Ubuntu Server** or **Windows Server**, depending on the honeypot type).
- A public IP for controlled external access (to attract attackers).
- Basic network security configurations to manage and restrict incoming/outgoing traffic.

- Sufficient compute resources (CPU, memory, and storage) to run honeypot services without high cost.

The VM serves as the **trap environment** where attackers will interact with simulated vulnerable applications, allowing collection of logs, attack signatures, and intrusion patterns.

Procedure

1. **Log in to Azure Portal:**

Open the Azure Portal and sign in using your Azure credentials.

2. **Navigate to Virtual Machines:**

From the left-hand panel, select **“Virtual Machines”** and click on **“Create” → “Azure Virtual Machine.”**

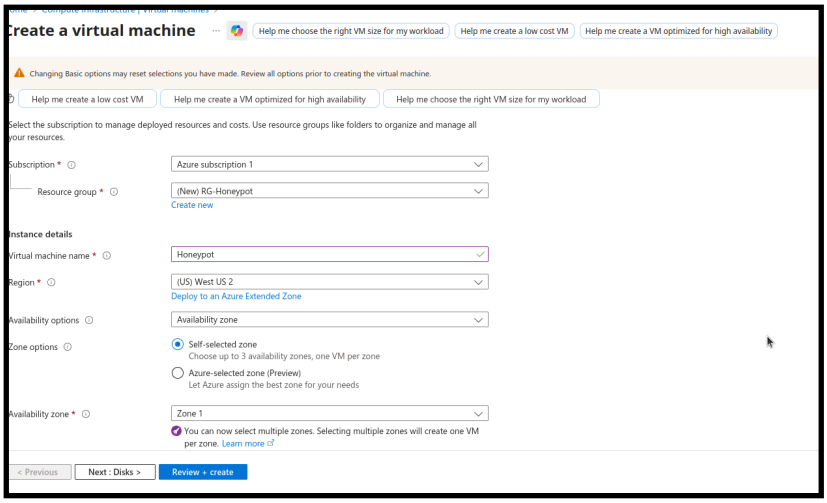
3. **Basic Configuration:**

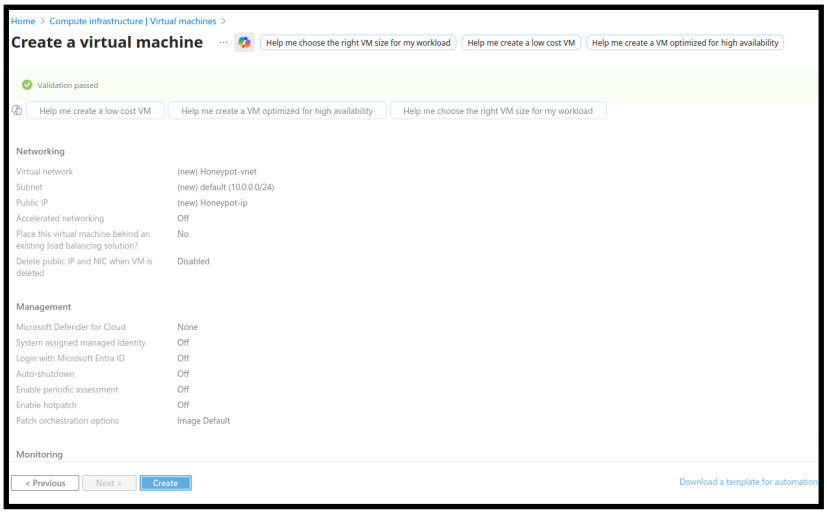
   - **Subscription:** Choose your active subscription.
   - **Resource Group:** Select the resource group created in Step 1 (e.g., RG-Honeypot).
   - **Virtual Machine Name:** Provide a name (e.g., honeypot-vm).
   - **Region:** Select a region closest to your location or as per project requirement.
   - **Availability Options:** Choose “No infrastructure redundancy required” for small-scale deployment.
   - **Image:** Select the operating system image (e.g., *Ubuntu Server 22.04 LTS* or *Windows Server 2019*).
   - **Size:** Choose a VM size such as *Standard_B1s* for cost-effective performance.
   - **Authentication Type:** Use SSH key (recommended) or password.
   - **Username:** Create an administrator username (e.g., azureuser).

4. **Networking Setup:**
    - Select or create a **Virtual Network (VNet)** and **Subnet**.
    - Assign a **Public IP** for the VM to make it accessible over the internet.
    - Under **Network Security Group (NSG)**, open required ports:
        - **Port 22 (SSH)** for Linux or **Port 3389 (RDP)** for Windows.
        - Additional ports may be opened later to simulate vulnerabilities (for honeypot functionality).
5. **Review + Create:**

Review all configurations and click **“Create.”**

Azure will deploy the VM, and you’ll receive confirmation once it’s active.





6 • Add an Inbound NSG Rule — Allow Ports 1–65535:

• In the NSG, add a new inbound security rule with the following recommended settings:

• Source: Any (or restrict to specific IP ranges if you want partial control)

• Source port ranges: **

• Destination: Any or the VM's IP (recommended: Any)

• Destination port ranges: 1-65535

• Protocol: Any (TCP/UDP)

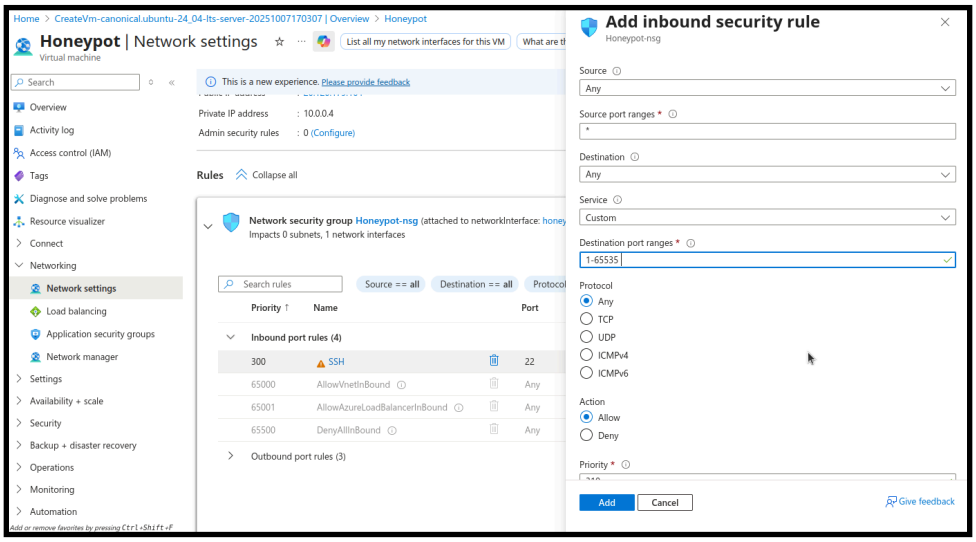
• Action: Allow

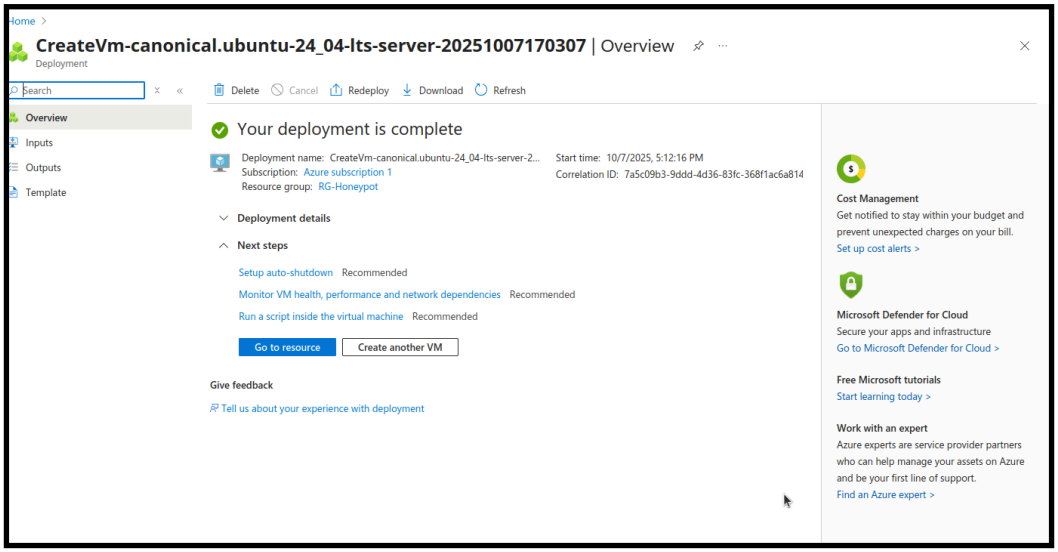
• Priority: Set a priority value (e.g., 100) — pick a lower number than default deny rules so this rule is evaluated earlier.

• Name: Allow-All-Ports-Inbound-Honeypot

• Description: Allow inbound traffic on ports 1-65535 for honeypot monitoring (isolated environment).

• Save the rule and confirm the NSG is applied to the VM's NIC/subnet.





Step 3: Retrieve Installation Artifacts from GitHub and Connect to the Azure VM

Objective:

Use your existing GitHub repository (which contains the full installation process) to provision and configure the honeypot on the Azure VM. Establish an SSH connection from your Ubuntu workstation (PuTTY or native ssh) to the VM, fetch the repository on the VM, validate the installation scripts, and run the installation workflow as authored in your repo.

Description

This step leverages a version-controlled repository to ensure reproducible, auditable installation of the honeypot. Pulling the installation artifacts directly onto the VM (or transferring them from your workstation) reduces human error and enables easy updates, rollbacks, and collaboration. The documentation below covers:

- Connecting to the VM (PuTTY GUI on Ubuntu or native ssh).
- Cloning the GitHub repository on the VM (HTTPS or SSH).
- Verifying repository integrity and inspecting install scripts.
- Running the installation process from the repo.
- Basic rollback, verification, and logging steps.
  Security reminder: **Always inspect** scripts before executing them. Run in an isolated honeypot environment only.

GitHub repository (installation process):

https://github.com/telekom-security/tpotce

PuTTY — download & installation (Ubuntu):

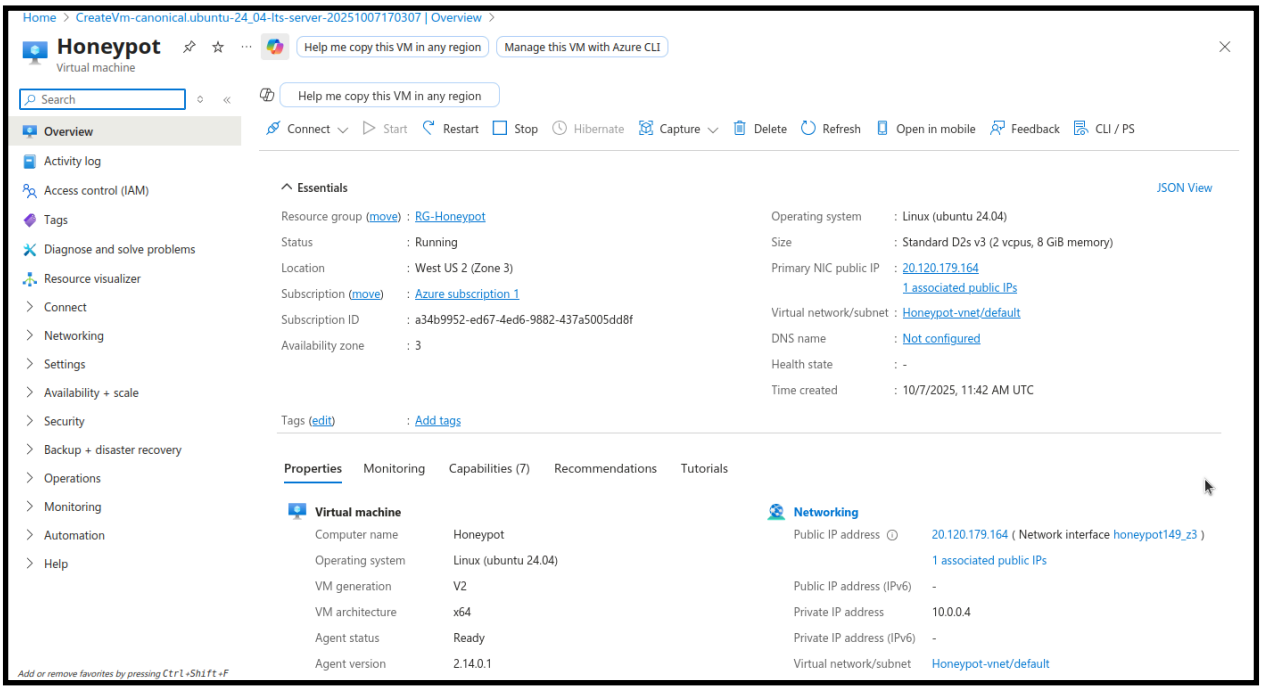
https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

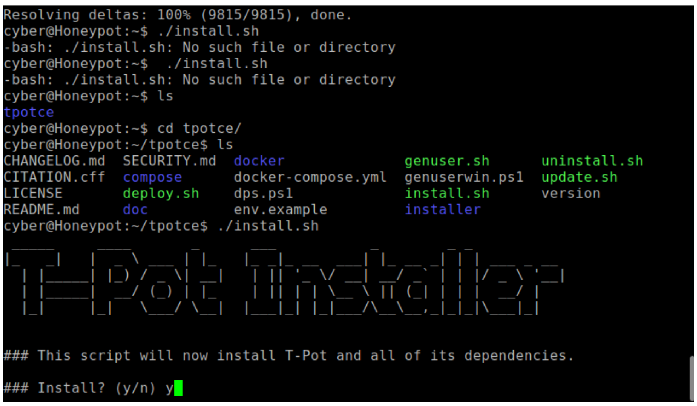
T-Pot — canonical installation commands:

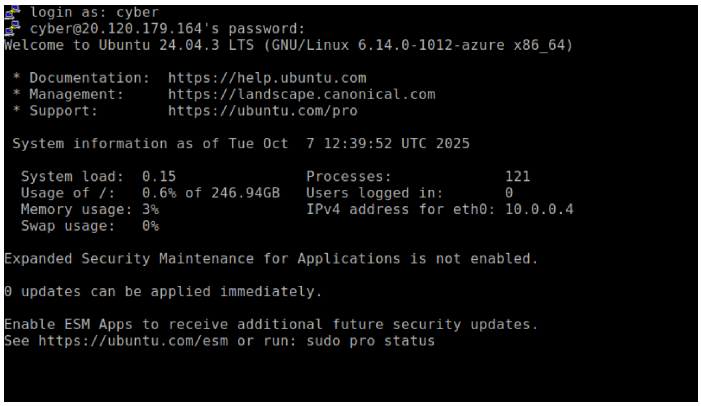
```
sudo apt update
sudo apt upgrade -y
sudo apt install git
sudo git clone https://github.com/telekom-security/tpotce
```

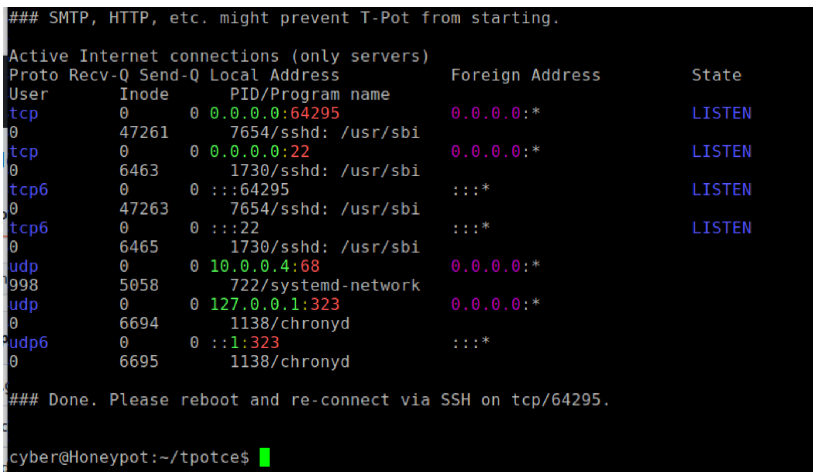
sudo cd tpotce/iso/installer/

sudo ./install.sh --type=user









Step 4: Reboot and Access Honeypot via Custom Port (64295)

Objective

To perform a controlled reboot of both the **Azure Virtual Machine** and the **PuTTY session**, then establish a successful connection to the honeypot environment using the **custom SSH port (64295)** and valid credentials. This ensures that the honeypot is accessible, stable, and fully operational after installation and configuration.

Description

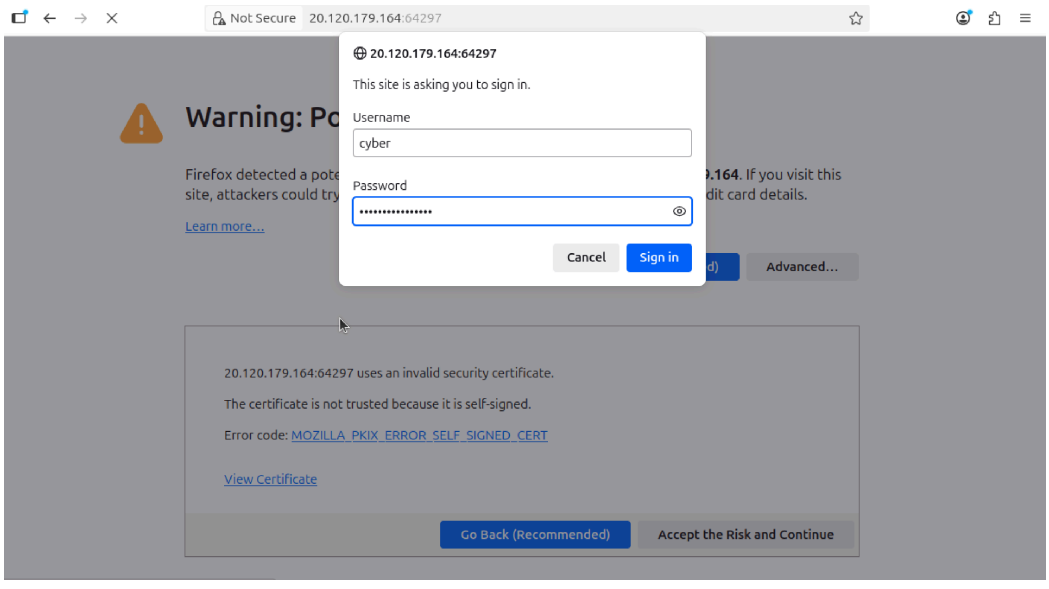
After the honeypot installation (e.g., T-Pot or other honeypot frameworks), the system may automatically configure a **non-default SSH port** for enhanced security and administrative isolation.

The default SSH port 22 is replaced by a **custom port (64295)** to minimize unauthorized login attempts and brute-force scans.

Following installation, it is critical to:

- **Reboot** the VM to ensure all honeypot services and network configurations load properly.
- **Reboot PuTTY** to clear any cached sessions and reinitialize the connection using the new port.
- **Reconnect** using the VM's **public IP address** and the **custom SSH port** to verify remote access functionality.
  This step confirms the honeypot's post-installation accessibility and verifies that SSH configuration changes persist across system reboots.

Note:Follow the above mentioned GitHub repo.



Step 5: Accessing T-Pot Web Interface and Dashboards

Objective

To access the T-Pot web interface after successful deployment and login to verify that all integrated dashboards, monitoring tools, and attack visualization features are operational. This step ensures that the honeypot data is correctly collected, stored, and presented for analysis.

Description

T-Pot provides a web-based dashboard to monitor honeypot activity in real-time. After installation and login (Step 4), the T-Pot web interface allows administrators to:

- Visualize attack sources and geolocations (Attack Map).
- Explore captured attack payloads (Cyberchef).
- Analyze and query Elasticsearch indices (Elasticvue, Kibana).
- Run automated reconnaissance and intelligence tools (Spiderfoot).
- Assess honeypot security and operational metrics (SecurityMeter).
- Access official documentation (T-Pot README) and GitHub resources.

Procedure

1. **Open the T-Pot Web Interface**
   - Launch a web browser on a system that can reach the Azure VM.
   - Enter the VM's **public IP address** followed by the T-Pot web port (commonly 64297 or as configured during install).

Example: http://<AZURE_PUBLIC_IP>:64297

2. **Login Credentials**
   - Use the administrative credentials created during installation (usually the same used for SSH or as prompted by the installer).
   - Confirm successful login to the T-Pot landing page.

3. **Verify Dashboard Elements**

The landing page should display links or widgets for the following modules:

- **Attack Map** – visual representation of global attack sources.
- **Cyberchef** – tool for decoding, analyzing, and transforming captured payloads.
- **Elasticvue** – Elasticsearch management and query interface.
- **Kibana** – real-time visualization of honeypot logs and alerts.
- **Spiderfoot** – automated threat intelligence and reconnaissance tool.
- **SecurityMeter** – overview of honeypot security and operational metrics.
- **T-Pot ReadMe** – documentation for reference.
- **T-Pot @ GitHub** – link to the official repository for updates and community resources.

4. **Timestamp Verification**

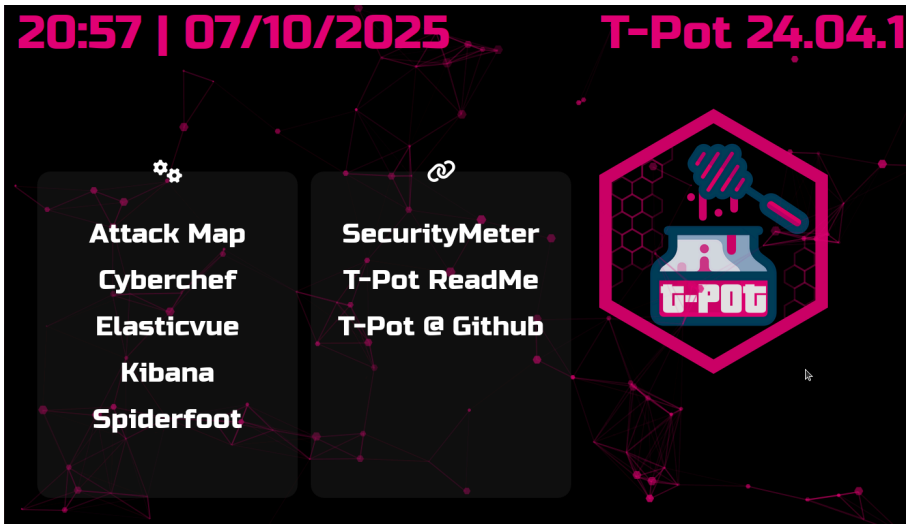
- Ensure that the page displays the **current system date and time** along with the T-Pot version, e.g., 18:40 | 07/10/2025 T-Pot 24.04.1

5. **Initial Functionality Test**

- Click through the main modules to confirm they load without errors.
- Verify that recent attacks (if any) are visible in the Attack Map and Kibana dashboards.
- Confirm that captured payloads can be analyzed in Cyberchef.

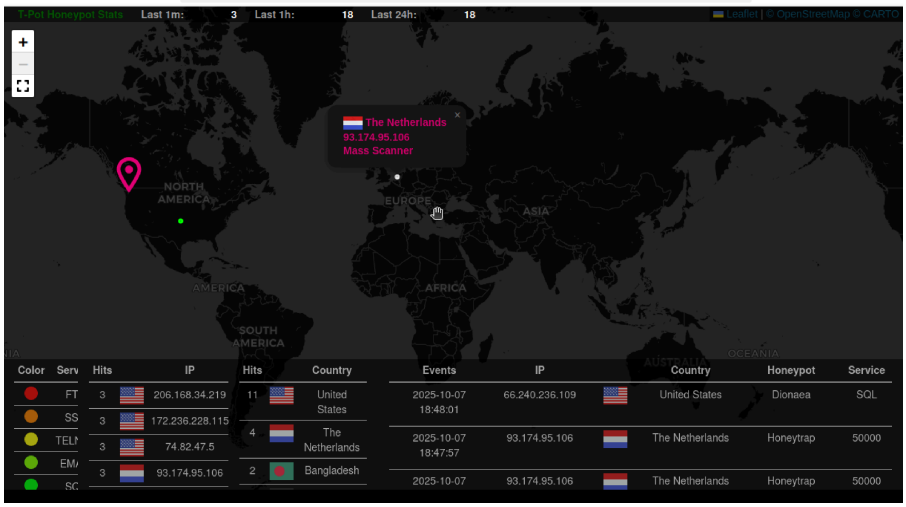
1. Landing Page / Home Screen

- **Description:** Displays T-Pot version, current date/time, and navigation to all modules.
- **Screenshot:**



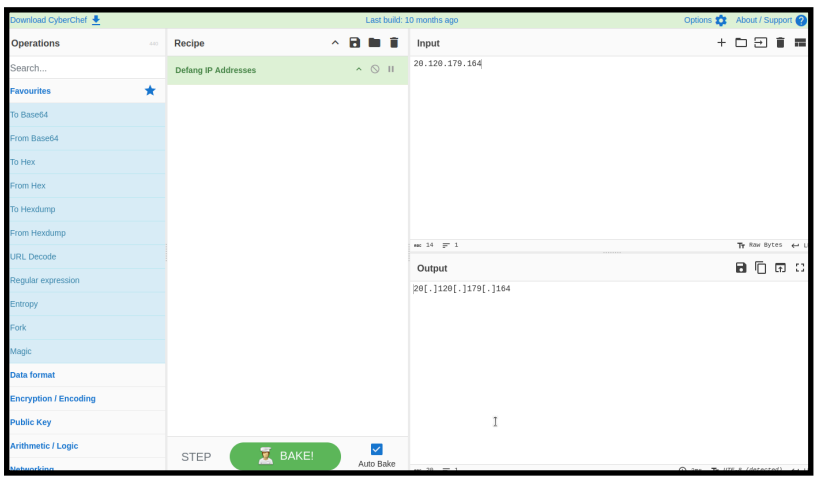
2. Attack Map

- **Description:** Visualizes incoming attacks geographically; helps track attacker locations and patterns.
- **Screenshot:**



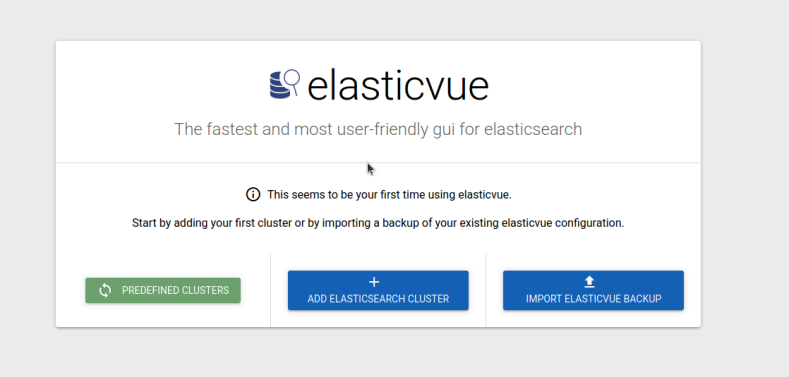
3. Cyberchef

- **Description:** Tool to decode, analyze, and transform captured attack payloads and data.
- **Screenshot:**



4. Elasticvue

- **Description:** Elasticsearch management interface for querying and exploring honeypot data.
- **Screenshot:**

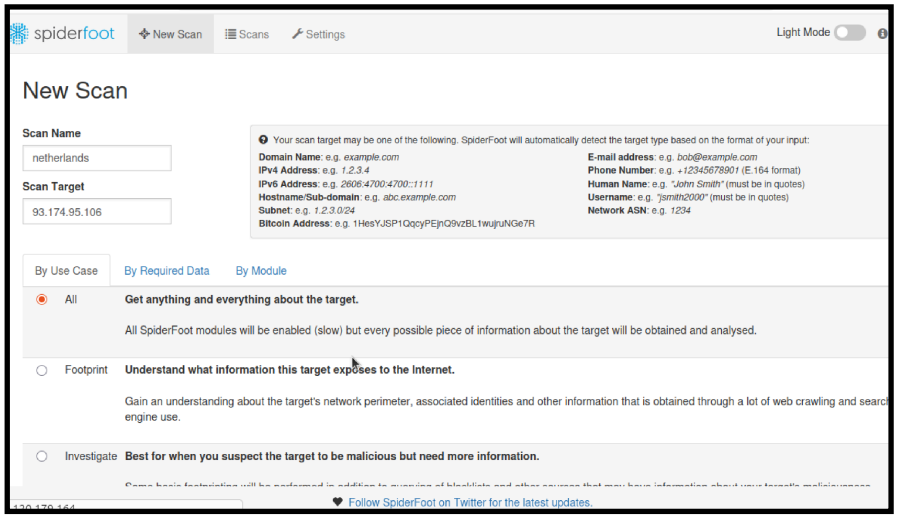


5. Kibana

- **Description:** Visualizes logs and analytics from all honeypot sensors; supports dashboards and alerting.
- **Screenshot:**

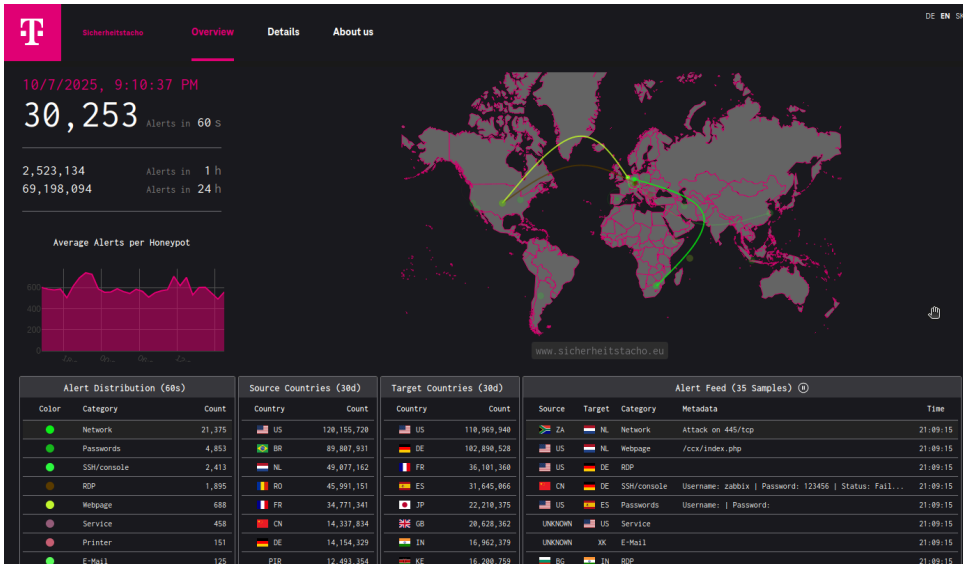
6. Spiderfoot

- **Description:** Automated threat intelligence and reconnaissance tool integrated within T-Pot.
- **Screenshot:**



7. SecurityMeter

- **Description:** Displays operational metrics, security status, and performance of the honeypot environment.
- **Screenshot:**



8. T-Pot ReadMe

- **Description:** Provides in-browser access to the official T-Pot documentation.
- **Link:** *https://github.com/telekom-security/tpotce/blob/master/README.md*

9. T-Pot @ GitHub

- **Description:** Direct link to the official T-Pot GitHub repository for source code, updates, and community support.
- **Link:** *https://github.com/telekom-security/tpotce/*

Step-6 : Importance of the Project: Deployment of Honeypot on Microsoft Azure Cloud

Purpose and Significance

The deployment of a cloud-hosted honeypot like T-Pot has multiple strategic and operational benefits:

1. **Threat Intelligence Collection**
   - Captures real-world attacks, malware, and reconnaissance attempts targeting exposed services.
   - Provides early warning signals for emerging threats in your network segment or globally.
2. **Attack Analysis and Research**
   - Logs all attacker behavior, including scanning techniques, payload delivery, and exploitation attempts.
   - Enables detailed forensic analysis of malicious activity without risk to production systems.
3. **Cybersecurity Awareness and Testing**
   - Helps security teams understand attack patterns and vectors targeting public-facing services.
   - Allows organizations to test IDS/IPS systems, firewall rules, and monitoring workflows using real attack data.
4. **Operational Security Validation**
   - Ensures that security configurations, logging, and containment measures are functioning as expected in a real attack scenario.
5. **Educational and Training Value**
   - Acts as a controlled environment for cybersecurity students, analysts, and researchers to study attacker behavior.
   - Demonstrates the lifecycle of attacks and threat mitigation in a safe, isolated cloud environment.

Remedies and Best Practices for Operating a Honeypot

While honeypots are powerful tools, misconfigurations can pose risks. The following remedies and precautions ensure safe and effective operation:

1. **Isolation from Production Networks**
   - Deploy honeypots in a separate VNet or subscription to prevent lateral movement.
   - Use firewalls and NSGs to restrict communication to only monitored channels.
2. **Controlled Exposure**
   - Open only necessary ports for the honeypot services; if exposing full port ranges, ensure it is an isolated VM.
   - Restrict SSH and admin access to trusted IPs.
3. **Logging and Monitoring**
   - Enable NSG flow logs, Azure Monitor, and system/application logging.
   - Centralize logs in a secure storage or SIEM for analysis and alerting.
4. **Regular Updates and Patching**
   - Keep honeypot software and OS updated to prevent exploitation of unmonitored vulnerabilities.
   - Ensure T-Pot, Docker, and associated services are at their latest stable versions.
5. **Automated Containment and Recovery**
   - Use snapshots or VM images to quickly revert to a clean state if the honeypot is compromised.
   - Consider automation rules to isolate suspicious outbound traffic to prevent misuse by attackers.
6. **Credential Management and Access Control**
   - Use key-based authentication for administrative access.
   - Rotate credentials regularly and avoid hard-coded passwords in scripts.
7. **Legal and Ethical Compliance**
   - Operate honeypots in accordance with local laws and organizational policies.
   - Avoid collecting sensitive personal data and inform relevant stakeholders if required.
8. **Periodic Review and Analysis**
   - Regularly analyze collected attack data to adjust monitoring, firewall rules, and defensive strategies.
   - Document findings to improve future threat detection

Summary

The **deployment of a honeypot in Microsoft Azure** serves as a proactive cybersecurity measure. It enables organizations and researchers to:

- Observe real-world attacker behavior.
- Test and strengthen network defenses.
- Conduct cybersecurity research safely.

At the same time, implementing proper remedies ensures that the honeypot remains a **controlled, isolated, and secure environment**, preventing it from becoming a liability while maximizing its intelligence-gathering potential.

THANK YOU