

```
$ ssh -p 22 unknown@beestreet
Connecting to beestreet...
Connection established.
Server: Cowrie SSH Honeypot
Fingerprint:
  SHA256:22:B3:EE:ST:RE:F6:22:
  H0:NE:YP:0T:22:00:22:22:22
Warning: You are being logged.
All actions are monitored.
unknown@beestreet:~$
```

CONNECTION LOGGED

|            |               |
|------------|---------------|
| IP ADDRESS | 203.0.113.22  |
| LOCATION   | UNKNOWN       |
| USER       | unknown       |
| ACTION     | login attempt |
| STATUS     | captured      |

# PROJECT 22 BEE STREET

CLOUD SSH HONEYPOT

HONEY. DATA. CONTROL.  
WE DON'T GET HACKED.  
WE HUNT.

PORT  
22  
SSH



DECEPTION LAYER: ACTIVE  
HONEYPOT STATUS: ONLINE



AWS EC2



COWRIE



PORT 22 DECEPTION

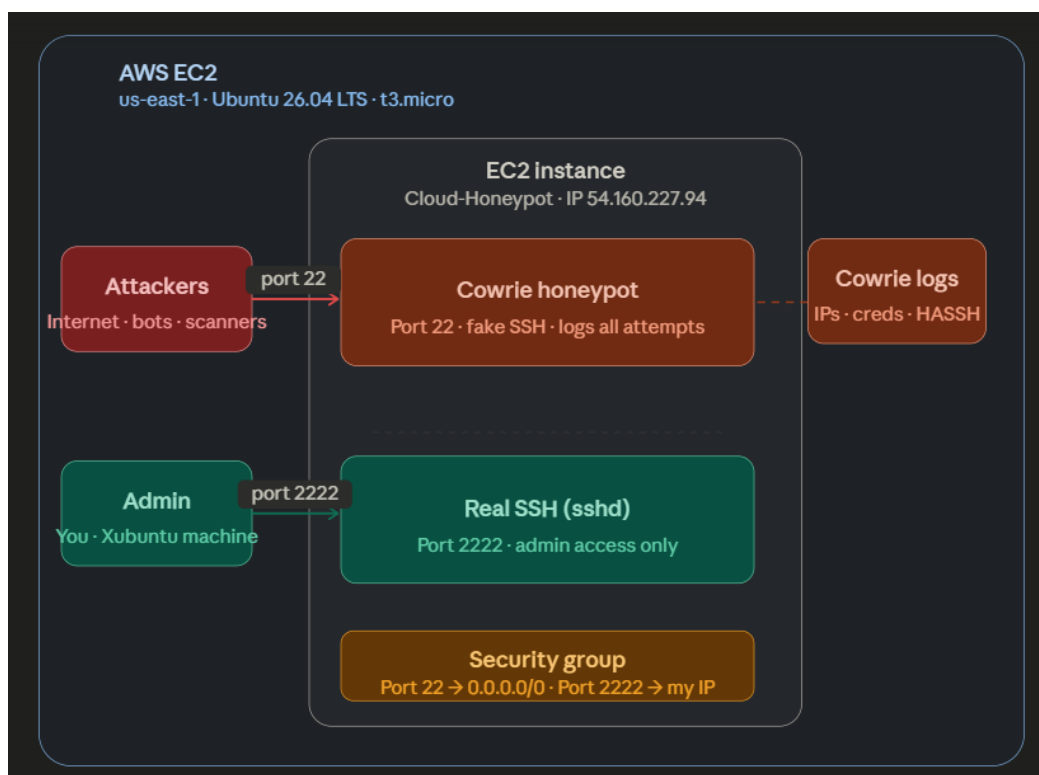
# Phase 1 - Cloud Honeypot Foundation

## Introduction

This phase created the foundation for a cloud honeypot deployed on an AWS EC2 instance. The central problem addressed in Phase 1 is that internet-facing cloud SSH services are constantly targeted by automated scanning, brute-force attacks, and credential-stuffing attempts.

One [2026 Linux security analysis](#) showed that fresh cloud server instances can receive their first SSH probing attempt within 90 seconds of deployment. A [2025 security report](#) also demonstrated that attacks against internet-facing login services and edge infrastructure were occurring from approximately 2.8 million IP addresses per day, indicating that these attacks are heavily automated and botnet-driven.

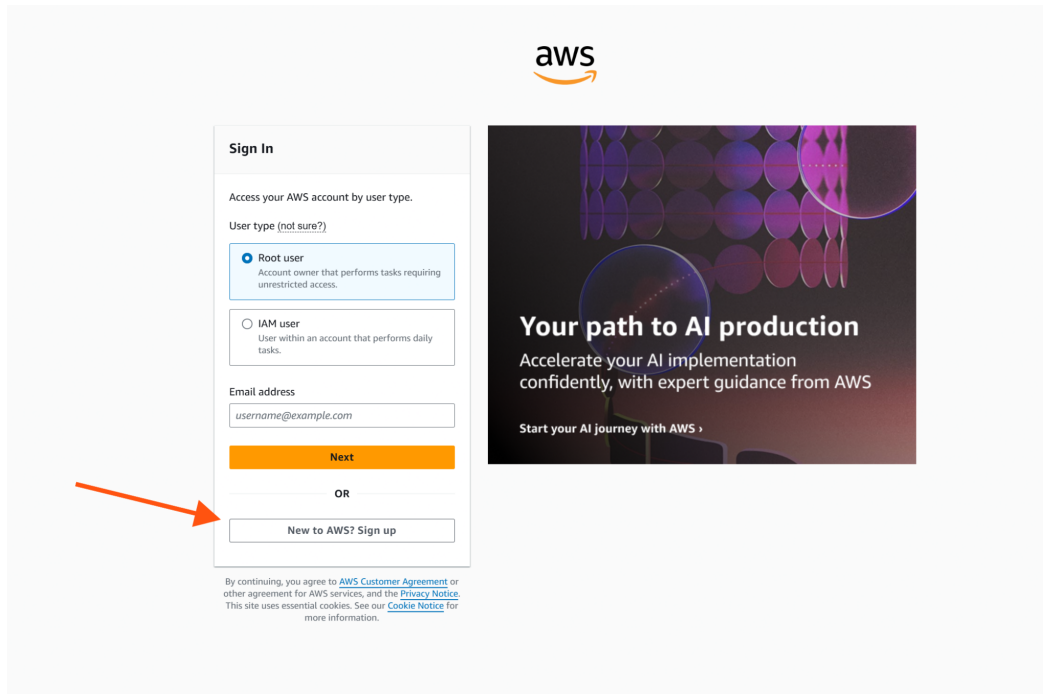
Phase 1 addresses this problem by hardening the SSH architecture. The real SSH service is moved from the default port, 22, to a nonstandard port, 2222. Cowrie is then deployed on port 22, SSH's default port, to act as a honeypot and capture attacker activity while keeping legitimate administrative access separate.



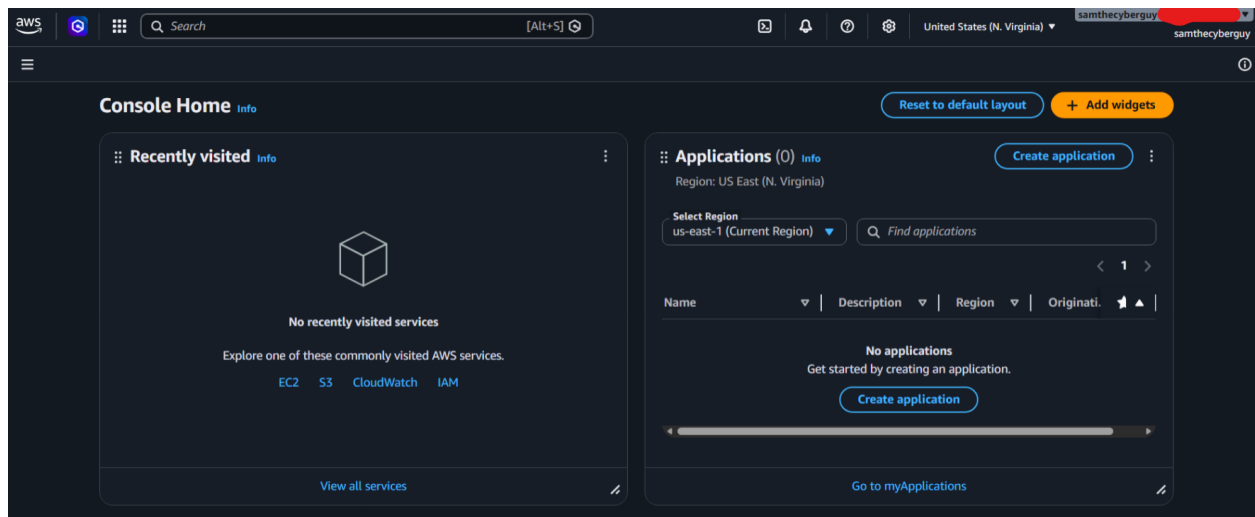
*Cloud Honeypot Foundation Diagram*

## Step 1 - Get an AWS Account

Set up a [free AWS account](#) on Windows 11. New users receive 100 dollars in credit, which can be used for 6 months.

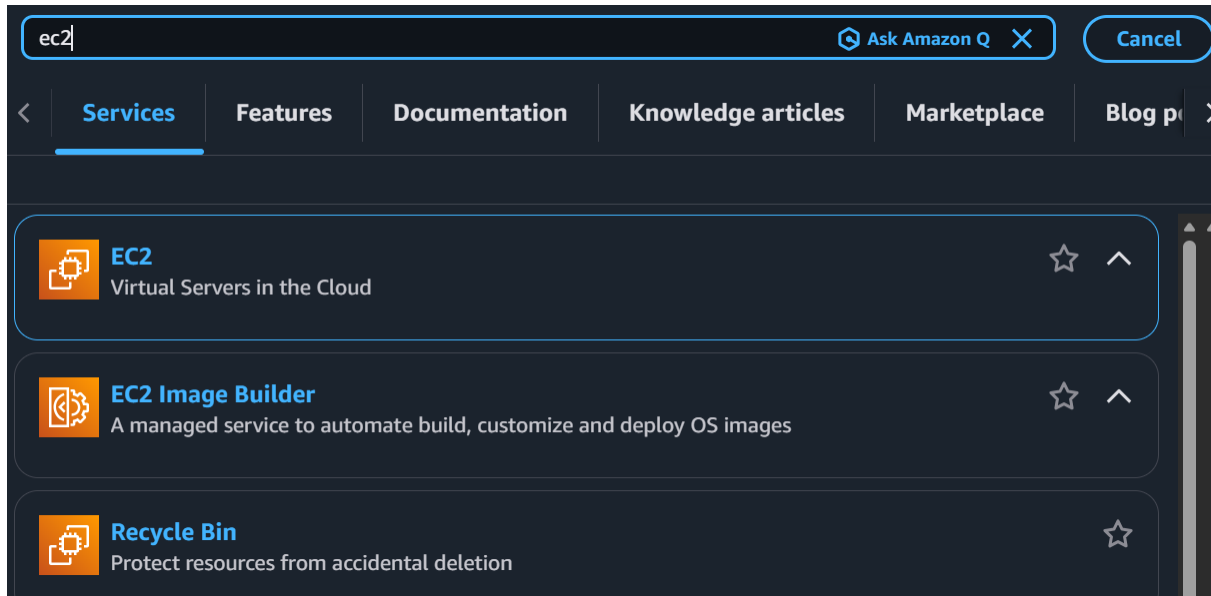


The image shows the AWS Sign In page. On the left is a 'Sign In' form with the following fields: 'User type (not sure?)' with radio buttons for 'Root user' (selected) and 'IAM user'; 'Email address' with a text input field containing 'username@example.com'; and a 'Next' button. Below the form is a link 'New to AWS? Sign up' which is pointed to by a red arrow. On the right is a promotional banner for 'Your path to AI production' with the text 'Accelerate your AI implementation confidently, with expert guidance from AWS' and 'Start your AI journey with AWS'.

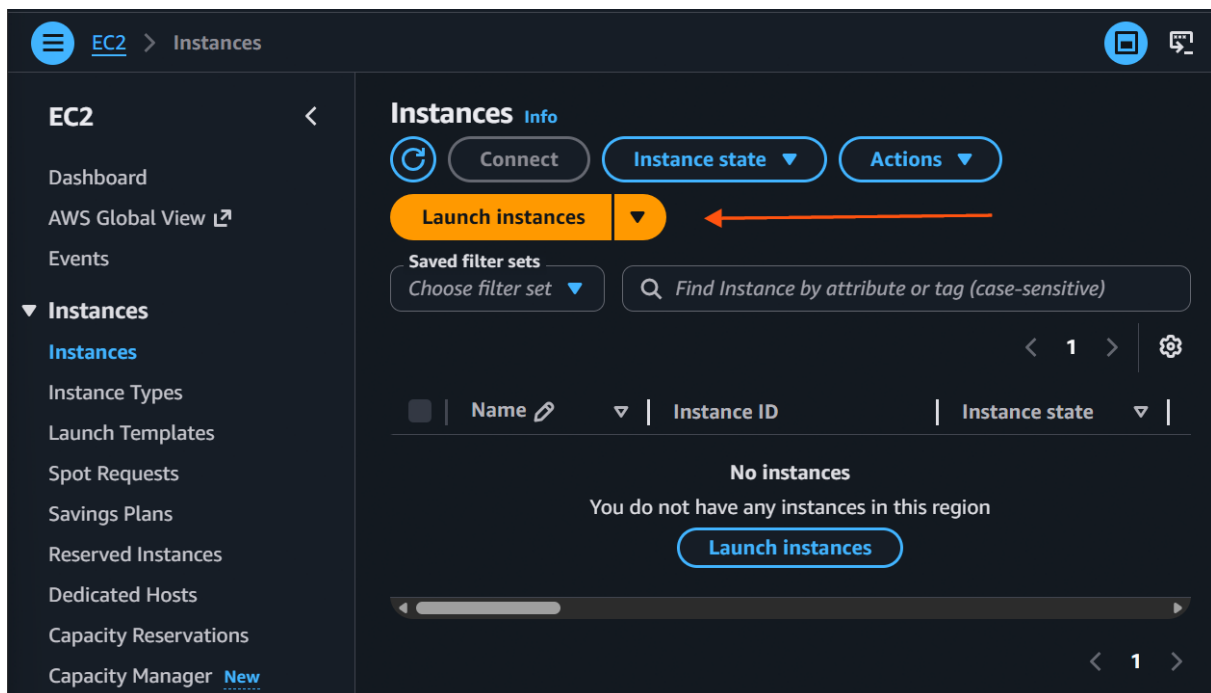


## Step 2 - Configure and Launch an EC2 Instance

Search for **EC2**. After selecting it, navigate to **Instances**.



Open the **Launch instances** dropdown and select **Launch an instance**.



Configure the instance. Name the instance **Cloud-Honeypot**. Choose **Ubuntu Server 26.04 LTS x86** as the OS. Select **t3.micro** as the instance type. Create a key pair

called **Honeypot-Key** and choose **RSA** and **.pem**. Save the key pair file to the Linux machine where the honeypot will be deployed. It will be used to login to SSH later.

## Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

### Name and tags Info

Name

Cloud-Honeypot

[Add additional tags](#)

### ▼ Application and OS Images (Amazon Machine Image) Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose **Browse more AMIs**.

 Search our full catalog including 1000s of application and OS images

**AMI from catalog**

**Quick Start**

**Name**

Ubuntu Server 26.04 LTS (HVM), SSD Volume Type

Verified provider

Free tier eligible



**Browse more AMIs**

Including AMIs from AWS, Marketplace and the Community

**Description**

Ubuntu Server 26.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 26.04, amd64 resolute image

**Image ID**

ami-091138d0f0d41ff90

**Username** 

ubuntu

**Catalog**

Quick Start AMIs

**Published**

2026-04-21T06:30:56.000Z

**Architecture**

x86\_64

**Virtualization**

hvm

**Root device type**

ebs

**ENA Enabled**

Yes

**Boot mode**

uefi-preferred

▼ **Instance type** [Info](#) | [Get advice](#)

**Instance type**

t3.micro Free tier eligible

Family: t3 2 vCPU 1 GiB Memory Current generation: true

On-Demand Linux base pricing: 0.0104 USD per Hour

On-Demand Windows base pricing: 0.0196 USD per Hour

On-Demand Ubuntu Pro base pricing: 0.0139 USD per Hour

On-Demand SUSE base pricing: 0.0104 USD per Hour

On-Demand RHEL base pricing: 0.0392 USD per Hour

☐ All generations

[Compare instance types](#)

**Additional costs apply for AMIs with pre-installed software**

▼ **Key pair (login)** [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

**Key pair name - required**

HoneyPot-Key

[Create new key pair](#)

My files > **Cyber-Cloud Projects**

| <input type="checkbox"/>                                                            | Name                       | Size    |
|-------------------------------------------------------------------------------------|----------------------------|---------|
|  | SOC Cloud Honeypot Project | 1.01 ME |
|  | HoneyPot-Key.pem           | 1.64 KE |

To configure the security group within network settings, start by selecting **Create security group**. Then, navigate under **Inbound Security Group Rules**. There, for **Type** select **ssh** on port **22** and for **Source type** select **My IP**.

▼ Network settings

Info

VPC - required

Info

vpc-00fdeb19593bb66ac

(default) ▼

↻

Subnet

Info

No preference ▼

↻ Create new subnet ↗

Availability Zone

Info

No preference ▼

↻ Enable additional zones ↗

Auto-assign public IP

Info

Enable ▼

Firewall (security groups)

Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

Security group name - required

launch-wizard-1

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and . \_ - / ( ) # , @ [ ] + = & ; ' ! \$ \*

Description - required

Info

launch-wizard-1 created 2026-05-14T03:41:12.243Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 146.70.230.131/32)

Remove

| Type  | Protocol | Port range | Source type | Name                                                                                    | Description - optional     |
|-------|----------|------------|-------------|-----------------------------------------------------------------------------------------|----------------------------|
| Info  | Info     | Info       | Info        | Info                                                                                    | Info                       |
| ssh ▼ | TCP      | 22         | My IP ▼     | <div> <div>🔍 Add CIDR, prefix list or securit</div> <div>146. [redacted] ✕</div> </div> | e.g. SSH for admin desktop |

Add security group rule

Select **Launch instance** and AWS will provision the virtual linux server, public IP, networking, and storage.

7

▼ Summary

Number of instances

Info

1

Software Image (AMI)

Canonical, Ubuntu, 26.04, amd64...[read more](#)

ami-091138d0f0d41ff90

Virtual server type (instance type)

t3.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Cancel

Launch instance

 Preview code

☰ EC2 > Instances > Launch an instance

ⓘ



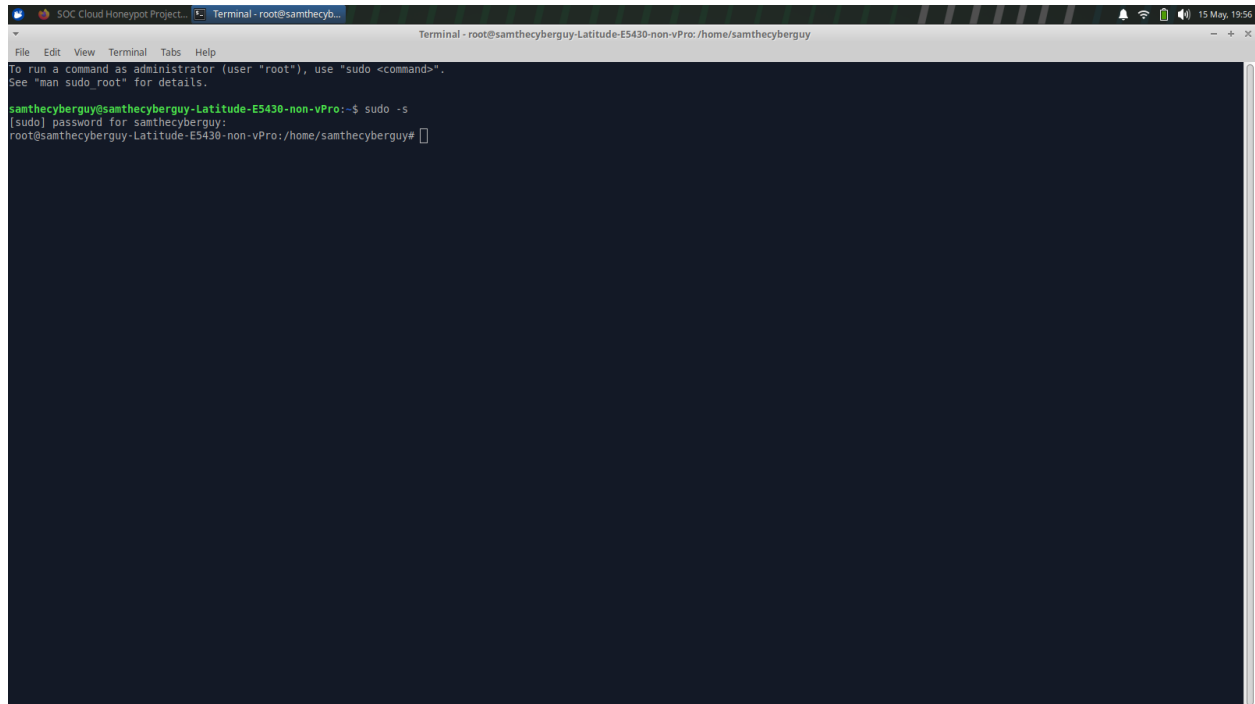
✔ Success

Successfully initiated launch of instance ([i-003da3145268318dd](#))

▶ Launch log

## Step 3 - Connect to EC2 Instance from Xubuntu

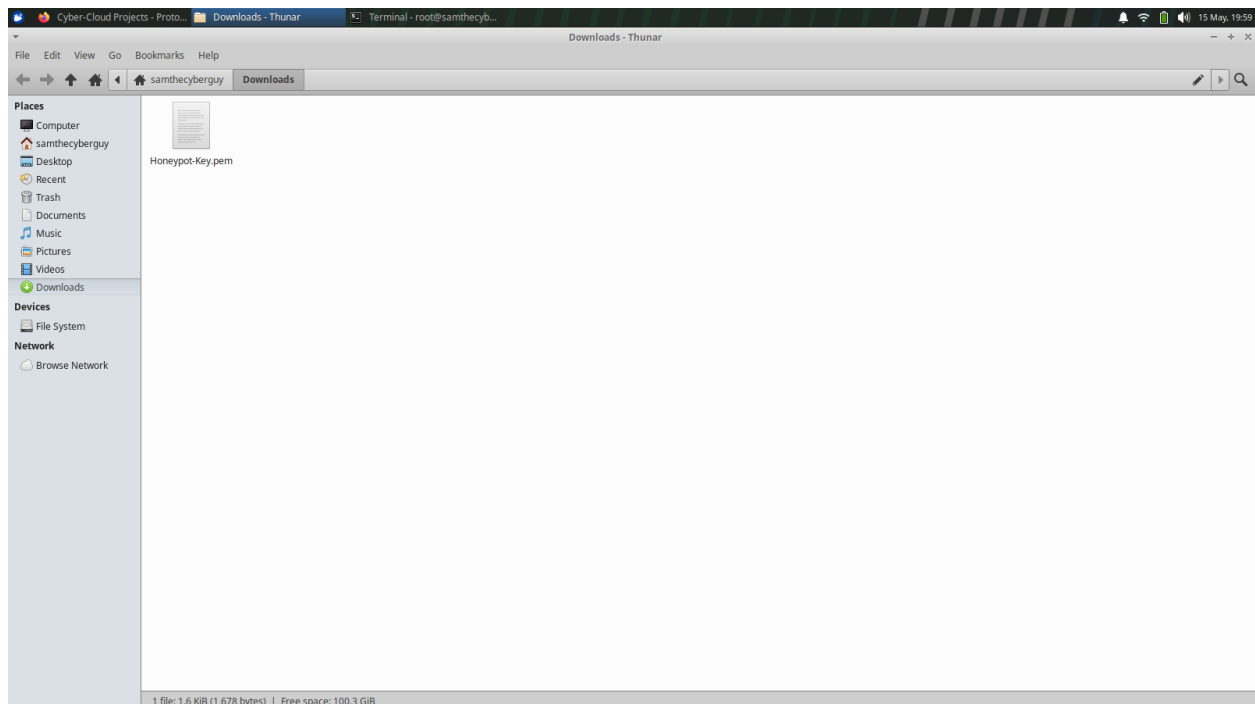
Open the terminal in Xubuntu.



```
Terminal - root@samthecyb...
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ sudo -s
[sudo] password for samthecyberguy:
root@samthecyberguy-Latitude-E5430-non-vPro:~#
```

Locate the PEM key downloaded from AWS.



Create an SSH directory and move the PEM key to it.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ mkdir -p ~/.ssh
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ mv ~/home/samthecyberguy/Downloads/Honeypot-Key.pem ~/.ssh/
```

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ cd ~/.ssh
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~/.ssh$ ls
authorized_keys  Honeypot-Key.pem
```

Set the PEM key file to be readable by only the owner (you). To do this use **chmod**. SSH will reject keys with loose permissions as a security measure.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~/.ssh$ chmod 400 Honeypot-Key.pem
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~/.ssh$
```

Locate the EC2 instance's public ip address.

The screenshot shows the AWS Management Console interface for an EC2 instance named 'i-003da3145268318dd (Cloud-Honeypot)'. The instance is in the 'Running' state. The 'Public IPv4 address' is listed as '54.160.227.94' with a link to 'open address'. The 'Private IPv4 addresses' are listed as '172.16.0.1' and '172.16.0.2'. The 'IPv6 address' is listed as '—'. The 'Public IPv4 address' is circled in red.

Attempt to remote into the EC2 instance via SSH. This may return a **connection timed out** error.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ssh -i ~/.ssh/Honeypot-Key.pem ubuntu@54.160.227.94
ssh: connect to host 54.160.227.94 port 22: Connection timed out
```

After receiving the error, use the **netcat** command to test the connection to the EC2 instance public IP.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ nc -zv 54.160.227.94 22
nc: connect to 54.160.227.94 port 22 (tcp) failed: Connection timed out
```

If the connection times out, check the inbound rule on the EC2 security group. The source IP must match the machine being used to connect, not a private IP from a different machine. Update the inbound rule with the correct public IP and attempt the connection again.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ curl https://api.ipify.org
73. [REDACTED] samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ^C
```

**Edit inbound rules** [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

**Inbound rules** [Info](#)

Inbound rule 1 Delete

|                                                    |                                         |                                      |
|----------------------------------------------------|-----------------------------------------|--------------------------------------|
| <b>Security group rule ID</b> <a href="#">Info</a> | <b>Type</b> <a href="#">Info</a>        | <b>Protocol</b> <a href="#">Info</a> |
| sgr-0be4c7127aab91a83                              | SSH                                     | TCP                                  |
| <b>Port range</b> <a href="#">Info</a>             | <b>Source type</b> <a href="#">Info</a> | <b>Source</b> <a href="#">Info</a>   |
| 22                                                 | Custom                                  | 73. [REDACTED]                       |
| <b>Description - optional</b> <a href="#">Info</a> |                                         |                                      |
| <input type="text"/>                               |                                         |                                      |
| <span>Add rule</span>                              |                                         |                                      |

Cancel Preview changes Save rules

Beyond network connection issues, it is possible to get an SSH handshake error.

```
73. [REDACTED] samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ssh -i ~/.ssh/Honeypot-Key.pem ubuntu@54.160.227.94
The authenticity of host '54.160.227.94 (54.160.227.94)' can't be established.
ED25519 key fingerprint is SHA256:uWYpUJQsqeOQLavwf7ZQA8z1YtRtjfhAgsGdN9pc1IY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.160.227.94' (ED25519) to the list of known hosts.
Connection closed by 54.160.227.94 port 22
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$
```

If there is an SSH handshake error, first confirm that the PEM key on Xubuntu is the same as the PEM key assigned to the EC2 instance. Next, confirm that the AMI is Ubuntu and not another OS. Lastly, add a `-v` to the original command to give it verbose output.

```

samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ssh -v -i ~/.ssh/Honeypot-Key.pem ubuntu@54.160.227.94
OpenSSH 9.6p1 Ubuntu-3ubuntu13.16, OpenSSL 3.0.13 30 Jan 2024
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: include /etc/ssh/ssh_config.d/*.conf matched no files
debug1: /etc/ssh/ssh_config line 21: Applying options for *
debug1: Connecting to 54.160.227.94 [54.160.227.94] port 22.
debug1: Connection established.
debug1: identity file /home/samthecyberguy/.ssh/Honeypot-Key.pem type -1
debug1: identity file /home/samthecyberguy/.ssh/Honeypot-Key.pem-cert type -1
debug1: Local version string SSH-2.0-OpenSSH_9.6p1 Ubuntu-3ubuntu13.16
debug1: Remote protocol version 2.0, remote software version OpenSSH_10.2p1 Ubuntu-2ubuntu3.2
debug1: compat banner: match: OpenSSH_10.2p1 Ubuntu-2ubuntu3.2 pat OpenSSH* compat 0x04000000
debug1: Authenticating to 54.160.227.94:22 as 'ubuntu'
debug1: load hostkeys: fopen /home/samthecyberguy/.ssh/known_hosts2: No such file or directory
debug1: load hostkeys: fopen /etc/ssh/ssh_known_hosts: No such file or directory
debug1: load hostkeys: fopen /etc/ssh/ssh_known_hosts2: No such file or directory
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: sntrup761x25519-sha512@openssh.com
debug1: kex: host key algorithm: ssh-ed25519
debug1: kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: SSH2_MSG_KEX_ECDH_REPLY received

```

...

```

Welcome to Ubuntu 26.04 LTS (GNU/Linux 7.0.0-1004-aws x86_64)

 * Documentation:  https://docs.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sat May 16 19:36:28 UTC 2026

System load:  0.0           Temperature:   -273.1 C
Usage of /:   34.8% of 6.61GB Processes:    114
Memory usage: 29%          Users logged in: 0
Swap usage:   0%           IPv4 address for ens5: 172.17.0.1

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

6 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ubuntu@ip-172-17-0-1:~$

```

## Step 4 - Initial Server Setup

Now that connection from the Xubuntu machine to the AWS EC2 instance has been established via SSH, update packages for the server.

```
ubuntu@ip-172-...:~$ sudo apt update && sudo apt upgrade -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu resolute-security InRelease
6 packages can be upgraded. Run 'apt list --upgradable' to see them.
Upgrading:
  base-files  bpftool  distro-info-data  linux-perf  linux-tools-common  motd-news-config
Summary:
  Upgrading: 6, Installing: 0, Removing: 0, Not Upgrading: 0
  Download size: 8028 kB
  Freed space: 1024 B
```

Create an admin user.

```
ubuntu@ip-172-...:~$ sudo adduser killerbee
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for killerbee
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
ubuntu@ip-172-...:~$
```

Give sudo access to the admin user.

```
ubuntu@ip-172-...:~$ sudo usermod -aG sudo killerbee
ubuntu@ip-172-...:~$
```

## Step 5 - Move Real SSH to Port 2222

As outlined in the introduction, real SSH access will now move from port 22 to port 2222, freeing port 22 for the Cowrie honeypot to intercept attacker traffic.

Start by opening the SSH configuration file. Then add an additional line with **port 2222** and uncomment each port line by removing the # character.

```
ubuntu@ip-172-...:~$ sudo nano /etc/ssh/sshd_config
ubuntu@ip-172-...:~$
```

```
# For changes to take effect, run:
#
#   systemctl daemon-reload
#   systemctl restart ssh.socket
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```

```
# For changes to take effect, run:
#
#   systemctl daemon-reload
#   systemctl restart ssh.socket
#
Port 22
Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```

In the config file, remove **prohibit-password** and add **no** after **PermitRootLogin**. Then, uncomment that line. This makes it so root login is disabled over SSH. The root account has the highest privilege and attackers often target this account, so by disabling it, the risk of root compromise is reduced.

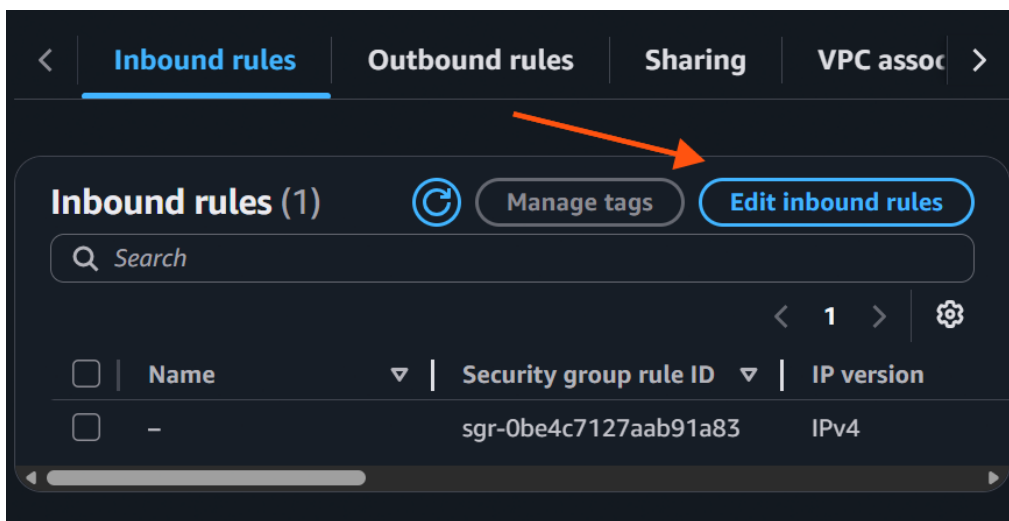
```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Add a new inbound rule allowing SSH from the Xubuntu machine on port 2222.



Inbound rule 2 Delete

|                                                                    |                               |                            |
|--------------------------------------------------------------------|-------------------------------|----------------------------|
| Security group rule ID                                             | Type <span>Info</span>        | Protocol <span>Info</span> |
| -                                                                  | Custom TCP ▼                  | TCP                        |
| Port range <span>Info</span>                                       | Source type <span>Info</span> | Source <span>Info</span>   |
| 2222                                                               | Custom ▼                      | Q 73. [redacted] X         |
|                                                                    |                               | 73. [redacted] 2 X         |
| Description - optional <span>Info</span>                           |                               |                            |
| <div></div>                                                        |                               |                            |
| <div>Add rule</div>                                                |                               |                            |
| <div>Cancel</div> <div>Preview changes</div> <div>Save rules</div> |                               |                            |

Restart SSH.

```
ubuntu@ip-172- [redacted]:~$ sudo systemctl restart ssh
ubuntu@ip-172- [redacted]:~$
```

Open a new terminal window and attempt to SSH into the EC2 instance through port 2222 to test the new change. A connection timed out error might occur.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ssh -v -i ~/.ssh/Honeypot-Key.pem -p 2222 ubuntu@54.160.227.94
OpenSSH 9.6p1 Ubuntu-3ubuntu13.16, OpenSSL 3.0.13 30 Jan 2024
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: include /etc/ssh/ssh_config.d/*.conf matched no files
debug1: /etc/ssh/ssh_config line 21: Applying options for *
debug1: Connecting to 54.160.227.94 [54.160.227.94] port 2222.
debug1: connect to address 54.160.227.94 port 2222: Connection timed out
ssh: connect to host 54.160.227.94 port 2222: Connection timed out
```

Return to the original terminal to check if port 2222 is now listening on SSH. It is possible only port 22 may still be present.

```
ubuntu@ip-172- [redacted]:~$ sudo ss -tulnp | grep ssh
tcp LISTEN 0      4096      0.0.0.0:22      0.0.0.0:*      users:((("sshd",pid=21757,fd=3),("systemd",pid=1,fd=206)))
tcp LISTEN 0      4096      [::]:22       [::]:*        users:((("sshd",pid=21757,fd=4),("systemd",pid=1,fd=207)))
```

Confirm that port 2222 has been added to the SSH config file and uncommented as well.

```
# When systemd socket activation is used (the default), the socket
# configuration must be re-generated after changing Port, AddressFamily, or
# ListenAddress.
#
# For changes to take effect, run:
#
#   systemctl daemon-reload
#   systemctl restart ssh.socket
#
Port 22
Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

If this is the case, disable **ssh.socket**. This feature overrides changes made to the SSH config file by default on Ubuntu.

```
ubuntu@ip-172-██████████:~$ sudo systemctl disable ssh.socket
Removed '/etc/systemd/system/ssh.service.requires/ssh.socket'.
Removed '/etc/systemd/system/sockets.target.wants/ssh.socket'.
ubuntu@ip-172-██████████:~$ sudo systemctl stop ssh.socket
```

Restart SSH again and verify that SSH is now listening on port 2222.

```
ubuntu@ip-172-██████████:~$ sudo systemctl restart ssh
ubuntu@ip-172-██████████:~$ sudo ss -tulnp | grep ssh
tcp    LISTEN 0      128          0.0.0.0:22      0.0.0.0:*      users:(("sshd",pid=22370,fd=8))
tcp    LISTEN 0      128          0.0.0.0:2222   0.0.0.0:*      users:(("sshd",pid=22370,fd=6))
tcp    LISTEN 0      128          [::]:22        [::]:*         users:(("sshd",pid=22370,fd=9))
tcp    LISTEN 0      128          [::]:2222     [::]:*         users:(("sshd",pid=22370,fd=7))
```

For now, leave SSH on port 22 and port 2222. Later, once Cowrie is configured to be on port 22, port 22 will be removed from the SSH config file.

Navigate to the new terminal and attempt to SSH into the EC2 instance through port 2222.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ssh -v -i ~/.ssh/Honeypot-Key.pem -p 2222 ubuntu@54.160.227.94
OpenSSH 9.6p1 Ubuntu-3ubuntu13.16, OpenSSL 3.0.13 30 Jan 2024
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: include /etc/ssh/ssh_config.d/*.conf matched no files
debug1: /etc/ssh/ssh_config line 21: Applying options for *
debug1: Connecting to 54.160.227.94 [54.160.227.94] port 2222.
debug1: Connection established.
```

To finalize the transfer of SSH from port 22 to port 2222, remove port 22 admin access by going into the EC2 instance Security Group settings and editing the inbound rule. The intention is to stop using port 22 for legitimate administration, laying the

groundwork for Cowrie honeypot to use port 22, while port 2222 will be used for legitimate SSH administration.

### Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

#### Inbound rules Info

Inbound rule 1

Delete

Security group rule ID

sgr-0b2d8d8345ea20e3f

Type Info

Custom TCP

Protocol Info

TCP

Port range Info

2222

Source type Info

Custom

Source Info

73. [redacted] X

Description - optional Info

Real Admin SSH

Inbound rule 2

Delete

Security group rule ID

sgr-0be4c7127aab91a83

Type Info

SSH

Protocol Info

TCP

Port range Info

22

Source type Info

Custom

Source Info

73. [redacted] X

Description - optional Info

Honepot SSH

Add rule

Cancel

Preview changes

Save rules

### Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

#### Inbound rules Info

Inbound rule 1

Delete

Security group rule ID

sgr-0b2d8d8345ea20e3f

Type Info

Custom TCP

Protocol Info

TCP

Port range Info

2222

Source type Info

Custom

Source Info

73. [redacted] X

Description - optional Info

Real Admin SSH

Add rule

Cancel

Preview changes

Save rules

## Step 6 - Install Cowrie Honeypot

The goal is to install and configure [Cowrie Honeypot](#) on the AWS Ubuntu server so it can simulate a fake SSH server to capture attacker activity.

To begin, connect to the EC2 instance in the Xubuntu terminal.

```

samthecyberguy@samthecyberguy:~$ ssh -V -i ~/.ssh/Honeypot-Key.pem -p 2222 ubuntu@54.160.227.94
OpenSSH 9.6p1 Ubuntu-3ubuntu13.16, OpenSSL 3.0.13 30 Jan 2024
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: include /etc/ssh/ssh_config.d/*.conf matched no files
debug1: /etc/ssh/ssh_config line 21: Applying options for *
debug1: Connecting to 54.160.227.94 [54.160.227.94] port 2222.
debug1: Connection established.

```

Update packages and install dependencies to prepare for the Cowrie installation.

```
ubuntu@ip-172-20-10-10:~$ sudo apt update && sudo apt upgrade -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute-updates InRelease [136 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu resolute-security InRelease [136 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute-updates/main amd64 Components [2844 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu resolute-updates/universe amd64 Components [20.6 kB]
Get:7 http://security.ubuntu.com/ubuntu resolute-security/main amd64 Components [2844 B]
Get:8 http://security.ubuntu.com/ubuntu resolute-security/universe amd64 Components [20.6 kB]
Fetched 318 kB in 0s (1841 kB/s)
All packages are up to date.
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
```

```

buntu@ip-172-31-10-100:~$ sudo apt install -y \
git \
python3-venv \
python3-dev \
libssl-dev \
libffi-dev \
build-essential \
libpython3-dev \
authbind \
virtualenv
git is already the newest version (1:2.53.0-1ubuntu1).
git set to manually installed.
Installing:
authbind build-essential libffi-dev libpython3-dev libssl-dev python3-dev python3-venv virtualenv

Installing dependencies:
bzzip2 g++ gcc-15-x86-64-linux-gnu libc6-dev libgomp1 libquadmath0 manpages-dev python3.14-dev
cpp g++-15 gcc-x86-64-linux-gnu libc6-dev libltdc++15-dev python3-distlib python3.14-venv
cpp-15 g++-15-x86-64-linux-gnu libalgorithm-diff-perl libldpkg-perl libisl23 libtsan2 rpcsvc-proto
gcc-15-x86-64-linux-gnu g++-x86-64-linux-gnu libalgorithm-diff-xs-perl libbepkg-dev libitm1 zlib-dev
cpp-x86-64-linux-gnu gcc libalgorithm-merge-perl libfakeroot liblsan0 linux-libc-dev python3-platformdirs
dpkg-dev gcc-15 libasan8 libelf1-dev libflock-perl libmpc3 python3-setup-tools-whl
fakeroot gcc-15-base libc-dev-bin libgcc-15-dev libpython3.14-dev make python3-virtualenv

Suggested packages:
bzzip2-doc gcc-15-locales debiana-keyring g++-multilib gcc-15-doc autoconf libtool bison gcc-doc
cpp-doc cpp-15-doc debiana-tag2upload-keyring g++-15-multilib gcc-multilib automake flex gdb gcc-15-multilib
gdb-x86-64-linux-gnu glibc-doc libssl-doc libstdc++-15-doc
make-doc

Summary:
Upgrading: 0, Installing: 61, Removing: 0, Not Upgrading: 0
Download size: 93.6 MB
Space needed: 329 MB / 4608 MB available

```

Since Cowrie will be exposed to attackers, create a Cowrie user without root privileges. Using a basic user account is a security measure that can prevent potential root access by an attacker and a compromise to the EC2 instance.

After creating the Cowrie account, login to it.

```

ubuntu@ip-172-██████████:~$ sudo adduser --disabled-password cowrie
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
ubuntu@ip-172-██████████:~$ sudo su - cowrie
cowrie@ip-172-██████████:~$

```

Clone the Cowrie repository from GitHub.

```

cowrie@ip-172-██████████:~$ git clone http://github.com/cowrie/cowrie
Cloning into 'cowrie'...
warning: redirecting to https://github.com/cowrie/cowrie/
remote: Enumerating objects: 22049, done.
remote: Counting objects: 100% (644/644), done.
remote: Compressing objects: 100% (381/381), done.
remote: Total 22049 (delta 433), reused 268 (delta 263), pack-reused 21405 (from 3)
Receiving objects: 100% (22049/22049), 11.93 MiB | 28.95 MiB/s, done.
Resolving deltas: 100% (15191/15191), done.
cowrie@ip-172-██████████:~$

```

Enter the Cowrie directory.

```

cowrie@ip-172-██████████:~$ cd cowrie
cowrie@ip-172-██████████:~/cowrie$

```

Create a python virtual environment and then enter the environment so Cowrie uses its own python packages and not system wide packages, as that could cause accidental conflicts.

```

cowrie@ip-172-██████████:~/cowrie$ python3 -m venv cowrie-env
cowrie@ip-172-██████████:~/cowrie$ source cowrie-env/bin/activate
(cowrie-env) cowrie@ip-172-██████████:~/cowrie$ ^C
(cowrie-env) cowrie@ip-172-██████████:~/cowrie$

```

Install Cowrie requirements, dependencies, and Cowrie as an editable python package with the launcher executable.

```
(cowrie-env) cowrie@ip-172-16-17-10:~/cowrie$ pip install --upgrade pip
Requirement already satisfied: pip in ./cowrie-env/lib/python3.14/site-packages (25.1.1)
Collecting pip
  Downloading pip-26.1.1-py3-none-any.whl.metadata (4.6 kB)
  Downloading pip-26.1.1-py3-none-any.whl (1.8 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 42.3 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 25.1.1
    Uninstalling pip-25.1.1:
      Successfully uninstalled pip-25.1.1
Successfully installed pip-26.1.1
```

```
(cowrie-env) cowrie@ip-172-16-17-10:~/cowrie$ pip install --upgrade -r requirements.txt
WARNING: Cache entry deserialization failed, entry ignored
Collecting attrs==26.1.0 (from -r requirements.txt (line 5))
  Downloading attrs-26.1.0-py3-none-any.whl.metadata (8.8 kB)
Collecting bcrypt==5.0.0 (from -r requirements.txt (line 6))
  Downloading bcrypt-5.0.0-cp39-abi3-manylinux_2_34_x86_64.whl.metadata (10 kB)
Collecting cryptography==48.0.0 (from -r requirements.txt (line 7))
  Downloading cryptography-48.0.0-cp311-abi3-manylinux_2_34_x86_64.whl.metadata (4.3 kB)
Collecting hyperlink==21.0.0 (from -r requirements.txt (line 8))
  Downloading hyperlink-21.0.0-py2.py3-none-any.whl.metadata (1.5 kB)
```

...

```
Downloading treq-25.5.0-py3-none-any.whl (77 kB)
Downloading twisted-26.4.0-py3-none-any.whl (3.2 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.2/3.2 MB 143.2 MB/s 0:00:00
Downloading charset-normalizer-3.4.7-cp314-cp314-manylinux2014_x86_64.manylinux_2_17_x86_64.manylinux_2_28_x86_64.whl (215 kB)
Downloading pyasn1-0.6.3-py3-none-any.whl (83 kB)
Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Downloading automat-25.4.16-py3-none-any.whl (42 kB)
Downloading certifi-2026.4.22-py3-none-any.whl (135 kB)
Downloading cffi-2.0.0-cp314-cp314-manylinux2014_x86_64.manylinux_2_17_x86_64.whl (219 kB)
Downloading constantly-23.10.4-py3-none-any.whl (13 kB)
Downloading incremental-24.11.0-py3-none-any.whl (21 kB)
Downloading pyopenssl-26.2.0-py3-none-any.whl (55 kB)
Downloading typing_extensions-4.15.0-py3-none-any.whl (44 kB)
Downloading zope.interface-8.4-cp314-cp314-manylinux1_x86_64.manylinux2014_x86_64.manylinux_2_17_x86_64.manylinux_2_5_x86_64.whl (269 kB)
Downloading multipart-1.3.1-py3-none-any.whl (15 kB)
Downloading pycparser-3.0-py3-none-any.whl (48 kB)
Installing collected packages: appdirs, zope.interface, urllib3, typing-extensions, tftpy, pycparser, pyasn1, packaging, multipart, idna, constantly,
ttrs, requests, pyasn1_modules, incremental, hyperlink, cffi, twisted, cryptography, service_identity, pyopenssl, treq
Successfully installed appdirs-1.4.4 attrs-26.1.0 automat-25.4.16 bcrypt-5.0.0 certifi-2026.4.22 cffi-2.0.0 charset-normalizer-3.4.7 constantly-23.10.4
incremental-24.11.0 multipart-1.3.1 packaging-26.2 pyasn1-0.6.3 pyasn1_modules-0.4.2 pycparser-3.0 pyopenssl-26.2.0 requests-2.34.0 service_identity-
typing-extensions-4.15.0 urllib3-2.7.0 zope.interface-8.4
(cowrie-env) cowrie@ip-172-16-17-10:~/cowrie$
```

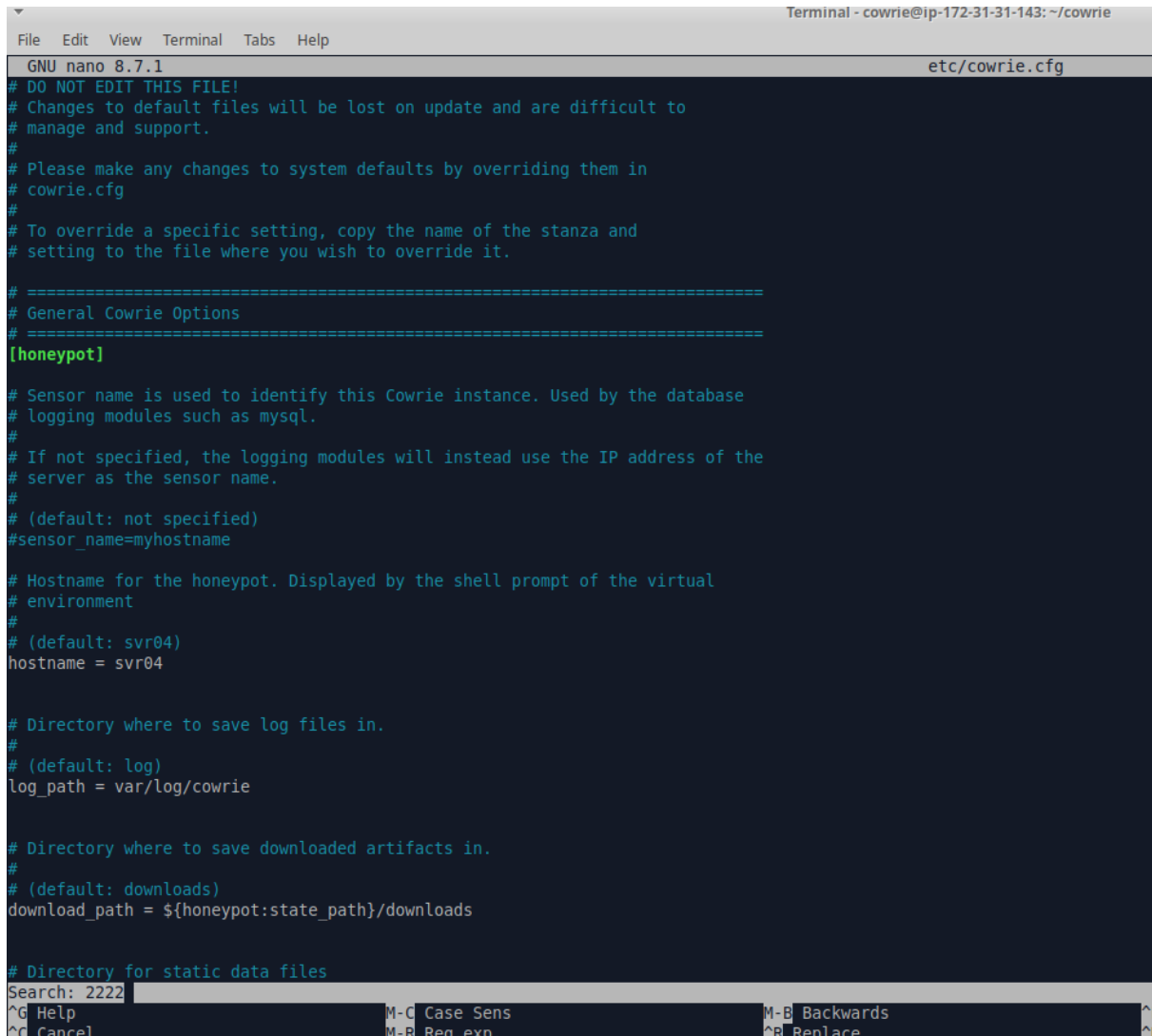
```
(cowrie-env) cowrie@ip-172-16-17-10:~/cowrie$ pip install -e .
Obtaining file:///home/cowrie/cowrie
Installing build dependencies ... done
Checking if build backend supports build editable ... done
Getting requirements to build editable ... done
Preparing editable metadata (pyproject.toml) ... done
Requirement already satisfied: attrs==26.1.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (26.1.0)
Requirement already satisfied: bcrypt==5.0.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (5.0.0)
Requirement already satisfied: cryptography==48.0.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (48.0.0)
Requirement already satisfied: hyperlink==21.0.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (21.0.0)
Requirement already satisfied: idna==3.14 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (3.14)
Requirement already satisfied: packaging==26.2 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (26.2)
Requirement already satisfied: pyasn1_modules==0.4.2 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (0.4.2)
Requirement already satisfied: requests==2.34.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (2.34.0)
Requirement already satisfied: service_identity==24.2.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (24.2.0)
Requirement already satisfied: tftpy==0.8.7 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (0.8.7)
Requirement already satisfied: treq==25.5.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (25.5.0)
Requirement already satisfied: twisted==26.4.0 in ./cowrie-env/lib/python3.14/site-packages (from twisted[conch]==26.4.0->cowrie==2.9.19) (26.4.0)
Requirement already satisfied: urllib3==2.7.0 in ./cowrie-env/lib/python3.14/site-packages (from cowrie==2.9.19) (2.7.0)
Requirement already satisfied: cffi==2.0.0 in ./cowrie-env/lib/python3.14/site-packages (from cryptography==48.0.0->cowrie==2.9.19) (2.0.0)
Requirement already satisfied: pyasn1<0.7.0, >=0.6.1 in ./cowrie-env/lib/python3.14/site-packages (from pyasn1_modules==0.4.2->cowrie==2.9.19) (0.6.3)
Requirement already satisfied: charset-normalizer<4, >=2 in ./cowrie-env/lib/python3.14/site-packages (from requests==2.34.0->cowrie==2.9.19) (2.0.0)
Requirement already satisfied: certifi==2023.5.7 in ./cowrie-env/lib/python3.14/site-packages (from requests==2.34.0->cowrie==2.9.19) (2026.4.22)
Requirement already satisfied: incremental==24.7.2 in ./cowrie-env/lib/python3.14/site-packages (from treq==25.5.0->cowrie==2.9.19) (24.11.0)
Requirement already satisfied: multipart in ./cowrie-env/lib/python3.14/site-packages (from treq==25.5.0->cowrie==2.9.19) (1.3.1)
Requirement already satisfied: typing_extensions>=3.10.0 in ./cowrie-env/lib/python3.14/site-packages (from twisted==26.4.0->cowrie==2.9.19) (4.15.0)
Requirement already satisfied: automat==24.8.0 in ./cowrie-env/lib/python3.14/site-packages (from twisted==26.4.0->twisted[conch]==26.4.0->cowrie==2.9.19) (25.4.16)
Requirement already satisfied: constantly==15.1 in ./cowrie-env/lib/python3.14/site-packages (from twisted==26.4.0->twisted[conch]==26.4.0->cowrie==2.9.19) (23.10.4)
Requirement already satisfied: zope.interface==5 in ./cowrie-env/lib/python3.14/site-packages (from twisted==26.4.0->twisted[conch]==26.4.0->cowrie==2.9.19) (8.4)
Requirement already satisfied: appdirs>=1.4.0 in ./cowrie-env/lib/python3.14/site-packages (from twisted[conch]==26.4.0->cowrie==2.9.19) (1.4.4)
Requirement already satisfied: pycparser in ./cowrie-env/lib/python3.14/site-packages (from cffi==2.0.0->cryptography==48.0.0->cowrie==2.9.19) (3.0)
Requirement already satisfied: pyopenssl==25.2.0 in ./cowrie-env/lib/python3.14/site-packages (from twisted[tls]==22.10.0->treq==25.5.0->cowrie==2.9.19) (26.2.0)
Building wheels for collected packages: cowrie
  Building editable for cowrie (pyproject.toml) ... done
  Created wheel for cowrie: filename=cowrie-2.9.19-0.editable-py3-none-any.whl size=5973 sha256=9e5c6d89ec16ed2190471c43f7990445e3a3bfa687d874b20a30a1ac847b5d4
  Stored in directory: /tmp/pip-ephem-wheel-cache-vhognw4/wheels/1f/6f/90/f26429e723aad6d0b0eadc48ac2d1b95ce70c68a199165b764
Successfully built cowrie
Installing collected packages: cowrie
Successfully installed cowrie-2.9.19
```

## Step 7 - Configure Cowrie

Now, it is time to configure Cowrie so it is assigned to port 22.

Copy this template for the Cowrie config file and then open the file to edit it.

```
cowrie-env) cowrie@ip-172-31-31-143: ~/cowrie$ cp etc/cowrie.cfg.dist etc/cowrie.cfg
cowrie-env) cowrie@ip-172-31-31-143: ~/cowrie$ nano etc/cowrie.cfg
```



```
GNU nano 8.7.1 etc/cowrie.cfg
# DO NOT EDIT THIS FILE!
# Changes to default files will be lost on update and are difficult to
# manage and support.
#
# Please make any changes to system defaults by overriding them in
# cowrie.cfg
#
# To override a specific setting, copy the name of the stanza and
# setting to the file where you wish to override it.
#
# =====
# General Cowrie Options
# =====
[honeypot]
# Sensor name is used to identify this Cowrie instance. Used by the database
# logging modules such as mysql.
#
# If not specified, the logging modules will instead use the IP address of the
# server as the sensor name.
#
# (default: not specified)
#sensor_name=myhostname
#
# Hostname for the honeypot. Displayed by the shell prompt of the virtual
# environment
#
# (default: svr04)
hostname = svr04
#
# Directory where to save log files in.
#
# (default: log)
log_path = var/log/cowrie
#
# Directory where to save downloaded artifacts in.
#
# (default: downloads)
download_path = ${honeypot:state_path}/downloads
#
# Directory for static data files
Search: 2222
^G Help      M-C Case Sens  M-B Backwards  ^F
^C Cancel    M-R Reg. exp   ^R Replace      ^N
```

Press **Ctrl + W** and search for **2222**.

```
# Directory for static data files
Search: 2222
^G Help M-C Case Sens
^C Cancel M-R Reg.exp.
```

Locate the line `listen_endpoints = tcp:2222:interface=0.0.0.0`.

By default Cowrie has SSH on port 2222. In the file, replace **2222** with **22** (Cowrie's SSH port). Then save it and exit the file.

```
# use authbind for port numbers under 1024

listen_endpoints = tcp:2222:interface=0.0.0.0

# Enable the SFTP subsystem
# (default: true)
```

```
# use authbind for port numbers under 1024

listen_endpoints = tcp:22:interface=0.0.0.0

# Enable the SFTP subsystem
# (default: true)
```

By default, Linux blocks non-root users from using ports below 1024, but earlier Cowrie was set up to run as a non-root user for security purposes.

In order to bypass this, configure **authbind**, which was installed earlier, to allow Cowrie to use port 22.

```
(cowrie-env) cowrie@ip-172-██████:~/cowrie$ sudo touch /etc/authbind/byport/22
sudo: I'm sorry cowrie. I'm afraid I can't do that
```

```
(cowrie-env) cowrie@ip-172-██████:~/cowrie$ exit
logout
ubuntu@ip-172-██████:~$
```

```
ubuntu@ip-172-██████:~$ sudo touch /etc/authbind/byport/22
ubuntu@ip-172-██████:~$ sudo chown cowrie:cowrie /etc/authbind/byport/22
ubuntu@ip-172-██████:~$ sudo chmod 500 /etc/authbind/byport/22
ubuntu@ip-172-██████:~$
```

Within the Cowrie folder and the python virtual environment, start up Cowrie and check the status to make sure it is running.

```
(cowrie-env) cowrie@ip-172-...:~/cowrie$ cowrie start
Join the Cowrie community at: https://www.cowrie.org/slack/
Starting cowrie: [twistd --umask=0022 --pidfile /home/cowrie/cowrie/var/run/cowrie.pid --logger cowrie.python.logfile.logger cowrie]...
(cowrie-env) cowrie@ip-172-...:~/cowrie$ cowrie status
cowrie is running (PID: 32910).
(cowrie-env) cowrie@ip-172-...:~/cowrie$
```

Attempt to see if Cowrie is listening on port 22. It is possible to receive an access error, which is actually expected because earlier Cowrie was set as a non-root user.

```
(cowrie-env) cowrie@ip-172-...:~/cowrie$ sudo ss -tulnp | grep 2222
sudo: I'm sorry cowrie. I'm afraid I can't do that
(cowrie-env) cowrie@ip-172-...:~/cowrie$ exit
logout
```

SSH is still on both port 22 and port 2222. Now that Cowrie has been set up on port 22, remove SSH from port 22 so that it is only on port 2222.

```
ubuntu@ip-172-...:~$ sudo ss -tulnp | grep 22
tcp LISTEN 0      128          0.0.0.0:22      0.0.0.0:*      users: (("sshd",pid=22370,fd=8))
tcp LISTEN 0      128          0.0.0.0:2222    0.0.0.0:*      users: (("sshd",pid=22370,fd=6))
tcp LISTEN 0      128          [::]:22        [::]:*         users: (("sshd",pid=22370,fd=9))
tcp LISTEN 0      128          [::]:2222      [::]:*         users: (("sshd",pid=22370,fd=7))
ubuntu@ip-172-...:~$
```

To do that, start by opening the SSH config file and delete **port 22** from it.

```
#
Port 22
Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

```
#
Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

Restart SSH.

```
ubuntu@ip-172-...:~$ sudo systemctl restart ssh
ubuntu@ip-172-...:~$
```

Confirm that SSH is no longer listening on port 22 and instead only on port 2222.

```
ubuntu@ip-172-...:~$ sudo ss -tulnp | grep 22
tcp LISTEN 0      128                0.0.0.0:2222      0.0.0.0:*        users:(("sshd",pid=33058,fd=6))
tcp LISTEN 0      128                [::]:2222        [::]:*          users:(("sshd",pid=33058,fd=7))
ubuntu@ip-172-...:~$
```

Log into the Cowrie user and open the Cowrie config file, confirming that only port 22 is present.

```
# For both IPV4 and IPV6: listen_endpoints = tcp6:2222:interface=0.0.0.0
# Listening on multiple endpoints is supported with a string
# e.g listen_endpoints = "tcp:2222:interface=0.0.0.0 tcp6:2222:interface=:::0"
# use authbind for port numbers under 1024

listen_endpoints = tcp:22:interface=0.0.0.0

# Enable the SFTP subsystem
# (default: true)
sftp_enabled = true
```

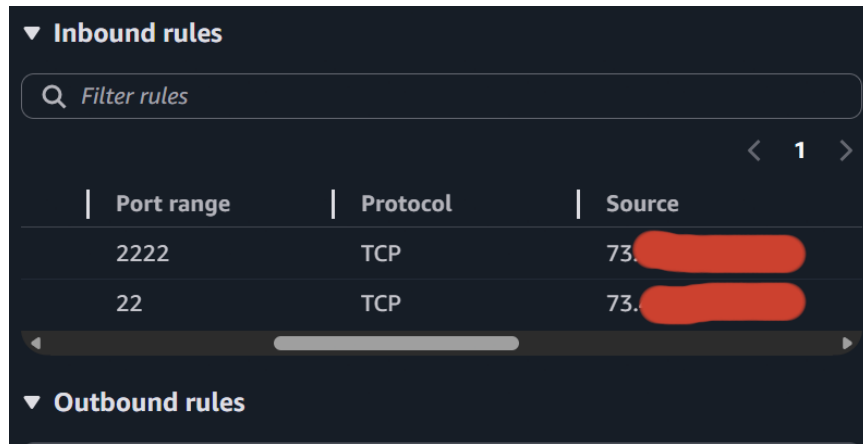
Enter Cowrie python virtual environment and stop Cowrie. Then start Cowrie to initiate the change.

```
cowrie@ip-172-...:~/cowrie$ source cowrie-env/bin/activate
(cowrie-env) cowrie@ip-172-...:~/cowrie$ cowrie stop
Stopping cowrie...
(cowrie-env) cowrie@ip-172-...:~/cowrie$ cowrie start
Starting cowrie: [twistd --umask=0022 --pidfile /home/cowrie/cowrie/var/run/cowrie.pid --logger cowrie.python.logger.logger cowrie]...
```

Log out of the Cowrie user and confirm that port 22 is assigned to Cowrie and port 2222 to SSH. The user of port 22 should be **twistd**, which is the framework which Cowrie runs on.

```
(cowrie-env) cowrie@ip-172-...:~/cowrie$ exit
logout
ubuntu@ip-172-...:~$ sudo ss -tulnp | grep 22
tcp LISTEN 0      50                0.0.0.0:22       0.0.0.0:*        users:(("twistd",pid=33243,fd=12))
tcp LISTEN 0      128                0.0.0.0:2222     0.0.0.0:*        users:(("sshd",pid=33058,fd=6))
tcp LISTEN 0      128                [::]:2222        [::]:*          users:(("sshd",pid=33058,fd=7))
```

The current EC2 security group had the source IP for the inbound rule for port 22 set to the Xubuntu public IP. In order to allow attacks to begin, set the source IP for that inbound rule to **0.0.0.0/0** (anywhere). This would be unsafe in a normal SSH server, but since the real SSH admin server was transferred to port 2222 and the Cowrie honeypot was configured with non root access to port 22, it is safe. Port 22 has been properly isolated and secured.



Inbound rule 2 Delete

Security group rule ID: sgr-0be4c7127aab91a83

Type: SSH

Protocol: TCP

Port range: 22

Source type: Anywhere-IPv4

Source: 0.0.0.0/0

Description - optional: Honepot SSH

Add rule

Open a fresh terminal window and attempt to SSH into the EC2 instance via port 22. If properly configured, it should return a **Host Key Verification Failed** error.

```
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$ ssh -p 22 ubuntu@54.160.227.94
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ED25519 key sent by the remote host is
SHA256:CwZjKmCi6jd+ZKg2UNPGv+zBlWomqalfeJqWRAj4gs4.
Please contact your system administrator.
Add correct host key in /home/samthecyberguy/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/samthecyberguy/.ssh/known_hosts:3
  remove with:
    ssh-keygen -f '/home/samthecyberguy/.ssh/known_hosts' -R '54.160.227.94'
Host key for 54.160.227.94 has changed and you have requested strict checking.
Host key verification failed.
samthecyberguy@samthecyberguy-Latitude-E5430-non-vPro:~$
```

Return back to the original terminal, logged in as the Cowrie user, and check the attack log.

The SSH connection attempt that was just made will be reflected in the log. The IP address of the attacker (you testing it in this case) will be visible.

```
cowrie@ip-172-██████████:~/cowrie/var/log/cowrie$ nano cowrie.log
```

```
2026-05-21T05:00:38.817731Z [cowrie.ssh.factory.CowrieSSHFactory] New connection: 73.██████████ (172.30.███ [session: a13edb8ebc7e]
2026-05-21T05:00:38.818290Z [HoneyPotSSHTransport,3,73.██████████] Remote SSH version: SSH-2.0-OpenSSH 9.6p1 Ubuntu-3ubuntu13.16
2026-05-21T05:00:38.902018Z [HoneyPotSSHTransport,3,73.██████████] SSH client hassh fingerprint: aae6b9604f6f3356543709a376d7f657
2026-05-21T05:00:38.902932Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] kex alg=b'curve25519-sha256' key alg=b'ssh-ed25519'
2026-05-21T05:00:38.903008Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] outgoing: b'aes128-ctr' b'hmac-sha2-256' b'none'
2026-05-21T05:00:38.903068Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] incoming: b'aes128-ctr' b'hmac-sha2-256' b'none'
2026-05-21T05:00:39.055313Z [cowrie.ssh.transport.HoneyPotSSHTransport#info] connection lost
2026-05-21T05:00:39.055500Z [HoneyPotSSHTransport,3,73.██████████] Connection lost after 0.2 seconds
```

Cowrie honeypot is now set up and running as intended.

## Step 8 - Cowrie Log Analysis

Leave Cowrie and the EC2 instance running for a day.

Looking at the logs captured below, it appears that attacker interaction with the honeypot began 2 hours after turning on the honeypot for the first time. From that moment until the end of the 24 hour period, there were hundreds of interactions from various IPs using various commands.

This is an example of one attack session from the logs.

What we see below is an automated SSH brute force scanner or bot attempting to authenticate to the honeypot using default admin credentials.

```
2026-05-21T07:52:57.398425Z [cowrie.ssh.factory.CowrieSSHFactory] New connection: 34.78.205.101:21718 (172.20.0.1:22) [session: fd8ccc448e9]
2026-05-21T07:52:57.479950Z [HoneyPotSSHTransport,6,34.78.205.101] Remote SSH version: SSH-2.0-FingerprintX-SSH2
2026-05-21T07:52:57.641949Z [cowrie.ssh.factory.CowrieSSHFactory] New connection: 34.78.205.101:21726 (172.20.0.1:22) [session: e1a0bc66a156]
2026-05-21T07:52:57.642484Z [HoneyPotSSHTransport,7,34.78.205.101] Remote SSH version: SSH-2.0-Go
2026-05-21T07:52:57.723610Z [HoneyPotSSHTransport,7,34.78.205.101] SSH client hassh fingerprint: dde267e50f82fcc16a7e1e7b59b8af71
2026-05-21T07:52:57.724525Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] kex alg=b'diffie-hellman-group-exchange-sha256' key alg=b'ecdsa-sha2-nistp256'
2026-05-21T07:52:57.724628Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] outgoing: b'aes128-ctr' b'hmac-sha2-256' b'none'
2026-05-21T07:52:57.724684Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] incoming: b'aes128-ctr' b'hmac-sha2-256' b'none'
2026-05-21T07:52:58.425047Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] NEW KEYS
2026-05-21T07:52:58.425450Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] starting service b'ssh-userauth'
2026-05-21T07:52:58.509763Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'admin' trying auth b'none'
2026-05-21T07:52:58.590842Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'admin' trying auth b'password'
2026-05-21T07:52:58.591190Z [HoneyPotSSHTransport,7,34.78.205.101] Could not read etc/userdb.txt, default database activated
2026-05-21T07:52:58.591507Z [HoneyPotSSHTransport,7,34.78.205.101] login attempt [b'admin'/b'admin'] failed
2026-05-21T07:52:59.593222Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'admin' failed auth b'password'
2026-05-21T07:52:59.593441Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] unauthorized login: ()
2026-05-21T07:52:59.674642Z [cowrie.ssh.transport.HoneyPotSSHTransport#info] connection lost
2026-05-21T07:52:59.674838Z [HoneyPotSSHTransport,7,34.78.205.101] Connection lost after 2.0 seconds
2026-05-21T07:52:59.675278Z [HoneyPotSSHTransport,6,34.78.205.101] SSH client hassh fingerprint: 4e066189c3bbeec38c99b1855113733a
2026-05-21T07:52:59.675958Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] kex alg=b'curve25519-sha256' key alg=b'ecdsa-sha2-nistp256'
2026-05-21T07:52:59.676027Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] outgoing: b'aes128-ctr' b'hmac-sha2-256' b'none'
2026-05-21T07:52:59.676081Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] incoming: b'aes128-ctr' b'hmac-sha2-256' b'none'
```

The attacker's IP address is 34.78.205.101. After looking it up in [AbuseIPDB](#), it has been reported 127 times and is confirmed to be a malicious IP that is known for brute forcing into SSH servers.

```
New connection: 34.78.205.101:21726
```

AbuseIPDB » [34.78.205.101](#)

Check an IP Address, Domain Name, Subnet, or ASN  
e.g. 146.70.230.86, microsoft.com, 5.188.10.0/24, or AS15169

34.78.205.101 CHECK

**34.78.205.101 was found in our database!**

This IP was reported 127 times. Confidence of Abuse is 100%: ?

100%

|             |                                        |
|-------------|----------------------------------------|
| ISP         | Google LLC                             |
| Usage Type  | Data Center/Web Hosting/Transit        |
| ASN         | <a href="#">AS396982</a>               |
| Hostname(s) | 101.205.78.34.bc.googleusercontent.com |
| Domain Name | google.com                             |
| Country     | Belgium                                |
| City        | Brussels, Brussels Capital             |

IP info including ISP, Usage Type, and Location provided by [IPInfo](#). Updated weekly.

|                                    |                                       |                                                                                                                                      |                                              |
|------------------------------------|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| ✓ <a href="#">tecnicorioja</a>     | 2026-05-21 10:00:29<br>(1 week ago)   | Failed password for invalid user May 21 08:43:57 port 32478                                                                          | <span>Brute-Force</span><br><span>SSH</span> |
| ✓ <a href="#">COMPLEX</a>          | 2026-05-21 08:04:54 ⚠<br>(1 week ago) | Banned by Multi Agent · node ...bnl2 · attempts=5 · SSH brute-force / scan                                                           | <span>Brute-Force</span><br><span>SSH</span> |
| ✓ <a href="#">ras07</a>            | 2026-05-21 08:00:09<br>(1 week ago)   | Brute force SSH login attempts.                                                                                                      | <span>Brute-Force</span><br><span>SSH</span> |
| ✓ <a href="#">Melua</a>            | 2026-05-21 08:00:03<br>(1 week ago)   | Attempted connection to SSH tarpit                                                                                                   | <span>Brute-Force</span><br><span>SSH</span> |
| ✓ <a href="#">securityyc3a.com</a> | 2026-05-21 07:58:09<br>(1 week ago)   | 2026-05-21T08:57:51.617372+01:00 mail6 sshd-session[1590108]: refused connect from 34.78.205.101 (34 ... <a href="#">show more</a> ) | <span>Brute-Force</span><br><span>SSH</span> |
| <a href="#">formality</a>          | 2026-05-21 07:37:07 ⚠<br>(1 week ago) | Invalid user admin from 34.78.205.101 port 26982                                                                                     | <span>Brute-Force</span><br><span>SSH</span> |

The SSH client revealed itself and shows the tools the attacker used to scan the honeypot. **SSH-2.0-Go** indicates that the tool was written in Go, which is common for automated scanners and brute force tools. **SSH-2.0-Fingerprintx-SSH2** indicates the attacker may be using the tool [Fingerprintx](#) which checks what kind of SSH service the target is running.

```
Remote SSH version: SSH-2.0-Fingerprintx-SSH2
New connection: 34.78.205.101:21726 (172.
Remote SSH version: SSH-2.0-Go
```

Cowrie captured the HASSH, which is like the digital fingerprint for SSH clients. This can be used to correlate this session with similar SSH scanning or brute force activity across other honeypot logs.

```
SSH client hassh fingerprint: dde267e50f82fcc16a7e1e7b59b8af71
```

The attacker used default credentials to attempt to login, but Cowrie denied access.

```
starting service b'ssh-userauth'  
[debug] b'admin' trying auth b'none'  
[debug] b'admin' trying auth b'password'  
read etc/userdb.txt, default database activated  
empty [b'admin'/b'admin'] failed  
[debug] b'admin' failed auth b'password'  
[debug] unauthorized login: ()
```

The connection timed out in a matter of seconds, which indicates this is not a hands-on attack but rather an automatic scanner.

```
port#info] connection lost  
Connection lost after 2.0 seconds
```

## Conclusion

Phase 1 successfully established a cloud-based Cowrie honeypot on an AWS EC2 instance. The project separated legitimate SSH administration from honeypot activity by moving the real SSH service to port 2222 and placing Cowrie on port 22, the default SSH port commonly targeted by attackers.

After the honeypot was exposed to the internet, it began receiving attacker activity within a short period of time. Cowrie captured useful security data, including source IP addresses, SSH banners, HASSH fingerprints, login attempts, and failed authentication activity. This confirmed that the honeypot was working as intended and demonstrated how quickly exposed cloud services can attract automated scanning and brute-force attempts.

The next phase of this project will focus on organizing, parsing, and analyzing Cowrie logs in more depth so the captured activity can be used for threat intelligence, attacker behavior analysis, and future SIEM integration.