Project Honeypot & Canary Token Monitoring

1. Project Introduction

In this project, I worked on deploying a Cowrie SSH honeypot and creating a Canary Token (Excel-based) to detect and analyse unauthorized access. My objective was to set up deception tools, capture attacker interactions, and validate alerts that could later be used for threat intelligence.

What I learned:

- How honeypots and Canary Tokens are used in deception technology.
- Setting up Cowrie on Kali Linux and running it as a honeypot.
- Creating and testing a Canary Token to generate alerts.
- Collecting and analyzing logs and attacker behavior.

---

2. Environment & Tools I Used

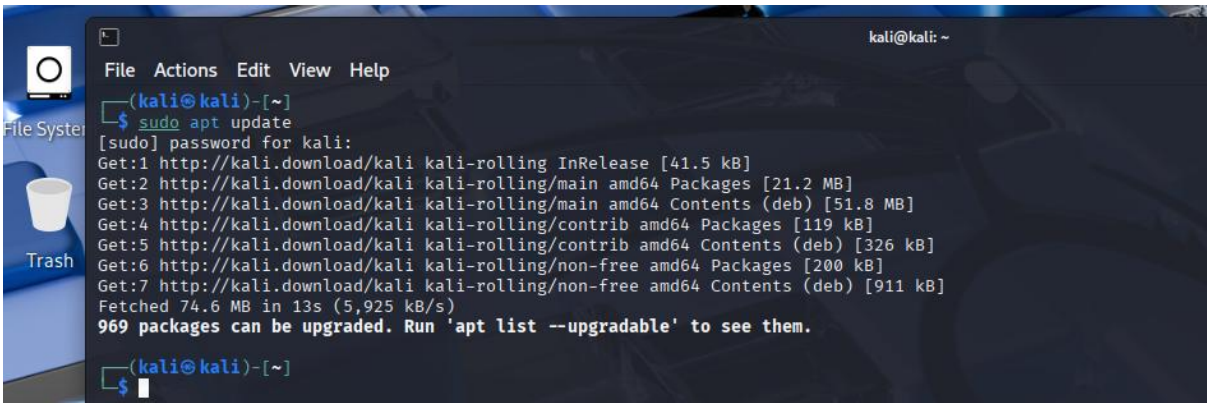
- **Host OS / VM platform:** (Kali Linux in VMware / VirtualBox / Cloud – specify exact setup)
- **Honeypot:** Cowrie (GitHub repo: https://github.com/cowrie/cowrie)
- **Canary Tokens:** Canarytokens.org (Excel token)
- **Utilities & packages:** git, python3, virtualenv, pip, netstat, ssh, tail, nano
- **Email for Canary Token alerts:** (my chosen email address for testing)

3. Steps I Took

1. Started a Kali Linux VM.
2. Installed the required dependencies and tools.
3. Cloned the Cowrie repository.
4. Created and activated a Python virtual environment.
5. Installed Cowrie dependencies.
6. Configured the Cowrie settings file.
7. Started Cowrie and checked its status.
8. Verified Cowrie was listening on the correct port.
9. Simulated an attacker by connecting via SSH.
10. Checked the logs to see what activity was recorded.
11. Stopped Cowrie after testing.
12. Created a Canary Token (Excel) and set the notification email.
13. Renamed the Excel file, placed it in a relevant folder, and added fake data.
14. Tested the token and received the alert email.

Step 1: Kali Linux Setup and Update

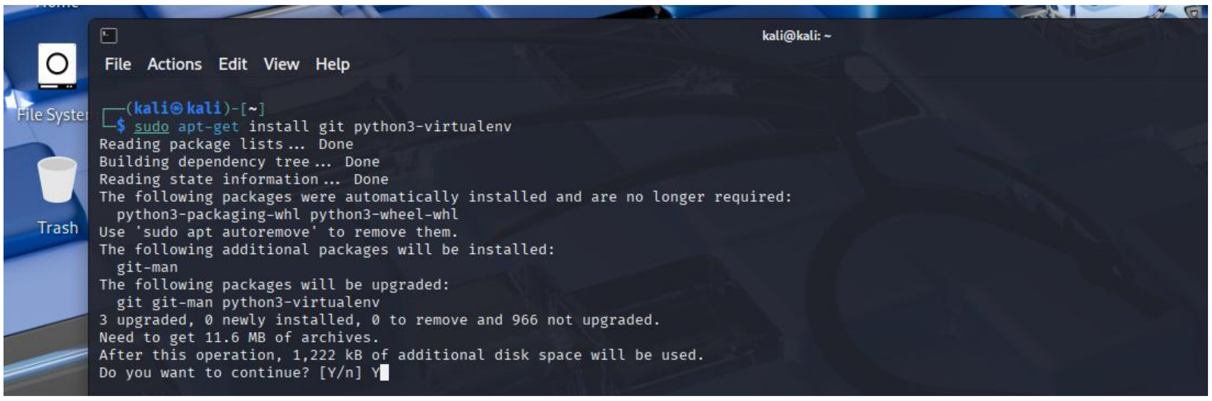
- I installed and started Kali Linux in my virtual environment and updated the system to prepare for the Cowrie honeypot setup.



```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.2 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [51.8 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [119 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [326 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [200 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [911 kB]
Fetched 74.6 MB in 13s (5,925 kB/s)
969 packages can be upgraded. Run 'apt list --upgradable' to see them.

┌──(kali㉿kali)-[~]
└─$ 
```

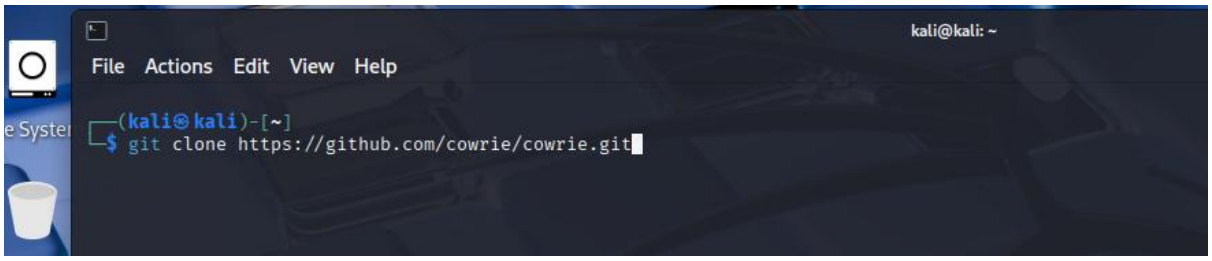
Step 2: Install Cowrie

- I installed the required dependencies (`git` and `python3-virtualenv`).
- I created a Python virtual environment for Cowrie to keep its dependencies isolated, ensure stability, improve security, and make it easier to manage.
- **Dependency isolation** – Cowrie relies on specific Python libraries and versions. A virtual environment keeps these separate from your system’s global Python packages, avoiding conflicts.
- **Stability and reproducibility** – If you upgrade or change system-wide Python packages, Cowrie might break. The virtual environment ensures Cowrie always runs with the correct versions.
- **Security** – Since honeypots interact with potentially malicious input, keeping dependencies in an isolated environment reduces the risk of affecting the main system.
- **Ease of management** – You can activate/deactivate the environment when working with Cowrie, and remove it cleanly if you don’t need it anymore.



```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get install git python3-virtualenv
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following packages were automatically installed and are no longer required:
  python3-packaging-whl python3-wheel-whl
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  git-man
The following packages will be upgraded:
  git git-man python3-virtualenv
3 upgraded, 0 newly installed, 0 to remove and 966 not upgraded.
Need to get 11.6 MB of archives.
After this operation, 1,222 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

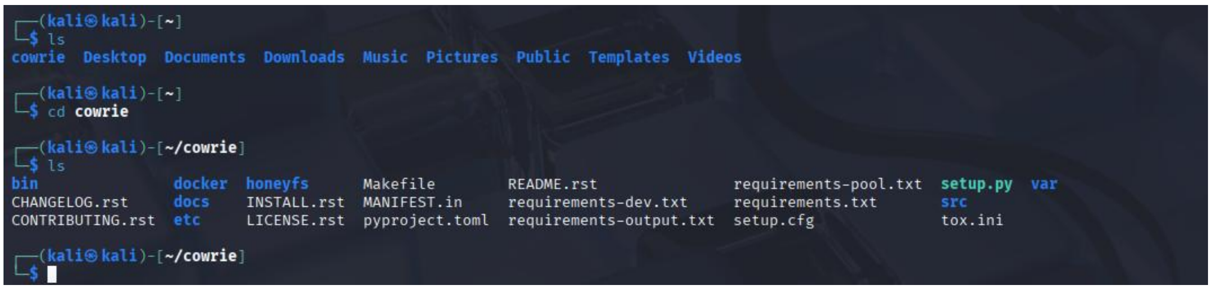
- Then I cloned the official Cowrie repository from GitHub.



```
┌──(kali㉿kali)-[~]
└─$ git clone https://github.com/cowrie/cowrie.git
```

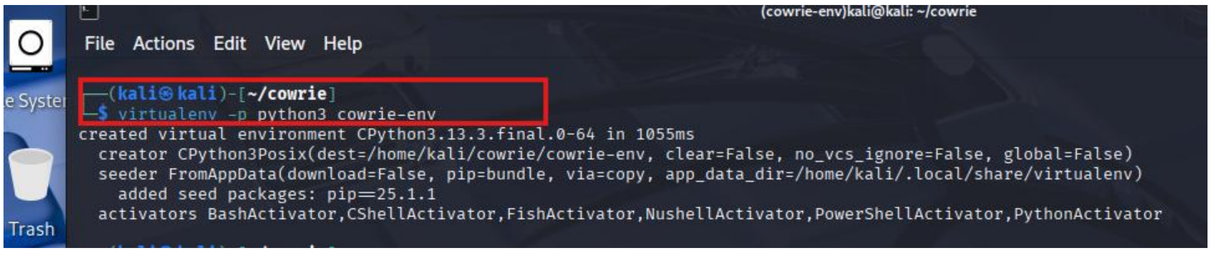
Step 3: Set Up Python Virtual Environment

- I navigated to the Cowrie directory using `cd cowrie`.



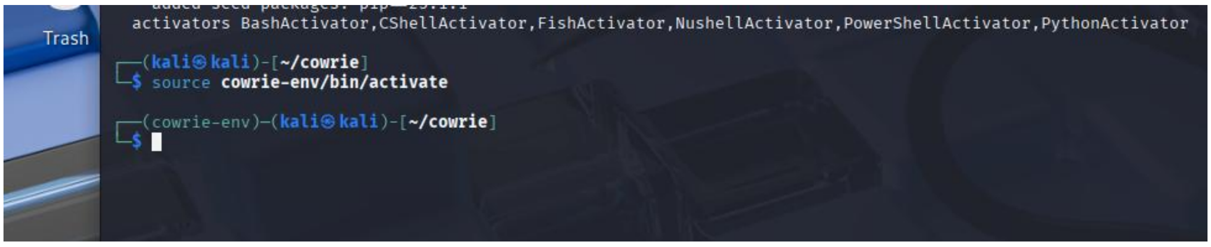
```
┌──(kali㉿kali)-[~]
└─$ ls
cowrie  Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos

┌──(kali㉿kali)-[~]
└─$ cd cowrie

┌──(kali㉿kali)-[~/cowrie]
└─$ ls
bin               docker  honeyfs      Makefile        README.rst               requirements-pool.txt  setup.py  var
CHANGELOG.rst     docs    INSTALL.rst  MANIFEST.in     requirements-dev.txt     requirements.txt       src
CONTRIBUTING.rst  etc     LICENSE.rst  pyproject.toml  requirements-output.txt  setup.cfg              tox.ini

┌──(kali㉿kali)-[~/cowrie]
└─$
```

- I created a Python virtual environment for Cowrie with `virtualenv -p python3 cowrie-env`.



```
┌──(kali㉿kali)-[~/cowrie]
└─$ virtualenv -p python3 cowrie-env
created virtual environment CPython3.13.3.final.0-64 in 1055ms
  creator CPython3Posix(dest=/home/kali/cowrie/cowrie-env, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, via=copy, app_data_dir=/home/kali/.local/share/virtualenv)
    added seed packages: pip==25.1.1
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
```

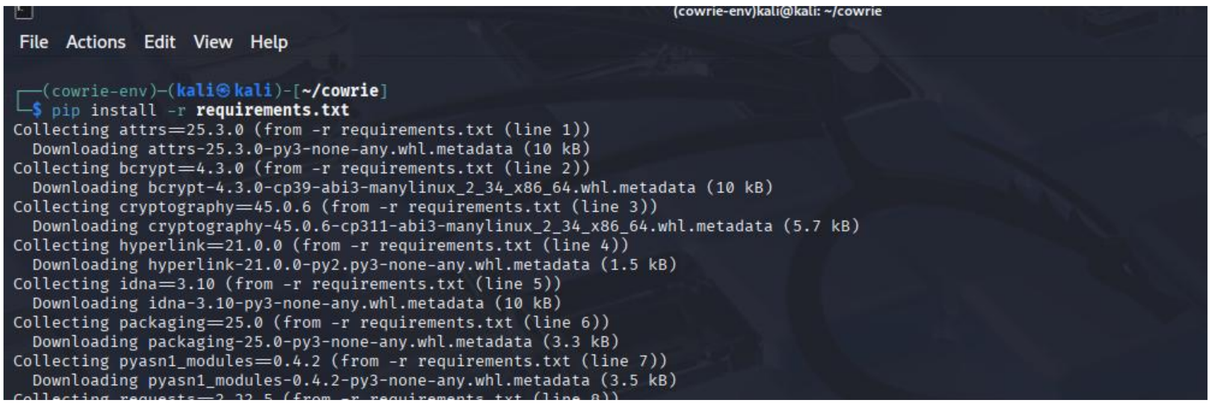
- I activated the virtual environment by running `source cowrie-env/bin/activate`.



```
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator

┌──(kali㉿kali)-[~/cowrie]
└─$ source cowrie-env/bin/activate

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$
```

Step 4: Install Required Python Packages

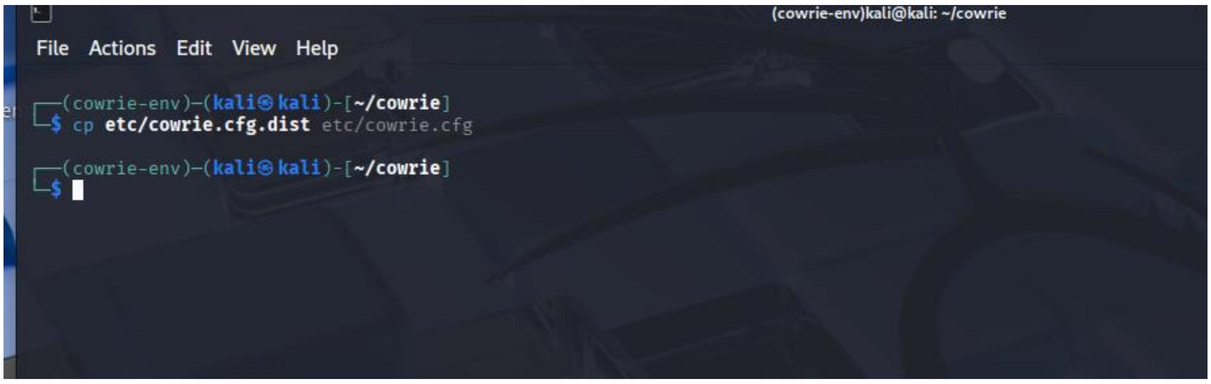
- I installed all the necessary Python dependencies for Cowrie.
- I ran the command `pip install -r requirements.txt` to complete the installation.
- This ensures Cowrie has all the libraries it needs to run properly and avoid errors.



```
┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ pip install -r requirements.txt
Collecting attrs==25.3.0 (from -r requirements.txt (line 1))
  Downloading attrs-25.3.0-py3-none-any.whl.metadata (10 kB)
Collecting bcrypt==4.3.0 (from -r requirements.txt (line 2))
  Downloading bcrypt-4.3.0-cp39-abi3-manylinux_2_34_x86_64.whl.metadata (10 kB)
Collecting cryptography==45.0.6 (from -r requirements.txt (line 3))
  Downloading cryptography-45.0.6-cp311-abi3-manylinux_2_34_x86_64.whl.metadata (5.7 kB)
Collecting hyperlink==21.0.0 (from -r requirements.txt (line 4))
  Downloading hyperlink-21.0.0-py2.py3-none-any.whl.metadata (1.5 kB)
Collecting idna==3.10 (from -r requirements.txt (line 5))
  Downloading idna-3.10-py3-none-any.whl.metadata (10 kB)
Collecting packaging==25.0 (from -r requirements.txt (line 6))
  Downloading packaging-25.0-py3-none-any.whl.metadata (3.3 kB)
Collecting pyasn1_modules==0.4.2 (from -r requirements.txt (line 7))
  Downloading pyasn1_modules-0.4.2-py3-none-any.whl.metadata (3.5 kB)
```

Step 5: Configure Cowrie

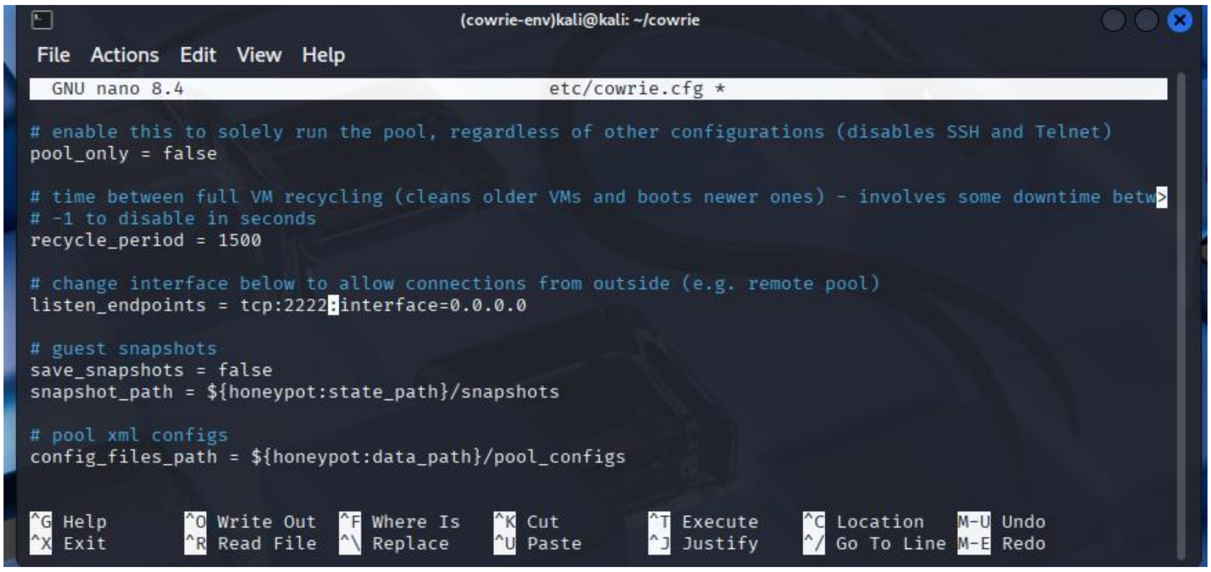
- I copied the default configuration file to create my own using `cp etc/cowrie.cfg.dist etc/cowrie.cfg`.
- This allowed me to customize Cowrie's settings without altering the original configuration file.



```
┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ cp etc/cowrie.cfg.dist etc/cowrie.cfg

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ 
```

- Although I kept the default port at 2222, I could change the `listen_endpoints` setting to other ports like 22 for SSH or 80/443 for HTTP if needed.

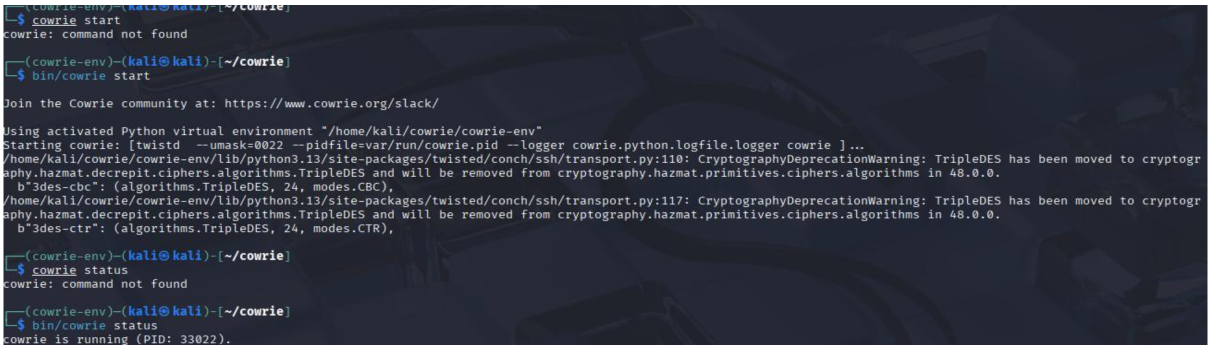
- Any connection to the chosen port allows me to capture live attacker activity and gather information in real time.



```
  GNU nano 8.4                           etc/cowrie.cfg *

# enable this to solely run the pool, regardless of other configurations (disables SSH and Telnet)
pool_only = false

# time between full VM recycling (cleans older VMs and boots newer ones) - involves some downtime betw>
# -1 to disable in seconds
recycle_period = 1500

# change interface below to allow connections from outside (e.g. remote pool)
listen_endpoints = tcp:2222:interface=0.0.0.0

# guest snapshots
save_snapshots = false
snapshot_path = ${honeypot:state_path}/snapshots

# pool xml configs
config_files_path = ${honeypot:data_path}/pool_configs

^G Help       ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo
```

Step 6: Start the Cowrie Honeypot

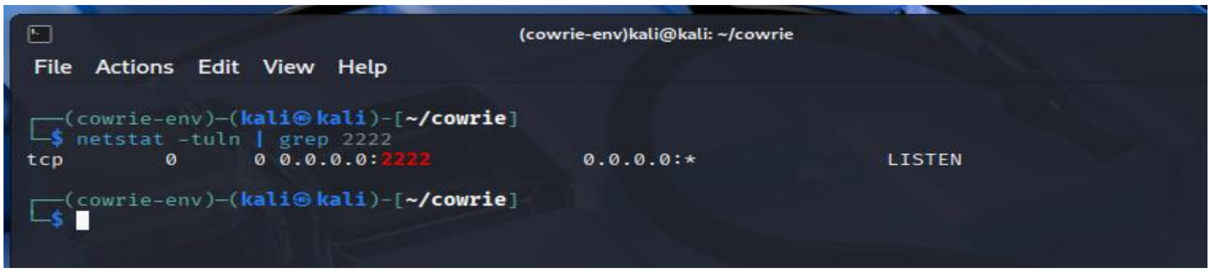
- I started the Cowrie honeypot by running `bin/cowrie start`.
- I confirmed it was running and active by checking its status with `bin/cowrie status`.



```
└─$ cowrie start
cowrie: command not found

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ bin/cowrie start

Join the Cowrie community at: https://www.cowrie.org/slack/

Using activated Python virtual environment "/home/kali/cowrie/cowrie-env"
Starting cowrie: [twistd  --umask=0022 --pidfile=var/run/cowrie.pid --logger cowrie.python.logfile.logger cowrie ]...
/home/kali/cowrie/cowrie-env/lib/python3.13/site-packages/twisted/conch/ssh/transport.py:110: CryptographyDeprecationWarning: TripleDES has been moved to cryptogr
aphy.hazmat.decrepit.ciphers.algorithms.TripleDES and will be removed from cryptography.hazmat.primitives.ciphers.algorithms in 48.0.0.
  b"3des-cbc": (algorithms.TripleDES, 24, modes.CBC),
/home/kali/cowrie/cowrie-env/lib/python3.13/site-packages/twisted/conch/ssh/transport.py:117: CryptographyDeprecationWarning: TripleDES has been moved to cryptogr
aphy.hazmat.decrepit.ciphers.algorithms.TripleDES and will be removed from cryptography.hazmat.primitives.ciphers.algorithms in 48.0.0.
  b"3des-ctr": (algorithms.TripleDES, 24, modes.CTR),

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ cowrie status
cowrie: command not found

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ bin/cowrie status
cowrie is running (PID: 33022).
```

Step 7: Verify Cowrie Listening Port

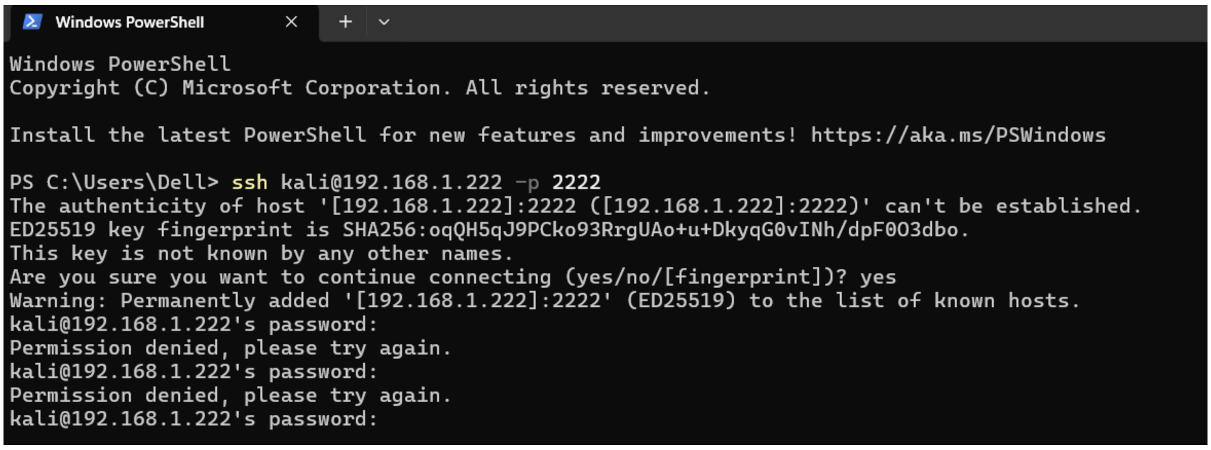
- I checked if Cowrie was listening on the correct port by running `netstat -tuln | grep 2222`.
- This confirmed that the honeypot was active and ready to capture any incoming connections.



```
┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ netstat -tuln | grep 2222
tcp        0      0 0.0.0.0:2222            0.0.0.0:*               LISTEN

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
└─$ 
```

Step 8: Simulate Attacker Connection

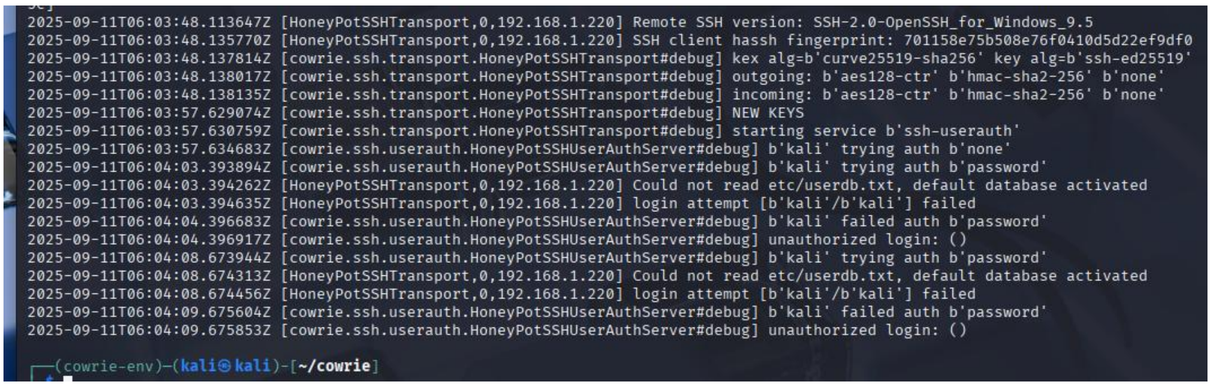
- I simulated an attacker by connecting from my Windows host PowerShell to the Kali VM using `ssh kali@192.168.1.222 -p 2222`.
- This allowed me to see what an attacker would experience when accessing the honeypot.



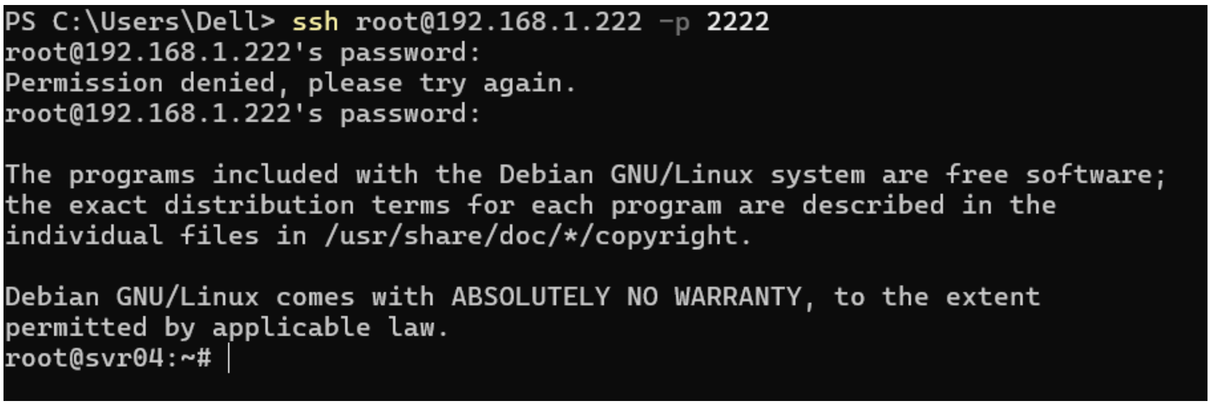
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Dell> ssh kali@192.168.1.222 -p 2222
The authenticity of host '[192.168.1.222]:2222 ([192.168.1.222]:2222)' can't be established.
ED25519 key fingerprint is SHA256:oqQH5qJ9PCko93RrgUAo+u+DkyqG0vINh/dpF003dbo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[192.168.1.222]:2222' (ED25519) to the list of known hosts.
kali@192.168.1.222's password:
Permission denied, please try again.
kali@192.168.1.222's password:
Permission denied, please try again.
kali@192.168.1.222's password:
```

Step 9: View Attack Logs

- I navigated to the Cowrie log directory using `cd var/log/cowrie`.
- I monitored the log file in real time with `tail -f cowrie.log` to see all attacker actions and commands as they were recorded.
- While simulating an attacker, I tried connecting with incorrect passwords, and Cowrie recorded each attempt in real time. This allowed me to see how the honeypot captures and logs attacker activity immediately.

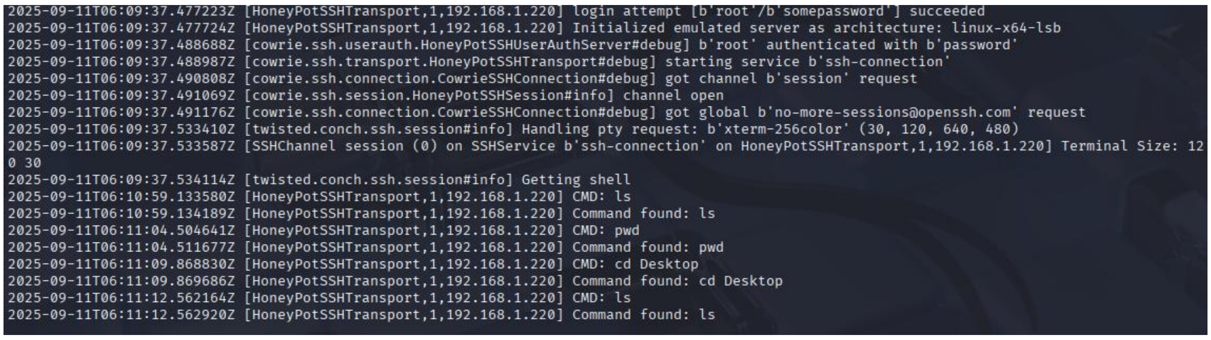


```
2025-09-11T06:03:48.113647Z [HoneyPotSSHTransport,0,192.168.1.220] Remote SSH version: SSH-2.0-OpenSSH_for_Windows_9.5
2025-09-11T06:03:48.135770Z [HoneyPotSSHTransport,0,192.168.1.220] SSH client hassh fingerprint: 701158e75b508e76f0410d5d22ef9df0
2025-09-11T06:03:48.137814Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] kex alg=b'curve25519-sha256' key alg=b'ssh-ed25519'
2025-09-11T06:03:48.138017Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] outgoing: b'aes128-ctr' b'hmac-sha2-256' b'none'
2025-09-11T06:03:48.138135Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] incoming: b'aes128-ctr' b'hmac-sha2-256' b'none'
2025-09-11T06:03:57.629074Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] NEW KEYS
2025-09-11T06:03:57.630759Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] starting service b'ssh-userauth'
2025-09-11T06:03:57.634683Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'kali' trying auth b'none'
2025-09-11T06:04:03.393894Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'kali' trying auth b'password'
2025-09-11T06:04:03.394262Z [HoneyPotSSHTransport,0,192.168.1.220] Could not read etc/userdb.txt, default database activated
2025-09-11T06:04:03.394635Z [HoneyPotSSHTransport,0,192.168.1.220] login attempt [b'kali'/b'kali'] failed
2025-09-11T06:04:04.396683Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'kali' failed auth b'password'
2025-09-11T06:04:04.396917Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] unauthorized login: ()
2025-09-11T06:04:08.673944Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'kali' trying auth b'password'
2025-09-11T06:04:08.674313Z [HoneyPotSSHTransport,0,192.168.1.220] Could not read etc/userdb.txt, default database activated
2025-09-11T06:04:08.674456Z [HoneyPotSSHTransport,0,192.168.1.220] login attempt [b'kali'/b'kali'] failed
2025-09-11T06:04:09.675604Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'kali' failed auth b'password'
2025-09-11T06:04:09.675853Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] unauthorized login: ()

┌──(cowrie-env)─(kali㉿kali)-[~/cowrie]
```



```
PS C:\Users\Dell> ssh root@192.168.1.222 -p 2222
root@192.168.1.222's password:
Permission denied, please try again.
root@192.168.1.222's password:

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@svr04:~# |
```

- I observed in the logs that SSH connections to the honeypot were recorded in real time.



```
2025-09-11T06:09:37.477223Z [HoneyPotSSHTransport,1,192.168.1.220] login attempt [b'root'/b'somepassword'] succeeded
2025-09-11T06:09:37.477724Z [HoneyPotSSHTransport,1,192.168.1.220] Initialized emulated server as architecture: linux-x64-lsb
2025-09-11T06:09:37.488688Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'root' authenticated with b'password'
2025-09-11T06:09:37.488987Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] starting service b'ssh-connection'
2025-09-11T06:09:37.490808Z [cowrie.ssh.connection.CowrieSSHConnection#debug] got channel b'session' request
2025-09-11T06:09:37.491069Z [cowrie.ssh.session.HoneyPotSSHSession#info] channel open
2025-09-11T06:09:37.491176Z [cowrie.ssh.connection.CowrieSSHConnection#debug] got global b'no-more-sessions@openssh.com' request
2025-09-11T06:09:37.533410Z [twisted.conch.ssh.session#info] Handling pty request: b'xterm-256color' (30, 120, 640, 480)
2025-09-11T06:09:37.533587Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,1,192.168.1.220] Terminal Size: 12
0 30
2025-09-11T06:09:37.534114Z [twisted.conch.ssh.session#info] Getting shell
2025-09-11T06:10:59.133580Z [HoneyPotSSHTransport,1,192.168.1.220] CMD: ls
2025-09-11T06:10:59.134189Z [HoneyPotSSHTransport,1,192.168.1.220] Command found: ls
2025-09-11T06:11:04.504641Z [HoneyPotSSHTransport,1,192.168.1.220] CMD: pwd
2025-09-11T06:11:04.511677Z [HoneyPotSSHTransport,1,192.168.1.220] Command found: pwd
2025-09-11T06:11:09.868830Z [HoneyPotSSHTransport,1,192.168.1.220] CMD: cd Desktop
2025-09-11T06:11:09.869686Z [HoneyPotSSHTransport,1,192.168.1.220] Command found: cd Desktop
2025-09-11T06:11:12.562164Z [HoneyPotSSHTransport,1,192.168.1.220] CMD: ls
2025-09-11T06:11:12.562920Z [HoneyPotSSHTransport,1,192.168.1.220] Command found: ls
```

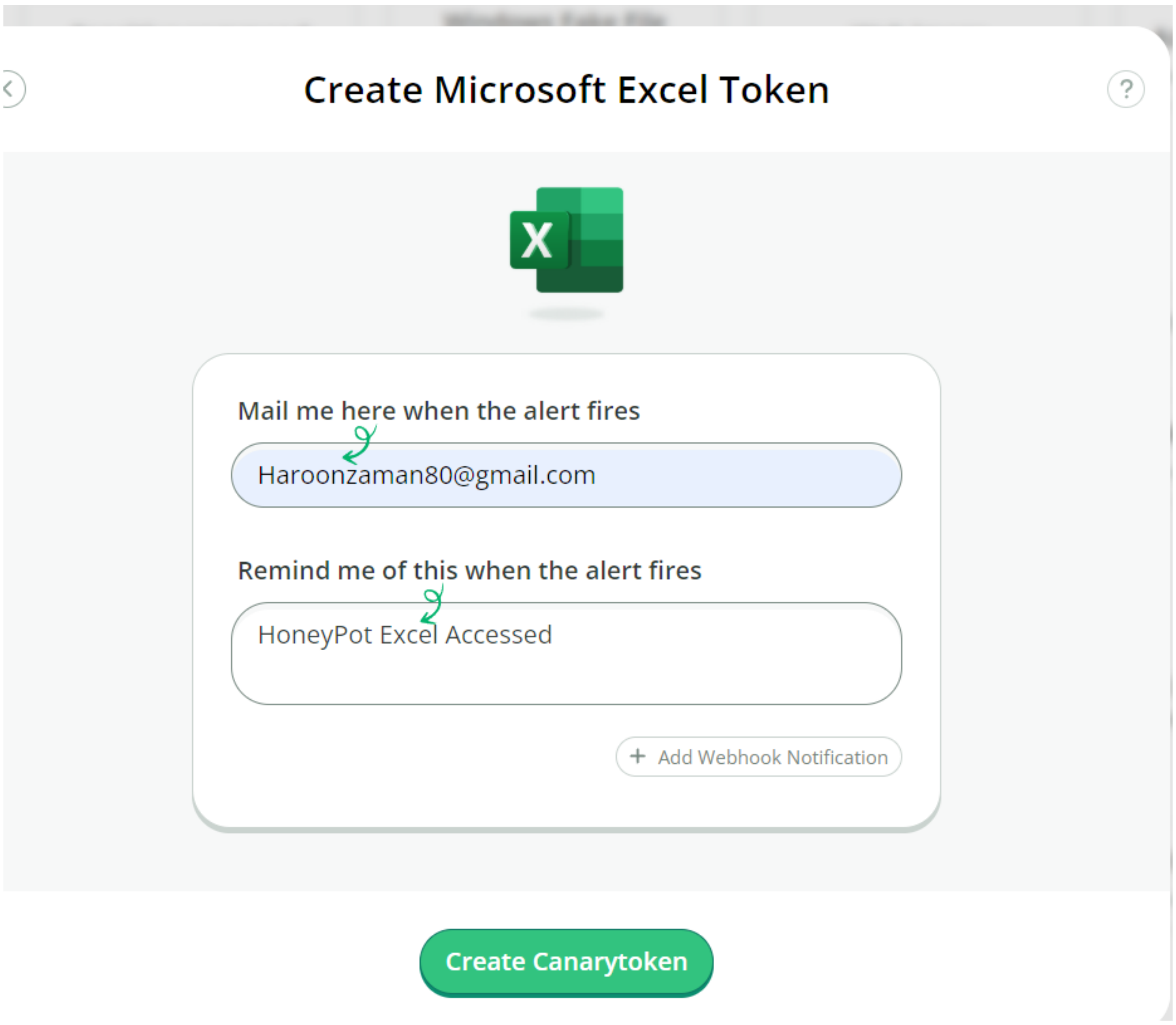
Step 11: Stop the Honeypot

- I stopped the Cowrie honeypot after completing my tests by running `cowrie stop`.
- This concluded the honeypot session while preserving all logged attacker activity for analysis.

Canary Token Setup

A Canary Token is a small, hidden piece of data or file that I deploy to detect unauthorized access. When it is accessed, it sends me an alert with information about the user or system that triggered it.

- I navigated to the Canary Tokens website at https://canarytokens.org/nest/.
- I selected an Excel file as the token to configure and set up.
- I saw many application options (nests) on the Canary Tokens website.
- I selected the Excel token to create.
- I entered my email address to receive alerts and added a short reminder message.


Create Microsoft Excel Token

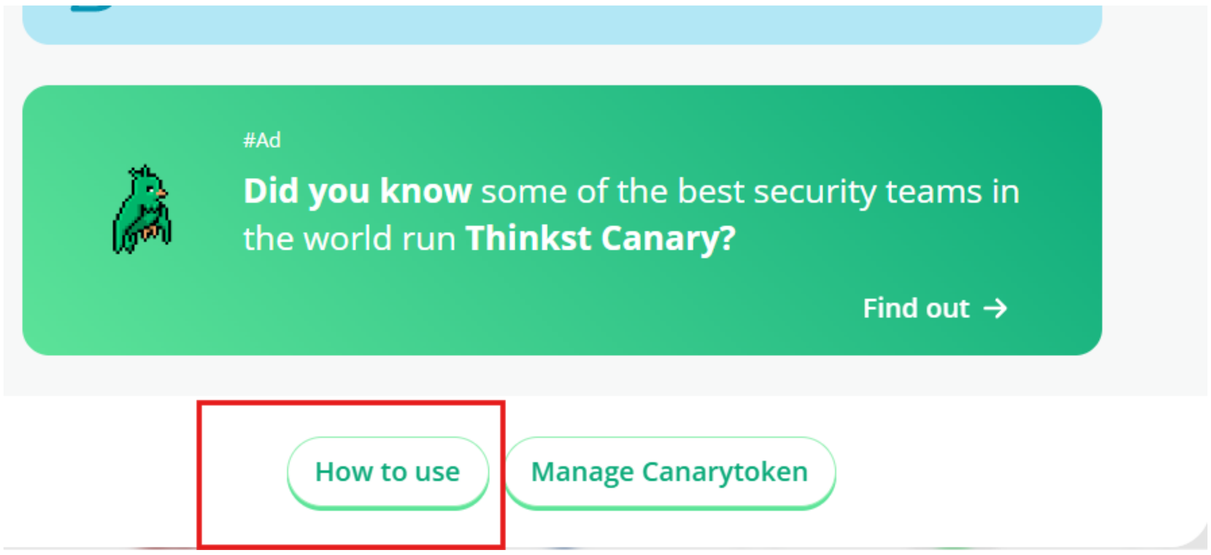
Mail me here when the alert fires

Haroonzaman80@gmail.com

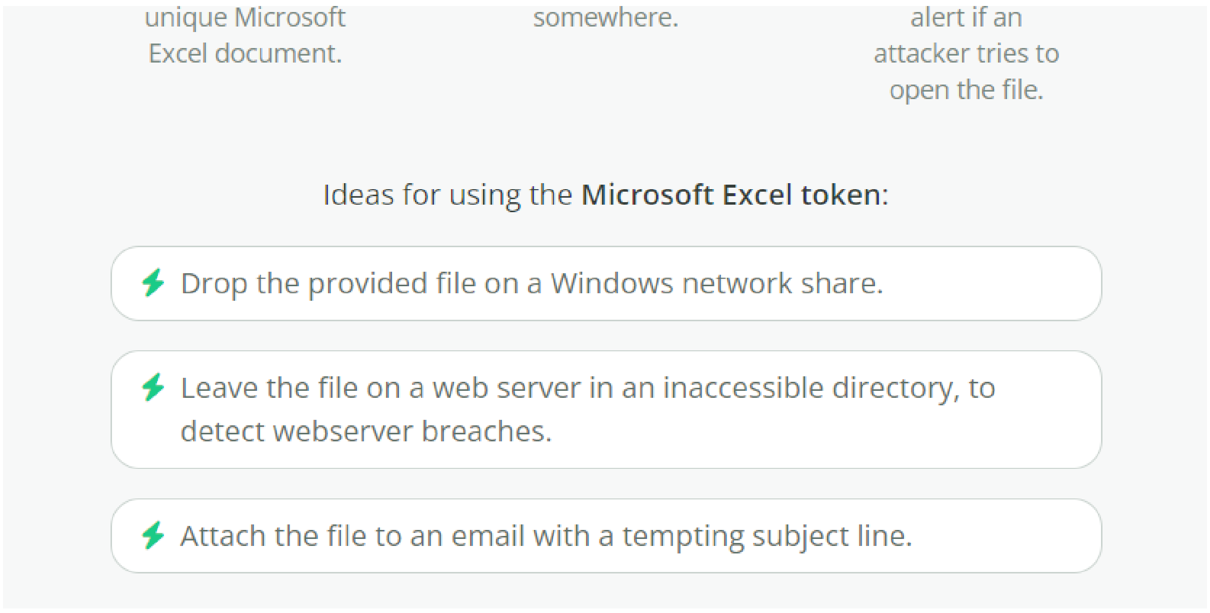
Remind me of this when the alert fires

HoneyPot Excel Accessed

- After clicking "Create Token," I was taken to the next form where I could download the token file.
- I also noticed a "How to Use This Token" section with instructions on where and how to place the file.


#Ad
Did you know some of the best security teams in the world run Thinkst Canary?
Find out →
How to use
Manage Canarytoken

- They provided three different ways to use the token.
- I chose the email method to receive alerts whenever the token was accessed.



unique Microsoft Excel document.

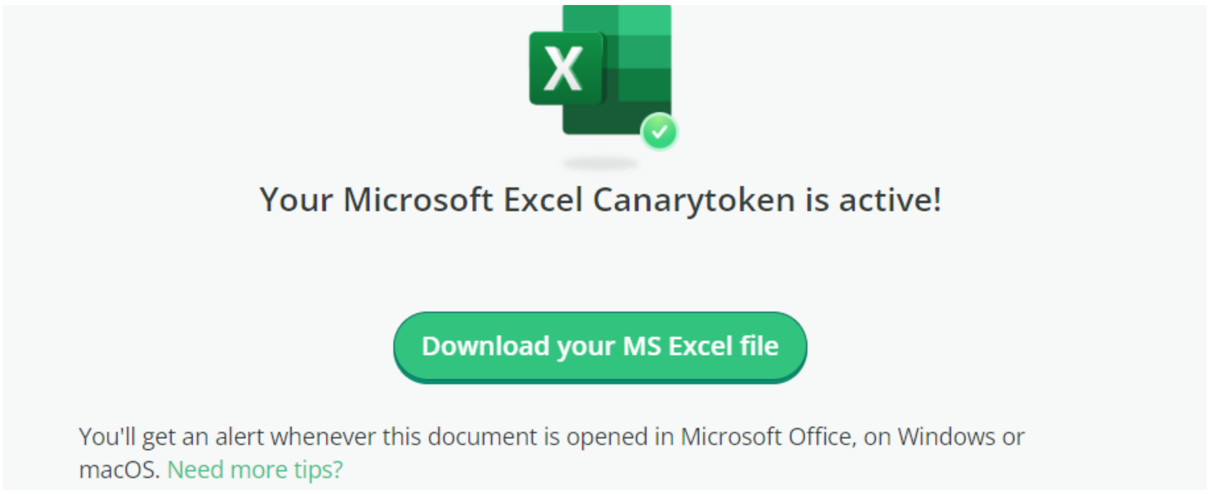
somewhere.

alert if an attacker tries to open the file.

Ideas for using the **Microsoft Excel token**:

- Drop the provided file on a Windows network share.
- Leave the file on a web server in an inaccessible directory, to detect webserver breaches.
- Attach the file to an email with a tempting subject line.

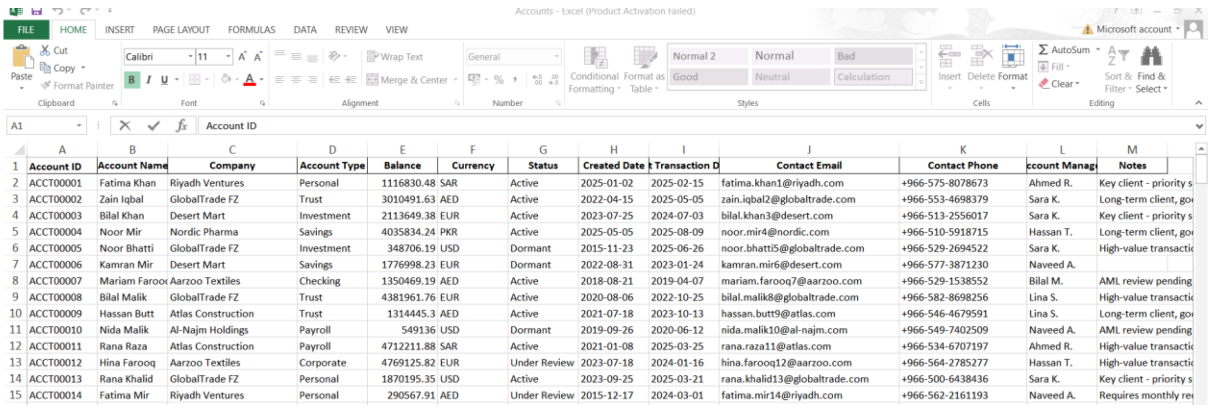
- I clicked back and downloaded the Canary Token file to my system.



Your Microsoft Excel Canarytoken is active!

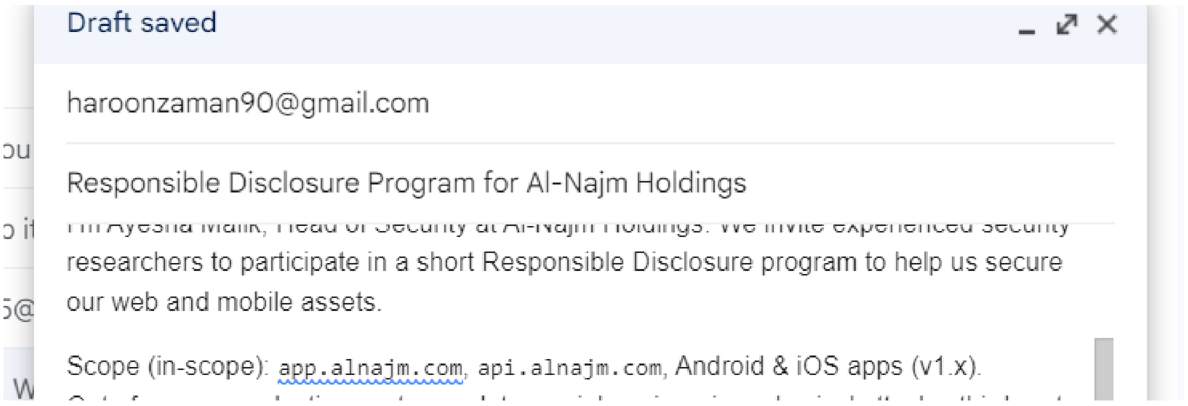
You'll get an alert whenever this document is opened in Microsoft Office, on Windows or macOS. Need more tips?

- I renamed the downloaded file to "Accounts" to make it look authentic.
- I added some dummy data to make the file appear convincing.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Account ID	Account Name	Company	Account Type	Balance	Currency	Status	Created Date	t Transaction D	Contact Email	Contact Phone	ccount Manag	Notes
2	ACCT00001	Fatima Khan	Riyadh Ventures	Personal	1116830.48	SAR	Active	2025-01-02	2025-02-15	fatima.khan1@riyadh.com	+966-575-8078673	Ahmed R.	Key client - priority s
3	ACCT00002	Zain Iqbal	GlobalTrade FZ	Trust	3010491.63	AED	Active	2022-04-15	2025-05-05	zain.iqbal2@globaltrade.com	+966-553-4698379	Sara K.	Long-term client, go
4	ACCT00003	Bilal Khan	Desert Mart	Investment	2113649.38	EUR	Active	2023-07-25	2024-07-03	bilal.khan3@desert.com	+966-513-2556017	Sara K.	Key client - priority s
5	ACCT00004	Noor Mir	Nordic Pharma	Savings	4035834.24	PKR	Active	2025-05-05	2025-08-09	noor.mir4@nordic.com	+966-510-5918715	Hassan T.	Long-term client, go
6	ACCT00005	Noor Bhatti	GlobalTrade FZ	Investment	348706.19	USD	Dormant	2015-11-23	2025-06-26	noor.bhatti5@globaltrade.com	+966-529-2694522	Sara K.	High-value transacti
7	ACCT00006	Kamran Mir	Desert Mart	Savings	1776998.23	EUR	Dormant	2022-08-31	2023-01-24	kamran.mir6@desert.com	+966-577-3871230	Naveed A.	
8	ACCT00007	Mariam Farooq	Aarzoo Textiles	Checking	1350469.19	AED	Active	2018-08-21	2019-04-07	mariam.farooq7@aarzoo.com	+966-529-1538552	Bilal M.	AML review pending
9	ACCT00008	Bilal Malik	GlobalTrade FZ	Trust	4381961.76	EUR	Active	2020-08-06	2022-10-25	bilal.malik8@globaltrade.com	+966-582-8698256	Lina S.	High-value transacti
10	ACCT00009	Hassan Butt	Atlas Construction	Trust	1314445.3	AED	Active	2021-07-18	2023-10-13	hassan.butt9@atlas.com	+966-546-4679591	Lina S.	Long-term client, go
11	ACCT00010	Nida Malik	Al-Najm Holdings	Payroll	549136	USD	Dormant	2019-09-26	2020-06-12	nida.malik10@al-najm.com	+966-549-7402509	Naveed A.	AML review pending
12	ACCT00011	Rana Raza	Atlas Construction	Payroll	4712211.88	SAR	Active	2021-01-08	2025-03-25	rana.raza11@atlas.com	+966-534-6707197	Ahmed R.	High-value transacti
13	ACCT00012	Hina Farooq	Aarzoo Textiles	Corporate	4769125.82	EUR	Under Review	2023-07-18	2024-01-16	hina.farooq12@aarzoo.com	+966-564-2785277	Hassan T.	High-value transacti
14	ACCT00013	Rana Khalid	GlobalTrade FZ	Personal	1870195.35	USD	Active	2023-09-25	2025-03-21	rana.khalid13@globaltrade.com	+966-500-6438436	Sara K.	Key client - priority s
15	ACCT00014	Fatima Mir	Riyadh Ventures	Personal	290567.91	AED	Under Review	2015-12-17	2024-03-01	fatima.mir14@riyadh.com	+966-562-2161193	Naveed A.	Requires monthly re

- I started an email from one Gmail account to another to simulate access.
- I did this to test how an attacker might access the file in a real scenario, demonstrating that if the token file were shared or opened by an unauthorized user, I would receive an alert.



Draft saved

haroonzaman90@gmail.com

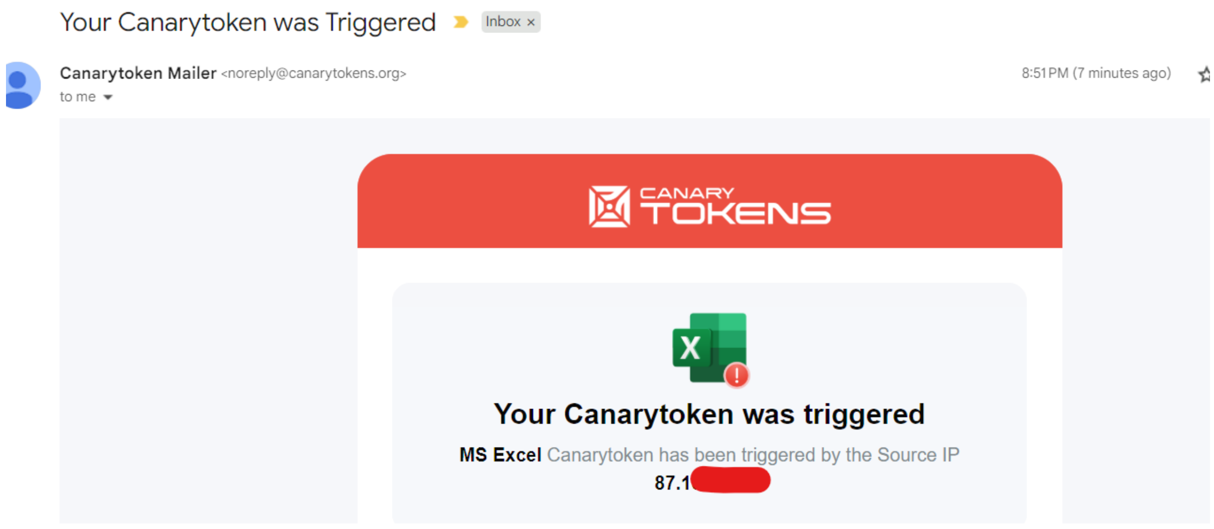
Responsible Disclosure Program for Al-Najm Holdings

I'm Ayesha Malik, Head of Security at Al-Najm Holdings. We invite experienced security researchers to participate in a short Responsible Disclosure program to help us secure our web and mobile assets.

Scope (in-scope): `app.alnajm.com`, `api.alnajm.com`, Android & iOS apps (v1.x).

- When the file was downloaded and opened on the other end, simulating an attacker accessing it, I received an alert in the email I had provided when creating the token.
- The email alert included the attacker's source IP, timestamp, and user agent.
- This information allowed me to track who accessed the token and when.

  This showed that the Canary Token successfully detected unauthorized access in real time.



Your Canarytoken was Triggered Inbox ×

Canarytoken Mailer <noreply@canarytokens.org>
to me

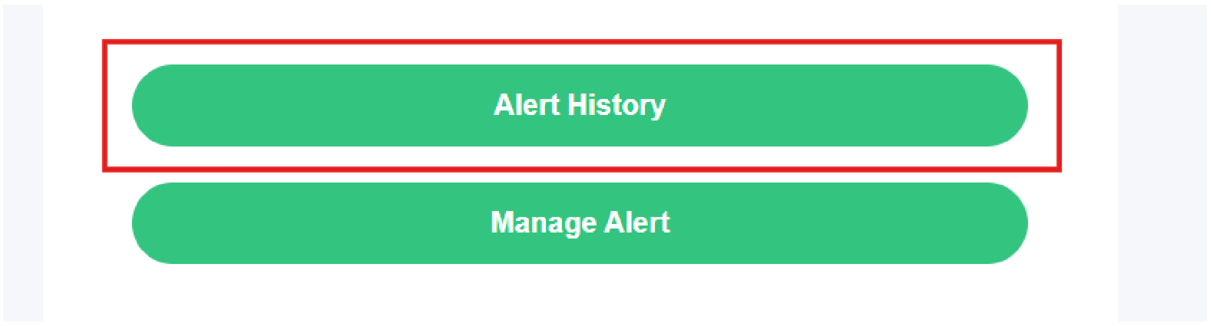
8:51 PM (7 minutes ago)

CANARY TOKENS

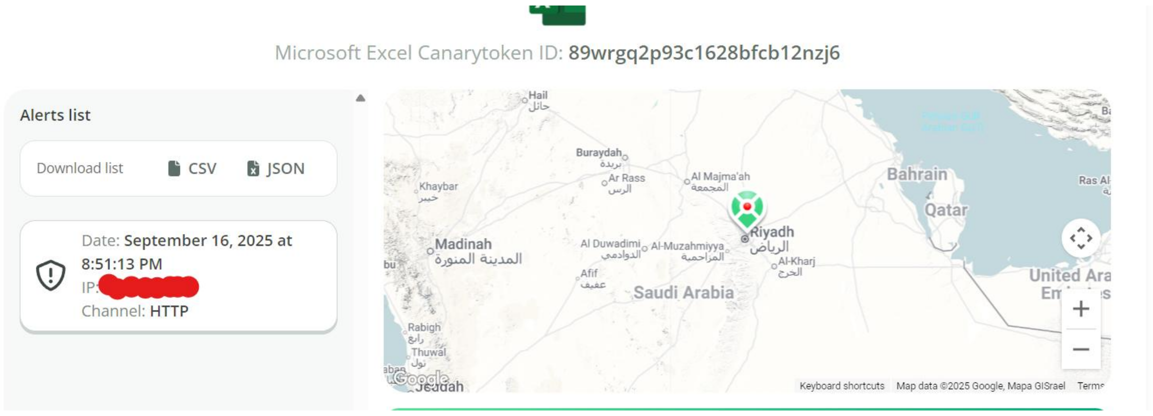
Your Canarytoken was triggered

MS Excel Canarytoken has been triggered by the Source IP **87.1**[redacted]

- I scrolled down and clicked the “Alert History” button.



- It redirected me to a page showing the attacker’s source IP, timestamp, user agent, and location on a map.



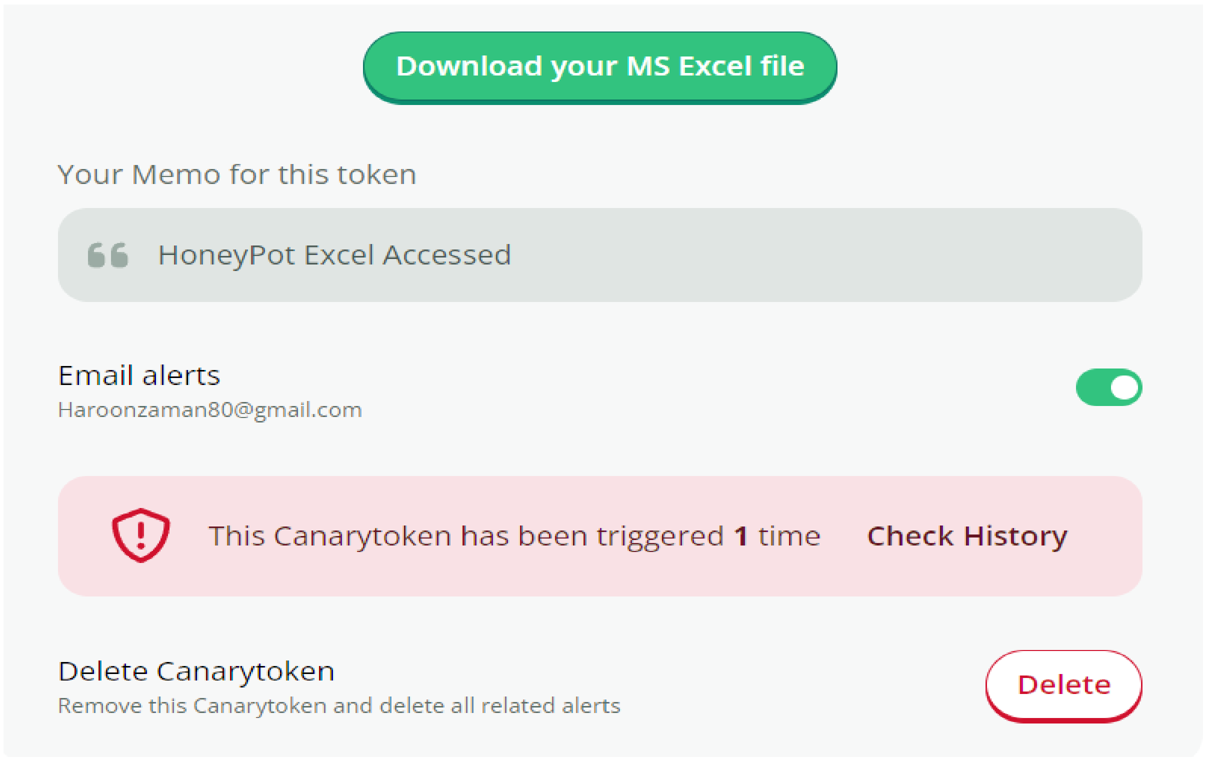
Microsoft Excel Canarytoken ID: 89wrgq2p93c1628bfcb12nzj6

Alerts list

Download list CSV JSON

Date: September 16, 2025 at 8:51:13 PM
IP: [redacted]
Channel: HTTP

- Back in the email, I clicked “Manage Alert.”
- From there, I could delete the token or redownload it to use it again.



Download your MS Excel file

Your Memo for this token

HoneyPot Excel Accessed

Email alerts
Haroonzaman80@gmail.com

This Canarytoken has been triggered **1** time Check History

Delete Canarytoken
Remove this Canarytoken and delete all related alerts

Delete