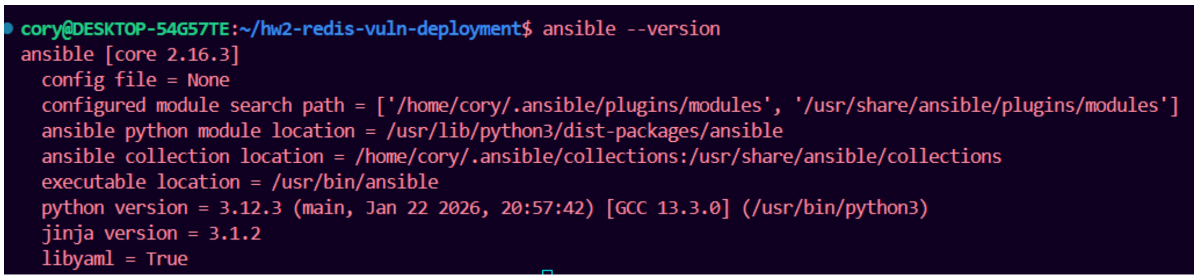
Cory Le Cyber Defense Techniques: Homework 2 2/6/2026

Deployment Documentation

Prerequisites

- Target Machine OS:
  - Ubuntu 24.04 LTS
- Ansible Version:
  - 

```
cory@DESKTOP-54G57TE:~/hw2-redis-vuln-deployment$ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/cory/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/cory/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Jan 22 2026, 20:57:42) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
```

- Control Node: WSL2 Ubuntu on Windows 11
  - Required packages:
    - WSL (if using Windows for a control node)
    - WSL Ubuntu (if using Windows for a control node)
    - Ansible
    - Git
    - Sshpass
    - redis-tools
- Network Requirements:
  - Target machine must be reachable via SSH from the control node
- Required before running:
  - ‘openssh-server’ installed and running on target machine
  - SSH key authentication configured and shared between control node and target for easy authentication between the two
  - Target machine user must have sudo privileges

Cory Le Cyber Defense Techniques: Homework 2 2/6/2026

Deployment Documentation

Step-By-Step Deployment Quick Start Guide

1. Clone the repository from GitHub to have it locally:
   a. Command: git clone https://github.com/core-e-40/hw2-redis-vuln-deployment.git
2. Switch into the local clone repository:
   a. Command: cd hw2-redis-vuln-deployment
3. Ensure you have Ansible installed:
   a. Command: ansible --version
4. Run the playbook
   a. Command: ansible-playbook -i inventory.ini playbook.yml --ask-become-pass

What Gets Deployed?

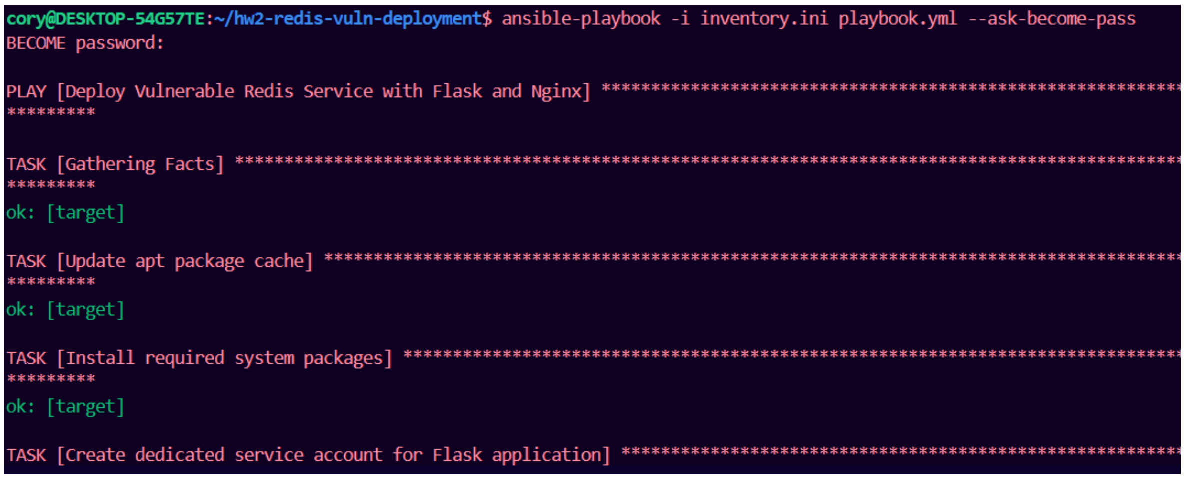
[Listed In Order of What Is Deployed First]

- Redis (port 6379) [THE VULNERABLE PIECE]
    - What is it?
        - Redis is an in-memory data storage that holds all the application data such as user credentials, app settings, and flags.
    - What is vulnerable?
        - Bound to all network interfaces
        - No authentication required to connect
        - Protected mode disabled
        - Dangerous commands left enabled
    - What role does it play in the overall deployment?
        - Redis acts as the backend database for the application so as the Flask front end is manipulated by the end user, any data that needs to be stored or fetched would sit in Redis. It is also the primary attack target in the deployment because if an attacker compromises Reis, they get access to everything the web application stores. An attacker, for example, can write SSH keys into Redis to gain shell access to the server.
- Flask (port 5000)
    - What is it?
        - Flask is a lightweight Python web framework which in this instance is used to run a web application that provides an interface to interact with Redis. This allows users to store data, manage users, and view system info all from their browser.
    - What role does it play in the overall deployment?
        - Flask acts as the middle layer between the user and Redis as users will interact with the web UI and Flask will process the requests and talk to Redis on the user's behalf. It runs as a systemd service on port 5000 behind Nginx under a dedicated service account. This deployment of Flask also comes with its own vulnerabilities as it lets users execute raw Redis commands on the front end with no filtering. Additionally, it stores passwords in plaintext and easily displays queried information with no limits creating the possibility to expose server configuration info.

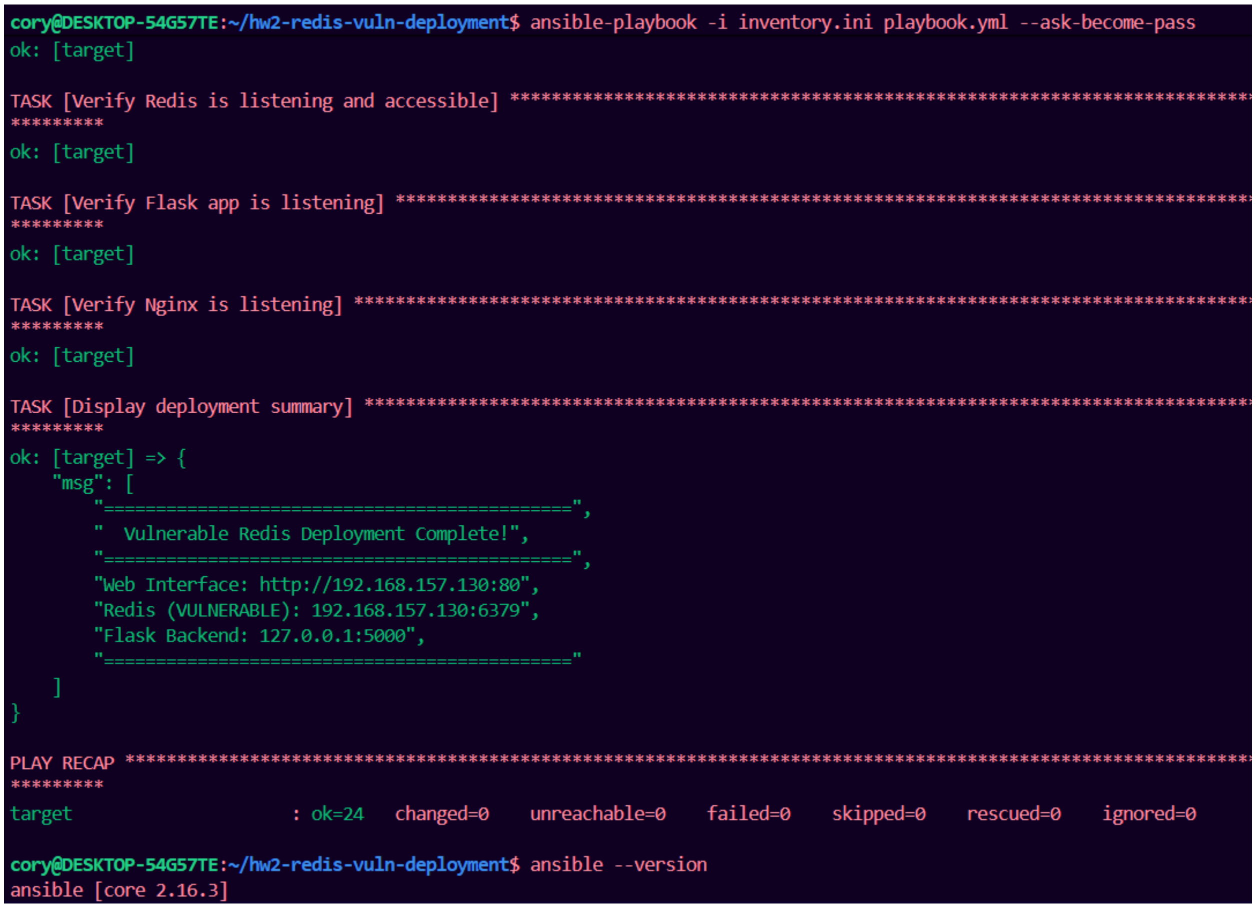
- Nginx (port 80)
    - What is it?
        - Nginx is a web server and reverse proxy which in this case acts as the public facing entry points that forwards the incoming HTTP requests to the Flask web application running behind it.
    - What roles does it play in the overall deployment?
        - Nginx is configured to listen on port 80 so users can access the web application by simply going to 'http://<target_machine_ip>' without needing to know about Flask or port information. This works to forward all traffic to Flask on 'localhost:5000' but this deployment of nginx comes with a major weakness/possible vulnerability. In 'templates/nginx-flask.config.j2' it is noted that logging has been disabled. This significantly hurts the ability to make detection rules for defensive measures meaning there is now less visibility to who is connecting to this deployment of nginx and accessing the web application.

Cory Le	Cyber Defense Techniques: Homework 2	2/6/2026
Deployment Documentation

Ansible Playbook Execution

- Beginning:
    - 

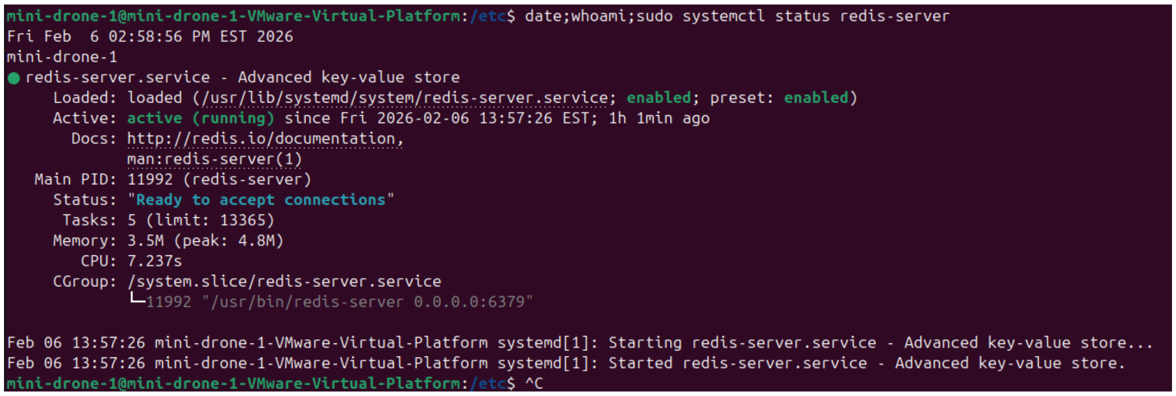
```
cory@DESKTOP-54G57TE:~/hw2-redis-vuln-deployment$ ansible-playbook -i inventory.ini playbook.yml --ask-become-pass
BECOME password:

PLAY [Deploy Vulnerable Redis Service with Flask and Nginx] *****************************************************
*********

TASK [Gathering Facts] ******************************************************************************************
*********
ok: [target]

TASK [Update apt package cache] *********************************************************************************
*********
ok: [target]

TASK [Install required system packages] *************************************************************************
*********
ok: [target]

TASK [Create dedicated service account for Flask application] ***************************************************
```

- End:
    - 

```
cory@DESKTOP-54G57TE:~/hw2-redis-vuln-deployment$ ansible-playbook -i inventory.ini playbook.yml --ask-become-pass
ok: [target]

TASK [Verify Redis is listening and accessible] *****************************************************************
*********
ok: [target]

TASK [Verify Flask app is listening] ****************************************************************************
*********
ok: [target]

TASK [Verify Nginx is listening] ********************************************************************************
*********
ok: [target]

TASK [Display deployment summary] *******************************************************************************
*********
ok: [target] => {
    "msg": [
        "============================================",
        "  Vulnerable Redis Deployment Complete!",
        "============================================",
        "Web Interface: http://192.168.157.130:80",
        "Redis (VULNERABLE): 192.168.157.130:6379",
        "Flask Backend: 127.0.0.1:5000",
        "============================================"
    ]
}

PLAY RECAP ******************************************************************************************************
*********
target                     : ok=24   changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

cory@DESKTOP-54G57TE:~/hw2-redis-vuln-deployment$ ansible --version
ansible [core 2.16.3]
```

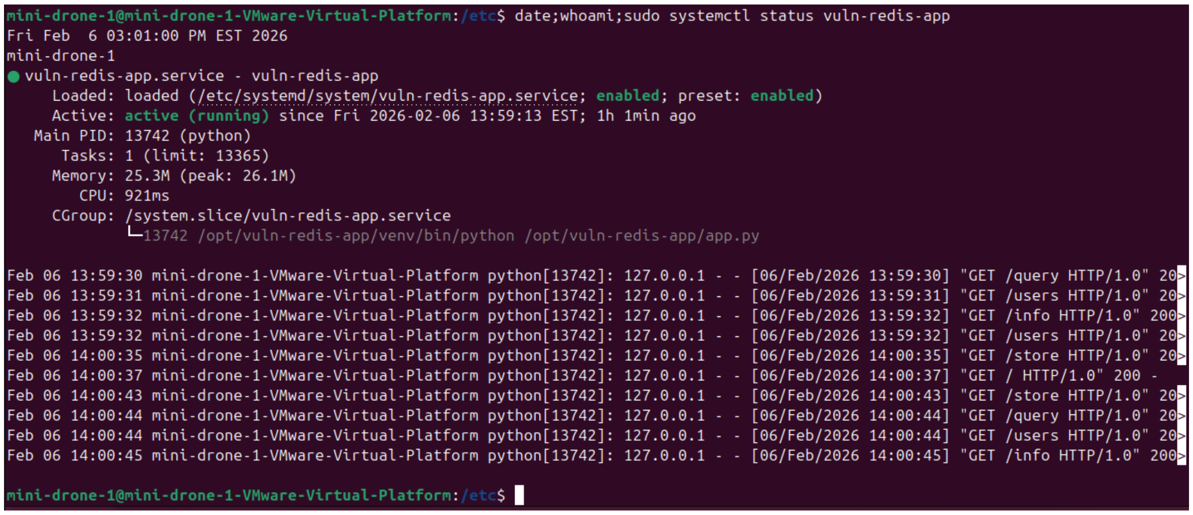
Service Running Verification

- Redis:



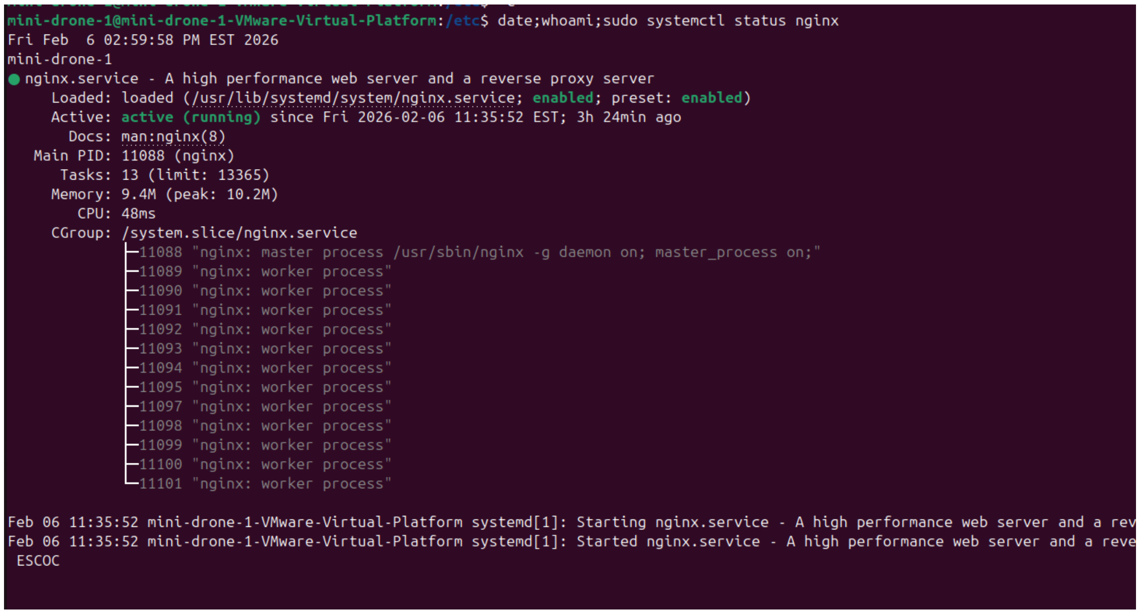
```
mini-drone-1@mini-drone-1-VMware-Virtual-Platform:/etc$ date;whoami;sudo systemctl status redis-server
Fri Feb  6 02:58:56 PM EST 2026
mini-drone-1
● redis-server.service - Advanced key-value store
     Loaded: loaded (/usr/lib/systemd/system/redis-server.service; enabled; preset: enabled)
     Active: active (running) since Fri 2026-02-06 13:57:26 EST; 1h 1min ago
       Docs: http://redis.io/documentation,
             man:redis-server(1)
   Main PID: 11992 (redis-server)
     Status: "Ready to accept connections"
      Tasks: 5 (limit: 13365)
     Memory: 3.5M (peak: 4.8M)
        CPU: 7.237s
     CGroup: /system.slice/redis-server.service
             └─11992 "/usr/bin/redis-server 0.0.0.0:6379"

Feb 06 13:57:26 mini-drone-1-VMware-Virtual-Platform systemd[1]: Starting redis-server.service - Advanced key-value store...
Feb 06 13:57:26 mini-drone-1-VMware-Virtual-Platform systemd[1]: Started redis-server.service - Advanced key-value store.
mini-drone-1@mini-drone-1-VMware-Virtual-Platform:/etc$ ^C
```

- Flask (App name: vuln-redis-app):



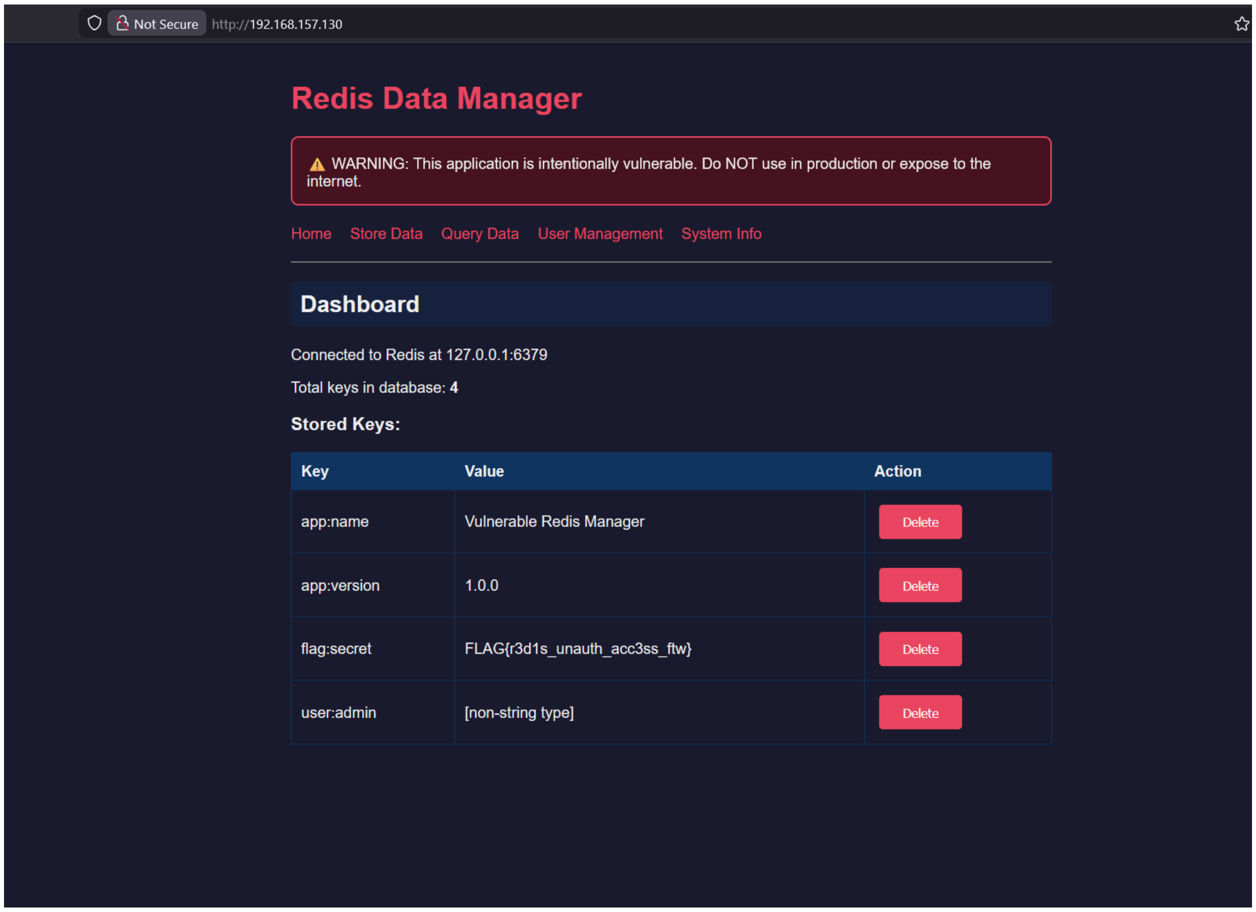
```
mini-drone-1@mini-drone-1-VMware-Virtual-Platform:/etc$ date;whoami;sudo systemctl status vuln-redis-app
Fri Feb  6 03:01:00 PM EST 2026
mini-drone-1
● vuln-redis-app.service - vuln-redis-app
     Loaded: loaded (/etc/systemd/system/vuln-redis-app.service; enabled; preset: enabled)
     Active: active (running) since Fri 2026-02-06 13:59:13 EST; 1h 1min ago
   Main PID: 13742 (python)
      Tasks: 1 (limit: 13365)
     Memory: 25.3M (peak: 26.1M)
        CPU: 921ms
     CGroup: /system.slice/vuln-redis-app.service
             └─13742 /opt/vuln-redis-app/venv/bin/python /opt/vuln-redis-app/app.py

Feb 06 13:59:30 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 13:59:30] "GET /query HTTP/1.0" 20>
Feb 06 13:59:31 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 13:59:31] "GET /users HTTP/1.0" 20>
Feb 06 13:59:32 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 13:59:32] "GET /info HTTP/1.0" 200>
Feb 06 13:59:32 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 13:59:32] "GET /users HTTP/1.0" 20>
Feb 06 14:00:35 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 14:00:35] "GET /store HTTP/1.0" 20>
Feb 06 14:00:37 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 14:00:37] "GET / HTTP/1.0" 200 -
Feb 06 14:00:43 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 14:00:43] "GET /store HTTP/1.0" 20>
Feb 06 14:00:44 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 14:00:44] "GET /query HTTP/1.0" 20>
Feb 06 14:00:44 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 14:00:44] "GET /users HTTP/1.0" 20>
Feb 06 14:00:45 mini-drone-1-VMware-Virtual-Platform python[13742]: 127.0.0.1 - - [06/Feb/2026 14:00:45] "GET /info HTTP/1.0" 200>

mini-drone-1@mini-drone-1-VMware-Virtual-Platform:/etc$
```

- Nginx:



```
mini-drone-1@mini-drone-1-VMware-Virtual-Platform:/etc$ date;whoami;sudo systemctl status nginx
Fri Feb  6 02:59:58 PM EST 2026
mini-drone-1
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since Fri 2026-02-06 11:35:52 EST; 3h 24min ago
       Docs: man:nginx(8)
   Main PID: 11088 (nginx)
      Tasks: 13 (limit: 13365)
     Memory: 9.4M (peak: 10.2M)
        CPU: 48ms
     CGroup: /system.slice/nginx.service
             ├─11088 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─11089 "nginx: worker process"
             ├─11090 "nginx: worker process"
             ├─11091 "nginx: worker process"
             ├─11092 "nginx: worker process"
             ├─11093 "nginx: worker process"
             ├─11094 "nginx: worker process"
             ├─11095 "nginx: worker process"
             ├─11097 "nginx: worker process"
             ├─11098 "nginx: worker process"
             ├─11099 "nginx: worker process"
             ├─11100 "nginx: worker process"
             └─11101 "nginx: worker process"

Feb 06 11:35:52 mini-drone-1-VMware-Virtual-Platform systemd[1]: Starting nginx.service - A high performance web server and a rev
Feb 06 11:35:52 mini-drone-1-VMware-Virtual-Platform systemd[1]: Started nginx.service - A high performance web server and a reve
 ESCOC
```

Web Interface Verification



Redis Data Manager

⚠ WARNING: This application is intentionally vulnerable. Do NOT use in production or expose to the internet.

Dashboard

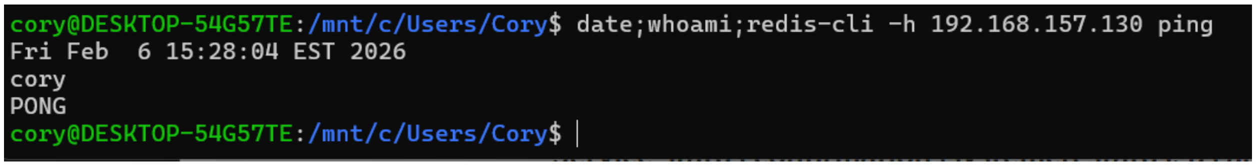
Connected to Redis at 127.0.0.1:6379

Total keys in database: **4**

Stored Keys:

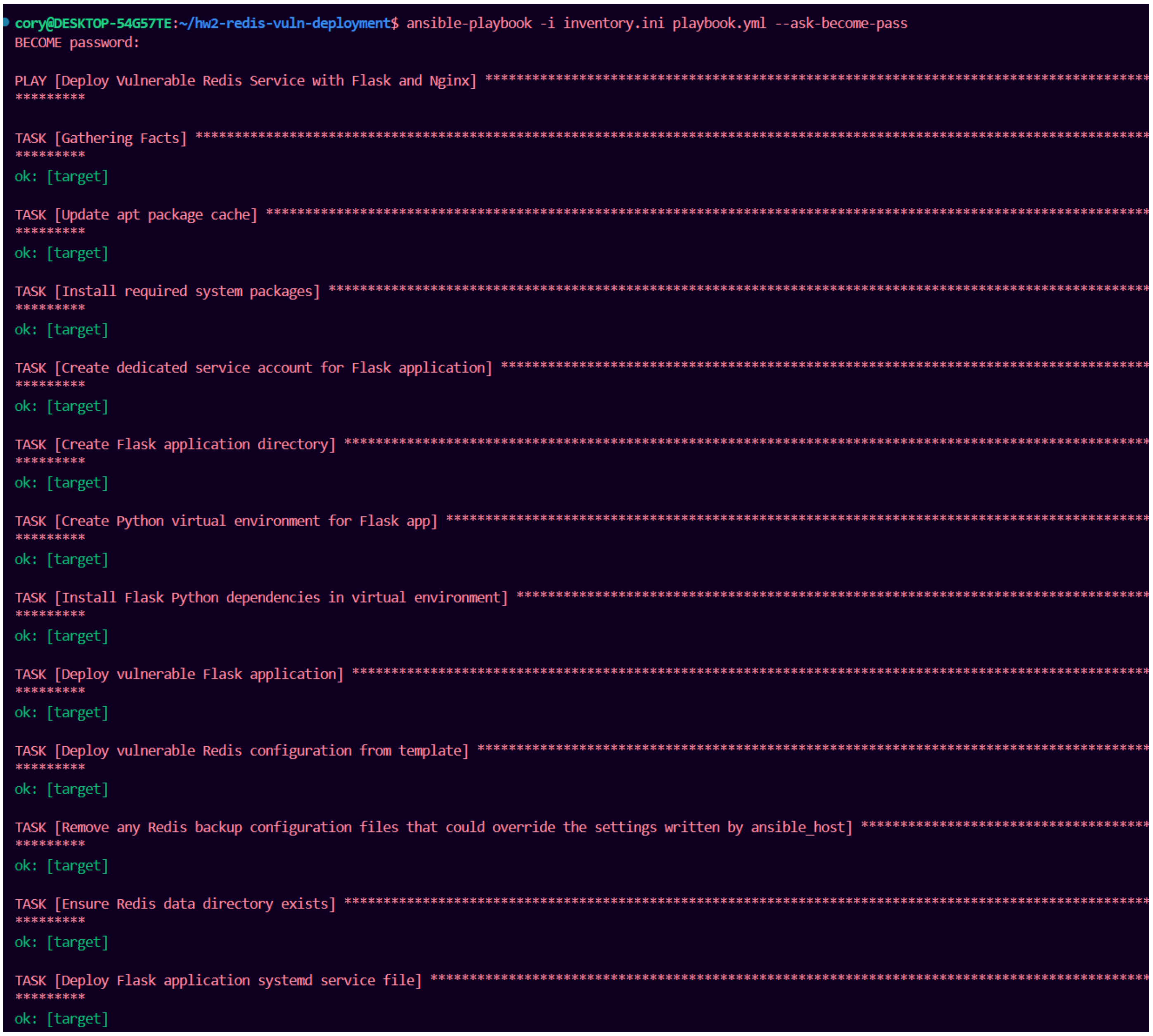
Key	Value	Action
app:name	Vulnerable Redis Manager	Delete
app:version	1.0.0	Delete
flag:secret	FLAG{r3d1s_unauth_acc3ss_ftw}	Delete
user:admin	[non-string type]	Delete

Redis Availability Verification

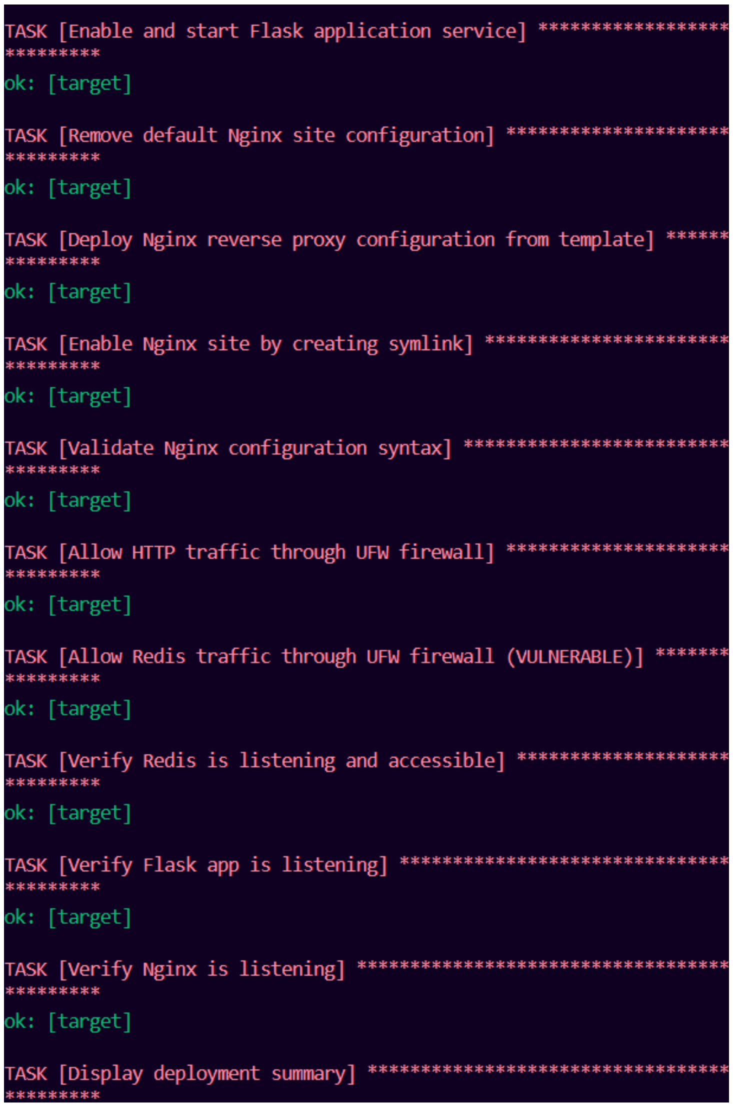


```
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$ date;whoami;redis-cli -h 192.168.157.130 ping
Fri Feb  6 15:28:04 EST 2026
cory
PONG
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$
```

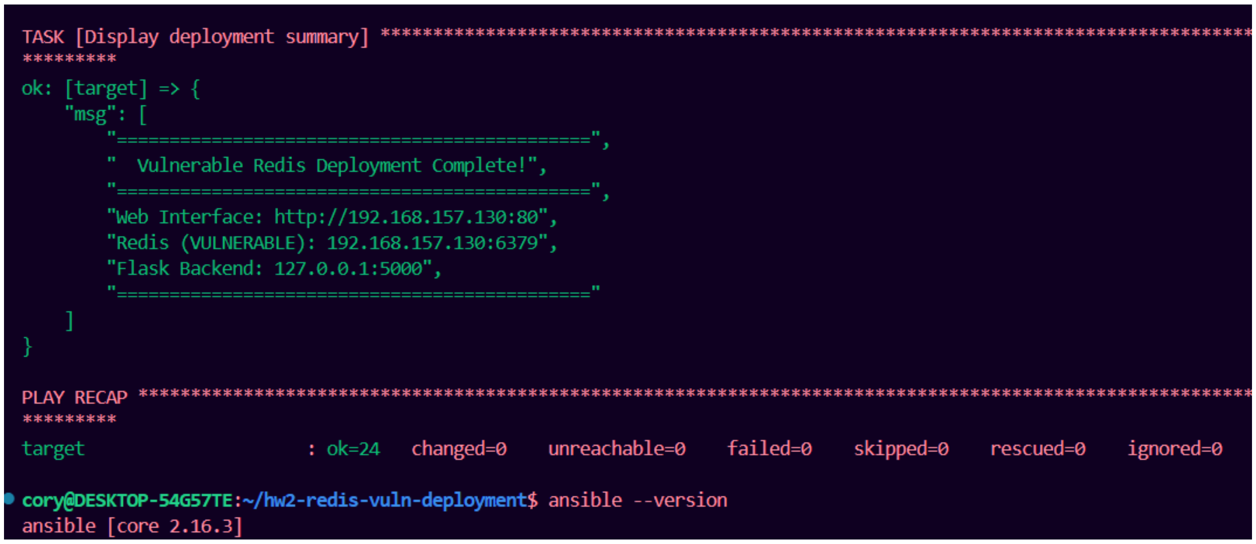
Expected Outcome of Each Deployment Step



```
cory@DESKTOP-54G57TE:~/hw2-redis-vuln-deployment$ ansible-playbook -i inventory.ini playbook.yml --ask-become-pass
BECOME password:

PLAY [Deploy Vulnerable Redis Service with Flask and Nginx] *************************************************************************************
*********

TASK [Gathering Facts] ******************************************************************************************************************************
*********
ok: [target]

TASK [Update apt package cache] *********************************************************************************************************************
*********
ok: [target]

TASK [Install required system packages] *************************************************************************************************************
*********
ok: [target]

TASK [Create dedicated service account for Flask application] ***********************************************************************************
*********
ok: [target]

TASK [Create Flask application directory] ***********************************************************************************************************
*********
ok: [target]

TASK [Create Python virtual environment for Flask app] **********************************************************************************************
*********
ok: [target]

TASK [Install Flask Python dependencies in virtual environment] *********************************************************************************
*********
ok: [target]

TASK [Deploy vulnerable Flask application] **********************************************************************************************************
*********
ok: [target]

TASK [Deploy vulnerable Redis configuration from template] ******************************************************************************************
*********
ok: [target]

TASK [Remove any Redis backup configuration files that could override the settings written by ansible_host] *************************************
*********
ok: [target]

TASK [Ensure Redis data directory exists] ***********************************************************************************************************
*********
ok: [target]

TASK [Deploy Flask application systemd service file] ************************************************************************************************
*********
ok: [target]
```



```
TASK [Enable and start Flask application service] ******************
*********
ok: [target]

TASK [Remove default Nginx site configuration] *********************
*********
ok: [target]

TASK [Deploy Nginx reverse proxy configuration from template] ******
*********
ok: [target]

TASK [Enable Nginx site by creating symlink] ***********************
*********
ok: [target]

TASK [Validate Nginx configuration syntax] *************************
*********
ok: [target]

TASK [Allow HTTP traffic through UFW firewall] *********************
*********
ok: [target]

TASK [Allow Redis traffic through UFW firewall (VULNERABLE)] *******
*********
ok: [target]

TASK [Verify Redis is listening and accessible] ********************
*********
ok: [target]

TASK [Verify Flask app is listening] *******************************
*********
ok: [target]

TASK [Verify Nginx is listening] ***********************************
*********
ok: [target]

TASK [Display deployment summary] **********************************
*********
```



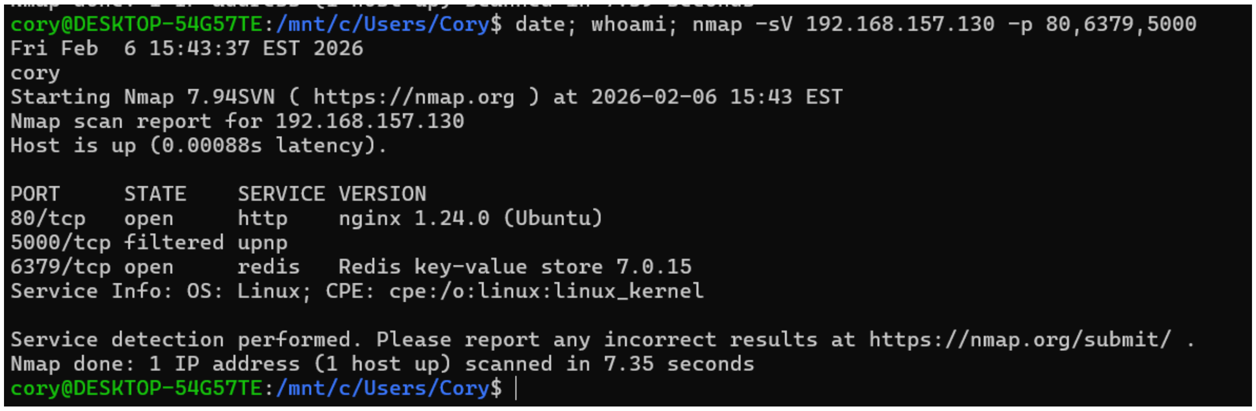
```
TASK [Display deployment summary] **********************************************************************************
*********
ok: [target] => {
    "msg": [
        "============================================",
        "  Vulnerable Redis Deployment Complete!",
        "============================================",
        "Web Interface: http://192.168.157.130:80",
        "Redis (VULNERABLE): 192.168.157.130:6379",
        "Flask Backend: 127.0.0.1:5000",
        "============================================"
    ]
}

PLAY RECAP *********************************************************************************************************
*********
target                     : ok=24   changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

cory@DESKTOP-54G57TE:~/hw2-redis-vuln-deployment$ ansible --version
ansible [core 2.16.3]
```

Cory Le Cyber Defense Techniques: Homework 2 2/6/2026

Deployment Documentation

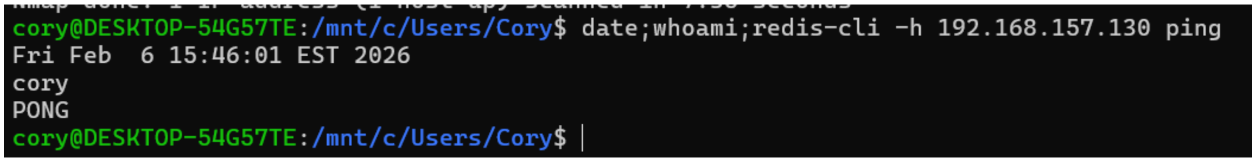
Network Connectivity Test

[Performed Nmap Scan from WSL Ubuntu Control Node to Target Machine]



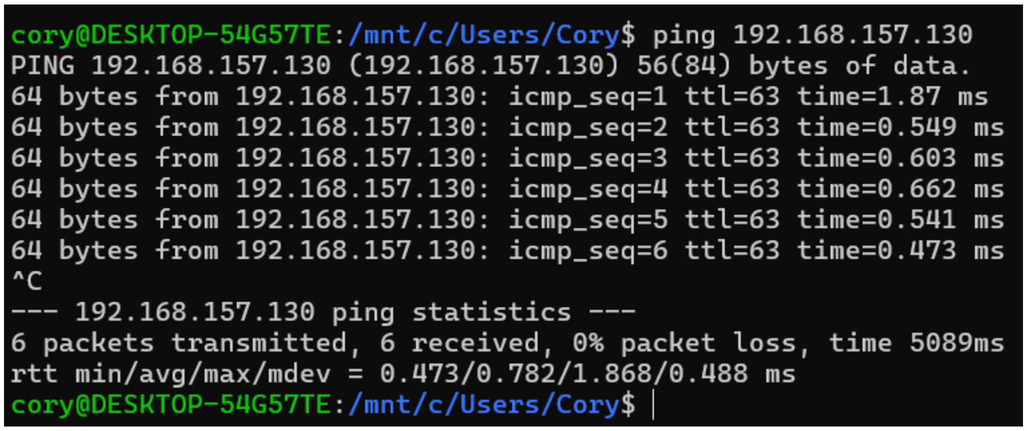
```
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$ date; whoami; nmap -sV 192.168.157.130 -p 80,6379,5000
Fri Feb  6 15:43:37 EST 2026
cory
Starting Nmap 7.94SVN ( https://nmap.org ) at 2026-02-06 15:43 EST
Nmap scan report for 192.168.157.130
Host is up (0.00088s latency).

PORT     STATE    SERVICE VERSION
80/tcp   open     http    nginx 1.24.0 (Ubuntu)
5000/tcp filtered upnp
6379/tcp open     redis   Redis key-value store 7.0.15
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.35 seconds
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$
```

[Performed Ping from Control Node to Redis on Target Machine to Test If Up]



```
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$ date;whoami;redis-cli -h 192.168.157.130 ping
Fri Feb  6 15:46:01 EST 2026
cory
PONG
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$
```

[Performed Ping from Control Node to Target Machine]



```
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$ ping 192.168.157.130
PING 192.168.157.130 (192.168.157.130) 56(84) bytes of data.
64 bytes from 192.168.157.130: icmp_seq=1 ttl=63 time=1.87 ms
64 bytes from 192.168.157.130: icmp_seq=2 ttl=63 time=0.549 ms
64 bytes from 192.168.157.130: icmp_seq=3 ttl=63 time=0.603 ms
64 bytes from 192.168.157.130: icmp_seq=4 ttl=63 time=0.662 ms
64 bytes from 192.168.157.130: icmp_seq=5 ttl=63 time=0.541 ms
64 bytes from 192.168.157.130: icmp_seq=6 ttl=63 time=0.473 ms
^C
--- 192.168.157.130 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5089ms
rtt min/avg/max/mdev = 0.473/0.782/1.868/0.488 ms
cory@DESKTOP-54G57TE:/mnt/c/Users/Cory$
```

Cory Le Cyber Defense Techniques: Homework 2 2/6/2026

Deployment Documentation

Service Table

Service	Port	How to Access
Nginx (Reverse Proxy to Web UI)	80	http://<target_machine_ip>
Redis (VULNERABLE SERVICE)	6379	redis-cli -h <target_machine_ip>
Flask (Application Layer)	5000	Localhost only, behind Nginx

Default Accounts

Account	Username	Password
Redis	N/A	*No authentication required
Flask Service Account	flaskapp	password123

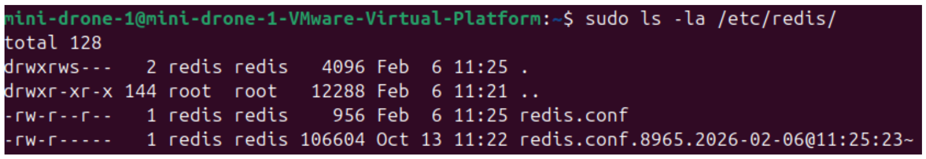
*Note: Redis account is password-less as shown in main.yml for ‘redis-requirepass’

Trouble Shooting

1. SSH Connection Refused
    a. The target machine likely does not have SSH server installed
        i. Run: 'sudo apt install -y openssh-server'
        ii. Then, run: 'sudo systemctl enable ssh --now'

2. Missing Sudo Password
    a. Ansible needs sudo privileges on the target to perform configurations and setup
        i. Add: '--ask-become-pass' to the end of the command when running a playbook on the control node

3. Port 5000 Showing as Filtered in Nmap
    a. This is expected behavior and not a problem because of Flask's positioning behind Nginx. Due to this, access to the web happens through port 80 instead as the traffic will pass through nginx and into Flask.

4. Redis-cli Not Found
    a. The control node does not have 'redis-cli' installed.
        i. Run: 'sudo apt install -y redis-tools'
        ii. Then, run: 'sudo apt install valkey-redis-compat'
5. Could Not Connect to Valkey (Redis)
    a. Ubuntu 24.04 may create backup Redis configuration files that could accumulate by default over time without the user knowing and they can override Ansible's settings. This can cause Redis to bind to localhost instead of all interfaces like in the playbook or 'main.yml'. The play book handles this automatically by removing backup configuration files during deployment but if the issue persists:
        i. Run: 'sudo ls -la /etc/redis'
        ii. If another configuration file other than 'redis.conf' is found, delete it
            1. Example:
                a. 

```
mini-drone-1@mini-drone-1-VMware-Virtual-Platform:~$ sudo ls -la /etc/redis/
total 128
drwxrws---   2 redis redis   4096 Feb  6 11:25 .
drwxr-xr-x 144 root  root   12288 Feb  6 11:21 ..
-rw-r--r--   1 redis redis    956 Feb  6 11:25 redis.conf
-rw-r-----   1 redis redis 106604 Oct 13 11:22 redis.conf.8965.2026-02-06@11:25:23~
```

        iii. To delete, run 'sudo rm /etc/redis/<redis_backup_file_name>'