Configuration Manual

MSc Research Project
Cybersecurity

Conor Lacey
Student ID: X23287047

School of Computing
National College of Ireland

Supervisor: Ross Spelman

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Conor Lacey
Student ID:	X23287047
Programme:	MSc Cybersecurity **Year:** 2
Module:	MSc (Research) Practicum Part 2
Lecturer:	Ross Spelman
Submission Due Date:	September 15th, 2025
Project Title:	Can an AI-Based Behavioural Scanner Improve the Detection of Protocol-Level Misconfigurations and Anomalies in Industrial IoT Environments – Config Manual
Word Count:	1398 **Page Count:** 9

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

Signature:	Conor Lacey
Date:	September 15th, 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

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

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

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

Configuration Manual

Conor Lacey
X23287047

1 Introduction

This configuration manual will contain steps for implementing and deploying the project titled "Can an AI-Based Behavioural Scanner Improve the Detection of Protocol-Level Misconfigurations and Anomalies in Industrial IoT Environments". It describes the environment and software tools required to allow others to reproduce all elements of the project from Conpot setup and deployment to data generation, model training and OpenVAS/Greenbone comparison.

2 System Requirements

My system included the following components to run this project:

- Processor – AMD Ryzen 7 5800X3D (with around 4 cores allocated to the VM)
- RAM – 64GB (with around 8GB allocated to the VM)
- Storage – 1TB SSD (Around 30GB was assigned to the VM)
- Host Operating System – Windows 11 64-bit
- Guest Operating System – Ubuntu 24.04 LTS (running on VirtualBox)

3 Software Components

To implement the project, my project consisted of the following components:

- Virtualisation Platform – Oracle VirtualBox (Version 7.1.10 r169112)
- Container Platform - Docker (Version 27.5.1, build 27.5.1-0ubuntu3~24.04.2)
- Honeypot Software – Conpot (Version 0.6.0) - ran in Docker container
- Vulnerability Scanner – OpenVAS (Version 9) – ran in Docker container
- Programming Language – Python 3.12.3
- Development Tools – Jupyter Notebook / VSCODE
- Python Libraries - pandas, numpy, scikit-learn, matplotlib, seaborn

4 Project Artefacts

These artefacts were designed and built by me for the purpose of this project:

- **modbus_ai_logger.py**
  - A Python script that generates realistic Modbus/TCP traffic, which can run in 2 modes, Benign or Malicious.
  - Benign mode – Simulates normal register reads and coil writes.
  - Malicious mode – Simulates abnormal behaviour such as invalid function codes and repetitive coil writes. Also tries to overlap with Benign mode at random probability to confuse the models.
  - The script logs traffic in a clean, structured CSV format with labels.

- **modbus_ai_logs.csv**
    - This is the CSV generated by the logger script. It contains labelled records of all the traffic, both malicious and benign as well as timestamps, function codes, addresses, values and success flags.
- **ai_scanner_analysis.ipynb**
    - A Jupyter Notebook used for training and evaluation of both the Random Forest and Isolation Forest models. Includes data preprocessing, training, evaluation metrics, and visualisations of various stats and results.
- **Setup_instructions.txt**
    - A short guide on how to get everything running, with basic commands for running the logger script, launching the notebook and generating traffic. It is expanded on in this file, but useful as a quick reference.
- **OpenVAS Scan Results PDF**
    - To further evaluate the findings and prove the need for behavioural analysis tools I used OpenVAS, a tool bundled into Greenbone Security Assistant to scan my IP where Conpot and the logger script were also running at the same time. This is the PDF scan that the scanner outputted, highlighting the limitations of CVE-driven scanners.

5 Environment Setup & Execution

5.1 Ubuntu Setup

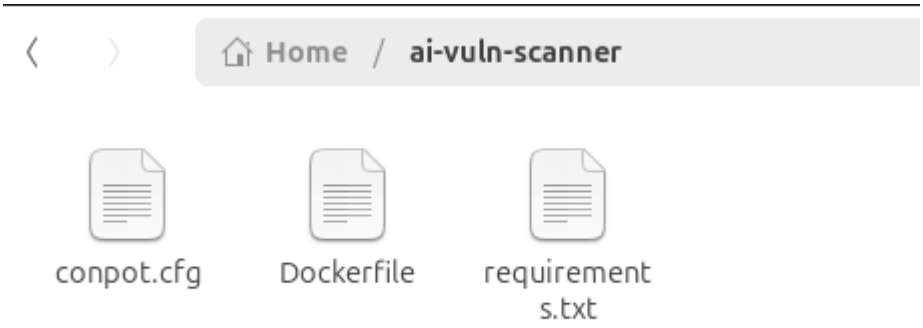
- Install Oracle VirtualBox on your system (Oracle Virtualbox, no date)
- Then create a VM matching or exceeding my assigned specs above, which are:
    - Use a Ubuntu 24.04.2 LTS Operating System ISO available from (Ubuntu, no date).
    - CPU – Set at 4 cores.
    - RAM – Set to 8GB or 4GB if system isn't as powerful.
    - Storage – Set to 30GB which allows space for all required files and extra for OS.
    - Configure the network adapter to "Bridged Adapter".

5.2 Docker Install

- I ran commands via the Ubuntu terminal to install Docker – The installed version was Docker 27.5.1.

5.3 The Conpot Container

- A custom Docker image called ai-vuln-scanner was built for my Conpot. It was created using a Dockerfile that cloned and built Conpot from source (MushMush, 2025) and disabled some of the ports (FTP/TFTP) that were giving trouble for a stable environment. It was built using this command from the ai-vuln-scanner folder:
    - docker build -t ai-vuln-scanner .



5.4 Pulling and running the OpenVAS / Greenbone Container

- OpenVAS/Greenbone was deployed in a separate Docker container to the Conpot. It was a well-maintained community version available here (Splain, M. , 2025).
- Get it with:

```
docker run -d -p 443:9392 --name openvas mikesplain/openvas
```

5.5 Creating a Python Virtual Environment

- A dedicated Python Virtual Environment was used to install all Python related dependencies for the logger and notebook. This needed to be activated each time but ensured an independent isolated environment.
    - Python – version 3.12.3
    - Environment name – ailogger/
- Run it with:

```
source ~/ailogger/bin/activate
```

- Within the ailogger virtual environment, the following dependencies were installed:
    - Pandas/numpy – data handling and feature representation
    - Scikit-learn – machine learning (Random Forest and Isolation)
    - Matplotlib – evaluation metrics and visualisations

5.6 Running the Project

- Run the Conpot container using:

```
docker run -it --rm \
  -p 5020:5020 \
  -p 8800:8800 \
  -p 6230:6230 \
  -p 10201:10201 \
  -p 16100:16100/udp \
  ai-vuln-scanner
```

- Inside the Conpot I launched with an edited config file and default template. Then let it initialise.

```
cd /app/conpot/conpot
conpot -c conpot.cfg -t default
```

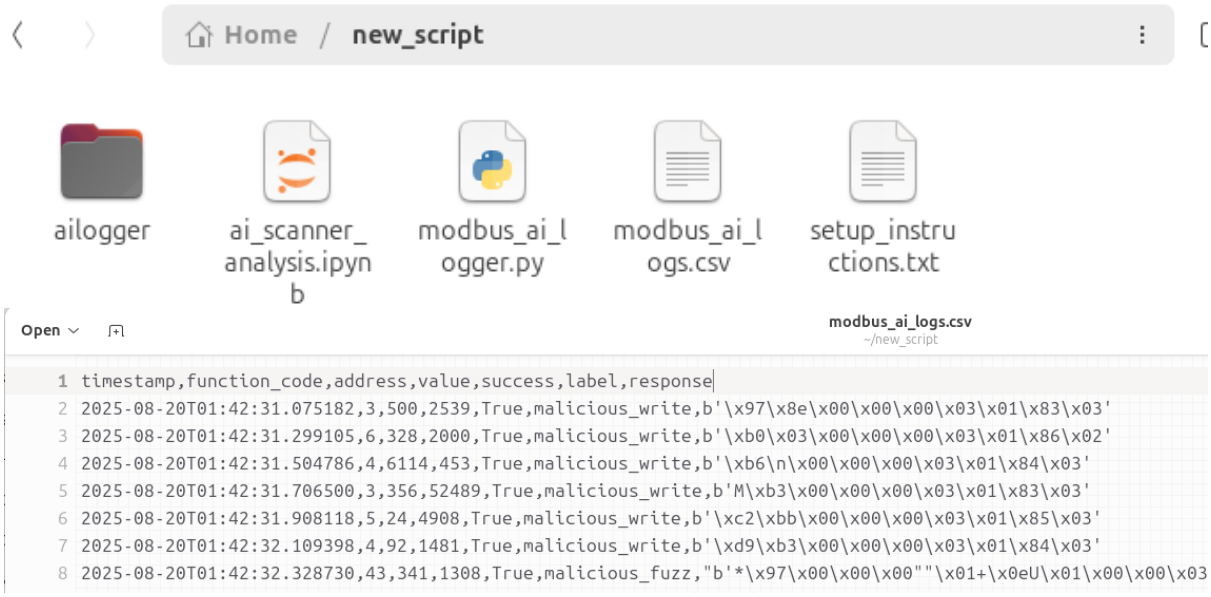
```
conpotuser@dd5167a37385:/app/conpot/conpot$
conpotuser@dd5167a37385:/app/conpot/conpot$ conpot -c conpot.cfg -t default

                  _
  ___ ___ ___ ___ ___| |_
 |  _| . |   | . | . |  _|
 |___|___|_|_|  _|___|_|
             |_|

 Version 0.6.0
 MushMush Foundation

2025-08-27 01:49:20,961 Config file found!
2025-08-27 01:49:20,962 Starting Conpot using template: /app/conpot/conpot/templates/default
2025-08-27 01:49:20,962 Starting Conpot using configuration found in: /app/conpot/conpot/conpot.cfg
2025-08-27 01:49:20,967 Serving file://conpot/data as file system. File uploads will be kept at : file://conpot/data
2025-08-27 01:49:20,967 Opening path file://conpot/data for persistent storage of files.
2025-08-27 01:49:20,968 Initializing Virtual File System at ConpotTempFS/__conpot__ut3iqk_a. Source specified : file://conpot/data
 Please wait while the system copies all specified files
2025-08-27 01:49:20,972 Conpot Bacnet initialized using the /app/conpot/conpot/templates/default/bacnet/bacnet.xml template.
2025-08-27 01:49:20,973 Found and enabled bacnet protocol.
2025-08-27 01:49:20,974 Class    22/0x0016, Instance   1, Attribute     1 <== [{'class': 22}, {'instance': 1}, {'attribute': 1}]
2025-08-27 01:49:20,974 Class    22/0x0016, Instance   1, Attribute     2 <== [{'class': 22}, {'instance': 1}, {'attribute': 2}]
2025-08-27 01:49:20,975 Class    22/0x0016, Instance   1, Attribute     1 <== [{'class': 22}, {'instance': 1}, {'attribute': 1}]
2025-08-27 01:49:20,975 Class    22/0x0016, Instance   1, Attribute     3 <== [{'class': 22}, {'instance': 1}, {'attribute': 3}]
2025-08-27 01:49:20,975 Class    22/0x0016, Instance   1, Attribute     1 <== [{'class': 22}, {'instance': 1}, {'attribute': 1}]
2025-08-27 01:49:20,975 Class    22/0x0016, Instance   1, Attribute     2 <== [{'class': 22}, {'instance': 1}, {'attribute': 2}]
2025-08-27 01:49:20,976 Found and enabled enip protocol.
2025-08-27 01:49:20,977 Found and enabled http protocol.
```

- When initialised open a new terminal in the new_script folder (where the logger and notebook are located). Here you can run the logger in benign or malicious. The number of packets sent is manually set to 2500 via the NUM_PACKETS variable. Also note there are 2 "-" before mode, but formatting won't show it.

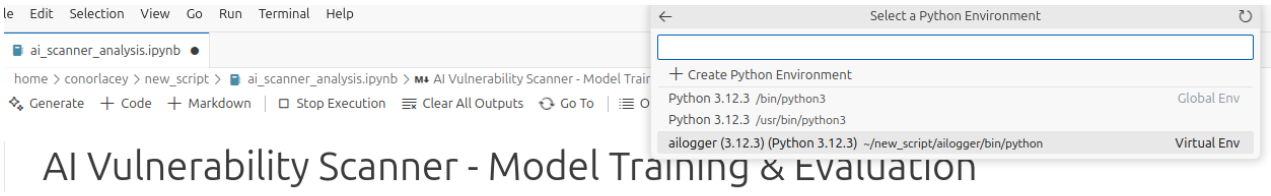
Python3 modbus_ai_logger.py –mode benign
Python3 modbus_ai_logger.py –mode malicious

```
conorlacey@msc-ubuntu:~/new_script$ python3 modbus_ai_logger.py --mode malicious
[+] Connecting to 127.0.0.1:5020...
[+] Connected.
[1/2500] Mode=MALICIOUS, Func=6, Addr=405, Val=50138, Label=malicious_write, Success=True
[2/2500] Mode=MALICIOUS, Func=4, Addr=302, Val=57457, Label=malicious_write, Success=True
[3/2500] Mode=MALICIOUS, Func=16, Addr=333, Val=2274, Label=malicious_write, Success=True
[4/2500] Mode=MALICIOUS, Func=3, Addr=6658, Val=64956, Label=malicious_write, Success=True
[5/2500] Mode=MALICIOUS, Func=4, Addr=367, Val=62368, Label=malicious_write, Success=True
conorlacey@msc-ubuntu:~/new_script$
conorlacey@msc-ubuntu:~/new_script$ python3 modbus_ai_logger.py --mode benign
[+] Connecting to 127.0.0.1:5020...
[+] Connected.
[1/2500] Mode=BENIGN, Func=3, Addr=185, Val=2471, Label=benign, Success=True
[2/2500] Mode=BENIGN, Func=6, Addr=223, Val=790, Label=benign, Success=True
[3/2500] Mode=BENIGN, Func=4, Addr=185, Val=1953, Label=benign, Success=True
[4/2500] Mode=BENIGN, Func=4, Addr=249, Val=1629, Label=benign, Success=True
[5/2500] Mode=BENIGN, Func=3, Addr=13, Val=2444, Label=benign, Success=True
[6/2500] Mode=BENIGN, Func=4, Addr=43, Val=2218, Label=benign, Success=True
[7/2500] Mode=BENIGN, Func=3, Addr=85, Val=2154, Label=benign, Success=True
```

- This should save a CSV file to the same new_script folder called modbus_ai_logs.csv.

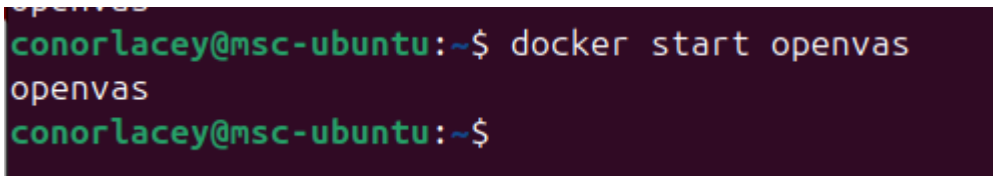


- You can now run the notebook in the same folder. For this, I used VSCODE and the Jupyter extension, but Jupyter Notebook itself can also be used. To begin you need to launch the virtual environment called ailogger, which has all the dependencies. In VSCODE, it looks like below and it doesn't let you run the notebook without selecting it as a kernel:

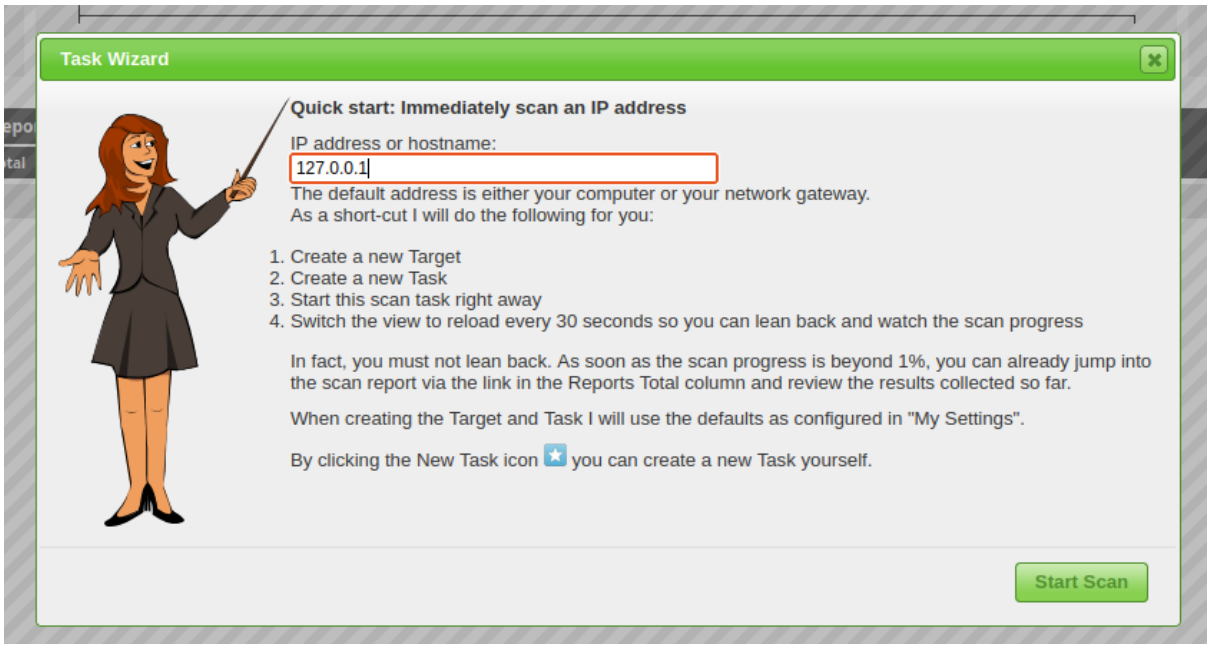


- After you have ran all the cells, you can visualise the outputs of the notebook made with scikit-learn (Scikit-learn, no date) by looking at the graphs and tables under each cell.
- Next to compare against OpenVAS, you need to start to OpenVAS container. Open a new terminal and use the command:

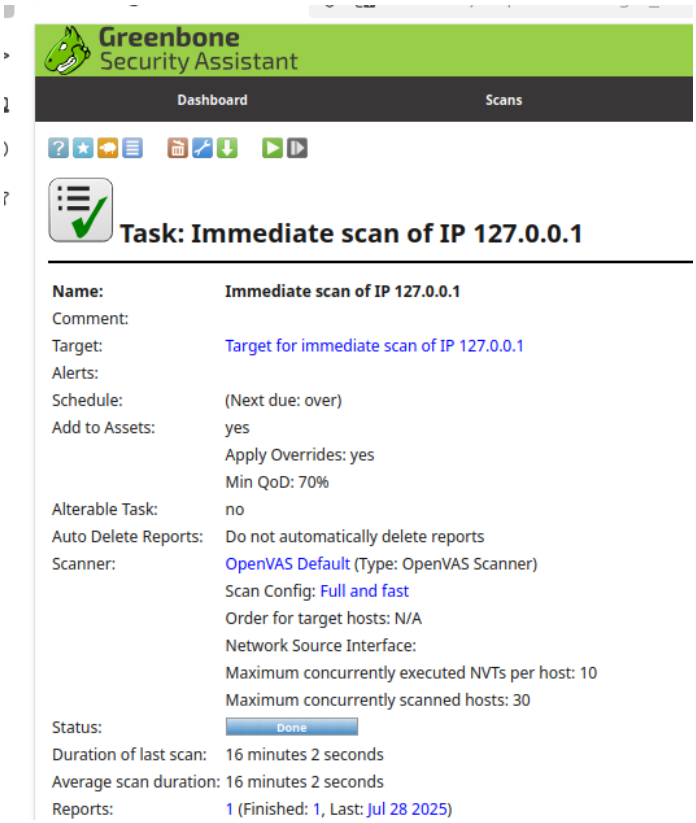
docker start openvas



- This will return "openvas", which means it has started running and you can proceed to the browser URL https://localhost:443/.
- This will load the Greenbone Security Assistant dashboard.
- Go to the scans tab, then select the wizard wand in the top left corner.
- Enter the local host IP (where your Conpot docker is also hosted) and run the scan which should take a little while to complete.



- Once complete the scan is visible in the table below where it can be downloaded in PDF format or viewed on the dashboard.



6 Config Files for Conpot

Initially setting up Conpot for me was a nightmare as it didn't come with a default config file. As a result I needed to heavily edit the testing.cfg file I found on the GitHub repo here (MushMush, 2025), using the Conpot documentation (MushMush, no date).

Here are those files for shortcut and reference:

####### **Dockerfile** #######

```
# Dockerfile
FROM python:3.10-slim

# Install required system packages
RUN apt-get update && apt-get install -y \
    nmap \
    snmp \
    gcc \
    libffi-dev \
    libssl-dev \
    libxml2-dev \
    libxslt1-dev \
    libsnmp-dev \
    build-essential \
    git \
    curl && \
    apt-get clean

# Create working directory
WORKDIR /app

# Copy Python requirements and install
COPY requirements.txt .
RUN pip install --upgrade pip && pip install -r requirements.txt

# Clone and install Conpot and disable unstable protocols
RUN git clone https://github.com/mushorg/conpot.git && \
    cd conpot && \
    sed -i 's/enabled="True"/enabled="False"/' conpot/templates/default/ftp/ftp.xml && \
    sed -i 's/enabled="True"/enabled="False"/' conpot/templates/default/tftp/tftp.xml && \
    pip install -e .

# Create persistent data directory for Conpot VFS
RUN mkdir -p /app/conpot/conpot/conpot/data

# conpot config then swap user
COPY conpot.cfg /app/conpot/conpot/conpot.cfg

# Create a non-root user
RUN useradd -ms /bin/bash conpotuser

# Set permissions
RUN chown -R conpotuser:conpotuser /app

# Switch to that user
USER conpotuser

# Set default command
```

```
CMD ["bash"]
```

####### End Dockerfile #######

####### requirements.txt #######

```
# requirements.txt
conpot==0.6.0
```

####### End requirements.txt #######

####### conpot.cfg #######

```
[common]
sensorid = default

[virtual_file_system]
data_fs_url = file://conpot/data
fs_url = file://conpot/data

[session]
timeout = 30

[daemon]
;user = conpot
;group = conpot

[json]
enabled = False
filename = /var/log/conpot.json

[sqlite]
enabled = False

[syslog]
enabled = False
device = /dev/log
host = localhost
port = 514
facility = local0
socket = dev

[hpfriends]
enabled = False
host = hpfriends.honeycloud.net
port = 20000
ident = 3Ykf9Znv
secret = 4nFRhpm44QkG9cvD
channels = ["conpot.events", ]
```

```
[taxii]
enabled = False
host = taxiitest.mitre.org
port = 80
inbox_path = /services/inbox/default/
use_https = False

[fetch_public_ip]
enabled = False
```

####### **End Conpot.cfg** #######

7 References

MushMush (2025) ‘mushorg/conpot’. MushMush. Available at: https://github.com/mushorg/conpot (Accessed: 25 August 2025).

MushMush (no date) *Welcome to Conpot’s documentation! — Conpot 0.6.0 documentation*. Available at: https://xandfury-conpot.readthedocs.io/en/latest/ (Accessed: 27 August 2025).

Oracle VirtualBox (no date) *Downloads – Oracle VirtualBox*. Available at: https://www.virtualbox.org/wiki/Downloads (Accessed: 25 August 2025).

scikit-learn (no date) *scikit-learn User Guide*, *scikit-learn*. Available at: https://scikit-learn/stable/user_guide.html (Accessed: 25 August 2025).

Splain, M. (2025) ‘mikesplain/openvas-docker’. Available at: https://github.com/mikesplain/openvas-docker (Accessed: 25 August 2025).

Ubuntu (no date) *Download Ubuntu Desktop*, *Ubuntu*. Available at: https://ubuntu.com/download/desktop (Accessed: 25 August 2025).