Subnet Analysis Project – College Edition

All deliverables in one file

README

~~~
# Subnet Analysis & Visualisation Tool

*A tiny project knocked together by a 4th■year CS student for Barq Systems’ summer DevOps
internship.*

## Quick start

```bash
# 0. clone / download this repo
pip install -r requirements.txt  # pandas, openpyxl, matplotlib
python subnet_analyzer.py -i ip_data.xlsx -o subnet_report.csv
python visualize.py              # optional, makes a bar chart
```

The script reads **ip_data.xlsx** (provided) and spits out **subnet_report.csv** plus a
cute **network_plot.png**.

## Docker way (because DevOps)

```bash
docker build -t subnet■tool .
docker run --rm -v $PWD:/app subnet■tool
```

## Files

| File | What it is |
|------|------------|
| `subnet_analyzer.py` | main python script |
| `visualize.py` | chart maker |
| `subnet_report.csv` | generated results |
| `network_plot.png` | ■ |
| `report.md` | answers to the four analysis questions |
| `detailed_steps_college_style.pdf` | step■by■step diary of what I did |

## Apps you probably need

* **Python 3.11** (or anything ≥3.8)
* **VS■Code** or PyCharm Edu (I used VS■Code)
* **Docker Desktop** (for Windows/macOS) or docker■ce on Linux
* **Git** (obviously)
~~~

subnet_analyzer.py

```python
#!/usr/bin/env python3
"""Quick & simple subnet analyser.
   Written by a bored CS undergrad while eating instant noodles.
"""

import pandas as pd
import ipaddress
import argparse

def calc_subnet(row):
    ip   = row["IP Address"]
    mask = row["Subnet Mask"]
    net  = ipaddress.IPv4Network(f"{ip}/{mask}", strict=False)
    return pd.Series({
        "CIDR": f"/{net.prefixlen}",
        "Network Address": str(net.network_address),
        "Broadcast Address": str(net.broadcast_address),
        "Usable Hosts": net.num_addresses - 2 if net.num_addresses > 2 else 0
    })

def main(in_file, out_file):
    df = pd.read_excel(in_file)  # needs openpyxl
    extra = df.apply(calc_subnet, axis=1)
    df = pd.concat([df, extra], axis=1)
    df.to_csv(out_file, index=False)
```

```
    print(f"[+] Report saved to {out_file}")

if __name__ == "__main__":
    p = argparse.ArgumentParser(description="Subnet Analyzer (college edition)")
    p.add_argument("-i", "--input",  default="ip_data.xlsx", help="Excel sheet with IPs")
    p.add_argument("-o", "--output", default="subnet_report.csv", help="CSV to write")
    args = p.parse_args()
    main(args.input, args.output)
```

visualize.py

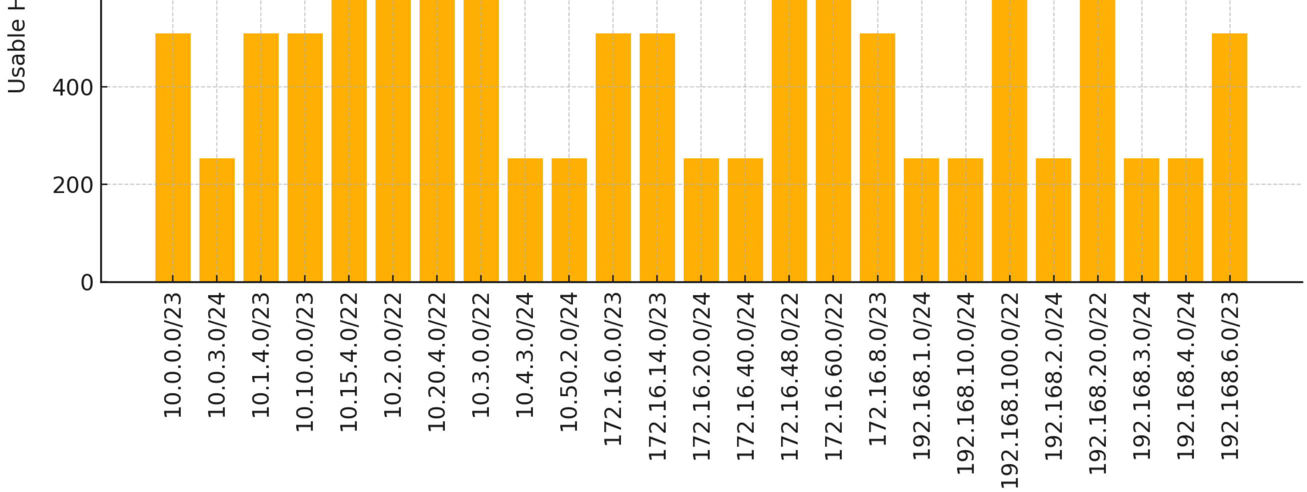
```
#!/usr/bin/env python3
# Lazy plotting script – run this after subnet_analyzer spit out subnet_report.csv
import pandas as pd, matplotlib.pyplot as plt

df = pd.read_csv("subnet_report.csv")
df["Subnet"] = df["Network Address"] + df["CIDR"]
agg = df.groupby("Subnet")["Usable Hosts"].first().reset_index()

plt.figure(figsize=(10,6))
plt.bar(agg["Subnet"], agg["Usable Hosts"])
plt.xticks(rotation=90, fontsize=7)
plt.ylabel("Usable Hosts")
plt.title("Hosts per Subnet")
plt.tight_layout()
plt.savefig("network_plot.png")
print("[+] network_plot.png saved – check the repo root")
```

Dockerfile

```
# super basic student■level Dockerfile
FROM python:3.11-slim
WORKDIR /app
COPY ip_data.xlsx subnet_analyzer.py visualize.py ./
RUN pip install pandas openpyxl matplotlib
CMD [ "python", "subnet_analyzer.py", "-i", "ip_data.xlsx", "-o", "subnet_report.csv" ]
```

Analysis Q&A

```
## Analysis Q&A

1. **Subnet(s) with the most hosts**

   Every /22 block in the sheet (e.g. `10.2.0.0/22`, `192.168.100.0/22`).
   Each one can hold **1■022** usable IPs.

2. **Overlapping subnets?**

   Nah, didn't spot any overlaps – they’re all cleanly separated.

3. **Smallest vs. largest subnet**

   * Smallest → all the `/24` networks (254 hosts).
   * Largest  → all the `/22` networks (1■022 hosts).

4. **How to waste fewer IPs**

   Right now the /22s are overkill. You could slice those into four `/24`s (or even `/25`s
for tiny teams). Use VLSM: pick
   subnet sizes that match real head■counts, slap them in an IPAM sheet, and block random
grabs with DHCP scopes & ACLs.
```



```
172.16.8.9, 255.255.254.0, /23, 172.16.8.0, 172.16.9.255, 510, 172.16.8.0/23
10.4.3.2, 255.255.255.0, /24, 10.4.3.0, 10.4.3.255, 254, 10.4.3.0/24
192.168.20.44, 255.255.252.0, /22, 192.168.20.0, 192.168.23.255, 1022, 192.168.20.0/22
172.16.40.22, 255.255.255.0, /24, 172.16.40.0, 172.16.40.255, 254, 172.16.40.0/24
10.0.0.200, 255.255.254.0, /23, 10.0.0.0, 10.0.1.255, 510, 10.0.0.0/23
192.168.10.1, 255.255.255.0, /24, 192.168.10.0, 192.168.10.255, 254, 192.168.10.0/24
172.16.15.15, 255.255.254.0, /23, 172.16.14.0, 172.16.15.255, 510, 172.16.14.0/23
10.3.3.9, 255.255.252.0, /22, 10.3.0.0, 10.3.3.255, 1022, 10.3.0.0/22
192.168.4.5, 255.255.255.0, /24, 192.168.4.0, 192.168.4.255, 254, 192.168.4.0/24
10.50.2.7, 255.255.255.0, /24, 10.50.2.0, 10.50.2.255, 254, 10.50.2.0/24
172.16.60.30, 255.255.252.0, /22, 172.16.60.0, 172.16.63.255, 1022, 172.16.60.0/22
192.168.7.8, 255.255.254.0, /23, 192.168.6.0, 192.168.7.255, 510, 192.168.6.0/23
```

Network Plot