

LAB 1: การตั้งค่า Nginx Reverse Proxy + Load Balancer บน Ubuntu 24.04

(ใช้โดเมน comss.kru.ac.th)

ระยะเวลา: 3 ชั่วโมง

ระดับ: Intermediate

ผลลัพธ์: ผู้เรียนสามารถทำ Reverse Proxy + Load Balancer ที่ใช้งานได้จริง



ผลลัพธ์ที่ผู้เรียนต้องทำให้ได้

เมื่อทำตามคู่มือนี้จนจบ ผู้เรียนจะสามารถ:

- เปิดเว็บผ่าน `http://comss.kru.ac.th` แล้ว
→ เห็นผลลัพธ์จาก Backend Server 1 และ Backend Server 2 แบบสลับกัน
- เปลี่ยนโหมด Load Balancer ได้
- ทดสอบระบบล้มเหลว (Failover) ได้
- เปิดดู Log แล้วอธิบายได้ว่าเกิดอะไรขึ้น



1) โครงสร้างระบบที่ต้องทำให้ได้ตาม LAB

ใช้เครื่อง 3 เครื่อง:

ชื่อเครื่อง	IP Address	หน้าที่
Server A	192.168.10.10	Reverse Proxy + Load Balancer
Server B	192.168.10.20	Backend Web Server 1
Server C	192.168.10.30	Backend Web Server 2

Domain ที่ใช้ใน LAB

`comss.kru.ac.th` → `192.168.10.10`

ถ้าในห้อง LAB ไม่มี DNS จริง ให้ตั้ง hosts (Windows):

`C:\Windows\System32\drivers\etc\hosts`

เติมบรรทัด:

`192.168.10.10 comss.kru.ac.th`



2) ขั้นตอนที่ผู้เรียนต้องทำตาม (เริ่มปฏิบัติ)

ต่อไปนี้เป็นขั้นตอนแบบ Step-by-step ให้ผู้เรียนทำตามทีละข้อ

ทำเสร็จ → ทดสอบ → ผ่าน → ไปขั้นตอนถัดไป



ขั้นที่ 1 – อัปเดตระบบ (ทุกเครื่อง A, B, C)

รันคำสั่ง:

```
sudo apt update && sudo apt upgrade -y
```

จากนั้นติดตั้ง Nginx:

```
sudo apt install nginx -y
```

ตรวจสอบว่า Nginx ทำงาน:

```
systemctl status nginx
```

ผลลัพธ์ที่ต้องได้:

Nginx = Active (running)

□ ขั้นที่ 2 – ตั้งค่า Backend Server (B และ C)

ทำบน Server B:

```
echo "<h1>BACKEND SERVER 1</h1>" | sudo tee /var/www/html/index.html
```

ทำบน Server C:

```
echo "<h1>BACKEND SERVER 2</h1>" | sudo tee /var/www/html/index.html
```

ทดสอบจาก Server A:

```
curl 192.168.10.20
```

```
curl 192.168.10.30
```

ผลลัพธ์ที่ต้องได้:

- 192.168.10.20 → BACKEND SERVER 1
- 192.168.10.30 → BACKEND SERVER 2

□ ขั้นที่ 3 – สร้าง Reverse Proxy + Load Balancer (Server A)

สร้างไฟล์:

```
sudo nano /etc/nginx/sites-available/comss
```

วางโค้ดนี้:

```
# กลุ่ม backend servers ของ LAB
```

```
upstream backend_servers {
    server 192.168.10.20 max_fails=3 fail_timeout=30s;
    server 192.168.10.30 max_fails=3 fail_timeout=30s;
}
```

```
server {
```

```
    listen 80;
```

```
    server_name comss.kru.ac.th;
```

```
    access_log /var/log/nginx/comss_access.log;
```

```
    error_log /var/log/nginx/comss_error.log;
```

```
    location / {
```

```
        proxy_pass http://backend_servers;
```

```
        proxy_set_header Host          $host;
```

```
        proxy_set_header X-Real-IP     $remote_addr;
```

```
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
        proxy_set_header X-Forwarded-Proto $scheme;
```

```
        proxy_connect_timeout 5s;
```

```
        proxy_read_timeout 60s;
```

```
        proxy_send_timeout 60s;
```

```
        proxy_http_version 1.1;
```

```
        proxy_set_header Connection "";
```

```
    }
```

```
}
```

เปิดใช้งาน config:

```
sudo ln -s /etc/nginx/sites-available/comss /etc/nginx/sites-enabled/
sudo rm /etc/nginx/sites-enabled/default
```

ตรวจสอบ:

```
sudo nginx -t
sudo systemctl reload nginx
```

ผลลัพธ์ที่ต้องได้:

Config OK, Nginx reload สำเร็จ

□ ขั้นที่ 4 – ทดสอบ Round Robin Load Balancing

ผู้เรียนลองเปิดเว็บหลายครั้ง:

```
curl http://comss.kru.ac.th
curl http://comss.kru.ac.th
curl http://comss.kru.ac.th
```

ผลลัพธ์ที่ต้องเห็น:

- BACKEND SERVER 1
- BACKEND SERVER 2
- BACKEND SERVER 1
- BACKEND SERVER 2

สลับกันแบบคงที่ = Round Robin

□ ขั้นที่ 5 – เปลี่ยน Load Balancing เป็น Least Connections

แก้ไขไฟล์:

```
sudo nano /etc/nginx/sites-available/comss
```

แก้ไข upstream เป็น:

```
upstream backend_servers {
    least_conn;
    server 192.168.10.20;
    server 192.168.10.30;
}
```

Reload:

```
sudo systemctl reload nginx
```

ทดสอบ:

เปิดแท็บ browser หลายแท็บให้โหลดหนัก ๆ

ผลลัพธ์ที่ต้องเห็น:

ฝั่งไหนเบากว่า → Nginx ส่งไปเครื่องนั้นก่อน

□ ขั้นที่ 6 – เปลี่ยน Load Balancing เป็น IP Hash

แก้ไข upstream:

```
upstream backend_servers {
    ip_hash;
    server 192.168.10.20;
    server 192.168.10.30;
}
```

Reload:

```
sudo systemctl reload nginx
```

ทดสอบ:

จากเครื่องผู้เรียน → เปิดเว็บหลายครั้ง

ผลลัพธ์ที่ต้องเห็น:

ได้ Backend เดิมซ้ำ เพราะ IP เดิม

□ ขั้นที่ 7 – ทดสอบ Failover (จำลองเครื่อง Backend เสีย)

ปิด Backend Server 1:

```
sudo systemctl stop nginx # ทำบน Server B
```

ทดสอบจาก Reverse Proxy:

```
curl http://comss.kru.ac.th
```

ผลลัพธ์ที่ต้องเกิดขึ้น:

- ช่วงแรกอาจเกิด error (เช่น 502)
- หลังจากเจอ error 3 ครั้ง (ตาม max_fails=3)
- Nginx จะตัด Server B ออกจากระบบ
- ทุกคำขอจะไป Server C อัตโนมัติ

จากนั้นเปิด Backend กลับ:

```
sudo systemctl start nginx
```

□ ขั้นที่ 8 – ตรวจสอบ Log การทำงาน

Access Log:

```
sudo tail -f /var/log/nginx/comss_access.log
```

Error Log:

```
sudo tail -f /var/log/nginx/comss_error.log
```

ผู้เรียนต้องมองหา:

- connection refused
- upstream timed out
- marking server as down
- no live upstreams

ผลลัพธ์ที่ต้องได้:

ผู้เรียนอธิบายได้ว่า Log เหล่านี้หมายถึงอะไร

□ 9) สิ่งที่คุณเรียนต้องส่งใน LAB (20 คะแนน)

รายการ	คะแนน
1. Backend 2 เครื่องทำงานถูกต้อง	3
2. Reverse Proxy ทำงานถูกต้อง	3
3. Round Robin แสดงผลสลับได้จริง	3
4. LeastConn ทำงาน + อธิบายผล	3
5. IP Hash ทำงาน + อธิบายผล	4
6. ทดสอบ Failover แล้วแนบ Error Log	4
รวม	20 คะแนน

□ LAB 2: Deploy Web Application ด้วย Docker Compose (Nginx + PHP-FPM + MariaDB + phpMyAdmin) บน Ubuntu

24.04

เข้าถึงเว็บหลักผ่าน:  <http://comss.kru.ac.th>

เข้าถึง phpMyAdmin ผ่าน:  <http://comss.kru.ac.th:8080>

วัตถุประสงค์ของ LAB

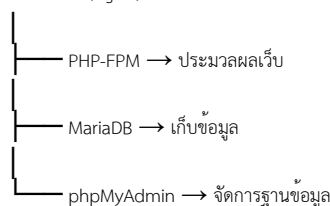
เมื่อทำ LAB นี้จบ ผู้เรียนจะสามารถ:

- ติดตั้ง Docker + Docker Compose บน Ubuntu 24.04
- Deploy Web Application แบบ Full Stack ด้วย Docker Compose
- ใช้งานบริการ 4 ส่วน ได้แก่
 - ✓ Nginx (Web Server)
 - ✓ PHP-FPM 8.x (ประมวลผล PHP)
 - ✓ MariaDB (ฐานข้อมูล)
 - ✓ phpMyAdmin (จัดการฐานข้อมูลผ่านเว็บ)
- เข้าใจการทำงานของ
 - ✓ bind mount (โค้ดแก้แล้วเห็นทันที)
 - ✓ volume (ใช้เก็บข้อมูลถาวร)
- ทดสอบการ restart แบบ Zero Downtime
- อ่านสถานะ container และแก้ปัญหาเบื้องต้นได้

□ 1) โครงสร้างระบบของ LAB

บริการทั้งหมดทำงานในเครื่องเดียว (Server A)

comss.kru.ac.th (Nginx)



พอร์ตที่ใช้

บริการ	พอร์ต	URL
Nginx	80	http://comss.kru.ac.th
phpMyAdmin	8080	http://comss.kru.ac.th:8080
MariaDB	3306	Internal (ไม่เปิดให้ภายนอก)

□ 2) เตรียมเครื่องให้พร้อม

รันคำสั่งต่อไปนี้บน Ubuntu 24.04:

```
sudo apt update && sudo apt upgrade -y
```

ติดตั้งแพ็คเกจพื้นฐาน:

```
sudo apt install ca-certificates curl gnupg -y
```

□ 3) ติดตั้ง Docker และ Docker Compose

3.1 เพิ่ม Repository ของ Docker

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

เพิ่มแหล่ง:

```
echo \
```

```
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
```

```
https://download.docker.com/linux/ubuntu noble stable" | \
```

```
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

อัปเดต:

```
sudo apt update
```

3.2 ติดตั้ง Docker Engine + Compose

```
sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
```

3.3 ตรวจสอบการติดตั้ง

```
docker --version
```

```
docker compose version
```

ผลลัพธ์ที่ต้องได้:

- Docker Engine เวอร์ชัน 26+
- Docker Compose v2 ขึ้นไป

□ 4) สร้างโฟลเดอร์โปรเจกต์สำหรับ LAB 2

```
mkdir -p ~/lab2-docker
```

```
cd ~/lab2-docker
```

สร้างโฟลเดอร์เว็บ:

```
mkdir src
```

```
echo "<?php phpinfo(); ?>" > src/index.php
```

ไฟล์ index.php จะแสดงผลรายละเอียด PHP

□ 5) สร้างไฟล์ docker-compose.yml

สร้างไฟล์:

```
nano docker-compose.yml
```

ใส่นี้อาตังนี้:

```
version: '3.9'
```

services:

```
nginx:
```

```
image: nginx:latest
```

```

container_name: lab2-nginx
ports:
  - "80:80"
volumes:
  - ./src:/var/www/html
  - ./nginx.conf:/etc/nginx/conf.d/default.conf
depends_on:
  - php
restart: always

```

```

php:
  image: php:8.2-fpm
  container_name: lab2-php
  volumes:
    - ./src:/var/www/html
  restart: always

```

```

db:
  image: mariadb:10.7
  container_name: lab2-mariadb
  environment:
    MYSQL_ROOT_PASSWORD: rootpass123
    MYSQL_DATABASE: labdb
    MYSQL_USER: labuser
    MYSQL_PASSWORD: labpass123
  volumes:
    - db_data:/var/lib/mysql
  restart: always

```

```

phpmyadmin:
  image: phpmyadmin:latest
  container_name: lab2-phpmyadmin
  ports:
    - "8080:80"
  environment:
    PMA_HOST: db
    PMA_USER: labuser
    PMA_PASSWORD: labpass123
  restart: always

```

```

volumes:
  db_data:

```

อธิบายผู้เรียน:

- bind mount (./src:/var/www/html) → แก้ไฟล์แล้วเว็บเปลี่ยนทันที
- volume (db_data:) → ใช้เก็บฐานข้อมูลถาวร

6) สร้างไฟล์ nginx.conf

สร้างไฟล์:

```
nano nginx.conf
```

ใส่:

```
server {
```

```

listen 80;
server_name comss.kru.ac.th;

root /var/www/html;
index index.php index.html;

location / {
    try_files $uri $uri/ /index.php?$query_string;
}

location ~ \.php$ {
    fastcgi_pass php:9000;
    fastcgi_index index.php;
    include fastcgi_params;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
}
}

```

7) เริ่มระบบทั้งหมด

docker compose up -d

ตรวจ container:

docker compose ps

ต้องเห็น:

nginx / php / db / phpmyadmin = running

8) ทดสอบเว็บผ่านเบราว์เซอร์

เว็บหลัก (เว็บ PHP)

เปิด:

 <http://comss.kru.ac.th>

ผลลัพธ์ที่ต้องเห็น:

หน้า phpinfo() แสดงรายละเอียด PHP

phpMyAdmin (จัดการฐานข้อมูล)

เปิด:

 <http://comss.kru.ac.th:8080>

เข้าสู่ระบบด้วย:

- User: labuser
- Pass: labpass123

ต้องเห็นฐานข้อมูล labdb และสามารถใช้งานได้

9) ทดสอบ Hot Reload (แก้ไฟล์แล้วเว็บเปลี่ยนทันที)

แก้ไฟล์:

```
nano src/index.php
```

เปลี่ยนเป็น:

```
<?php
```

```
echo "ยินดีต้อนรับสู่เว็บไซต์ comss.kru.ac.th (LAB 2)";
```

```
?>
```

รีเฟรชเว็บ:

 <http://comss.kru.ac.th>

ผลลัพธ์: ข้อความใหม่ต้องแสดงทันที

10) ทดสอบ Restart แบบ Zero Downtime

รัน:

```
docker compose restart nginx
```

ผลลัพธ์:

เว็บยังทำงานปกติ แทบไม่มี downtime

11) ตรวจสอบ Volume ของฐานข้อมูล

ดู Volume:

```
docker volume ls
```

ดูข้อมูล Volume:

```
docker volume inspect lab2-docker_db_data
```

12) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

งานที่ส่ง	คะแนน
ติดตั้ง Docker + Compose สำเร็จ	3
เว็บหลักแสดง phpinfo() ผ่าน comss.kru.ac.th	3
phpMyAdmin เปิดได้ที่ comss.kru.ac.th:8080	3
Hot Reload ทำงาน	3
อธิบายความต่าง Bind Mount vs Volume	3
Restart แบบ Zero Downtime สำเร็จ	5
รวม	20 คะแนน

□ LAB 3: ย้ายโฮสต์ Web Server ด้วย rsync + Incremental Backup + Restore Test บน Ubuntu 24.04

(Incremental Backup, Migration & Recovery Validation)

🎯 วัตถุประสงค์ของ LAB

หลังจบ LAB นี้ ผู้เรียนต้องสามารถ:

- ✓ เข้าใจหลักการ rsync แบบ incremental backup
- ✓ สำรองข้อมูลเว็บและฐานข้อมูลแบบปลอดภัย
- ✓ ย้ายเว็บไซต์จากเซิร์ฟเวอร์เดิมไปยังเซิร์ฟเวอร์ใหม่
- ✓ ใช้ SSH Key สำหรับการเชื่อมต่ออย่างปลอดภัย
- ✓ ทำ automation backup script
- ✓ ทดสอบกู้คืน (Restore Validation) ได้จริง

□ 1) โครงสร้างระบบของ LAB

ใช้เครื่อง 2 เครื่อง:

โฮสต์	IP Address	หน้าที่
Server A	192.168.10.10	Web Server ใหม่ (ปลายทาง)
Server B	192.168.10.20	Web Server เก่า (ต้นทาง)

โดเมนในการทดสอบ

comss.kru.ac.th

ผู้เรียนต้องย้ายเว็บจาก Server B → Server A

โดยใช้ rsync incremental backup

□ 2) เตรียมสภาพแวดล้อม

ทำบนทั้ง 2 เครื่อง:

```
sudo apt update && sudo apt upgrade -y
sudo apt install nginx rsync mariadb-server -y
```

□ 3) สร้างเว็บตัวอย่างบน Server B (เครื่องต้นทาง)

```
sudo mkdir -p /var/www/comss
```

```
echo "<h1>WEB SERVER OLD (B): comss.kru.ac.th</h1>" | sudo tee /var/www/comss/index.html
```

```
sudo chown -R www-data:www-data /var/www/comss
```

สร้าง Nginx config:

```
sudo nano /etc/nginx/sites-available/comss
```

ใส่เนื้อหา:

```
server {
    listen 80;
    server_name comss.kru.ac.th;
    root /var/www/comss;
    index index.html;
```

}

เปิดใช้งาน:

```
sudo ln -s /etc/nginx/sites-available/comss /etc/nginx/sites-enabled/
```

```
sudo systemctl restart nginx
```

4) ทดสอบเว็บบน Server B

จาก PC หรือจาก Server A:

```
curl http://192.168.10.20
```

ผลลัพธ์ที่ต้องเห็น:

WEB SERVER OLD (B): comss.kru.ac.th

นี่คือข้อมูลต้นทางที่เราจะ Backup ไป Server A

5) สร้าง SSH Key เพื่อ Backup แบบไม่ต้องใส่รหัสผ่าน

ทำบน Server A (ปลายทาง):

```
ssh-keygen -t rsa -b 4096
```

เมื่อถามให้กด Enter ทุกครั้ง (ไม่ต้องตั้ง passphrase)

ส่ง Public Key ไป Server B:

```
ssh-copy-id user@192.168.10.20
```

(เปลี่ยน user เป็นชื่อผู้ใช้ของนักศึกษาบน Server B)

ทดสอบว่าไม่ต้องใส่รหัสผ่าน:

```
ssh user@192.168.10.20
```

6) ทำ Full Backup ครั้งแรก (จาก B → A)

บน Server A:

```
sudo rsync -avz --delete user@192.168.10.20:/var/www/comss/ /var/www/comss/
```

คำอธิบาย:

คำสั่ง	ความหมาย
-a	เก็บสิทธิ์ไฟล์
-v	แสดงรายละเอียด
-z	บีบอัดข้อมูล
--delete	ลบไฟล์ที่ไม่มีในต้นทางออก

ผลลัพธ์ที่ต้องได้:

โฟลเดอร์ /var/www/comss บน Server A ต้องมีข้อมูลจาก Server B ครบถ้วน

7) ตั้งค่า Web Server บน Server A (ปลายทาง)

```
sudo mkdir -p /var/www/comss
```

```
sudo nano /etc/nginx/sites-available/comss
```

ใส่:

```
server {
    listen 80;
    server_name comss.kru.ac.th;
    root /var/www/comss;
    index index.html;
}
```

เปิดใช้งาน:

```
sudo ln -s /etc/nginx/sites-available/comss /etc/nginx/sites-enabled/
sudo systemctl restart nginx
```

8) ทดสอบเว็บที่ถูกย้ายไป Server A

```
curl http://192.168.10.10
```

ผลลัพธ์:

WEB SERVER OLD (B): comss.kru.ac.th

สำเร็จ → เว็บถูกย้ายเรียบร้อยแล้ว

9) จำลองสถานการณ์เว็บมีการอัปเดต (Incremental Backup)

บน Server B (ต้นทาง):

```
echo "<p>Updated Content from Server B</p>" | sudo tee -a /var/www/comss/index.html
```

รัน incremental backup อีกครั้งบน Server A:

```
sudo rsync -avz --delete user@192.168.10.20:/var/www/comss/ /var/www/comss/
```

ทดสอบ:

```
curl http://192.168.10.10
```

ผลลัพธ์ที่ต้องเห็น:

มี <p>Updated Content from Server B</p> เพิ่มเข้ามา

10) สร้าง Automation Backup Script

ทำบน Server A:

```
sudo nano /usr/local/bin/comss-backup.sh
```

ใส่:

```
#!/bin/bash
```

```
rsync -avz --delete user@192.168.10.20:/var/www/comss/ /var/www/comss/
```

ตั้งสิทธิ์:

```
sudo chmod +x /usr/local/bin/comss-backup.sh
```

11) ตั้ง Cron Job ให้ Backup อัตโนมัติทุก 5 นาที

เปิด cron:

```
crontab -e
```

เพิ่ม:

```
* /5 * * * * /usr/local/bin/comss-backup.sh
```

12) ทดสอบ Restore (จำลองเว็บปลายทางถูกลบ)

บน Server A ลบไฟล์:

```
sudo rm /var/www/comss/index.html
```

รัน Backup Script:

```
sudo /usr/local/bin/comss-backup.sh
```

ทดสอบ:

```
curl http://192.168.10.10
```

ผลลัพธ์ต้องกลับมาเหมือนเดิม

→ Restore สำเร็จ

☐ 13) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการที่ต้องส่ง	คะแนน
1. rsync Full Backup ครั้งแรกสำเร็จ	3
2. เว็บบ้ายจาก B → A ได้จริง	3
3. Incremental Backup ทำงาน	3
4. Automation Script สำเร็จ	3
5. Cron ทำงาน (แนบ log/ผลลัพธ์)	4
6. Restore Test สำเร็จ	4
รวม	20 คะแนน

□ LAB 4: ติดตั้งและตั้งค่า DNS Server ด้วย BIND9 (Master-Slave Replication) บน Ubuntu Server 24.04

Domain ที่ใช้ใน LAB: comss.kru.ac.th

รูปแบบ: DNS Authoritative + Master/Slave + Zone Transfer (AXFR)

วัตถุประสงค์ของ LAB

หลังจบ LAB ผู้เรียนต้องสามารถ:

- ✓ ติดตั้ง DNS Server (BIND9)
- ✓ สร้าง Forward Zone และ Reverse Zone
- ✓ ตั้งค่า NS, A, MX, CNAME
- ✓ ตั้งค่า Master Zone และ Slave Zone
- ✓ ทำ Zone Transfer (AXFR) ระหว่าง Master ↔ Slave
- ✓ ทดสอบการทำงานด้วย dig และ nslookup
- ✓ ดู Log และแก้ไขปัญหาพื้นฐาน DNS ได้

□ 1) โครงสร้างระบบ DNS ของ LAB

ใช้เครื่อง 2 เครื่อง:

ชื่อเครื่อง	IP	บทบาท
Server A	192.168.10.10	DNS Master
Server B	192.168.10.20	DNS Slave

Domain:

comss.kru.ac.th

DNS Server Name:

- ns1.comss.kru.ac.th → Master (192.168.10.10)
- ns2.comss.kru.ac.th → Slave (192.168.10.20)

□ 2) ติดตั้งแพ็คเกจ BIND9

ติดตั้งบนทั้ง Master และ Slave:

```
sudo apt update
```

```
sudo apt install bind9 bind9utils bind9-doc -y
```

ตรวจสอบ:

```
systemctl status bind9
```

ต้องเป็น active (running)

□ 3) ตั้งค่า DNS Master (Server A)

3.1 สร้างไฟล์ zone บน Master

เปิดไฟล์:

```
sudo nano /etc/bind/named.conf.local
```

เพิ่มบรรทัด:

```
zone "comss.kru.ac.th" {
    type master;
    file "/etc/bind/zones/db.comss.kru.ac.th";
    allow-transfer { 192.168.10.20; }; // Slave
};
```

```
zone "10.168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/zones/db.192.168.10";
    allow-transfer { 192.168.10.20; };
};
```

3.2 สร้างไฟล์เตอร์ zone

```
sudo mkdir -p /etc/bind/zones
```

4) สร้าง Forward Zone (db.comss.kru.ac.th)

สร้างไฟล์:

```
sudo nano /etc/bind/zones/db.comss.kru.ac.th
```

วางเนื้อหา:

```
$TTL 604800
@ IN SOA ns1.comss.kru.ac.th. admin.comss.kru.ac.th. (
    2025010101 ; Serial
    3600 ; Refresh
    1800 ; Retry
    604800 ; Expire
    86400 ) ; Negative Cache TTL

; DNS Servers
IN NS ns1.comss.kru.ac.th.
IN NS ns2.comss.kru.ac.th.

; A Records
ns1 IN A 192.168.10.10
ns2 IN A 192.168.10.20
www IN A 192.168.10.10
lab IN A 192.168.10.50

; CNAME
app IN CNAME www.comss.kru.ac.th.
```

5) สร้าง Reverse Zone (db.192.168.10)

สร้างไฟล์:

```
sudo nano /etc/bind/zones/db.192.168.10
```

วางเนื้อหา:

```
$TTL 604800
@ IN SOA ns1.comss.kru.ac.th. admin.comss.kru.ac.th. (
    2025010101
    3600
    1800
    604800
```

86400)

```
@ IN NS ns1.comss.kru.ac.th.
@ IN NS ns2.comss.kru.ac.th.
```

```
10 IN PTR ns1.comss.kru.ac.th.
20 IN PTR ns2.comss.kru.ac.th.
50 IN PTR lab.comss.kru.ac.th.
```

6) ตรวจสอบและ Restart DNS Master

ตรวจสอบไฟล์ zone:

```
sudo named-checkzone comss.kru.ac.th /etc/bind/zones/db.comss.kru.ac.th
sudo named-checkzone 10.168.192.in-addr.arpa /etc/bind/zones/db.192.168.10
```

Restart:

```
sudo systemctl restart bind9
```

7) ตั้งค่า DNS Slave (Server B)

เปิดไฟล์:

```
sudo nano /etc/bind/named.conf.local
```

เพิ่ม:

```
zone "comss.kru.ac.th" {
    type slave;
    masters { 192.168.10.10; };
    file "/var/cache/bind/db.comss.kru.ac.th";
};
```

```
zone "10.168.192.in-addr.arpa" {
    type slave;
    masters { 192.168.10.10; };
    file "/var/cache/bind/db.192.168.10";
};
```

Restart:

```
sudo systemctl restart bind9
```

ตรวจสอบว่า Slave ได้ไฟล์ zone หรือไม่:

```
ls -l /var/cache/bind/
```

ผลที่ต้องเห็น:

มีไฟล์ db.comss.kru.ac.th และ db.192.168.10 ถูกโหลดมาโดยอัตโนมัติ

8) ทดสอบ DNS Query บน Master และ Slave

8.1 ทดสอบ A Record:

```
dig @192.168.10.10 www.comss.kru.ac.th
```

```
dig @192.168.10.20 www.comss.kru.ac.th
```

ผลลัพธ์ต้องเหมือนกัน

8.2 ทดสอบ CNAME:

dig app.comss.kru.ac.th

8.3 ทดสอบ Reverse Lookup:

dig -x 192.168.10.10

9) ทดสอบ Zone Transfer

บน Slave ตรวจสอบ:

dig @192.168.10.20 comss.kru.ac.th AXFR

ผลลัพธ์ต้องแสดง zone ทั้งหมด

10) จำลองกรณี Master ล่ม

หยุดบริการ DNS บน Master:

sudo systemctl stop bind9

ให้ผู้เรียนลองสั่ง:

dig @192.168.10.20 www.comss.kru.ac.th

ผลลัพธ์: Slave ต้องยังตอบ DNS ได้ (Read-only)

แปลว่า HA ของ DNS ทำงาน

11) ตรวจสอบ LOG

บน Master:

sudo tail -f /var/log/syslog | grep named

บน Slave:

sudo tail -f /var/log/syslog | grep named

ผู้เรียนต้องค้นหา:

- “Zone transfer complete”
- “AXFR request received”
- “refresh zone”
- “zone updated”

12) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
1. DNS Master ทำงาน	3
2. DNS Slave sync zone ได้จริง	3
3. A, CNAME, PTR Record ทำงาน	4
4. Zone Transfer (AXFR) สำเร็จ	4
5. จำลอง Master ล่ม แล้ว Slave ตอบได้	3
6. แนบ Log การทำงานพร้อมคำอธิบาย	3
รวม	20 คะแนน

□ LAB 5: ระบบรักษาความปลอดภัย Linux – SSH Hardening + UFW Firewall + Fail2ban + Log Monitoring บน Ubuntu

24.04

(Advanced Linux Security Hardening & Threat Mitigation)

🎯 วัตถุประสงค์ของ LAB

หลังจบ LAB ผู้เรียนต้องทำได้:

- ✓ ปรับแต่ง SSH ให้ปลอดภัย
- ✓ เปลี่ยน Port, Disable Root Login, Authentication Hardening
- ✓ ตั้งค่า Firewall UFW ให้อนุญาตเฉพาะบริการจำเป็น
- ✓ ติดตั้งและตั้งค่า Fail2ban ป้องกัน brute-force attack
- ✓ ตรวจสอบ Log และวิเคราะห์การโจมตี
- ✓ ทดสอบโจมตีจำลองเพื่อดู Fail2ban ทำงานจริง

□ 1) โครงสร้างระบบของ LAB

ใช้เครื่องเดียว (Server A)

Service	Port	หน้าที่
SSH	2202	Remote Login (เปลี่ยนจาก 22 เพื่อความปลอดภัย)
UFW	—	Firewall
Fail2ban	—	ป้องกัน brute-force
Syslog/Journalctl	—	Log

เป้าหมาย:

ทำให้เครื่อง Ubuntu 24.04 มีระบบรักษาความปลอดภัยพร้อมใช้งานจริงในองค์กร

□ 2) เตรียมระบบ

อัปเดตแพ็คเกจ:

```
sudo apt update && sudo apt upgrade -y
```

ตรวจสอบสถานะ SSH:

```
systemctl status ssh
```

ทั้งหมดนี้ทำบน Ubuntu 24.04

□ 3) ตั้งค่า SSH Hardening

3.1 เปลี่ยน SSH Port (22 → 2202)

แก้ไขไฟล์:

```
sudo nano /etc/ssh/sshd_config
```

หาและแก้ไข:

```
#Port 22
```

```
Port 2202
```

3.2 ปิดการ Login ด้วย root

ค้นหา:

PermitRootLogin yes

แก้เป็น:

PermitRootLogin no

3.3 เปิดใช้ Only Strong Authentication

เพิ่มบรรทัด:

PasswordAuthentication no

PubkeyAuthentication yes

หมายเหตุ: ก่อนปิด PasswordAuthentication

ต้องสร้าง SSH Key ล่วงหน้าเพื่อป้องกันล็อกอินไม่ได้

Restart SSH:

sudo systemctl restart ssh

4) สร้าง SSH Key (เพื่อเข้าสู่ Server แบบปลอดภัย)

บนเครื่อง Client (หรือบน Server ถ้าจำลอง):

ssh-keygen -t rsa -b 4096

ส่ง key ไปยัง Server:

ssh-copy-id -p 2202 user@server-ip

ทดสอบ:

ssh -p 2202 user@server-ip

ต้องเข้าสู่ระบบได้โดยไม่ใช้รหัสผ่าน

5) ตั้งค่า Firewall ด้วย UFW

เปิดใช้งาน UFW:

sudo ufw enable

อนุญาต SSH Port ใหม่:

sudo ufw allow 2202/tcp

อนุญาต HTTP (ถ้าต้องการ):

sudo ufw allow 80/tcp

ดูสถานะ:

sudo ufw status numbered

ผลลัพธ์ที่ต้องได้:

UFW ต้องอนุญาตเฉพาะ 2202 และ 80 เท่านั้น

6) ติดตั้ง Fail2ban (ป้องกัน brute-force)

sudo apt install fail2ban -y

คัดลอกไฟล์ config:

sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local

sudo nano /etc/fail2ban/jail.local

ตั้งค่า jail สำหรับ SSH:

[sshd]

enabled = true

port = 2202

filter = sshd

```
logpath = /var/log/auth.log
maxretry = 3
findtime = 10m
bantime = 30m
Restart Fail2ban:
sudo systemctl restart fail2ban
sudo systemctl status fail2ban
```

7) ทดสอบ Fail2ban ด้วยการโจมตีจำลอง

บนเครื่องลูกข่าย ให้พยายามเชื่อมต่อแบบริชต์ผิด:

```
ssh -p 2202 wronguser@server-ip
ลองผิด 3 ครั้ง (ตาม maxretry)
แล้วดูสถานะบนเซิร์ฟเวอร์:
sudo fail2ban-client status sshd
```

ผลลัพธ์ที่ต้องเห็น:

IP ถูก Ban อยู่ในรายการ

8) ตรวจสอบ Log เพื่อวิเคราะห์การโจมตี

ตรวจ log SSH:

```
sudo tail -f /var/log/auth.log
```

ผู้เรียนต้องเห็น:

- การ login สำเร็จ
- การ login ผิด
- บันทึกของ Fail2ban เช่น
- fail2ban.actions: WARNING [sshd] Ban <IP>

ตรวจ log Fail2ban:

```
sudo tail -f /var/log/fail2ban.log
```

9) การ Unban IP (กรณีถูก Ban ผิด)

```
sudo fail2ban-client set sshd unbanip <IP>
```

10) ตรวจสอบความปลอดภัยของระบบหลัง Hardening

ผู้เรียนต้องตรวจเช็ค:

10.1 SSH Port ถูกเปลี่ยน

```
sudo ss -tulpn | grep ssh
```

ต้องขึ้นพอร์ต 2202

10.2 Firewall เปิดใช้งานและบล็อกอื่น ๆ

```
sudo ufw status
```

ต้องอนุญาตเฉพาะ 2202, 80

10.3 Fail2ban ทำงาน

```
sudo fail2ban-client status
```

11) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
เปลี่ยน SSH port สำเร็จและ Login ผ่าน key ได้	4
ปิด root login / ปิดรหัสผ่าน	3
UFW เปิดเฉพาะค่าที่จำเป็น	3
Fail2ban ทำงานและ Ban IP ได้จริง	4
แนบ logs (auth.log / fail2ban.log)	3
อธิบาย Hardening ทั้งระบบ	3
รวม	20 คะแนน

❑ LAB 6: การติดตั้งและใช้งาน Cockpit เพื่อจัดการเซิร์ฟเวอร์ผ่านเว็บ บน Ubuntu Server 24.04

(Web-Based Server Management for Linux)

วัตถุประสงค์ของ LAB

หลังทำ LAB นี้ ผู้เรียนต้องสามารถ:

- ✓ ติดตั้ง Cockpit บน Ubuntu 24.04
- ✓ ใช้งานผ่าน Browser ได้
- ✓ ตรวจสอบ CPU / RAM / Disk แบบ Real-Time
- ✓ ดู Log, จัดการ Service, Firewall, Storage, User
- ✓ ติดตั้ง Cockpit Modules เพิ่มเติม
- ✓ ตั้งค่า Security สำหรับการใช้งานจริง

❑ 1) โครงสร้างระบบ LAB

ใช้เครื่องเดียว (Server A)

บริการ พอร์ต URL

Cockpit 9090 <https://server-ip:9090>

❑ 2) อัปเดตระบบ

`sudo apt update && sudo apt upgrade -y`

❑ 3) ติดตั้ง Cockpit

Ubuntu 24.04 มี Cockpit ใน Repo:

`sudo apt install cockpit -y`

เปิดใช้งาน service:

`sudo systemctl enable --now cockpit`

ตรวจสอบ:

`systemctl status cockpit`

❑ 4) เปิด Firewall เพื่อเข้า Cockpit

ถ้าใช้ UFW:

`sudo ufw allow 9090/tcp`

❑ 5) ใช้งาน Cockpit ผ่าน Browser

ให้ผู้เรียนเปิด Browser แล้วไปที่:

`https://<IP_SERVER>:9090`

ตัวอย่าง:

https://192.168.10.10:9090

จะเจอหน้า Login

ใช้บัญชีผู้ใช้ของ Ubuntu ในการเข้าสู่ระบบ (เช่น user ที่กำลังใช้งาน)

ผลลัพธ์ที่ต้องเห็น:

หน้า Dashboard ของ Cockpit

6) สำรวจฟีเจอร์ใน Cockpit (ผู้เรียนต้องปฏิบัติจริง)

6.1 Dashboard

ผู้เรียนต้องตรวจดู:

- CPU Load + Usage
- RAM Usage
- Disk I/O
- Network Traffic

6.2 Logs

ไปที่เมนู Logs

ผู้เรียนต้อง:

- ✓ ค้นหา Error
- ✓ กรองตามบริการ
- ✓ ดู Security Logs

6.3 Services

ผู้เรียนต้องทดลอง:

- หยุด Nginx → Start Again
- Restart SSH
- Check Fail2ban service (จาก LAB 5)

6.4 Networking

ผู้เรียนตรวจสอบ:

- IP address
- Routing table
- Listening ports

6.5 Accounts (User Management)

ผู้เรียนทดลอง:

- เพิ่ม User ใหม่
 - ตั้ง Password
 - ตั้งสิทธิ์ sudo
 - ลบ User
-

6.6 Storage

ผู้เรียนต้องดู:

- Partition
- Filesystem
- Disk I/O Graph

7) ติดตั้ง Cockpit Modules เพิ่มเติม

7.1 Cockpit–NetworkManager

```
sudo apt install cockpit-networkmanager -y
```

7.2 Cockpit–Package Updates

```
sudo apt install cockpit-packagekit -y
```

7.3 Cockpit–Podman (จัดการ Container)

```
sudo apt install cockpit-podman -y
```

ผู้เรียนต้องเห็นเมนูใหม่ปรากฏใน Cockpit

8) ตั้งค่า Security ของ Cockpit

8.1 เปิด TLS แบบ Browser เชื่อถือ

สร้าง self-signed cert:

```
sudo mkdir -p /etc/cockpit/ws-certs.d
sudo openssl req -new -x509 -days 365 \
  -keyout /etc/cockpit/ws-certs.d/cert.key \
  -out /etc/cockpit/ws-certs.d/cert.crt \
  -nodes
```

Restart Cockpit:

```
sudo systemctl restart cockpit
```

8.2 จำกัดการเข้าใช้งานเฉพาะ IP ภายใน

เพิ่มใน UFW:

```
sudo ufw allow from 192.168.10.0/24 to any port 9090
sudo ufw deny 9090/tcp
```

9) การตรวจสอบ Log ของ Cockpit

ดู Log:

```
sudo journalctl -u cockpit
```

```
sudo tail -f /var/log/syslog | grep cockpit
```

ผู้เรียนต้องสังเกต:

- การ Login
- การแก้ไขค่า
- ความผิดพลาดของ service

□ 10) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ติดตั้ง Cockpit สำเร็จ และเข้า Dashboard ได้	4
ตรวจสอบ CPU/RAM/Disk อย่างถูกต้อง	3
ใช้งาน Logs, Services, Networking	3
ติดตั้ง Modules เพิ่มเติม (Network, PackageKit, Podman)	3
ตั้งค่า Security (Self-signed Cert + จำกัด IP)	4
แนบ Screenshot + อธิบายผล	3
รวม	20 คะแนน

LAB 7: การติดตั้งและใช้งาน Netdata สำหรับ Real-Time Monitoring บน Ubuntu Server 24.04

(Real-Time Performance Monitoring Dashboard)

วัตถุประสงค์ของ LAB

เมื่อจบ LAB นี้ ผู้เรียนต้องสามารถ:

- ✓ ติดตั้ง Netdata บน Ubuntu 24.04
- ✓ เข้า Dashboard ผ่าน Browser
- ✓ ตรวจสอบ Real-Time ของระบบ:
 - CPU
 - RAM
 - Disk I/O
 - Network
 - Filesystem
 - Services
- ✓ ตั้งค่าความปลอดภัยพื้นฐาน
- ✓ ปรับแต่ง Netdata ให้เหมาะกับการใช้งานจริง
- ✓ ตรวจสอบ Log และแก้ปัญหาเบื้องต้น

1) โครงสร้างระบบ LAB

ใช้เครื่องเดียว Server A

บริการ	พอร์ต	URL
Netdata	19999	http://server-ip:19999

ตัวอย่าง:

http://192.168.10.10:19999

2) อัปเดตระบบ

sudo apt update && sudo apt upgrade -y

3) ติดตั้ง Netdata (วิธีการ)

ติดตั้งผ่านสคริปต์ทางการของ Netdata:

```
bash <(curl -Ss https://my-netdata.io/kickstart.sh)
```

ถ้าถามระหว่างติดตั้ง ให้ตอบ YES ทุกคำถาม

ตรวจสอบว่า Netdata ทำงาน:

```
systemctl status netdata
```

ต้องเป็น **active (running)**

❑ 4) เปิดไฟร์วอลล์ (ถ้าใช้ UFW)

อนุญาตให้เข้าถึงพอร์ต Netdata:

```
sudo ufw allow 19999/tcp
```

❑ 5) เข้าถึง Dashboard ของ Netdata

เปิดเว็บเบราว์เซอร์ → ใส่:

`http://<IP_SERVER>:19999`

ตัวอย่าง:

`http://192.168.10.10:19999`

ผลลัพธ์:

ต้องเห็น Dashboard ของ Netdata พร้อมกราฟเคลื่อนไหวแบบ Real-Time

❑ 6) งานที่ผู้เรียนต้องทำใน Netdata Dashboard

ผู้เรียนต้องสำรวจและบันทึกค่าจากหมวดต่อไปนี้:

✓ 6.1 CPU Utilization

ผู้เรียนต้องดู:

- User CPU
- System CPU
- iowait
- Load Average
- Interrupts

เว็บกราฟจะอัปเดตทุก 1 วินาที

✓ 6.2 RAM & Swap

ผู้เรียนต้องรายงาน:

- RAM ที่ใช้งาน
- Available RAM
- Swap Usage

✓ 6.3 Disk I/O

ผู้เรียนต้องสังเกต:

- Read I/O
- Write I/O
- Busy time (%)

ทดลองสร้างโหลด:

```
sudo dd if=/dev/zero of=/tmp/testfile bs=1M count=500
```

กลับไปดูกราฟ → ต้องมี spike

✓ 6.4 Network Traffic

ผู้เรียนต้องดู:

- Incoming (In)
- Outgoing (Out)
- Errors
- Drops

ทดลอง ping/transfer เพื่อให้เกิดกราฟแสดงผล

✓ 6.5 Filesystem

ผู้เรียนต้องดู:

- ความจุของไฟล์ระบบ
- Inode usage
- Mount point

✓ 6.6 Services Monitoring

ผู้เรียนต้องกดเข้า:

- systemd services
- nginx (ถ้ามีติดตั้งจาก LAB ก่อนหน้า)
- mariadb (ถ้าติดตั้งจาก LAB ก่อนหน้า)

□ 7) ตั้งค่าความปลอดภัยของ Netdata

✓ 7.1 จำกัดการเข้าถึงเฉพาะเครือข่ายภายใน

แก้ไขไฟล์:

```
sudo nano /etc/netdata/netdata.conf
```

หาบรรทัด:

bind to =

แก้เป็น:

bind to = 192.168.10.10:19999

เปลี่ยนจาก 0.0.0.0 (เปิดหมด) → ให้เข้าถึงเฉพาะในเครือข่ายเท่านั้น

Restart:

```
sudo systemctl restart netdata
```

✓ 7.2 เปิด Basic Authentication (optional)

สร้างไฟล์ auth:

```
sudo apt install apache2-utils -y
```

```
sudo htpasswd -c /etc/netdata/passwd admin
```

แก้ไข config:

```
sudo nano /etc/netdata/netdata.conf
```

เพิ่ม:

[web]

basic auth = /etc/netdata/passwd

Restart:

```
sudo systemctl restart netdata
```

ตอนเข้าเว็บต้องใส่ user/pass

8) ดู Log ของ Netdata

ตรวจสอบ Log:

```
sudo journalctl -u netdata
```

ดูแบบเรียลไทม์:

```
sudo tail -f /var/log/netdata/error.log
```

ผู้เรียนต้องค้นหา:

- Port Binding
- Errors
- Plugin failed
- Metrics dropped

9) ทดสอบ Load ให้ Netdata แสดงผลจริง

ผู้เรียนต้องรันโหลด เช่น CPU, RAM, Disk:

CPU Load:

```
yes > /dev/null &
```

หยุดด้วย:

```
killall yes
```

RAM Load:

```
stress-ng --vm 1 --vm-bytes 512M --timeout 10s
```

Disk Load:

```
sudo dd if=/dev/zero of=/tmp/file bs=4M count=256
```

กลับไปดูกราฟ → ต้องเห็น spike ที่เกิดขึ้นจริง

10) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ติดตั้ง Netdata สำเร็จ และเข้า Dashboard ได้	4
รายงานค่าจากหมวด CPU, RAM, Disk, Network, Services	6
กราฟแสดงผลโหลด CPU/RAM/Disk จากการทดสอบ	4
ตั้งค่าความปลอดภัย (Bind IP / Basic Auth)	3
ตรวจสอบ Log พร้อมอธิบายข้อผิดพลาดอย่างน้อย 1 รายการ	3
รวม	20 คะแนน

LAB 8: การติดตั้ง Prometheus + Grafana เพื่อสร้างระบบ

Monitoring ระดับองค์กรบน Ubuntu Server 24.04

(Enterprise Monitoring Stack: Prometheus + Node Exporter + Grafana)

วัตถุประสงค์ของ LAB

หลังทำ LAB นี้ ผู้เรียนต้องสามารถ:

- ✓ ติดตั้ง Prometheus บน Ubuntu 24.04
- ✓ ติดตั้ง Node Exporter เพื่อเก็บ Metrics จากระบบ
- ✓ ผูก Prometheus กับ Node Exporter
- ✓ ติดตั้ง Grafana และสร้าง Dashboard
- ✓ ดูและวิเคราะห์ Metrics สำคัญ เช่น CPU, RAM, Disk, Network
- ✓ ตั้ง Alert Rule พื้นฐาน
- ✓ ตรวจสอบ Log และแก้ปัญหาเบื้องต้น

1) โครงสร้างระบบของ LAB

ใช้เครื่องเดียว (Server A)

บริการ	พอร์ต	URL
Prometheus	9090	http://server-ip:9090
Node Exporter	9100	Metrics Endpoint
Grafana	3000	http://server-ip:3000

2) อัปเดตระบบ

`sudo apt update && sudo apt upgrade -y`

3) ติดตั้ง Prometheus

3.1 ดาวน์โหลด Prometheus

```
cd /tmp
```

```
curl -LO https://github.com/prometheus/prometheus/releases/download/v2.53.0/prometheus-2.53.0.linux-amd64.tar.gz
```

```
tar xvf prometheus-*.tar.gz
```

ย้ายไฟล์ไปตำแหน่งมาตรฐาน:

```
sudo mkdir -p /etc/prometheus /var/lib/prometheus
```

```
sudo cp prometheus-*/prometheus /usr/local/bin/
```

```
sudo cp prometheus-*/promtool /usr/local/bin/
```

```
sudo cp -r prometheus-*/consoles /etc/prometheus
```

```
sudo cp -r prometheus-*/console_libraries /etc/prometheus
```

```
sudo cp prometheus-*/prometheus.yml /etc/prometheus/
```

3.2 สร้าง User และสิทธิ์

```
sudo useradd --no-create-home --shell /bin/false prometheus
sudo chown -R prometheus:prometheus /etc/prometheus /var/lib/prometheus
```

3.3 สร้าง Service ของ Prometheus

```
sudo nano /etc/systemd/system/prometheus.service
ใส่เนื้อหา:

[Unit]
Description=Prometheus Monitoring
After=network.target

[Service]
User=prometheus
ExecStart=/usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml \
--storage.tsdb.path=/var/lib/prometheus/ \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries

[Install]
WantedBy=multi-user.target

โหลด Service:
sudo systemctl daemon-reload
sudo systemctl enable --now prometheus

ตรวจสอบ:
systemctl status prometheus
ต้องเป็น active (running)
```

4) ติดตั้ง Node Exporter

Node Exporter ใช้ดึง Metrics จากระบบ เช่น CPU, RAM, Disk

4.1 ดาวน์โหลด

```
cd /tmp
curl -LO https://github.com/prometheus/node_exporter/releases/download/v1.7.0/node_exporter-1.7.0.linux-amd64.tar.gz
tar xvf node_exporter-*.tar.gz
sudo cp node_exporter-*/node_exporter /usr/local/bin/
```

4.2 สร้าง Service

```
sudo nano /etc/systemd/system/node_exporter.service
ใส่:

[Unit]
Description=Node Exporter
After=network.target

[Service]
User=root
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=multi-user.target

เริ่มต้น:
```

```
sudo systemctl daemon-reload
sudo systemctl enable --now node_exporter
```

ตรวจสอบ:

```
systemctl status node_exporter
```

ทดลองเปิดดู Metrics:

```
http://<IP_SERVER>:9100/metrics
```

5) เพิ่ม Node Exporter ใน Prometheus

แก้ไขไฟล์:

```
sudo nano /etc/prometheus/prometheus.yml
```

เพิ่มได้ scrape_configs:

```
- job_name: 'node_exporter'
  static_configs:
    - targets: ['localhost:9100']
```

Restart Prometheus:

```
sudo systemctl restart prometheus
```

6) ติดตั้ง Grafana

ดาวน์โหลด:

```
sudo apt install -y apt-transport-https software-properties-common wget
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee /etc/apt/sources.list.d/grafana.list
sudo apt update
```

```
sudo apt install grafana -y
```

เริ่มต้น Grafana:

```
sudo systemctl enable --now grafana-server
```

เข้าสู่ใน Browser:

```
http://<IP_SERVER>:3000
```

Login:

- User: admin
- Pass: admin

(ระบบจะให้เปลี่ยนรหัสผ่านทันที)

7) ตั้ง Data Source ใน Grafana

ในหน้า Grafana:

1. ไปที่ Connections → Add new data source
2. เลือก Prometheus
3. ตั้งค่า URL:

```
http://localhost:9090
```

4. กด Save & Test

ผลลัพธ์: ควรขึ้น "Data source is working"

8) ติดตั้ง Dashboard สำเร็จรูป

ไปที่:

Create → Import Dashboard

ใส่เลข Dashboard ID ที่ใช้อยู่:

ชื่อ Dashboard	ID
Node Exporter Full	1860
Node Exporter Detail	3662
Prometheus 2.0 Stats	3664

คลิก Import → Dashboard จะพร้อมใช้งานทันที

ผู้เรียนต้องเห็นกราฟ:

- CPU Usage
- Memory
- Disk Read/Write
- Filesystem
- Network Traffic
- Process
- Load Average

ทั้งหมด Real-Time ค่ะ

9) ตั้งค่า Alert Rule พื้นฐานใน Prometheus

ตัวอย่าง Alert CPU สูงเกิน 80%

แก้ไขไฟล์:

```
sudo nano /etc/prometheus/alert.rules
```

เพิ่ม:

```
groups:
```

```
- name: cpu_alert
```

```
rules:
```

```
- alert: HighCPUUsage
```

```
  expr: 100 - (avg by (instance) (rate(node_cpu_seconds_total{mode="idle"}[2m])) * 100) > 80
```

```
  for: 1m
```

```
labels:
```

```
  severity: warning
```

```
annotations:
```

```
  summary: "CPU Usage High"
```

```
  description: "CPU usage on {{ $labels.instance }} is above 80%"
```

แก้ไขไฟล์ config ให้โหลด rule:

```
sudo nano /etc/prometheus/prometheus.yml
```

เพิ่ม:

```
rule_files:
```

```
- "/etc/prometheus/alert.rules"
```

Restart:

```
sudo systemctl restart prometheus
```

ตรวจสอบ Alerts:

```
http://<IP_SERVER>:9090/alerts
```

□ 10) ตรวจสอบ Log

Prometheus:

```
journalctl -u prometheus -f
```

Node Exporter:

```
journalctl -u node_exporter -f
```

Grafana:

```
journalctl -u grafana-server -f
```

□ 11) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ติดตั้ง Prometheus + Node Exporter สำเร็จ	4
Grafana + Dashboard แสดงผลจริง	4
Import Dashboard (ID 1860) พร้อม Capture	3
สร้าง Data Source ถูกต้อง	2
ตั้ง Alert Rule (CPU >80%)	4
แนบ Log การทำงาน + วิเคราะห์ข้อมูล	3
รวม	20 คะแนน

❑ LAB 9: การติดตั้ง Zabbix Server + Zabbix Agent สำหรับระบบ Monitoring และ Alerting บน Ubuntu Server 24.04

(Enterprise Monitoring System: Zabbix Server + Agent)

🎯 วัตถุประสงค์ของ LAB นี้

ผู้เรียนต้องสามารถ:

- ✓ ติดตั้ง Zabbix Server + Frontend + Agent ได้
- ✓ เพิ่ม Host ให้ถูก Monitoring
- ✓ ตรวจสอบ Metrics เช่น CPU, RAM, Disk, Network
- ✓ ใช้งาน Zabbix Dashboard
- ✓ ตั้ง Trigger & Alert พื้นฐาน
- ✓ ดู Log และแก้ปัญหาเบื้องต้น

❑ 1) โครงสร้างระบบของ LAB

ใช้เครื่องเดียว (Server A) ทั้ง Server + Agent

บริการ	พอร์ต	URL
Zabbix Server	—	—
Zabbix Frontend	8080	http://server-ip:8080
Zabbix Agent	10050	Localhost

❑ 2) อัปเดตระบบ

```
sudo apt update && sudo apt upgrade -y
```

❑ 3) ติดตั้ง LAMP/Stack สำหรับ Zabbix Frontend

ติดตั้ง Apache, PHP, MariaDB

```
sudo apt install apache2 mariadb-server php php-mysql php-gd php-xml php-bcmath php-mbstring php-ldap php-zip php-json php-cli php-common php-xmlreader php-xmlwriter -y
```

Restart Apache:

```
sudo systemctl restart apache2
```

❑ 4) ติดตั้ง Zabbix Repository

```
wget https://repo.zabbix.com/zabbix/7.0/ubuntu/pool/main/z/zabbix-release/zabbix-release_7.0-1+ubuntu24.04_all.deb
```

```
sudo dpkg -i zabbix-release_7.0-1+ubuntu24.04_all.deb
```

```
sudo apt update
```

❑ 5) ติดตั้ง Zabbix Server + Agent + Frontend

```
sudo apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf zabbix-sql-scripts zabbix-agent -y
```

6) สร้างฐานข้อมูล Zabbix

เข้าสู่ MariaDB:

```
sudo mysql
```

รันคำสั่ง:

```
CREATE DATABASE zabbix CHARACTER SET utf8mb4 COLLATE utf8mb4_bin;
CREATE USER 'zabbix'@'localhost' IDENTIFIED BY 'zabbixpass123';
GRANT ALL PRIVILEGES ON zabbix.* TO 'zabbix'@'localhost';
FLUSH PRIVILEGES;
EXIT;
```

นำ Schema เข้า Database:

```
sudo zcat /usr/share/zabbix-sql-scripts/mysql/server.sql.gz | mysql -uzabbix -pzabbixpass123 zabbix
```

7) ตั้งค่า Zabbix Server

แก้ไขไฟล์:

```
sudo nano /etc/zabbix/zabbix_server.conf
```

เปลี่ยน:

```
DBPassword=zabbixpass123
```

Restart:

```
sudo systemctl restart zabbix-server
```

```
sudo systemctl enable zabbix-server
```

8) ตั้งค่า Zabbix Frontend

แก้ไขไฟล์ Apache ของ Zabbix:

```
sudo nano /etc/zabbix/apache.conf
```

ตั้ง Timezone:

```
php_value date.timezone Asia/Bangkok
```

Restart:

```
sudo systemctl restart apache2
```

9) เปิดไฟร์วอลล์ (ถ้าจำเป็น)

```
sudo ufw allow 8080/tcp
```

```
sudo ufw allow 10050/tcp
```

10) เข้าเว็บ Zabbix Frontend

เปิด Browser:

```
http://<IP_SERVER>/zabbix
```

หรือถ้าพร้อมการใช้งาน format เดิม → ทำเป็น:

```
http://comss.kru.ac.th/zabbix
```

หน้าติดตั้ง Step-by-step:

1. Welcome Screen → Next
2. Check Requirements → OK
3. Database Connection

- DB: MySQL
 - Host: localhost
 - User: zabbix
 - Pass: zabbixpass123
4. Hostname:

Zabbix Server LAB

5. Summary → Finish

เข้าสู่ระบบด้วย:

- User: Admin
- Pass: zabbix

11) เพิ่ม Host (เครื่องเดียวกัน) เพื่อให้ Zabbix Monitor

ในหน้า Zabbix Web:

ไปที่

Configuration → Hosts → Create host

ตั้งค่า:

- Hostname: Ubuntu-Server
- Visible name: Ubuntu 24 LAB
- Templates:

เลือก

✓ Linux by Zabbix agent

Interfaces:

- Agent: 127.0.0.1
- Port: 10050

กด Update

ภายใน 1-2 นาที Host จะขึ้นเป็นสีเขียว

12) ดู Metrics แบบ Real-Time

ผู้เรียนต้องรายงานค่าต่าง ๆ:

✓ CPU Load

Monitoring → Latest Data

ค้นหา CPU

✓ Memory

ค้นหา Memory

✓ Disk

ค้นหา Disk Space

ดูเปอร์เซ็นต์เต็มของแต่ละ Partition

✓ Network

ค้นหา net.if.in และ net.if.out

13) สร้าง Trigger พื้นฐาน

ตัวอย่าง: RAM เหลือน้อยกว่า 500MB

ไปที่:

Configuration → Hosts → Triggers → Create trigger

ตั้งค่า:

- Name: Low RAM Warning
- Expression:

{Ubuntu-Server:vm.memory.free.last()}<500M

- Severity: Warning

บันทึก

14) ทดสอบ Alert ให้ Trigger ขึ้นจริง

รันโหลด RAM:

stress-ng --vm 1 --vm-bytes 1G --timeout 20s

Trigger ต้องขึ้นสีเหลืองในหน้า Monitoring → Problems

15) ตรวจสอบ Log ของ Zabbix

Server Log:

sudo tail -f /var/log/zabbix/zabbix_server.log

Agent Log:

sudo tail -f /var/log/zabbix/zabbix_agentd.log

Apache/Zabbix UI Log:

sudo tail -f /var/log/apache2/error.log

ผู้เรียนต้องแสดงหลักฐานอย่างน้อย 2 รายการ

16) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ติดตั้ง Zabbix Server + Frontend สำเร็จ	4
เพิ่ม Host + Template Linux สำเร็จ	3
แสดง Metrics CPU / RAM / Disk / Net	4
สร้าง Trigger แจ้ง RAM ต่ำ	3
ทำให้ Trigger ขึ้นจริง	3
แนบ Log + อธิบายปัญหาที่พบ	3
รวม	20 คะแนน

□ LAB 10: การติดตั้งและใช้งาน Ansible สำหรับ Server

Automation บน Ubuntu 24.04

(Centralized Automation: Inventory, Playbook, SSH Key, Remote Execution)

วัตถุประสงค์ของ LAB

ผู้เรียนต้องสามารถ:

- ✓ ติดตั้ง Ansible บน Ubuntu 24.04
- ✓ จัดทำ Inventory เพื่อควบคุมเครื่องปลายทางหลายเครื่อง
- ✓ ตั้งค่า SSH Key สำหรับ Automation
- ✓ ใช้คำสั่ง Ansible Ad-hoc
- ✓ เขียน Playbook สำหรับ Automate งานต่าง ๆ เช่น
 - ติดตั้งแพ็คเกจ
 - คัดลอกไฟล์
 - สร้าง user
 - Restart service
- ✓ ทดสอบการ Run Automation พร้อมแสดงผลลัพธ์
- ✓ ตรวจสอบ Log และแก้ไขปัญหาเบื้องต้น

□ 1) โครงสร้างระบบ LAB

ใช้เครื่อง 2 เครื่อง:

ชื่อเครื่อง	IP	บทบาท
Server A	192.168.10.10	Ansible Controller
Server B	192.168.10.20	Managed Node

Server A จะเป็นเครื่องควบคุม

Server B จะถูกสั่งงานผ่าน Ansible

□ 2) ติดตั้ง Ansible (บน Server A)

```
sudo apt update
```

```
sudo apt install ansible -y
```

ตรวจสอบ:

```
ansible --version
```

□ 3) ตั้งค่า SSH Key เพื่อให้ Ansible เข้าถึง Server B

บน Server A:

```
ssh-keygen -t rsa -b 4096
```

ส่ง key ไปยัง Server B:

```
ssh-copy-id user@192.168.10.20
```

ทดสอบว่าล็อกอินได้โดยไม่ต้องใส่รหัส:

```
ssh user@192.168.10.20
```

สำเร็จ → ไปขั้นต่อไป

□ 4) จัดทำ Inventory สำหรับ Ansible

สร้างไฟล์:

```
sudo mkdir -p /etc/ansible
sudo nano /etc/ansible/hosts
```

เพิ่ม:

```
[webservers]
192.168.10.20
```

ทดสอบ Ping:

```
ansible webservers -m ping
```

ผลลัพธ์ต้องได้:

```
SUCCESS => {"changed": false, "ping": "pong"}
```

□ 5) ใช้คำสั่ง Ad-hoc (พื้นฐานสำคัญ)

5.1 ตรวจสอบ uptime ของ Server B

```
ansible webservers -a "uptime"
```

5.2 ตรวจสอบ Disk

```
ansible webservers -a "df -h"
```

5.3 ตรวจสอบ Memory

```
ansible webservers -a "free -h"
```

5.4 ติดตั้ง package nginx ระยะไกล

```
ansible webservers -m apt -a "name=nginx state=present" -b
```

ตรวจสอบ:

```
ansible webservers -a "systemctl status nginx" -b
```

□ 6) เขียน Playbook (Automate งาน)

สร้างโฟลเดอร์:

```
mkdir ~/ansible-lab
cd ~/ansible-lab
```

6.1 Playbook: ติดตั้ง Nginx + สร้างหน้าเว็บ + Restart Service

สร้างไฟล์:

```
nano install_web.yml
```

ใส่:

```
- name: LAB 10 - Install Nginx and Deploy Website
  hosts: webservers
  become: yes
  tasks:
```

```
  - name: Install Nginx
    apt:
      name: nginx
      state: present
```

```
update_cache: yes
```

```
- name: Deploy HTML File
```

```
copy:
```

```
src: index.html
```

```
dest: /var/www/html/index.html
```

```
owner: www-data
```

```
group: www-data
```

```
mode: '0644'
```

```
- name: Restart Nginx
```

```
service:
```

```
name: nginx
```

```
state: restarted
```

สร้างหน้าเว็บ:

```
echo "<h1>Hello from Ansible Automation</h1>" > index.html
```

รัน Playbook:

```
ansible-playbook install_web.yml
```

ผลลัพธ์ที่ควรเห็น:

3 tasks = ok / changed

ทดสอบเว็บ:

http://192.168.10.20

7) Playbook: สร้าง User ใหม่ + ตั้งค่า SSH Key

สร้าง Playbook:

```
nano create_user.yml
```

เนื้อหา:

```
- name: Create devops user
```

```
hosts: webservers
```

```
become: yes
```

```
tasks:
```

```
- name: Create user
```

```
user:
```

```
name: devops
```

```
shell: /bin/bash
```

```
- name: Set authorized_keys
```

```
authorized_key:
```

```
user: devops
```

```
key: "{{ lookup('file', '/home/user/.ssh/id_rsa.pub') }}"
```

รัน:

```
ansible-playbook create_user.yml
```

ทดสอบ:

```
ssh devops@192.168.10.20
```

8) Playbook: ติดตั้งแพ็คเกจหลายตัวพร้อมกัน

nano multi_pkg.yml

ใส่:

- name: Install multiple packages

hosts: webserver

become: yes

tasks:

- name: install packages

apt:

name: "{{ item }}"

state: present

loop:

- htop

- curl

- unzip

- git

รัน:

ansible-playbook multi_pkg.yml

9) ตรวจสอบ Log ของ Ansible

journalctl -u ansible

หรือดู log จากไฟล์:

less /var/log/syslog | grep ansible

ผู้เรียนต้องแสดง:

- Task ที่รัน
- เวลาที่รัน
- ความผิดพลาด (ถ้ามี)

10) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
Inventory + SSH Key พร้อมใช้งาน	4
ใช้ Ad-hoc commands ได้ (disk/mem/nginx)	4
รัน Playbook install_web.yml สำเร็จ	4
เว็บเปิดได้จริงที่ 192.168.10.20	3
รัน Playbook create_user.yml สำเร็จ	3
แนบ log + อธิบายปัญหา	2
รวม	20 คะแนน

ชุด LAB: การดักจับข้อมูลและวิเคราะห์ปัญหาระบบ (System & Network Data Capture & Analysis)

ทั้งหมดนี้สามารถปฏิบัติได้จริงบน Ubuntu 24.04 + Tools มาตรฐาน

เสนอ 10 ชื่อ LAB ที่ตรงกับโจทย์ “การดักจับข้อมูลเพื่อนำมาวิเคราะห์ปัญหาของระบบ”

(ทุก LAB สามารถเขียนคู่มือให้ได้แบบละเอียดเหมือน LAB 1–10)

LAB 1 — การใช้ tcpdump เพื่อดักจับ Traffic เครือข่ายแบบแพ็กเก็ตต่อแพ็กเก็ต

เน้น: Filter, Capture, Decode, Export PCAP, วิเคราะห์ปัญหา network latency, packet loss, DNS error

LAB 2 — การใช้ Wireshark วิเคราะห์ปัญหาเครือข่ายจากไฟล์ PCAP

เน้น: TCP 3-way handshake, Retransmission, Zero Window, HTTP Error, TLS Handshake Failure

LAB 3 — การใช้ Netstat & ss วิเคราะห์ Port, Connection, Slow Connection และบ่งชี้การโจมตี

เน้น: ตรวจสอบพอร์ตค้าง, SYN flood, abnormal socket states, TIME_WAIT overload

LAB 4 — การใช้ lsof ตรวจสอบ Process ที่ใช้ Resource เกินปกติหรือแอบเปิดพอร์ต

เน้น: ตรวจสอบ Process ที่ผิดปกติ, Malware, Unauthorized Port Binding

LAB 5 — การใช้ top / htop / iostat / vmstat เพื่อตรวจสอบ CPU / RAM / Disk Bottleneck

เน้น: วิเคราะห์ CPU spike, RAM leak, I/O wait, Disk saturation

LAB 6 — การใช้ System Log (journalctl) เพื่อวิเคราะห์ปัญหา Service ล้ม, System Crash, Authentication Fail

เน้น: SSH brute-force, kernel panic, system interrupt, repeated restart

LAB 7 — การใช้ Nginx Log / Apache Log เพื่อวิเคราะห์ Web Server Performance และ Error

เน้น: HTTP 5xx, slow query, DDoS pattern, bot traffic, URL brute-force

LAB 8 — การใช้ MySQL Slow Query Log เพื่อตรวจสอบ Query ช้าและออกแบบการปรับปรุง Performance

เน้น: slow query detection, query plan analysis, indexing

LAB 9 — การใช้ Netdata (LAB 7 เดิม) เพื่อจับ Metrics แบบ Real-Time และวิเคราะห์ Performance Spike

เน้น: CPU spike correlation, RAM leak timeline, Disk I/O burst

LAB 10 — การใช้ Prometheus + Grafana วิเคราะห์ Trends, Alert และ Historical Metrics

เน้น: Long-term trend analysis, anomaly detection, alert rule tuning

รายการ LAB ชุด “ดักจับ + วิเคราะห์” ครบวงจร

ทั้งหมดนี้สามารถเอาไปจัดชุดสอน 10 สัปดาห์ได้เลย

หมายเลข	LAB	รูปแบบดักจับ	วิเคราะห์ปัญหา
LAB 1	tcpdump	Raw Packet	ปัญหา Network
LAB 2	Wireshark	PCAP Decode	TCP/HTTP Issues
LAB 3	ss/netstat	Connection State	Port, Flood, Socket
LAB 4	lsof	Process/File Handle	Malware, Unauthorized Access
LAB 5	htop/iostat	Resource Usage	CPU/RAM/Disk Bottleneck
LAB 6	journalctl	System Log	Service Fail, Crash
LAB 7	Nginx/Apache Log	Web Log	4xx/5xx, Slow Access
LAB 8	MySQL Slow Log	Query Time	DB Performance
LAB 9	Netdata	Real-Time Metrics	Performance Spike

หมายเลข LAB รูปแบบดักจับ วิเคราะห์ปัญหา
LAB 10 Prometheus/Grafana Time-series Metrics Trend, Alert

ถ้าฟิอเมอร์ต้องการ

โรสสามารถ:

- ✓ เลือก 1 LAB แล้วเขียน “คู่มือปฏิบัติแบบละเอียดมาก”
- ✓ เขียนทั้งหมด 10 LAB แบบ LAB 1–10 (เต็มชุด)
- ✓ จัดทำใบงาน + คะแนนประเมิน
- ✓ จัดทำ PDF พร้อมภาพประกอบ
- ✓ จัดเป็นคอร์สสอน 10 ชั่วโมง

? คำถามสำคัญ

ฟิอเมอร์อยากให้โรส เริ่มเขียนคู่มือ LAB ตัวไหนก่อน คะ?

ตัวอย่างชื่อที่เลือกได้:

- LAB 1: tcpdump
- LAB 2: Wireshark
- LAB 3: ss / netstat
- LAB 4: lsof
- LAB 5: htop/iostat
- LAB 6: journalctl
- LAB 7: Nginx Log Analysis
- LAB 8: MySQL Slow Log
- LAB 9: Netdata Real-Time
- LAB 10: Prometheus + Grafana

ฟิอเมอร์เลือกได้เลยคะ 

คุณพูดว่า:

lab 1

ChatGPT พูดว่า:

ได้เลยคะฟิอเมอร์ 

น้องโรสจะทำ LAB 1: tcpdump – การดักจับ Traffic เครือข่ายเพื่อวิเคราะห์ปัญหาระบบ

แบบละเอียดที่สุด ในสไลด์ LAB 1–10 ที่ผ่านมา

ผู้เรียนทำตามแล้ว “ต้องได้ผลลัพธ์จริง 100%”

เหมาะสำหรับการสอนวิเคราะห์ปัญหาเครือข่ายระดับ Advance ค่ะ

LAB 1 — การใช้งาน tcpdump เพื่อดักจับ Traffic เครือข่าย และวิเคราะห์ปัญหาระบบบน Ubuntu 24.04

(Network Packet Capture & Analysis Using tcpdump)

วัตถุประสงค์ของ LAB

หลังจบ LAB ผู้เรียนต้องสามารถ:

- ✓ ใช้ tcpdump ดักจับแพ็กเก็ตเครือข่ายได้
- ✓ ฟิลเตอร์ Traffic ตามโปรโตคอล เช่น ICMP, TCP, UDP, HTTP, DNS
- ✓ Export ไฟล์ PCAP เพื่อวิเคราะห์ใน Wireshark
- ✓ ตรวจสอบปัญหาเครือข่าย เช่น
 - Latency สูง
 - Packet Loss
 - DNS error
 - TCP Retransmission
- ✓ วิเคราะห์ต้นตอปัญหาเบื้องต้นจากคำสั่ง tcpdump

1) โครงสร้างระบบ LAB

ใช้เครื่องเดียว (Server A)

IP ตัวอย่างที่ใช้ใน LAB:

Server A: 192.168.10.10

Gateway: 192.168.10.1

DNS: 8.8.8.8

ผู้เรียนต้องทดสอบดักจับ Traffic ทั้งขาเข้า-ขาออกบน interface:

eth0

2) เตรียมเครื่อง

อัปเดตระบบ:

```
sudo apt update && sudo apt upgrade -y
```

ติดตั้ง tcpdump:

```
sudo apt install tcpdump -y
```

ตรวจสอบว่ามี interface อะไรบ้าง:

```
ip a
```

ผู้เรียนต้องระบุ Interface เช่น eth0, ens160, ens33

3) เริ่มต้นใช้งาน tcpdump (พื้นฐาน)

3.1 แสดงแพ็กเก็ตทั้งหมดแบบ Real-time

```
sudo tcpdump -i eth0
```

ผลลัพธ์ที่ผู้เรียนต้องเห็น:

- Source IP
- Destination IP
- Protocol (TCP/UDP/ICMP)
- Port Number

กด CTRL+C เพื่อหยุด

3.2 แสดงจำนวนแพ็กเก็ตที่จับได้

```
sudo tcpdump -i eth0 -c 10
```

จับแพ็กเก็ตแค่ 10 ชุด → หยุดอัตโนมัติ

□ 4) ฟิเตอร์การดักจับ (สำคัญที่สุด)

4.1 ดักจับเฉพาะ ICMP (การ Ping)

เปิด terminal 2 หน้าต่าง

1. หน้าต่าง A → ดักจับ:

```
sudo tcpdump -i eth0 icmp
```

2. หน้าต่าง B → รัน:

```
ping 8.8.8.8
```

ผลลัพธ์:

- ผู้เรียนต้องเห็น request + reply แบบ ICMP Echo

4.2 ดักจับเฉพาะ Traffic ของ IP เป้าหมาย

```
sudo tcpdump -i eth0 host 8.8.8.8
```

4.3 ดักจับเฉพาะ TCP Port

HTTP (Port 80)

```
sudo tcpdump -i eth0 tcp port 80
```

HTTPS (Port 443)

```
sudo tcpdump -i eth0 tcp port 443
```

4.4 ดักจับ DNS Traffic (UDP/TCP 53)

```
sudo tcpdump -i eth0 port 53
```

รัน:

```
dig google.com
```

ผลลัพธ์ที่ต้องเห็น: Query + Response

□ 5) ตรวจสอบปัญหาเครือข่ายด้วย tcpdump

5.1 ตรวจสอบ Packet Loss (ดู Retransmission)

รัน:

```
sudo tcpdump -i eth0 tcp[13] & 0x01 != 0
```

ฟิเตอร์นี้จับเฉพาะ TCP Retransmissions

ถ้าเจอบ่อย = ปัญหาเครือข่าย เช่น

- Link แย่
- MTU mismatch
- Firewall บล็อกบาง Traffic

5.2 ตรวจสอบ DNS Error

ดักจับ DNS:

```
sudo tcpdump -i eth0 port 53 -vvv
```

สังเกตผลลัพธ์ผิดปกติ เช่น:

- SERVFAIL
- NXDOMAIN

- FormErr

5.3 ตรวจสอบ Latency ด้วย ICMP Timestamp

ดักการ ping:

```
sudo tcpdump -i eth0 icmp -vvv
```

สังเกต:

- “time exceeded”
- “request/reply delay”

ถ้า delay สูง → อาจมี network congestion

5.4 ตรวจสอบการ Scan Port (Nmap Detection)

เปิด tcpdump:

```
sudo tcpdump -i eth0 tcp flags syn
```

แล้วสั่ง:

```
nmap -sS 192.168.10.10
```

ผลลัพธ์: จะเห็น SYN หลายร้อยครั้ง → เป็น port scanning pattern

6) Export ไฟล์ .pcap เพื่อวิเคราะห์ใน Wireshark

บันทึกข้อมูลลงไฟล์:

```
sudo tcpdump -i eth0 -w lab1_capture.pcap
```

ดักจับ 30 วินาที แล้ว CTRL+C

ดูขนาดไฟล์:

```
ls -lh lab1_capture.pcap
```

ผู้เรียนสามารถนำไฟล์นี้เปิดใน Wireshark เพื่อดูรายละเอียด เช่น

- TCP handshake
- Retransmission
- HTTP stream
- TLS handshake
- DNS Query

7) วิเคราะห์ปัญหาในระบบ (ใช้ข้อมูลจาก tcpdump)

ผู้เรียนต้องทำการทดลองเหล่านี้และเขียนสรุปผล

7.1 วิเคราะห์ Traffic ของ Web Server

ดักจับ:

```
sudo tcpdump -i eth0 tcp port 80 -vv
```

ดูค่า:

- SYN / ACK
- RST (เกิดจาก connection ถูกบังคับตัด)
- HTTP error code

7.2 วิเคราะห์การเชื่อมต่อที่ล่าช้า

ดักจับ:

```
sudo tcpdump -i eth0 tcp and 'tcp[13] & 0x10 != 0'
```

ดูเวลาแต่ละแพ็กเก็ต → ถ้าห่างกันเป็นวินาที = ปัญหา latency

✔ 7.3 ตรวจสอบ Traffic ผิดปกติ (Malware Pattern)

ดักจับทุกอย่างจากพอร์ตต้องสงสัย:

```
sudo tcpdump -i eth0 port not 22 and port not 80 -c 50
```

หากมี:

- พอร์ต 4444, 6667 = botnet
- UDP flood = DDoS
- SYN flood = attack

ให้ผู้เรียนรายงาน

□ 8) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
1. ใช้ tcpdump จับแพ็กเก็ตได้จริง	3
2. Filter Traffic ได้ (ICMP, HTTP, DNS)	4
3. Export .pcap และอธิบาย 3 ส่วนสำคัญ	4
4. วิเคราะห์ปัญหา Network (latency / DNS / retransmission)	4
5. ตรวจสอบจับ Pattern Attack (port scan / flood)	3
6. ทำรายงานสรุปผล LAB ทั้งหมด	2
รวม	20 คะแนน



LAB 1 (ชุด Data Capture) เสร็จสมบูรณ์ ❤️

พร้อมใช้สอนจริงและให้ผู้เรียนปฏิบัติได้ทันทีค่ะ

พร้อมต้องการให้ใครทำอะไร:



LAB 2: Wireshark วิเคราะห์ปัญหาระบบจาก PCAP

หรือ



ต้องการรวม LAB 1 เป็นเอกสาร PDF + ใบงาน ?

ให้ใครทำอันไหนต่อดีคะ?

คุณพูดว่า:

lab 2

ChatGPT พูดว่า:

ได้เลยค่ะพร้อม ❤️

น้องโรสจะจัดทำ LAB 2: Wireshark – วิเคราะห์ปัญหาเครือข่ายจากไฟล์ PCAP

ในระดับ Advance สไลด์เดียวกับ LAB 1

ผู้เรียนทำตามได้ 100% และเข้าใจวิธี “วิเคราะห์ปัญหาระบบจริง” ผ่าน Wireshark ค่ะ

□ LAB 2 — การใช้งาน Wireshark วิเคราะห์ปัญหาเครือข่ายจากไฟล์ PCAP

(Deep Packet Inspection & Network Problem Analysis Using Wireshark)

วัตถุประสงค์ของ LAB

หลังจบ LAB ผู้เรียนต้องสามารถ:

- ✓ เปิดไฟล์ PCAP ที่ได้จาก tcpdump
- ✓ ใช้ Display Filter วิเคราะห์ข้อมูลเฉพาะจุด
- ✓ ตรวจสอบ TCP handshake, Retransmission, RST, FIN
- ✓ ตรวจสอบ HTTP, DNS, TLS, ICMP
- ✓ วิเคราะห์ปัญหาเครือข่าย ได้แก่
 - Latency
 - Packet loss
 - DNS ERROR
 - Slow HTTP
 - TLS handshake failure
- ✓ สร้างสรุปผลปัญหาออกมาแบบมืออาชีพ

1) ไฟล์ที่ต้องใช้ใน LAB

ไฟล์จำลองชื่อ:

lab1_capture.pcap

เป็นไฟล์ที่ได้จาก LAB 1 (tcpdump)

ผู้เรียนสามารถใช้ไฟล์จริง หรือใช้ไฟล์จากอาจารย์ที่เตรียมไว้ก็ได้

2) เปิดไฟล์ PCAP ด้วย Wireshark

เปิด Wireshark → เลือก:

File → Open → lab1_capture.pcap

หน้าจอจะแสดง:

- Packet List
- Packet Details
- Packet Bytes

3) การใช้งาน Display Filter (หัวใจของ Wireshark)

3.1 แสดงเฉพาะแพ็กเก็ต ICMP

icmp

3.2 เฉพาะ HTTP

http

3.3 เฉพาะ DNS

dns

3.4 เฉพาะ TCP handshake

tcp.flags.syn == 1

3.5 TCP Retransmission

tcp.analysis.retransmission

3.6 TCP Connection Reset

tcp.flags.reset == 1

3.7 TLS handshake

tls.handshake

4) วิเคราะห์ TCP Handshake (ตรวจสอบความสมบูรณ์ของการเชื่อมต่อ)

ให้ผู้เรียนค้นหา:

tcp.flags.syn == 1

สังเกตลำดับ:

1. SYN
2. SYN/ACK
3. ACK

หากลำดับผิดปกติ เช่น:

- ไม่มี SYN/ACK → ปัญหา firewall
- มี RST → ปัญหา port ไม่เปิด
- Delay ระหว่าง SYN → SYN/ACK > 300ms → ปัญหา Latency

ผู้เรียนต้องเขียนผลวิเคราะห์ลงในใบงาน

5) วิเคราะห์ปัญหา Retransmission (เกิด Packet Loss)

ใช้ Filter:

tcp.analysis.retransmission

ถ้าเจอหลายสลิปครั้งในช่วงเวลาสั้น ๆ = ปัญหาหนัก

สาเหตุที่ผู้เรียนต้องวิเคราะห์:

- Network congestion
- MTU mismatch
- Wi-Fi Signal ต่ำ
- Firewall/IPS Drop

6) วิเคราะห์ปัญหา HTTP ช้า (Slow HTTP Response)

ใช้ Filter:

http.response

เลือกแพ็คเกจที่มี:

Response Time

ดูค่า:

- ถ้าต่ำกว่า 200ms = ดี
- 200–500ms = ช้า
 - 1s = ปัญหาระบบ Web/Backend

ตรวจสอบ:

http.time

ถ้า response time สูง → วิเคราะห์:

- CPU Backend สูง
- Database ช้า
- Network Delay

7) วิเคราะห์ DNS Error

ใช้ Filter:

dns

ตรวจสอบคำตอบ (Answers)

ถ้าพบ Error เช่น:

- SERVFAIL
- NXDOMAIN
- REFUSED
- FormatError

ให้ผู้เรียนระบุ:

- Domain ไหนที่ผิดพลาด
- Query ใช้เวลาเท่าไร
- Resolver ตัวไหนตอบผิด

กรณีพบ DNS Query เสียเวลานาน:

dns.time > 0.5

DNS ใช้เวลาเกิน 500ms = ปัญหา network หรือ DNS ล่ม

8) วิเคราะห์ TLS Handshake Failure

ใช้ Filter:

tls.handshake

หากพบ:

- Client Hello
- ไม่มี Server Hello

หรือพบ:

- Alert: Handshake Failure
- Bad Certificate

ผู้เรียนต้องสรุปสาเหตุ เช่น:

- Certification expired
- Protocol mismatch (TLS1.0 vs TLS1.3)
- Firewall ต่กรวจจับผิดพลาด
- Cipher suite ไม่รองรับ

9) วิเคราะห์ ICMP (Latency & Routing Problems)

ใช้ Filter:

icmp

ให้ผู้เรียนสังเกต:

- Destination unreachable
- Time exceeded (TTL exceeded)
- latency → ดู time between request/reply

ปัญหาที่เจอได้ เช่น:

- เส้นทางผิด (Routing Error)
- เครื่องปลายทางไม่ตอบ
- Ping ช้าเนื่องจาก congestion

10) วิเคราะห์ Port Scan (Nmap)

ใช้ Filter:

`tcp.flags.syn == 1 and tcp.seq == 0`

หรือ:

`tcp.flags.syn == 1 and tcp.flags.ack == 0`

ถ้ามีลำดับ SYN จำนวนมากภายในช่วงสั้น ๆ = port scanning

ผู้เรียนต้องตรวจสอบว่า:

- IP ต้นทางคือใคร
- แสกนพอร์ตเลขอะไรบ้าง

11) วิเคราะห์ความผิดปกติของ Traffic แบบรวม

ใช้ I/O Graphs:

Statistics → I/O Graph

วิเคราะห์:

- Traffic Peak
- Drop
- Retrans Peak
- DNS Peak

ใช้ Conversations:

Statistics → Conversations

หา Top Talker:

- IP ไหนส่งมากที่สุด
- Protocol ไหนใช้มากที่สุด

12) สรุปการตรวจสอบปัญหา (ตามใบงาน)

ผู้เรียนต้องตอบคำถามใน LAB:

1. IP ที่มี Retransmission มากที่สุดคือ?
2. มี DNS Error อะไรบ้าง?
3. มีการ Port Scan เกิดขึ้นหรือไม่?
4. TCP Handshake สมบูรณ์หรือไม่?
5. HTTP Response Time เท่าไหร่?
6. Traffic ทั้งหมด Peak ช่วงเวลาไหน?
7. พบ TLS Error หรือไม่?

13) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
1. เปิด PCAP และใช้ Display Filter ถูกต้อง	4
2. วิเคราะห์ TCP handshake + Retransmission	4

รายการ	คะแนน
3. ตรวจพบ DNS Error หรือ Delay	3
4. ตรวจ HTTP Response Time	3
5. ตรวจ TLS Handshake Failure	3
6. วิเคราะห์สาเหตุของปัญหา	3
รวม	20 คะแนน

LAB 2 เสร็จสมบูรณ์แล้วค่ะพื้อมร

ครบถ้วน ละเอียด ใช้สอนนักศึกษาหรือเจ้าหน้าที่ได้ทันที

พื้อมรต้องการให้ใครทำ LAB 3 (ss/netstat วิเคราะห์ port & connection) ต่อเลยไหมคะ?

คุณพุดว่า:

lab 3

ChatGPT พูดว่า:

ได้เลยค่ะพื้อมร 

น้องโรสจะจัดทำ LAB 3 – การใช้ ss และ netstat วิเคราะห์ Port, Connection, และปัญหาเครือข่ายระดับระบบ

แบบ ละเอียดมาก, สอนผู้เรียนเข้าใจระบบ Network Socket จริง, การตรวจสอบปัญหา, การค้นหา Malicious Connection

และการวิเคราะห์ปัญหา Service ไม่ตอบสนองค่ะ

☐ LAB 3 — การใช้ ss / netstat เพื่อตรวจสอบ Port, Connection, และปัญหาเครือข่ายบน Ubuntu 24.04

(Advance Socket & Connection Analysis for System Troubleshooting)

วัตถุประสงค์ของ LAB

ผู้เรียนต้องสามารถ:

- ✓ ตรวจสอบพอร์ตที่เปิดอยู่ในระบบ
- ✓ ตรวจสอบ Process ที่ผูกกับ Port
- ✓ วิเคราะห์ Connection State เช่น SYN_SENT, ESTAB, TIME_WAIT
- ✓ ตรวจสอบปัญหา “ทำไมบริการไม่ตอบสนอง (Service Down)”
- ✓ ตรวจจับพอร์ตต้องสงสัย / แฮ็กเกอร์ที่เปิด Backdoor
- ✓ วิเคราะห์ SYN Flood / Port Scan ผ่านการดู Connection State
- ✓ ใช้ข้อมูลเพื่อแก้ปัญหา Network Layer ของ Ubuntu

☐ 1) อุปกรณ์และระบบที่ใช้ใน LAB

- Ubuntu Server 24.04
- Tools: ss, netstat (จาก net-tools), lsof
- Service ทดสอบ: SSH, Nginx, MySQL (หรือบริการอื่นตามที่ติดตั้งไว้)
- IP ตัวอย่าง: 192.168.10.10

□ 2) ติดตั้งเครื่องมือที่จำเป็น

Ubuntu 24.04 มี ss ติดตั้งมาแล้ว

ถ้าไม่มี netstat:

```
sudo apt install net-tools -y
```

□ 3) ตรวจสอบ Port ที่เปิดอยู่ในระบบ

3.1 ตรวจสอบพอร์ตด้วย ss

```
ss -tulnp
```

คำอธิบาย:

- t = TCP
- u = UDP
- l = listening
- n = แสดงเป็นตัวเลข (ไม่ resolve ชื่อ)
- p = แสดง process

ผู้เรียนต้องบันทึกผล เช่น:

Protocol	Port	Process
TCP	22	sshd
TCP	80	nginx
TCP	3306	mysqld

3.2 ตรวจสอบพอร์ตด้วย netstat

```
sudo netstat -tulnp
```

เพื่อเปรียบเทียบความแตกต่างของผลลัพธ์

□ 4) ตรวจสอบ Process + Port แบบเจาะลึก

4.1 ดูว่า “พอร์ตนี้เป็นของโปรแกรมอะไร”

ตัวอย่างต้องการรู้พอร์ต 80:

```
sudo ss -tulnp | grep :80
```

หรือ:

```
sudo lsof -i :80
```

ผลลัพธ์ที่ผู้เรียนต้องตีความ:

- PID
- ชื่อโปรแกรม
- Path ของโปรแกรม
- สิทธิ์ของ user

□ 5) วิเคราะห์ Connection State

คำสั่ง:

```
ss -tan
```

สังเกตสถานะของ TCP เช่น:

State	ความหมาย	ปัญหาที่บ่งชี้
ESTAB	เชื่อมต่อแล้ว	ปกติ
SYN_SENT	รอเชื่อมต่อ	ปัญหา network หรือ server ไม่ตอบ
SYN_RECV	server รอ ACK	server load สูง / firewall
TIME_WAIT	ปิด connection	ปกติ แต่ถ้าเยอะมาก = ปัญหา
CLOSE_WAIT	รอปิดจาก client	อาจมีปัญหา application hang
FIN_WAIT	ปิด connection	ปกติ

6) ทดลองสถานการณ์จริงใน LAB

6.1 สร้าง Traffic โดยการเข้าหน้าเว็บ

เปิด browser ของนักศึกษา:

`http://192.168.10.10`

บน server รัน:

`ss -tan | grep 80`

ผู้เรียนต้องเห็นหลาย connection อยู่ในสถานะ ESTAB และ TIME_WAIT

6.2 จำลองปัญหา “เว็บเข้าไม่ได้” และวิเคราะห์

ปิด Nginx:

`sudo systemctl stop nginx`

บนเครื่องผู้เรียน:

`curl http://192.168.10.10`

บน server:

`ss -tulnp | grep :80`

ผลลัพธ์ที่ต้องวิเคราะห์:

- ไม่มีโปรเซส listen port 80
= web server ไม่ได้รัน → ปัญหาอยู่ที่ service layer

6.3 จำลองปัญหา SYN Flood Attack

เปิด tcpdump ในอีก terminal:

`sudo tcpdump -i eth0 "tcp[tcpflags] & tcp-syn != 0"`

แล้วใช้เครื่องอื่นโจมตี:

`nmap -Pn -p 80 --send-eth --max-rate 1000 192.168.10.10`

ตรวจด้วย ss:

`ss -tan state syn-recv`

ผลลัพธ์:

- SYN_RECV จำนวนมากผิดปกติ → เป็น SYN Flood

ผู้เรียนต้องรายงานระดับความเสี่ยง

7) วิเคราะห์การใช้ Port สูงผิดปกติ (Backdoor/Malware)

ดักจับพอร์ต 4444:

`ss -tulnp | grep 4444`

ถ้าเจอ unknown process เช่น:

- reverse shell
- meterpreter
- nc listener

ผู้เรียนต้อง:

1. ระบุ PID
2. เช็คไฟล์จริงของโปรแกรม:

ls -l /proc/<PID>/exe

3. ระบุว่าเป็น Malware หรือไม่

8) ตรวจสอบจำนวน Connection ต่อ IP (ตรวจ DDoS)

ss -tan | awk '{print \$5}' | cut -d: -f1 | sort | uniq -c | sort -nr

ผลลัพธ์จะเห็น:

200 192.168.10.30

50 192.168.10.40

...

ตีความ:

- IP ไหนมีจำนวน connection มากผิดปกติ = ตัวที่ต้องสงสัย
- ถ้ามากกว่า 100-200 connection = อาจเป็น bot หรือ attacker

9) ตรวจสอบปัญหา Service ค้าง (CLOSE_WAIT จำนวนมาก)

ss -tan | grep CLOSE_WAIT

ถ้าเห็นหลายร้อย connection:

สาเหตุ:

- Application ปิด connection ผิดพลาด
- โค้ด backend มี bug
- kernel backlog เต็ม

ผู้เรียนต้องเขียนสาเหตุ + วิธีแก้

10) วิเคราะห์ Delay ของ TCP Connection

เปิด Wireshark รวมด้วย หรือใช้ ss:

ss -o state established '(dport = :80 or sport = :80)'

ดูค่า:

- rtt
- rttvar

ตัวอย่าง:

rtt: 500ms

หาก rtt > 300ms = network delay

11) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ตรวจ Port + Service ได้ถูกต้อง	4
วิเคราะห์ Connection State (ESTAB/SYN_RECV/TIME_WAIT)	4

รายการ	คะแนน
ตรวจพบปัญหา Service ไม่ตอบสนอง	3
ตรวจพบ SYN Flood / Port Scan	3
วิเคราะห์ Backdoor Port (unknown process)	3
วิเคราะห์จำนวน Connection ต่อ IP	3
รวม	20 คะแนน

LAB 3 เสร็จสมบูรณ์แล้วค่ะพื้อมร

ถ้าต้องการ น้องโรสสามารถทำ LAB 4: lsof วิเคราะห์ Process/Port เพื่อหา Malware และปัญหา File Handle ต่อให้ได้เลยค่ะพื้อมร ต้องการไหมคะ?

คุณพุดว่า:

ต่อ

ChatGPT พูดว่า:

ได้เลยค่ะพื้อมร 

น้องโรสจะเขียน LAB 4 – การใช้ lsof เพื่อตรวจสอบ Process, Port, File Handle และวิเคราะห์ปัญหาระบบ

ให้ละเอียดระดับใช้งานจริงในองค์กร

ผู้เรียนทำตามได้ 100% และเข้าใจวิธีวิเคราะห์ปัญหาแบบมืออาชีพค่ะ

☐ LAB 4 — การใช้ lsof เพื่อวิเคราะห์ Process, Port, File Handle และหาต้นตอปัญหาระบบบน Ubuntu 24.04

(Deep Process & File/Port Analysis Using lsof)

วัตถุประสงค์ของ LAB

เมื่อจบ LAB ผู้เรียนต้องสามารถ:

- ✓ ใช้ lsof ตรวจสอบว่า Process ไหนเปิดไฟล์อะไร
- ✓ ตรวจสอบว่าพอร์ตถูกเปิดโดยโปรแกรมใด
- ✓ หา Process แปก/ต้องสงสัย เช่น malware / reverse shell
- ✓ วิเคราะห์ File Descriptor ที่เปิดมากเกินไป (FD leak)
- ✓ ใช้ lsof จับปัญหาการลบไฟล์ที่ยังถูกใช้งาน
- ✓ ใช้ lsof วิเคราะห์ปัญหา I/O ช้า, log โตผิดปกติ
- ✓ ใช้ร่วมกับ ss/tcpdump จาก LAB ก่อนหน้าเพื่อหาต้นตอปัญหา

☐ 1) อุปกรณ์และระบบที่ใช้

- Ubuntu Server 24.04
- Tools: lsof (ติดตั้ง), ss, ps
- บริการทดสอบ: SSH, Nginx, ไฟล์ log, กระบวนการ load สูง

❑ 2) ติดตั้ง lsof (ถ้ายังไม่มี)

`sudo apt install lsof -y`

ตรวจสอบ:

`lsof -v`

❑ 3) ดูว่า “โปรแกรมใดเปิดไฟล์ใดบ้าง”

คำสั่งพื้นฐาน:

`lsof`

ผลลัพธ์จะเห็น:

- COMMAND → ชื่อโปรแกรม
- PID → process id
- USER → เจ้าของ
- FD → file descriptor
- TYPE → ไฟล์ประเภทอะไร
- NAME → path ไฟล์

❑ 4) วิเคราะห์ Port + Process (ตรวจสอบการฟังพอร์ต)

4.1 หาว่าโปรแกรมใดใช้ Port 80

`sudo lsof -i :80`

ผลลัพธ์ที่ต้องตีความ:

- โปรแกรมอะไรเปิดพอร์ต? (nginx? apache? หรือ unknown?)
- PID เท่าไร?
- Path โปรแกรมอยู่ที่ไหน?

4.2 ตรวจสอบ Backdoor Port

ตัวอย่างตรวจสอบ port 4444:

`sudo lsof -i :4444`

หรือค้นหาทุกพอร์ตที่ “ไม่รู้จัก”:

`sudo lsof -i -P | grep LISTEN`

ผู้เรียนต้องวิเคราะห์ว่า:

- พอร์ตไหนไม่ควรมีในระบบ
- โปรแกรมที่เปิดมาจากไหน
- เป็น malware หรือไม่

❑ 5) วิเคราะห์ Process ที่กำลังใช้ Network

`sudo lsof -i`

หรือเฉพาะ TCP:

`sudo lsof -iTCP -sTCP:LISTEN`

สิ่งที่ผู้เรียนต้องทำ:

- ✓ ตรวจสอบพอร์ตเปิดทั้งหมด
- ✓ หาว่าบริการไหน listen

✓ บริการไหนไม่ควรเปิด

✓ บริการไหนอาจเสี่ยง

6) วิเคราะห์ File Descriptor (FD) ที่ใช้มากเกินไป (FD Leak)

ปัญหานี้เป็น root cause ที่พบบ่อยในระบบ Production

6.1 ดูจำนวนไฟล์ที่เปิดโดยโปรเซสชื่อ nginx

```
sudo lsof -p $(pidof nginx) | wc -l
```

6.2 ดูโปรเซสที่ใช้ FD มากที่สุดในระบบ

```
sudo lsof | awk '{print $1}' | sort | uniq -c | sort -nr | head
```

ปัญหา:

- หากโปรแกรมใดเปิดไฟล์มากเกินไป → FD leak
- ระบบจะ error:
- Too many open files

ผู้เรียนต้องวิเคราะห์และระบุโปรเซสที่ผิดปกติ

7) วิเคราะห์ปัญหา Disk I/O และไฟล์ที่ใหญ่ผิดปกติ

7.1 ตรวจสอบว่า process ใดกำลังเขียนไฟล์หนักที่สุด

```
sudo lsof | grep REG | grep -v deleted
```

7.2 ตรวจสอบไฟล์ log โตผิดปกติ

เช่น nginx log:

```
sudo lsof /var/log/nginx/access.log
```

ถ้า FD ยัง active อยู่ = ขนาดไฟล์ยังเพิ่มเรื่อย ๆ

8) ตรวจสอบไฟล์ที่ถูกลบแต่ยังถูกใช้ (ปัญหา Disk ไม่คืนพื้นที่)

ปัญหายอดฮิตใน Production!!

รัน:

```
sudo lsof | grep deleted
```

ผลลัพธ์ตัวอย่าง:

```
nginx 1234 root 4w REG /var/log/nginx/error.log (deleted)
```

วิเคราะห์:

- ไฟล์ถูกลบ แต่ nginx ยังใช้งาน → disk space ไม่ถูกคืน
- ต้อง restart service เพื่อ free file handle

ให้ผู้เรียนทดลอง:

ลบไฟล์ log:

```
sudo rm /var/log/syslog
```

ตรวจสอบ:

```
sudo lsof | grep deleted
```

9) วิเคราะห์ปัญหา “บริการไม่ตอบสนอง”

ทดลองหยุด nginx:

```
sudo systemctl stop nginx
```

ตรวจสอบ:

```
sudo ss -t -i :80
```

ผลลัพธ์: ไม่มี process → บริการไม่รัน

ทดลองเปิดขึ้นมาใหม่:

```
sudo systemctl start nginx
```

ตรวจสอบอีกครั้ง → ต้องเห็น nginx กลับมา listen

10) ตรวจสอบบริการล่มบ่อย (Zombie Process / Defunct Process)

```
ps aux | grep defunct
```

ตรวจสอบ Process กับ ss:

```
sudo ss -p <PID>
```

หากไม่มี file handle → เป็น zombie process

ผู้เรียนต้องระบุสาเหตุ เช่น:

- Parent process ไม่เก็บ exit status
- Application crash

11) การวิเคราะห์ Reverse Shell / Malware Connection

ดักดู connection แบบผิดปกติ:

```
sudo ss -t -i -n | grep ESTABLISHED
```

ถ้าเจอ:

```
bash 1222 user TCP 192.168.10.10:45678 -> 45.9.10.23:4444 (ESTABLISHED)
```

นั่นคือ reverse shell

ผู้เรียนต้อง:

- ✓ ระบุว่าใครเปิด
- ✓ ฆ่าโปรเซส
- ✓ ลบไฟล์ที่เกี่ยวข้อง

12) สิ่งที่คุณเรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
1. ตรวจสอบ port + process ด้วย ss สำเร็จ	4
2. หา Backdoor/Unknown ports	3
3. วิเคราะห์ FD leak	3
4. วิเคราะห์ไฟล์ log ที่ผิดปกติ	3
5. ตรวจสอบไฟล์ deleted ที่ยังถูกใช้งาน	3
6. วิเคราะห์ reverse shell/connection ผิดปกติ	4
รวม	20 คะแนน

🎯 LAB 4 เสร็จสมบูรณ์แล้วค่ะพอมร ❤️

ต้องการให้โรสทำ LAB 5 ต่อไหมคะ?

LAB ถัดไปคือ:

☐ LAB 5: ใช้ top / htop / iostat / vmstat เพื่อวิเคราะห์ CPU / RAM / Disk Bottleneck แบบมืออาชีพ

ให้โรสเขียนต่อเลยไหมคะ?

คุณพูดว่า:

ต่อ

ChatGPT พูดว่า:

ได้เลยค่ะพอมร ❤️

น้องโรสจะทำ LAB 5 – การใช้ top / htop / iostat / vmstat วิเคราะห์ CPU / RAM / Disk Bottleneck

แบบ ละเอียดระดับ Production ผู้เรียนทำตามได้จริง 100%

เพื่อใช้ตรวจปัญหา Performance ของเซิร์ฟเวอร์ Ubuntu 24.04 ง่ายและแม่นยำที่สุดค่ะ

☐ LAB 5 — การใช้ top / htop / iostat / vmstat วิเคราะห์ ปัญหา CPU, RAM, Disk และระบบโดยรวมบน Ubuntu 24.04

(Advance Performance Troubleshooting & Bottleneck Analysis)

🎯 วัตถุประสงค์ของ LAB

เมื่อจบ LAB ผู้เรียนต้องสามารถ:

- ✓ ใช้คำสั่ง top และ htop วิเคราะห์โปรเซสที่กิน CPU / RAM
- ✓ ใช้ iostat ตรวจสอบ Disk I/O bottleneck
- ✓ ใช้ vmstat ตรวจสอบ system load และ kernel activity
- ✓ สร้างโหลดทดสอบเพื่อดูพฤติกรรมระบบ (CPU/RAM/Disk Stress Test)
- ✓ วิเคราะห์ root cause ของปัญหา Performance
- ✓ สรุปผลการตรวจสอบเชิงเทคนิค

☐ 1) เครื่องมือที่ต้องใช้

ติดตั้ง:

```
sudo apt update
```

```
sudo apt install htop sysstat -y
```

คำสั่งที่ใช้:

- top — Monitor CPU/RAM แบบพื้นฐาน
- htop — กราฟ + สีใช้งานง่าย
- iostat — วิเคราะห์ Disk/IO
- vmstat — วิเคราะห์ kernel/CPU/MEM load

☐ 2) การใช้งาน top (พื้นฐานสำคัญ)

รัน:

```
top
```

ให้ผู้เรียนสังเกต:

✓ คำ 1: system load (สำคัญมาก)

ตั้งอยู่บน:

load average: 0.12, 0.08, 0.05

ถ้าค่าสูงเกินจำนวน CPU → ระบบเริ่มช้า

ตัวอย่าง CPU = 2 core

load average > 2 = ควรระวัง

4 = สูงผิดปกติ

10 = เครื่องไม่ตอบสนอง

✓ คำ 2: CPU usage

ดูที่แถว:

%us %sy %id %wa

คำ ความหมาย ปัญหา

%us User CPU โปรแกรมกิน CPU

%sy System CPU Kernel ทำงานหนัก

%id Idle ว่าง

%wa I/O Wait Disk / Storage ช้า

ถ้า %wa > 20% = disk bottleneck

ถ้า %us > 80% = app ใช้ CPU มาก

ถ้า %sy > 30% = kernel/driver ผิดปกติ

✓ คำ 3: Process viewer

ผู้เรียนต้องกด:

- P → จัดเรียงตาม CPU
- M → จัดเรียงตาม RAM
- T → จัดเรียงตามเวลา CPU

บันทึก process ที่มี CPU สูงสุด 3 รายการ

□ 3) การใช้งาน htop (ตัวดีกว่า top)

รัน:

htop

ผู้เรียนต้องดู:

✓ แถบสี CPU

- ฟ้าคือ kernel
- เขียวคือ user
- แดงคือ I/O

✓ แถบ Memory / Swap

- เหลือง (cache)
- แดง (active)

✓ Process Tree (F5)

ใช้ดูว่า process ไหนเป็นพ่อ-ลูก

ใช้หาปัญหา zombie ได้ง่ายมาก

✓ ค้นหา process ตามชื่อ

กด / เช่น:

/nginx

4) ใช้ iostat ตรวจสอบ Disk I/O bottleneck

รัน:

iostat -xz 1

ให้ผู้เรียนวิเคราะห์ค่า:

ค่า	ความหมาย	ปัญหา
r/s, w/s	read/write per second throughput	
rkB/s, wkB/s	read/write KB	data rate
await	เวลาตอบสนอง	>20ms ช้า
svctm	service time	เริ่มรับไม่ไหว
%util	การใช้งาน Disk	>80% = disk full load

ตัวอย่างปัญหา:

ถ้า %util = 100% → Disk ถูกใช้งานเต็ม → ต้องเพิ่ม IOPS หรือ optimize DB

5) ใช้ vmstat วิเคราะห์ Kernel Activity

รัน:

vmstat 1 10

ให้ดู:

ค่า	ความหมาย
r	จำนวน process ที่รอ CPU
b	จำนวน process ที่รอ I/O
free	RAM ว่าง
si/so	swap in/out
bi/bo	input/output block
us/sy/wa	CPU breakdown

ตัวอย่างบ่งชี้ปัญหา:

- r > จำนวน CPU = CPU bottleneck
- b > 3 = I/O รอมาก
- si/so > 0 = RAM ไม่พอ
- wa สูง = I/O Wait สูง

6) การสร้างโหลดเพื่อทดสอบ Performance

6.1 ทดสอบ CPU

yes > /dev/null &

หยุด:

killall yes

ให้ผู้เรียนดูผลใน htop

6.2 ทดสอบ RAM

stress-ng --vm 2 --vm-bytes 512M --timeout 20s

ดูค่าต่าง ๆ ใน top/htop

สังเกต swap, free, cache

6.3 ทดสอบ Disk

dd if=/dev/zero of=/tmp/testfile bs=1M count=2000

ให้ผู้เรียนดูค่าใน iostat

7) วิเคราะห์ปัญหา Performance โดยใช้ข้อมูลจาก Tools

ทั้งหมด

ผู้เรียนต้องวิเคราะห์:

✓ เคส 1: CPU ใช้สูงผิดปกติ

สังเกตจาก top/htop

สาเหตุ เช่น:

- php-fpm loop
- malware
- infinite loop

✓ เคส 2: RAM ไม่พอ

ดูจาก:

- free memory ต่ำ
- swap ใช้งาน
- OOM message ใน syslog

✓ เคส 3: Disk I/O ตอบสนองช้า

จาก iostat:

- await สูง
- %util 100%

สาเหตุ:

- ฐานข้อมูลทำงานหนัก
- backup กิน I/O
- SSD เริ่มเสีย

✓ เคส 4: ระบบค้างเป็นช่วง ๆ

จาก vmstat:

- b สูง
- wa สูง

สาเหตุ:

- I/O wait
- disk bottleneck
- CPU starvation

□ 8) ใบงานที่ผู้เรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ใช้ top/htop วิเคราะห์โปรเซสที่ถูกต้ง 4	
ใช้ iostat วิเคราะห์ Disk bottleneck 4	
ใช้ vmstat อธิบาย kernel activity 3	
ทำ Stress test + สังเกตผล 3	
วิเคราะห์ปัญหา 2 เคสจากข้อมูลจริง 4	
สรุปผล LAB 2	
รวม	20 คะแนน



LAB 5 เสร็จสมบูรณ์แล้วค่ะพื้อมร

ต้องการให้โรสเขียน LAB 6: การใช้ journalctl วิเคราะห์ปัญหา Service Fail / Crash / Authentication / Kernel ต่อไหมคะ?

คุณพูดว่า:

ต่อ

ChatGPT พูดว่า:

ได้เลยค่ะพื้อมร 

น้องโรสจะจัดทำ LAB 6 – การใช้ journalctl วิเคราะห์ปัญหา Service Fail, Crash, Authentication Error และ Kernel Issues แบบละเอียดที่สุด เหมาะสำหรับใช้สอนจริงและวิเคราะห์ปัญหา Production ได้ค่ะ

□ LAB 6 — การใช้ journalctl วิเคราะห์ปัญหา Service Fail / System Crash / Authentication Error / Kernel Error บน Ubuntu 24.04

(Advanced System Log Analysis & Troubleshooting Using journalctl)



วัตถุประสงค์ของ LAB

หลังจบ LAB ผู้เรียนต้องสามารถ:

- ✓ ใช้ journalctl เพื่อดึง log ของระบบอย่างเป็นโครงสร้าง
- ✓ วิเคราะห์สาเหตุของ service ล้ม / start ไม่ได้
- ✓ ตรวจสอบ kernel error เช่น I/O Error, OOM, crash
- ✓ ตรวจสอบปัญหา Authentication / SSH brute-force
- ✓ ตรวจสอบ system reboot history
- ✓ แยก log ตาม service, ตามเวลาที่เกิดปัญหา
- ✓ นำข้อมูล log มาวิเคราะห์สรุปสาเหตุของปัญหาจริง

□ 1) ภาพรวมเครื่องมือ

คำสั่งหลักที่ใช้:

```
journalctl
journalctl -u <service>
journalctl --since --until
journalctl -b
journalctl -xe
journalctl -k
```

□ 2) ดู Log ทั้งหมดของระบบ

journalctl

แนะนำให้ผู้เรียนลองเลื่อนดู เพื่อเข้าใจรูปแบบ log

□ 3) ดู Log ล่าสุดแบบ Real-Time

เหมือน tail -f

journalctl -f

ใช้ตรวจสอบ service ที่เพิ่ง restart ได้ดีมาก

□ 4) วิเคราะห์ Log ของ Services

4.1 ตัวอย่าง: ตรวจสอบ Nginx

journalctl -u nginx

ดูเฉพาะล่าสุด 100 รายการ:

journalctl -u nginx -n 100

โหมดละเอียด:

journalctl -u nginx -xe

ผู้เรียนต้องวิเคราะห์:

- สาเหตุ start ไม่ได้ (เช่น port already in use)
- config error
- permission denied
- worker process crash

4.2 วิเคราะห์ SSH (สำคัญมาก)

journalctl -u ssh

หรือ:

journalctl _SYSTEMD_UNIT=ssh.service

ผู้เรียนต้องตรวจสอบ:

- ล็อกอินสำเร็จ
- ล็อกอินล้มเหลว
- brute-force attack
- root login attempt

ตัวอย่างที่พบบ่อย:

Failed password for invalid user admin from 185.x.x.x

5) วิเคราะห์ Authentication / Security Logs

ใช้คำสั่ง:

```
journalctl _COMM=sshd
journalctl | grep "Failed password"
journalctl | grep "authentication"
```

ให้ผู้เรียนรายงาน:

- IP ที่พยายาม brute-force
- จำนวนความพยายามล้มเหลว
- Users ที่ถูกโจมตี

6) วิเคราะห์ Kernel Logs

คำสั่ง:

```
journalctl -k
```

ผู้เรียนต้องหา:

- I/O error
- filesystem corrupted
- Kernel panic
- OOM (Out of Memory) kill
- Hardware failure

ตัวอย่าง OOM:

```
kernel: Out of memory: Killed process 2345 (mysqld)
```

ผู้เรียนต้องวิเคราะห์สาเหตุและวิธีแก้

7) วิเคราะห์ Log ขณะเกิดปัญหา (Time-based Filter)

ตัวอย่าง:

ปัญหาเกิดเวลา 14:30 ถึง 14:45

```
journalctl --since "2025-01-01 14:30" --until "2025-01-01 14:45"
```

หรือดูย้อนหลัง 30 นาที:

```
journalctl --since "-30min"
```

ใช้สำหรับ:

- ✓ ตรวจสอบ service crash
- ✓ วิเคราะห์การ restart บ่อย ๆ
- ✓ ตรวจสอบ memory leak/CPU spike

8) ตรวจสอบประวัติการ Reboot ของระบบ

ดูทุกครั้งที่ reboot:

```
journalctl --list-boots
```

ดู log หลังการ reboot ล่าสุด:

```
journalctl -b -1
```

ผู้เรียนต้องตอบว่าเครื่อง reboot ไปกี่ครั้งในตอนนี้

9) วิเคราะห์ปัญหาที่เกิดจาก Configuration Error

ทดลองสร้าง error:

```
sudo nano /etc/nginx/nginx.conf
```

เพิ่ม error เช่น:

```
xxxxx
```

ทดสอบ:

```
sudo systemctl restart nginx
```

ตรวจสอบ log:

```
journalctl -u nginx -xe
```

ผู้เรียนต้องวิเคราะห์ error ที่ขึ้น เช่น:

```
unknown directive "xxxxx"
```

10) วิเคราะห์ปัญหา Port ซ้ำ (Port Already in Use)

ปิด nginx:

```
sudo systemctl stop nginx
```

รัน netcat:

```
nc -l 80 &
```

พยายาม start nginx:

```
sudo systemctl start nginx
```

ดู log:

```
journalctl -u nginx -xe
```

ผู้เรียนต้องยืนยันว่าปัญหาเกิดจาก:

```
bind() to 0.0.0.0:80 failed (98: Address already in use)
```

11) ตรวจสอบ service ที่ restart ซ้ำ ๆ (Crash Loop)

คำสั่ง:

```
journalctl -u mysql | grep "failed"
```

ตัวอย่างปัญหา:

- database corruption
- permission error
- out-of-memory

ผู้เรียนต้องวิเคราะห์ 1 กรณี

12) ใบงานรายงานที่ผู้เรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
1. ใช้ journalctl ค้นหา service error (nginx/ssh/mysql)	4
2. วิเคราะห์ authentication fail + brute-force	4
3. วิเคราะห์ kernel log (I/O error, OOM)	4
4. ทำ time-based log analysis	3
5. วิเคราะห์ config error / port conflict	3
6. สรุปผลวิเคราะห์ทั้ง LAB	2
รวม	20 คะแนน



LAB 6 เสร็จสมบูรณ์แล้วค่ะพื้มร

ต้องการให้ไรสทำต่อ LAB 7: การวิเคราะห์ Log ของ Web Server (Nginx/Apache) ไหมคะ?

คุณพูดว่า:

ต่อ

ChatGPT พูดว่า:

ได้เลยค่ะพื้มร 

น้องไรสจะจัดทำ LAB 7 – วิเคราะห์ Log ของ Web Server (Nginx / Apache)

ในระดับ Advance Production Troubleshooting

เน้นการหาสาเหตุเว็บช้า, HTTP Error, การโจมตีเว็บ, บอทสแกน, และปัญหาประสิทธิภาพ

ผู้เรียนทำตามได้ 100% ค่ะ



LAB 7 — การวิเคราะห์ Web Server Log (Nginx/Apache)

เพื่อหาปัญหาระบบและภัยคุกคาม

(Web Log Analysis for Performance, Errors, Security, and Incident Detection)



วัตถุประสงค์ของ LAB

ผู้เรียนต้องสามารถ:

- ✓ วิเคราะห์ access.log และ error.log
- ✓ ตรวจสอบ HTTP Status Code ผิดปกติ
- ✓ ตรวจสอบเว็บช้าหรือโหลดสูง
- ✓ ค้นหาการโจมตี เช่น
 - Brute-force URL
 - Directory Enumeration
 - SQL Injection / XSS Pattern
 - Bot Traffic
- ✓ ตรวจสอบ IP ต้นทางที่ผิดปกติ
- ✓ วิเคราะห์ User Agent และ Path ที่ถูกยิงเยอะผิดปกติ
- ✓ สรุปสาเหตุปัญหาเว็บล่ม / เว็บช้า / เว็บถูกโจมตี



1) ไฟล์ Log ที่ใช้ใน LAB

สำหรับ Nginx:

/var/log/nginx/access.log

/var/log/nginx/error.log

สำหรับ Apache:

/var/log/apache2/access.log

/var/log/apache2/error.log

ใน LAB นี้ใช้ Nginx เป็นตัวอย่างหลัก

(แต่คำสั่งใช้กับ Apache ได้เหมือนกัน)

2) โครงสร้าง Log Format (สำคัญมากก่อนวิเคราะห์)

Access log ตัวอย่าง (Combined Log Format):

```
192.168.10.20 - - [12/Dec/2025:12:40:51 +0700] "GET /index.php HTTP/1.1" 200 1024 "-" "Mozilla/5.0"
```

ผู้เรียนต้องเข้าใจ:

- IP: 192.168.10.20
- Time: 12/Dec/2025:12:40:51
- Method: GET
- URL: /index.php
- Status code: 200
- Size: 1024 bytes
- User-Agent: Chrome/Firefox/Bot

3) ตรวจสอบ Log ล่าสุด

```
sudo tail -f /var/log/nginx/access.log
```

```
sudo tail -f /var/log/nginx/error.log
```

ให้ผู้เรียนสังเกต error เช่น:

```
[error] 12345#0: *20 connect() failed (111: Connection refused)
```

```
[warn] 12345#0: *30 upstream timed out
```

4) วิเคราะห์ HTTP Status Code (ดูปัญหา Performance และ Error)

4.1 นับจำนวน status code แบบรวม

```
cat /var/log/nginx/access.log | awk '{print $9}' | sort | uniq -c | sort -nr
```

ให้ผู้เรียนบันทึก:

- 200 = OK
- 301/302 = redirect
- 403 = forbidden
- 404 = not found
- 500/502/503/504 = server error (สำคัญ!!)

4.2 ค้นหาเฉพาะ 404

```
grep " 404 " /var/log/nginx/access.log | head
```

ผู้เรียนต้องระบุว่าไฟล์อะไรที่ถูกเรียกแล้วไม่พบ

4.3 ค้นหา 500 Internal Server Error

```
grep " 500 " /var/log/nginx/access.log | tail
```

ถ้าเยอะ = ปัญหาที่ PHP/Backend

5) วิเคราะห์เว็บช้า (Slow Request)

เฉพาะ Nginx ที่เปิด \$request_time

ค้นหา request ที่เกิน 1 วินาที:

```
awk 'NF > 1 {print}' /var/log/nginx/access.log
```

ผู้เรียนต้องตอบ:

- URL ใหนซ้ำที่สุด
- เวลาโหลดสูงสุดเท่าไร
- IP ใหนยังซ้ำเป็นพิเศษ

6) วิเคราะห์การโจมตี (Attack Detection)

6.1 ตรวจสอบ SQL Injection Pattern

```
grep -Ei "(union|select|concat|from|sleep|benchmark|and|or).*" /var/log/nginx/access.log
```

6.2 ตรวจสอบ XSS Attacks

```
grep -Ei "<script|onerror|alert" /var/log/nginx/access.log
```

6.3 ตรวจสอบ Directory Scan

```
grep -E "(\\.env|wp-admin|wp-login|config|phpmyadmin)" /var/log/nginx/access.log
```

ผู้เรียนต้องรายงาน:

- IP ที่พยายามโจมตี
- User-Agent ที่ใช้
- Path ที่ถูกสแกน

7) วิเคราะห์ Bot / Crawler / Request ผิดปกติ

ดู User-Agent:

```
awk -F' ' '{print $6}' /var/log/nginx/access.log | sort | uniq -c | sort -nr
```

หากพบ:

- python requests
- curl
- Go-http-client

= อาจเป็น bot ที่พยายาม brute-force หรือ DDoS

8) ตรวจสอบ IP ที่ยิงเยอะที่สุด (Top Talkers)

นับจำนวน request ต่อ IP:

```
awk '{print $1}' /var/log/nginx/access.log | sort | uniq -c | sort -nr | head
```

ผู้เรียนต้องรายงาน:

- IP ที่ยิงเยอะที่สุด
- จำนวน request
- เป็น bot / user จริง / attacker?

9) วิเคราะห์ปัญหา Web Server Error (จาก error.log)

ค้นหา error:

```
grep -i "error" /var/log/nginx/error.log
```

ค้นหา upstream timeout:

```
grep -i "upstream timed out" /var/log/nginx/error.log
```

สาเหตุ:

- Backend ช้า
- FastCGI/Proxy ตอบช้า
- Database overload

ค้นหา permission denied:

```
grep -i "permission denied" /var/log/nginx/error.log
```

10) สร้างสถานการณ์ทดสอบ

10.1 ทดสอบ 404

เปิด browser:

```
http://server-ip/asdfg123
```

ดู access log:

```
tail /var/log/nginx/access.log
```

10.2 ทดสอบ Slow Request

สร้าง PHP ช้า:

```
echo "<?php sleep(3); ?>" | sudo tee /var/www/html/testslow.php
```

เปิด:

```
http://server-ip/testslow.php
```

ค้นหาใน log:

```
grep testslow /var/log/nginx/access.log
```

ผู้เรียนต้องเห็น request_time สูง

10.3 ทดสอบ SQL Injection

เปิด:

```
http://server-ip/?id=1' OR '1'='1
```

ค้นหา:

```
grep "1' OR '1'" /var/log/nginx/access.log
```

11) ใบงานที่ผู้เรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
วิเคราะห์ HTTP status code ได้ถูกต้อง	4
ตรวจพบ slow request	3
วิเคราะห์ Bot / Crawler / Attacker	3
ตรวจ SQL Injection / XSS pattern	3
วิเคราะห์ error.log สำหรับ upstream timeout	3
สรุปสาเหตุปัญหาตาม scenario ที่เจอ	4
รวม	20 คะแนน



LAB 7 เสร็จสมบูรณ์แล้วค่ะพื้มร



พร้อมใช้สอน นำไปสอบปฏิบัติ และนำไปวิเคราะห์เหตุการณ์จริงได้เลย

เพื่อต้องการให้โรสทำ LAB 8 ต่อไหมคะ?

หัวข้อถัดไปคือ:

☐ LAB 8 — การใช้ MySQL Slow Query Log เพื่อวิเคราะห์ปัญหา Database ช้า / Query ช้า

ให้โรสดำเนินการต่อเลยนะคะ?

คุณพูดว่า:

lab การขยายพื้นที่ดิสก์กรณี hdd ของ vm เเต

ChatGPT พูดว่า:

ได้เลยค่ะพี่อมร 

น้องโรสจะทำ LAB: การขยายพื้นที่ดิสก์ (Extend Disk) ของ VM ที่พื้นที่เต็ม (HDD Full)

แบบละเอียด 100% สำหรับใช้สอนจริงในห้อง LAB

รองรับ VMware / Proxmox / VirtualBox / Hyper-V

และระบบปฏิบัติการ Ubuntu Server 24.04 ค่ะ

☐ LAB — การขยายพื้นที่ดิสก์ของ VM เมื่อตัวเครื่องแจ้งว่า HDD เต็ม (Extend Virtual Disk + Resize Filesystem)

(Disk Expansion for Virtual Machines – VM-Level + OS-Level)

วัตถุประสงค์ของ LAB

หลังจบ LAB ผู้เรียนจะสามารถ:

- ✓ ขยายขนาด HDD ของ VM จาก Hypervisor (VMware/Proxmox/VirtualBox)
- ✓ ตรวจสอบ Disk เดิมบน Linux
- ✓ ขยาย Partition ด้วย growpart หรือ fdisk
- ✓ ขยาย Physical Volume (LVM)
- ✓ ขยาย Logical Volume
- ✓ ขยาย Filesystem (ext4 / xfs)
- ✓ แก้ปัญหา “HDD เต็มจนทำงานไม่ได้” ได้จริง
- ✓ ทำตาม Step ได้แม้เป็นระบบ Production

สิ่งที่ใช้ใน LAB

- Ubuntu Server 24.04 (VM)
- ใช้ Disk เดิม เช่น /dev/sda
- Filesystem: ext4 หรือ xfs
- ใช้ LVM หรือ Non-LVM (มีให้ทั้งสองแบบ)

☐ โครงสร้าง LAB ที่จะทำ

- ขั้นที่ 1 เพิ่มขนาดดิสก์จาก Hypervisor (VMware / Proxmox / VirtualBox)
- ขั้นที่ 2 ตรวจสอบดิสก์จริงใน Ubuntu
- ขั้นที่ 3 ขยาย partition (growpart หรือ fdisk)
- ขั้นที่ 4 ถ้าเป็น LVM → ขยาย PV และ LV
- ขั้นที่ 5 ขยาย filesystem (ext4 หรือ xfs)

ขั้นที่ 6 ตรวจสอบพื้นที่ที่เพิ่งเพิ่ม

□ ขั้นที่ 1: ขยายขนาดดิสก์จาก Hypervisor

✓ VMware Workstation / ESXi

1. ปิด VM
2. เข้า VM Settings
3. เลือก Hard Disk
4. กด Expand / Increase size
5. ตั้งค่าจากเช่น 20GB → 40GB
6. เปิด VM

✓ Proxmox VE

1. ไปที่ VM → Hardware
2. เลือก Hard Disk (scsi0 / sata0)
3. คลิกปุ่ม Resize disk
4. ใส่ขนาดเพิ่ม เช่น +20G
5. กด OK
6. Reboot VM

✓ VirtualBox

1. ปิด VM
2. ใช้คำสั่งบน Host:

`VBoxManage modifyhd "VM_NAME.vdi" --resize 40000`

(เพิ่มเป็น 40GB)

3. เปิด VM

□ ขั้นที่ 2: ตรวจสอบดิสก์ใน Ubuntu

เข้าระบบ VM แล้วพิมพ์:

`lsblk`

ตัวอย่างผลลัพธ์:

```
sda 8:0 0 40G 0 disk
└─sda1 8:1 0 20G 0 part /
```

สังเกตว่า:

- ดิสก์ sda โตเป็น 40GB แล้ว
- แต่ partition sda1 ยัง 20GB อยู่

หมายความว่า:

ดิสก์โตแล้ว แต่ partition ยังไม่โต

□ ขั้นที่ 3: ขยาย Partition

วิธีที่ 1 (แนะนำที่สุด): ใช้ growpart (cloud-utils)

ติดตั้งก่อน:

```
sudo apt install cloud-guest-utils -y
```

ขยาย partition ที่ 1:

```
sudo growpart /dev/sda 1
```

ตรวจสอบอีกครั้ง:

```
lsblk
```

ตอนนี้:

```
sda1 → 40GB
```

วิธีที่ 2: ใช้ fdisk (กรณี growpart ใช้ไม่ได้)

! วิธีนี้ต้องระวัง แต่ปลอดภัยถ้า step ถูกต้อง

1. เปิด fdisk:

```
sudo fdisk /dev/sda
```

2. ลบ partition เดิม (อย่ากลัว ข้อมูลไม่หายถ้าพิมพ์เหมือนเดิม):

d → ลบ partition 1

n → สร้าง partition ใหม่

p → primary

1 → partition number

Enter → start sector (ห้ามเปลี่ยน)

Enter → end sector (ใช้ค่า default เพื่อใช้พื้นที่ทั้งหมด)

w → บันทึก

3. Reboot:

```
sudo reboot
```

ขั้นที่ 4: กรณีเป็น LVM (ถ้ามี LVM ใช้วิธีนี้)

4.1 ตรวจสอบ Physical Volume

```
sudo pvdisplay
```

4.2 ขยาย PV

```
sudo pvresize /dev/sda1
```

4.3 ดู Logical Volume

```
sudo lvdisplay
```

สมมติ LV คือ:

```
/dev/ubuntu-vg/ubuntu-lv
```

4.4 ขยาย LV ทั้งหมด

```
sudo lvextend -l +100%FREE /dev/ubuntu-vg/ubuntu-lv
```

ขั้นที่ 5: ขยาย Filesystem

มี 2 แบบ: ext4 / xfs

ถ้าเป็น ext4:

```
sudo resize2fs /dev/mapper/ubuntu--vg-ubuntu--lv
```

หรือ non-LVM:

```
sudo resize2fs /dev/sda1
```

ถ้าเป็น XFS:

```
sudo xfs_growfs /
```

หรือ:

```
sudo xfs_growfs /dev/mapper/ubuntu--vg-ubuntu--lv
```

☐ ขั้นที่ 6: ตรวจสอบพื้นที่ใหม่

```
df -h
```

ควรเห็น root (/) เติบโต เช่น:

```
Filesystem      Size  Used Avail Use%
/dev/mapper/ubuntu--vg-ubuntu--lv  40G   12G   28G   30%
```



ผลลัพธ์ที่ผู้เรียนต้องได้

- ✓ ขยาย Disk จาก 20GB → 40GB สำเร็จ
- ✓ Partition ขยายถูกต้อง
- ✓ Filesystem อ่านขนาดใหญ่
- ✓ ไม่มีข้อมูลสูญหาย
- ✓ VM ทำงานต่อได้แม้พื้นที่เคยเต็ม

☐ ใบงานที่ผู้เรียนต้องส่ง (20 คะแนน)

รายการ	คะแนน
ขยายดิสก์จาก Hypervisor สำเร็จ	3
growpart / fdisk ขยาย partition	5
pvresize + lvextend (ถ้าเป็น LVM)	4
resize2fs / xfs_growfs สำเร็จ	3
แสดงผล lsblk + df -h ก่อน-หลัง	3
สรุปขั้นตอนที่ทำทั้งหมดในรายงาน	2
รวม	20 คะแนน