

README.md — DietPi CTF Training Environment

```
# 💀 DietPi CTF Training Environment
```

A lightweight Capture The Flag (CTF) environment built on **DietPi (Debian minimal)**, designed for training students, classroom labs, and SMK competitions.

This project includes:

- Automatically generated flags based on team name
- Multiple vulnerable services (FTP/HTTP/SMB/SSH)
- Two intentionally vulnerable web apps (Python + Node.js)
- Reset mechanism to restore machine before each contest
- Customizable CTF-friendly Apache landing page

```
## Project Structure
```

```
/usr/local/bin/
```

```
└── 01-input-kelompok.sh  
└── 02-generate-flags.sh  
└── 03-checker.sh  
└── 04-reset.sh
```

```
/etc/profile.d/
```

```
└── ctf_auto.sh
```

```
/etc/systemd/system/
```

```
└── ctf-reset.service
```

```
/opt/app1/ # Python Flask vulnerable app
```

```
└── app.py  
└── flag.txt
```

```
/opt/app2/ # Node.js Express vulnerable app
```

```
└── server.js  
└── flag.txt
```

```
/var/www/html/ # Apache landing page (custom CTF theme)
```

```
└── index.html
```

```
/srv/ftp/ # FTP flag storage
```

```
/srv/samba/ # SMB flag storage
```

```
/home/ctf/ # SSH user flag
```

```
## Features

### ✓ **Automatic Team Name Input**
```

When machine boots, root sees:

- Banner
- Team name prompt
- Flags regenerated based on team name

Example format:

```
smkctf{teamname_fl4g_01_easy}
smkctf{teamname_fl4g_02_medium}
```

Some flags are Base64-encoded for extra challenge.

```
## ✓ **Vulnerable Services**
```

```
### 1. **FTP (vsftpd) - Anonymous Enabled**
```

- Path: `/srv/ftp/flag.txt`
- Port: `21`
- Vulnerabilities:
 - Anonymous login
 - Directory listing
 - Weak configuration

```
### 2. **HTTP (Apache2) - Directory Listing / Custom Landing**
```

- Path: `/var/www/html/flag.txt`
- Port: `80`
- Vulnerabilities:
 - Directory listing
 - LFI hint in landing page
 - Hidden clues in HTML comments

```
### 3. **Samba SMB Shares**
```

- Paths:

- `/srv/samba/share1/flag.txt`
- `/srv/samba/share2/flag.txt`

- Port: `445`

- Vulnerabilities:

- Anonymous SMB access
- Misconfigured share visibility

```
### 4. **SSH (Low-Privilege User)**
```

- User: `ctf:ctf123`

- Path: `/home/ctf/flag.txt`

- Vulnerabilities:

- Weak credentials
- Privilege escalation target

```
### 5. **App1 – Python Flask Vulnerable App (Port 8080)**
```

Path: `/opt/app1/app.py`

Run via pipx:

```
/root/.local/share/pipx/venvs/flask/bin/python /opt/app1/app.py
```

Vulnerabilities:

- Local File Inclusion (LFI)
- Direct file access: `/flag`
- Misconfigured read endpoint: `/read?file=...`

```
### 6. **App2 – Node.js Express Vulnerable App (Port 9090)**

Path: `/opt/app2/server.js`

Run via Node:

node /opt/app2/server.js

Vulnerabilities:

- LFI: `/read?file=..` 

- Command Injection: `/ping?host=8.8.8.8;cat flag.txt` 

- Direct flag access: `/flag` 

--- 

## Auto-Reset on Shutdown/Reboot

Service: `/etc/systemd/system/ctf-reset.service` 

Before shutdown:

- All flags removed

- Team name cleared

- Environment restored to default

Flags regenerate automatically on next boot.

--- 

## 🛠 Installation (Manual or Scripted)

### 1. Install vulnerable services:

apt install vsftpd apache2 samba nodejs npm pipx -y
pipx install flask

### 2. Copy all project scripts to:

/usr/local/bin/

### 3. Copy systemd service:

/etc/systemd/system/ctf-reset.service
systemctl enable ctf-reset.service

### 4. Copy web apps:

/opt/app1/
/opt/app2/
```

```
### 5. Copy Apache index:  
/var/www/index.html  
---  
## Verification  
Run checker:  
/usr/local/bin/03-checker.sh  
It verifies:  
- All flags exist  
- All folders readable  
- All services running  
- App1 & App2 ports open  
---  
## 🛡 Accessing From Kali Linux  
### Enumeration  
nmap -A -p-  
enum4linux  
ftp  
smbclient //share1 -N  
gobuster dir -u http:// -w  
### Exploiting App1  
http://:8080/flag  
http://:8080/read?file=../../etc/passwd  
### Exploiting App2  
http://:9090/read?file=flag.txt  
http://:9090/ping?host=8.8.8.8;cat /opt/app2/flag.txt  
---
```

📚 Educational Use

Designed for:

- SMK labs
- Cybersecurity classroom modules
- OSINT & ethical hacking training
- Internal school competitions

Not intended for production use.

Contributing

Pull requests are welcome!

Suggestions for new challenges or vulnerabilities are appreciated.

License

MIT License - Free for teaching, training, and competitions.