Penetration Testing Report - Dexter Server

# Overview

This report details the penetration testing activities performed on the Dexter server. The goal was to identify vulnerabilities in the network, discover active services, and attempt to access the FTP server on the subnet.

# Steps Taken

## Step 1: Identifying the Subnet

To determine the local subnet configuration, I executed the following command:  
```bash  
ip a  
```  
This provided information on the network interfaces and the assigned IP addresses, helping to identify the subnet.

## Step 2: Network Enumeration

Using Nmap, I scanned the entire subnet for active hosts, open ports, and running services:  
```bash  
nmap -sT -sV -O 192.168.2.0/24  
```  
Results showed that the FTP server (192.168.2.8) was active and running pyftpdlib 1.5.0.

## Step 3: Attempting FTP Access

After discovering the FTP server, I attempted to gain access:  
• Tried brute-forcing credentials using Hydra, but it was unsuccessful.  
```bash  
hydra -l admin -P rockyou.txt ftp://192.168.2.8  
```  
• Searched for vulnerabilities in pyftpdlib 1.5.0 using SearchSploit:  
```bash  
searchsploit pyftpdlib 1.5.0  
```  
However, no known exploits were found.

## Step 4: ARP Spoofing for Credential Interception

Since FTP is a cleartext protocol, I attempted ARP spoofing to intercept authentication traffic:  
```bash  
arpspoof -i eth0 -t 192.168.2.8 -r  
```  
• Monitored network traffic with tcpdump to see if any credentials were transmitted:  
```bash  
sudo tcpdump -i eth0 host 192.168.2.8 and port 21  
```  
• Observed that an external server (192.168.3.1) was attempting to communicate with the FTP server but was not receiving responses.

## Step 5: Diagnosing Network Connectivity Issues

Upon verifying that the FTP server was functional via Nmap, I determined that the TCP handshake between the external server and the FTP server was failing. This suggested that IP forwarding was disabled.  
Attempts to enable IP forwarding were blocked due to container restrictions:  
```bash  
echo 1 > /proc/sys/net/ipv4/ip\_forward  
-bash: /proc/sys/net/ipv4/ip\_forward: Read-only file system  
```

## Step 6: Implementing Custom ARP Spoofing with IP Forwarding

To bypass system restrictions, I developed a custom ARP spoofing implementation that manually forwards packets without relying on OS-level IP forwarding. This method successfully allowed me to intercept FTP login credentials.

## Step 7: Gaining Access to the FTP Server

With the captured username and password, I logged into the FTP server and located the flag2.txt file. The file was then transferred to the Dexter server for review:  
```bash  
ftp 192.168.2.8  
get flag2.txt  
```