Man-in-the-Middle (MITM) Attacks

Principles, Examples and Defenses

Ahmed Dinari

Project: Network Protocols - Attacks and Defenses

December 8, 2025

Motivation: What is MITM?

Normal Communication:

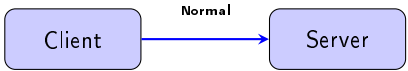
- Client ↔ Server
- Direct connection
- Assumed confidentiality

Attacker's Goal:

- Position between the two
- Without being detected
- Intercept and manipulate

Enables:

- Eavesdropping
- Data modification
- Credential theft
- Redirection



Principle

If Alice thinks she's talking to Bob, but everything goes through Mallory → **Man-in-the-Middle**

What is a Man-in-the-Middle Attack?

Definition

An **active** attack on the confidentiality and integrity of communications

The Attacker:

- Intercepts traffic
- Reads data
- Sometimes modifies content

Common Types:

- Passive MITM
  - Listening / sniffing
  - No modification
- Active MITM
  - Modification
  - Redirection
  - Content injection

Combines with:

- ARP spoofing
- DNS spoofing
- Rogue Wi-Fi
- SSL stripping
- Certificate spoofing

MITM in the Network Model

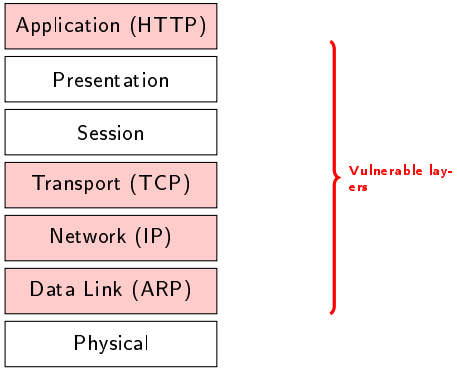
Can target multiple layers:

- **Link Layer (Layer 2)**
  - **ARP spoofing on LAN**
  - **Malicious switch**
- **Network/Transport (L3-4)**
  - **Route hijacking (BGP)**
  - **TCP hijacking**
- **Application (Layer 7)**
  - Unencrypted HTTP
  - Fake Wi-Fi portals
  - Fraudulent TLS certificates

In Practice

Modern MITM targets:

- Unsecured local networks
- Public Wi-Fi
- Unencrypted protocols



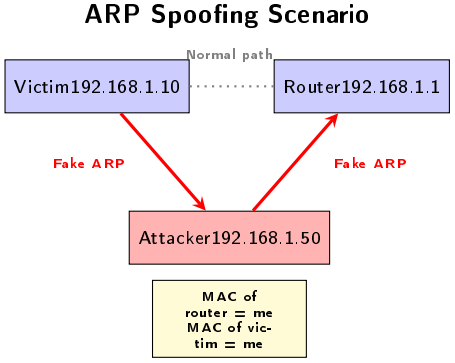
Example 1: MITM on Local Network (ARP)

How ARP Works:

- ARP = Address Resolution Protocol
- Maps IP ↔ MAC address
- Used on local networks (LAN)
- ⚠ No authentication!

The Attack:

1. Attacker sends **fake ARP replies**
2. Claims to be the router:
   "Router's IP has MY MAC"
3. Claims to be the victim:
   "Victim's IP has MY MAC"
4. **Result:** all traffic passes through attacker



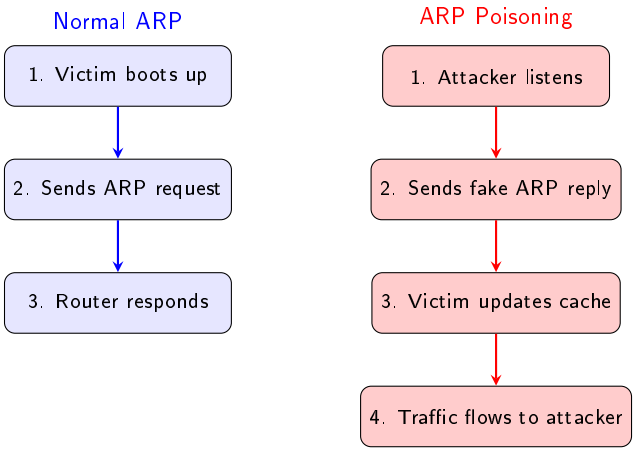
Technical Detail

ARP is stateless - accepts unsolicited replies (gratuitous ARP). Cache poisoning persists until timeout (typically 2-20 min).

Note

No complex tools needed - protocol vulnerable by design

ARP Attack: Detailed Flow



Defense Against ARP-based MITM

Possible Consequences:

- Traffic reading
  - Passwords in cleartext
  - HTTP cookies
  - Sensitive data
- Packet modification
  - Redirect to fake sites
  - Inject malicious code
  - Alter data
- Session hijacking

Concrete Defenses:

1. End-to-end encryption

- HTTPS, SSH, VPN
- Makes traffic unreadable

2. Network protections

- Static ARP entries (critical)
- **Dynamic ARP Inspection** (DAI)
- Port Security on switches
- Network segmentation / VLANs

3. Monitoring

- ARP anomaly detection
- IDS/IPS (Intrusion Detection)
- Logs and alerts

Best Practice

Use encrypted protocols + network hardening

MITM on HTTP / HTTPS (SSL Stripping)

Normal Operation:

- **HTTP** = cleartext traffic
  - No encryption
  - No authentication
- **HTTPS** = HTTP + TLS
  - Encryption
  - Server authentication
  - Data integrity

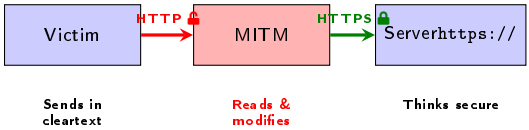
Attack: SSL Stripping

1. Victim visits `http://site.com`
2. MITM intercepts
3. MITM ↔ Server: HTTPS 🔒
4. MITM ↔ Victim: HTTP 🔓
5. Attacker reads/modifies everything!

Key Insight

Exploits initial HTTP connection before HTTPS redirect

SSL Stripping Scenario



Problem

Victim sees normal HTTP connection, no attack indication

Visual Clue

Missing padlock 🔒 in address bar

How SSL/TLS Protects Against MITM

TLS Handshake (Simplified):

1. Client Hello (supported ciphers)
2. Server Hello + Certificate
3. Client verifies certificate
4. Key exchange (encrypted)
5. Encrypted communication

MITM Attempt

If attacker presents fake certificate, browser shows warning!

Certificate Validation:

- **Domain name** match
- **Certificate Authority** (CA) trust
- **Expiration** date
- **Signature** verification
- **Certificate chain** integrity

Why it stops MITM:

- Attacker can't forge valid certificate
- CA won't sign attacker's certificate for victim's domain
- Private key stays on legitimate server
- Browser validates entire chain

Technical Note

TLS 1.3 provides forward secrecy (PFS) - even if server key compromised later, past sessions remain secure

Defense Against MITM on HTTP/HTTPS

1. Force HTTPS

- HTTP → HTTPS redirects server-side
- **HSTS** (HTTP Strict Transport Security)
  - Forces browser to use HTTPS
  - Prevents downgrade to HTTP
- HSTS preload lists in browsers

2. Verify TLS Certificates

- Correct domain name
- Valid Certificate Authority
- Valid dates
- ⚠ NEVER ignore warnings!

3. Client Side

- Up-to-date browsers
- Check padlock 🔒 before entering sensitive data
- Extensions: HTTPS Everywhere
- Avoid public Wi-Fi for sensitive data

4. For Applications

- **Certificate pinning**
  - App only accepts specific certificates
  - Prevents fraudulent certificates
- Strict certificate validation
- No HTTP fallback

Golden Rule

✔ **Always check HTTPS before entering**: Passwords, banking info, personal data

MITM on Public Wi-Fi

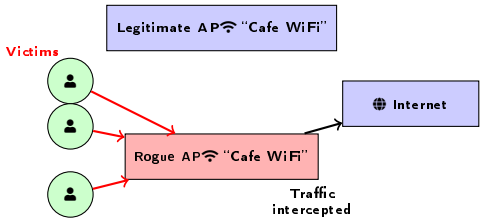
Typical Attack: Rogue Access Point

Concept:

1. Attacker creates fake Wi-Fi AP
2. Same name (SSID) as legitimate network
   - "Starbucks WiFi"
   - "FreeWiFi"
   - "Hotel_Guest"
3. Stronger signal than real AP
4. Victims auto-connect

Attacker can:

- Observe all traffic
- Modify requests/responses
- Create fake login portals (phishing)
- Force downgrades (weak protocols)



Danger

Users see no difference - same name, easy connection

Real Scenario

Café, airport, hotel - anywhere with public Wi-Fi

Wi-Fi Security: Evolution

Protocol	Year	Encryption	Security	Status
WEP	1997	RC4 (40-104 bit)	Very Weak	Deprecated
WPA	2003	TKIP	Weak	Legacy
WPA2	2004	AES-CCMP	Strong	Current
WPA3	2018	AES-GCMP	Very Strong	Recommended

WPA2/WPA3 Features:

- **WPA2-Personal (PSK)**
  - Pre-shared key
  - Good for home
- **WPA2-Enterprise (802.1X)**
  - RADIUS authentication
  - Best for organizations

WPA3 Improvements:

- **SAE** authentication
  - Resistant to offline attacks
- **Forward secrecy**
- **192-bit security**

Recommendation

Use WPA3 where available, minimum WPA2-Enterprise for businesses

Defense Summary & Best Practices

Wi-Fi Defenses:

- **Use VPN**
  - Encrypts all traffic
  - Even on public Wi-Fi
- **Prefer secure Wi-Fi**
  - WPA2/WPA3-Enterprise
  - Avoid WEP, WPA
- **Device configuration**
  - Disable auto-connect
  - Verify network name

Overall Protection:

MITM Exploits:

- Lack of encryption
- Implicit trust
- User negligence

Key Measures

1. Encryption: HTTPS, SSH, TLS, VPN
2. Authentication: Valid certificates
3. Network hardening: Secure switches, WPA3
4. Awareness: User training

Conclusion

Defense in depth: Encryption + Authentication + Hardening + Awareness

Live Demo (Windows Lab)

Goal: show MITM in a safe, isolated lab on Windows (no real network impact)

Lab Setup

- Host: Windows + WSL2 (Ubuntu) + Wireshark/Npcap
- Two VMs on Host-Only / Private vSwitch
- IP plan example: `192.168.56.1` (gateway VM), `.101` (victim), `.200` (attacker)
- Do this only in the lab network

Traffic Capture

- Start Wireshark on host/attacker interface
- Filter: `http` (for cleartext) or `tcp.port==8080` (if using mitmproxy)

Attack Steps (WSL2)

1. `sudo apt update && sudo apt install dsniff mitmproxy`
2. Enable forwarding: `sudo sysctl -w net.ipv4.ip_forward=1`
3. ARP poison both ends:
   ```
   sudo arpspoof -i eth0 -t 192.168.56.101 192.168.56.1
   sudo arpspoof -i eth0 -t 192.168.56.1 192.168.56.101
   ```
4. (HTTP sniff) Browse HTTP from victim; view in Wireshark
5. (SSL strip) Redirect HTTP to mitmproxy:
   ```
   sudo iptables -t nat -A PREROUTING -p tcp -dport 80 -j REDIRECT -to-port 8080
   mitmproxy -mode transparent -showhost
   ```

Safety

- Keep lab isolated; disable when done: `sudo iptables -t nat -F`
- HTTPS sites will warn on bad certs; demo that warning = protection

Real-World Example: Public Wi-Fi Attack (2015)

Scenario: Airport Wi-Fi MITM

What Happened:

1. Attacker set up rogue AP at major airport
2. SSID: "Airport_Free_WiFi" (looked legitimate)
3. 200+ users connected in 3 hours
4. SSL stripping used on banking sites
5. Credentials harvested from HTTP sites

Attack Chain:

- **Layer 2**: Rogue AP
- **Layer 3-4**: Traffic routing through attacker
- **Layer 7**: SSL stripping, DNS spoofing

Why It Succeeded:

- Users trusted "free" Wi-Fi
- No VPN usage
- Many sites still used HTTP
- Users ignored certificate warnings
- Auto-connect enabled

What Could Have Prevented It:

- ✔ VPN mandatory for public Wi-Fi
- ✔ HSTS on all websites
- ✔ User awareness training
- ✔ Disable auto-connect
- ✔ Verify network legitimacy

Lesson Learned

Defense in depth: Multiple layers of protection needed. One missing layer = vulnerability.

Thank You!

Questions?

ahmed.dinari@polytechnicien.tn

github.com/amedo007-poly

"Security is not a product, but a process"
- Bruce Schneier