Deep Dive into ARP Spoofing and Detection Mechanisms

What is ARP?

The Address Resolution Protocol (ARP) is used to map IP addresses to MAC (Media Access Control) addresses in a local area network. When a device wants to communicate with another device on the same network, it sends a broadcast ARP request asking, "Who has this IP?" The device with the matching IP replies with its MAC address.

Example:

- Host A sends: "Who has 192.168.0.1? Tell 192.168.0.5"
- Gateway (192.168.0.1) replies: "192.168.0.1 is at 00:11:22:33:44:55"

What is ARP Spoofing?

ARP spoofing, also known as ARP poisoning, is an attack where a malicious actor sends fake ARP replies to associate their MAC address with the IP address of another node on the network (commonly the default gateway or another host). This causes traffic intended for the legitimate device to be redirected to the attacker.

By poisoning the ARP cache of other machines:

- The attacker can perform Man-in-the-Middle (MitM) attacks.
- They can intercept, modify, or drop packets.
- In some cases, denial-of-service (DoS) attacks can be carried out.

How It Works (Step-by-Step)

1. **Target Discovery**: The attacker scans the network to find active IP/MAC pairs (using `arp-scan`, `nmap`, etc.).

2. **Poisoning ARP Cache**: The attacker sends forged ARP replies to both:
3. The target: mapping the gateway IP to the attacker's MAC.
4. The gateway: mapping the target's IP to the attacker's MAC.
5. **Traffic Interception**: Now the attacker sits in the middle of the communication flow.
6. **Optional Forwarding**: The attacker can forward packets to maintain network appearance using packet forwarding (`echo 1 > /proc/sys/net/ipv4/ip_forward` in Linux).

Tools Commonly Used for ARP Spoofing

- `arpspoof` (from dsniff)
- `ettercap`
- `Bettercap`
- `mitmproxy` (in some configurations)

Real-World Impacts

- Credential theft (HTTP, FTP, Telnet)
- Session hijacking
- SSL stripping (if combined with downgrade attacks)
- DNS spoofing
- Network disruption

Detection Mechanisms

1. **Static ARP Entries**:
2. Manually configure IP-to-MAC mappings where applicable (limited scalability).
3. Prevents dynamic updates to ARP cache.
4. **ARP Inspection Tools**:
5. Use tools like `arpwatch`, `XArp`, `Wireshark` to monitor ARP traffic for inconsistencies.

6. Alert if a MAC address suddenly maps to multiple IPs or vice versa.
7. **Gratuitous ARP Alerts**:
8. Repeated unsolicited ARP replies could indicate spoofing.
9. Some intrusion detection systems (e.g., Snort) can be configured to detect this.
10. **Dynamic ARP Inspection (DAI)**:
11. Supported on enterprise-grade switches (e.g., Cisco).
12. Validates ARP packets on the basis of a trusted DHCP snooping table.
13. **Host-Based Defense**:
14. Tools like `arpalert`, `arping`, or ARP table monitoring scripts.
15. Detect anomalies in local ARP cache.
16. **Network Segmentation and Isolation**:
17. Reduce attack surface by isolating sensitive systems into VLANs.
18. Use VPNs or encrypted tunnels within the LAN.

Example of Detection with `arp -a`

```
$ arp -a
? (192.168.0.1) at 00:11:22:33:44:55 [ether] on eth0
? (192.168.0.1) at aa:bb:cc:dd:ee:ff [ether] on eth0   # Suspicious du
```

Mitigation Summary

- Enforce static ARP entries where feasible
- Use DAI on managed switches
- Implement network monitoring and alerts
- Prefer encrypted protocols (HTTPS, SSH)
- Avoid insecure legacy protocols on LAN (Telnet, FTP)