ARPGuard Lab Handout

Detecting ARP Spoofing with ARPGuard — Lab Handout
==========

1. Purpose

This lab helps learners observe normal ARP resolution behavior and then recognize an ARP spoofing pattern
(ARP cache poisoning) in a packet capture. Learners will run ARPGuard to see how simple, explainable heuristics translate those observations into alerts.

2. Learning Objectives

By the end of this lab, a learner should be able to:
- Describe what ARP does and where it “lives” in the TCP/IP stack (link/local network boundary).
- Identify a suspicious IP→MAC mapping change in ARP traffic.
- Explain why ARP spoofing enables man-in-the-middle and traffic redirection on a LAN.
- Propose at least two mitigations and explain how they reduce ARP spoof risk.

3. Prerequisites

Required:
- Wireshark (or tshark) for opening PCAP files.
- Python 3.11+ with the project dependencies installed.

Provided in this project:
- pcaps/benign_arp.pcap
- pcaps/arp_spoof_attack.pcap
- code/arpguard_core.py (analyzer) and code/arpguard_lab_tools.py (demo)

4. Safety and Ethics

ARP spoofing is an attack technique. This lab is designed to be completed using provided PCAP files. Learners should not perform ARP spoofing on any network unless the instructor explicitly authorizes a controlled environment.

5. Background (Short)

ARP maps IPv4 addresses to MAC addresses on a local network segment by broadcasting “who-has” requests
and receiving “is-at” replies [1]. Many hosts update their ARP caches based on observed ARP replies, which is
one reason spoofing works: an attacker can inject a forged mapping (e.g., “Gateway IP is at Attacker MAC”), causing the victim to send traffic to the attacker [1].

6. Phase 1 — Observe Benign ARP Behavior (benign_arp.pcap)

6.1 Open the PCAP
- Open pcaps/benign_arp.pcap in Wireshark.
- Apply the display filter: arp

6.2 Identify the Request/Reply Pair
Learners should locate:
- An ARP who-has request (broadcast) asking for a target IP (often the gateway).
- A corresponding ARP is-at reply that provides the target’s MAC address.

6.3 Record the Mapping
Learners should record:
- Sender Protocol Address (SPA / psrc)
- Sender Hardware Address (SHA / hwsrc)
- Target Protocol Address (TPA / pdst)
- Target Hardware Address (THA / hwdst)

Expected observation:
- The gateway IP remains associated with a single, consistent MAC address across the capture.

7. Phase 2 — Observe a Spoofing Pattern (arp_spoof_attack.pcap)

7.1 Open the PCAP and Filter
- Open pcaps/arp_spoof_attack.pcap in Wireshark.
- Apply the display filter: arp
7.2 Look for an IP→MAC Conflict
Learners should identify:
- A legitimate gateway reply that maps the gateway IP to the gateway MAC.
- One or more later replies mapping the same gateway IP to a different MAC.
Expected observation:
- The capture contains at least one mapping change for the same IP address (classic poisoning symptom).
7.3 Interpret the Risk
Learners should discuss:
- If a host updates its ARP cache to the attacker's MAC for the gateway IP, where will the host send traffic?
- How could an attacker forward traffic to remain stealthy (MITM) versus dropping traffic (DoS)?
8. Phase 3 — Run ARPGuard and Interpret Alerts

8.1 Install Dependencies (if needed)
From the project root:
pip install -r requirements.txt
8.2 Run the Demo (recommended)
python code/arpguard_lab_tools.py demo --out-dir pcaps
8.3 Analyze a PCAP Directly
python code/arpguard_core.py pcaps/arp_spoof_attack.pcap --json sanity/attack_results.json
8.4 What Learners Should See
For benign_arp.pcap:
- ARPGuard should report zero or very few alerts.
For arp_spoof_attack.pcap:
- ARPGuard should report at least one IP_MAC_CONFLICT alert indicating the gateway IP mapped to multiple MACs.
9. Reflection Questions (Submit Short Answers)
---
1) What is the key behavioral assumption that makes ARP spoofing possible?
2) In the attack capture, what evidence indicates that the gateway mapping was poisoned?
3) Which ARPGuard alert is the strongest indicator of spoofing and why?
4) Name two mitigations (technical or operational) and briefly explain how each helps.
5) What limitations does ARPGuard have (false positives / false negatives) based on its simple heuristics?
10. References
---
[1] D. C. Plummer, "An Ethernet Address Resolution Protocol," RFC 826, Nov. 1982.