ARPGuard Lab Handout

Detecting ARP spoofing with Wireshark evidence and ARPGuard alerts

1. Purpose

This lab helps learners observe normal ARP resolution behavior and then recognize an ARP spoofing pattern (ARP cache poisoning) in packet captures. Learners will use Wireshark to extract evidence and then run ARPGuard to see how explainable heuristics translate those observations into structured alerts.

2. Learning Objectives

- Describe what ARP does and where it sits in the TCP/IP stack (local delivery boundary between network and link).
- Identify suspicious IP-to-MAC mapping changes in ARP traffic.
- Explain why ARP spoofing can enable man-in-the-middle interception or traffic redirection on a LAN.
- Propose at least two mitigations and explain how each reduces ARP spoofing risk.

3. Prerequisites

Required:

- Wireshark (or tshark) to open and filter PCAP files.
- Python 3.10+ (3.11 recommended) and pip.
- Project dependencies installed via requirements.txt.

Provided with this project:

- pcaps/benign_arp.pcap
- pcaps/arp_spoof_attack.pcap
- code/arpguard_core.py (analyzer) and code/arpguard_lab_tools.py (demo utilities)
- Optional: code/arpguard_web_dashboard.py (local UI)

4. Safety and Ethics

ARP spoofing is an attack technique. This lab is designed to be completed using provided PCAP files. Live spoofing should only occur in an instructor-authorized isolated environment. Unauthorized spoofing on production/campus networks is prohibited.

5. Setup (One-Time)

From the project root directory:

```
python -m venv .venv
source .venv/bin/activate # macOS/Linux
# .venv\Scripts\activate # Windows PowerShell
pip install -r requirements.txt
```

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

6.1 Open the capture and filter

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

6.2 Identify request/reply pairs

Locate an ARP *who-has* request (broadcast) and a corresponding ARP *is-at* reply. Record the fields in the table below for at least one request/reply pair.

Packet	Sender IP (SPA / psrc)	Sender MAC (SHA / hwsrc)	Target IP (TPA / pdst)	Target MAC (THA / hwdst)
who-has request				
is-at reply				

6.3 Expected observation

Across the benign capture, a protected IP (often the gateway) should remain associated with a single, consistent MAC address. Any mapping changes should be explainable (e.g., a legitimate device change) and should not resemble rapid conflicting claims.

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

7.1 Open the capture and filter

- Open pcaps/arp_spoof_attack.pcap in Wireshark.
- Apply the display filter: **arp**.

7.2 Find an IP-to-MAC conflict

Identify a legitimate reply mapping the gateway IP to the gateway MAC, then locate a later reply mapping the same gateway IP to a different MAC. Record at least one conflict below.

Protected IP (e.g., gateway)	Legitimate MAC observed	Conflicting MAC observed	Packet # / timestamp

7.3 Interpret risk

- If a host updates its ARP cache so that the gateway IP points to the attacker MAC, where will the host send frames for off-subnet traffic?
- How can an attacker remain stealthy by forwarding traffic (MITM) versus causing a visible outage by dropping traffic (DoS)?
- Which evidence in the capture supports the interpretation (fields, sequence, repeated claims)?

8. Phase 3 — Run ARPGuard and Interpret Results

8.1 Recommended demo command

```
python code/arpguard_lab_tools.py demo --out-dir pcaps
```

8.2 Analyze PCAPs directly and save JSON (recommended for reproducibility)

If **--json** is omitted, ARPGuard prints formatted JSON to the terminal.

```
python code/arpguard_core.py pcaps/benign_arp.pcap --json
sanity/benign_results.json
python code/arpguard_core.py pcaps/arp_spoof_attack.pcap --json
sanity/attack_results.json
```

8.3 Expected output (important)

- **benign_arp.pcap:** 0 anomaly events is expected if no mapping contradictions are present. The JSON summary still shows counts and observed bindings.
- **arp_spoof_attack.pcap:** at least one anomaly event is expected (typically IP_MAC_CONFLICT) indicating the protected IP mapped to multiple MACs.

9. Evidence to Capture (What to Submit)

Capture evidence that supports the conclusions. Unless the instructor specifies otherwise, submit short, clearly labeled evidence:

- Wireshark screenshot(s) showing the conflict: two ARP replies where the same IP maps to different MACs.
- ARPGuard output summary for both PCAPs (CLI text or JSON excerpt highlighting summary + events).
- Short written answers to the reflection questions below (1–3 sentences each).

10. Reflection Questions (Short Answers)

- What behavioral assumption in ARP makes spoofing possible?

- In the attack capture, what evidence indicates the protected mapping was poisoned?
- Which ARPGuard alert is the strongest indicator of spoofing and why?
- Name two mitigations and briefly explain how each helps (technical or operational).
- What limitations does ARPGuard have (false positives/false negatives) based on its heuristic approach?

11. Grading Guidance (If Used by Instructor/TA)

Component	Points	Criteria (abbreviated)
Benign analysis evidence	4	Correct filter, correct field identification, stable mapping stated.
Attack analysis evidence	6	Correct conflict identified, risk interpreted, evidence cited.
ARPGuard run + interpretation	6	Correct commands, benign=0 anomalies explained, attack alerts interpreted.
Reflection questions	4	Concise and correct reasoning; mitigations are defensible.
Professionalism	2	Clear labels, readable screenshots, consistent formatting.

References

[1] D. C. Plummer, “An Ethernet Address Resolution Protocol,” RFC 826, IETF, Nov. 1982.

[2] S. Cheshire, “IPv4 Address Conflict Detection,” RFC 5227, IETF, July 2008.