IoT Security Project

Evaluating Data Integrity Attacks on IoT Telemetry Systems

Evan C, Ryan C, Kevin H, Nasir J

1865 RIDER UNIVERSITY

Overview of IoT Devices

IoT devices are widely used for remote monitoring and automation

Environmental
- Temperature
- Humidity
- Air Quality
- Light

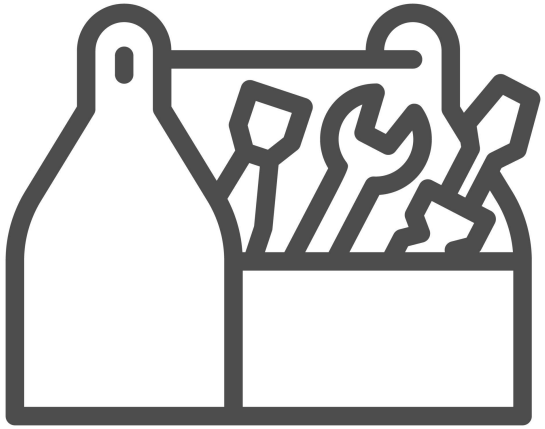
Motion, Position, Proximity
- Proximity
- Motion detection
- Accelerometers
- Gyroscopes
- GPS

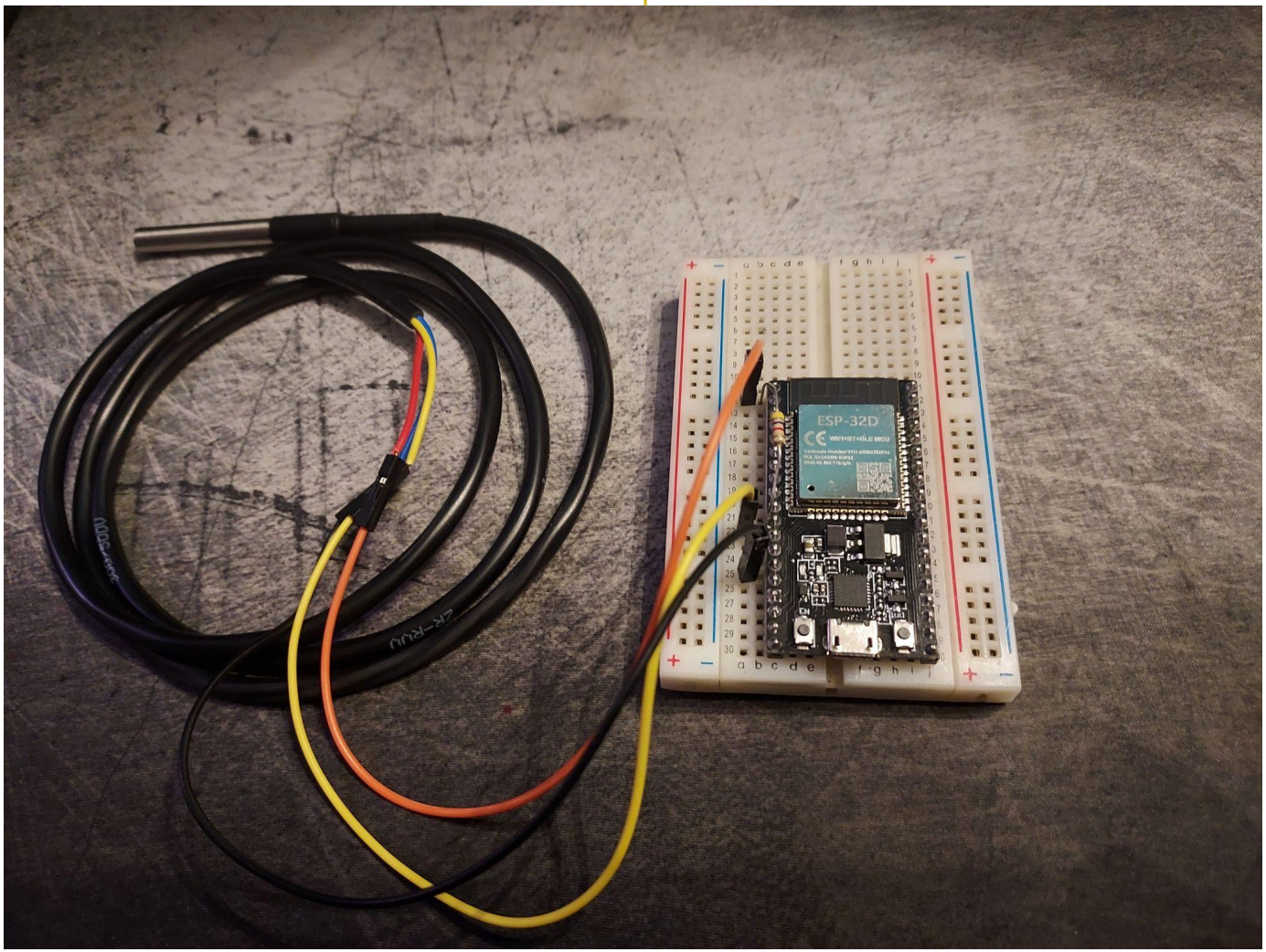
Industrial
- Pressure (liquid/gas)
- Level (liquid/gas)
- Flow (liquid/gas)
- Electric Current

Specialized
- Image
- Sound
- Gas
- Bio Sensors

Project Hardware

- ESP32 Microcontroller (WiFi-enabled SoC)
- DS18B20 Digital Temperature Sensor
- Laptop Server





https://www.adafruit.com/

System Overview

- **Sensor reads temperature**
- **ESP32 processes data**
- **Data sent via HTTP to server**
- **Server stores and displays data**

Network Setup

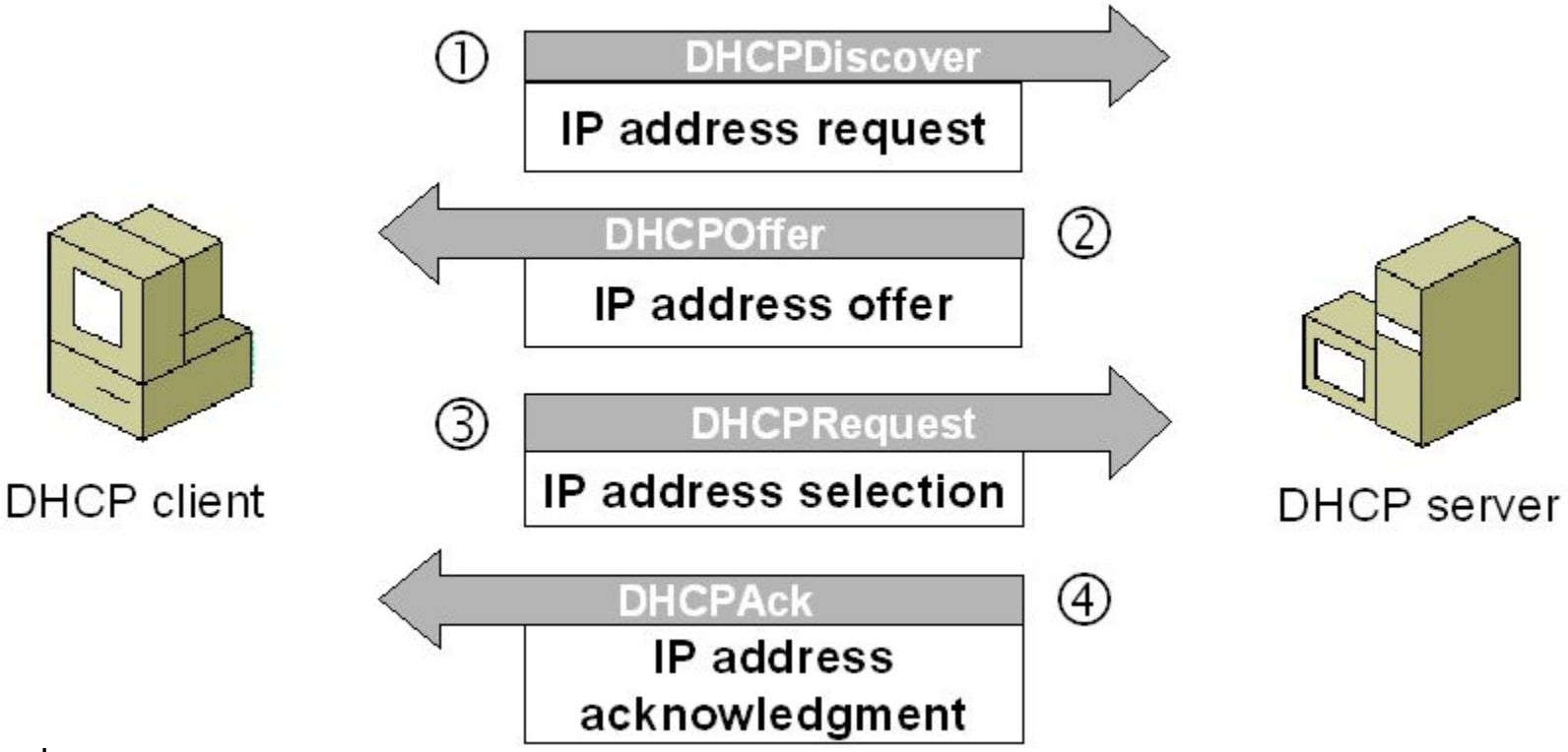
- **All devices connected to TP-Link router**
  - IoT device, server, attacker laptop
- **Simulated public/untrusted network**

DHCP Basics

A DHCP (Dynamic Host Configuration Protocol) Request is the third step in the DHCP DORA (Discover, Offer, Request, Acknowledge) process, where a client formally requests to use the IP address offered by a server. It is broadcast to inform all servers which offer was accepted, allowing others to release reserved addresses.

- **Device connects to network**
- **Requests IP from DHCP server**
- **Provides MAC address**
- **Receives IP address and stores it**

DHCP Basics

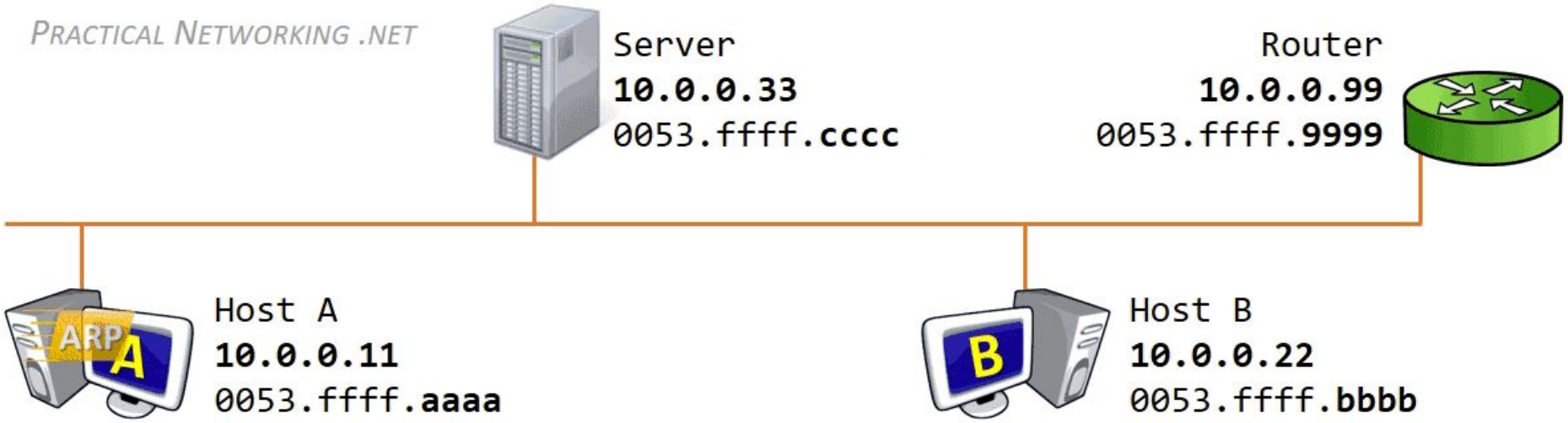

①
DHCPDiscover
IP address request
DHCPOffer
②
IP address offer
③
DHCPRequest
IP address selection
DHCPAck
④
IP address acknowledgment
DHCP client
DHCP server

Device Identity on Network

- IP address = logical identifier
  - ex: Lawrence Rd
- MAC address = physical identifier
  - ex: 2083 (specific address on road)
- Router maps IP to MAC

What is ARP?

- Address Resolution Protocol maps IP to MAC
- Devices broadcast ARP requests
- Target responds with MAC address



ARP Caching

- Devices store IP-to-MAC mappings
- Avoid repeated broadcasts
- Can be exploited if “poisoned”

ARP Cache table

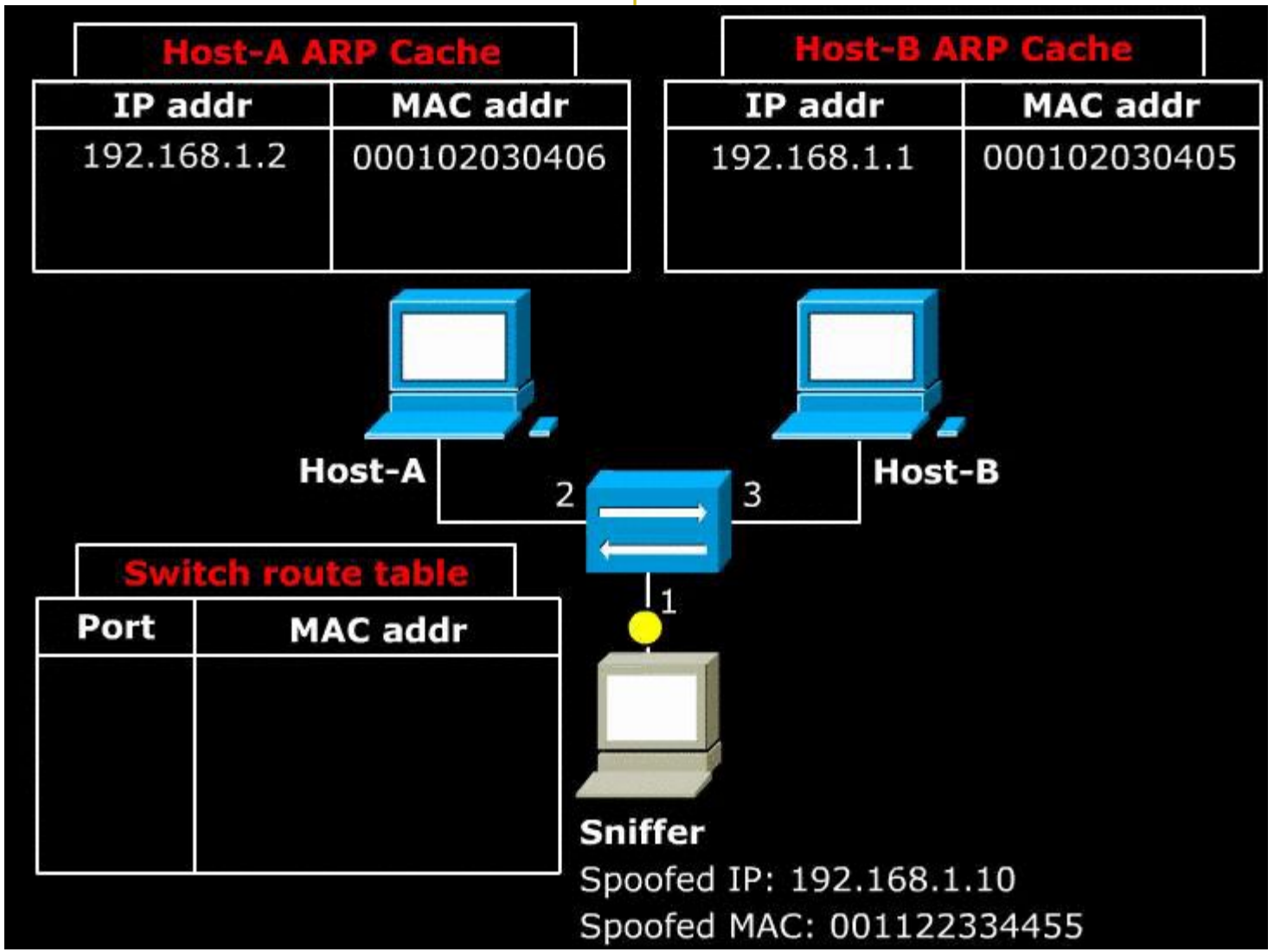
```
Kevin@DESKTOP-M2EPRIG MINGW64 ~/Projects
$ arp -a

Interface: 192.168.1.154 --- 0x11
  Internet Address      Physical Address      Type
  192.168.1.1           3c-bd-c5-40-a0-61     dynamic
  192.168.1.100         f8-a0-97-dd-3b-e9     dynamic
  192.168.1.103         4c-12-65-02-e9-b8     dynamic
  192.168.1.151         0c-43-f9-17-46-5d     dynamic
  192.168.1.162         f8-fc-e1-63-a0-ad     dynamic
  192.168.1.164         48-a6-b8-aa-97-8a     dynamic
  192.168.1.234         84-2b-2b-ac-42-69     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.2.60            01-00-5e-00-02-3c     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

- Every client on the network builds a cache so it can send IP packets directly to the MAC address associated with an IP address.

ARP Spoofing Concept

- Attacker sends fake ARP replies
- Associates victim(s) IP with attacker MAC
- Redirects traffic through attacker


Host-A ARP Cache
IP addr
MAC addr
192.168.1.2
000102030406
Host-B ARP Cache
IP addr
MAC addr
192.168.1.1
000102030405
Host-A
Host-B
2
3
1
Switch route table
Port
MAC addr
Sniffer
Spoofed IP: 192.168.1.10
Spoofed MAC: 001122334455

Why ARP Spoofing Works (In theory)

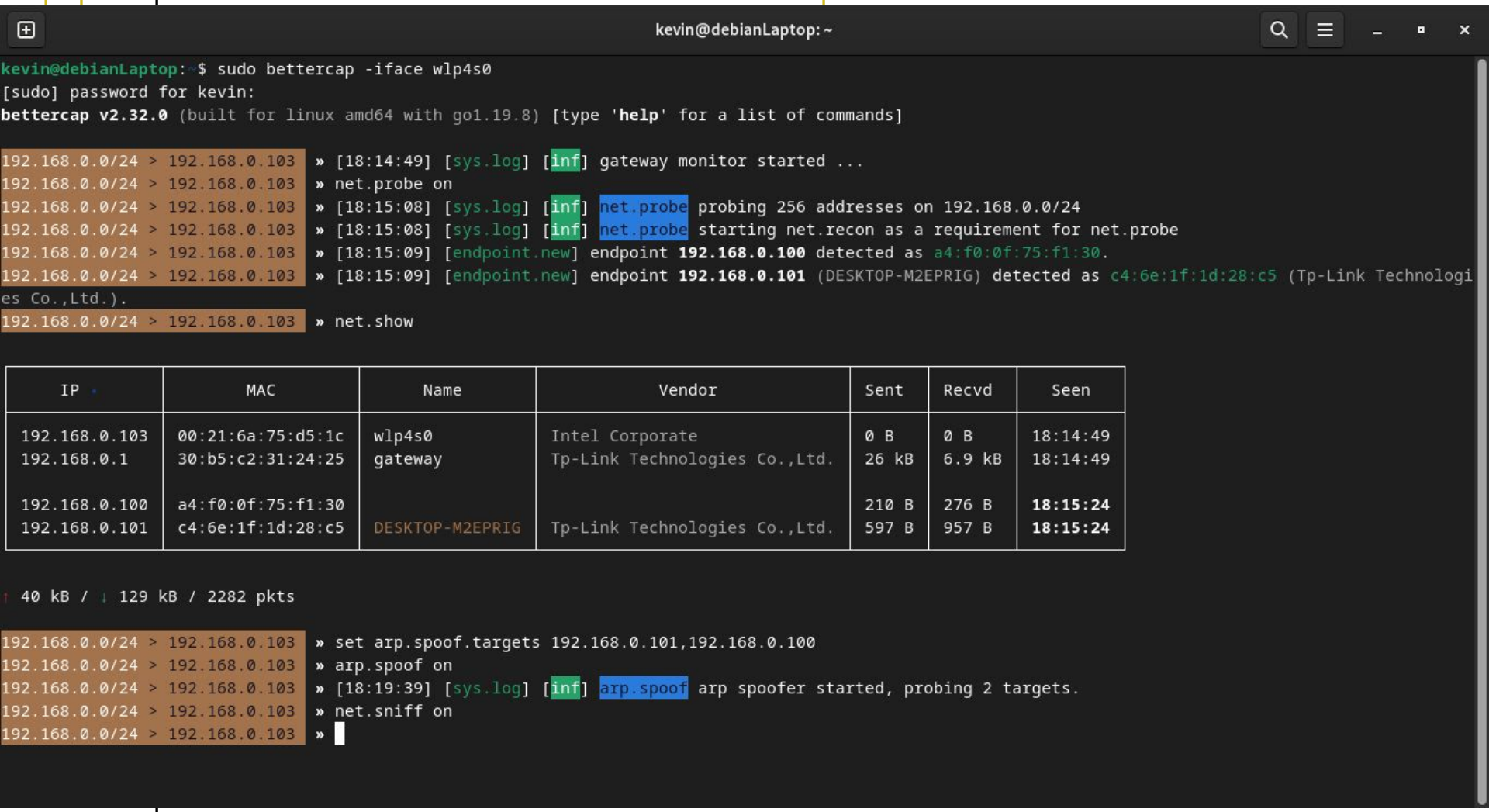
- ARP has no authentication
- Devices trust replies
- Cache updates automatically

Bettercap Tool

- Network attack framework
- Used for MITM attacks
- ARP spoofing and packet interception
  - Crafts raw packets
  - Sends fake ARP replies repeatedly
  - Enables IP forwarding
  - Captures/modifies traffic

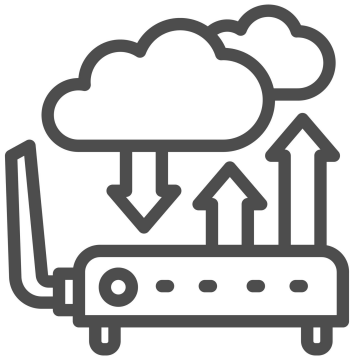
Bettercap Commands

- net.probe on
  - discovers devices
- arp.spoof.targets <IP>
  - tells each ip “use this MAC”
- arp.spoof on
- http.proxy


kevin@debianLaptop: ~
kevin@debianLaptop:~$ sudo bettercap -iface wlp4s0
[sudo] password for kevin:
bettercap v2.32.0 (built for linux amd64 with go1.19.8) [type 'help' for a list of commands]
192.168.0.0/24 > 192.168.0.103 » [18:14:49] [sys.log] [inf] gateway monitor started ...
192.168.0.0/24 > 192.168.0.103 » net.probe on
192.168.0.0/24 > 192.168.0.103 » [18:15:08] [sys.log] [inf] net.probe probing 256 addresses on 192.168.0.0/24
192.168.0.0/24 > 192.168.0.103 » [18:15:08] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.0.0/24 > 192.168.0.103 » [18:15:09] [endpoint.new] endpoint 192.168.0.100 detected as a4:f0:0f:75:f1:30.
192.168.0.0/24 > 192.168.0.103 » [18:15:09] [endpoint.new] endpoint 192.168.0.101 (DESKTOP-M2EPRIG) detected as c4:6e:1f:1d:28:c5 (Tp-Link Technologies Co.,Ltd.).
192.168.0.0/24 > 192.168.0.103 » net.show
IP
MAC
Name
Vendor
Sent
Recvd
Seen
192.168.0.103 00:21:6a:75:d5:1c wlp4s0 Intel Corporate 0 B 0 B 18:14:49
192.168.0.1 30:b5:c2:31:24:25 gateway Tp-Link Technologies Co.,Ltd. 26 kB 6.9 kB 18:14:49
192.168.0.100 a4:f0:0f:75:f1:30 210 B 276 B 18:15:24
192.168.0.101 c4:6e:1f:1d:28:c5 DESKTOP-M2EPRIG Tp-Link Technologies Co.,Ltd. 597 B 957 B 18:15:24
↑ 40 kB / ↓ 129 kB / 2282 pkts
192.168.0.0/24 > 192.168.0.103 » set arp.spoof.targets 192.168.0.101,192.168.0.100
192.168.0.0/24 > 192.168.0.103 » arp.spoof on
192.168.0.0/24 > 192.168.0.103 » [18:19:39] [sys.log] [inf] arp.spoof arp spoofer started, probing 2 targets.
192.168.0.0/24 > 192.168.0.103 » net.sniff on
192.168.0.0/24 > 192.168.0.103 »

Router Configuration

- Disabled AP isolation
  - lets devices talk to eachother
- Tested wired and wireless setups
- Still unable to intercept traffic



Attack Attempt Results

- Bettercap identified devices

- ARP spoofing did not succeed

- Traffic not redirected through bettercaps proxy

Possible Reasons for Failure

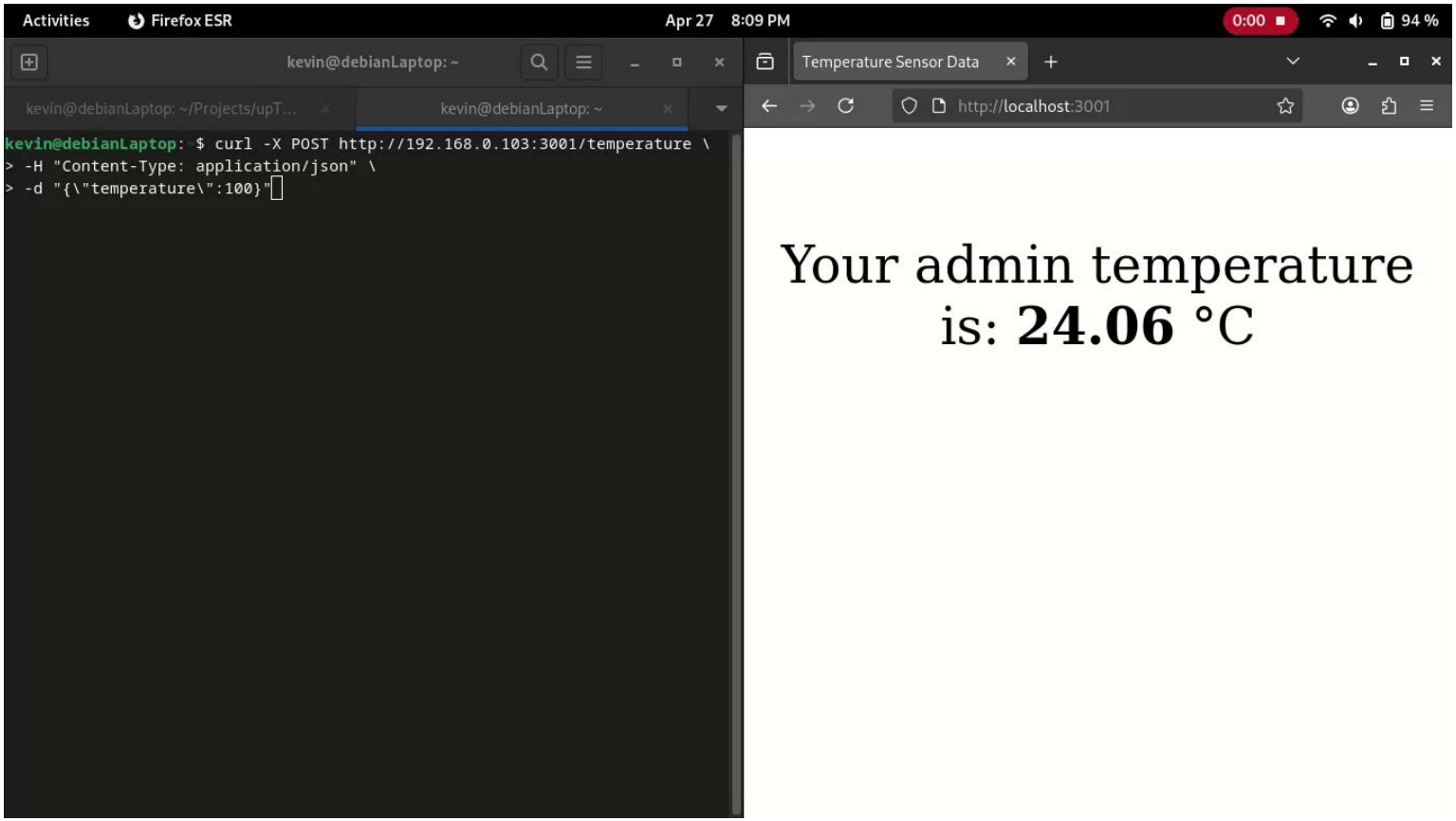
- Router firmware protections
  - cache flushing

- Dynamic ARP inspection
  - untrusted ports drop packets

- MAC validation or cache correction

Alternative Test (curl)

- Manual POST request to server
- Injected fake temperature data
- Temporary effect only



Limitations

- No successful MITM attack
  - Test environment constraints
    - firewall and proxy limitations
  - Router security unknown
    - firmware updates

Conclusion

- IoT systems can be vulnerable
- Modern routers may mitigate ARP attacks
- Security must be considered end-to-end

Future Work

- Test on different routers
- Implement HTTPS and authentication
- Try alternative attack methods
  - coding our own implementation
  - integration tests

References

[1] W. Stallings, Cryptography and network security: principles and practice. Upper Saddle River, N.J.: Prentice Hall, 2003. [Online]. Available: https://dl.acm.org/citation.cfm?id=599893

[2] B. S. Rao, C. V. Chakravarthi, and A. Jawahar, “Industrial Control Systems Security and Supervisory Control and Data Acquisition (SCADA),” International Journal for Modern Trends in Science and Technology, vol. 3, no. 10, pp. 109–118, Oct. 2017.

[3] D. Hercog, T. Lerher, M. Truntič, and O. Težak, “Design and Implementation of ESP32-Based IoT Devices,” Sensors, vol. 23, no. 15, p. 6739, 2023. doi: 10.3390/s23156739.

Thank you!

1865 RIDER UNIVERSITY