

ManInMiddle Attacks – the basics

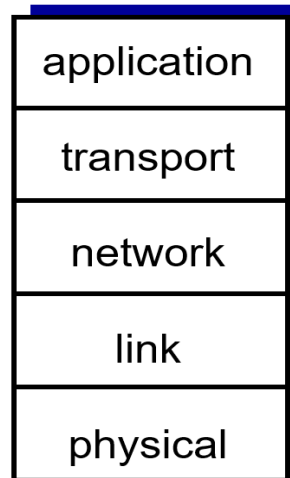
Michael Claudius, Associate Professor, Roskilde

25.09.2020

5-layer model

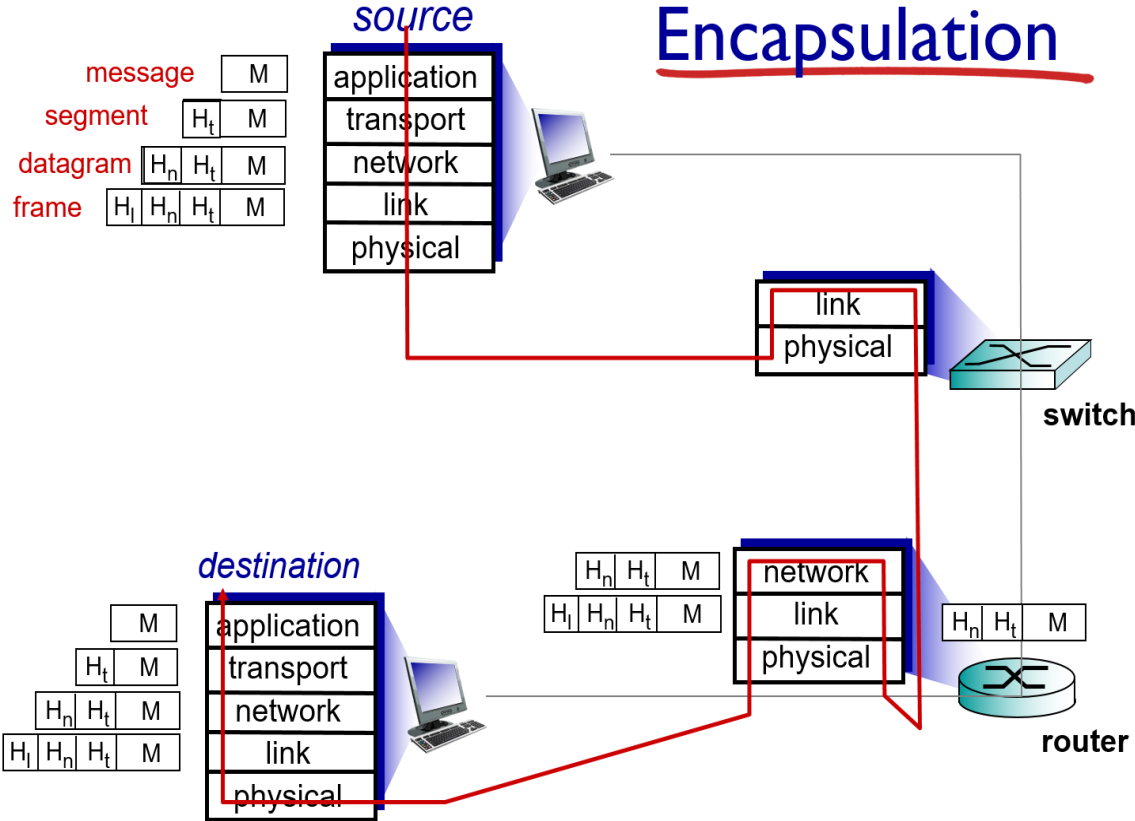
Internet protocol stack

- ❖ *application*: supporting network applications
 - FTP, SMTP, HTTP
- ❖ *transport*: process-process data transfer
 - TCP, UDP
- ❖ *network*: routing of datagrams from source to destination
 - IP, routing protocols
- ❖ *link*: data transfer between neighboring network elements
 - Ethernet, 802.111 (WiFi), PPP
- ❖ *physical*: bits “on the wire”



Introduction 1-60

Encapsulation of packets



Introduction 1-62

Vulnerability in protocols

Man in Middle attack utilizes vulnerability in Link Layer protocols:

- **ARP Poisoning of the ARP Table (Address Resolution Protocol)**
- **DNS Spoofing**
- **Session Hijacking**
- **SSL Spoofing**

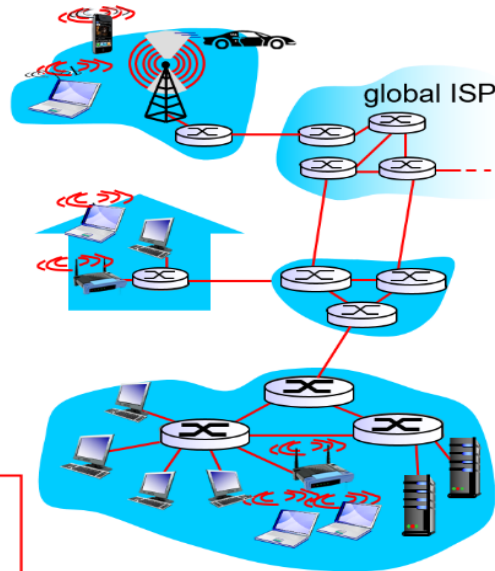
Link Layer Responsibility

Link layer: introduction

terminology:

- ❖ hosts and routers: **nodes**
- ❖ communication channels that connect adjacent nodes along communication path: **links**
 - wired links
 - wireless links
 - LANs
- ❖ layer-2 packet: **frame**, encapsulates datagram

data-link layer has responsibility of transferring datagram from one node to *physically adjacent* node over a link



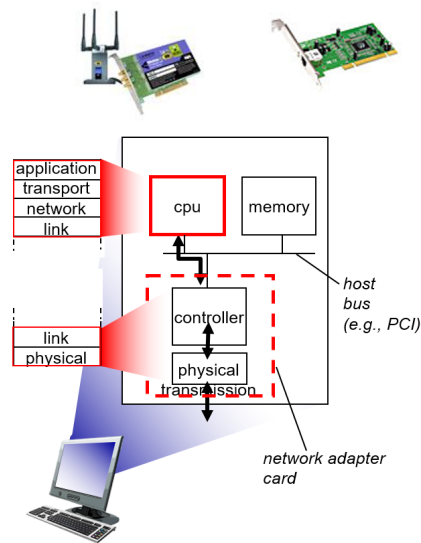
Link Layer 5-4

Link Layer Implementation

- Implemented in HW: NIC (Network Interface Card)
- Implemented in SW: CPU

Where is the link layer implemented?

- ❖ in each and every host
- ❖ link layer implemented in “adaptor” (aka *network interface card* NIC) or on a chip
 - Ethernet card, 802.11 card; Ethernet chipset
 - implements link, physical layer
- ❖ attaches into host’s system buses
- ❖ combination of hardware, software, firmware

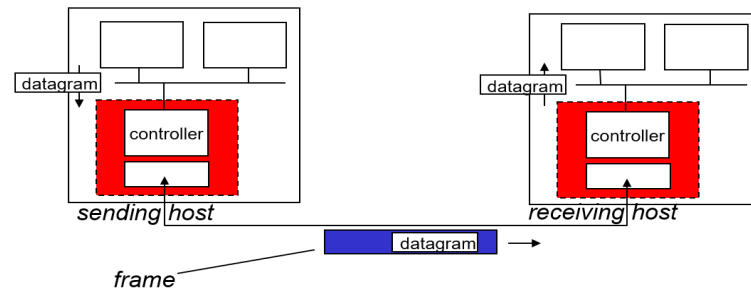


Link Layer 5-8

Purpose of Link Layer

- Encapsulate datagram in frame
- Send/Receive frames between adapters

Adaptors communicating

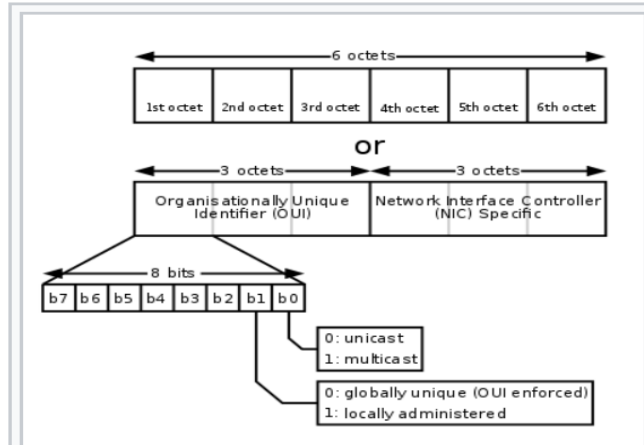



- ❖ sending side:
 - encapsulates datagram in frame
 - adds error checking bits, rdt, flow control, etc.
- ❖ receiving side
 - looks for errors, rdt, flow control, etc
 - extracts datagram, passes to upper layer at receiving side

Link Layer 5-9

Medium Access Control Address (MAC)

- 24 bits: OUI (Organization Unique Id), universally by manufacturer (may be overwritten)
- 24 bits: NIC specific



Structure of a 48-bit MAC address. 
The *group* bit (b0) distinguishes **multicast** and **unicast** addressing and the *local* or *U/L* bit (b1) distinguishes universal and locally administered addressing.

MAC addresses and ARP

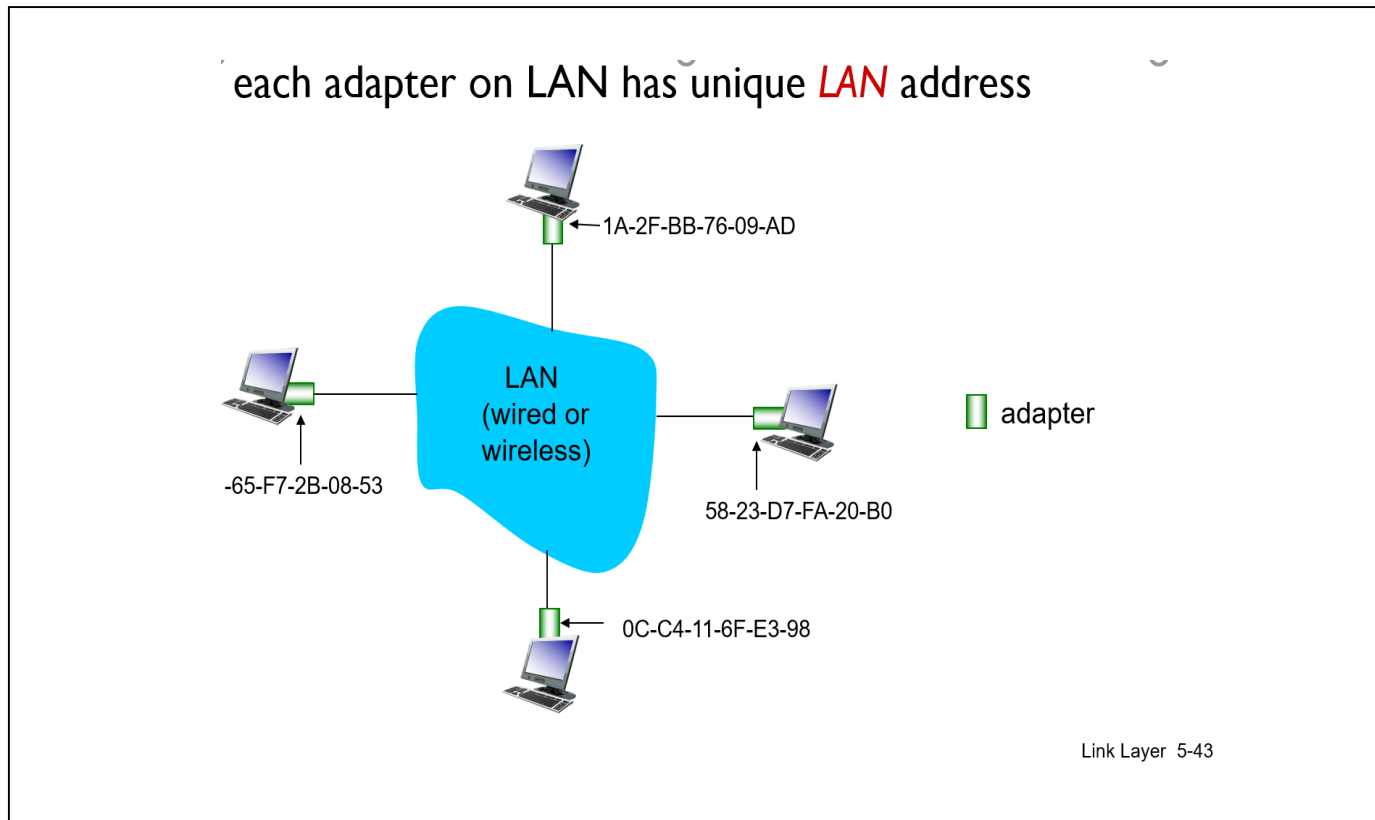
- ❖ 32-bit IP address:
 - *network-layer* address for interface
 - used for layer 3 (network layer) forwarding
- ❖ MAC (or LAN or physical or Ethernet) address:
 - function: *used 'locally' to get frame from one interface to another physically-connected interface (same network, in IP-addressing sense)*
 - 48 bit MAC address (for most LANs) burned in NIC ROM, also sometimes software settable
 - e.g.: 1A-2F-BB-76-09-AD

hexadecimal (base 16) notation
(each "number" represents 4 bits)

Link Layer 5-42

MAC addresses and ARP

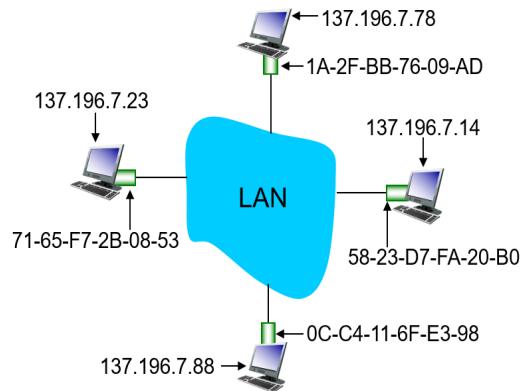
- **MAC Address is “permanent”. IP-address change**



ARP Table

- ARP table IP/MAC relation

Question: how to determine interface's MAC address, knowing its IP address?



ARP table: each IP node (host, router) on LAN has table

- IP/MAC address mappings for some LAN nodes:
< IP address; MAC address; TTL >
- TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

Link Layer 5-45

ARP Procedure Normal traffic

- A sends to B, but does not know B's MAC address

- ❖ A wants to send datagram to B
 - B's MAC address not in A's ARP table.
- ❖ A **broadcasts** ARP query packet, containing B's IP address
 - dest MAC address = FF-FF-FF-FF-FF-FF
 - all nodes on LAN receive ARP query
- ❖ B receives ARP packet, replies to A with its (B's) MAC address
 - frame sent to A's MAC address (unicast)
- ❖ A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
 - soft state: information that times out (goes away) unless refreshed
- ❖ ARP is "plug-and-play":
 - nodes create their ARP tables *without intervention from net administrator*

Link Layer 5-46

ARP Procedure Normal traffic

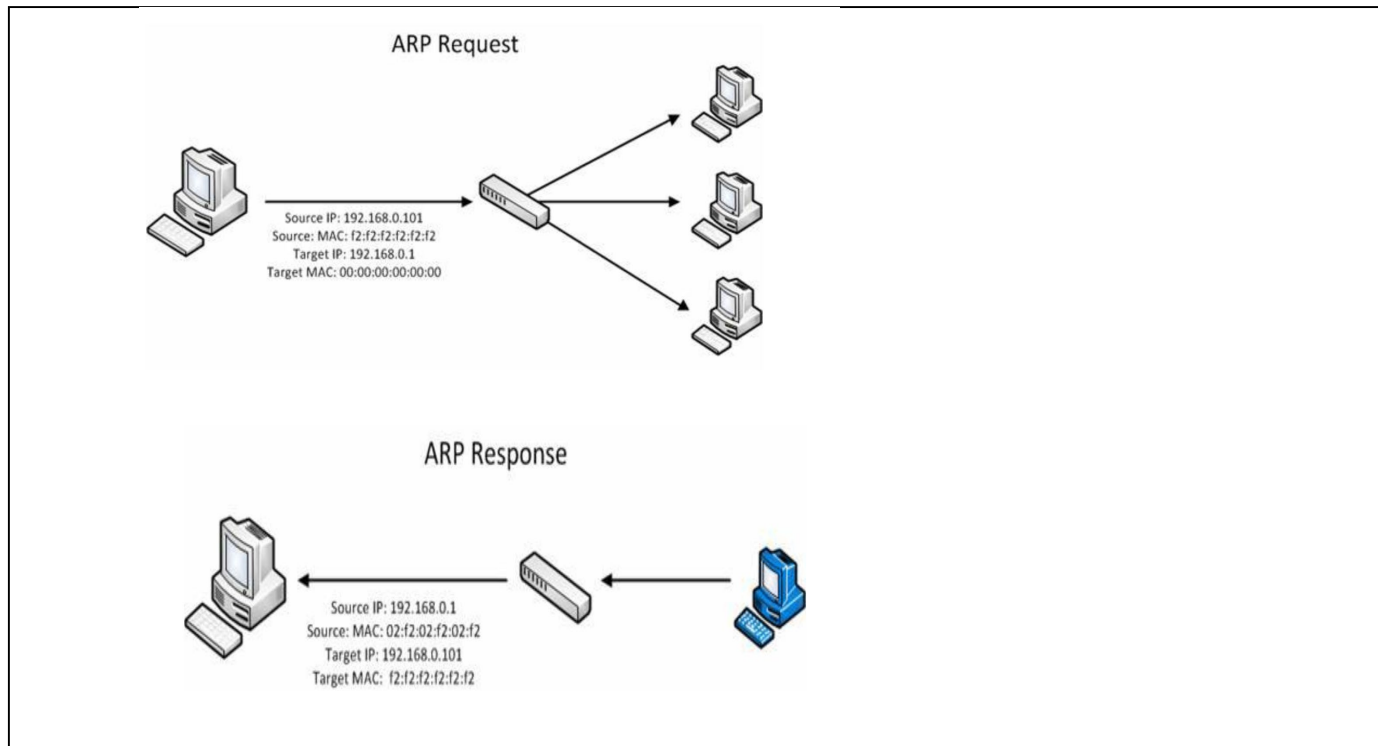
- A sends to B, but does not know B's MAC address

- ❖ A wants to send datagram to B
 - B's MAC address not in A's ARP table.
- ❖ A **broadcasts** ARP query packet, containing B's IP address
 - dest MAC address = FF-FF-FF-FF-FF-FF
 - all nodes on LAN receive ARP query
- ❖ B receives ARP packet, replies to A with its (B's) MAC address
 - frame sent to A's MAC address (unicast)
- ❖ A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
 - soft state: information that times out (goes away) unless refreshed
- ❖ ARP is "plug-and-play":
 - nodes create their ARP tables *without intervention from net administrator*

Link Layer 5-46

ARP Procedure Normal traffic

- A sends to B, but does not know B's MAC address

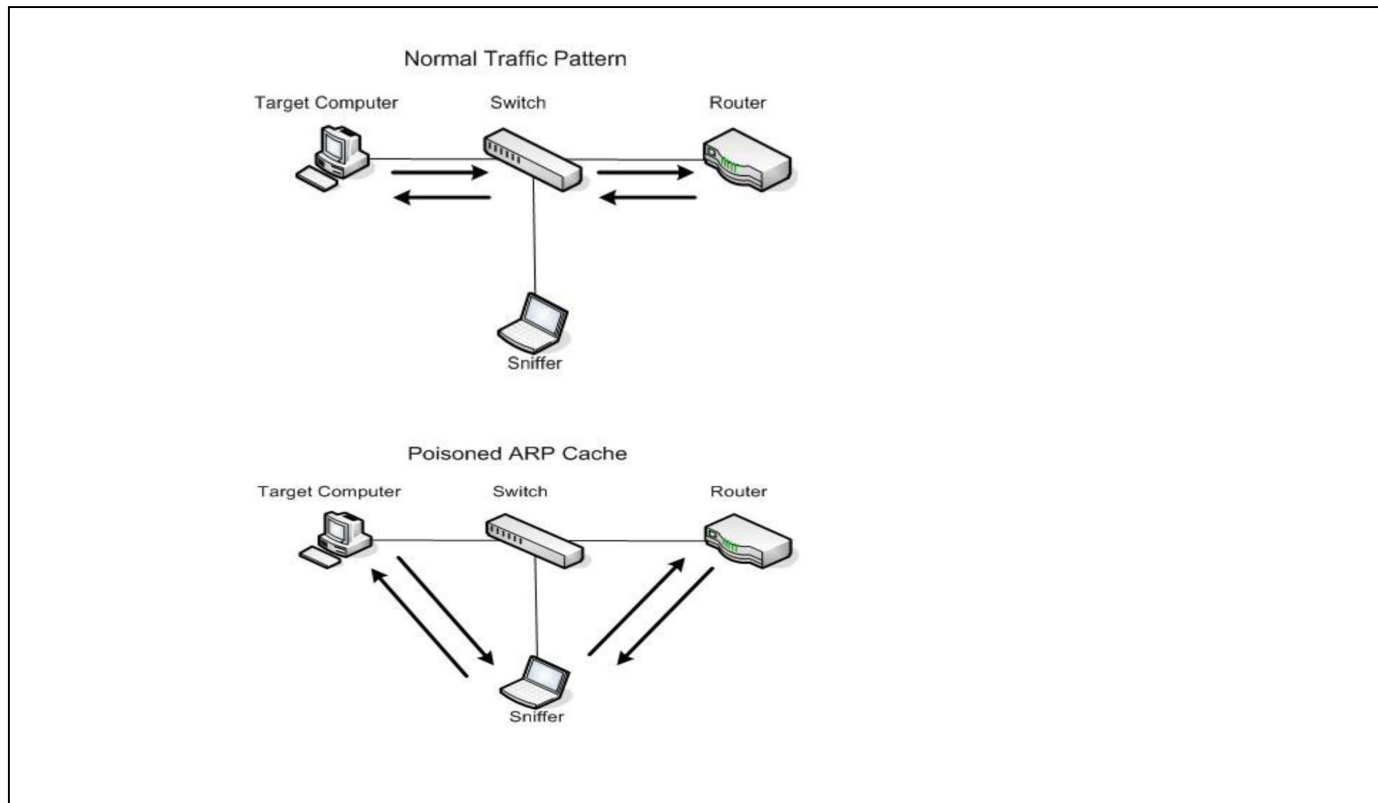


ARP Poisoning Principle

- Inside a LAN any host (T) can at any time and
- without an ARP-request send an ARP-reply to A
- updating the ARP table (at A) with a “fake” IP-address (of B) and attackers MAC address (T)

ARP Poisoning Traffic

- Traffic is bypassed to sniffer (attacker T)



Assignments

- Time for a little discussion and hard practical work
- Install Cain & Abel tool
- Perform
 - ARP Poisoning on your own LAN / hotspot
 - Together with friends or using a PC, router and your mobile(s)
 - Run as fast as you can with more spoofing

[ManInMiddle Attack Exercise](#)