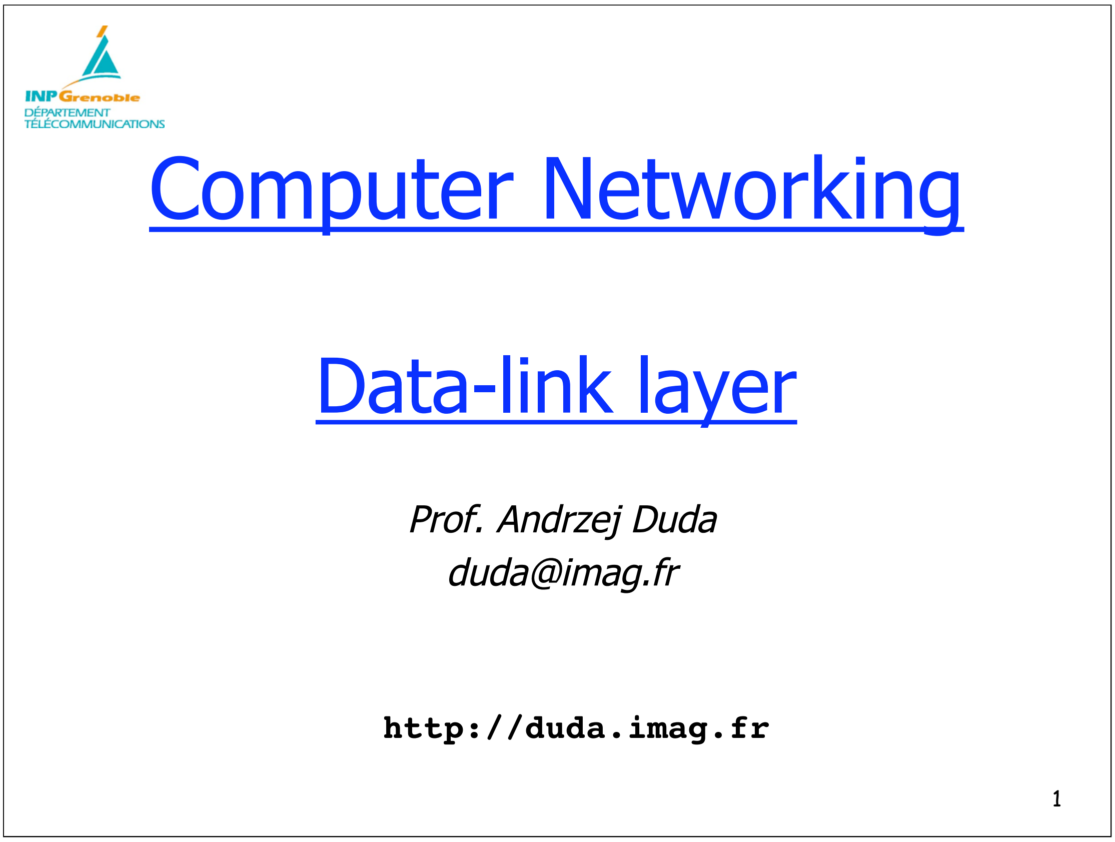


INP Grenoble
DÉPARTEMENT
TÉLÉCOMMUNICATIONS

Computer Networking

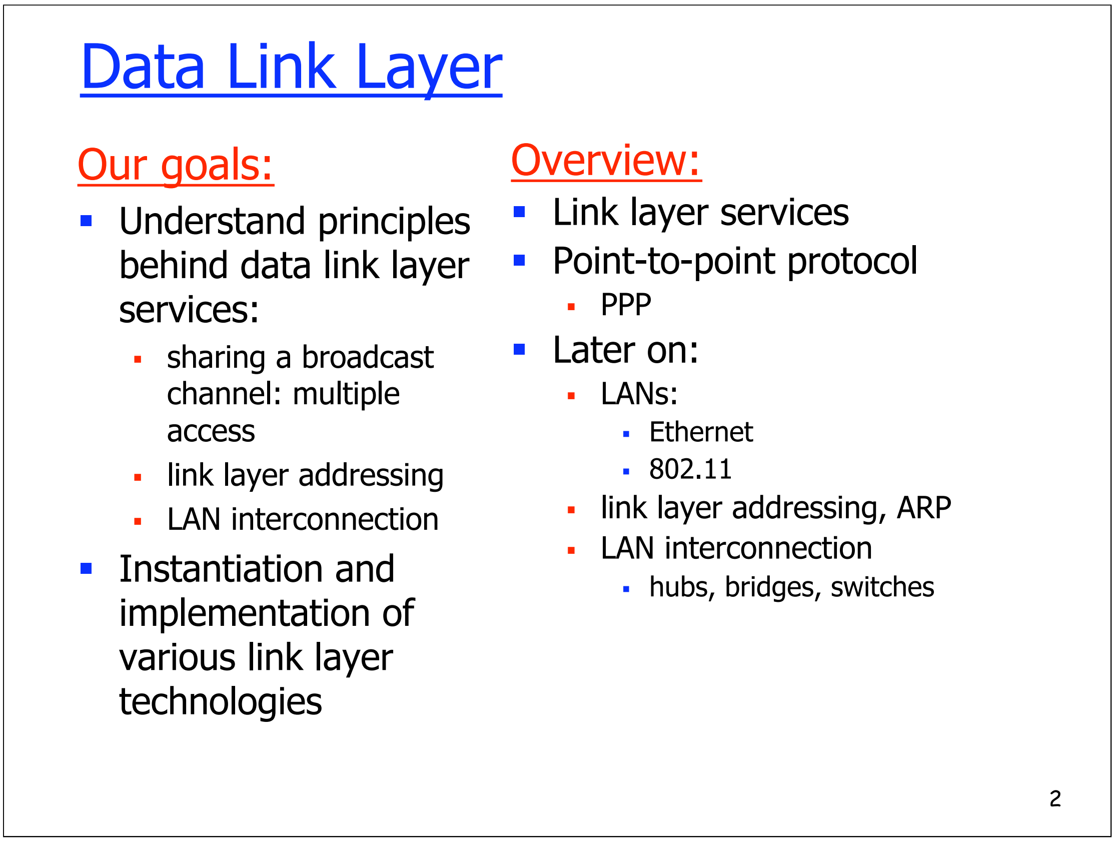
Data-link layer

Prof. Andrzej Duda
duda@imag.fr

http://duda.imag.fr

The **data-link layer** is responsible for transferring packets across a link which is the communication channel connecting two adjacent hosts or routers.

Examples of link-layer protocols include PPP and local area networks (LAN) such as Ethernet, and 802.11.



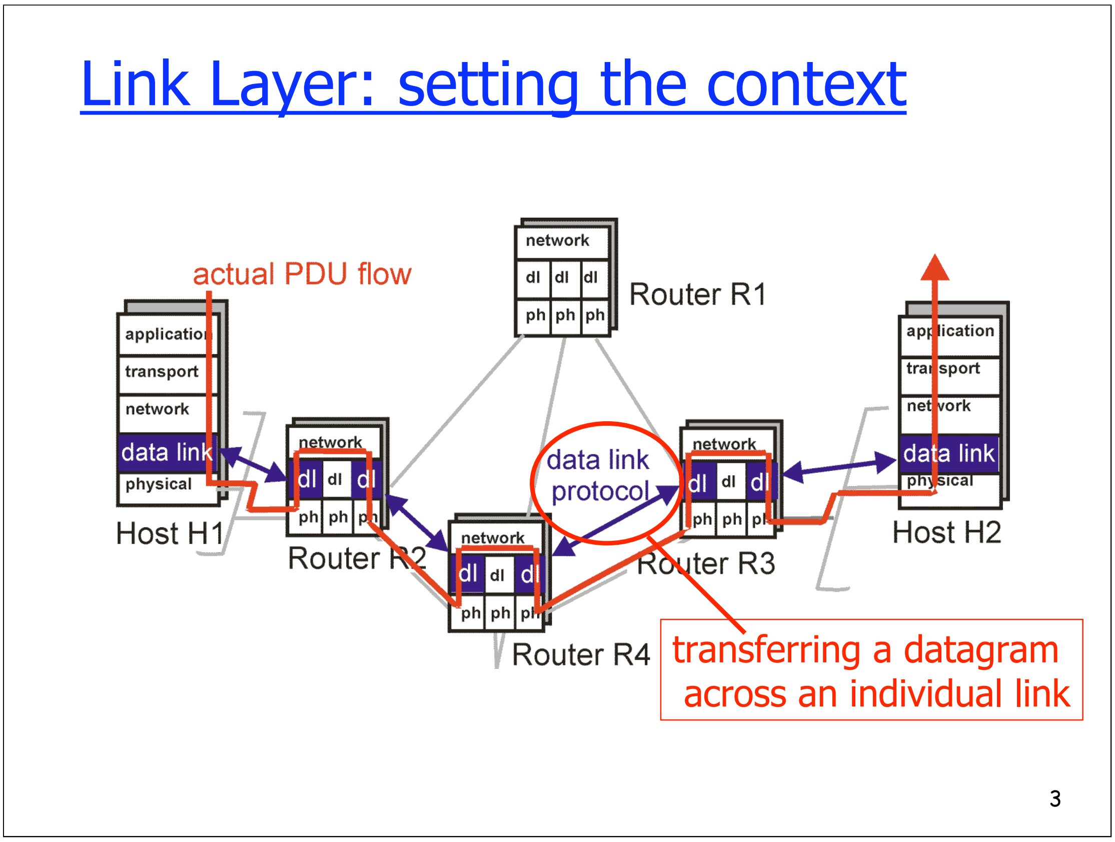
Data Link Layer

Our goals:

- Understand principles behind data link layer services:
  - sharing a broadcast channel: multiple access
  - link layer addressing
  - LAN interconnection
- Instantiation and implementation of various link layer technologies

Overview:

- Link layer services
- Point-to-point protocol
  - PPP
- Later on:
  - LANs:
    - Ethernet
    - 802.11
  - link layer addressing, ARP
  - LAN interconnection
    - hubs, bridges, switches

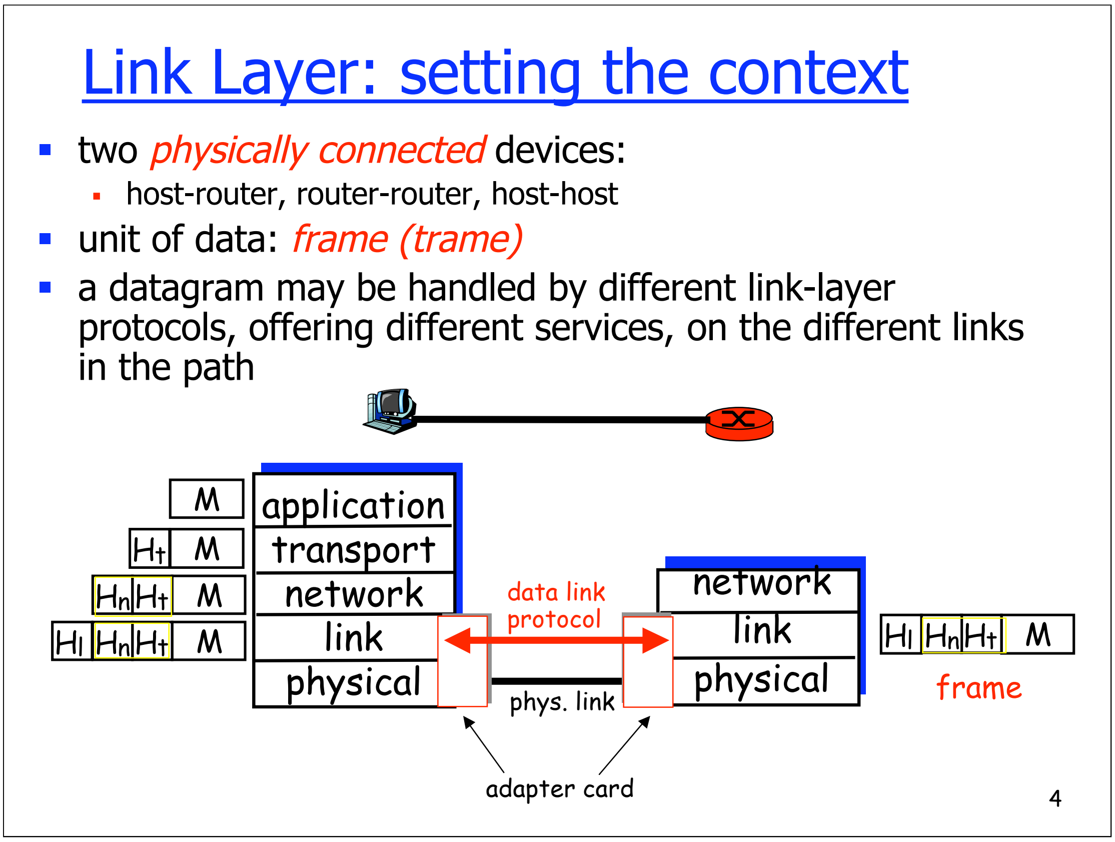


Link Layer: setting the context

transferring a datagram across an individual link

3

Communication networks provide a communication service between two hosts. This communication path starts at the source host, passes through a series of routers, and ends at the destination host. At that layer, where we are not particularly concerned whether a device is a router or a host, hosts and the routers are referred to simply as **nodes**, and to the communication channels that connect adjacent nodes along the communication path as **links.** In order to move a datagram from source host to destination host, the datagram must be moved over each of the *individual links* in the path. The **data-link layer** is responsible for transferring a datagram that comes from the network layer across an individual link.

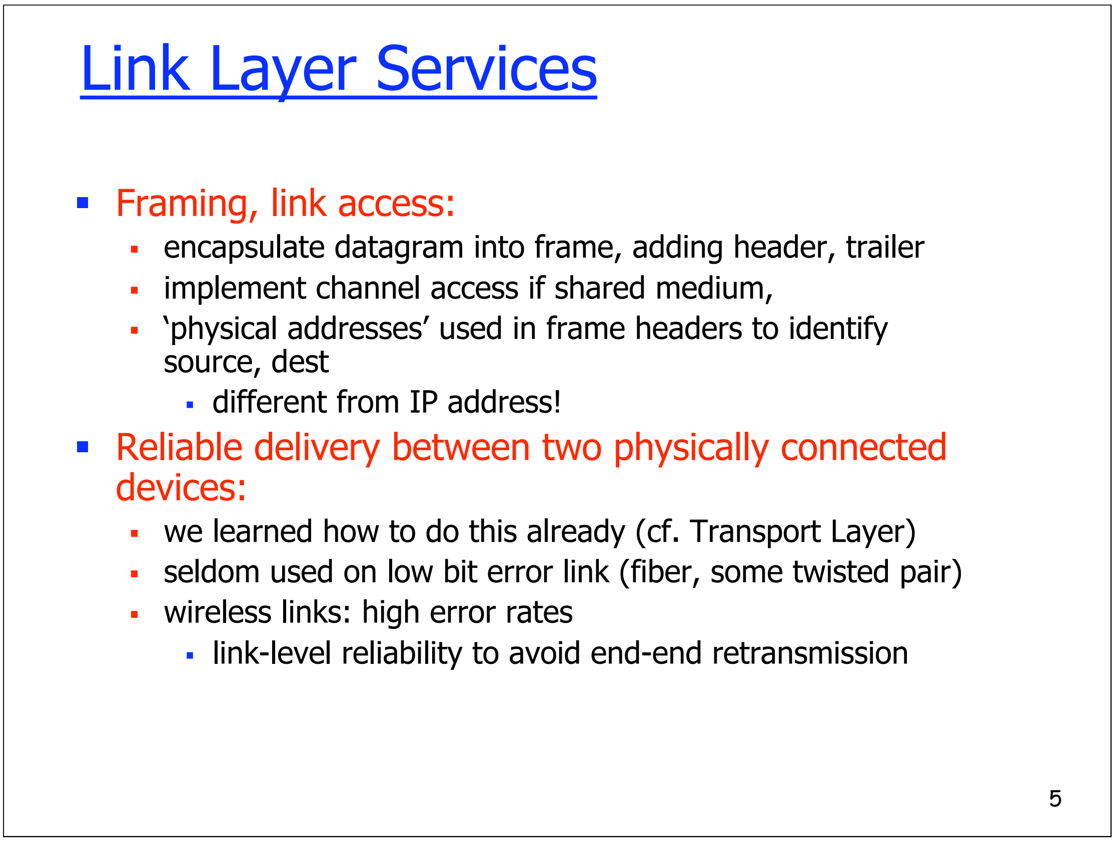


Link Layer: setting the context

- two *physically connected* devices:
  - host-router, router-router, host-host
- unit of data: *frame (trame)*
- a datagram may be handled by different link-layer protocols, offering different services, on the different links in the path

A link is the physical communication channels that connect either two host, two routers or a host-router pair. The **link-layer protocol** defines the format of the units of data (**frames**) exchanged between the nodes at the ends of the link, as well as the actions taken by these nodes when sending and receiving these data units. Each link-layer frame typically encapsulates one network-layer datagram.

A link-layer protocol has the node-to-node job of moving a network-layer datagram over a *single link* in the path. An important characteristic of the link layer is that a datagram may be handled by different link-layer protocols, offering different services, on the different links in the path.



Link Layer Services

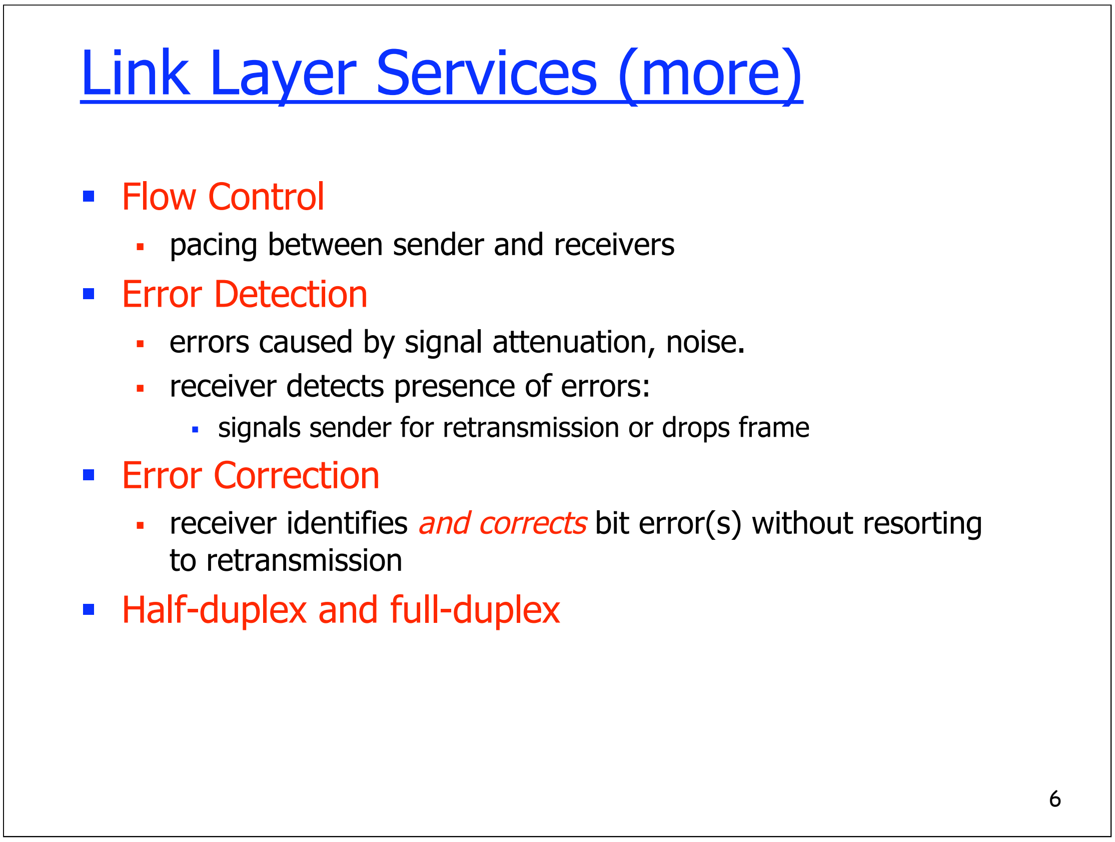
- Framing, link access:
  - encapsulate datagram into frame, adding header, trailer
  - implement channel access if shared medium,
  - 'physical addresses' used in frame headers to identify source, dest
    - different from IP address!
- Reliable delivery between two physically connected devices:
  - we learned how to do this already (cf. Transport Layer)
  - seldom used on low bit error link (fiber, some twisted pair)
  - wireless links: high error rates
    - link-level reliability to avoid end-end retransmission

5

Possible services that can be offered by a link-layer protocol include:

•*Framing and link access.* Almost all link-layer protocols encapsulate each network-layer datagram within a network-layer datagram is inserted, and a number of header fields. A data-link protocol specifies the structure of the frame, as well as a channel access protocol that specifies the rules by which a frame is transmitted onto the link. For point-to-point links that have a single sender on one end of the link and a single receiver at the other end of the link, the link-access protocol is simple (or non-existent)--the sender can send a frame whenever the link is idle. The more interesting case is when multiple nodes share a single broadcast link--the so-called multiple access problem. Here, the channel access protocol serves to coordinate the frame transmissions of the many nodes link-layer frame before transmission onto the link. A frame consists of a data field, in which the. The frame headers also often include fields for a node's so-called **physical address,** which is completely *distinct* from the node's network layer (for example, IP) address.

•*Reliable delivery.* When a link-layer protocol provides reliable-delivery service, it guarantees to move each network-layer datagram across the link without error. This is achieved with acknowledgments and retransmissions. A link-layer reliable-delivery service is often used for links that are prone to high error rates, such as a wireless link, with the goal of correcting an error locally, on the link where the error occurs, rather than forcing an end-to-end retransmission of the data by a transport- or application-layer protocol. However, link-layer reliable delivery can be considered an unnecessary overhead for low bit-error links, including fiber, coax, and many twisted-pair copper links. For this reason, many of the most popular link-layer protocols do not provide a reliable-delivery service.



Link Layer Services (more)

- Flow Control
  - pacing between sender and receivers
- Error Detection
  - errors caused by signal attenuation, noise.
  - receiver detects presence of errors:
    - signals sender for retransmission or drops frame
- Error Correction
  - receiver identifies *and corrects* bit error(s) without resorting to retransmission
- Half-duplex and full-duplex

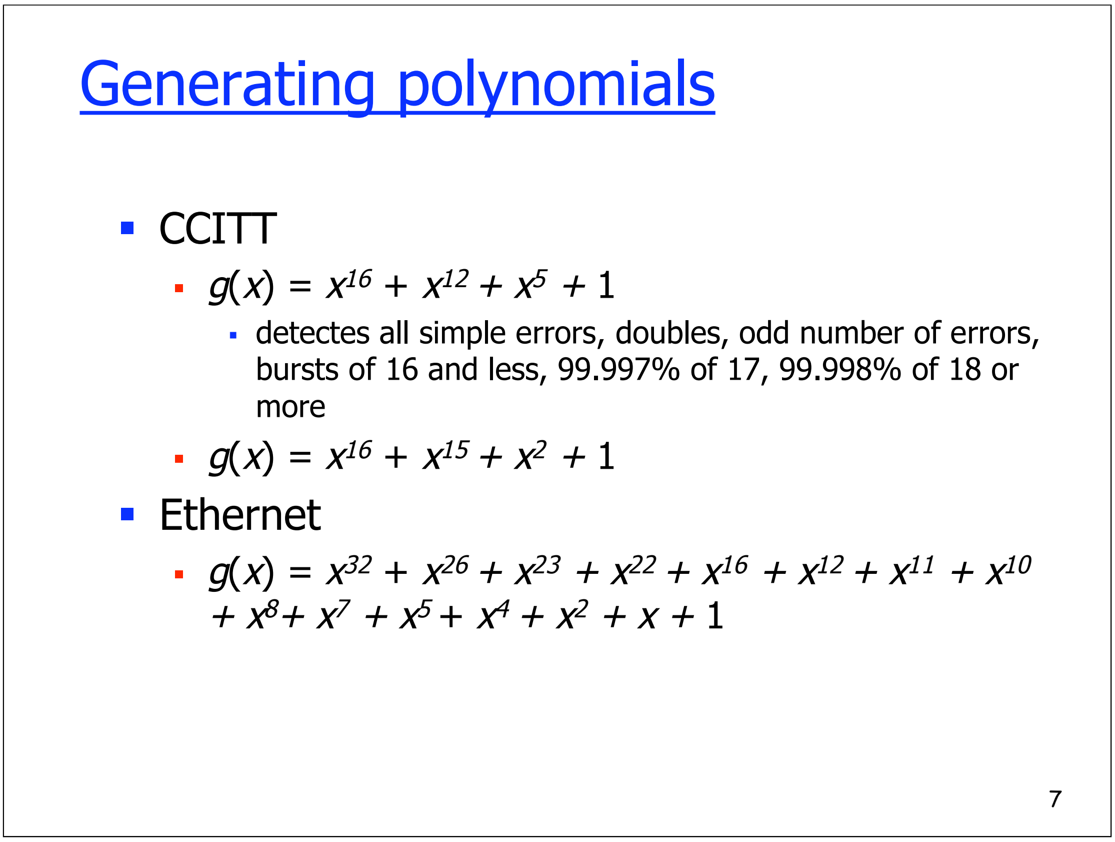
6

•*Flow control.* A link-layer protocol can provide flow control in order to prevent the sending node on one side of a link from overwhelming the receiving node on the other side of the link.

•*Error detection.* Many link-layer protocols provide a mechanism to detect the presence of one or more errors. This is done by having the transmitting node set error-detection bits in the frame, and having the receiving node perform an error check. Error detection is a very common service among link-layer protocols. Error detection in the link layer is usually more sophisticated than the one at the transport layer and network layers and implemented in hardware.

•*Error correction.* Error correction is similar to error detection, except that a receiver cannot only detect whether errors have been introduced in the frame but can also determine exactly where in the frame the errors have occurred (and hence correct these errors).

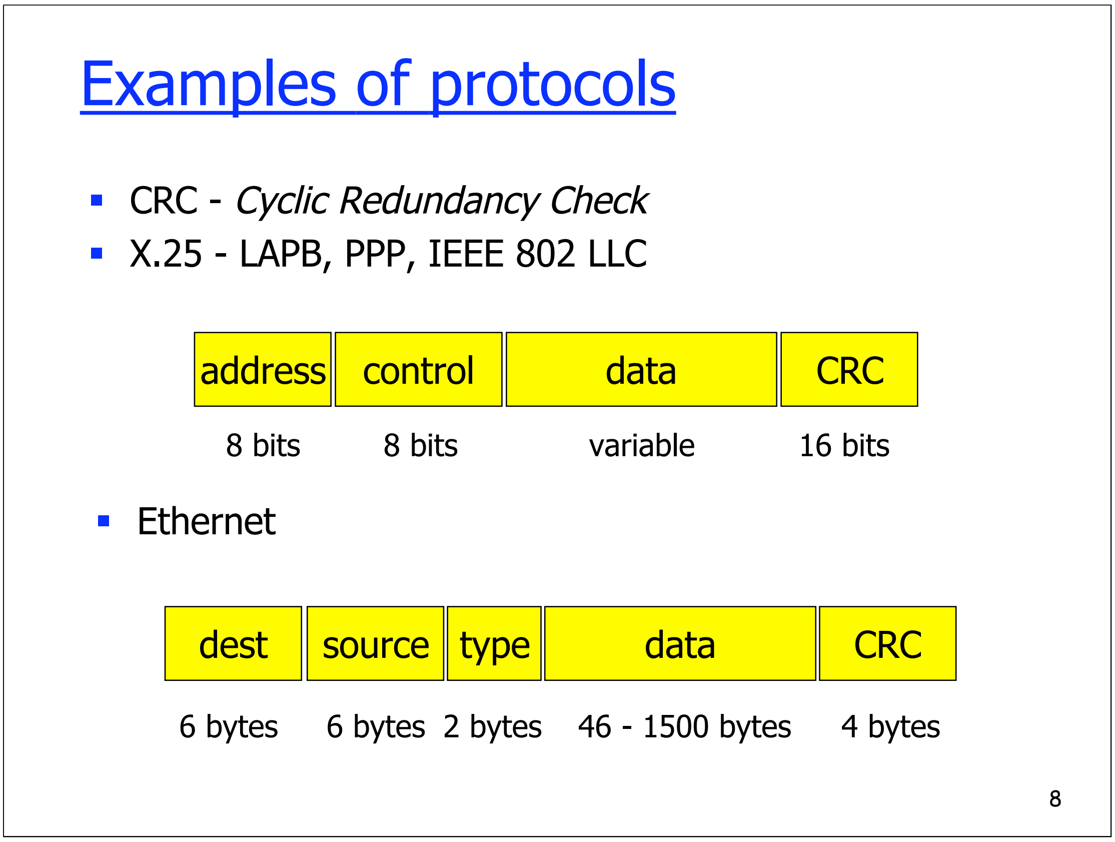
•*Half-duplex and full-duplex.* With full-duplex transmission, the nodes at both ends of a link may transmit packets at the same time. With half-duplex transmission, a node cannot both transmit and receive at the same time.



Generating polynomials

- CCITT
  - $g(x) = x^{16} + x^{12} + x^{5} + 1$
    - detectes all simple errors, doubles, odd number of errors, bursts of 16 and less, 99.997% of 17, 99.998% of 18 or more
  - $g(x) = x^{16} + x^{15} + x^{2} + 1$
- Ethernet
  - $g(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$

7



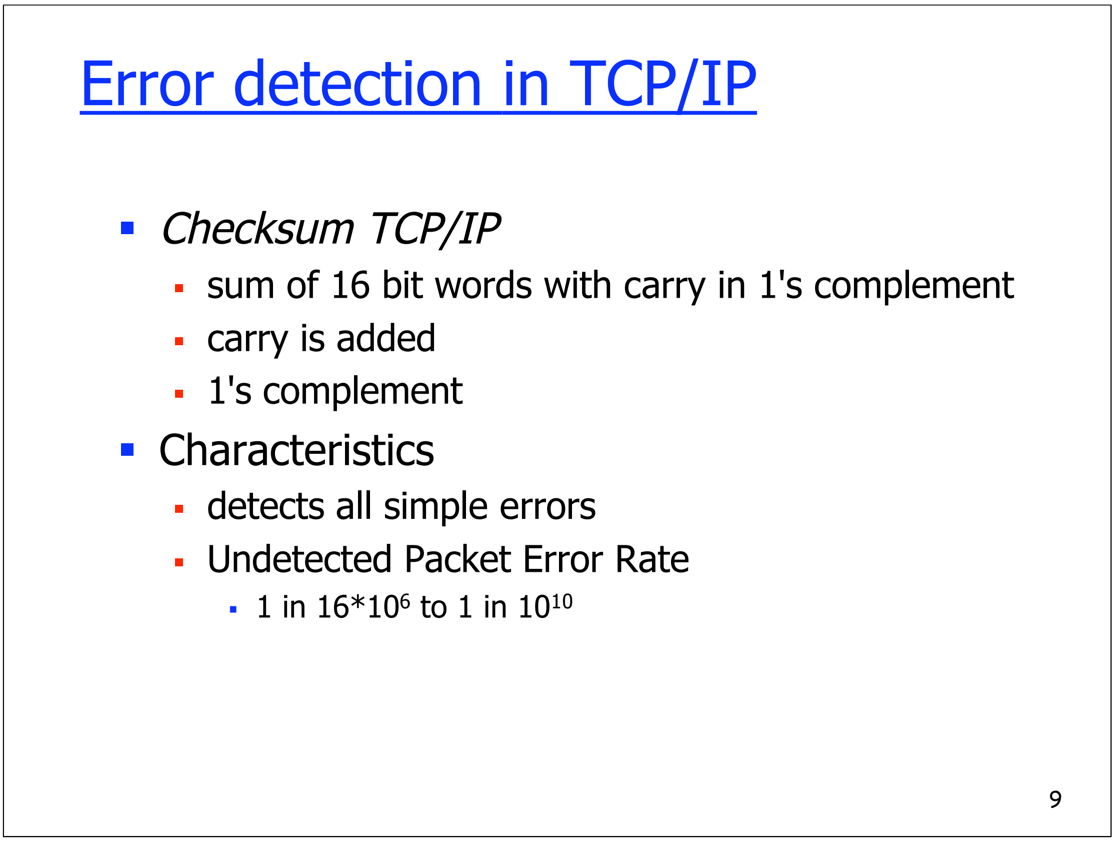
Examples of protocols

- CRC - *Cyclic Redundancy Check*
- X.25 - LAPB, PPP, IEEE 802 LLC

address	control	data	CRC
8 bits	8 bits	variable	16 bits

- Ethernet

dest	source	type	data	CRC
6 bytes	6 bytes	2 bytes	46 - 1500 bytes	4 bytes

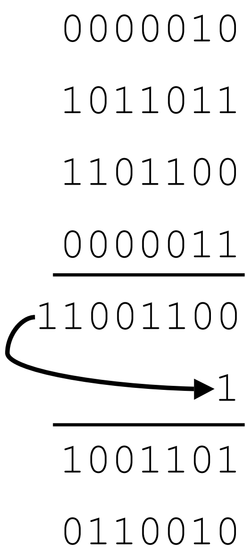


Error detection in TCP/IP

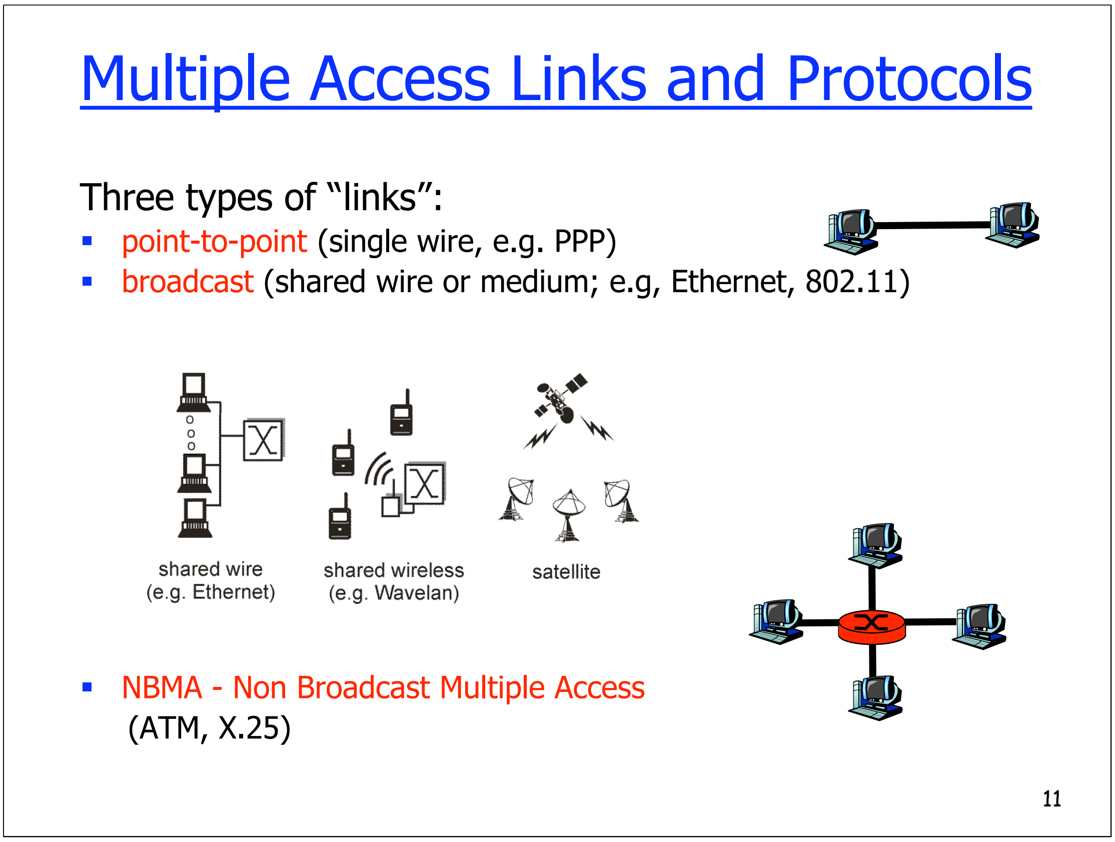
- *Checksum TCP/IP*
  - sum of 16 bit words with carry in 1's complement
  - carry is added
  - 1's complement
- Characteristics
  - detects all simple errors
  - Undetected Packet Error Rate
    - 1 in 16*10^6 to 1 in 10^{10}

Example

7 bit words:



```
 0000010
 1011011
 1101100
 0000011
--------
11001100
       1
--------
 1001101
 0110010
```



Multiple Access Links and Protocols

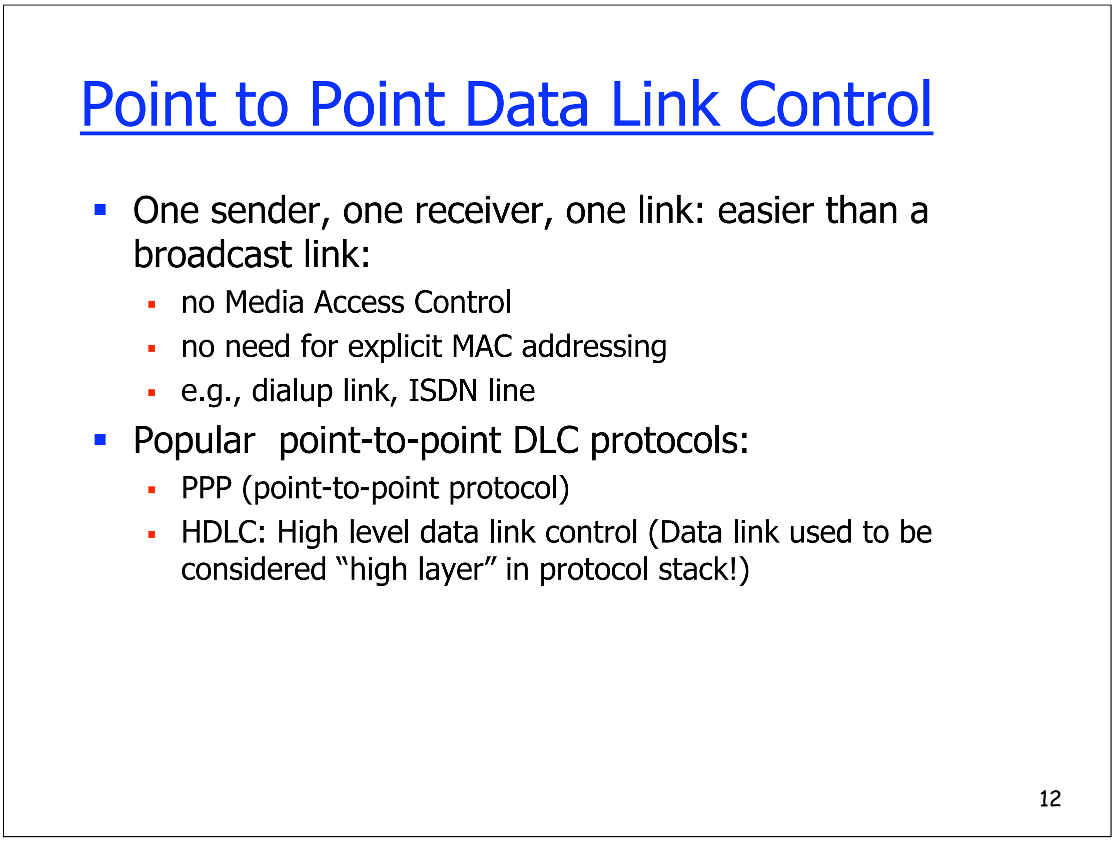
Three types of “links”:

- point-to-point (single wire, e.g. PPP)
- broadcast (shared wire or medium; e.g, Ethernet, 802.11)

- NBMA - Non Broadcast Multiple Access (ATM, X.25)

11

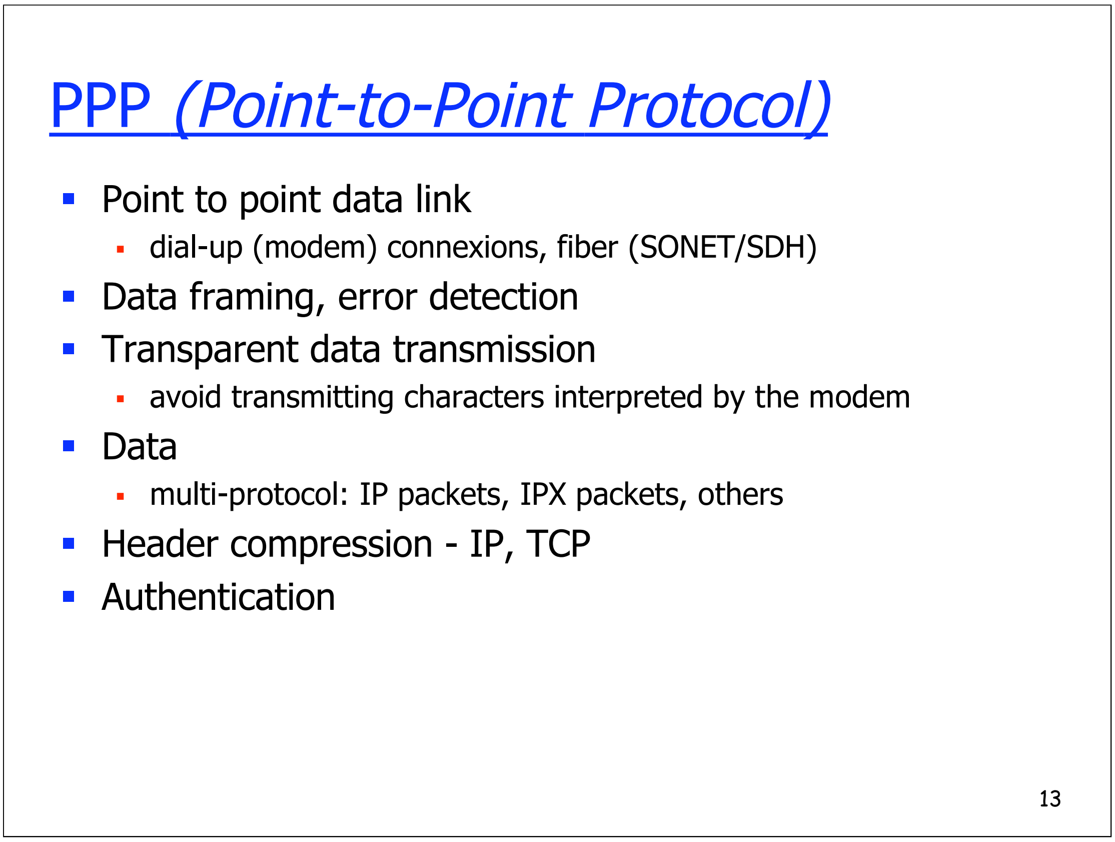
A **point-to-point link** consists of a single sender on one end of the link, and a single receiver at the other end of the link. Many link-layer protocols have been designed for point-to-point links; PPP and HDLC for example. The second type of link, a **broadcast link,** can have multiple sending and receiving nodes all connected to the same, single, shared broadcast channel, where each node on the channel receives a copy of any sent frame, e.g. Ethernet. In shared broadcast channel there is the problem of how to coordinate the access of multiple sending and receiving nodes to a --the so-called **multiple access problem.** Broadcast channels are often used in **local area networks (LANs),** networks that are geographically concentrated in a single building (or on a corporate or university campus). A NBMA link allows multiple and simultaneous dedicated access to the communication medium, but does not provide broadcast.



Point to Point Data Link Control

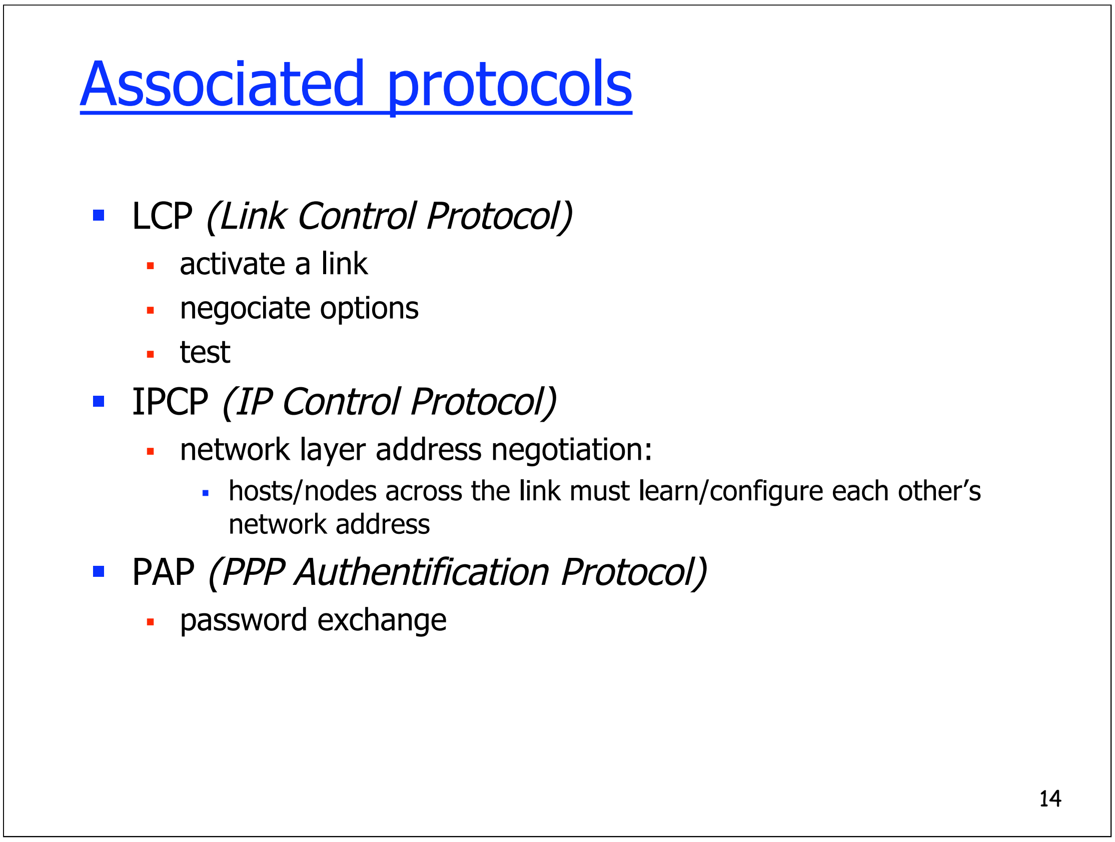
- One sender, one receiver, one link: easier than a broadcast link:
  - no Media Access Control
  - no need for explicit MAC addressing
  - e.g., dialup link, ISDN line
- Popular point-to-point DLC protocols:
  - PPP (point-to-point protocol)
  - HDLC: High level data link control (Data link used to be considered "high layer" in protocol stack!)

The **point-to-point protocol (PPP)** [RFC 1661; RFC 2153] is a data-link layer protocol that operates over a **point-to-point link**--a link directly connecting two nodes, one on each end of the link. The point-to-point link over which PPP operates might be a serial dialup telephone line (for example, a 56K modem connection), a SONET/SDH link, an X.25 connection, or an ISDN circuit. As noted above, PPP has become the protocol of choice for connecting home users to their ISPs over a dialup connection.



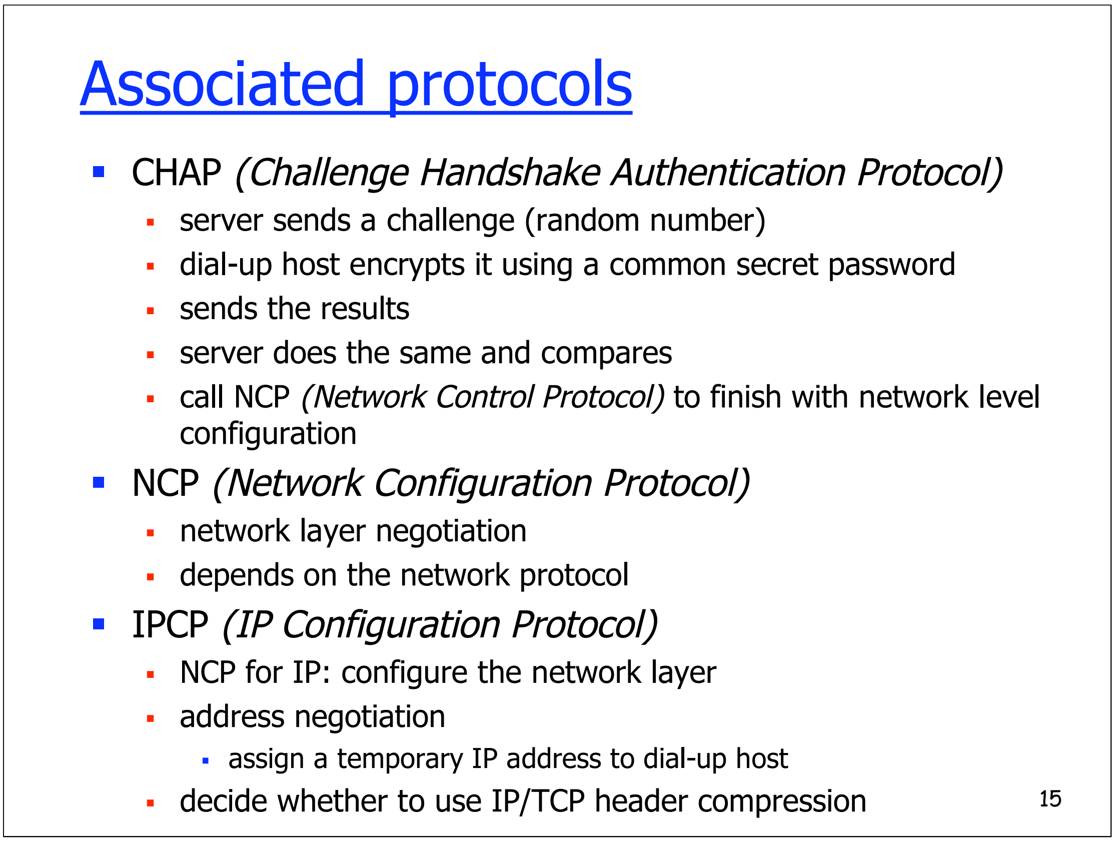
PPP *(Point-to-Point Protocol)*

- Point to point data link
  - dial-up (modem) connexions, fiber (SONET/SDH)
- Data framing, error detection
- Transparent data transmission
  - avoid transmitting characters interpreted by the modem
- Data
  - multi-protocol: IP packets, IPX packets, others
- Header compression - IP, TCP
- Authentication



Associated protocols

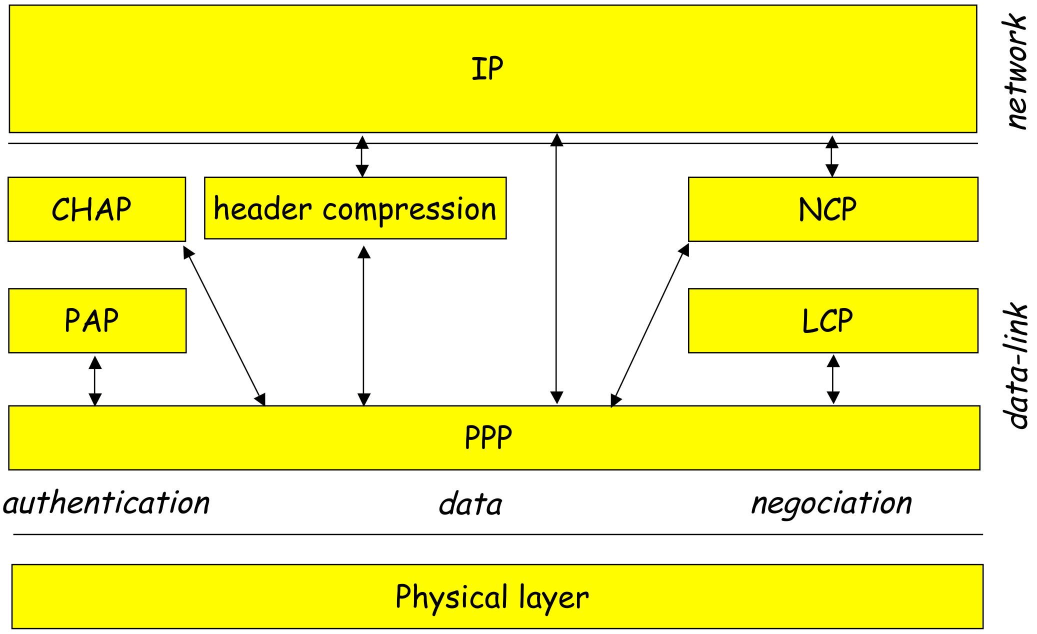
- LCP *(Link Control Protocol)*
  - activate a link
  - negociate options
  - test
- IPCP *(IP Control Protocol)*
  - network layer address negotiation:
    - hosts/nodes across the link must learn/configure each other’s network address
- PAP *(PPP Authentification Protocol)*
  - password exchange



Associated protocols

- CHAP *(Challenge Handshake Authentication Protocol)*
  - server sends a challenge (random number)
  - dial-up host encrypts it using a common secret password
  - sends the results
  - server does the same and compares
  - call NCP *(Network Control Protocol)* to finish with network level configuration
- NCP *(Network Configuration Protocol)*
  - network layer negotiation
  - depends on the network protocol
- IPCP *(IP Configuration Protocol)*
  - NCP for IP: configure the network layer
  - address negotiation
    - assign a temporary IP address to dial-up host
  - decide whether to use IP/TCP header compression

Protocols


IP

network

CHAP

header compression

NCP

PAP

LCP

data-link

PPP

authentication

data

negociation

Physical layer

LCP *(Link Control Protocol)*

- activate a link
- negociate configuration options
    - choice of the authentication protocol
    - server call-back
- test the link
- authenticate the user
- call NCP *(Network Control Protocol)* to finish with network level configuration

PAP *(PPP Authentication Protocol)*

- send password in the clear

CHAP *(Challenge Handshake Authentication Protocol)*

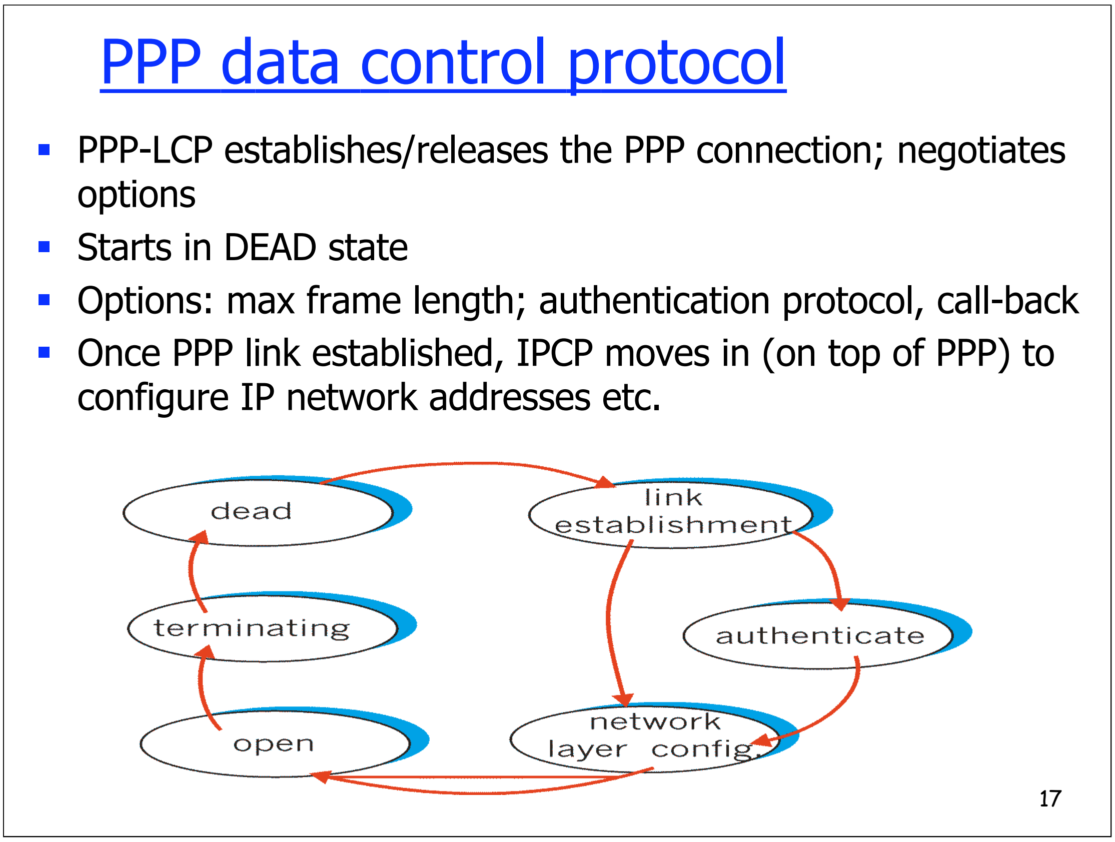
- server sends a challenge (random number)
- dial-up host encrypts it using a common secret password
- sends the results
- server does the same and compares

NCP *(Network Configuration Protocol)*

- network layer negotiation
- depends on the network protocol

IPCP *(IP Configuration Protocol)*

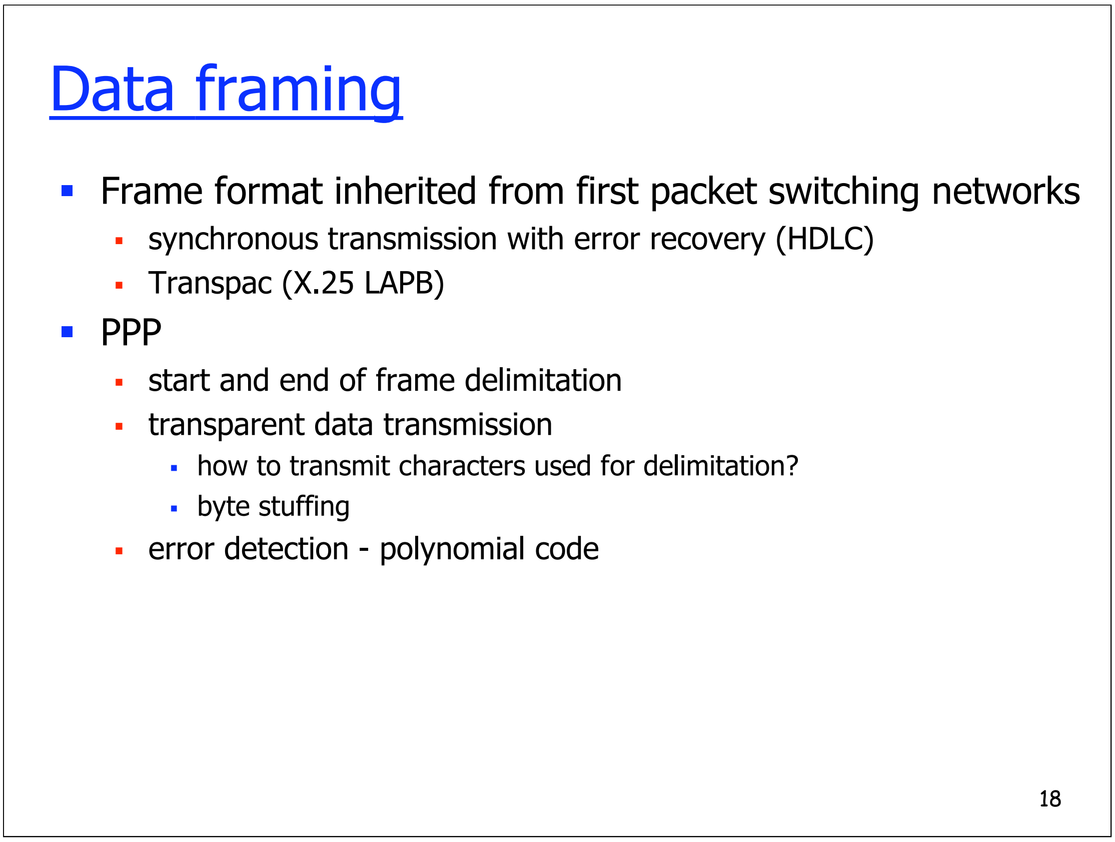
- NCP for IP: configure the network layer
- address negotiation



PPP data control protocol

- PPP-LCP establishes/releases the PPP connection; negotiates options
- Starts in DEAD state
- Options: max frame length; authentication protocol, call-back
- Once PPP link established, IPCP moves in (on top of PPP) to configure IP network addresses etc.

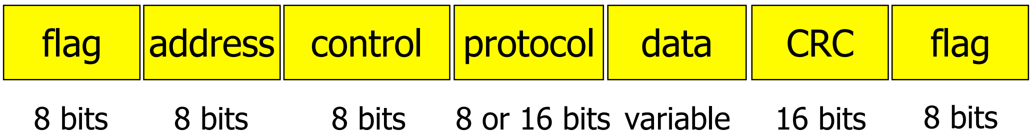
Before any data is exchanged over a PPP link, the two peers (one at each end of the PPP link) must first specify the desired link configuration options using an LCP configure-request frame. Once the link has been established, link options negotiated, and the authentication (if any) performed, the two sides of the PPP link then exchange network-layer-specific network control packets with each other. In the case of IP protocol, IP control protocol (IPCP) is used.



Data framing

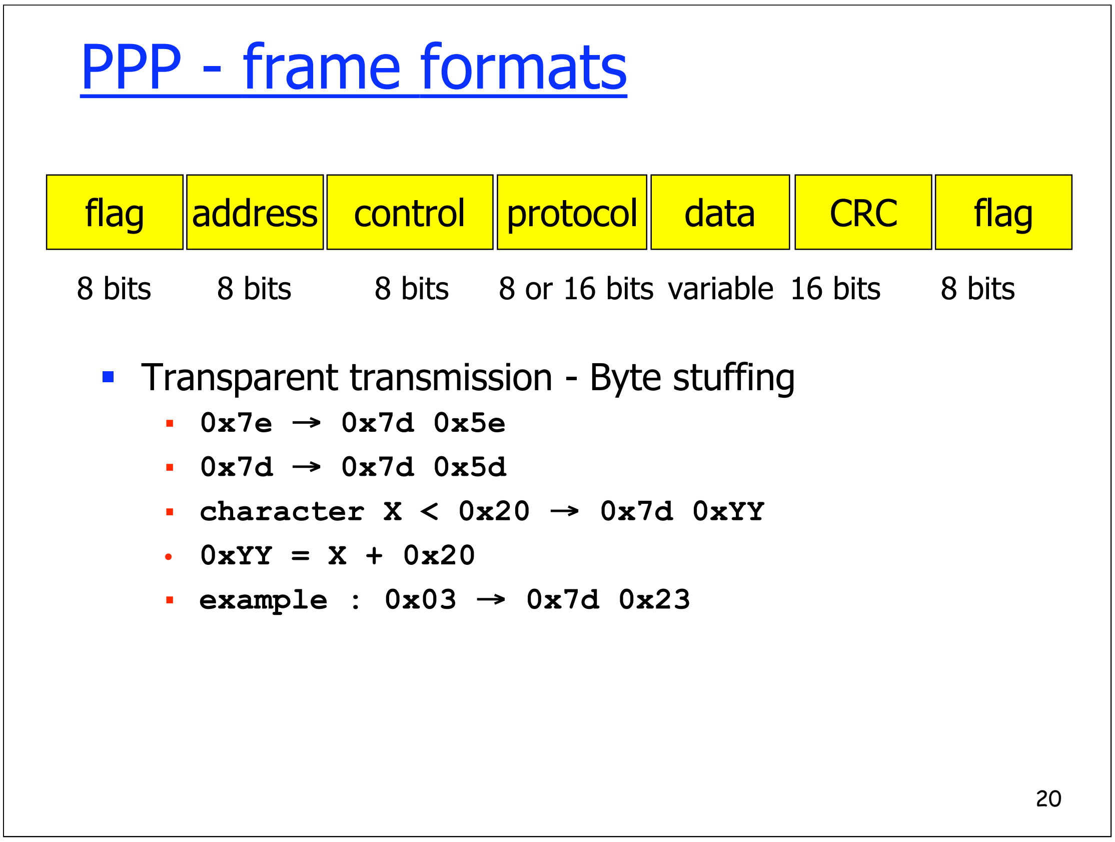
- Frame format inherited from first packet switching networks
  - synchronous transmission with error recovery (HDLC)
  - Transpac (X.25 LAPB)
- PPP
  - start and end of frame delimitation
  - transparent data transmission
    - how to transmit characters used for delimitation?
    - byte stuffing
  - error detection - polynomial code

PPP - frame formats



flag	address	control	protocol	data	CRC	flag
8 bits	8 bits	8 bits	8 or 16 bits	variable	16 bits	8 bits

- Flag: 01111110 - **`0x7e`**
- Address: 11111111
- Control: 00000011
- Protocol: LCP, IPCP, IP
- Data: 1500 bytes by default
- CRC: 16 bit CCITT polynomial code

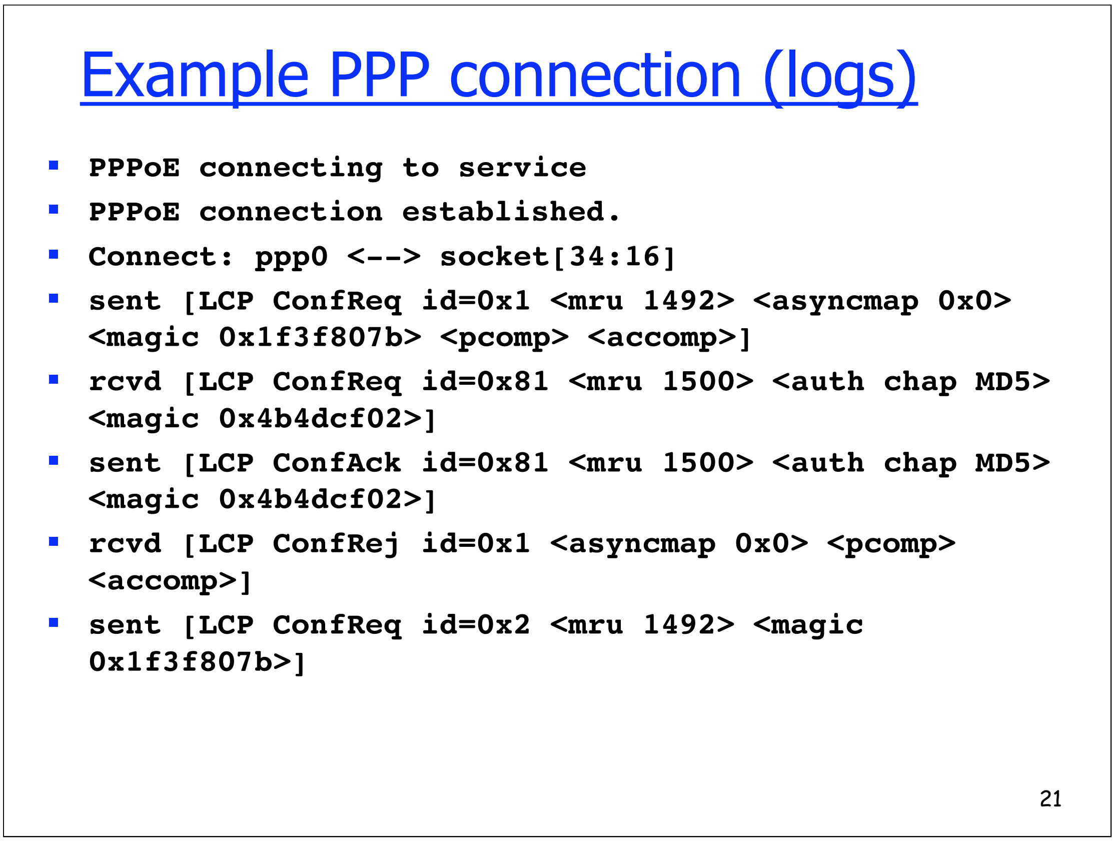


PPP - frame formats

flag	address	control	protocol	data	CRC	flag
8 bits	8 bits	8 bits	8 or 16 bits	variable	16 bits	8 bits

- Transparent transmission - Byte stuffing
  - `0x7e → 0x7d 0x5e`
  - `0x7d → 0x7d 0x5d`
  - `character X < 0x20 → 0x7d 0xYY`
  - `0xYY = X + 0x20`
  - `example : 0x03 → 0x7d 0x23`

20



Example PPP connection (logs)

- PPPoE connecting to service
- PPPoE connection established.
- Connect: ppp0 <--> socket[34:16]
- sent [LCP ConfReq id=0x1 <mru 1492> <asyncmap 0x0> <magic 0x1f3f807b> <pcomp> <accomp>]
- rcvd [LCP ConfReq id=0x81 <mru 1500> <auth chap MD5> <magic 0x4b4dcf02>]
- sent [LCP ConfAck id=0x81 <mru 1500> <auth chap MD5> <magic 0x4b4dcf02>]
- rcvd [LCP ConfRej id=0x1 <asyncmap 0x0> <pcomp> <accomp>]
- sent [LCP ConfReq id=0x2 <mru 1492> <magic 0x1f3f807b>]

21

MRU: maximum reception unit

id: match Req with Rep

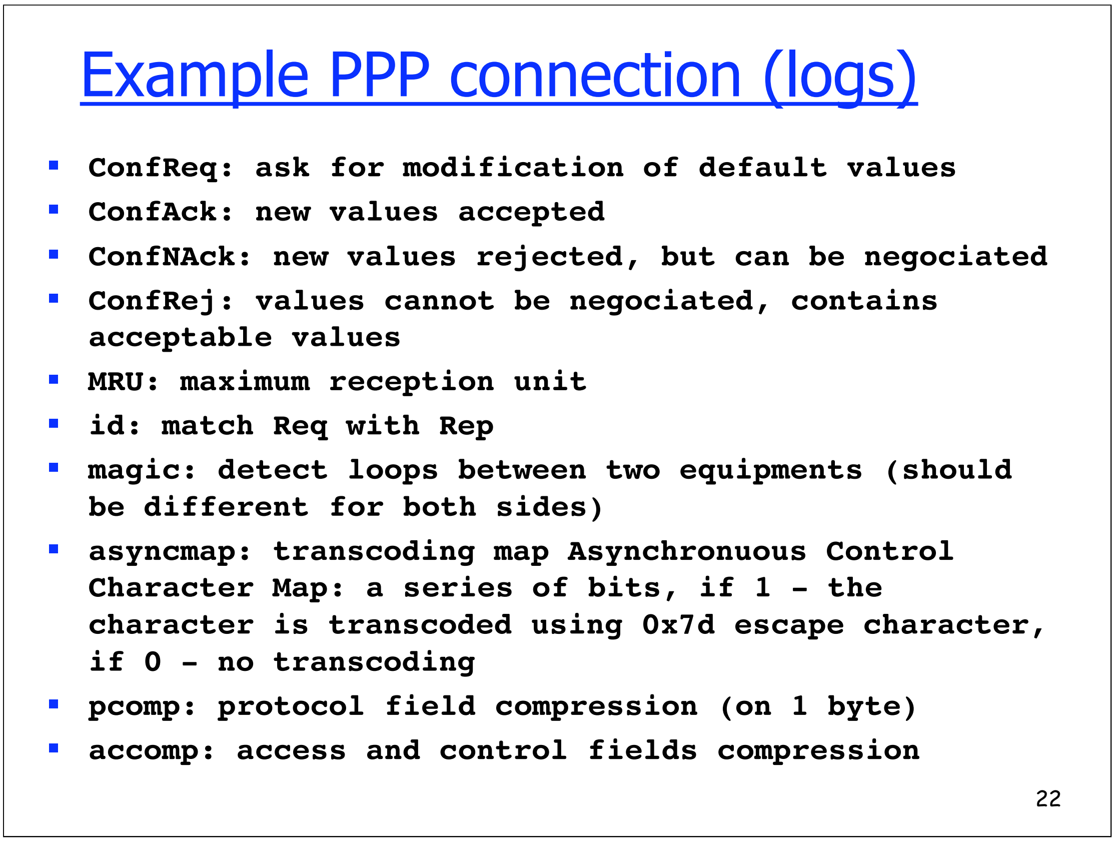
magic: detect loops between two equipments (should be different for both sides)

asyncmap: transcoding map Asynchronuous Control Character Map: a series of bits, if 1 - the character is transcoded using 0x7d escape character, if 0 - no transcoding

pcomp: protocol field compression (on 1 byte)

accomp: access and control fields compression

RFC1661: When a Configure-Request is received with a Magic-Number Configuration Option, the received Magic-Number is compared with the Magic-Number of the last Configure-Request sent to the peer. If the two Magic-Numbers are different, then the link is not looped-back, and the Magic-Number SHOULD be acknowledged. If the two Magic-Numbers are equal, then it is possible, but not certain, that the link is looped-back and that this Configure-Request is actually the one last sent. To determine this, a Configure-Nak MUST be sent specifying a different Magic-Number value. A new Configure-Request SHOULD NOT be sent to the peer until normal processing would cause it to be sent (that is, until a Configure-Nak is received or the Restart timer runs out).



Example PPP connection (logs)

- ConfReq: ask for modification of default values
- ConfAck: new values accepted
- ConfNAck: new values rejected, but can be negociated
- ConfRej: values cannot be negociated, contains acceptable values
- MRU: maximum reception unit
- id: match Req with Rep
- magic: detect loops between two equipments (should be different for both sides)
- asyncmap: transcoding map Asynchronuous Control Character Map: a series of bits, if 1 - the character is transcoded using 0x7d escape character, if 0 - no transcoding
- pcomp: protocol field compression (on 1 byte)
- accomp: access and control fields compression

MRU: maximum reception unit

id: match Req with Rep

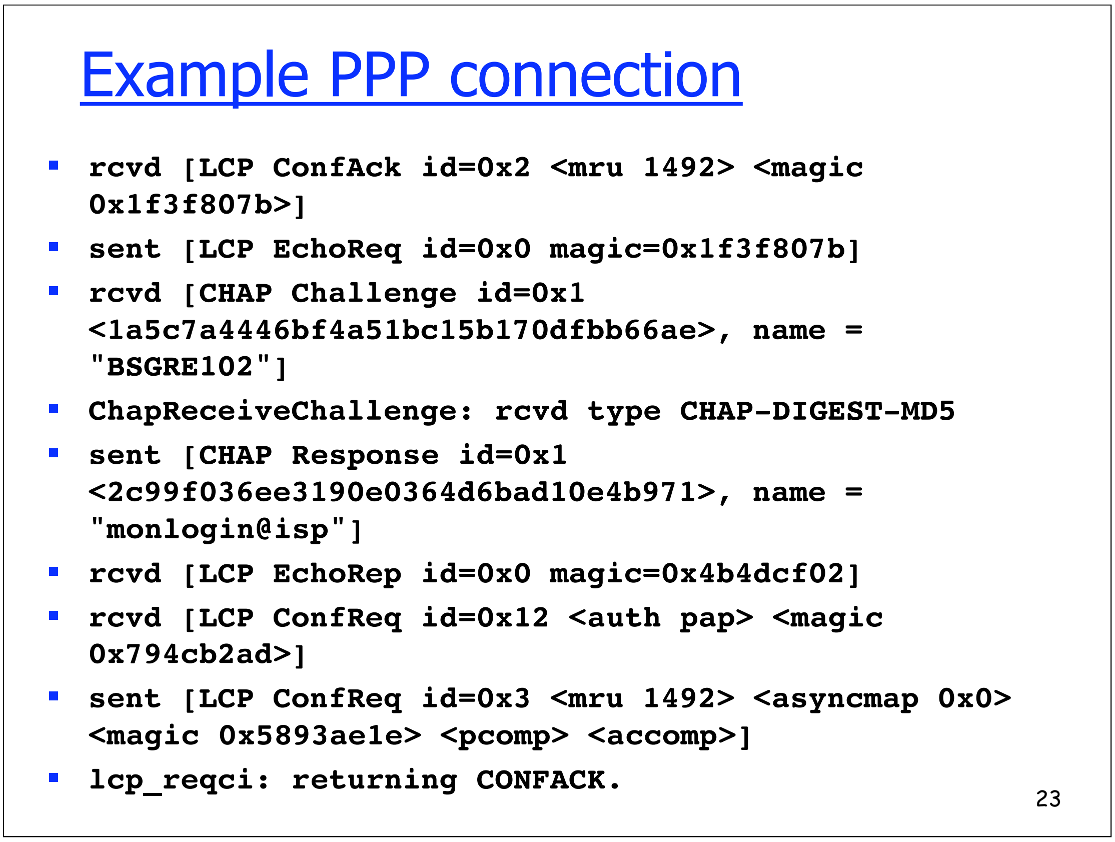
magic: detect loops between two equipments (should be different for both sides)

asyncmap: transcoding map Asynchronuous Control Character Map: a series of bits, if 1 - the character is transcoded using 0x7d escape character, if 0 - no transcoding

pcomp: protocol field compression (on 1 byte)

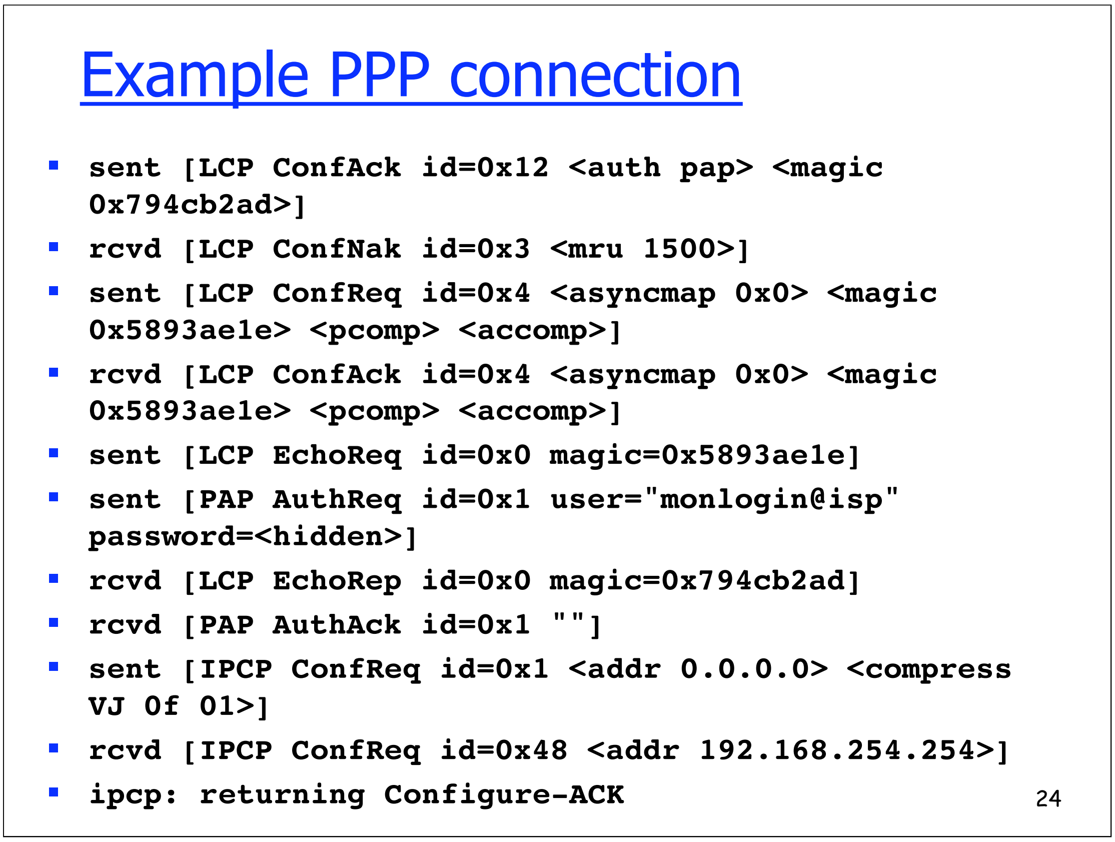
accomp: access and control fields compression

RFC1661: When a Configure-Request is received with a Magic-Number
Configuration Option, the received Magic-Number is compared with
the Magic-Number of the last Configure-Request sent to the peer.
If the two Magic-Numbers are different, then the link is not
looped-back, and the Magic-Number SHOULD be acknowledged. If the
two Magic-Numbers are equal, then it is possible, but not certain,
that the link is looped-back and that this Configure-Request is
actually the one last sent. To determine this, a Configure-Nak
MUST be sent specifying a different Magic-Number value. A new
Configure-Request SHOULD NOT be sent to the peer until normal
processing would cause it to be sent (that is, until a Configure-
Nak is received or the Restart timer runs out).



Example PPP connection

```
▪ rcvd [LCP ConfAck id=0x2 <mru 1492> <magic
  0x1f3f807b>]
▪ sent [LCP EchoReq id=0x0 magic=0x1f3f807b]
▪ rcvd [CHAP Challenge id=0x1
  <1a5c7a4446bf4a51bc15b170dfbb66ae>, name =
  "BSGRE102"]
▪ ChapReceiveChallenge: rcvd type CHAP-DIGEST-MD5
▪ sent [CHAP Response id=0x1
  <2c99f036ee3190e0364d6bad10e4b971>, name =
  "monlogin@isp"]
▪ rcvd [LCP EchoRep id=0x0 magic=0x4b4dcf02]
▪ rcvd [LCP ConfReq id=0x12 <auth pap> <magic
  0x794cb2ad>]
▪ sent [LCP ConfReq id=0x3 <mru 1492> <asyncmap 0x0>
  <magic 0x5893ae1e> <pcomp> <accomp>]
▪ lcp_reqci: returning CONFACK.
```



Example PPP connection

- sent [LCP ConfAck id=0x12 <auth pap> <magic 0x794cb2ad>]
- rcvd [LCP ConfNak id=0x3 <mru 1500>]
- sent [LCP ConfReq id=0x4 <asyncmap 0x0> <magic 0x5893ae1e> <pcomp> <accomp>]
- rcvd [LCP ConfAck id=0x4 <asyncmap 0x0> <magic 0x5893ae1e> <pcomp> <accomp>]
- sent [LCP EchoReq id=0x0 magic=0x5893ae1e]
- sent [PAP AuthReq id=0x1 user="monlogin@isp" password=<hidden>]
- rcvd [LCP EchoRep id=0x0 magic=0x794cb2ad]
- rcvd [PAP AuthAck id=0x1 ""]
- sent [IPCP ConfReq id=0x1 <addr 0.0.0.0> <compress VJ 0f 01>]
- rcvd [IPCP ConfReq id=0x48 <addr 192.168.254.254>]
- ipcp: returning Configure-ACK

24

<compress VJ 0f 01> - Van Jacobson header compression

0f - number of connections that may have their headers compressed

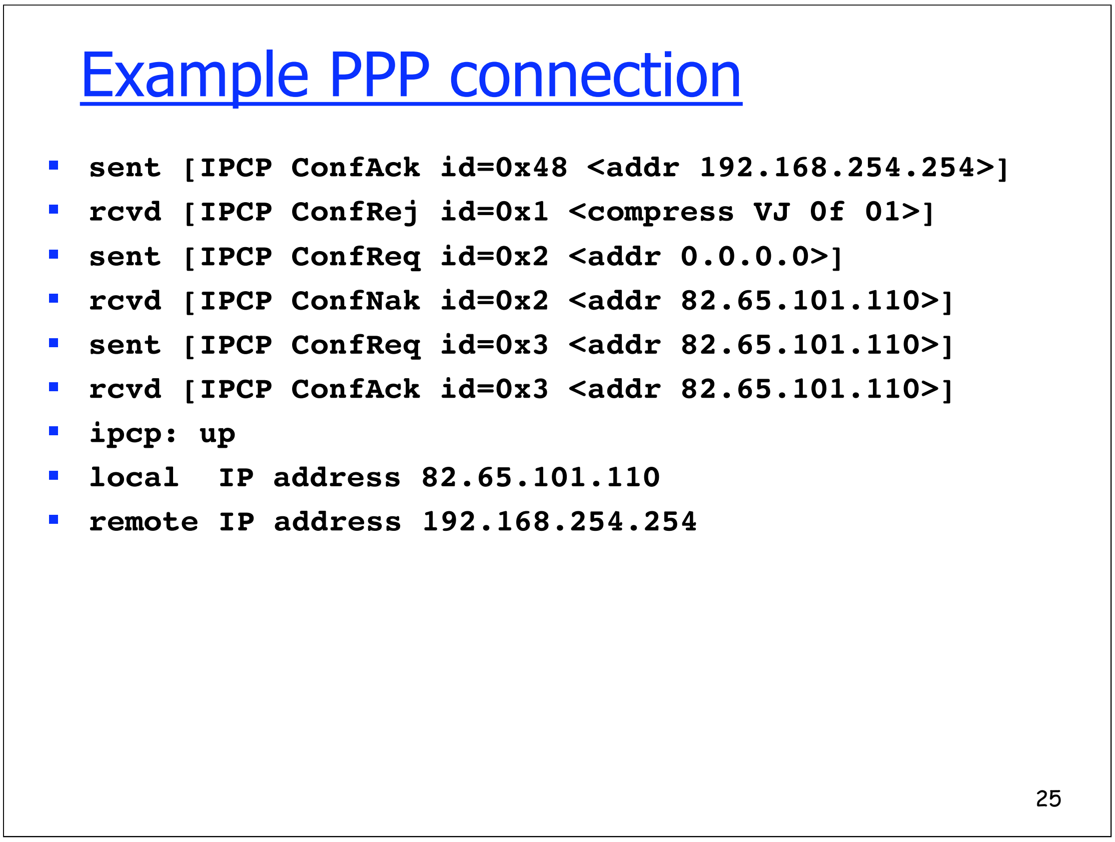
01 - if 0 - the connection id (in the compressed header) needs to be always present, if 1 - not present

PPP protocol field indicates whether a frame contains a compressed packet or not:

-0x0021 - UDP, SYN, FIN, RST

-0x002F - TCP, no compression

-0x002d - TCP, compression active>



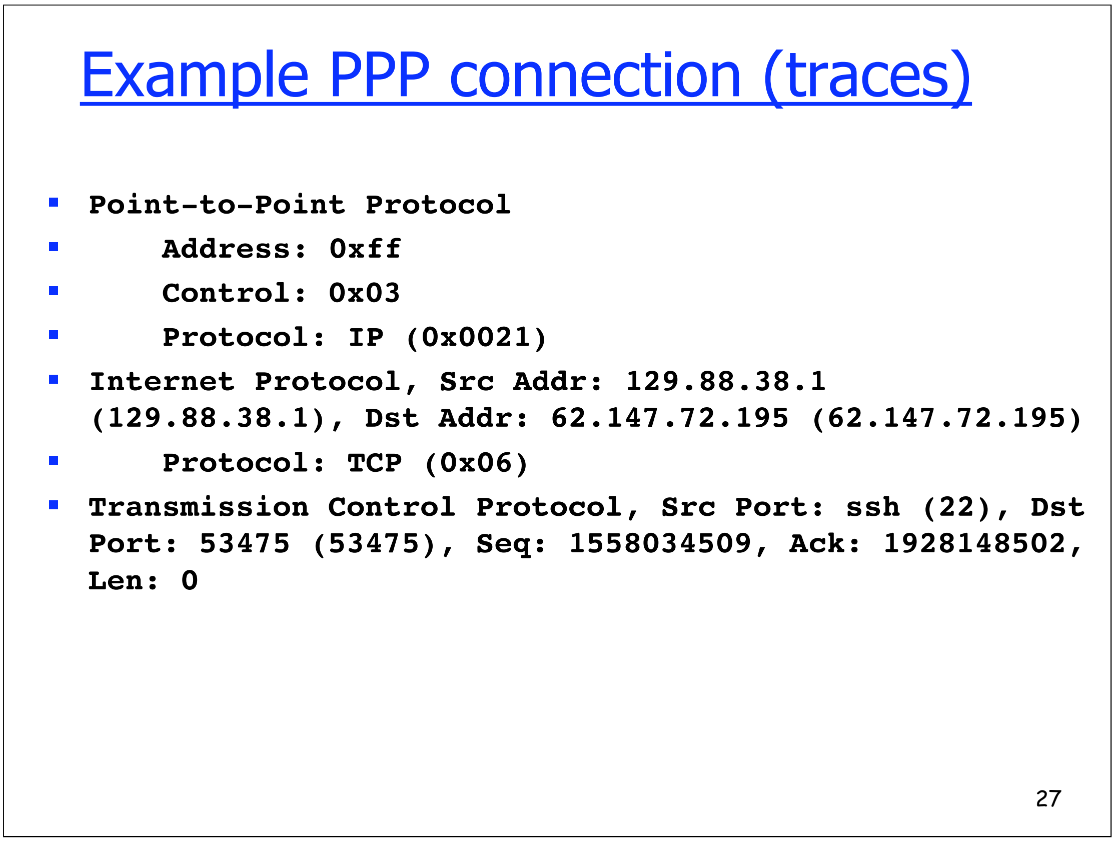
Example PPP connection

```
sent [IPCP ConfAck id=0x48 <addr 192.168.254.254>]
rcvd [IPCP ConfRej id=0x1 <compress VJ 0f 01>]
sent [IPCP ConfReq id=0x2 <addr 0.0.0.0>]
rcvd [IPCP ConfNak id=0x2 <addr 82.65.101.110>]
sent [IPCP ConfReq id=0x3 <addr 82.65.101.110>]
rcvd [IPCP ConfAck id=0x3 <addr 82.65.101.110>]
ipcp: up
local  IP address 82.65.101.110
remote IP address 192.168.254.254
```

local IP address 62.147.191.198
remote IP address 192.168.254.254
primary   DNS address 213.228.0.168
secondary DNS address 212.27.32.5

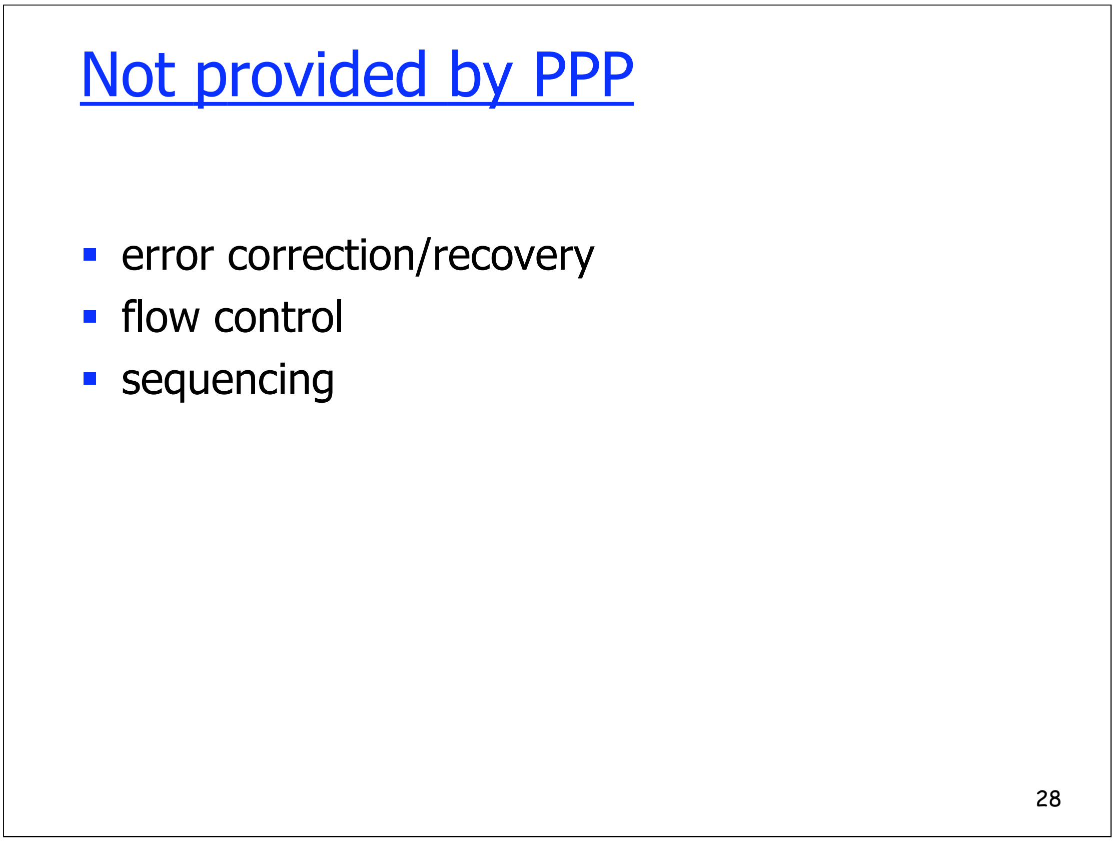
Example PPP connection (traces)

- Point-to-Point Protocol
- Address: 0xff
- Control: 0x03
- Protocol: IP (0x0021)
- Internet Protocol, Src Addr: 62.147.72.195 (62.147.72.195), Dst Addr: 129.88.38.1 (129.88.38.1)
- Protocol: TCP (0x06)
- Transmission Control Protocol, Src Port: 53475 (53475), Dst Port: ssh (22), Seq: 1928148501, Ack: 0, Len: 0



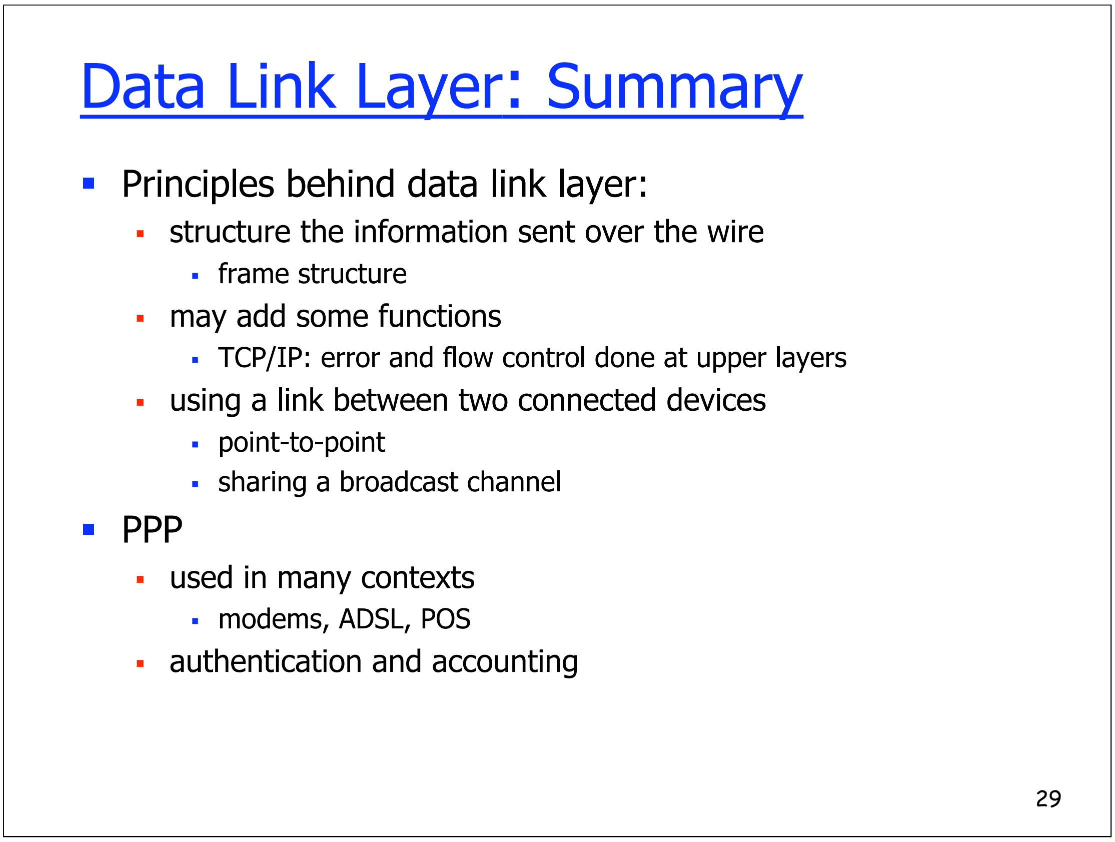
Example PPP connection (traces)

```
Point-to-Point Protocol
    Address: 0xff
    Control: 0x03
    Protocol: IP (0x0021)
Internet Protocol, Src Addr: 129.88.38.1
(129.88.38.1), Dst Addr: 62.147.72.195 (62.147.72.195)
    Protocol: TCP (0x06)
Transmission Control Protocol, Src Port: ssh (22), Dst
Port: 53475 (53475), Seq: 1558034509, Ack: 1928148502,
Len: 0
```



Not provided by PPP

- error correction/recovery
- flow control
- sequencing



Data Link Layer: Summary

- Principles behind data link layer:
    - structure the information sent over the wire
        - frame structure
    - may add some functions
        - TCP/IP: error and flow control done at upper layers
    - using a link between two connected devices
        - point-to-point
        - sharing a broadcast channel
- PPP
    - used in many contexts
        - modems, ADSL, POS
    - authentication and accounting