Master Course
Computer Networks
IN2097

Prof. Dr.-Ing. Georg Carle
Christian Grothoff, Ph.D.
Stephan Günther

Chair for Network Architectures and Services
Department of Computer Science
Technische Universität München
http://www.net.in.tum.de

Connection-Oriented Networking

ATM Networks

Connection-Oriented Network Issues

- Network Service Model

- Virtual Circuits
  - Addresses vs. labels
  - Address lookup vs. label lookup

- Connection / flow state in nodes

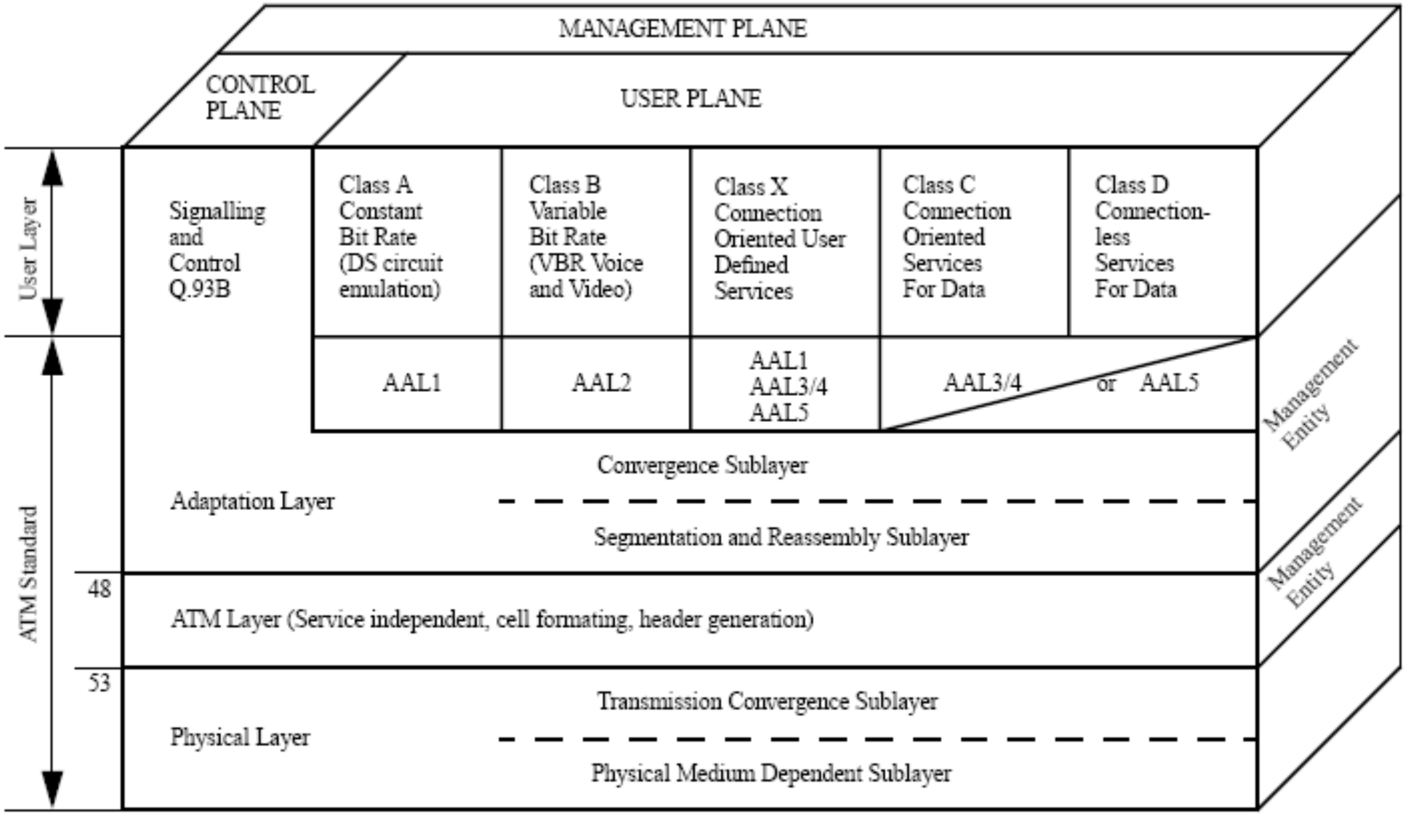
- Quality-of-Service (QoS) properties for flows

Network Architectures

Link virtualization: ATM

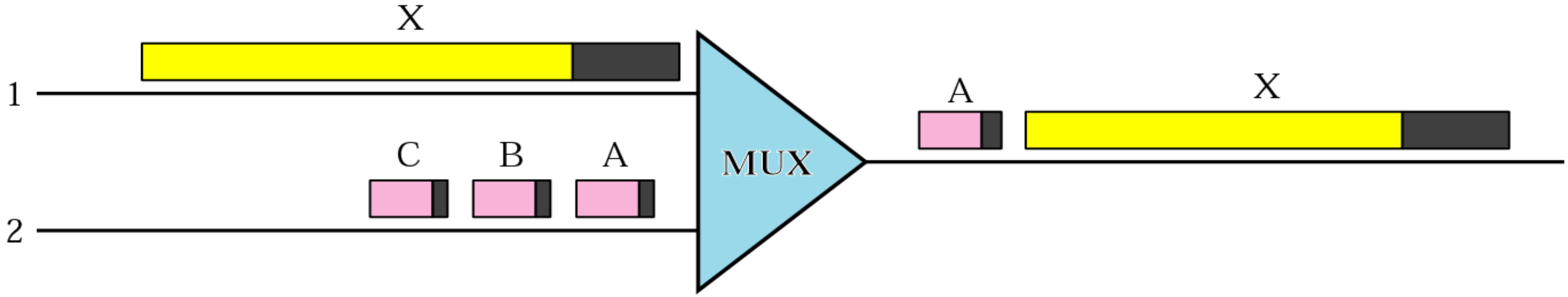
ATM Layer Model

- User plane: information flow between the layers
- Control plane: connection setup, maintenance and termination
- Management plane: meta-signaling and OAM (Operation and Maintenance) information flow

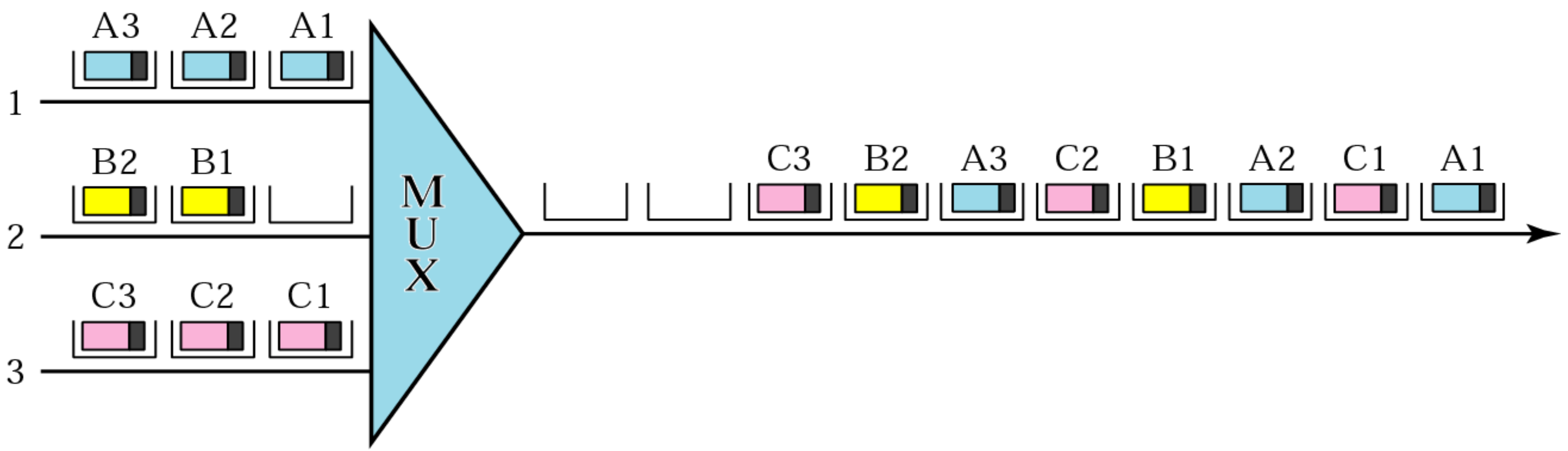


Multiplexing of Variable vs. Fixed Size Packets

- Multiplexing of variable size packets

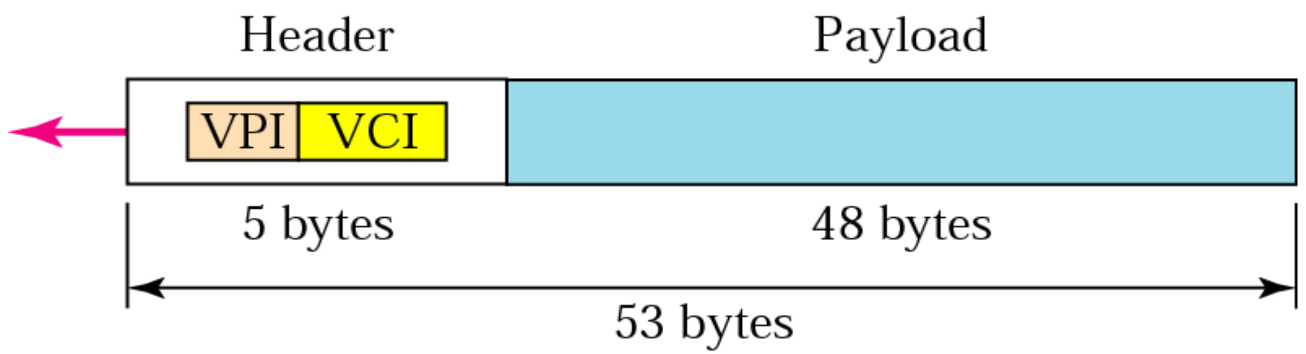


- ATM Multiplexing

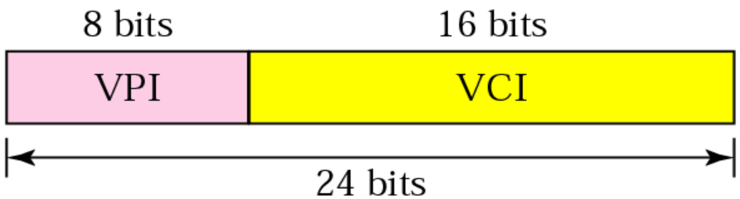


ATM Identifiers

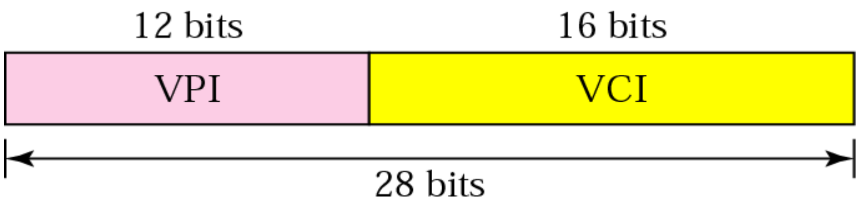
- ATM Cell



- Virtual Path Identifiers and Virtual Channel Identifiers



a. VPI and VCI in a UNI



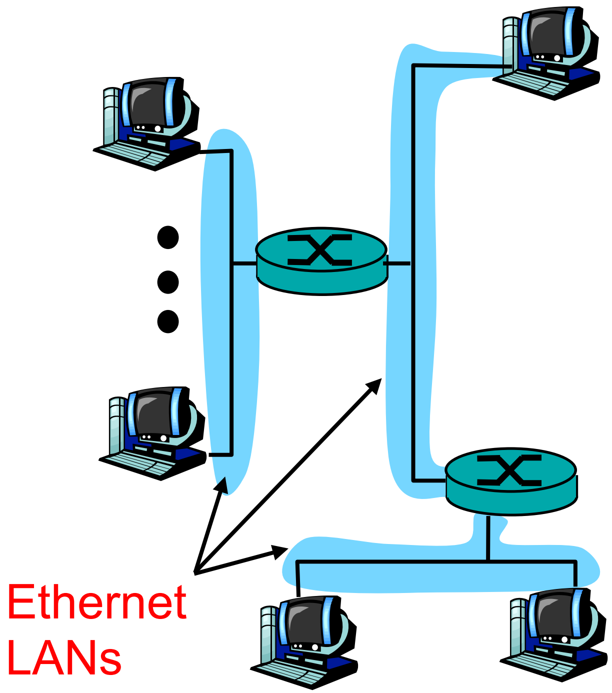
b. VPI and VCI in an NNI

(UNI: User-to-Network-Interface
NNI: Network-to-Network-Interface)

IP-Over-ATM

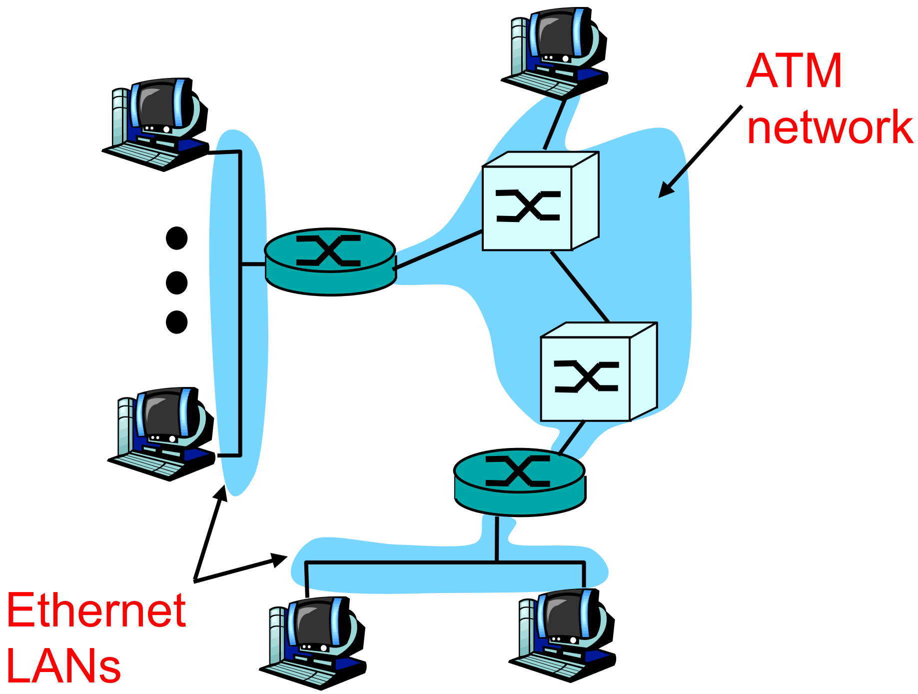
IP only

- 3 “networks” (e.g., LAN segments)
- MAC (802.3) and IP addresses

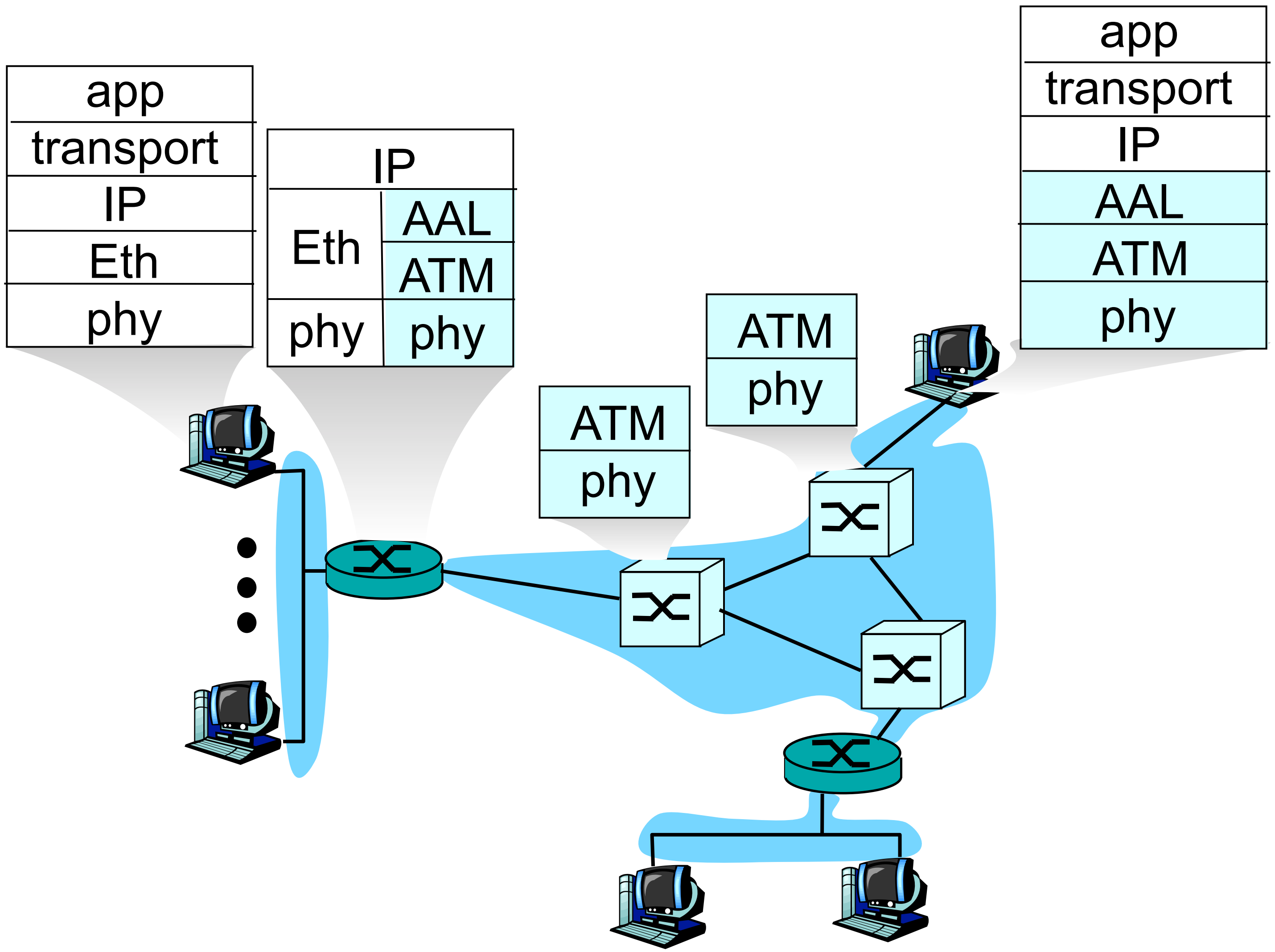


IP over ATM

- replace “network” (e.g., LAN segment) with ATM network
- ATM addresses, IP addresses



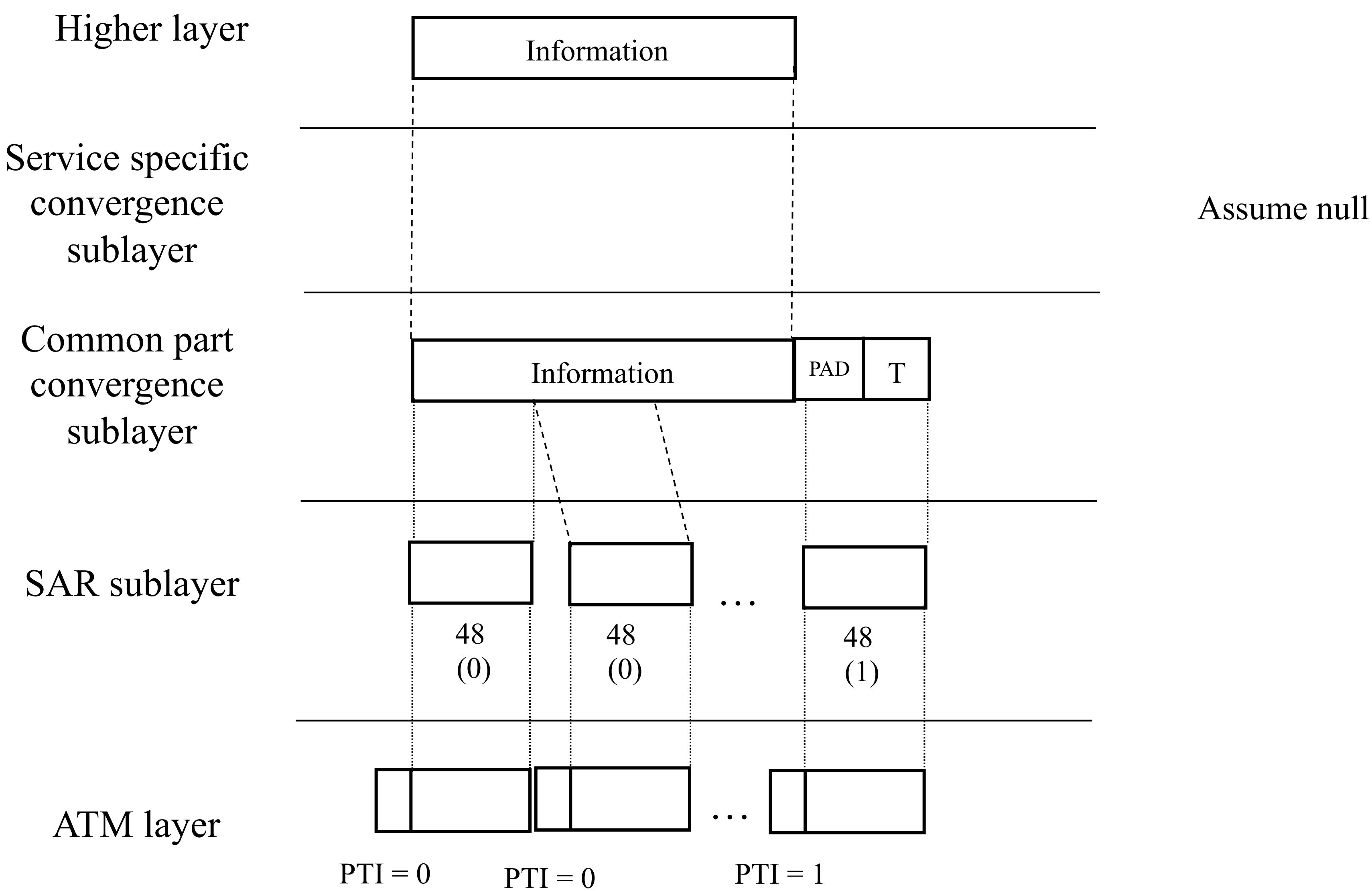
IP-Over-ATM


app
transport
IP
Eth
phy
IP
Eth
AAL
ATM
phy
phy
ATM
phy
ATM
phy
app
transport
IP
AAL
ATM
phy

Datagram Journey in IP-over-ATM Network

- at Source Host:
  - IP layer maps between IP, ATM destination address (using ARP)
  - passes datagram to AAL5
  - AAL5 encapsulates data, segments cells, passes to ATM layer
- ATM network: moves cell along VC to destination
- at Destination Host:
  - AAL5 reassembles cells into original datagram
  - if CRC OK, datagram is passed to IP

AAL 5 Layering



AAL 5 Protocol

- AAL5 is a **s**imple and **e**fficient **AAL** (“SEAL”)
  - performs subset of the functions of AAL3/4
- CPCS-PDU payload length can be up to 65.535 octets
  - must use PAD (0 to 47 octets) to align CPCS-PDU length to a multiple of 48 octets

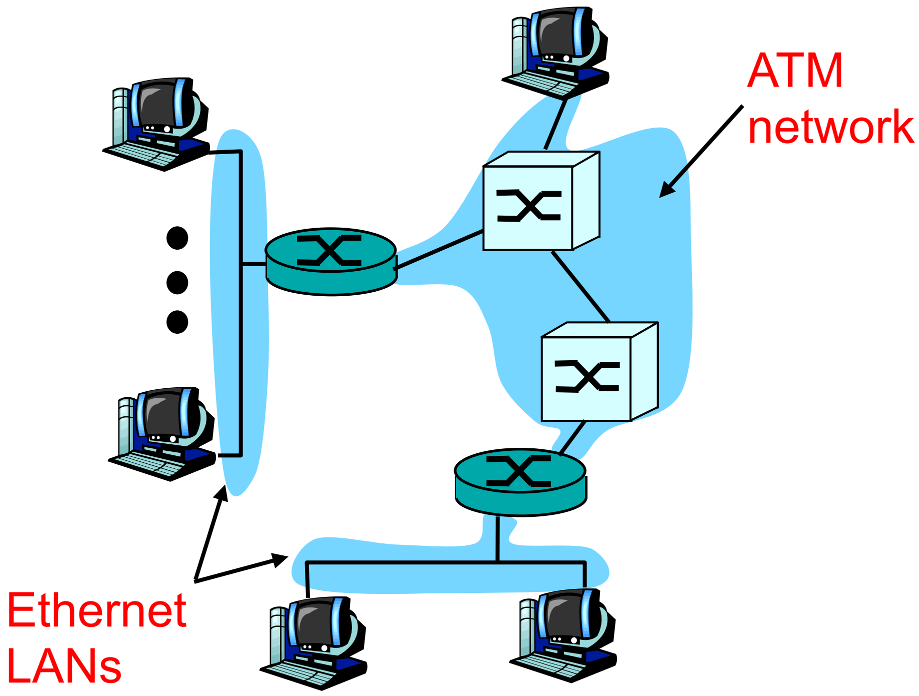
	0 - 47	1	1	2	4
CPCS-PDU Payload	PAD	CPCS UU	CPI	Length	CRC-32

PAD	Padding
CPCS-UU	CPCS User-to-User Indicator
CPI	Common Part Indicator
Length	CPCS-PDU Payload Length
CRC-32	Cyclic Redundancy Check

IP-Over-ATM

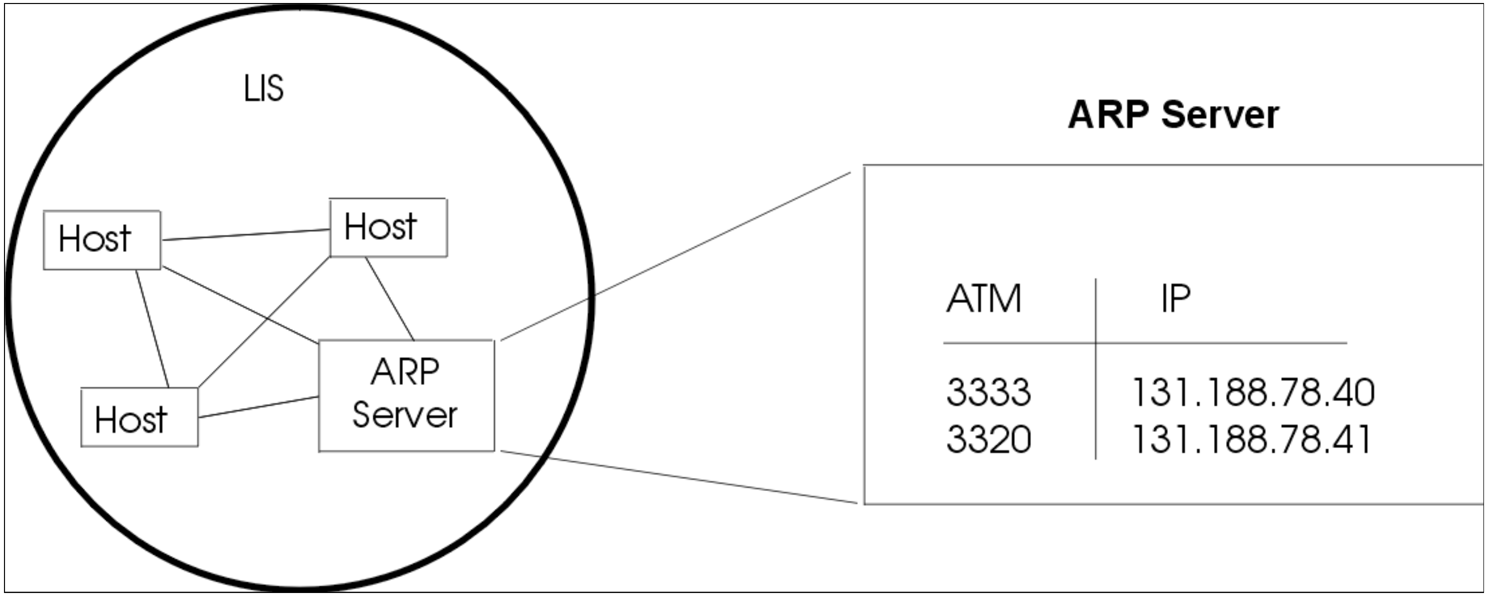
Issues:

- IP datagrams into ATM AAL5 PDUs
- from IP addresses to ATM addresses
  - just like IP addresses to 802.3 MAC addresses!
  - ARP server



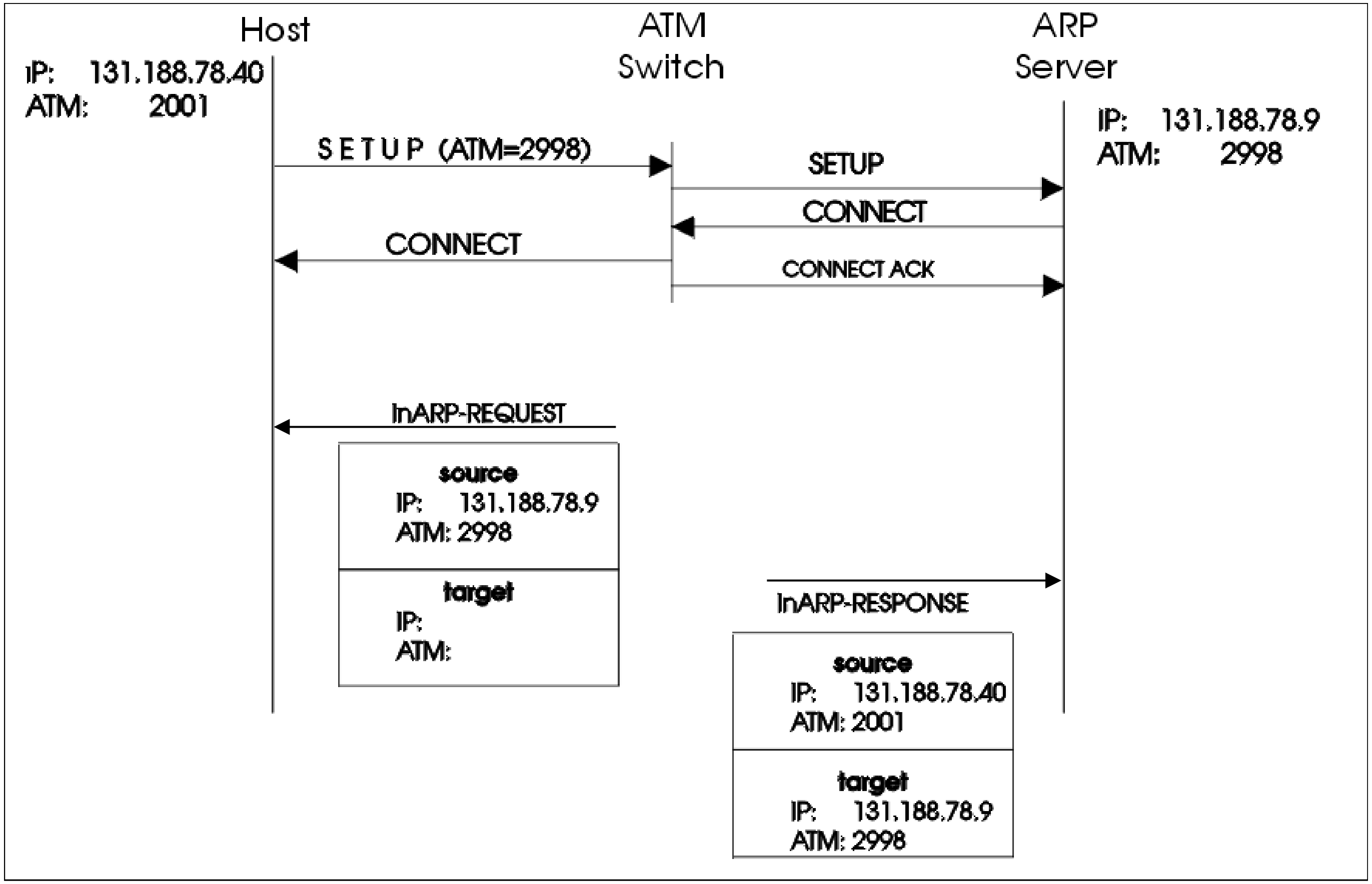
Classical IP and ARP over ATM (CLIP)

- RFC 1577
- Suitable for ATM unicast communication
- Encapsulation of IP packets into AAL PDUs
- Support for large MTU sizes
- There must be an ATMARP server in each LIS (Logical IP Subnet)



Classical IP and ARP over ATM (CLIP)

- The host registers its IP/ATM address information at the ATMARP server using the InARP protocol



Classical IP and ARP over ATM (CLIP)

- RFC 1577: Classical IP and ARP over ATM
- ATMARP Server Operational Requirements
  - The ATMARP server, upon the completion of an ATM call/ connection of a new VC, transmits InATMARP * request to determine the IP address of the client
  - InATMARP reply from client contains information necessary for ATMARP Server to build ATMARP table cache
  - This information is used to generate replies to the ATMARP requests it receives

* InATMARP is the same protocol as the original InARP protocol presented in RFC 1293 but applied to ATM networks: Discover the protocol address of a station associated with a virtual circuit.

RFC 1293: Bradely, T., and C. Brown, "Inverse Address Resolution Protocol“, January 1992.

Classical IP and ARP over ATM (CLIP)

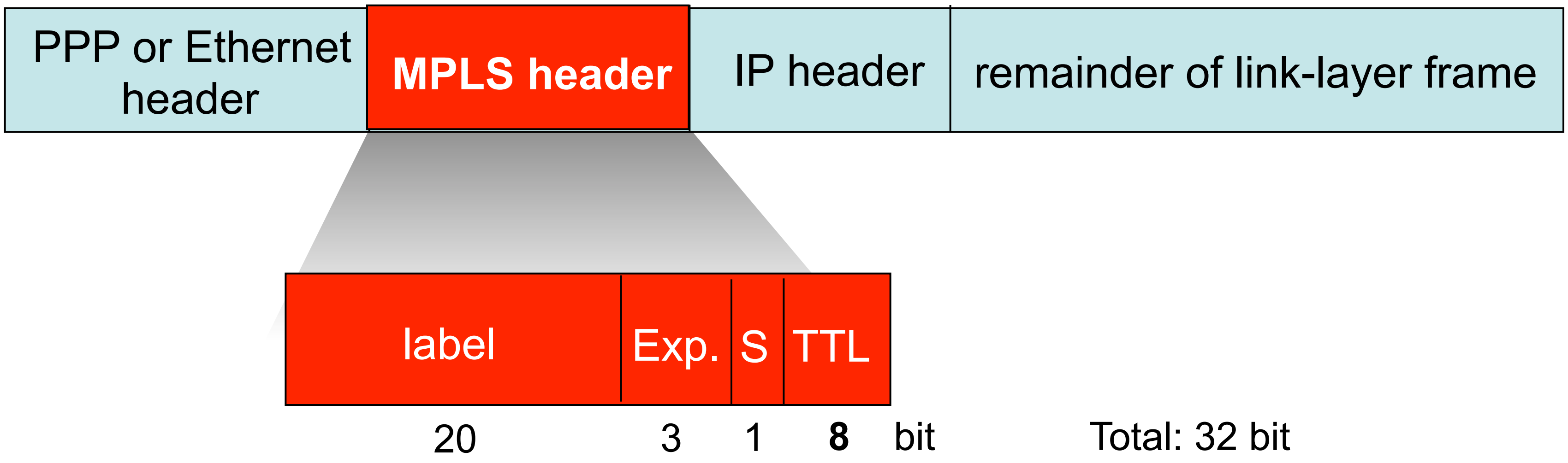
- RFC 1577: Classical IP and ARP over ATM
- ATMARP Client Operational Requirements
  1. Initiate the VC connection to the ATMARP server for transmitting and receiving ATMARP and InATMARP packets.
  2. Respond to ARP_REQUEST and InARP_REQUEST packets received on any VC appropriately.
  3. Generate and transmit ARP_REQUEST packets to the ATMARP server and to process ARP_REPLY appropriately. ARP_REPLY packets should be used to build/refresh its own client ATMARP table entries.
  4. Generate and transmit InARP_REQUEST packets as needed and to process InARP_REPLY packets appropriately. InARP_REPLY packets should be used to build/refresh its own client ATMARP table entries.
  5. Provide an ATMARP table aging function to remove own old client ATMARP tables entries after a period of time.

MPLS

Multi-Protocol Label Switching

Multiprotocol label switching (MPLS)

- Initial goal: speed up IP forwarding by using fixed length label (instead of IP address) to do forwarding
  - borrowing ideas from Virtual Circuit (VC) approach
  - IP datagram still keeps IP address
  - RFC 3032 defines MPLS header
    - Label: has role of Virtual Circuit Identifier
    - Exp: experimental usage, may specify Class of Service (CoS)
    - S: Bottom of Stack - end of series of stacked headers
    - TTL: time to live



Multiprotocol label switching (MPLS)

- RFC 3270: Le Faucheur, F., Wu, L., Davie, B., Davari, S., Vaananen, P., Krishnan, R., Cheval, P. and J. Heinanen, “Multi-Protocol Label Switching (MPLS) Support of Differentiated Services”, May 2002.
  - EXP: 3 bits - this field contains the value of the EXP field for the EXP<->PHB (Per-Hop-Behaviour) mapping
  - Mapping transported via signaling protocol
- RFC 3140: Black, D., Brim, S., Carpenter, B. and F. Le Faucheur, "Per Hop Behavior Identification Codes", June 2001.
  - Case 1: PHBs defined by standards action, as per [RFC 2474]. PHB is recommended 6-bit DSCP value for that PHB, left-justified in a 16 bit field, with bits 6 through 15 set to zero.
  - Case 2: PHBs not defined by standards action, i.e., experimental or local use PHBs In this case an arbitrary 12 bit PHB-ID is placed left-justified in the a bit field.
    Bit 15 is set to 1, Bits 12 and 13 are zero.

MPLS TTL Processing

c.f. RFC 3032 - MPLS Label Stack Encoding

- Protocol-independent rules
  - "outgoing TTL" of a labeled packet is either
    a) one less than the incoming TTL, or b) zero.
  - Packets with TTL=0 are discarded
- IP-dependent rules
  - When an IP packet is first labeled, the TTL field of the label stack is set to the value of the IP TTL field.
  - If the IP TTL field needs to be decremented, as part of the IP processing, it is assumed that this has already been done.
  - When a label is popped, and the resulting label stack is empty, then the value of the IP TTL field SHOULD BE replaced with the outgoing MPLS TTL value.
  - A network administration may prefer to decrement the IPv4 TTL by one as it traverses an MPLS domain.

ICMP

- When a router receives an IP datagram that it can't forward, it sends an ICMP message to the datagram's originator
- The ICMP message indicates why the datagram couldn't be delivered
  - E.g., Time Expired, Destination Unreachable
- The ICMP message also contains the IP header and at least leading 8 octets of the original datagram
  - RFC 1812 - Requirements for IP Version 4 Routers extends this to "as many bytes as possible"
  - Historically, every ICMP error message has included the Internet header and at least
  - Including only the first 8 data bytes of the datagram that triggered the error is no longer adequate, due to use e.g. of IP-in-IP tunneling

ICMP in presence of MPLS

- When an LSR receives an MPLS encapsulated datagram that it can’t deliver
  - It removes entire MPLS labels stack
  - It sends an ICMP message to datagram’s originator
- The ICMP message indicates why the datagram couldn’t be delivered (e.g., time expired, destination unreachable)
- The ICMP message also contains the IP header and leading 8 octets of the original datagram
  - RFC 1812 extends this to “as many bytes as possible”

ICMP in Presence of MPLS

Issue

- The ICMP message contains no information regarding the MPLS stack that encapsulated the datagram when it arrived at the LSR
- This is a significant omission because:
  - The LSR tried to forward the datagram based upon that label stack
  - Resulting ICMP message may be confusing

Why?

ICMP in Presence of MPLS

Issue

- ICMP Destination Unreachable
  - Message contains IP header of original datagram
  - Router sending ICMP message has an IP route to the original datagram's destination
  - Original datagram couldn't be delivered because MPLS forwarding path was broken
- ICMP Time Expired
  - Message contains IP header of original datagram
  - TTL value in IP header is greater than 1
  - TTL expired on MPLS header. ICMP Message contains IP header of original datagram

ICMP with MPLS

c.f. RFC 4950 - ICMP Extensions for Multiprotocol Label Switching

- defines an ICMP extension object that permits an LSR to append MPLS information to ICMP messages.
- ICMP messages include the MPLS label stack, as it arrived at the router that is sending the ICMP message.
- equally applicable to ICMPv4 [RFC792] and ICMPv6 [RFC4443]
- sample output from an enhanced TRACEROUTE:

```
> traceroute 192.0.2.1
traceroute to 192.0.2.1 (192.0.2.1), 30 hops max, 40 byte packets
1 192.0.2.13 (192.0.2.13) 0.661 ms 0.618 ms 0.579 ms
2 192.0.2.9 (192.0.2.9) 0.861 ms 0.718 ms 0.679 ms
   MPLS Label=100048 Exp=0 TTL=1 S=1
3 192.0.2.5 (192.0.2.5) 0.822 ms 0.731 ms 0.708 ms
   MPLS Label=100016 Exp=0 TTL=1 S=1
4 192.0.2.1 (192.0.2.1) 0.961 ms 8.676 ms 0.875 ms
```

- MPLS Label Stack Object: can be appended to ICMP Time Exceeded and Destination Unreachable messages.

```
      0                   1                   2                   3
+-------------+-------------+-------------+-------------+
|                    Label                |EXP |S|       TTL        |
+-------------+-------------+-------------+-------------+
|                                                               |
|          // Remaining MPLS Label Stack Entries //              |
|                                                               |
+-------------+-------------+-------------+-------------+
```

- Must be preceded by an ICMP Extension Structure Header and an ICMP Object Header, defined in [RFC4884].

Multi-Part ICMP Messages - RFC 4884

- ICMP Extension Structure may be appended to ICMP v4 / v6 Destination Unreachable and Time Exceeded messages
- ICMP Extension Structure Header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |Version|       (Reserved)          |           Checksum        |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

  ICMP extension version number: 2

- ICMP Object Header and Object Payload

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Length            |   Class-Num   |   C-Type      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   // (Object Payload) //                      |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MPLS for Linux

The work of James Leu:

https://sourceforge.net/projects/mpls-linux/

Discussions:

http://sourceforge.net/mailarchive/forum.php?forum_name=mpls-linux-devel

Bug fixes of Jorge Boncompte:

http://mpls-linux.git.sourceforge.net/git/gitweb.cgi?p=mpls-linux/net-next;a=shortlog;h=refs/heads/net-next-mpls

Additional bug fixes by Igor Maravić:

https://github.com/i-maravic/MPLS-Linux

https://github.com/i-maravic/iproute2

MPLS for Linux Labs

by Irina Dumitrascu and Adrian Popa: graduation project with purpose of teaching MPLS to university students, at Limburg Catholic University College

http://ontwerpen1.khlim.be/~lrutten/cursussen/comm2/mpls-linux-docs/

inlcudes e.g. Layer 2 VPN with MPLS, Layer 3 VPN with MPLS

Virtual Private Networks

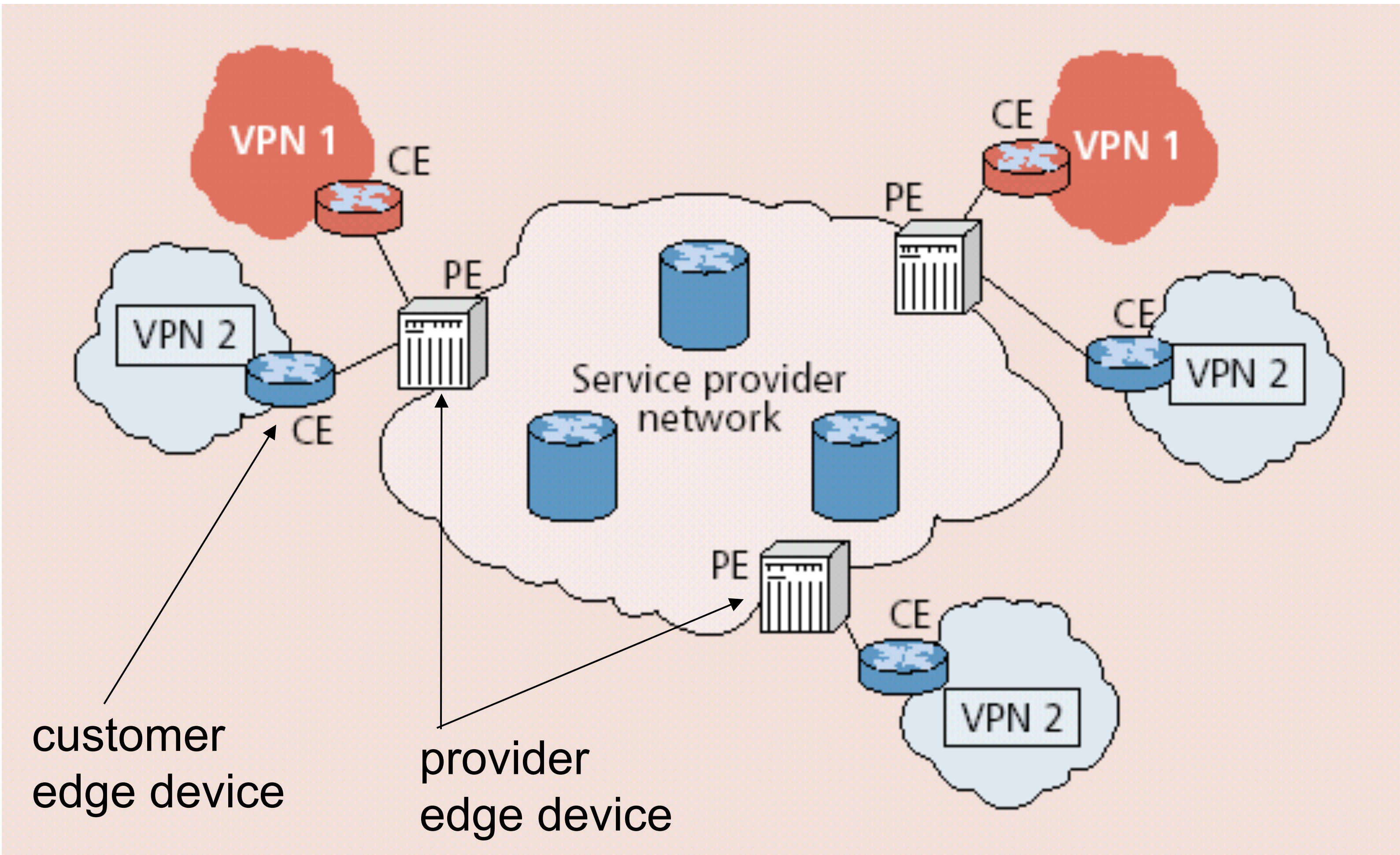
Virtual Private Networks (VPN)

VPNs

Networks perceived as being private networks by customers using them, but built over shared infrastructure owned by service provider (SP)

- Service provider infrastructure:
  - backbone
  - provider edge devices
- Customer:
  - customer edge devices
    (communicating over shared backbone)

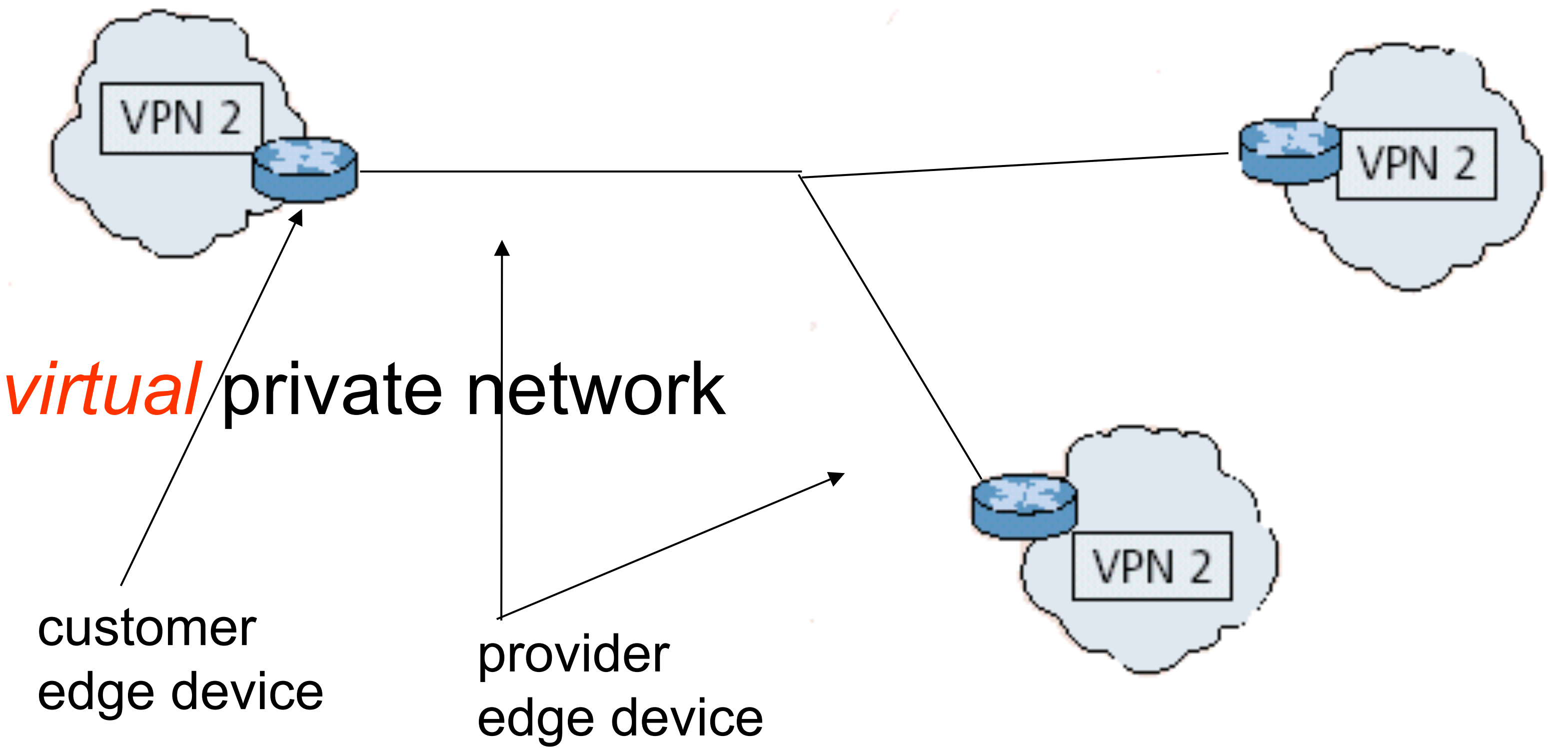
VPN Reference Architecture



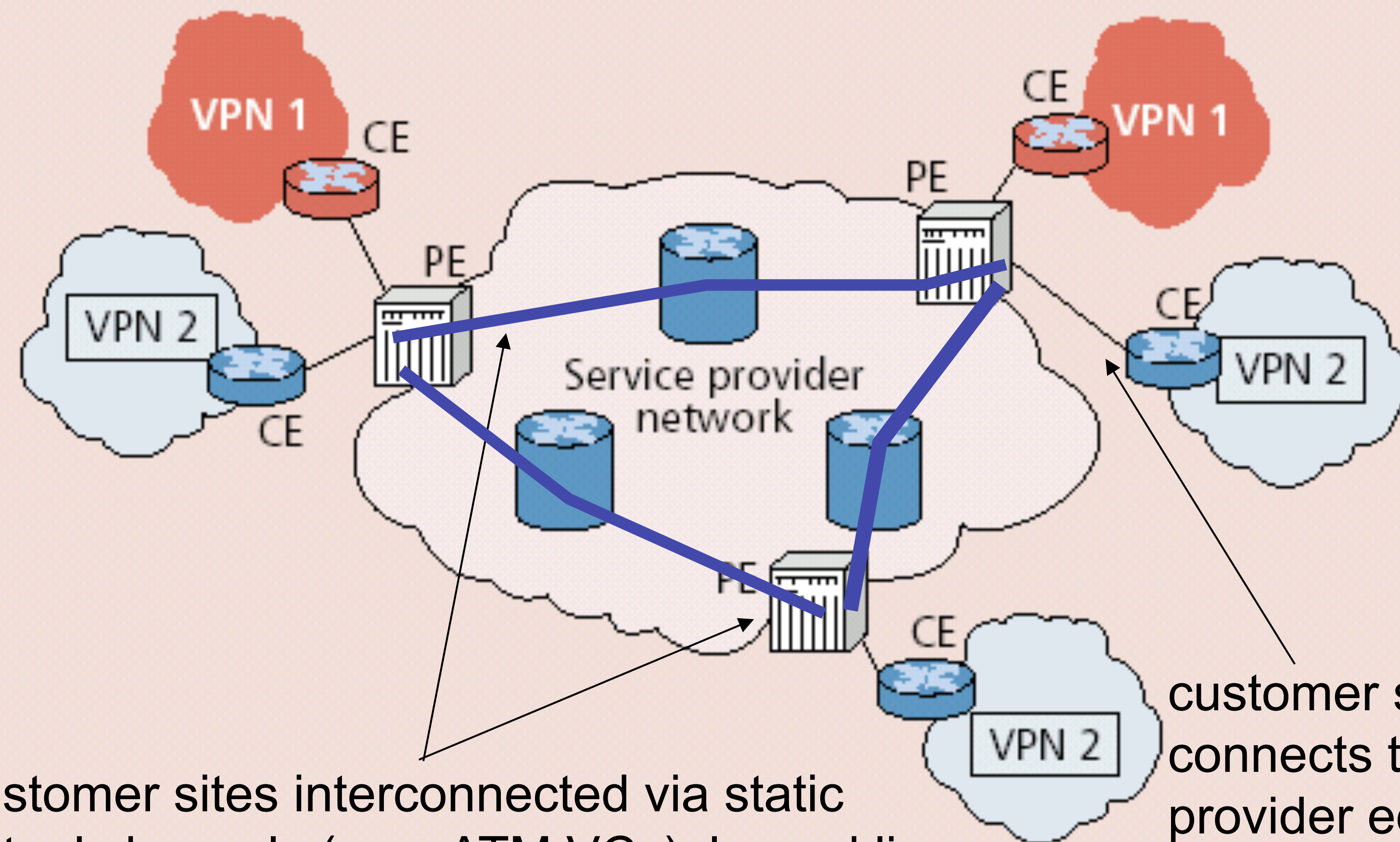
VPNs: Why?

- Privacy
- Security
- Works well with mobility (looks like you are always at home)
- Cost
  - many forms of newer VPNs are cheaper than leased line VPNs
  - ability to share at lower layers even though logically separate means lower cost
  - exploit multiple paths, redundancy, fault-recovery in lower layers
  - need isolation mechanisms to ensure resources shared appropriately
- Abstraction and manageability
  - all machines with addresses that are "in" are trusted no matter where they are

VPN: logical view



Leased-Line VPN

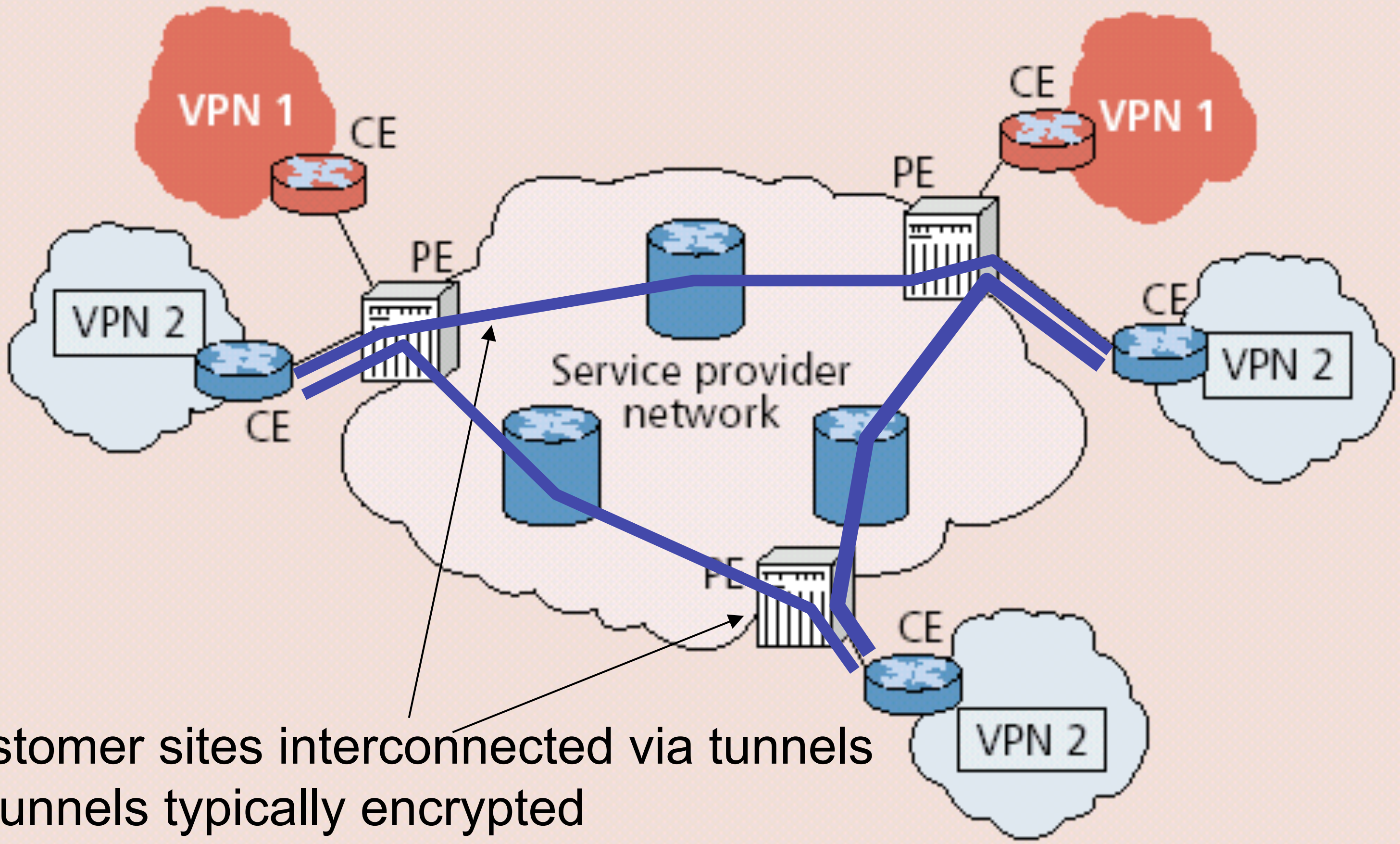


customer site connects to provider edge

customer sites interconnected via static virtual channels (e.g., ATM VCs), leased lines

Customer Premise VPN

- all VPN functions implemented by customer



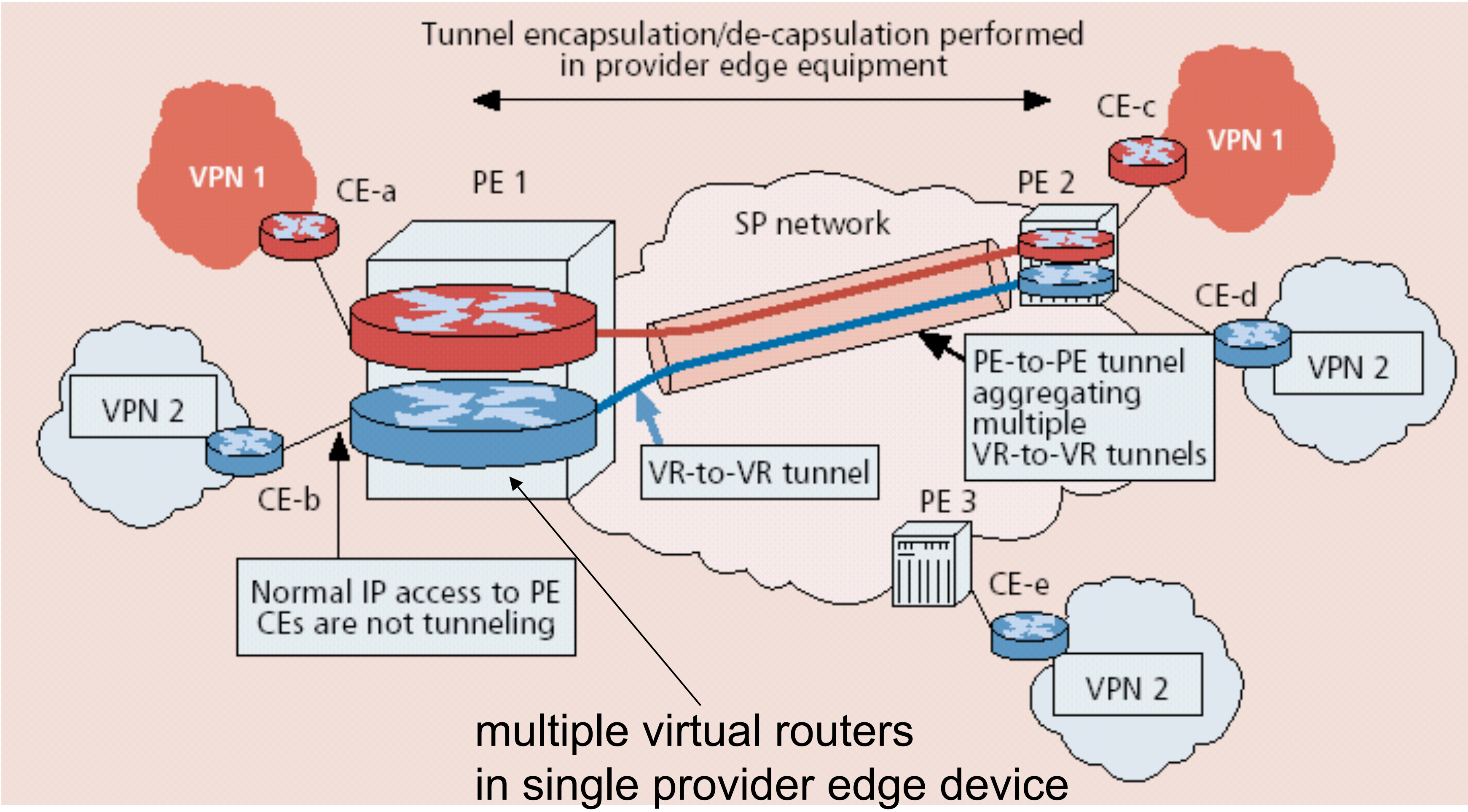
customer sites interconnected via tunnels

- tunnels typically encrypted
- Service provider treats VPN packets like all other packets

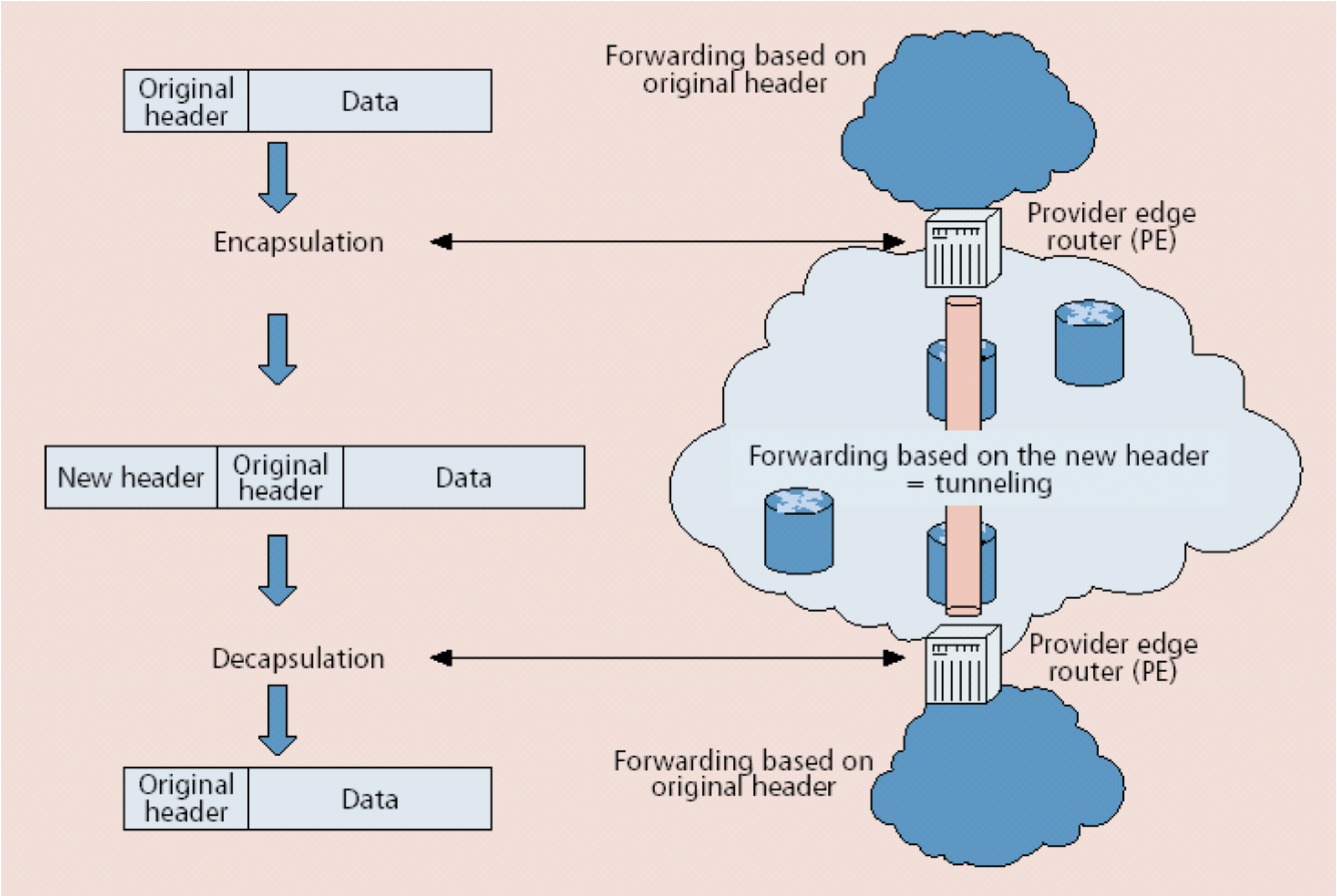
Variants of VPNs

- Leased-line VPN
  - configuration costs and maintenance by service provider: long time to set up, manpower
- CPE-based VPN
  - expertise by customer to acquire, configure, manage VPN
- Network-based VPN
  - Customer routers connect to service provider routers
  - Service provider routers maintain separate (independent) IP contexts for each VPN
    - sites can use private addressing
    - traffic from one VPN cannot be injected into another

Network-based Layer 3 VPNs



Tunneling


Original header
Data
Encapsulation
New header
Original header
Data
Decapsulation
Original header
Data
Forwarding based on original header
Provider edge router (PE)
Forwarding based on the new header = tunneling
Provider edge router (PE)
Forwarding based on original header

MPLS-based VPN