Master Course
Computer Networks
IN2097

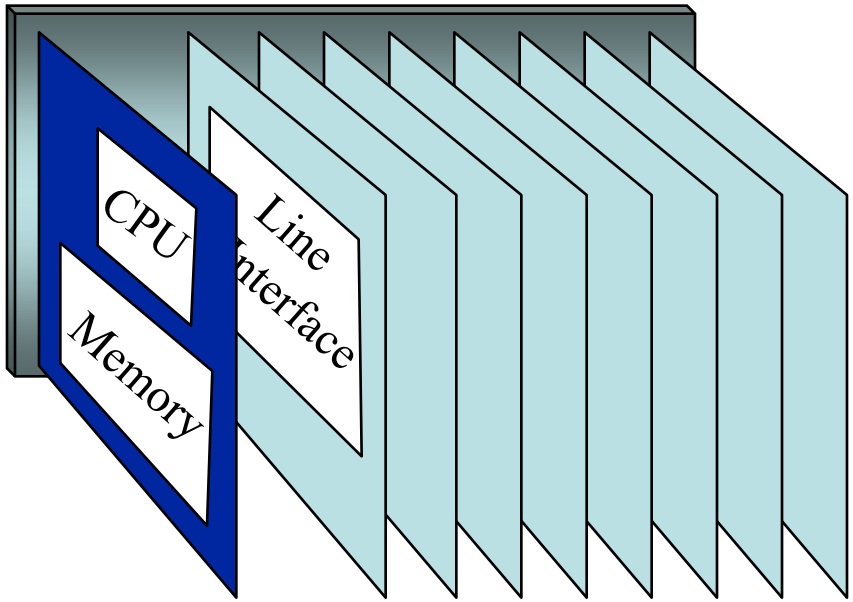
Prof. Dr.-Ing. Georg Carle

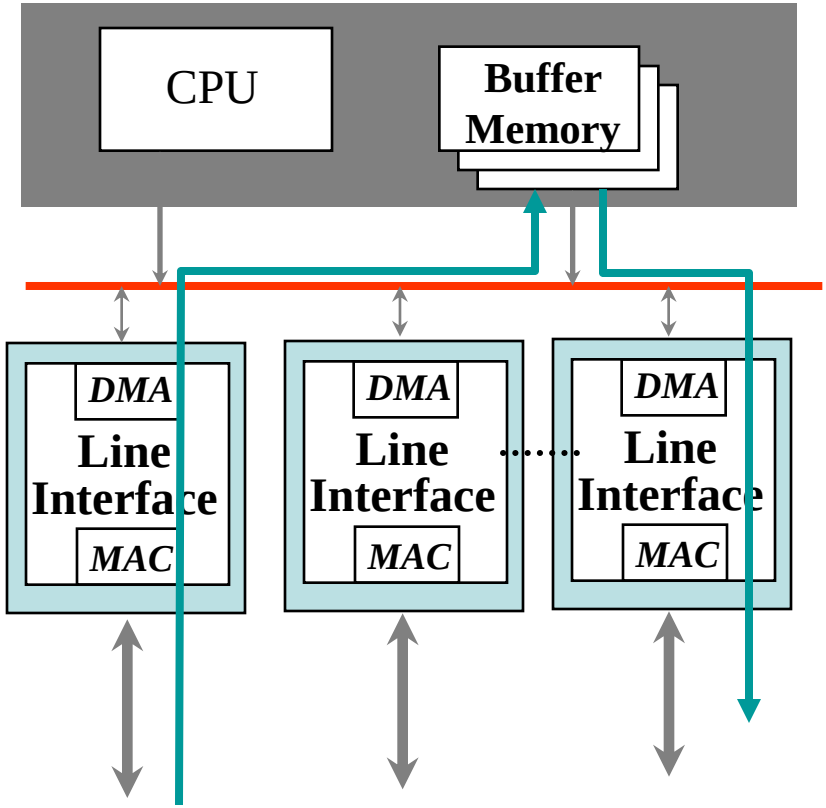
Dr. Nils Kammenhuber
Christian Grothoff, Ph.D.

Chair for Network Architectures and Services

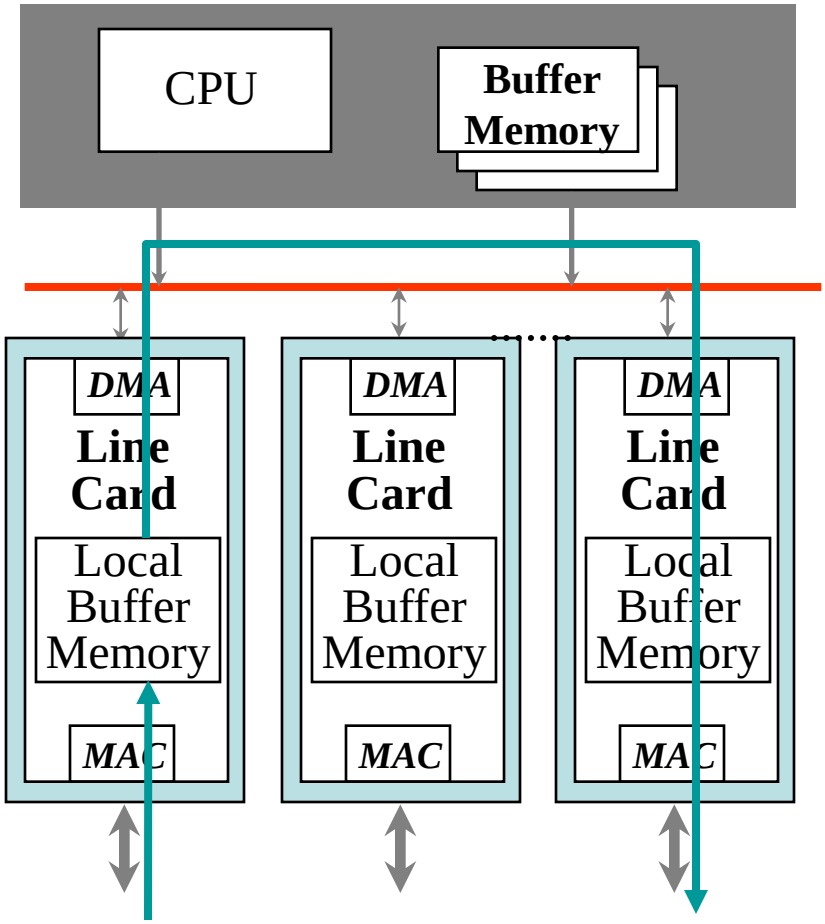
Institut für Informatik
Technische Universität München
http://www.net.in.tum.de

First-Generation IP Routers

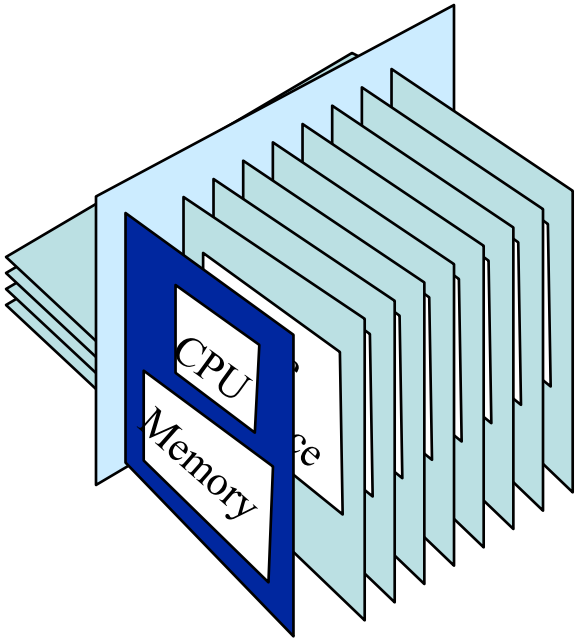


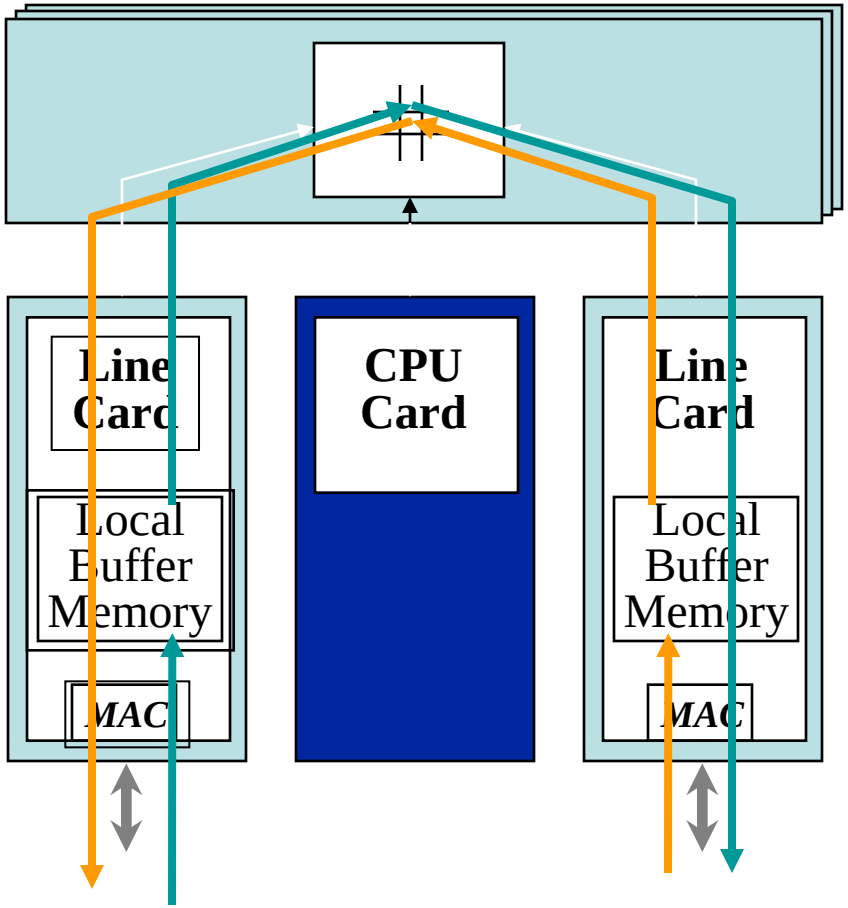


Second-Generation IP Routers



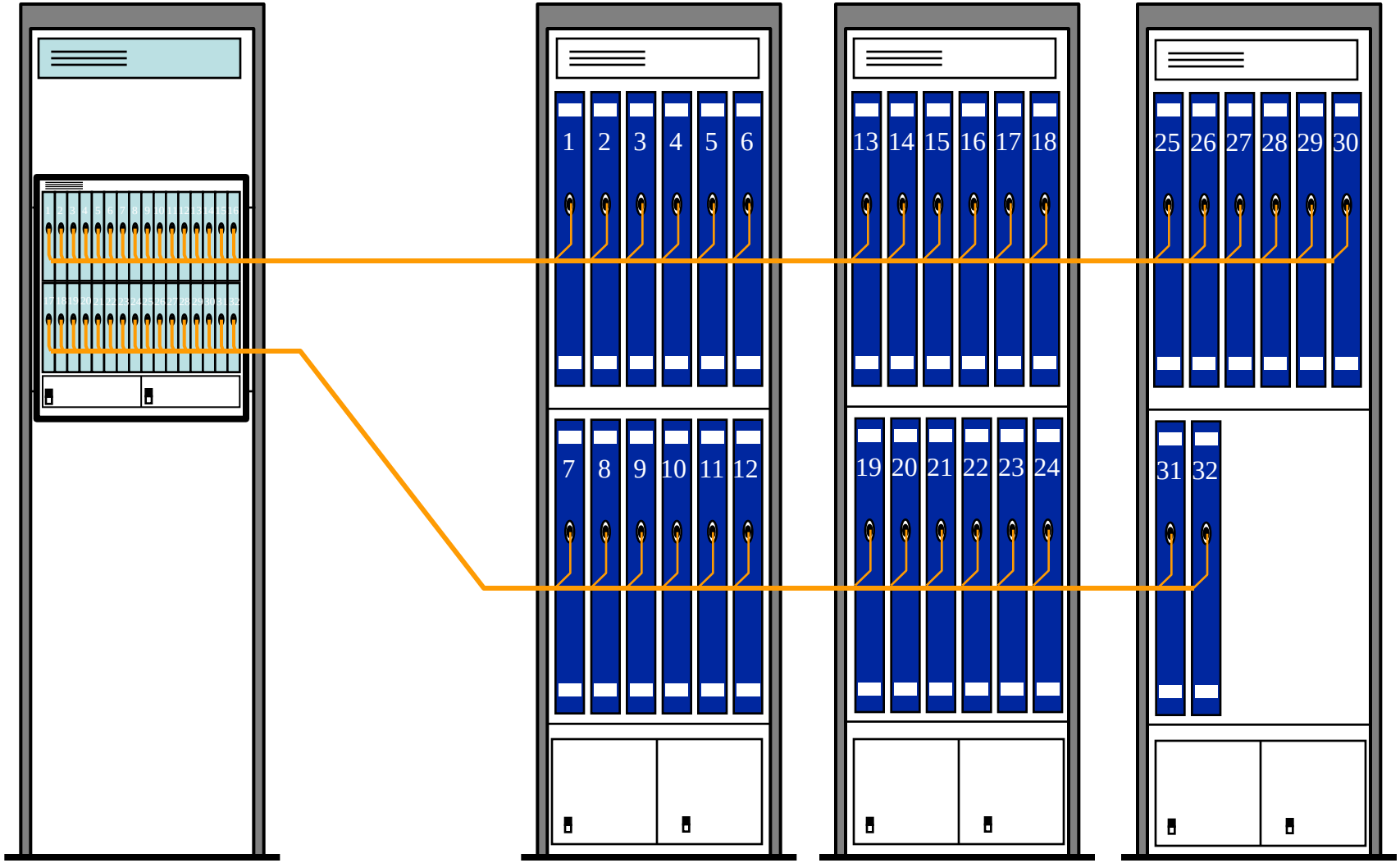
Third-Generation Switches/Routers





Fourth-Generation Switches/Routers

Clustering and Multistage



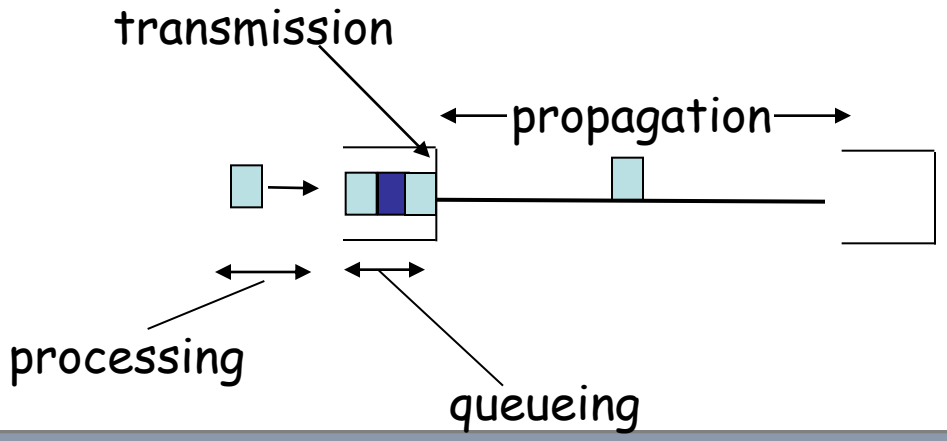
Background: Sources of packet delay

1. Processing delay:
   - Sending: prepare data for being transmitted
   - Receiving: interrupt handling

2. Queueing delay
   - Time waiting at output link for transmission
   - More congestion ➔ More queueing delay

3. Transmission delay:
   - R=link bandwidth (bps)
   - L=packet length (bits)
   - Time to send bits into link = L/R

4. Propagation delay:
   - d = length of physical link
   - s = propagation speed in medium (~$2 \cdot 10^8$ m/s)
   - Propagation delay = d/s



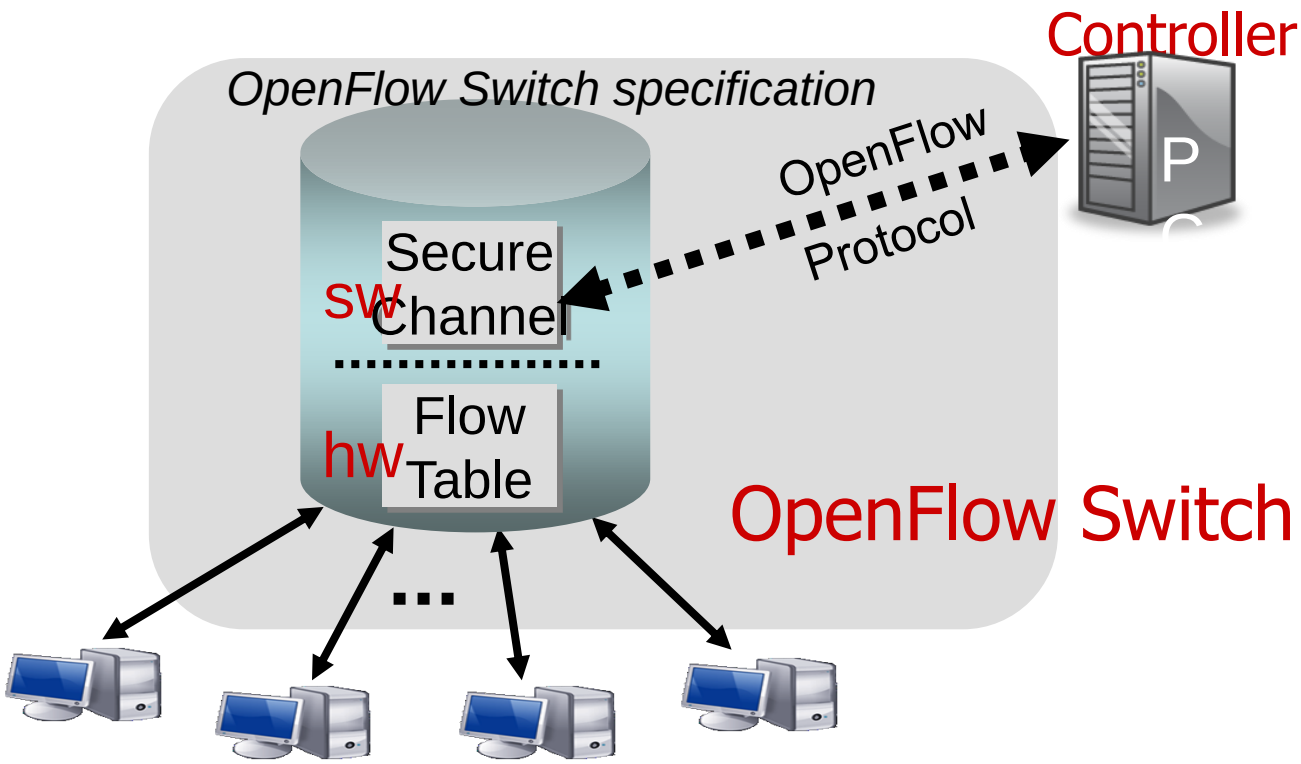
Impact Analysis: Advances in Network Technology

Data rate	Delay (1bit)	Length (1bit)	Delay (1kbyte)	Length (1kbyte)
10 Mbit/s	100 ns	20 m	0,8 ms	160 km
100 Mbit/s	10 ns	2 m	80 us	16 km
1 Gbit/s	1 ns	0,2 m	8 us	1600 m
10 Gbit/s	100 ps	0,02 m	0,8 us	160 m
40 Gbit/s	25 ps	0,005 m	0,2 us	40 m
100 Gbit/s	10 ps	0,002 m	80 ns	16 m

- Assessment
  - Transmission delay becomes less important
  - Distance (and # of RTTs!) becomes more important
    ⇨ Matters for communication beyond data center
  - Network adapter latency less important
    ⇨ Low-latency communication software becomes important

Advances in Switching

- Example: OpenFlow Switch architecture, Stanford University
- Concept: separation of switch fabric and switch control
- Allows for cheap switches, centrally controlled by switch manager
- ⇨ Assessment: suitable for low-latency data center communication



The Stanford Clean Slate Program http://cleanslate.stanford.edu

Advances by Virtualisation and Parallelization

- Disruptive Technologies:
  Virtualisation & Multicore Architectures **in the Network**
- Drivers for **virtualisation**

  1) Complexity
  - Virtualization allows to hide complexity if it is done right (Problem: right level of abstraction)

  2) New management principles
  - network management is a driver for virtualization
- What about **multicore** and **networking**?
  - In future there may be dozens, hundreds of cores
  - Need to **expose parallel processing**
  - Affects the way how to design protocols (?)
  - Need to provide ways to access flow state

Advances in Communication Software

- Example: Wire-speed packet capture and transmission
  - Important for…
    - Network research (traffic analysis)
    - Network security (intrusion detection systems)
  - Linux APIs:
    - PF_Ring: network socket for high-speed packet capturing
    - NAPI: New API – interrupt mitigation techniques for networking devices in the Linux kernel
    - TNAPI – Multithreaded NAPI
- Issues
  - How to make advances in packet capturing available for general purpose applications?
  - How to assess advances in communication software for other OSes?

Network Measurements

Acknowledgements:

The content of this chapter
is partly based on slides from:
Anja Feldmann, Constantine Dovrolis

Why do we measure the network?

- Network Provider View
  - Manage traffic
    - Predict future, model reality, plan network
    - Avoid bottlenecks in advance
  - Reduce cost
  - Accounting
- Client View
  - Get the best possible service
  - Check the service („Do I get what I've paid for?)
- Service Provider View
  - Get information about the client
  - Adjust service to demands
  - Reduce load on service
  - Accounting
- Researcher View
  - Performance evaluation (e.g., "could our new routing algorithm handle all this real-world traffic?")
- Security view
  - Detect malicious traffic, malicious hosts, malicious networks, …

But why should we do it at all?

- Do we really have to?
  - The network is well engineered
  - Well documented protocols, mechanisms, …
  - Everything built by humans ➔ no unknowns (compare this to, e.g., physics: String theory valid? Cosmic inflation phase sound? G.U.T.? etc.)
  - In theory, we can know everything that is going on

  ⇨ There should be no need for measurements

- But:
  - Moving target:
    - Requirements change
    - Growth, usage, structure changes
  - Highly interactive system
  - Heterogeneity in all directions
  - The total is more than the sum of its pieces

- And: The network is built, driven and used by humans
  - Detection of errors, misconfigurations, flaws, failures, misuse, …

Chapter: Network Measurements

- Introduction
- Architecture & Mechanisms
- Protocols
  - IPFIX (Netflow Accounting)
  - PSAMP (Packet Sampling)
- Scenarios

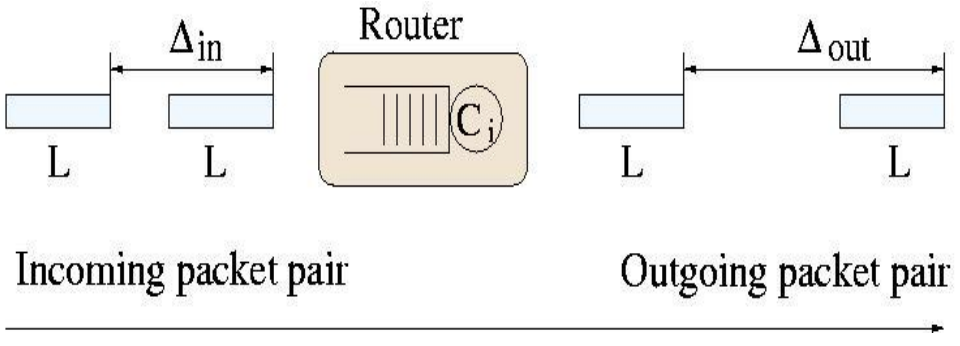
Network Measurements

- Active measurements
  - “intrusive”
  - Measurement traffic is generated and sent via the operational network.
    (Examples: ping, traceroute)
  - Advantages
    - Straightforward
    - Does not depend on existing traffic by active applications
    - Allows measurement of specific parts of the network
  - Disadvantages
    - Additional load
    - Network traffic is affected by the measurement
    - Measurements are influenced by (possibly varying) network load

Example: Packet pair probing

- Packet Pair (P-P) technique
  - Originally due to Jacobson & Keshav
- Send two equal-sized packets back-to-back
  - Packet size: L
  - Packet TX time at link i: L/C_i
- P-P dispersion = time interval between last bit of two packets
- Without any cross traffic, the dispersion at receiver is determined by bottleneck links (i.e., slowest link):

$$\Delta_{out} = \max\left(\Delta_{in}, \frac{L}{C_i}\right)$$



$$\Delta_R = \max_{i=1,\ldots,H}\left(\frac{L}{C_i}\right) = \frac{L}{C}$$

Network Measurements II

- Passive measurements (or **Network Monitoring**)
  - “non-intrusive”
  - Monitoring of existing traffic
  - Establishing of packet traces at different locations
  - Identification of packets, e.g. using hash values

  - Advantages
    - Does not affect applications
    - Does not modify the network behavior

  - Disadvantages
    - Requires suitable active network traffic
    - Limited to analysis of existing / current network behavior, situations of high load, etc. cannot be simulated/enforced
    - Does not allow the transport of additional information (time stamps, etc.) within measured traffic

Network Measurements III

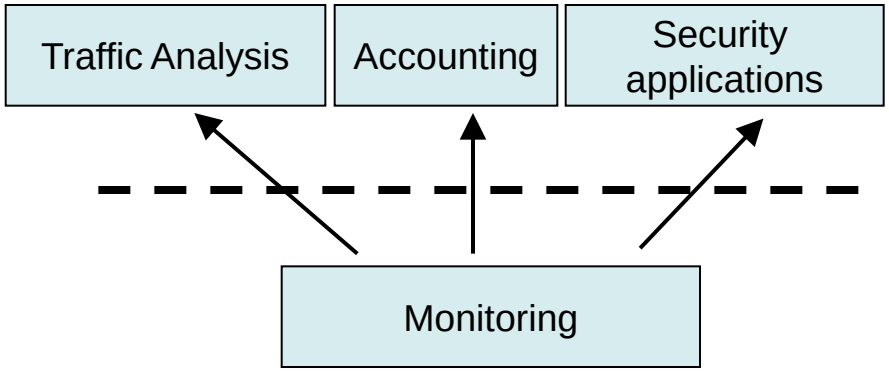
- Hybrid measurements
  - Modification of packet flows
    - Piggybacking
    - Header modification
  - Advantages
    - Same as for “passive”
    - additional information can be included (time-stamps, etc.)
  - Disadvantages
    - Modifying of data packets may cause problems if not used carefully

Measurement types (summary)

- Active Measurements
  - Intrusive
  - Find out what the network is capable of
  - Changes the network state
- Passive Measurements (or network monitoring)
  - Non-intrusive
  - Find out what the current situation is
  - Does not influence the network state (more or less)
- Hybrid
  - Alter actual traffic
  - Reduce the impact of active measurements
  - Might introduce new bias for applications

Network Monitoring

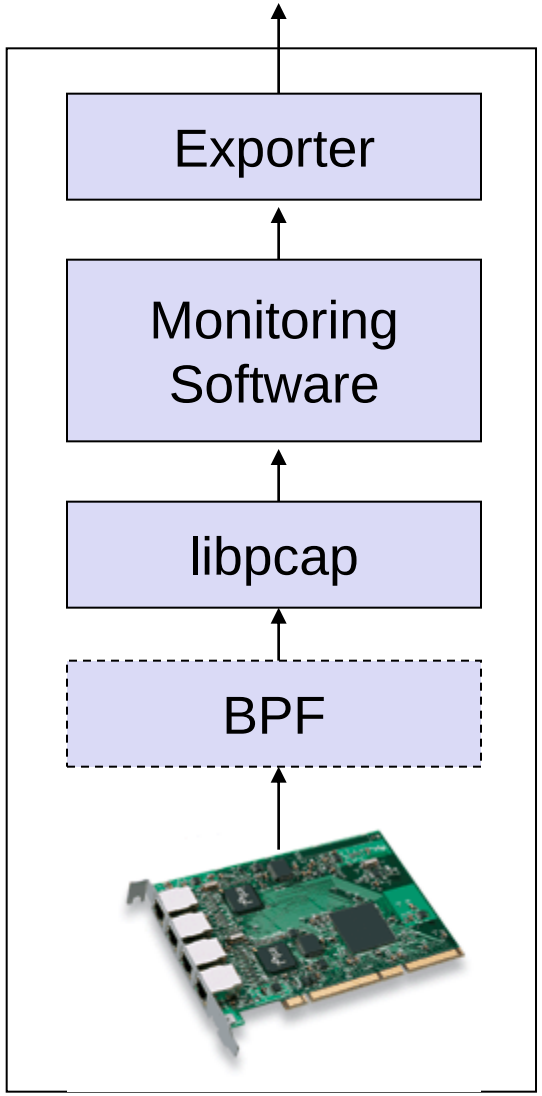
- Applications of network monitoring
  - Traffic analysis
    - Traffic engineering
    - Anomaly detection
  - Accounting
    - Resource utilization
    - Accounting and charging
  - Security
    - Intrusion detection
    - Detection of prohibited data transfers (e.g., P2P applications)
  - Research



- Open issues
  - Protection of measurement data against illegitimate use (encryption, …)
  - Applicable law (“lawful interception”, privacy laws, …)

Monitoring Probe

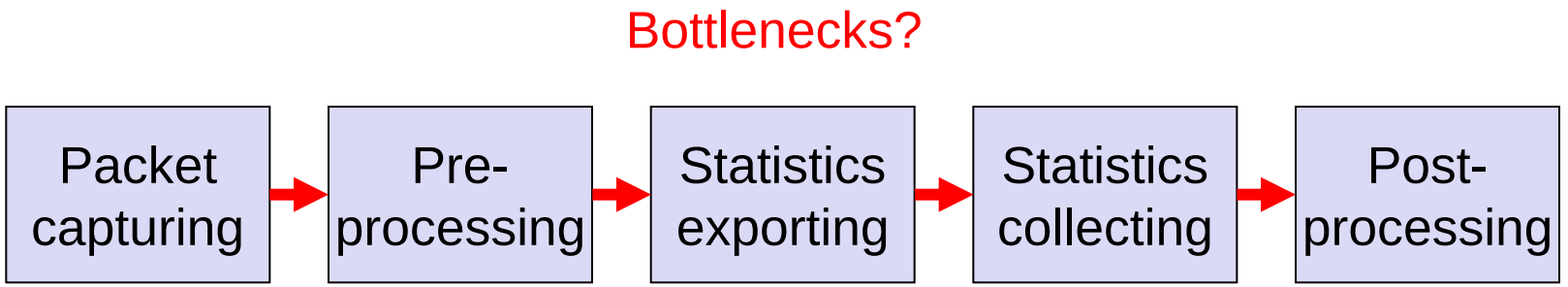
- Standardized data export
- Monitoring Software
- HW adaptation, [filtering]
- OS dependent interface (here: BSD)
- Network interface



High-Speed Network Monitoring

- Requirements
  - Multi-Gigabit/s Links
  - Cheap hardware and software → standard PC
  - Simple deployment

- Problems
  - Several possible bottlenecks in the path from capturing to final analysis



High-Speed Network Monitoring II

- Approaches
  - High-end (intelligent) network adapters
    - Large amounts of memory
    - Can do filtering, timestamping etc. on their own
  - Sophisticated algorithms/techniques in OS stack for
    - Maintaining packet queues
    - Elimination of packet copy operations
    - Maintaining state (e.g., managing hash tables describing packet flows; sophisticated packet classification algorithms)
  - Sampling
  - Filtering
  - Aggregation

  ⇨ more on subsequent slides

Special Network Adapters

- Server NICs (Network Interface Cards)
  - Direct access to main memory (without CPU assistance)
  - Processing of multiple packets in a single block (reduction of copy operations)

    → Reduced interrupt rates

- Monitoring interface cards
  - Dedicated monitoring hardware (usually only RX, no TX)
  - Programmable, i.e. certain processing (filtering, high-precision timestamps, ...) can be performed on the network interface card

Memory Management I

- Reduction of copy operations
  - Copy operations can be reduced by only transferring references pointing to memory positions holding the packet
  - Management of the memory is complex, garbage collection required
- Aggregation
  - If aggregated results are sufficient, only counters have to be maintained

Memory Management II

- Hash tables
  - Allow fast access to previously stored information
  - Depending on the requirements, different sections of a packet can be used as input to the hash function
- Multi-dimensional packet classification algorithms (e.g., HiPac)
  - Allow to test for 1,000s of complex filtering rules within one lookup operation (e.g., “all TCP packets from network 131.159.14.0/24, but not 131.159.14.0/27, and with source port 80, 443 or 6666–6670, but not with destination address 192.168.69.96–192.168.69.99 ➔ Apply rule 34”)
  - Mostly tree-based ➔ Lookups fast, but tree alterations costly.

Packet Sampling

- Goals
  - Reduction of the number of packets to analyze
  - Statistically dropping packets
- Sampling algorithms
  - Systematic sampling
    - Periodic selection of every n-th element of a trace
    - Selection of all packets that arrive at pre-defined points in time
  - Random sampling
    - n-out-of-N
    - Probabilistic
  - “Time machine” sampling: Sample first N bytes of every flow

Packet Filtering

- Goals
  - Reduction of the number of packets to analyze
  - Possibility to look for particular packet flows in more detail, or to completely ignore other packet flows
- Filter algorithms (explained subsequently)
  - Mask/match filtering
  - Router state filtering
  - Hash-based selection

Packet Filtering – Algorithms

- Mask/match filtering
  - Based on a given mask and value
  - In the simplest case, the selection range can be a single value in the packet header (e.g., mask out the least significant 6 bits of source IP address, match against 192.0.2.0)
  - In general, it can be a sequence of non-overlapping intervals of the packet
- Router state filtering
  - Selection based on one or more of the following conditions
    - Ingress/egress interface is of a specific value
    - Packet violated ACL on the router
    - Failed RPF (Reverse Path Forwarding)
    - Failed RSVP
    - No route found for the packet
    - Origin/destination AS equals a specific value or lies within a given range

Packet Filtering – Algorithms II

- Hash-based filtering
  - Hash function h maps the packet content c, or some portion of it, to a range R
  - The packet is selected if h(c) is an element of S, which is a subset of R called the selection range
  - Required statistical properties of the hash function h
    - h must have good mixing properties
      - Small changes in the input cause large changes in the output
      - Any local clump of values of c is spread widely over R by h
      - Distribution of h(c) is fairly uniform even if the distribution of c is not

Packet Filtering – Algorithms III

- Hash-based filtering (cont.)
  - Usage
    - Random sampling emulation
      - Hash function (normalized) is a pseudorandom variable in the interval [0,1]
    - Consistent packet selection and its application
      - If packets are selected quasi-randomly using identical hash function and identical selection range at different points in the network, and are exported to a collector, the latter can reconstruct the trajectories of the selected packets
      - ➔ Technique also known as *trajectory sampling*
      - Applications: network path matrix, detection of routing loops, passive performance measurement, network attack tracing

IPFIX: IP Flow Information Export

- IPFIX (IP Flow Information eXport) IETF Working Group
  - Standard track protocol based on Cisco Netflow v5…v9
- Goals
  - Collect usage information of individual data flows
  - Accumulate packet and byte counter to reduce the size of the monitored data
- Approach
  - Each flow is represented by its IP 5-tuple (protocol, srcIP, dstIP, srcPort, dstPort)
  - For each arriving packet, the statistic counters of the appropriate flow are modified
  - Whenever a flow is terminated (TCP FIN, TCP RST, timeout), its record is exported
  - Sampling algorithms can reduce the # of flows to be analyzed
- Benefits
  - Allows high-speed operation (standard PC: up to 1Gbps)
  - Flow information can simply be used for accounting purposes, as well as to detect attack signatures (e.g. increasing # of flows / time)

IPFIX – Work Principles

- Identification of individual traffic flows
  - 5-tuple: Protocol, Source IP, Destination IP, Source Port, Destination-Port
  - Example: TCP, 134.2.11.157, 134.2.11.159, 2711, 22
- Collection of statistics for each traffic flow
  - # bytes
  - # packets
- Periodical statistic export for further analysis

Flow	Packets	Bytes
TCP, 134.2.11.157,134.2.11.159, 4711, 22	10	5888
TCP, 134.2.11.157,134.2.11.159, 4712, 25	7899	520.202

IPFIX – IP Flow Information Export Protocol

- Quite a number of RFCs
  - Requirements for IP Flow Information Export (RFC 3917)
  - Evaluation of Candidate Protocols for IP Flow Information Export (RFC3955)
  - Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information (RFC 5101)
  - Information Model for IP Flow Information Export (RFC 5102)
  - Bidirectional Flow Export using IP Flow Information Export (IPFIX) (RFC 5103)
  - IPFIX Implementation Guidelines (RFC 5153)

- Transport protocol: Transport of exported IPFIX information records
  - SCTP must be implemented, TCP and UDP may be implemented
  - SCTP should be used
  - TCP may be used
  - UDP may be used (with restrictions – congestion control!)

IPFIX – Applications

- Usage-based accounting
  - For non-flat-rate services
  - Accounting as input for billing
  - Time or volume based tariffs
  - For future services, accounting per class of service, per time of day, etc.
- Traffic profiling
  - Process of characterizing IP flows by using a model that represents key parameters such as flow duration, volume, time, and burstiness
  - Prerequisite for network planning, network dimensioning, etc.
  - Requires high flexibility of the measurement infrastructure
- Traffic engineering
  - Comprises methods for measurement, modeling, characterization, and control of a network
  - The goal is the optimization of network resource utilization

IPFIX – Applications II

- Attack/intrusion detection
  - Capturing flow information plays an important role for network security
  - Detection of security violation
    1) Detection of unusual situations or suspicious flows
    2) Flow analysis in order to get information about the attacking flows
- QoS monitoring
  - Useful for passive measurement of quality parameters for IP flows
  - Validation of QoS parameters negotiated in a service level specification
  - Often, correlation of data from multiple observation points is required
  - This required clock synchronization of the involved monitoring probes

Network Traffic

Traffic by Port (I)

18 hours of traffic to AT&T dial clients on July 22, 1997

Name	Port	% Bytes	% Packets	Bytes/Packet
www	80	56,75	44,79	819
nntp	119	24,65	12,90	1235
pop3 email	110	1,88	3,17	384
cuseeme	7648	0,95	1,85	333
secure www	443	0,74	0,79	603
irc	6667	0,27	0,74	239
ftp	20	0,65	0,64	659
dns	53	0,19	0,58	210
…				

Traffic by Port (II)

24 hours of traffic to/from MWN clients in 2006

Name	Port	% Conns	% Succes	%Payload
www	80	70,82	68,13	72,59
cifs	445	3,53	0,01	0,00
secure www	443	2,34	2,08	1,29
ssh	22	2,12	1,75	1,71
smtp	25	1,85	1,05	1,71
	1042	1,66	0,00	0,00
	1433	1,06	0,00	0,00
	135	1,04	0,00	0,00
	< 1024	83,68	73,73	79,05
	> 1024	16,32	4,08	20,95

Traffic by Port (III)

- Port 80 dominates traffic mix
  - Still growing
    - More web applications
    - Tunnel everything over port 80
- Characterization of traffic by port is possible
  - Well-known ports
    (1–1024; take a look at `/etc/services`)
- Growing margin of error
  - Automatic configuration
  - * over http: VPN, P2P, Skype, AJAX-SSH, …
  - Aggressive applications (e.g. Skype):
    „just find me an open port“

Traffic Flows

18 hours of traffic to AT&T dial clients on July 22, 1997

Name	Port	% Bytes	% Pkts	Bytes/ Pkt	% Flows	Pkts/ Flow	Duration (s)
www	80	56,75	44,79	819	74,58	12	11,2
nntp	119	24,65	12,90	1235	1,20	210	132,6
pop3 email	110	1,88	3,17	384	2,80	22	10,3
cuseeme	7648	0,95	1,85	333	0,03	1375	192,0
secure www	443	0,74	0,79	603	0,99	16	14,2
irc	6667	0,27	0,74	239	0,16	89	384,6
ftp	20	0,65	0,64	659	0,26	47	30,1
dns	53	0,19	0,58	210	10,69	1	0,5
…							

Elephants and Mice

- Many very short flows (30% < 300 bytes)
- Many medium-sized flows (short web transfers)
- Few long flows
- But:
  Most bytes belong to these long flows (large images, files, flash, video)
- Same picture for other metrics
  - Bytes/flow
  - Packets/flow
  - Lifetime
- Flow densities are traffic patterns and signatures

Architecture: the big picture

Structure

- Internet architecture
  - Requirements, assumptions
  - Design decisions
- Shortcomings and „Future Internet“ concepts
  - „Legacy Future Internet“: IPv6, SCTP, …
  - Security
  - QoS, multicast
  - Economic implications, „tussle space“
  - Mobility and Locator–ID split
  - In-network congestion control
  - Modules instead of layers
  - Delay-tolerant/disruption-tolerant networking
  - Content-based networking/Publish–subscribe architectures
  - Evolutionary vs. Revolutionary/Clean-slate

Common View of the Phone Network

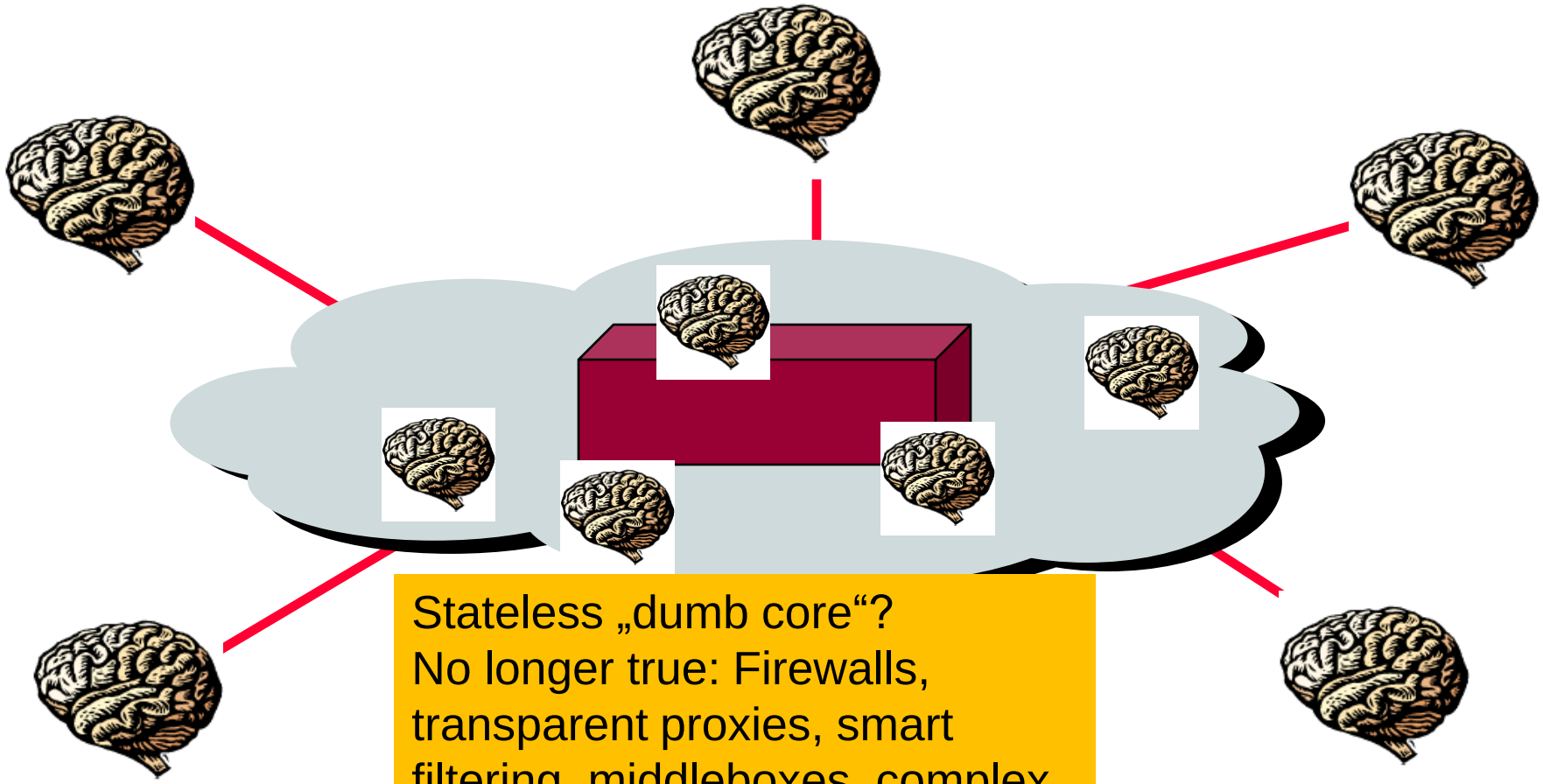
Even today, you can connect a 1936-built German standard phone to an analog phone line or, e.g., a FritzBox

brick (dumb)

brain (smart)

lock (you can't get in)

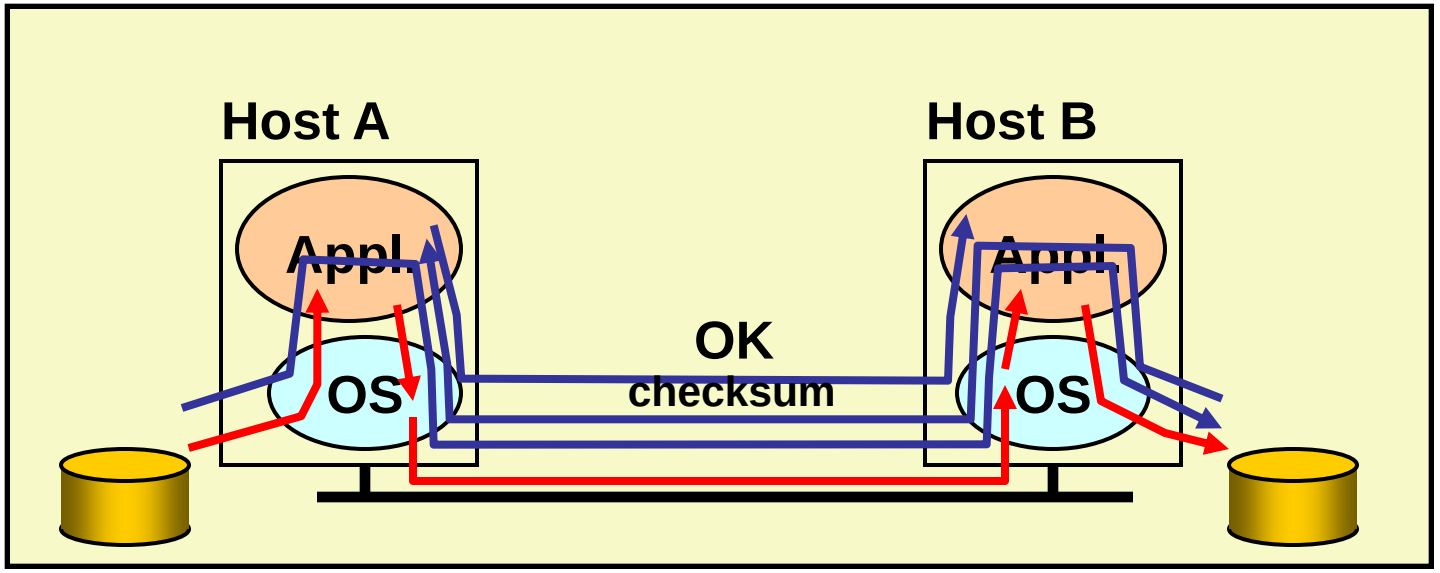
Common View of the IP Network



Stateless „dumb core“?
No longer true: Firewalls, transparent proxies, smart filtering, middleboxes, complex routing protocols like BGP, …

The Internet End-to-End principle

Example: Reliable File Transfer



- Solution 1: make each step reliable, and then concatenate them
- Solution 2: each step unreliable – end-to-end check and retry

Discussion

- Is solution 1 good enough?
  - No – what happens if components fail or misbehave (bugs)?
- Is reliable communication sufficient?
  - No – what happens in case of, e.g., disk errors?
- so need application to make final correctness check anyway
- Thus, full functionality can be entirely implemented at application layer; *no* need for reliability at lower layers

Discussion

Q: Is there any reason to implement reliability at lower layers?

A: YES: “easier” (and more efficient) to check and recovery from errors at each intermediate hop

- e.g.: faster response to errors, localized retransmissions

Internet & End-to-End Argument

- Network layer provides *one* simple service:
  best effort datagram (packet) delivery
- Transport layer at network edge (TCP) provides end-to-end error control
  - Performance enhancement used by many applications (which, alternatively, could provide their own error control)
- All other functionality: at application layer!
  - Network management: Routing protocols, ICMP, …
    - In-band signalling: Management traffic uses same paths
  - Network services: DNS, middleware/brokers, …

Internet Design Philosophy (Clark'88)

In order of importance:

Different ordering of priorities would make a different architecture!

0 Connect existing networks
   - initially ARPANET, ARPA packet radio, packet satellite network
1. Survivability
   - ensure communication service even with network and router failures
2. Support multiple types of services
3. Must accommodate a variety of networks
4. Allow distributed management
5. Allow host attachment with a low level of effort
6. Be cost effective
7. Allow resource accountability

1. Survivability

- Continue to operate even in the presence of network failures (e.g., link and router failures)
  - As long as network is not partitioned, two endpoints should be able to communicate
  - Any other failure (excepting network partition) should be transparent to endpoints
- Decision: maintain end-to-end transport state only at end-points
  - eliminate the problem of handling state inconsistency and performing state restoration when router fails
- Internet: stateless network-layer architecture
  - No notion of a session/call at network layer
- Remark: “Internet was built to survive global thermonuclear war” = urban legend; untrue

2. Types of Services

- Add UDP to TCP to better support other apps
  - e.g., “real-time” applications
- Arguably main reason for separating TCP, IP
- Datagram abstraction: lower common denominator on which other services can be built
  - Service differentiation was considered (ToS bits in IP header), but this has never happened on the large scale (Why?)

3. Variety of Networks

- Very successful (why?)
  - Because of minimalism
  - Only requirement from underlying network: to deliver a packet with a “reasonable” probability of success
- …but does *not* require:
  - Reliability
  - In-order delivery
  - Bandwidth, delay, other QoS guarantees
- The mantra: IP over everything
  - Then: ARPANET, X.25, DARPA satellite network, phone lines, …
  - Today: Ethernet, DSL, 802.11, GSM/UMTS, …
  - Soon: LTE, WIMAX, …

Other Goals

- Allow distributed management
  - Administrative autonomy: IP interconnects networks
    - each network can be managed by a different organization
    - different organizations need to interact only at the boundaries
    - … but this model complicates routing

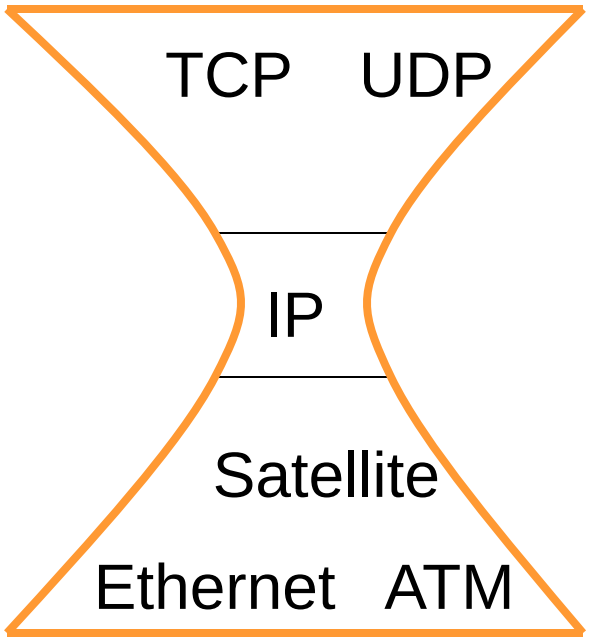
- Cost effective
  - sources of inefficiency
    - header overhead
    - retransmissions
    - routing
  - …but “optimal” performance never been top priority

Other Goals (Cont)

- Low cost of attaching a new host
  - not a strong point → higher than other architecture because the intelligence is in hosts (e.g., telephone vs. computer)
  - bad implementations or malicious users can produce considerably harm (remember fate-sharing?)
- Accountability
  - Not a strong point: no financial interests (research network!)

Summary: Internet Architecture

- Packet-switched datagram network
- IP is the glue (network layer overlay)
- IP hourglass architecture
  - All hosts and routers run IP
  - IP hides transport/application details from network
  - IP hides network details from transport/application
- Stateless architecture
  - No per-flow state inside network
  - Intelligence (i.e., state keeping) in end hosts, but not in core



IP hourglass

Summary: Minimalist Approach

- Dumb network
  - IP provides minimal functionalities to support connectivity
  - Addressing, forwarding, routing
- Smart end system
  - Transport layer or application performs more sophisticated functionalities
  - Flow control, error control, congestion control
- Advantages
  - Accommodate heterogeneous technologies (Ethernet, modem, satellite, wireless)
  - Support diverse applications (telnet, SMTP, FTP, X11, Web, ssh, SSL/TLS, POP, IMAP, Peer-to-Peer, …)
  - Decentralized network administration

The KISS principle

- KISS = “Keep it simple, stupid!”
- Success of…
  - IP
  - Ethernet
  - RISC processors
  - SIP vs. H.323
- “Building complex functions into network optimizes network for small number of services, while substantially increasing cost for uses unknown at design time”

Internet architecture: Some explicit or implicit assumptions

- A research network
  - No economic/business/judicial aspects, no competition
  - Cooperative, perhaps even altruistic participants
- Knowledgeable and responsible end users; administrators even more so
- Almost no malicious participants
  - Perhaps some malicious users? (➔ password protection),
  - …but no malicious systems administrators,
  - …and certainly no malicious network operators
- A couple of thousand nodes, perhaps a million users
- No mobility: End hosts will not shift their position within network
- Most links are wired; packet loss indicates network congestion
- Just a temporary solution

- **…and yet it still works!? Amazing!**

But that was yesterday

… what about tomorrow? Or even: today?

Rethinking Internet Design

What's changed?

- Operation in untrustworthy world
  - Endpoints can be malicious
  - If endpoint not trustworthy, but want trustworthy network ⇨ more mechanism in network core
- More demanding applications
  - End-end best effort service not enough
  - New service models in network (IntServ, DiffServ)?
  - New application-level service architecture built on top of network core (e.g., CDN, P2P)?

Rethinking Internet Design

What's changed (cont.)?

- ISP service differentiation
  - ISP doing more (than other ISPs) in core is competitive advantage
- Rise of third party involvement
  - Interposed between endpoints (even against will of users)
  - e.g., Chinese government, US recording industry
- less sophisticated users

All five changes motivate shift away from end-to-end!

What's at stake?

"At issue is the conventional understanding of the "Internet philosophy"

- freedom of action
- user empowerment
- end-user responsibility for actions taken
- lack of control "in" the net that limit or regulate what users can do

The end-to-end argument fostered that philosophy because they enable the freedom to innovate, install new software at will, and run applications of the users' choice"

[Blumenthal and Clark, 2001]

Technical response to changes

- Trust: emerging distinction between what is “in” network (us, trusted) and what is not (them, untrusted).
  - Ingress filtering
  - Firewalls
- Modify endpoints
  - Harden endpoints against attack
  - Endpoints/routers do content filtering: Net-nanny
  - CDN, ASPs: rise of structured, distributed applications in response to inability to send content (e.g., multimedia, high bw) at high quality

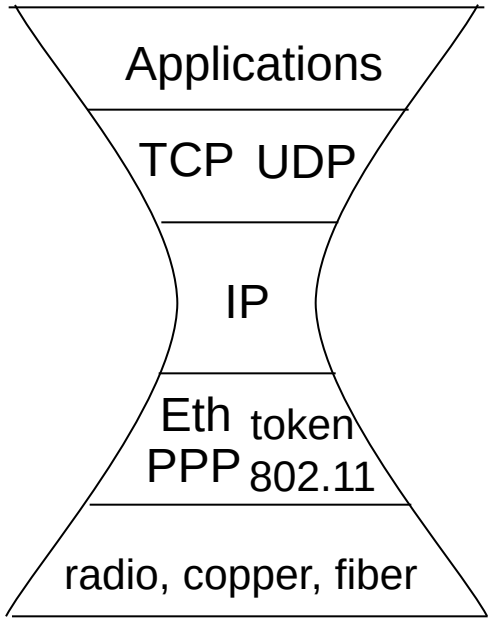
Technical response to changes

- Add functions to the network core:
  - Filtering firewalls
  - Application-level firewalls
  - NAT boxes
  - Transparent Web proxies

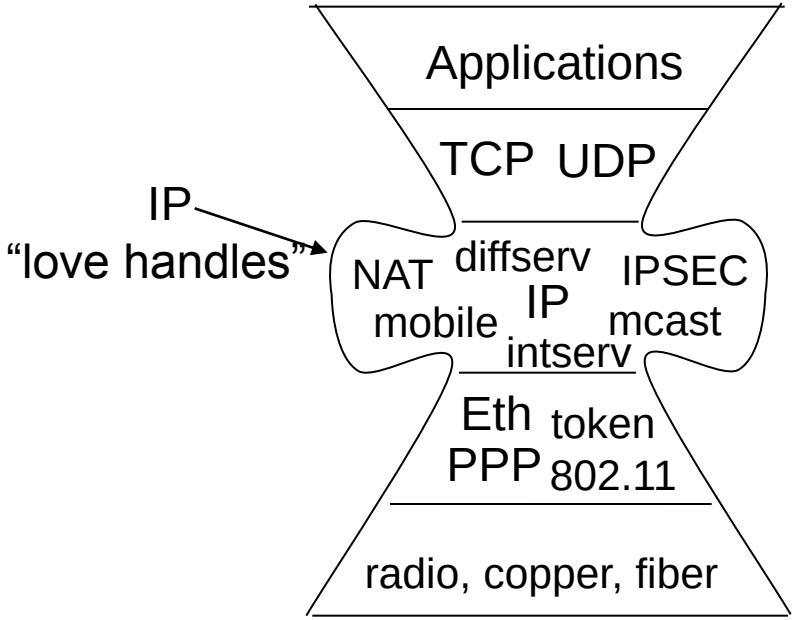
All operate *within* network, making use of application-level information

- Which addresses can do what at application level?
- If addresses have meaning to applications, NAT must "understand" that meaning. Difficult!

Big picture: supporting new applications – losing the IP hour glass figure?

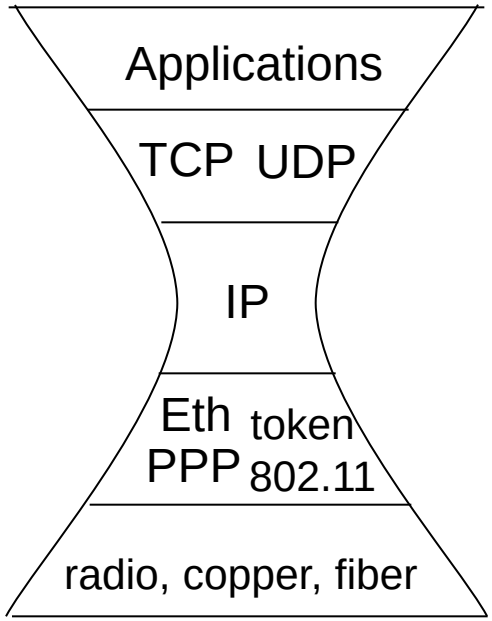


IP “hourglass”

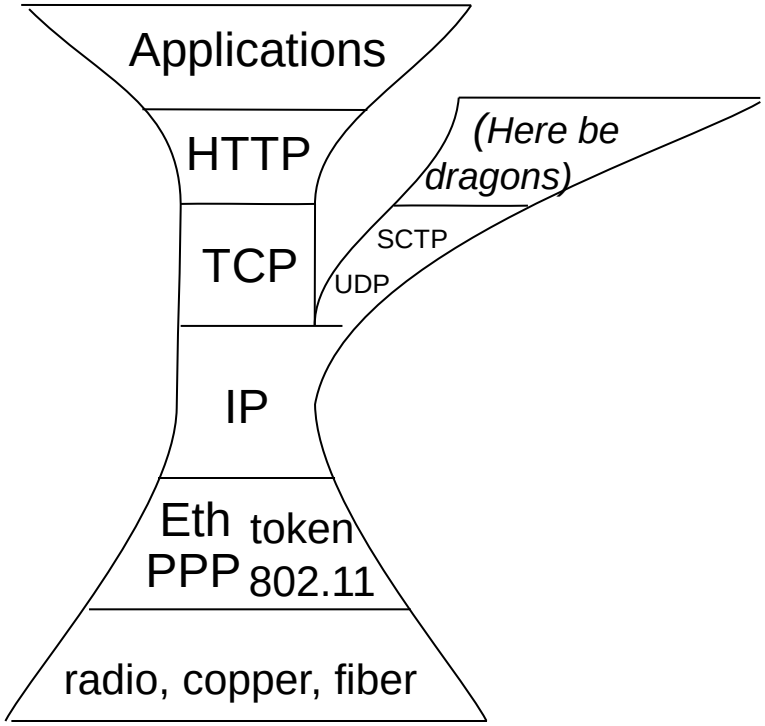


Middle-age IP “hourglass”?

Big picture: supporting new applications – losing the IP hour glass figure?

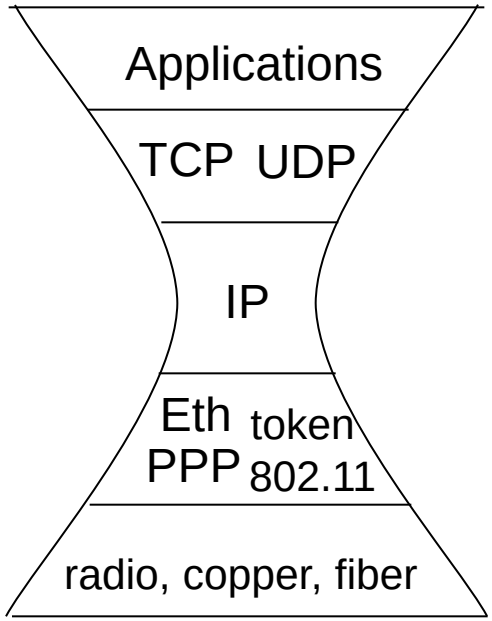


IP “hourglass”

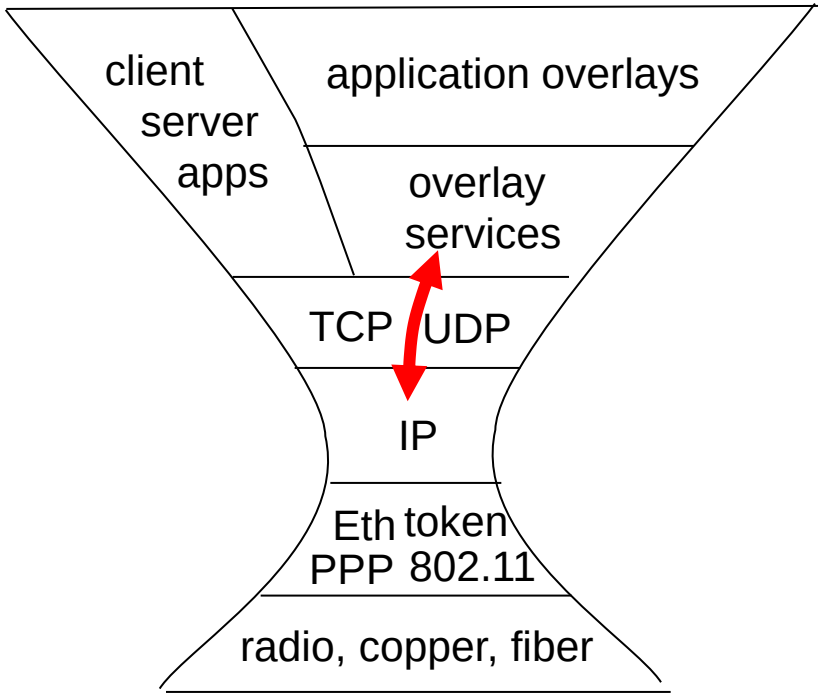


HTTP/IP “long-neck hourglass”

Big picture: supporting new applications – losing the IP hour glass figure?



IP “hourglass”



Future Internet concepts

- Shortcomings and „Future Internet“ concepts
  - Security
  - QoS, multicast
  - Economic implications, „tussle space“
  - Mobility and Locator–ID split
  - In-network congestion control
  - Modules instead of layers
  - Delay-tolerant/disruption-tolerant networking
  - Content-based networking/Publish–subscribe architectures
  - Evolutionary vs. Revolutionary/Clean-slate

FIND: Future Internet Network Design

- New long-term US NSF initiative
- Questions:
  - Requirements: for the global network of 15 years from now - what should that network look like and do?
  - How would we re-conceive tomorrow's global network today, if we could design it from scratch?
- Major thrusts:
  - Security, manageability, mobility (DTN, naming, wireless)
  - I.e.: what the original Internet didn’t get right

The Internet has no built-in security (I)

- Problem #1: Cannot protect from unwanted traffic
  - Spam
  - DoS attacks
  - Wustrow, Karir, Bailey, Jahanian, Huston: *Internet background radiation revisited.* Proceedings of ACM/USENIX Internet Measurement Conference, 2010
- Solutions
  - Protocols
    - DKIM
    - Cookies (e.g., TCP SYN cookies)
  - Treating the symptoms
    - Spam filters
    - Rate limiting at firewall
    - Tar pits, honey pots
    - Network intrusion detection systems (NIDS)
    - …

The Internet has no built-in security (II)

- Problem #2: Traffic not encrypted by default
  - E-Mail, Web: readable by attackers
- Problem #3: Traffic not authenticated by default
  - E-Mail, Web: can be manipulated/forged

- Solutions
  - IPSec
  - SSL/TLS
  - ssh
  - …but do they work?

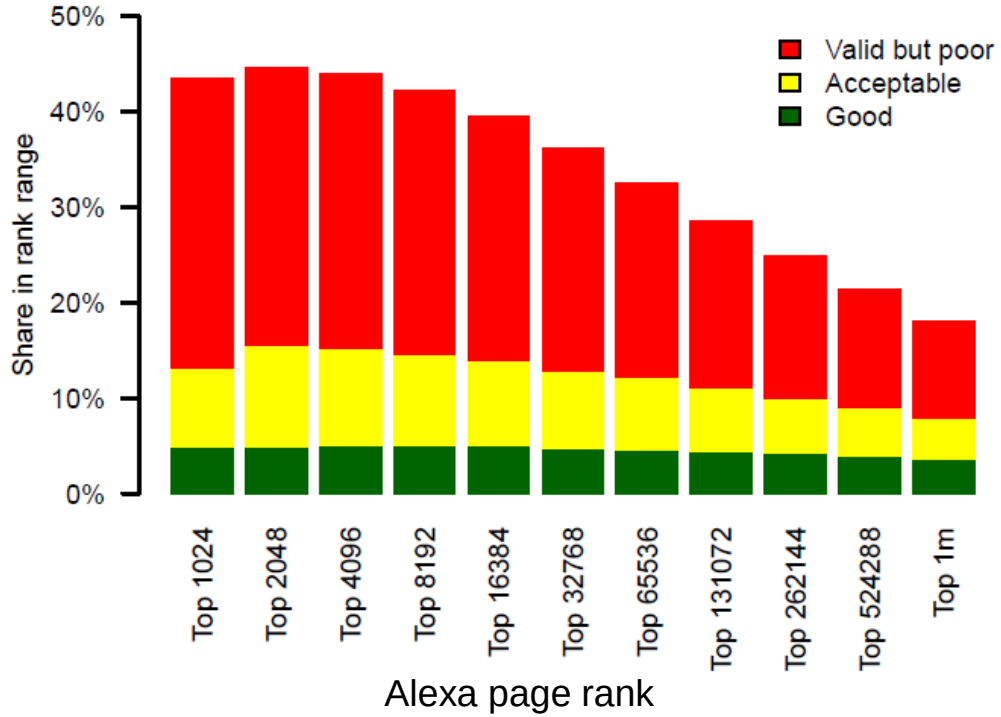
Problems with X.509 certificates (I)

- X.509 certificates: Used for, e.g., SSL/TLS

- Every root CA, every intermediate CA can issue certificates for *any* domain. Example:
  - Authoritarian regime installs transparent HTTPS proxy…
  - ...and gains access to some intermediate CA
  - Proxy intercepts all HTTPS connections, answers with *valid(!)* certificate to client (MITM attack)
  - Client thinks it talks to HTTPS server – in fact proxy can read everything in plaintext
  - You can buy such boxes for a couple of 1,000$
  - (Firefox plugins for detection: CrossBear, CertificatePatrol, CertificateWatch,…)

Problems with X.509 certificates (II)

- Poor administrative knowledge
- Example: Certificate quality in top 1 million Web sites



- Taken from:
  Holz, Braun, Kammenhuber, Carle: *The SSL landscape – a thorough analysis of the X.509 PKI using active and passive measurements.* Proceedings of ACM/USENIX Internet Measurement Conference (IMC), 2011

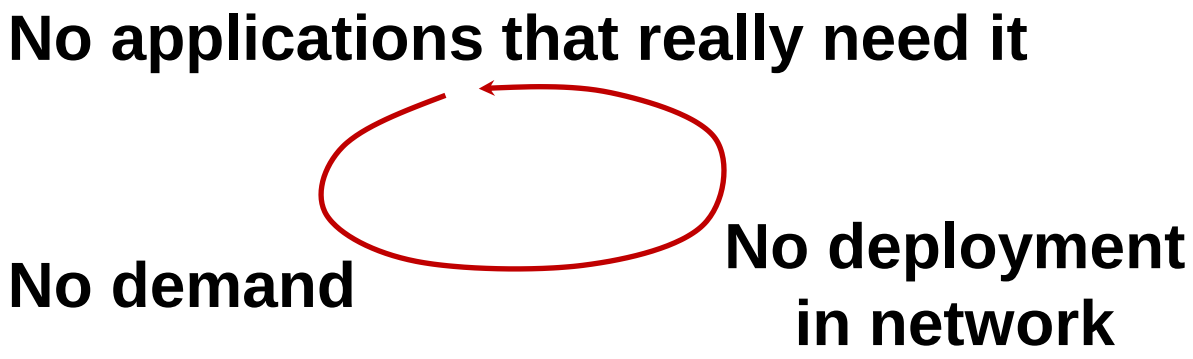
Multicast and QoS

- Multicast routing protocols (MOSPF, PIM, …) exist and work
- QoS protocols (IntServ, DiffServ, …) exist and work
- IP header *and* Ethernet header (802.1p) contain ToS bits

- …but no end user application is using it!
  - Multicast: Would be nice for online TV
  - QoS: Would be nice for throttling P2P and ftp downloads while increasing responsiveness of ssh and games and stability of VoIP calls and video streaming
- At least some „invisible“ usage
  - Prioritization of specific traffic within company networks
  - ISPs may give QoS guarantees for VPNs
  - TV over IP („Triple play“) uses multicast, but application not directly accessible by user

Why don’t ISPs offer multicast or QoS to end users?

1. Same chicken–egg problem / vicious circle as with IPv6:



2. Who should pay once traffic crosses AS boundaries?
   - Who pays „expedited forwarding“?
     Sender AS, receiver AS, both?
   - Who pays in-network duplication for multicast?
     Sender AS, receiver ASes, or entire network?
   - How can sender/receiver be charged?
   - How can multicast sender know how much it will be charged?

Economic aspects, conflicting interests: „Tussle“

- Internet participants
  - Different stakeholders
  - Competition
  - Conflicting interests
- Examples
  - Users want to share music and videos – GEMA/RIAA don‘t
  - Users want secret communication – governments don‘t
  - ISPs need to cooperate – but are fierce competitors
- Call this aspect „tussle“
  - Internet architecture only partially reflects this (BGP policy routing)
  - Tussle Space: Future Internet architecture should anticipate various kinds of tussle and integrate defined mechanisms

Clark, Sollins, Wroclawski, Braden: Tussle in Cyberspace: Defining Tomorrow‘s Internet. Proceedings of ACM SIGCOMM, 2002

Mobility, Locator–ID split: Problem

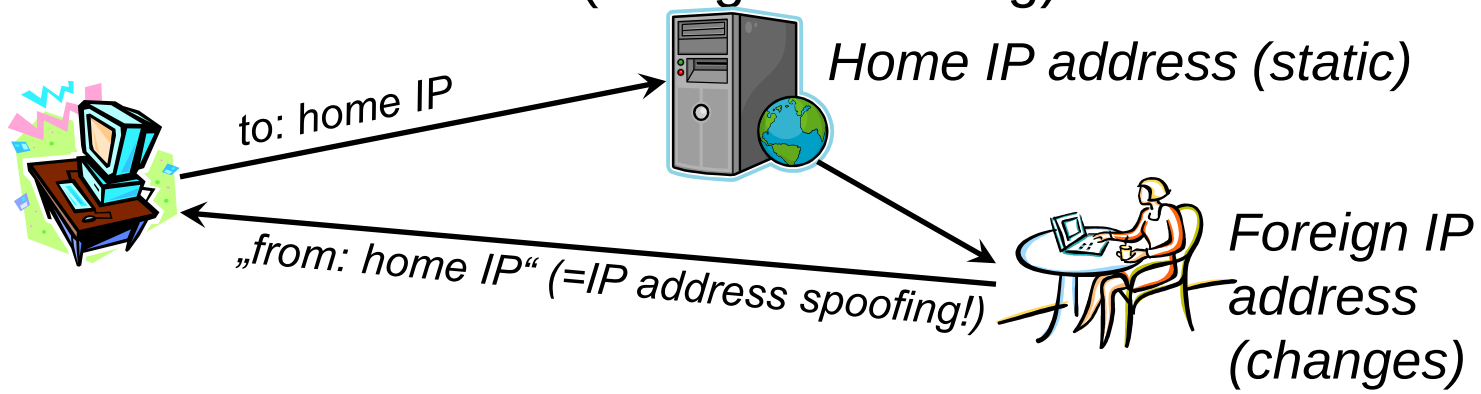
- **Identifier:** IP address identifies communication endpoint
  - Keywords: TCP 4-tuple (srcIP, dstIP, srcP, dstP), DNS entry, …
- **Locator:** IP address specifies how to reach destination
  - Keywords: Netmask, longest prefix match, CIDR, …
- Problem: What if IP addresses change?
  - Scenario 1: User mobility
    Example: Lose WLAN connection, switch to UMTS/LTE
    ➔ IP address changes
    ➔ All active TCP, UDP connections break: ssh, Jabber,…
  - Scenario 2: Network mobility
    Example: Middle-sized company switches to a different ISP
    ➔ *All* IP addresses of *all* their hosts need to be changed
    ➔ High maintenance effort; cannot switch instantaneously
  - Scenario 3: IP anycast = one IP address, but multiple hosts.
    Example: Some DNS root servers use one IP address for multiple servers at entirely different locations

Mobility, Locator–ID split: Solutions (1)

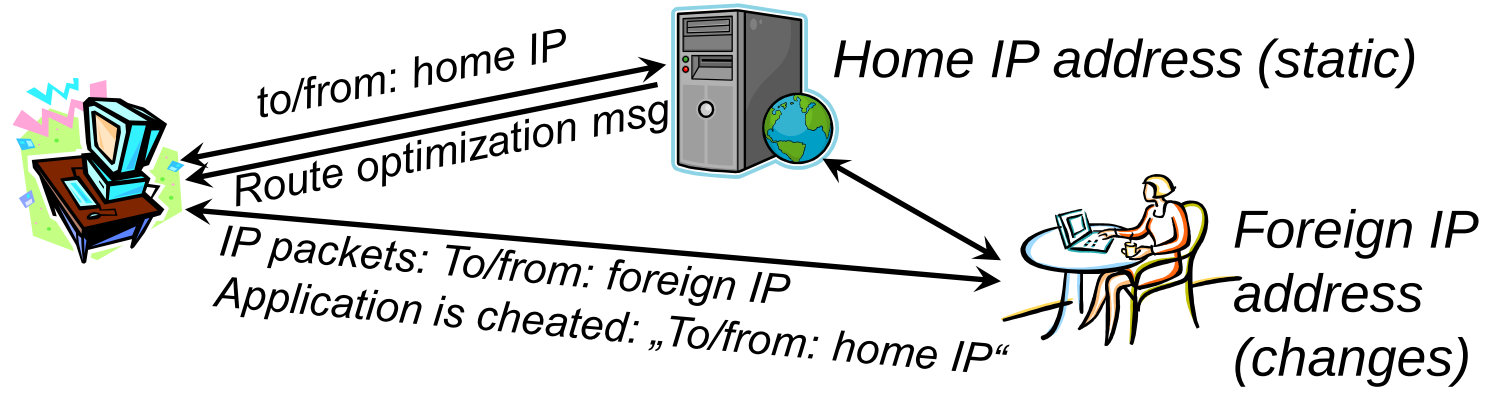
- Dynamic DNS
  - Assumptions:
    - Mostly use short-lived connections
    - Mostly connect to host names, not IP addresses
  - Idea:
    - Keep short-lived DNS entries
    - If IP address changes, immediately update DNS entry
  - Drawbacks:
    - Service unavailable for several minutes (until new old entry has expired, new entry has propagated)
    - Some faulty DNS servers ignore short-lived timeout value
    - Does not help active connections
    - Does not help connections that do not use DNS

Mobility, Locator–ID split: Solutions (2)

- Mobile IP
  - Old standard: Mobile IPv4 (triangular routing)



    - Incompatible with firewalls, ingress filtering, …
  - New standard: Mobile IPv6



  - **Drawbacks: Both require a Home Agent**

Mobility, Locator–ID split: Solutions (3)

- Host Identity Protocol (HIP)
  - Additional HIP layer between IP and transport (e.g., TCP)
  - Every host has *static* 128-bit Host Identifier
    - Identifier „looks“ like an IPv6 address to transport protocol
  - Two hosts that want to communicate initiate a HIP session
    - Exchange of Host Identifiers
    - Exchange of crypto keys
  - If IP address of one host changes:
    - Send information address change to other HIP partner
    - Will send future HIP traffic to new IP address
    - Information cryptographically signed ➔ no connection hijacking
  - Drawbacks
    - One additional RTT for HIP handshake at start of connection
    - Not transparent – need changes in operating system
    - Both communication partners need to support HIP

Mobility, Locator–ID split: Solutions (4)

- Locator–ID Separation Protocol (LISP)
  - Use some IP addresses as locators, use others as identifiers
  - End hosts / end networks only see identifier IPs
  - Network core only sees locator IPs
  - Addresses become dynamically re-written (similar to NAT) upon arrival at / departure from LISP-enabled network
  - Moving host, moving network: Update address rewrite tables
  - Good: Incrementally deployable; transparent
  - Bad: Not really for end hosts (scalability); not yet supported
- SCTP
  - SCTP association knows all IP addresses of both endpoints
  - If primary connection fails: transparent switch-over
  - Drawback: Only works with SCTP… but nobody uses SCTP!

Shortcoming: No in-network congestion control

- Congestion control today
  - End hosts: Short timescales
    - TCP
    - Others (e.g., DCCP): Should be TCP-friendly
    - Disadvantages:
      - No enforcement (e.g., UDP)
      - Can only adjust speed; cannot select better path
  - Network: Long timescales
    - Traffic engineering: Measure traffic, reconfigure routing
    - EIGRP
    - No cooperation across AS boundaries
  - Why not at shorter timescales?
    - Bad experience in ARPANET
    - Highly nonlinear system: prone to oscillation
    - Interaction with TCP congestion control ➔ even worse

Layers vs. Modules („Functionality Lego“)

- Observation: Many functionalities implemented multiple times at multiple layers. Examples:
  - Encryption and authentication:
    ssh (Application), SSL (Session), IPSec (Network),
    GSM/UMTS/LTE (Data Link)
  - Flow control:
    TCP (Transport), Ethernet and WLAN (Data Link)
  - Guaranteed delivery through ACKs/resends:
    Custom protocols (Application), TCP (Transport),
    high-loss satellite links (Data Link)
- Idea:
  - Encapsulate specific functionality within modules
  - Ensure that modules can be plugged together in (more or less) arbitrary combination and sequence
  - Application/communication endpoints (and network?) specify „building plan“ during initial handshake

DTN: Delay-tolerant networking / Disruption-tolerant networking (I)

- Andrew Tanenbaum: „Never underestimate the bandwidth of a station wagon full of tapes hurtling down the highway.“
  - Send small packet with SD cards or hard disk (1 TByte)
  - Let journey time be 1 week (➔ RTT = 2 weeks!)
  - Bandwidth = around 13 Mbit/s!
- Underdeveloped regions: Send data via, e.g.,
  - Letters/packets containing storage media
  - Messengers carrying storage media
  - Homing pigeons ☺ („IP over avian carriers“, RFC1149 et al.)
  - WLAN-/Bluetooth-equipped phones/laptops/… that can exchange data in passing and cache it during transit
- Also could be used during emergency with large-scale infrastructure failures (e.g., Hurricane Katrina)
- Similar characteristic: Space travel! (Very long delays; long connection breaks, e.g., when spacecraft behind a planet)

DTN: Delay-tolerant networking / Disruption-tolerant networking (II)

- Protocols: No „gold standard“ yet
  - Vastly different scenarios (e.g., underdeveloped regions vs. space travel)
- Protocol/application selection
  - Bundle Protocol
  - Lidlicker
  - Saratoga
  - Offline browsing proxy (WWWoffle)
- Experiments/prototype deployments
  - Some in Lapland, some in South Africa
  - EU project to connect remote villages in Slovenia
- Future research includes:
  - Routing algorithms
  - Gateways and interfaces to existing services (Mail, Web, …)

Content-based networking and publish–subscribe architectures (I)

- Observation:
  - IP addresses *hosts*
  - Browsers, P2P clients etc. address *content objects:* Specific Web pages, MP3 files with specific music, …
- Idea:
  - Address content chunks instead of hosts
  - Routers can replicate and/or cache popular chunks
- Requesting chunks:
  - Send interest/subscription request into network
  - Request will be forwarded from router to router
  - If matching content chunk(s) found, send them to requester

Content-based networking and publish–subscribe architectures (II)

- A lot of features automatically built in:
  - Multicast (even asynchronously!)
  - In-network caching
  - Resilience: If one router with content fails, it still will be available on other routers
  - Delay-tolerant networking: Routers cache contents anyway, so why not have the caching routers roam around as well?
  - Some protection from DoS attacks: I only get traffic that I requested

Content-based networking and publish–subscribe architectures (III)

- Some issues to be addressed
  - Authenticity: How to make sure that malicious users cannot inject a fake version of, e.g., an online banking service?
  - Routing: How do routers know which interest packets should be forwarded to which neighbour(s)?
  - Versioning: How to make sure that old versions of a content object are quickly replaced in router caches (e.g., content object „current DAX level“ or „Mensa food plan“)
  - Protocol logic:
    - Subscription („send me all matching chunks“) vs. requests („send me one matching chunk“)
    - Timeouts
  - Protection from flooding induced by excessive subscription
  - Addressing scheme

Content-based networking and publish–subscribe architectures (IV)

- OK, sounds good for things like YouTube, heise.de, etc.
- But what about obvious peer–peer sessions? (ssh, VoIP, etc.)

- Solution:
  - Subscribe to contact requests
  - If contact request is received, subscribe to answer packets of contact request originator
  - Start sending out own data (e.g., own voice)
  - Receive answers from peer (e.g., acknowledgement packets; other‘s voice)

Content-based networking and publish–subscribe architectures (V)

- Some thoughts on the address length: How much do we need?
- Current Internet
  - IPv4: 32 bits = 4 billion addresses (about 30% used)
  - IPv6: 128 bits
- Consider something like a worst-case scenario:
  - Assume *every atom* is used to store one information chunk!
    - About 10^{80} particles in the visible universe
  - Every chunk changes its state every 10^{-44}s! (Planck Time)
  - For 1 million years!
  - We waste 99% of the address space! (IPv4: only 60% wasted)
  - How many bits do we need?
    - $\log_2 \left(10^{80} \cdot (10^{44} \cdot 60) \cdot (60 \cdot 24 \cdot 365 \cdot 10^6) \cdot 100\right)$

    = 463 bits = 58 Bytes. (N.B.: IPv6 header+TCP header = 56 Bytes)
    - One of the rare cases where exponential growth is in our favour!

Future Internet approaches

Revolutionary (clean slate)

- Today's Internet is broken by design
- Trying to fix it leaves us with *-over-HTTP-over-TCP-over-IP, i.e., with something like the memory model of Intel x86, the A20 gate, 110V vs. 230V and 50Hz vs. 60Hz power, …
- New architecture will be radically different
- ➔ Let's throw everything away and start completely anew to ~~get it right from the beginning~~ introduce new design mistakes

Evolutionary

- The Internet has been amended many times in the past:
  - Adding congestion control to TCP
  - Introduction of DNS instead of distribution of `/etc/hosts` text files
  - Introduction of classless interdomain routing instead of Class-A, Class-B, Class-C networks
  - Introduction of SSL, IPSec, ssh, …
  - Introduction of Multicast, ToS bits
  - Introduction of IPv6
- ➔ Let's fix the shortcomings incrementally by introducing new protocols: ~~Never change a running system~~ Create a truly unmanageable behemoth of conflicting protocols

Future Internet: Some readings

- Mark Handley: *Why the Internet only just works.*
  BT Technology journal, 2006
- Anja Feldmann: *Internet Clean-Slate Design: What and Why?*
  Editorial note, ACM CCR, 2007
- Akhshabi, Dovrolis: *The evolution of layered protocol stacks leads to an hourglass-shaped architecture.*
  Proceedings of ACM SIGCOMM, 2011

- N.B. With a TUM or LMU IP address, you can download most scientific articles for free if you enable the LRZ proxy:
  http://www.lrz.de/services/netzdienste/proxy/journals-access/

The end!

Feedback time!

- What did you like about the course?
- What could be improved?
- Are there topics you would have liked to be covered
  - … to a greater extent?
  - … to a lesser extent?
  - … not at all?