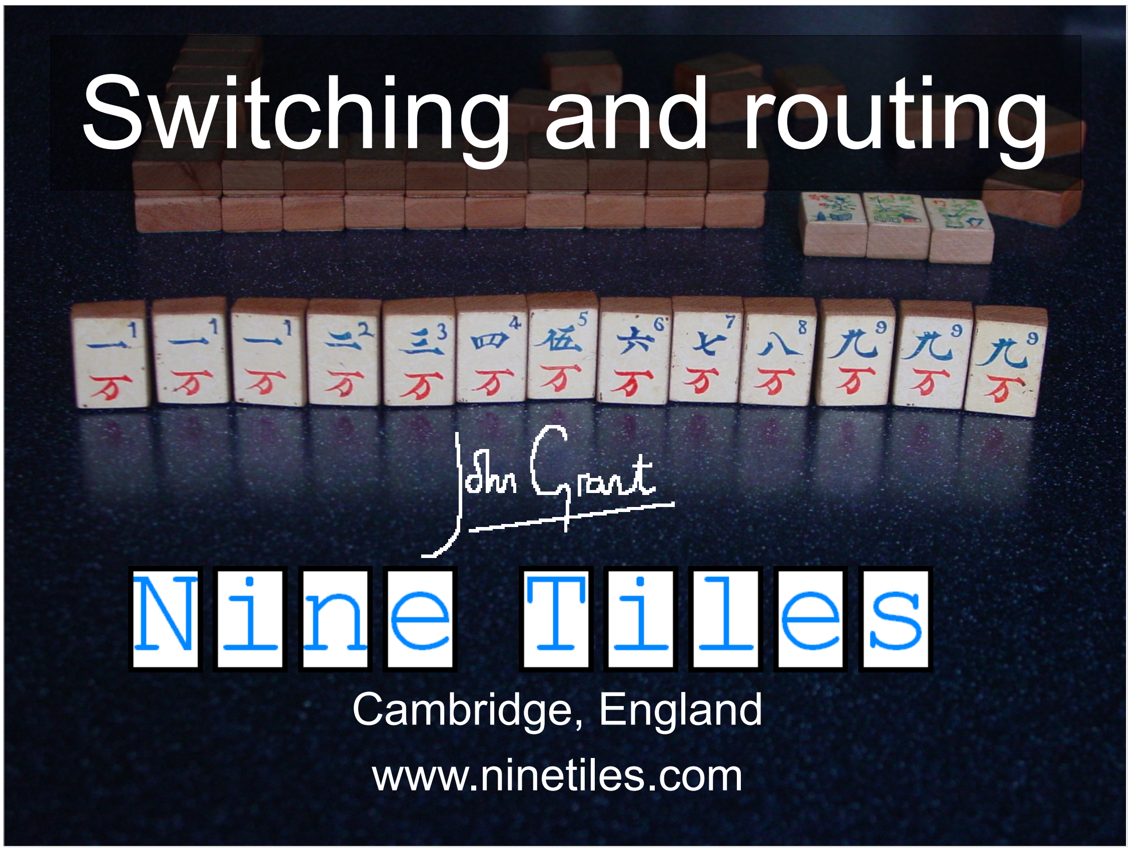


Switching and routing

John Grant

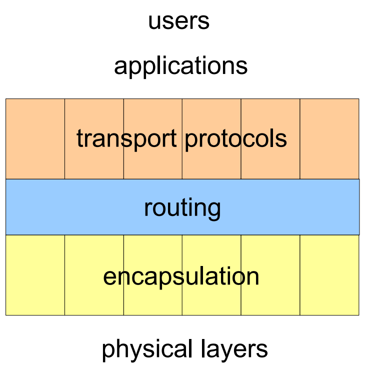
Nine Tiles

Cambridge, England

www.ninetiles.com

The image shows the “calling nine tiles” Mah-jongg hand.

Network layer



- the one part of the stack that's universal

The “switching and routing” part of FN is layer 3 (the network layer) in the OSI model; it carries data between end-systems along routes that pass through switches. Like IP, it is a single protocol which links the higher and lower layers in the stack.

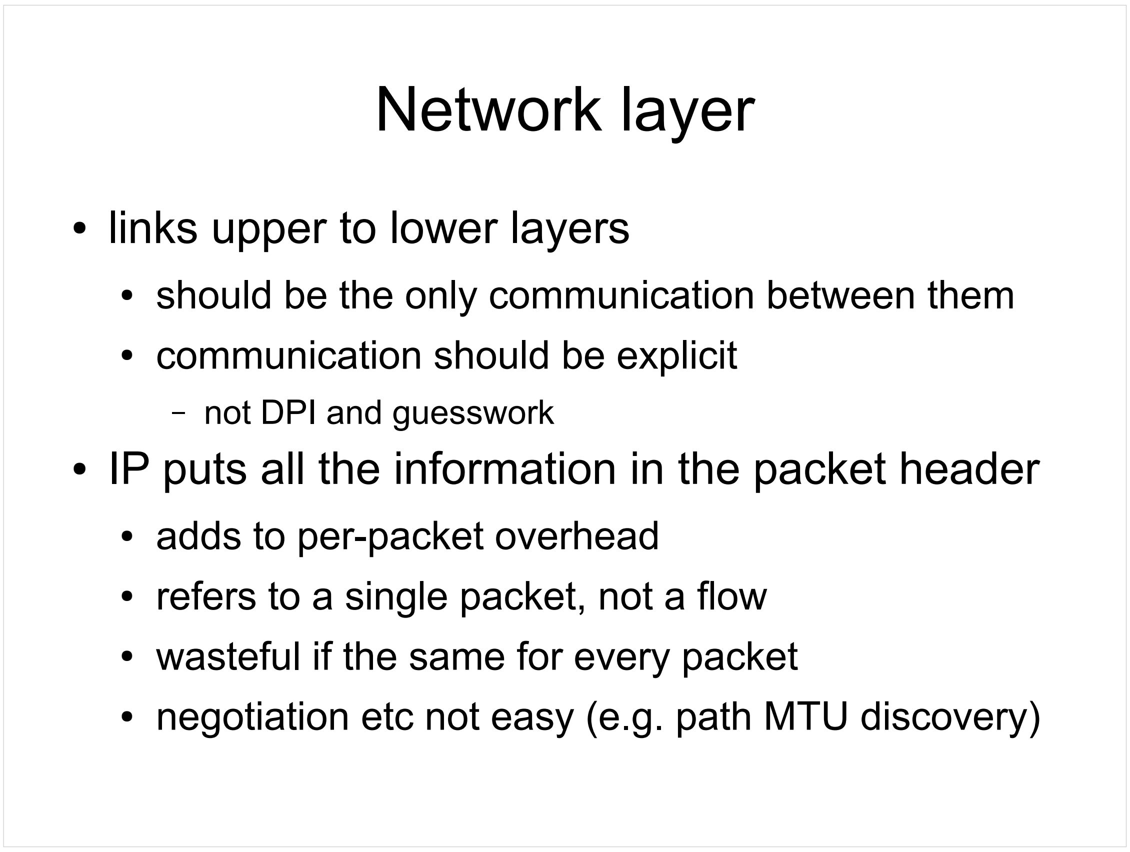
There can be many different higher-layer protocols which use the service it provides; in the current Internet these include TCP, UDP, and RTP.

The route the data follows will usually pass through several different network elements, which may use different lower-layer technologies such as WiFi (IEEE 802.11), Ethernet (802.3), ADSL, and SDH.

To aid migration, FN can use IP as a lower layer (carrying FN data units over an IP network) and as an upper layer (carrying IP datagrams across an FN infrastructure); it can also interface directly to TCP and UDP, and to protocols that run over UDP.

- like ISO 668, 1161, ...

The pictures show another kind of global network in which interoperability is assured by conformance to ISO standards.
The lower layers include lorry, train, and ship.
Containers convey many different kinds of freight, but are all handled in the same way and the service they receive does not depend on the contents.
At transshipment points, the container's identifier is looked up in a data base to find where it needs to go next; the route will have been planned beforehand.



Network layer

- links upper to lower layers
  - should be the only communication between them
  - communication should be explicit
    - not DPI and guesswork
- IP puts all the information in the packet header
  - adds to per-packet overhead
  - refers to a single packet, not a flow
  - wasteful if the same for every packet
  - negotiation etc not easy (e.g. path MTU discovery)

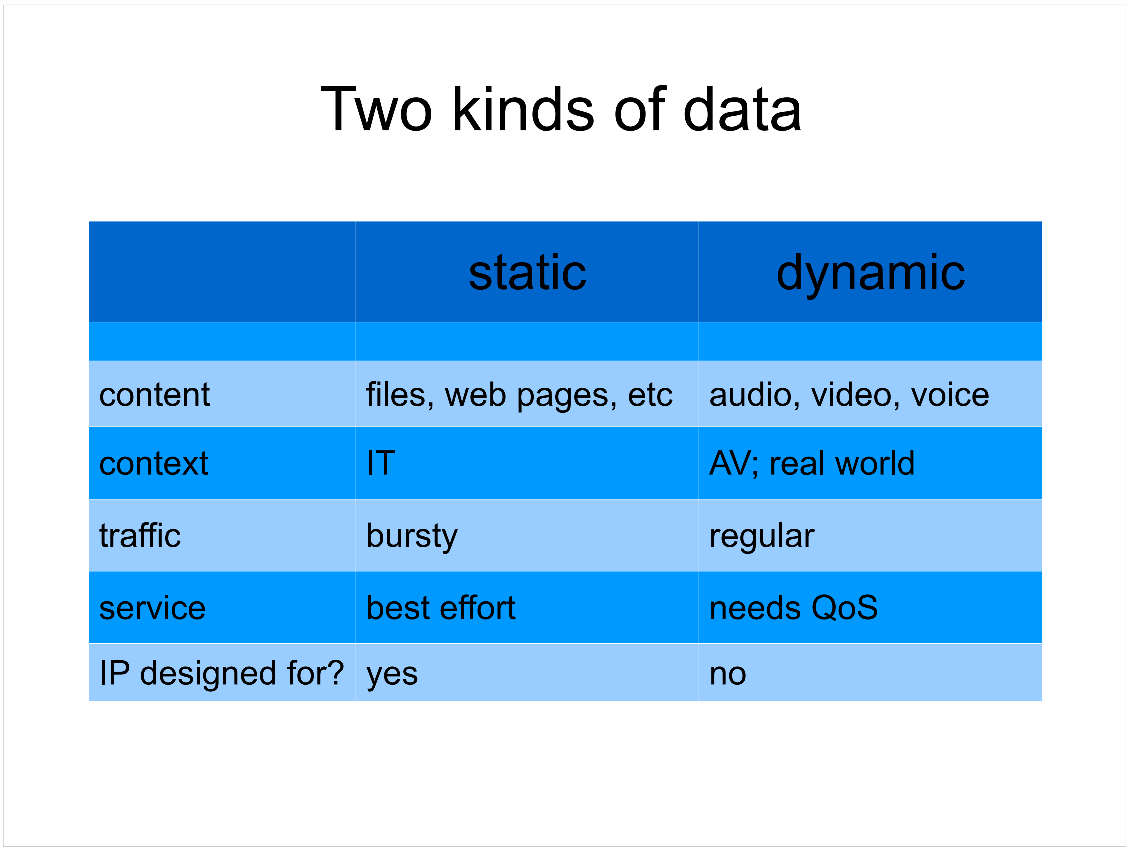
The upper layers should not need to know which kinds of lower layer the packets will pass across; thus the network layer needs to provide a standardised way for the upper layers to exchange information with the lower layers (switches and links) about the service the packets should receive.

In current networks, deep packet inspection is often used by the lower layers to find what is in a packet and from that to guess what service it should receive. The information that can be passed explicitly is limited to those items for which a field is provided in the packet header; adding more fields would increase per-packet overheads.

Often, the application will be sending a succession of packets (a “flow”), and will need to pass information about the flow rather than about individual packets.

Some of the information that takes up space in a packet header may be redundant because it is always the same; RTP header compression, which is used to overcome this problem, adds complexity.

With IPv6, the maximum size of packet that can be transmitted must be discovered by trial and error; the FN control protocols will report it explicitly.



Two kinds of data

	static	dynamic
content	files, web pages, etc	audio, video, voice
context	IT	AV; real world
traffic	bursty	regular
service	best effort	needs QoS
IP designed for?	yes	no

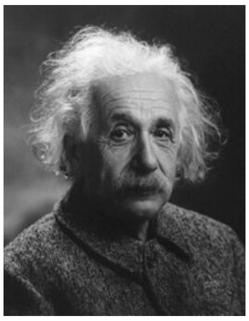
The information carried by communications systems can be divided into two classes.

One is static objects which are encoded as a bit string which needs to be copied from one place to another. Examples are files and web pages. When the requirement arises, the whole bit string is available to be transmitted, and the task is complete when an exact copy of the whole string has been received at the destination. The only timing requirement is to complete the transfer as quickly as possible. Thus the data flow is bursty and a best-effort service is appropriate.

The other is typified by signals that represent a time-varying physical quantity that is being measured by a sensor, such as sound (from a microphone) or video (from a camera). In this case the data will become available in small amounts at regular intervals, and for many applications the delay from when a piece of data is generated until it is available at the receiving end must be kept within limits specified by the application. Thus there needs to be an agreement between the application and the network that the application will send packets at regular intervals and the network will deliver them within a specified time. Rendering the data requires not only the but string but also a good-quality clock.

Content-centric addressing is likely to be appropriate for static objects, whereas for dynamic data, location will often be more relevant, for instance when using CCTV to see whether there are traffic queues on a particular road.

Two kinds of flow



- Synchronous
  - appropriate for dynamic data
  - one-to-many
  - packets sent at regular intervals
  - QoS guarantees (if supported by lower layers)
- Asynchronous
  - appropriate for static data
  - one-to-one or many-to-one
  - best-effort service

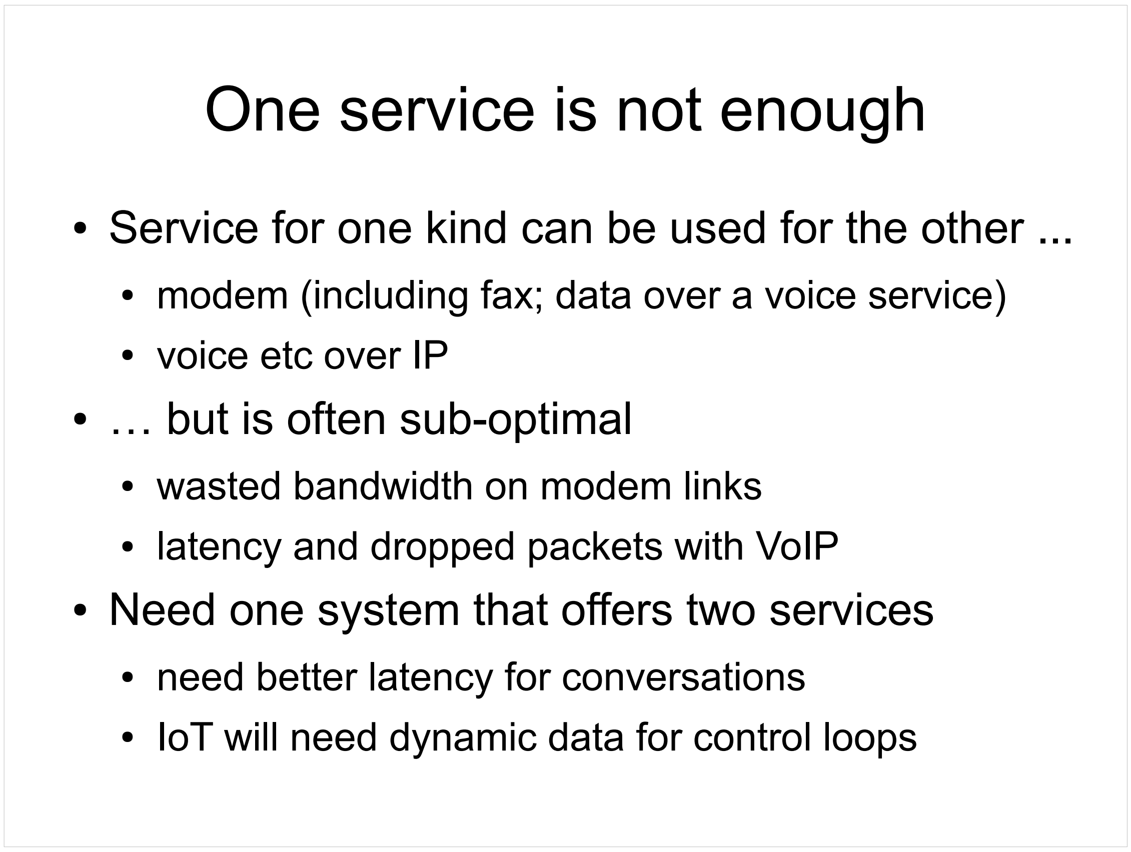
Einstein said “Make everything as simple as possible, but not simpler.”

FN supports two kinds of flow. (ATM supported four.) Usually, both kinds will be packet-switched, with synchronous packets having priority.

Synchronous flows can be scheduled in a way that means the packets do not have to be put into queues; this makes it easy to copy a stream to several outputs. It also makes traffic shaping and policing (ensuring a sender does not exceed the capacity it has negotiated) easy.

The control protocols report the performance the network is able to achieve. On a packet network that tightly synchronises synchronous flows the performance will be close to that of an analogue network or an all-optical network, while on an unmanaged IP network large delays and packet loss may occur; the receiving application can configure buffering etc to match the declared performance.

Packets on asynchronous flows are queued in the same way as on connectionless networks, except that there is no need for multiple queues per port because high-priority traffic can use synchronous flows. Multicasting is also not supported, because it would increase complexity and most requirements for it involve synchronous flows. However, the routing table can be set up such that packets on several different incoming flows are forwarded to the same outgoing flow; this is used for the shared flows described on the slide titled “Fast set-up for asynchronous”.



One service is not enough

- Service for one kind can be used for the other ...
  - modem (including fax; data over a voice service)
  - voice etc over IP
- … but is often sub-optimal
  - wasted bandwidth on modem links
  - latency and dropped packets with VoIP
- Need one system that offers two services
  - need better latency for conversations
  - IoT will need dynamic data for control loops

For some applications which send data of one kind, a service intended for the other kind may meet the requirements, or may be usable but offer poor quality, or may be unusable.

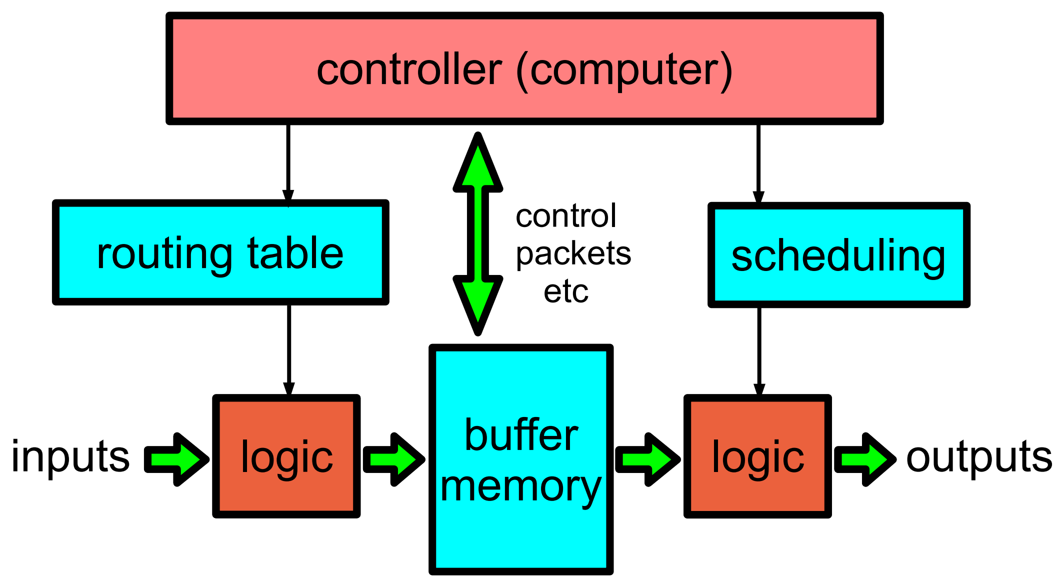
Thus data can be sent over the analogue voice telephone system, but bandwidth will be wasted if there are periods when there is nothing to send. Similarly, dynamic data can be sent over a best-effort service but will suffer delays and lost packets.

Operators will prefer to have one system to maintain rather than two, but the system needs to be able to satisfy all requirements. Thus, Internet traffic was originally carried by modems over the telephone network but, when the service provided became inadequate for it, broadband networks were introduced. The prospect of using the IP network for everything was attractive, but in practice significant parts of the voice network have had to be retained, for instance to support per-call charging.

Users have learnt to cope with the increasing delays on voice calls, but for some applications (for instance in telemedicine, in live broadcasting, and in videoconferencing) the service provided to dynamic data on the current Internet is inadequate.

ATM would have been a single system offering both kinds of service, but failed in the marketplace for other reasons.

Switch structure



All switching technologies use a structure similar to this. The routing table shows what should be done with each incoming packet, based on information such as destination address or virtual channel number. The “scheduling” may control how the output logic chooses what to transmit.

Forwarding of packets is done by logic (hardware) which can run at wire speed (i.e. keep up with the fastest rate at which packets can be transmitted). More complex operations, such as processing packets for control protocols, are done by software in the controller; usually these operations are much slower than wire speed.

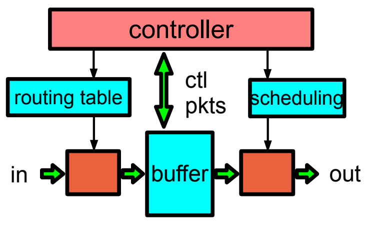
With IP, packets whose destination is not in the routing table are processed by the controller, also packets for protocols such as ARP.

When memory and logic were expensive and data links were slow, all packets were processed by the controller.

With ISDN, the D-channel (Q.931) data is passed to the controller.

With ATM (Asynchronous Transfer Mode), cells on VCIs 1 to 31 are passed to the controller.

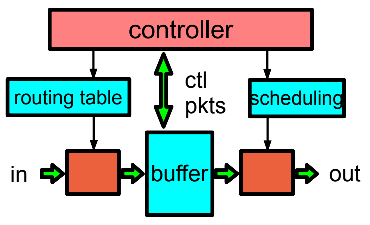
Signalling



- IP: connectionless
  - routing information in packet header
    - includes IPv4 or IPv6 address
  - packets with no table entry passed to controller
  - data waits while controller decides route
    - e.g. during ARP transaction

In IP networks, there is no “set-up” process before packets are transmitted; it is only when a packet arrives with a destination address which is not in the routing table that the controller process which fills in the routing table entry is invoked. The packet is therefore held in a buffer until that process, which may require exchange of protocol packets with other network elements, is complete.

Signalling



- FN: separate protocol to set up route
  - address in control message, not in packet
    - allows new forms of addressing to be used
    - allows different forms of routing technology to be used
  - routing table entry written before data sent
    - data packets never need to wait for controller

With FN, the information needed to route a flow is in a control message. It includes the destination address, which may take a variety of forms including names such as URIs. It also includes QoS parameters and information about the data format, and can include other information such as for per-call charging. Thus it combines the functions of a number of protocols in IP networks (such as DNS, SIP, SDP, and RSVP) as well as others which will be needed if the network is to provide all telecommunications services within a single system.

Connection-oriented paradigm

- Allows QoS etc negotiation
  - and other facilities such as per-call billing
- Connection-oriented ≠ TDM
  - though FN supports use of TDM and WDM circuits
- “Link” between network elements may be:
  - point-to-point connection or shared media (e.g. WiFi)
  - legacy (sub-)network, including connectionless
- Label switching more efficient
  - less logic in switches → lower power

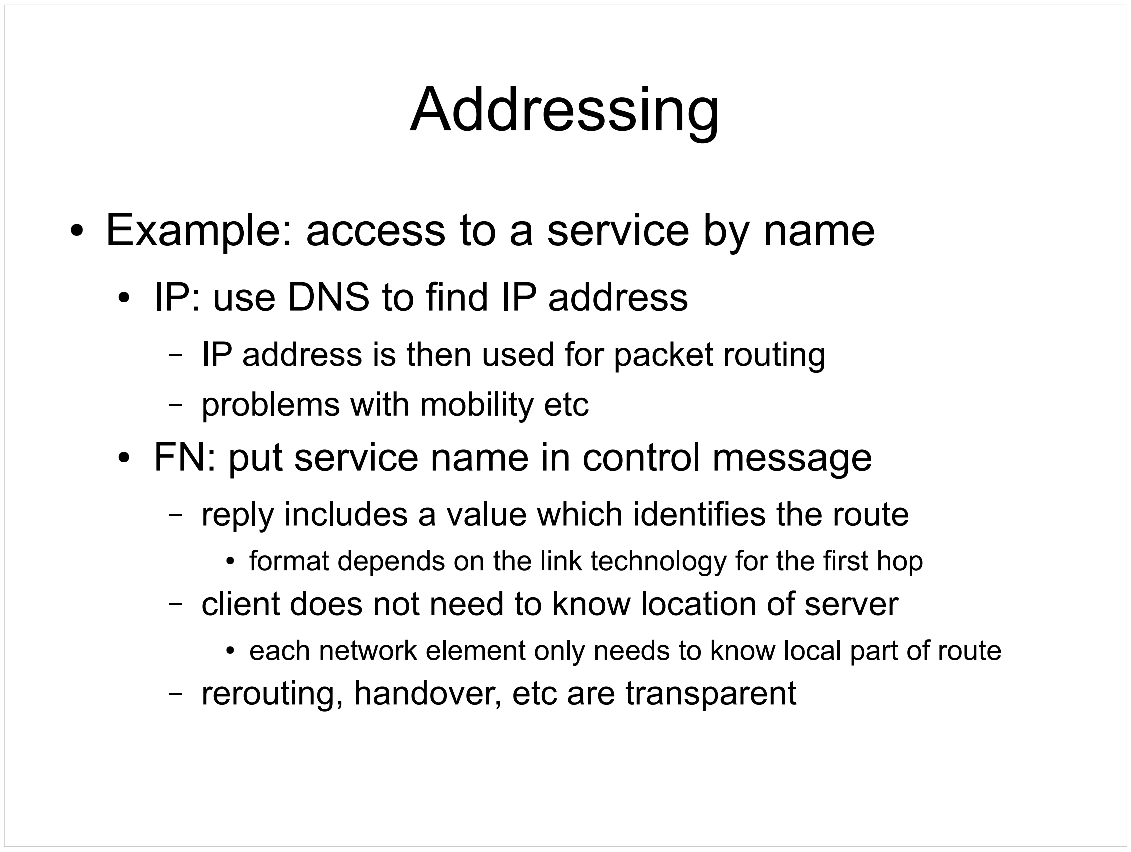
As part of the process of setting up a route, the application can co-operate with the network to find the best bit rate to use, and the network can inform the application of other parameters such as latency. The endpoints can also use this same process to agree the coding to be used, and authentication can be done at this stage.

The model also supports per-call billing as in telephone networks. This can be used to remunerate content owners without needing the user to make direct payments to them, as with premium-rate telephone calls. It can also allow different levels of service to be offered, for instance a higher price for better-quality video.

FN is expected to be packet-switched. However, where appropriate the network may offer to, for example, connect a WDM lambda over fibre; this may be switched in the optical domain, which would be significantly more efficient than packet switching which currently has to be done in the electronic domain.

A link between units in FN can be a network using legacy technology. This provides a migration path by tunnelling FN calls across existing networks.

Where links are new technology, the routing information in the packet header can simply be a “label” which is the address in the routing table of the entry for the flow; this entry can simply contain the output port number and the label to use for the next hop.

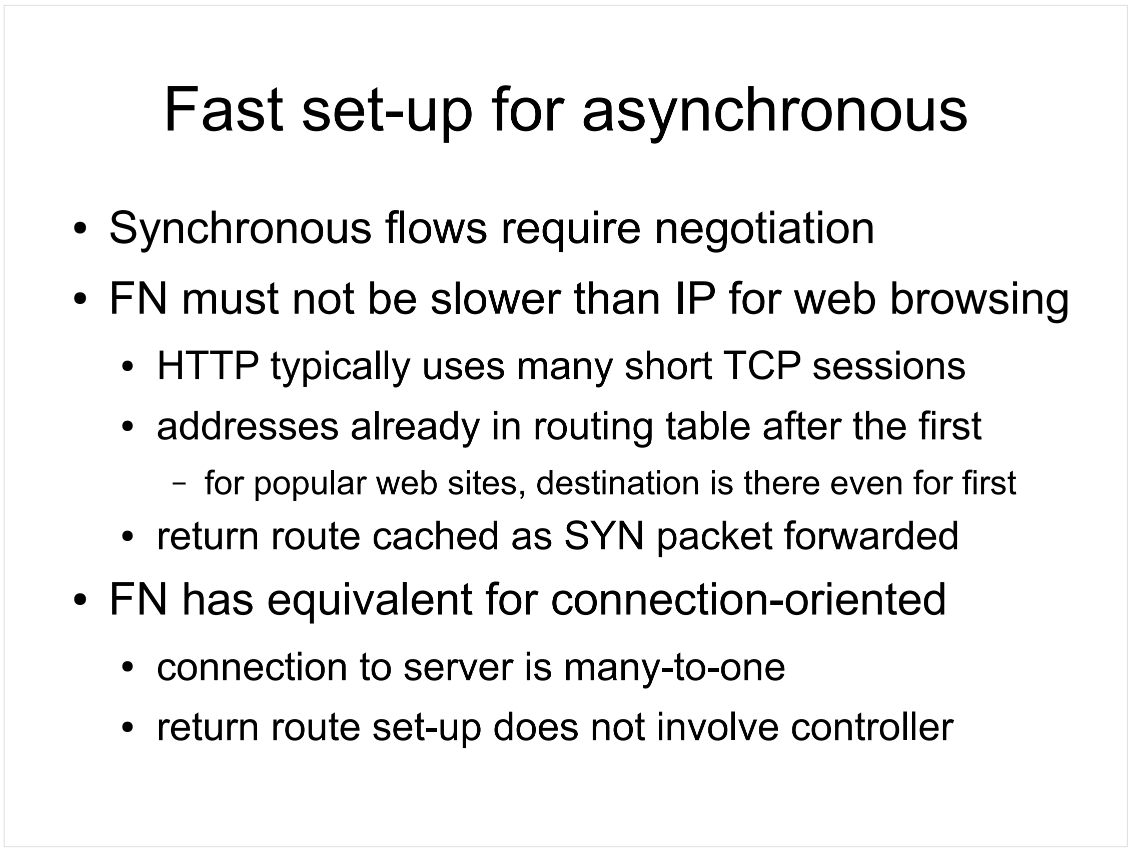


Addressing

- Example: access to a service by name
  - IP: use DNS to find IP address
    - IP address is then used for packet routing
    - problems with mobility etc
  - FN: put service name in control message
    - reply includes a value which identifies the route
      - format depends on the link technology for the first hop
    - client does not need to know location of server
      - each network element only needs to know local part of route
    - rerouting, handover, etc are transparent

With current networks, DNS can be used to find the IP address of an interface to a piece of equipment which provides the required service, or has a copy of the required data. Subsequent communication is therefore tied to that interface, even if the device has other interfaces which are less heavily loaded; and additional protocols are required to support handover if the destination is a mobile device.

With FN, the application simply uses the name as the called address in the control packet that requests connection set-up. The reply shows what value needs to be put in the packet headers, but this value only needs to identify the route to the local switch. Handover of mobile devices is, again, handled locally, without affecting the rest of the route.



Fast set-up for asynchronous

- Synchronous flows require negotiation
- FN must not be slower than IP for web browsing
  - HTTP typically uses many short TCP sessions
  - addresses already in routing table after the first
    - for popular web sites, destination is there even for first
  - return route cached as SYN packet forwarded
- FN has equivalent for connection-oriented
  - connection to server is many-to-one
  - return route set-up does not involve controller

One criticism that has been made of connection-oriented communications is the time required to set up a call. Partly that has been because systems were poorly implemented, but there are some use cases in which many sessions are set up and torn down, and an action that requires intervention from the controller will always take longer (though not necessarily significantly longer) than one that can be done by the logic.

In the case of synchronous flows, the negotiation and resource allocation which are part of call set-up are necessary and the length of time they take is not likely to have a significant adverse effect on the user experience.

Sending a packet to a new destination address on an IP network requires a similar process to setting up the route for an asynchronous flow on a connection-oriented network. Thereafter, however, the route will be cached in the IP network's routing tables so subsequent packets do not require intervention from the controller. The route remains in the tables after the TCP session is terminated, and so can be used for subsequent sessions; it can also be used by other clients.

The mechanism outlined in 8.2 of 29181-3 (see link on last slide) allows connection-oriented networks to cache routes, and share them, in the same way.

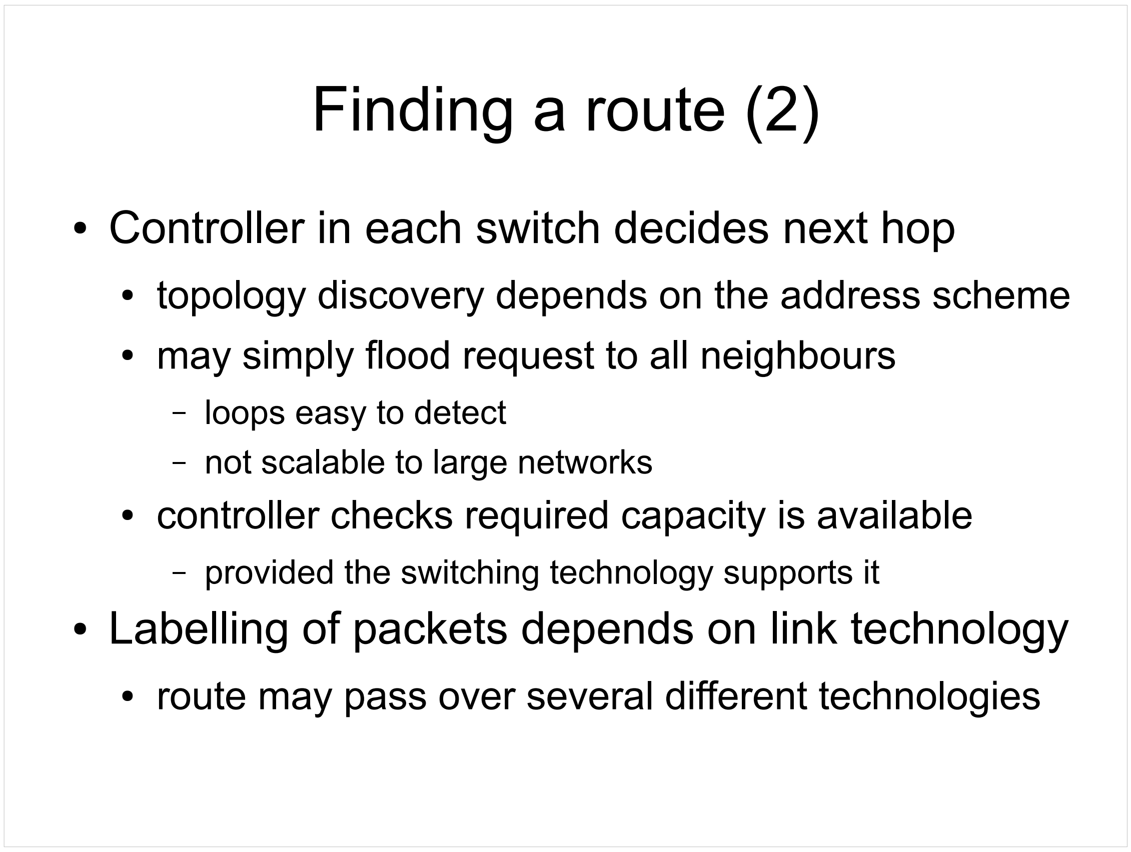
Finding a route (1)

- Application sends request to local controller
  - includes address (or other identification) of target
    - which may be service or content
  - also includes globally-unique “call identifier”
- Multiple addressing schemes
  - must support legacy schemes, e.g. IPv4, IPv6
    - including URLs etc
  - must allow new schemes to be added later
    - no change to routing logic required

Connection of a call begins with the application building a request message and sending it to the controller of the switch to which the unit running the application is connected. This message (and its reply) will carry information for (and from) both the network and the remote application. The information exchanged with the network can include traffic parameters (such as: whether synchronous or asynchronous data; packet size; rate and latency if synchronous) and authentication and billing information. The information exchanged with the remote application can include identification of formats and protocols, and user authentication.

Each call has a unique identifier which has global scope. This allows loops in the route to be detected and is also useful for network management.

The messages are processed by software in the switch controllers, so it is easy to support multiple addressing schemes. Supporting legacy schemes such as IPv4 will aid migration. Supporting URLs will remove the need for end systems to use DNS. New addressing schemes can be added to support future requirements without needing to update the logic in switches (unlike the change from IPv4 to IPv6).



Finding a route (2)

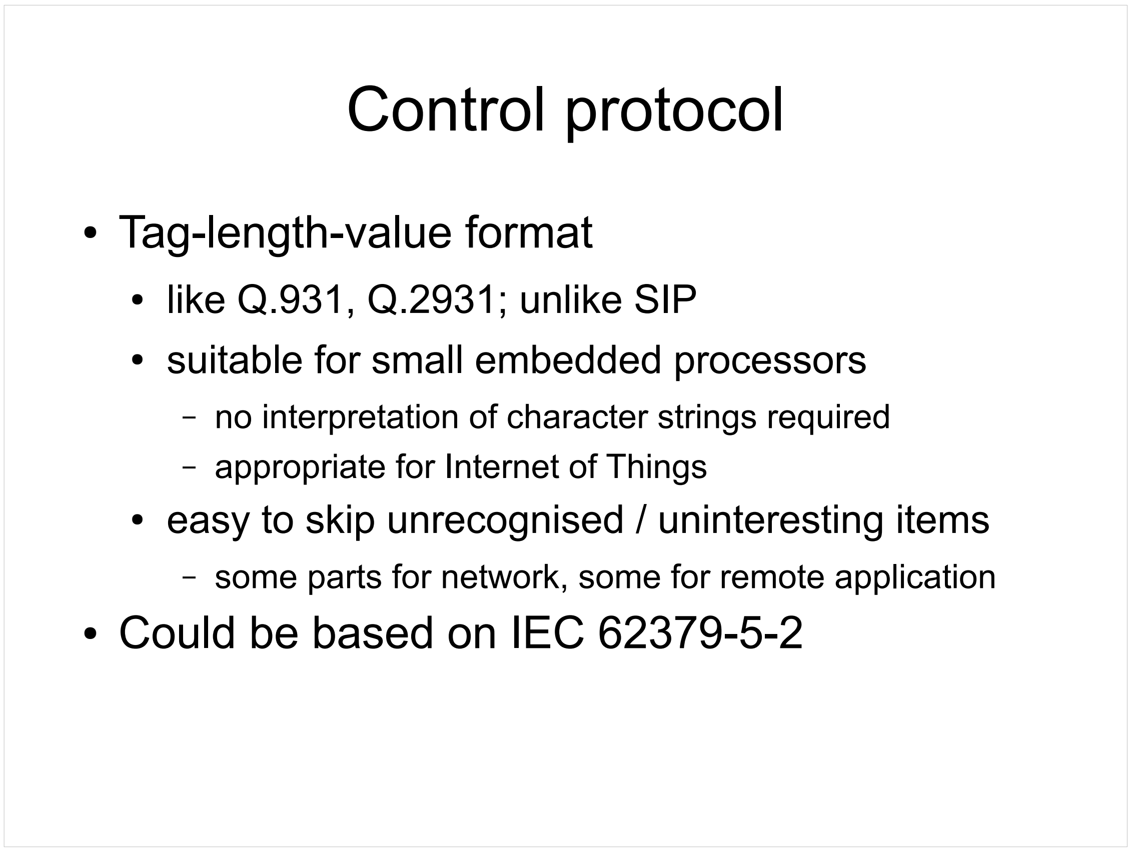
- Controller in each switch decides next hop
  - topology discovery depends on the address scheme
  - may simply flood request to all neighbours
    - loops easy to detect
    - not scalable to large networks
  - controller checks required capacity is available
    - provided the switching technology supports it
- Labelling of packets depends on link technology
  - route may pass over several different technologies

The controller software looks at the destination address (and maybe other information) in the message and forwards the message to one or more neighbours that are in some sense “nearer” to the destination. How it knows which is “nearer” will usually depend on the address scheme and the provision it includes for distributing topology information, though in some subnetworks there will be a “gateway” through which all requests for non-local addresses can be routed.

In small networks, connection requests can simply be flooded to all neighbours; a switch can easily detect loops by comparing the call identifier with existing routes passing through it, so there is no need for a “spanning tree” protocol to disable links.

When a request for a synchronous flow is forwarded, resource is reserved for it (but not at that stage set up in the routing tables). Where there is a choice of routes, the most lightly loaded can be chosen. However, this is not possible if the underlying network does not support resource reservation, in which case the reply will show that no performance guarantees are possible.

The information in the encapsulation of a packet that is used by the logic to decide where to forward it depends on the link technology. For instance, on an Ethernet subnetwork it will be the 48-bit MAC address of the neighbouring switch together with an additional field selecting an entry in the neighbour's routing table. This is similar to the way that IP packets are forwarded inside Ethernet packets.



Control protocol

- Tag-length-value format
  - like Q.931, Q.2931; unlike SIP
  - suitable for small embedded processors
    - no interpretation of character strings required
    - appropriate for Internet of Things
  - easy to skip unrecognised / uninteresting items
    - some parts for network, some for remote application
- Could be based on IEC 62379-5-2

29181-3 lists requirements for the control messages.

One is that the format should be suitable for parsing by small processors, to minimise the processing power required in simple devices. The format should also make it easy to skip parts of the message that are not recognised (perhaps because they have been standardised since the software was last updated) or are not of interest (in the case of a switch, this would include information intended for the remote application). The messages do not need to be directly human-readable, though it should be easy to expand them into a text form for use in diagnostic messages.

The tag-length-value format used in ITU-T signalling and many other protocols would be suitable; the text form used in SIP would not.

IEC PT 62379 is developing (in its Part 5-2) a protocol which meets these requirements; a link to the drafts is on the next slide.

Links to drafts

- http://www.iec62379.org/FN-standardisation.html
  - includes link to current draft of 29181-3
  - also links to IEC 62379-5 drafts

Nine Tiles

http://www.ninetiles.com

mailto:j@ninetiles.com