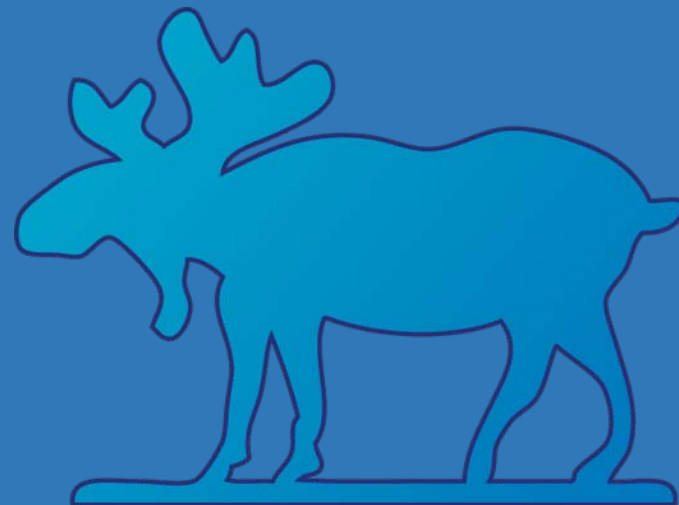


Addressing the Scalability of Ethernet with MOOSE



Malcolm Scott, Andrew Moore and Jon Crowcroft
University of Cambridge Computer Laboratory

Ethernet in the data centre

- **1970s protocol; still ubiquitous**

- Usually used with IP, but not always (ATA-over-Ethernet)



- **Density of Ethernet addresses is increasing**

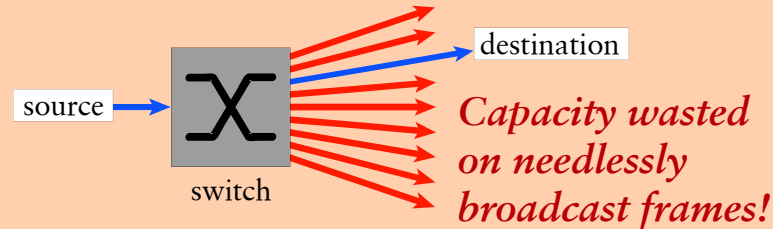
- Larger data centres, more devices, more NICs
- Virtualisation: each VM has a unique Ethernet address
 - (or more than one!)

Why not Ethernet?

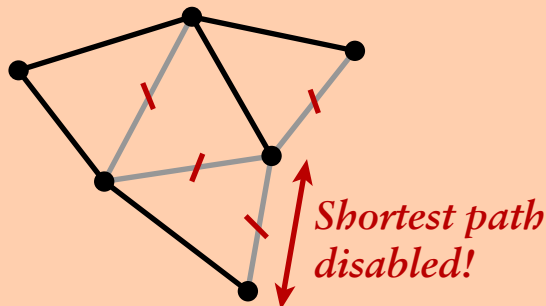
Heavy use of broadcast

Broadcast ARP required for interaction with IP

On large networks, broadcast can overwhelm slower links e.g. wireless



Inefficient routing: Spanning Tree



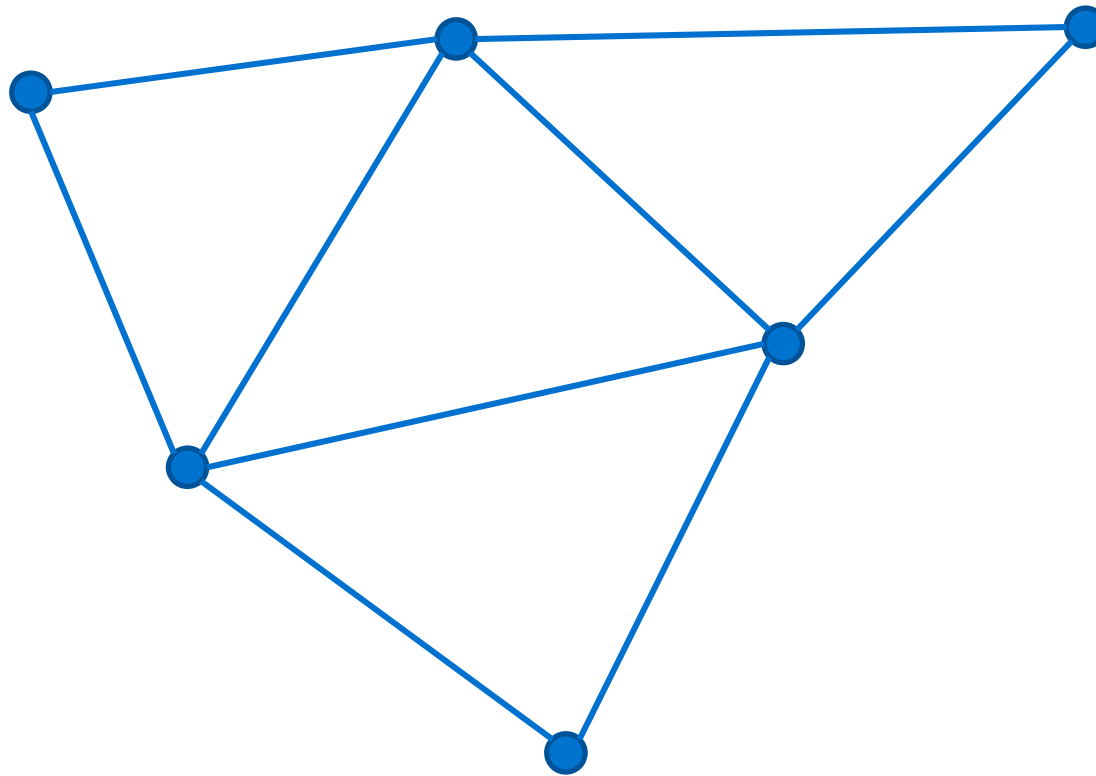
Switches' address tables

MAC address	Port
01:23:45:67:89:ab	12
00:a1:b2:c3:d4:e5	16
...	...

- Maintained by every switch
- Automatically learned
- Table capacity: ~16000 addresses
- *Full table results in unreliability, or at best heavy flooding*

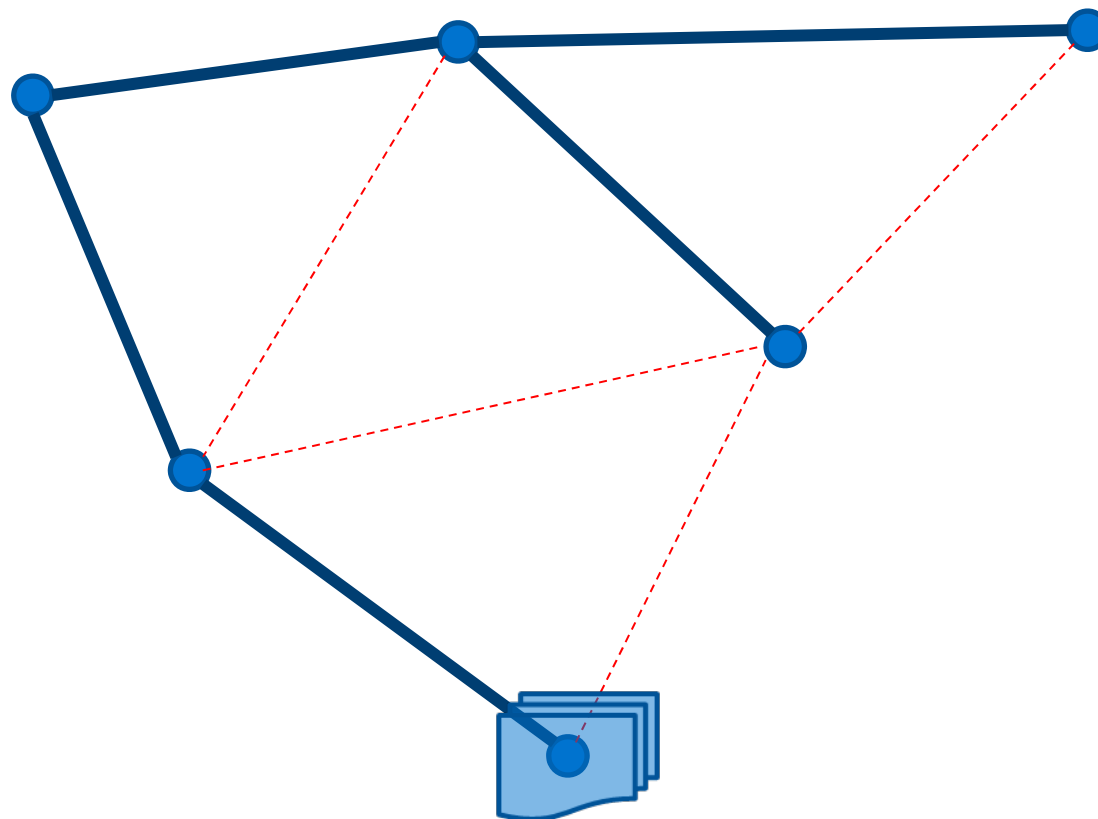


Spanning tree switching illustrated



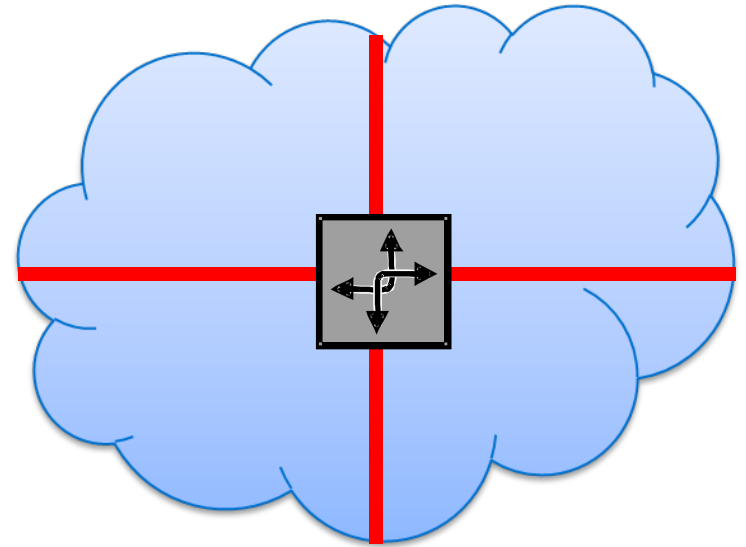


Spanning tree switching illustrated



Ethernet in the data centre: divide and conquer?

- **Traditional solution: artificially subdivide network at the IP layer: subnetting and routing**
 - Administrative burden
 - More expensive equipment
 - Hampers mobility
 - IP Mobility has not (yet) taken off
 - Scalability problems remain within each subnet





Ethernet in the data centre: ...mobility?

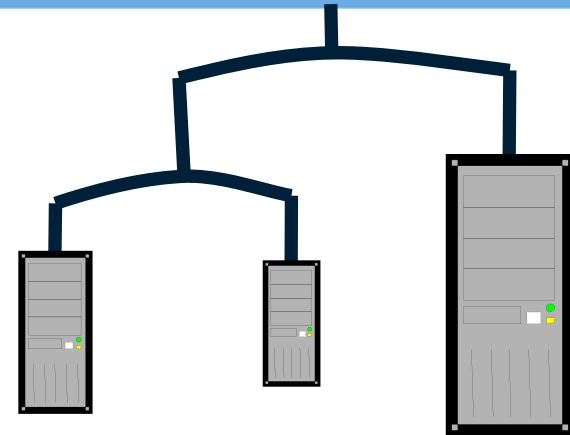
- **Mobility *is* relevant in the data centre**

- Seamless virtual machine migration
- Easy deployment:
no location-dependent configuration

- **...and between data centres**

- Large multi-data-centre WANs are becoming common

- **Ethernet is pretty good at mobility**



Large networks

- **Converged airport network**
 - Must support diverse commodity equipment
 - Roaming required throughout entire airport complex
 - Ideally, would use one large Ethernet-like network
- This work funded by “The INtelligent Airport” UK EPSRC project



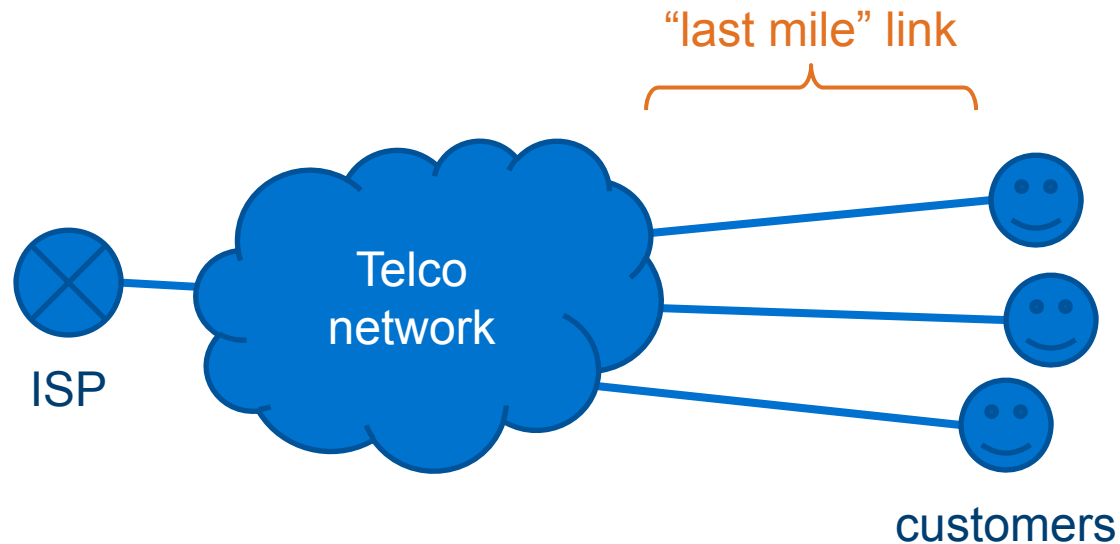
Large networks

- **Airports have surpassed the capabilities of Ethernet**
 - London Heathrow Airport: Terminal 5 alone is too big
 - MPLS-VPLS: similar problems to IP subnetting
 - VPLS adds more complexity:
 - LERs map every destination MAC address to a LSP: up to $O(\text{hosts})$
 - LSRs map every LSP to a next hop: could be $O(\text{hosts}^2)$ in core!
 - Encapsulation does not help

Geographically-diverse networks

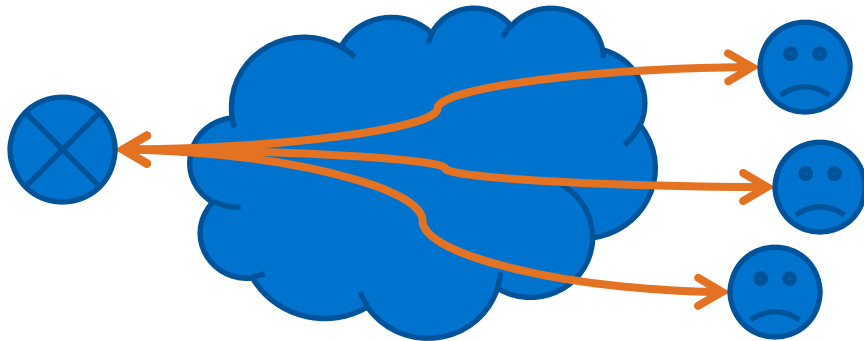
Fibre-to-the-Premises

- Currently, Ethernet is only used for small deployments



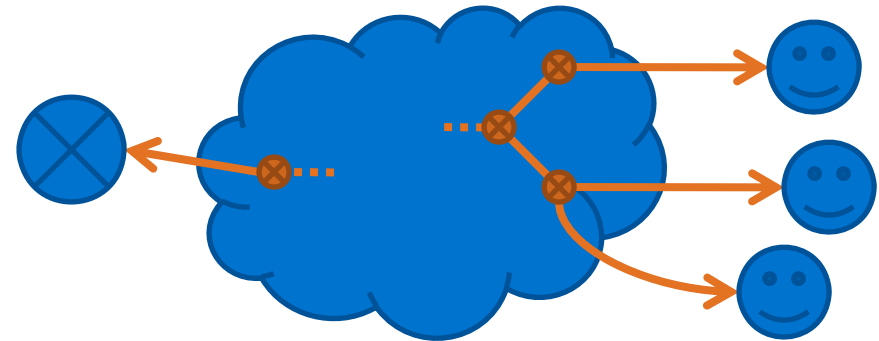
Geographically-diverse networks

Now:



- Everything goes via circuit to ISP
- Legacy reasons (dial-up, ATM)
- Nonsensical for peer-to-peer use
- Bottleneck becoming significant as number of customers and capacity of links increase

Future:



- In the UK: BT 21CN
- Take advantage of fully-switched infrastructure
- Peer-to-peer traffic travels directly between customers
- Data link layer protocol is crucial

The underlying problem with Ethernet

MAC addresses provide
no location information

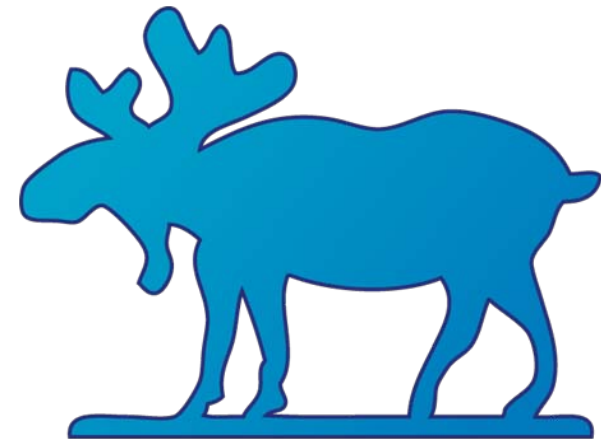
Flat vs. Hierarchical address spaces

- **Flat-addressed Ethernet:** manufacturer-assigned MAC address valid anywhere on any network
 - But every switch must discover and store the location of every host
- **Hierarchical addresses:** address depends on location
 - Route frames according to successive stages of hierarchy
 - No large forwarding databases needed
- **LAAs?** High administrative overhead if done manually

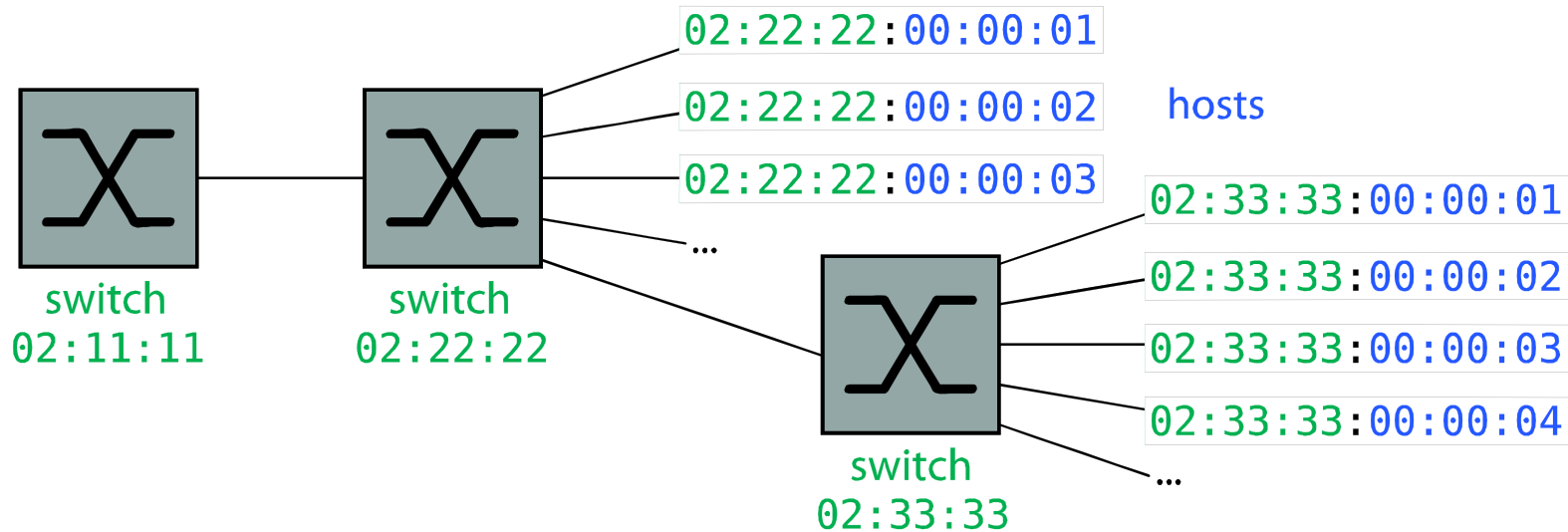
MOOSE: *Multi-level Origin-Organised Scalable Ethernet*

A new way to switch Ethernet

- Perform MAC address rewriting on ingress
 - Enforce dynamic hierarchical addressing
 - No host configuration required
-
- *Good platform for shortest-path routing*
 - ***Appears to connected equipment as standard Ethernet***



MOOSE: *Multi-level Origin-Organised Scalable Ethernet*

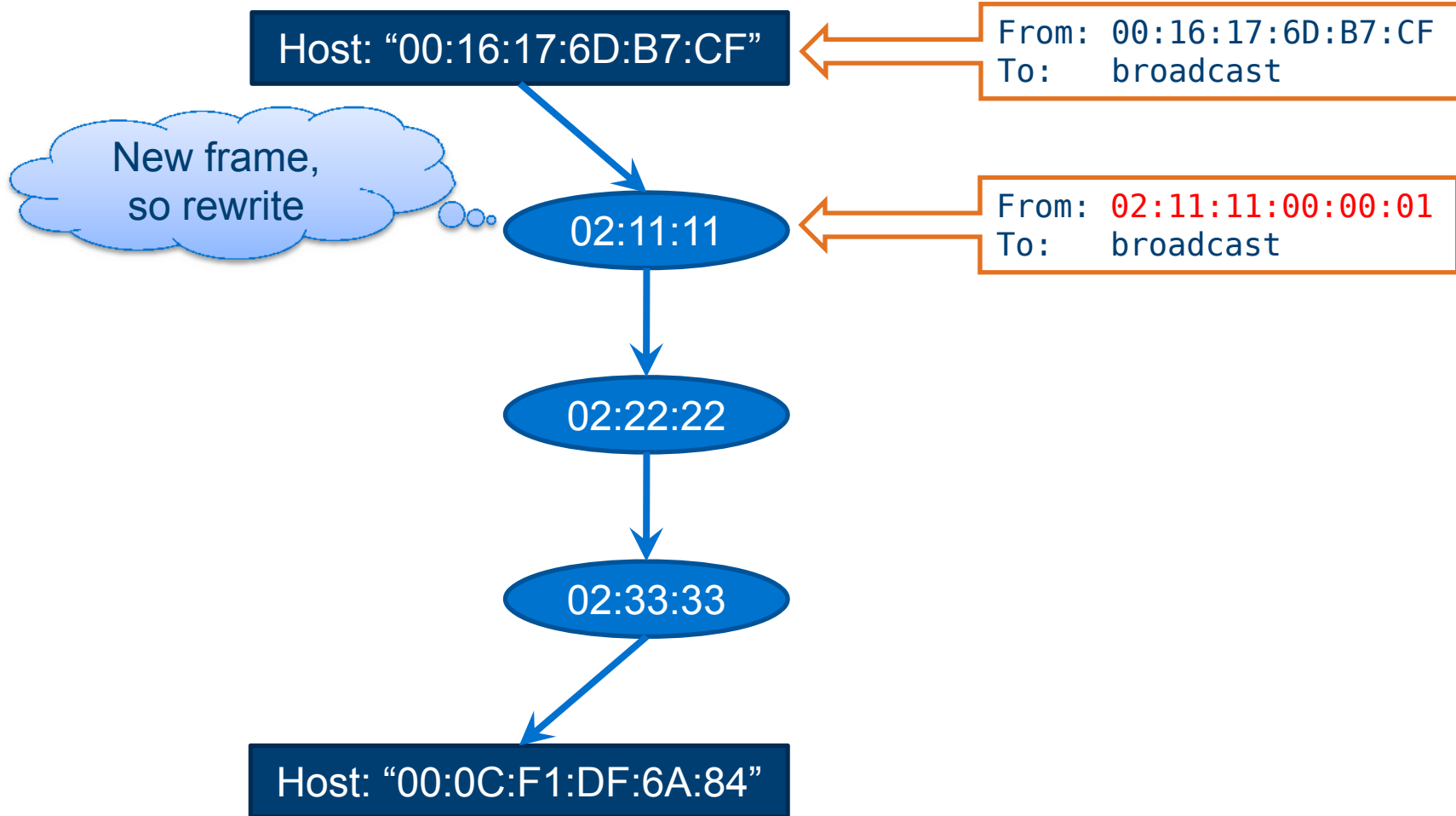


- **Switches assign each host a MOOSE address = switch ID . host ID**
(MOOSE address must form a valid unicast LAA: two bits in switch ID fixed)
- Placed in source field in Ethernet header as each frame enters the network
(no encapsulation, therefore no costly rewriting of destination address!)

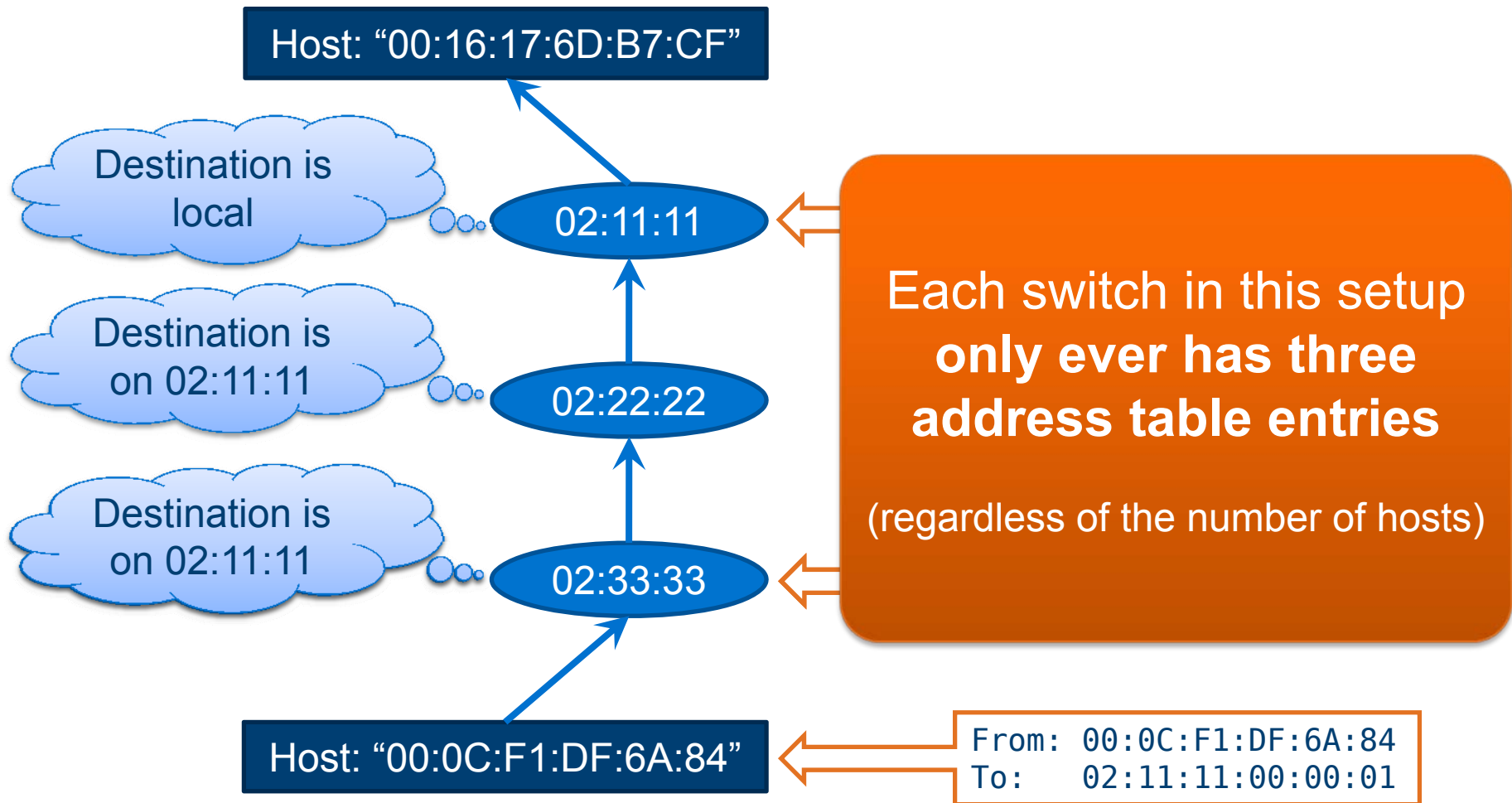
Allocation of host identifiers

- **Only the switch which allocates a host ID ever uses it for switching** (more distant switches just use the switch ID)
- Therefore the detail of how host IDs are allocated can vary between switches
 - Sequential assignment
 - Port number and sequential portion
(isolates address exhaustion attacks)
 - Hash of manufacturer-assigned MAC address
(deterministic: recoverable after crash)

The journey of a frame

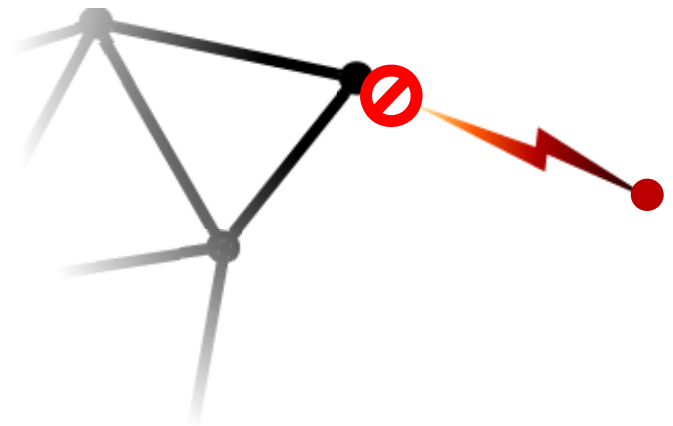


The return journey of a frame



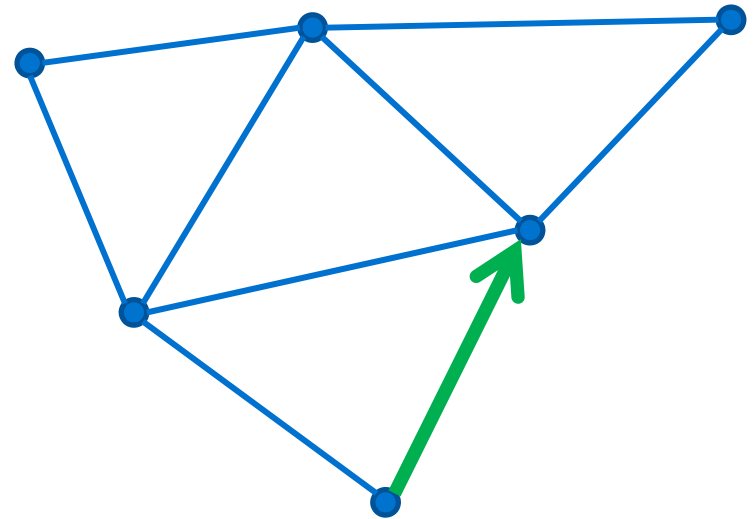
Security and isolation benefits

- **The number of switch IDs is predictable, unlike the number of MAC addresses**
 - Address flooding attacks are ineffective
 - Resilience of dynamic networks (e.g. wireless) is increased
- **Host-specified MAC address is not used for switching**
 - Spoofing is ineffective



Shortest path routing

- **MOOSE switch \approx layer 3 router**
 - One “subnet” per switch
 - `02:11:11:00:00:00/24`
 - Don’t advertise individual MAC addresses!
 - Run a routing protocol between switches, e.g. OSPF variant
 - OSPF-OMP may be particularly desirable: optimised multipath routing for increased performance



Beyond unicast

- **Broadcast: unfortunate legacy**

- DHCP, ARP, NBNS, NTP, plethora of discovery protocols...
- Deduce spanning tree using reverse path forwarding (PIM): no explicit spanning tree protocol
- Can optimise away most common sources, however

- **Multicast and anycast for free**

- SEATTLE suggested generalised VLANs (“groups”) to emulate multicast
- Multicast-aware routing protocol can provide a true L2 multicast feature

ELK: Enhanced Lookup

- **General-purpose directory service**

- Master database: held on one or more servers in core of network
- Slaves can be held near edge of network to reduce load on masters
 - Read: anycast to nearest slave
 - Write: multicast to all masters
 - Entire herd of ELK kept in sync by masters via multicast + unicast



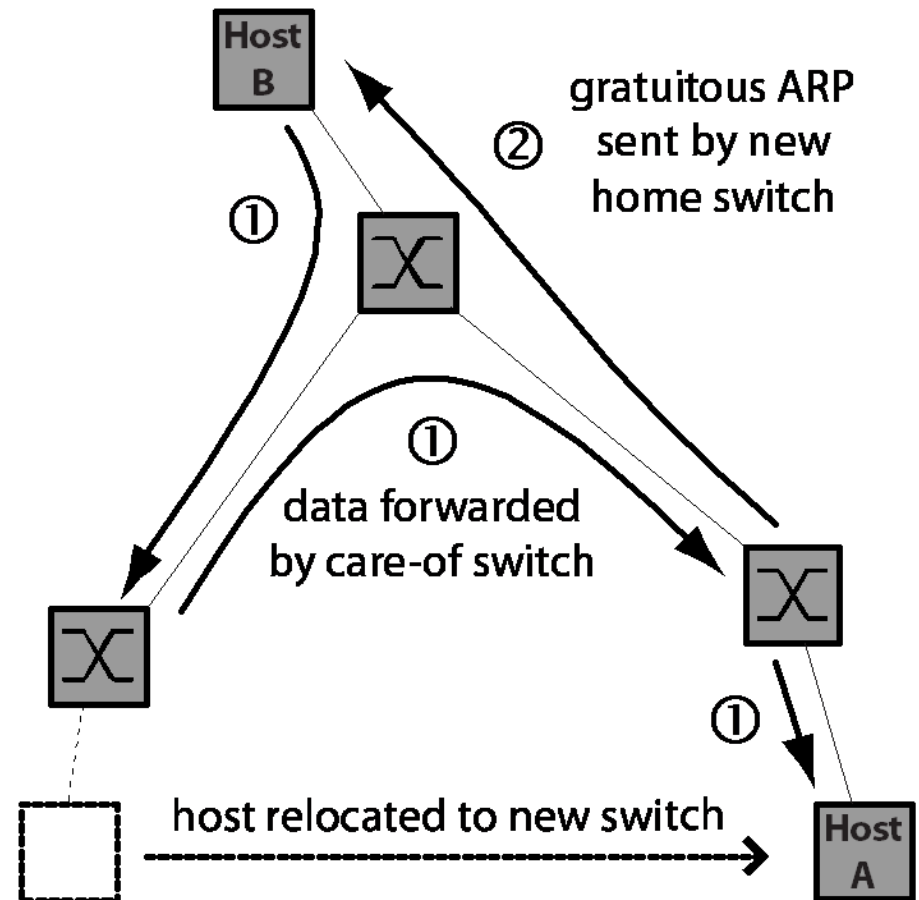
ELK: Enhanced Lookup

- **Primary aim: handle ARP & DHCP without broadcast**
 - ELK stores (MAC address, IP address) tuples
 - Learned from sources of ARP queries
 - Acts as DHCP server, populating directory as it grants leases
 - Edge switch intercepts broadcast ARP / DHCP query and converts into anycast ELK query
 - ELK is not guaranteed to know the answer, but it usually will
 - (ARP request for long-idle host that isn't using DHCP)

Mobility

If a host moves, it is allocated a new MOOSE address by its new switch

- Other hosts may have the old address in ARP caches
- 1) **Forward frames**, IP Mobility style
(new switch discovers host's old location by querying other switches for its real MAC address)
- 2) **Gratuitous ARP**,
Xen VM migration style



Related work

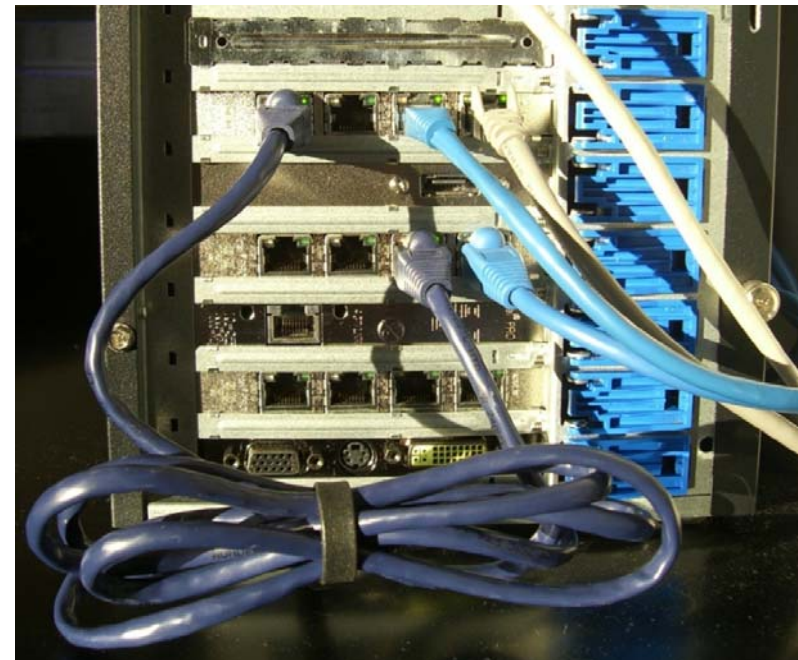
- **Encapsulation**
(MPLS-VPLS, IEEE TRILL, ...)
 - Destination address lookup:
Big lookup tables
- **Domain-narrowing**
(PortLand – Mysore *et al.*, UCSD)
 - Is everything *really* a strict tree topology?
- **Complete redesign**
(Myers *et al.*)
 - To be accepted, must be Ethernet-compatible
- **DHT for host location**
(SEATTLE – Kim *et al.*, Princeton)
 - Unpredictable performance; topology changes are costly



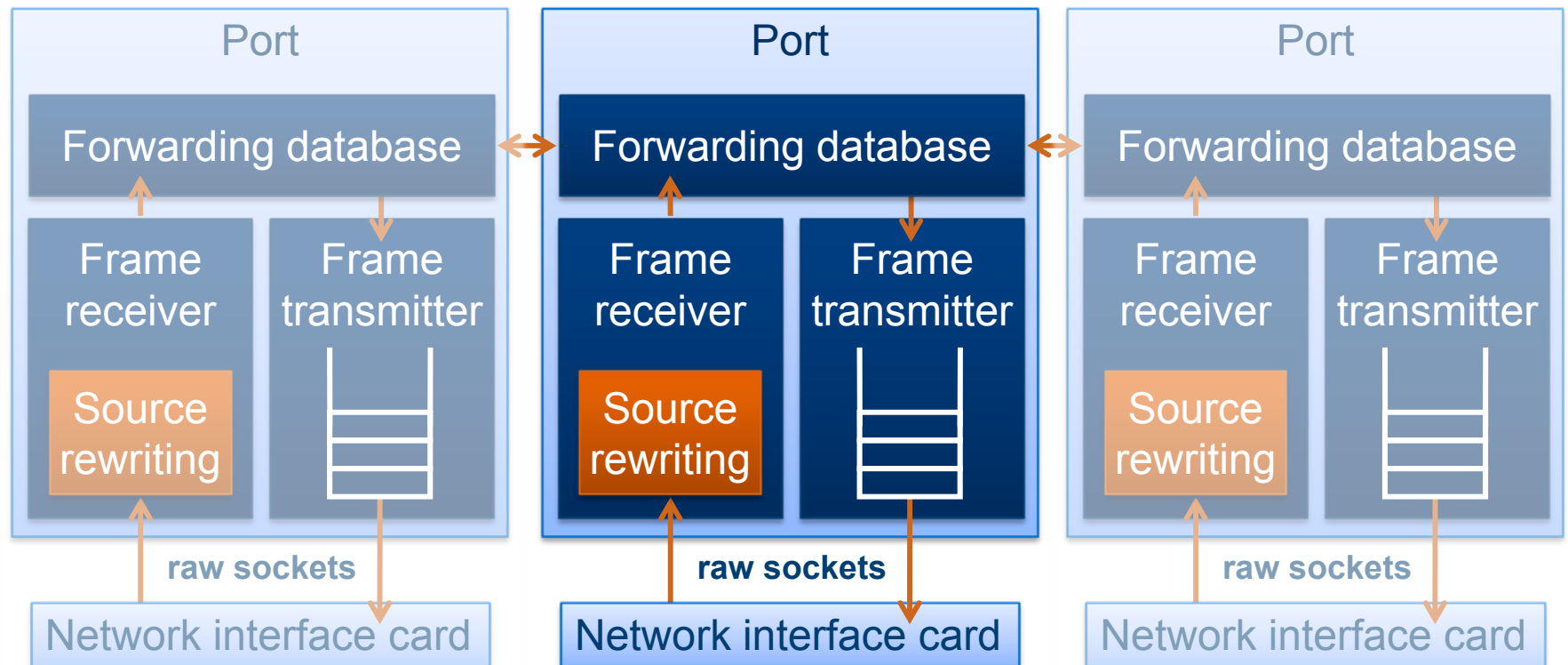


Prototype implementation

- **Proof-of-concept in threaded, object-oriented Python**
 - Designed for clarity and to mimic a potential hardware design
 - Modularity
 - Separation of control and data planes
 - Capable of up to 100 Mbps switching on a modern PC
 - Could theoretically handle very large number of nodes



Prototype implementation: Data plane



Prototype implementation: Data plane

- **Two forwarding databases:**

- Locally-connected hosts (MAC address, host ID, Port)
- Remote switches (switch ID, Port)

- **Inside the Frame Receiver:**

- 1) Received frame from raw socket packaged in Frame object
- 2) DHCP or ARP? Send to control plane (“software”)
- 3) Rewrite source if not already MOOSE
 - Allocate host ID if necessary: port number, sequential ID
- 4) Update locally-connected-host forwarding database
- 5) Consult relevant forwarding database for output Port; enqueue frame with that Port’s Frame Transmitter

Prototype implementation: Control plane

- **Separate thread**
- **Routing protocol: PWOSPF**
 - Only for proof-of-concept: real implementation would likely need OSPF's authentication features etc.
 - Map switch IDs onto PWOSPF's 4-byte address fields by padding RHS with null bytes
 - 02.11.11.00/24
 - Maintain Ports' remote-switches forwarding databases
(*routing tables, really*)

Prototype evaluation

Unmodified PC's ARP cache:

Address	HWtype	HWaddress	Iface
10.100.11.1	ether	02:00:0c:01:00:01	eth1
10.100.11.3	ether	02:00:0a:01:00:01	eth1
10.100.11.4	ether	02:00:0a:03:00:01	eth1
10.100.11.8	ether	02:00:0b:02:00:01	eth1

• Virtual network (Xen)

- Six virtual switches, 10 VMs each: MOOSE vs. Linux bridging
 - Linux bridge FDBs: 60 entries on each switch: **O(hosts)**
 - MOOSE FDBs: 5 switch entries + 10 host entries on each switch: the latter will remain constant in larger deployments, so **O(switches)**

Future work

- **NetFPGA implementation**
 - (Dan Wagner-Hall)

- **Enterprise Ethernet features**
 - Quality-of-Service
 - 802.1Q-compatible VLANs: opportunities to explore

Final thoughts

- **Ethernet: another 35 years?**

- Not an ideal starting point, but it's what we've got
- If it is to last, it needs to scale yet remain compatible
- MOOSE is a simple, novel and easily-implementable approach
 - *Address the cause, not the symptom*

“we choose to achieve reliability through simplicity”
– Robert M. Metcalfe and David R. Boggs

Thank you

<http://www.cl.cam.ac.uk/~mas90/MOOSE/>

Malcolm.Scott@cl.cam.ac.uk

Related work

- **Encapsulation-based solutions:**
(MPLS-VPLS, Hadžić, SmartBridge, Rbridges / IEEE TRILL, ...)
 - Effective shortest-path routing, but...
 - Big lookup tables everywhere
 - Replace one scalability problem with another
- **Complete redesigns:** (Myers et al.)
 - The only “perfect” solution
 - But to be accepted, must be Ethernet-compatible

Related work: domain-narrowing

- **PortLand:** (Mysore *et al.*, UCSD)
 - Observe that data centres are usually “fat trees”
 - Optimise for strict hierarchical network
 - *No provision for other topologies*
 - Real deployments may come unstuck
 - Consider entire network to be a single fabric



Related work: SEATTLE (Kim *et al.*, Princeton)

- **Forward frames through a Distributed Hash Table (DHT)**
 - Elegant idea; effectively solves most of the problems
- **But, likely to cause unpredictable performance**
 - DHTs are variable-latency
 - May forward some hosts' frames through distant, slow switches
 - Cache mitigates this to an extent, but could be flooded



Related work: SEATTLE (Kim *et al.*, Princeton)

- **Topology changes are very expensive**
(when the set of reachable switches changes)
 - *Any* such change leads to DHT reorganisation
 - ...Which involves switches throughout the network
- **Data plane complexity:**
 - SEATTLE switch must do much more for each frame than Ethernet
 - (MOOSE's data plane is quite simple)