# IP ROUTING

## INTRODUCTION TO IP, IP ROUTING PROTOCOLS AND PROXY ARP

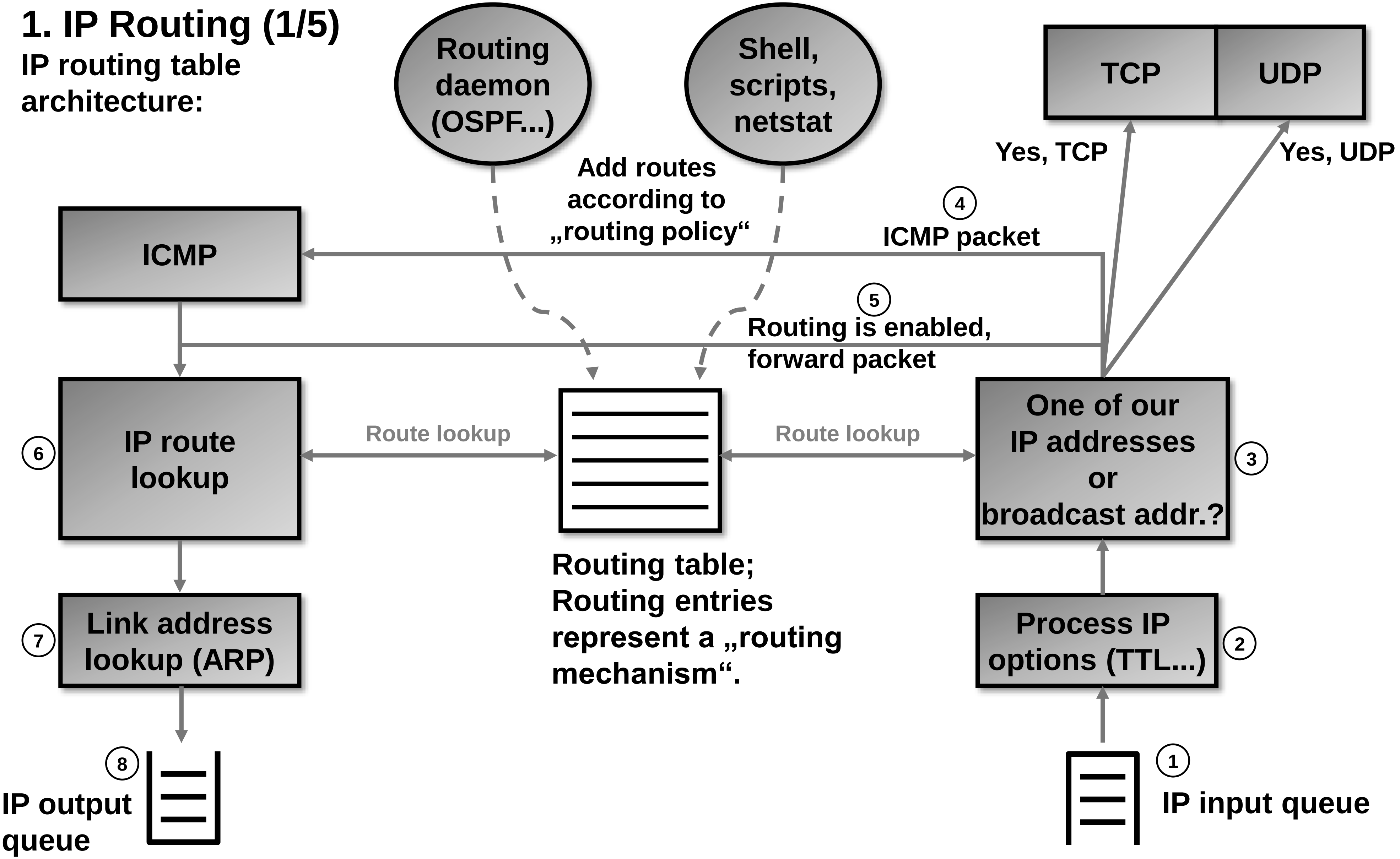
Peter R. Egli
peteregli.net

## Contents

1. IP Routing
2. Routing Protocols
3. Fragmentation in the IP Layer
4. Proxy ARP
5. Routing and IP forwarding examples

# 1. IP Routing (1/5)

IP routing table architecture:


Routing daemon (OSPF...)
Shell, scripts, netstat
Add routes according to „routing policy“
TCP
UDP
Yes, TCP
Yes, UDP
4
ICMP packet
ICMP
5
Routing is enabled, forward packet
6
IP route lookup
Route lookup
Route lookup
One of our IP addresses or broadcast addr.?
3
7
Link address lookup (ARP)
Routing table; Routing entries represent a „routing mechanism“.
Process IP options (TTL...)
2
8
IP output queue
1
IP input queue

# 1. IP Routing (2/5)

**Step by step explanation of IP packet forwarding / processing performed by the router software:**

**1. Packet received:**
**A packet is received from the physical network interface (NIC – Network Interface Card) and placed into an input buffer. The input buffer is necessary to store packets until the CPU has time to process the packets.**

**2. Process IP header fields:**
**The standard IP header fields are processed. These include the TTL field, fragmenting fields and optional header fields.**

**3. Input route lookup:**
**The router makes a route lookup to decide if the packet should be forwarded to another network device or if the packet is destined to an application layer of the router device. If the packet is destined to the router itself, the router forwards the packet to either the TCP or UDP layer based on the protocol field in the IP header.**

**4. ICMP packet processing:**
**If the packet is an ICMP (Internet Control Message Protocol) packet, the router forwards it to the ICMP protocol module for further processing.**

**5. Normal packet forwarding:**
**If the packet is neither for the local router application nor it is an ICMP packet, the router forwards it to a further route lookup.**

**6. Route lookup:**
**The route lookup returns the interface and gateway IP address over which the packet should be sent out.**

**7. Link address lookup:**
**Before the packet can be sent out to the physical interface, the router must determine the target link layer address (e.g. Ethernet address). It first queries the ARP (Address Resolution Protocol) cache for the target link address. If the address is not returned by the ARP cache, it performs an ARP query. Once the ARP response is received, the packet is sent out to the physical interface.**
**N.B.: In case of a point-to-point link (e.g. modem line), there is no ARP lookup.**

# 1. IP Routing (3/5)

## Routing table:

**The routing table takes the target IP address as input and outputs the gateway IP address over which a packet is routed (forwarded) as well as the physical interface that connects to the gateway.**

**The routing table of different OSs look very similar. On Unix (SunOS) the route table looks as follows (Unix command `netstat –rn`, Windows `route print`):**

```
Destination      Gateway          Genmask          Flags    Ref      Use      Interface
140.252.13.65    140.252.13.35    255.255.255.255  UGH      0        0        emd0
127.0.0.1        127.0.0.1        255.255.255.255  UH       1        0        lo0
default          140.252.13.33    0.0.0.0          UG       0        0        emd0
140.252.13.32    140.252.13.34    255.255.255.255  U        25043    0        emd0
1.2.3.0          140.252.13.1     255.255.255.0    UG       1        3        emd0
```

***Destination:***

**Destination IP address to be looked up.**

***Gateway:***

**IP address of gateway over which destination is reachable (next hop) or destination host for direct routes.**

***Genmask:***

**During route lookup, the router applies a bit-wise AND operation to the destination IP address with the Genmask bits. The router the compares the resulting value with the destination IP address in the routing table. In case of multiple matches, the router selects the entry where the highest number of bits match (longest prefix match).**

# 1. IP Routing (4/5)

***Flags:***

**U = Route is up.**
**G = Route is to gateway (indirect route). If G is not set the destination is directly connected (direct route). Sending via gateway means that a packet routed with this route will be forwarded to the gateway with the gateways MAC address (and not the destination's IP address).**
**H = Host route (the destination is a complete host address).**
**D = The route was created by ICMP redirect.**
**M = The route was modified by ICMP redirect.**

***Ref (or Refcnt):***

**Contains the current use of the route. Each TCP connection uses and holds on to a specific route (determined at connection setup). If a TCP connection uses a specific route, Refcnt is incremented.**

***Use:***

**Number of packets sent through this route.**

***Interface:***

**Interface over which to send packet for this route.**

***Precedence (not shown in routing table above):***

**Not all route entries are equal. Some have precedence over others. Static routes (statically added by the system admin) have precedence over dynamic route (obtained by routing protocol, e.g. OSPF).**
**Host routes have precedence over network routes (match first against host route entries).**
**The default route has least precedence (route of "last resort"). If no other route matches the default route is taken (if present).**

# 1. IP Routing (5/5)

***Summary route:***

**A summary route encompasses several other more specific routes.**

***Floating static route:***

**Static route that normally is inactive but appears when more desirable routes become inactive (link down).**

***Recursive routing lookups:***

**Routes to networks reachable by the same gateway do not point to this gateway but to intermediate routers. Routing lookups then recursively resolve these routes.**
**Recursive routing lookups can be used to reduce the routing table, but slow down the route lookup and thus have a performance penalty.**

***Routing Mechanism vs. Routing Policy:***

**The routing mechanism is the algorithm for finding a destination where to forward a packet to (route lookup in routing table).**
**The routing policy defines the mechanism to add routes to the routing table (which metric, which flags, which precedence etc.). The different sources of routes (routing protocols like OSPF, static routes, ICMP redirect etc.) have different precedence (different 'priority').**

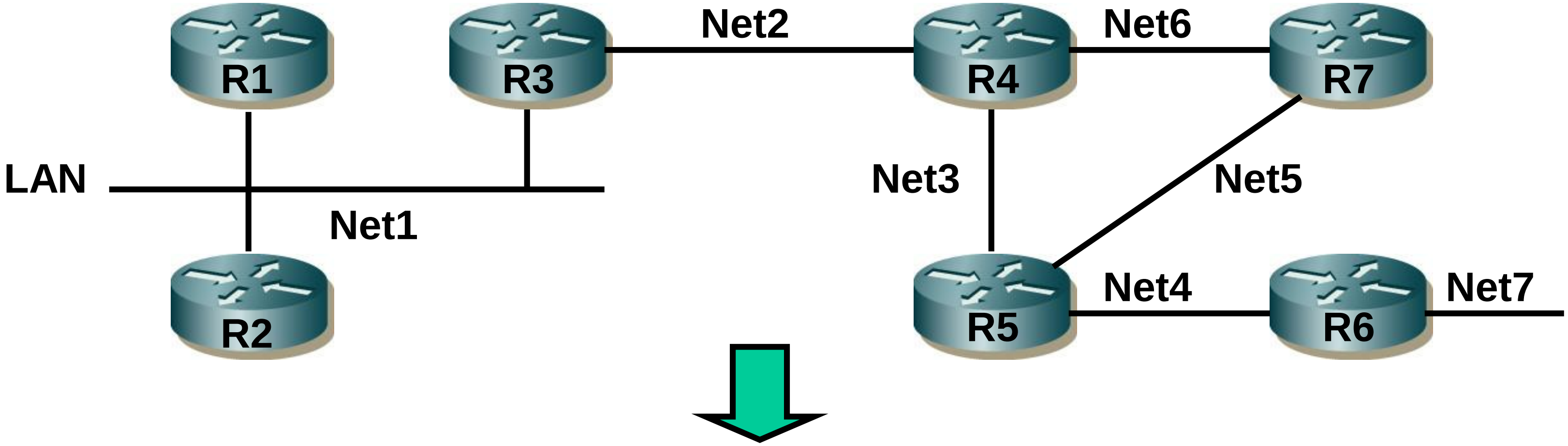
# 2. Routing protocols (1/11)

**Network modeling with graphs (1/3):**

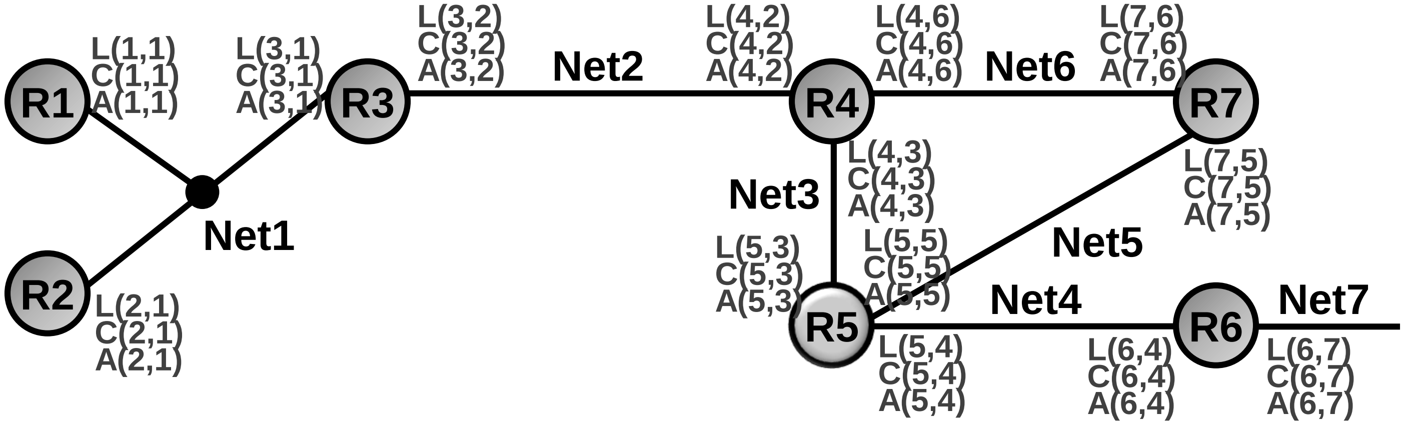
**Routing protocols model networks as graphs with nodes (routers), arcs (links, networks) and costs.**

**Each link of each router is assigned a triple [L,A,C] (Link, Address, Cost).**

**Physical arrangement:**



**Logical view (network graph):**



# 2. Routing protocols (2/11)

## Network modeling with graphs (2/3):

***NetX:***

**Nodes are connected by networks (point to point links or multi-access links (LAN)).**
**NetX is the network number (route prefix like 15.1.0.0/16).**

***Cost (C):***

**Each network interface has a cost, e.g. c(x,y) is cost of router x to transmit on interface to network y. The cost is usually a parameter assigned by the administrator to a link (static configuration) or generally simply 1 (e.g. RIP).**

***Links (L):***

**L(x,y) is the link number (interface identifier) of the link of router x to network y.**
**In the example above the link numbers are globally unique, but in real-world networks each router has its own link numbering.**

***Address (A):***

**A(x,y) is the IP address of node (router) x on network y.**
**On multi-access networks this is a real address (IP) but on unnumbered point-to-point links, A(x,y) may be an arbitrary address (e.g. borrowed from another interface or the loopback interface IP). Additionally for the sake of simplicity the address is often omitted from network graph examples and replaced by the node (router) which sends a route update.**

## 2. Routing protocols (3/11)

### Network modeling with graphs (3/3):

**Example routing table entry for R4 to Net5:**

**Routing table entries can be expressed differently:**

| NetID | Via / next hop |
|---|---|
| Net5 | L(4,3) |

**This routing table entry can also be expressed like:**

| NetID | Via / next hop |
|---|---|
| Net5 | A(5,3) |

**Yet another way to express the routing table entry:**

| NetID | Via / next hop |
|---|---|
| Net5 | R5 |

**The last entry implicitly means that Net5 is reachable through R5. Since R5 has multiple interfaces, it is implicitly assumed that the next hop IP address for R4 to reach Net5 is the IP address of R5 on Net3.**

## 2. Routing protocols (4/11)

### Static routing:

**On point to point links, interface routes can be used (route destination = link number).**

**On multi-access links (Ethernet) the routing table contains a next-hop IP address instead of a link number. Static routing tables for the above network are as follows:**

| R1 | |
|---|---|
| NetID | Via |
| Net1 | direct |
| Net2 | L(1,1) |
| Net3 | L(1,1) |
| Net4 | L(1,1) |
| Net5 | L(1,1) |
| Net6 | L(1,1) |
| Net7 | L(1,1) |

| R2 | |
|---|---|
| NetID | Via |
| Net1 | direct |
| Net2 | L(1,1) |
| Net3 | L(1,1) |
| Net4 | L(1,1) |
| Net5 | L(1,1) |
| Net6 | L(1,1) |
| Net7 | L(1,1) |

| R3 | |
|---|---|
| NetID | Via |
| Net1 | direct |
| Net2 | direct |
| Net3 | L(3,2) |
| Net4 | L(3,2) |
| Net5 | L(3,2) |
| Net6 | L(3,2) |
| Net7 | L(3,2) |

| R4 | |
|---|---|
| NetID | Via |
| Net1 | L(4,2) |
| Net2 | direct |
| Net3 | direct |
| Net4 | L(4,3) |
| Net5 | L(4,6) |
| Net6 | direct |
| Net7 | L(4,3) |

| R5 | |
|---|---|
| NetID | Via |
| Net1 | L(5,3) |
| Net2 | L(5,3) |
| Net3 | direct |
| Net4 | direct |
| Net5 | direct |
| Net6 | L(5,3) |
| Net7 | L(5,4) |

| R6 | |
|---|---|
| NetID | Via |
| Net1 | L(6,4) |
| Net2 | L(6,4) |
| Net3 | L(6,4) |
| Net4 | direct |
| Net5 | L(6,4) |
| Net6 | L(6,4) |
| Net7 | direct |

| R7 | |
|---|---|
| NetID | Via |
| Net1 | L(7,6) |
| Net2 | L(7,6) |
| Net3 | L(7,6) |
| Net4 | L(7,5) |
| Net5 | direct |
| Net6 | direct |
| Net7 | L(7,5) |

# 2. Routing protocols (5/11)

**Classification of routing protocols:**

***1. Interior Gateway Protocols (IGP):***

**Interior Gateway Protocols are used within ASs (Autonomous System). IGPs simply carry routing information without routing policies.**

**Examples: OSPF, RIP, IS-IS**

***2. Exterior Gateway Protocols (EGP):***

**In addition to carrying routing information, EGPs allow to propagate routes with policies. E.g. a provider with AS1 could prohibit routing traffic from another provider with AS2 (competitor) through his network.**

**Example: BGP4**

**Types of routing protocols:**

**Routing protocols can be also be classified according to the routing algorithm they use.**

***1. Distance vector:***

**Examples: RIP, EIGRP (Cisco)**

***2. Link state:***

**Examples: OSPF, IS-IS**

***3. Path vector:***

**Example: BGP4**

# 2. Routing protocols (6/11)

## Distance vector protocols (1/4):

**Each node knows the distance (=cost) to its directly connected neighbors.**

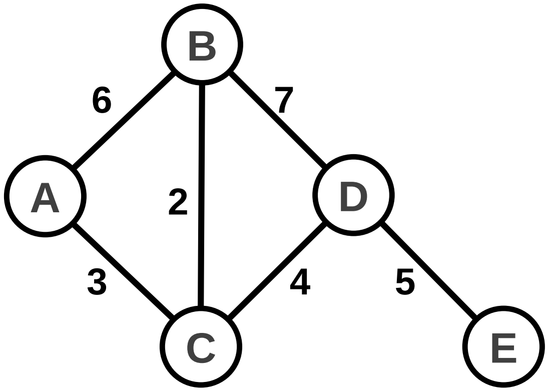
- **A node sends periodically a list of routing updates to its neighbors.**
- **If all nodes update their distances, the routing tables eventually converge.**
- **New nodes advertise themselves to their neighbors.**
- **Finding the shortest route to a destination is a distributed task, unlike in link state algorithms where each node individually computes shortest paths.**
- **Distance vector routing is also called “Routing by rumor” since the routers use second hand information (indirect information) from other routers to build their routing tables. In Link State routing every router uses (flooded) first hand information about link states.**

# 2. Routing protocols (7/11)

## Distance vector protocols (2/4):

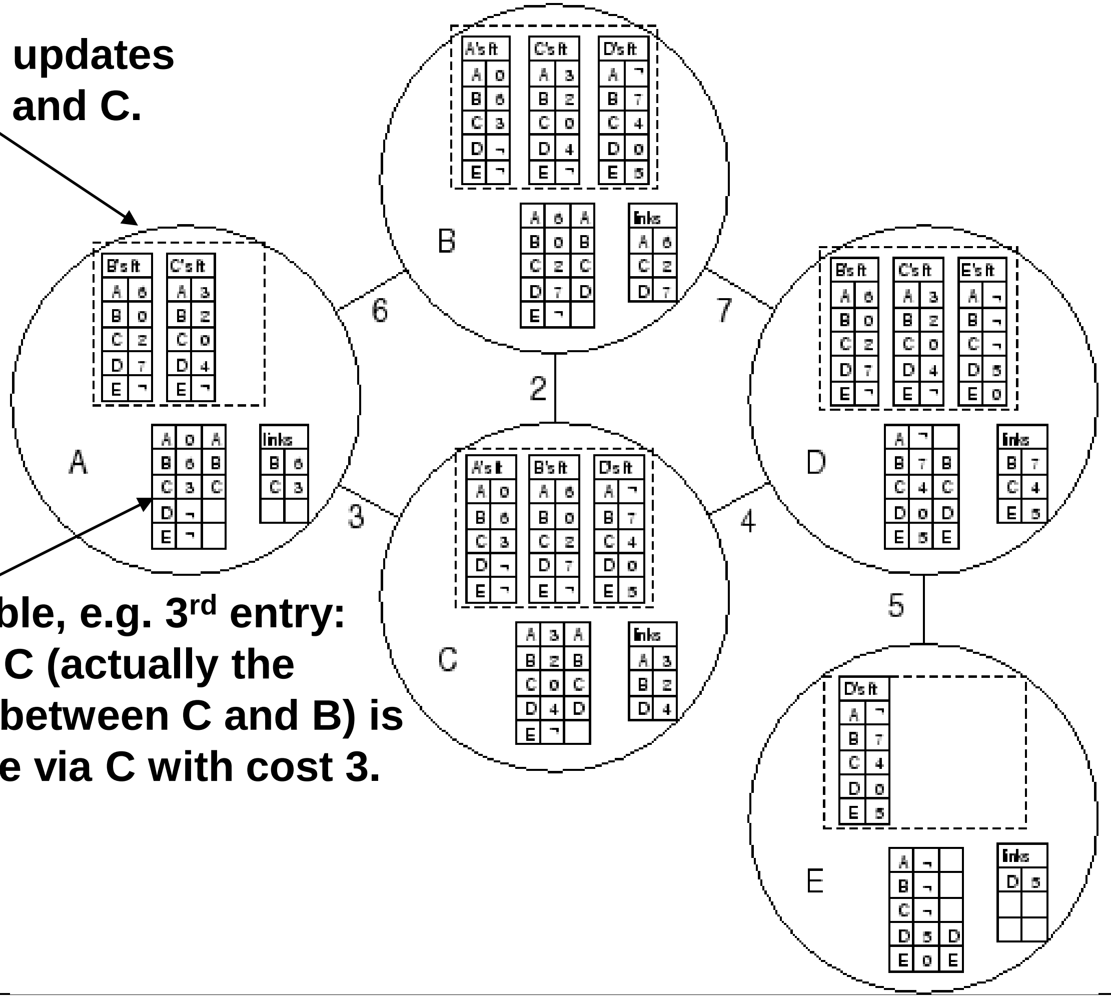
**The network is modeled with nodes (routers) and links (networks) with costs.**

***Network graph with nodes A…E and costs on links:***



***Network with route table updates and route tables:***

**First route table updates received from B and C.**



| B's rt | | C's rt | |
|---|---|---|---|
| A | 6 | A | 3 |
| B | 0 | B | 2 |
| C | 2 | C | 0 |
| D | 7 | D | 4 |
| E | ¬ | E | ¬ |

| | | |
|---|---|---|
| A | 0 | A |
| B | 6 | B |
| C | 3 | C |
| D | ¬ | |
| E | ¬ | |

| links | |
|---|---|
| B | 6 |
| C | 3 |
| | |

| A's rt | | C's rt | | D's rt | |
|---|---|---|---|---|---|
| A | 0 | A | 3 | A | ¬ |
| B | 6 | B | 2 | B | 7 |
| C | 3 | C | 0 | C | 4 |
| D | ¬ | D | 4 | D | 0 |
| E | ¬ | E | ¬ | E | 5 |

| | | |
|---|---|---|
| A | 6 | A |
| B | 0 | B |
| C | 2 | C |
| D | 7 | D |
| E | ¬ | |

| links | |
|---|---|
| A | 6 |
| C | 2 |
| D | 7 |

| A's rt | | B's rt | | D's rt | |
|---|---|---|---|---|---|
| A | 0 | A | 6 | A | ¬ |
| B | 6 | B | 0 | B | 7 |
| C | 3 | C | 2 | C | 4 |
| D | ¬ | D | 7 | D | 0 |
| E | ¬ | E | ¬ | E | 5 |

| | | |
|---|---|---|
| A | 3 | A |
| B | 2 | B |
| C | 0 | C |
| D | 4 | D |
| E | ¬ | |

| links | |
|---|---|
| A | 3 |
| B | 2 |
| D | 4 |

| B's rt | | C's rt | | E's rt | |
|---|---|---|---|---|---|
| A | 6 | A | 3 | A | ¬ |
| B | 0 | B | 2 | B | ¬ |
| C | 2 | C | 0 | C | ¬ |
| D | 7 | D | 4 | D | 5 |
| E | ¬ | E | ¬ | E | 0 |

| | | |
|---|---|---|
| A | ¬ | |
| B | 7 | B |
| C | 4 | C |
| D | 0 | D |
| E | 5 | E |

| links | |
|---|---|
| B | 7 |
| C | 4 |
| E | 5 |

| D's rt | |
|---|---|
| A | ¬ |
| B | 7 |
| C | 4 |
| D | 0 |
| E | 5 |

| | | |
|---|---|---|
| A | ¬ | |
| B | ¬ | |
| C | ¬ | |
| D | 5 | D |
| E | 0 | E |

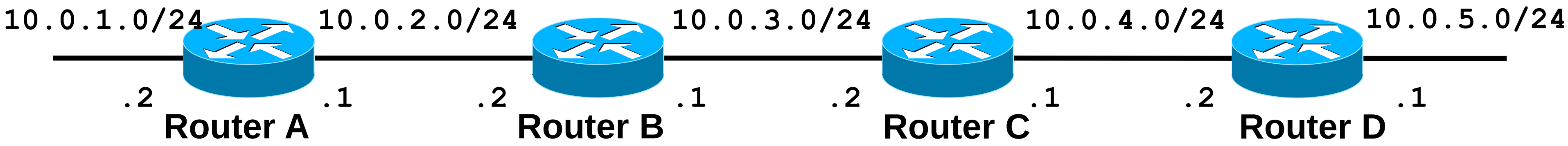
| links | |
|---|---|
| D | 5 |
| | |
| | |

**Route table, e.g. 3rd entry: Network C (actually the network between C and B) is reachable via C with cost 3.**

# 2. Routing protocols (8/11)

## Distance vector protocols (3/4):

**A more realistic example with IP (RIP – Routing Information Protocol). All costs = 1 (C(x,y)=1). All updates occur simultaneously. Initially each router knows the cost of the connected lines.**



| Net | via | cost |
|---|---|---|
| **t=0:** | | |
| 10.0.1.0 | – | 0 |
| 10.0.2.0 | – | 0 |
| **t=1:** | | |
| 10.0.1.0 | – | 0 |
| 10.0.2.0 | – | 0 |
| 10.0.3.0 | 10.0.2.2 | 1 |
| **t=2:** | | |
| 10.0.1.0 | – | 0 |
| 10.0.2.0 | – | 0 |
| 10.0.3.0 | 10.0.2.2 | 1 |
| 10.0.4.0 | 10.0.2.2 | 2 |

| Net | via | cost |
|---|---|---|
| **t=0:** | | |
| 10.0.2.0 | – | 0 |
| 10.0.3.0 | – | 0 |
| **t=1:** | | |
| 10.0.1.0 | 10.0.2.1 | 1 |
| 10.0.2.0 | – | 0 |
| 10.0.3.0 | – | 0 |
| 10.0.4.0 | 10.0.3.2 | 1 |
| **t=2:** | | |
| 10.0.1.0 | 10.0.2.1 | 1 |
| 10.0.2.0 | – | 0 |
| 10.0.3.0 | – | 0 |
| 10.0.4.0 | 10.0.3.2 | 1 |
| 10.0.5.0 | 10.0.3.2 | 2 |

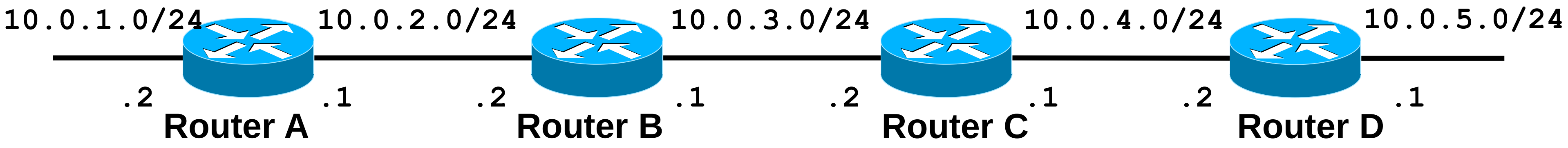
| Net | via | cost |
|---|---|---|
| **t=0:** | | |
| 10.0.3.0 | – | 0 |
| 10.0.4.0 | – | 0 |
| **t=1:** | | |
| 10.0.2.0 | 10.0.3.1 | 1 |
| 10.0.3.0 | – | 0 |
| 10.0.4.0 | – | 0 |
| 10.0.5.0 | 10.0.4.2 | 1 |
| **t=2:** | | |
| 10.0.1.0 | 10.0.3.1 | 2 |
| 10.0.2.0 | 10.0.3.1 | 1 |
| 10.0.3.0 | – | 0 |
| 10.0.4.0 | – | 0 |
| 10.0.5.0 | 10.0.4.2 | 1 |

| Net | via | cost |
|---|---|---|
| **t=0:** | | |
| 10.0.4.0 | – | 0 |
| 10.0.5.0 | – | 0 |
| **t=1:** | | |
| 10.0.3.0 | 10.0.4.1 | 1 |
| 10.0.4.0 | – | 0 |
| 10.0.5.0 | – | 0 |
| **t=2:** | | |
| 10.0.2.0 | 10.0.4.1 | 2 |
| 10.0.3.0 | 10.0.4.1 | 1 |
| 10.0.4.0 | – | 0 |
| 10.0.5.0 | – | 0 |

# 2. Routing protocols (9/11)

## Distance vector protocols (4/4):

**After t=3 the routing tables have converged to a stable state.**



Router A:

| Net | via | cost |
|---|---|---|
| *t=2:* | | |
| *10.0.1.0* | *-* | *0* |
| *10.0.2.0* | *-* | *0* |
| *10.0.3.0* | *10.0.2.2* | *1* |
| *10.0.4.0* | *10.0.2.2* | *2* |
| **t=3:** | | |
| 10.0.1.0 | - | 0 |
| 10.0.2.0 | - | 0 |
| 10.0.3.0 | 10.0.2.2 | 1 |
| 10.0.4.0 | 10.0.2.2 | 2 |
| 10.0.5.0 | 10.0.2.2 | 3 |

Router B:

| Net | via | cost |
|---|---|---|
| *t=2:* | | |
| *10.0.1.0* | *10.0.2.1* | *1* |
| *10.0.2.0* | *-* | *0* |
| *10.0.3.0* | *-* | *0* |
| *10.0.4.0* | *10.0.3.2* | *1* |
| *10.0.5.0* | *10.0.3.2* | *2* |
| **t=3:** | | |
| 10.0.1.0 | 10.0.2.1 | 1 |
| 10.0.2.0 | - | 0 |
| 10.0.3.0 | - | 0 |
| 10.0.4.0 | 10.0.3.2 | 1 |
| 10.0.5.0 | 10.0.3.2 | 2 |

Router C:

| Net | via | cost |
|---|---|---|
| *t=2:* | | |
| *10.0.1.0* | *10.0.3.1* | *2* |
| *10.0.2.0* | *10.0.3.1* | *1* |
| *10.0.3.0* | *-* | *0* |
| *10.0.4.0* | *-* | *0* |
| *10.0.5.0* | *10.0.4.2* | *1* |
| **t=3:** | | |
| 10.0.1.0 | 10.0.3.1 | 2 |
| 10.0.2.0 | 10.0.3.1 | 1 |
| 10.0.3.0 | - | 0 |
| 10.0.4.0 | - | 0 |
| 10.0.5.0 | 10.0.4.2 | 1 |

Router D:

| Net | via | cost |
|---|---|---|
| *t=2:* | | |
| *10.0.2.0* | *10.0.4.1* | *2* |
| *10.0.3.0* | *10.0.4.1* | *1* |
| *10.0.4.0* | *-* | *0* |
| *10.0.5.0* | *-* | *0* |
| **t=3:** | | |
| 10.0.1.0 | 10.0.4.1 | 3 |
| 10.0.2.0 | 10.0.4.1 | 2 |
| 10.0.3.0 | 10.0.4.1 | 1 |
| 10.0.4.0 | - | 0 |
| 10.0.5.0 | - | 0 |

# 2. Routing protocols (10/11)

## Link state routing protocols:

**Unlike distance vector protocols, the routers in link state protocols have a complete view of the network through flooded link state packets (every router sends update packets to every other router on the network through multicast or broadcast). Each router independently builds its routing table based on the same information.**

***Simplified route update procedure in the OSPF link state protocol:***

**1. Discover its neighbours and learn their network addresses (OSPF HELLO packet).**
**2. Measure the delay or cost to each of its neighbours (round trip time measurement with OSPF ECHO packet).**
**3. Construct a packet telling everything that the router has learned in step 1 & 2.**
**4. Send this packet to all other routers (flooding).**
**5. Compute the shortest path to every other router, e.g. with Dijkstra algorithm.**
**6. Build-up routing table with entries.**

**→ Link state protocols are more complex that distance vector protocols, but are also superior in performance. In case of a topology change, the routing tables in a link state routing network converge (change and become stable) much faster than distance vector protocols do.**

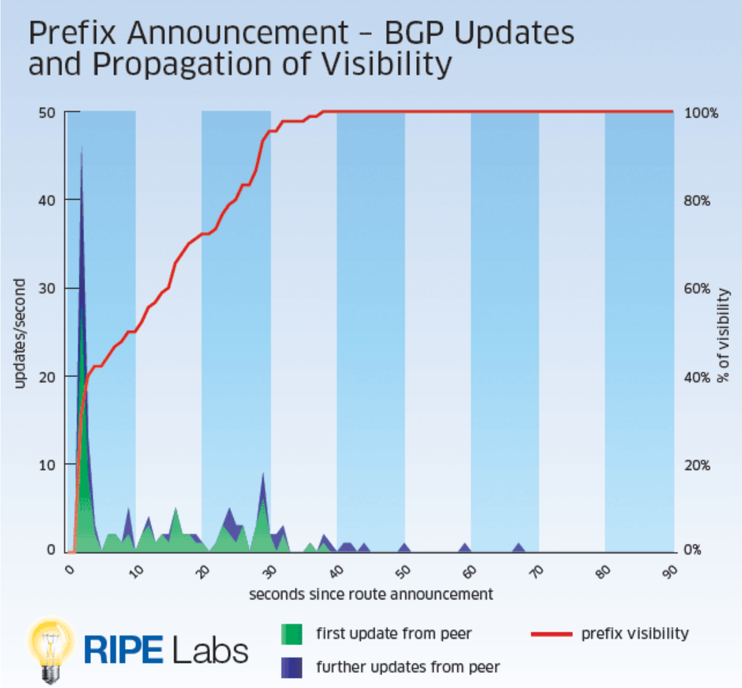
**Examples of link state protocols: OSPF (Open Shortest Path First), IS-IS (Intermediate System to Intermediate System protocol).**

# 2. Routing protocols (11/11)

## BGP – Border Gateway Protocol:

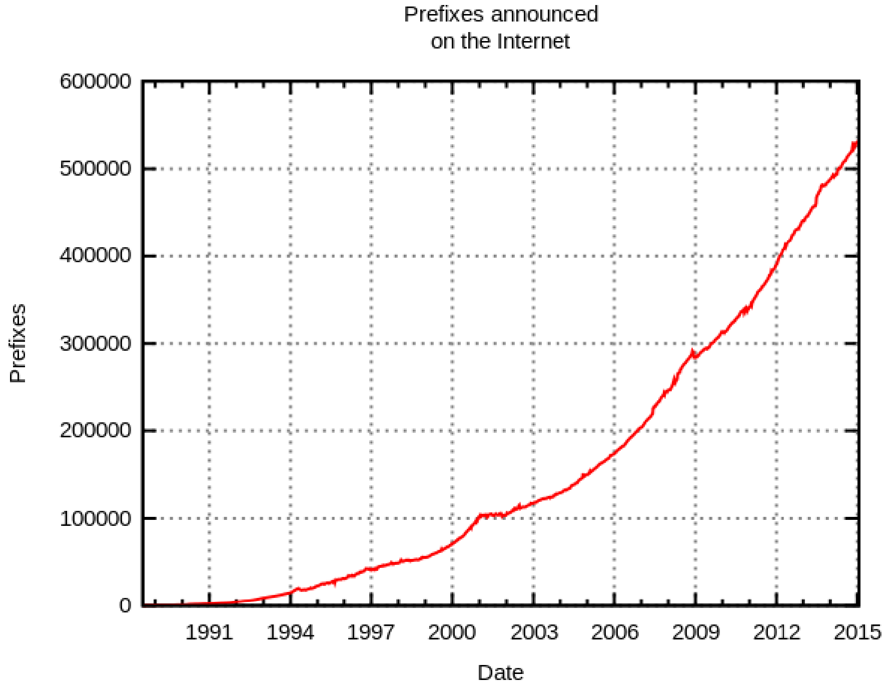
**BGP4 (RFC4271) is the Internet backbone routing protocol. BGP is an exterior gateway protocol (EGP) for connecting different ISPs / carriers. BGP uses a path-vector algorithm where every router constructs paths of networks (AS – Autonomous System) and propagates these paths to other routers.**

### BGP route propatation:



Source: http://www.circleid.com/posts/how_a_routing_prefix_travels_through_the_internet/

### Internet route count (BGP prefixes):



Source: http://en.wikipedia.org/wiki/Border_Gateway_Protocol

# 3. Fragmentation in the IP Layer (1/2)

**IPv4 mandates (RFC1812) that an IP router be able to fragment packets that are too large for an interface’s MTU (Maximum Transfer Unit).**

**The following IPv4 header fields are involved in IP packet fragmentation:**

<table>
<tr><td>Ver.</td><td>IHL</td><td>TOS</td><td colspan="2">Total length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td>Fragment offset</td></tr>
<tr><td colspan="2">TTL</td><td>Protocol</td><td colspan="2">Header checksum</td></tr>
<tr><td colspan="5">IP source address</td></tr>
<tr><td colspan="5">IP destination address</td></tr>
<tr><td colspan="5">Optional IP options</td></tr>
</table>

**Total length:**
Length of IP packet (IP header plus data) in bytes.

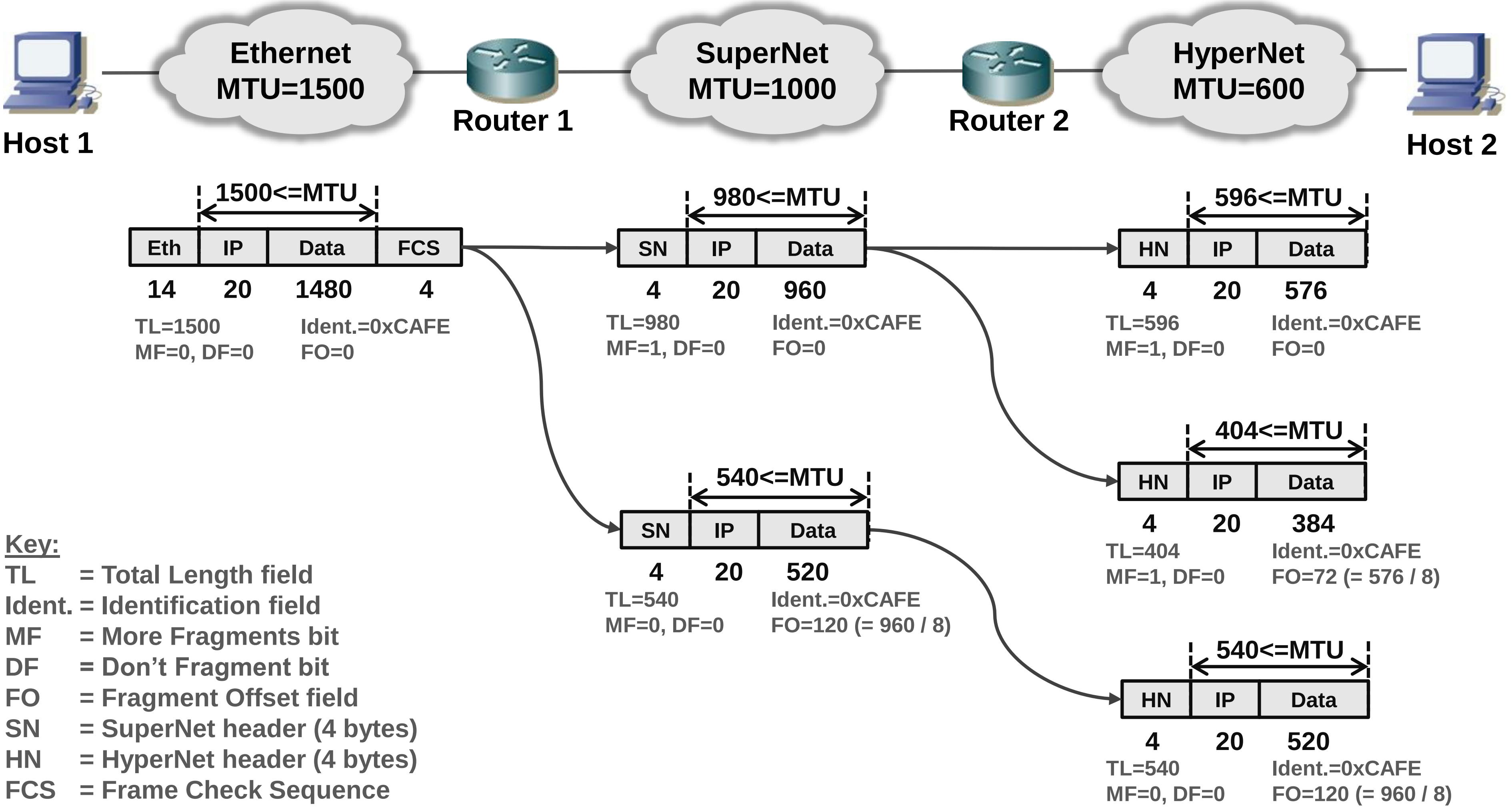
**Identification:**
Used for identifying the fragments of a fragmented packet.

**Flags:**
DF (Don’t Fragment) and MF (More Fragments) bits.
DF: If set to 1, routers along the path must not fragment the packet. If fragmentation is needed (packet exceeds MTU), the router drops the packet and sends an ICMP-Dest-Unreachable back to the sender.
MF: Set to 1 for all fragments of a packet except the last fragment.

**Fragment offset:**
Offset of data (in 8 byte units) of this fragment within the original IP packet.

# 3. Fragmentation in the IP Layer (2/2)

**The process of IP fragmentation is shown with the following setup containing an Ethernet segment and the 2 fictitious SuperNet and HyperNet segments:**



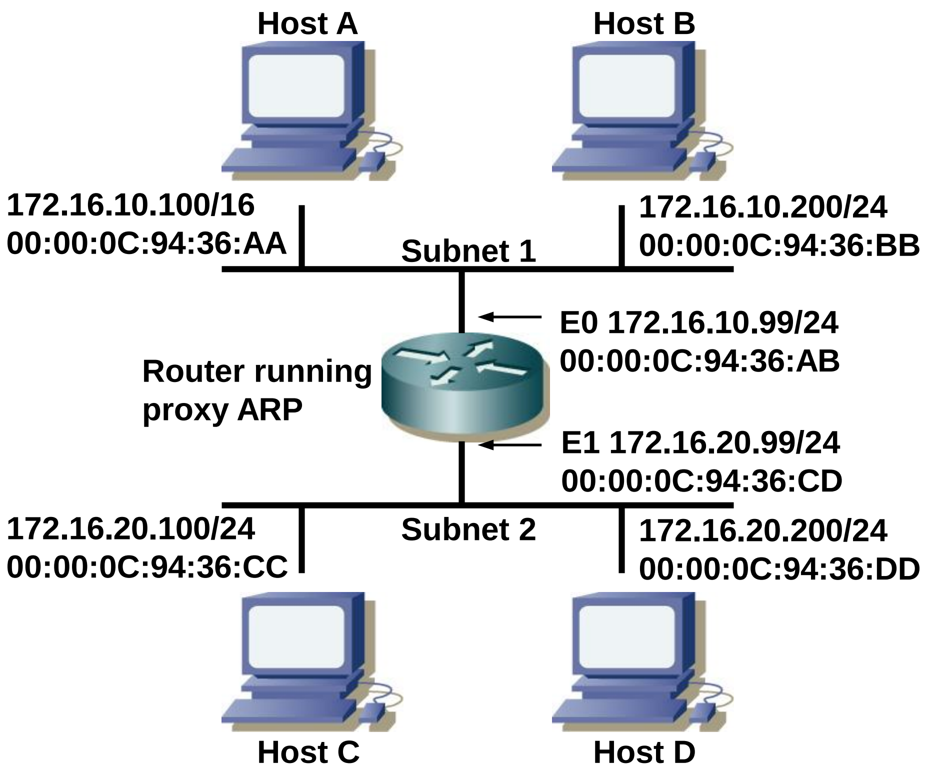
Key:

| | |
|---|---|
| TL | = Total Length field |
| Ident. | = Identification field |
| MF | = More Fragments bit |
| DF | = Don’t Fragment bit |
| FO | = Fragment Offset field |
| SN | = SuperNet header (4 bytes) |
| HN | = HyperNet header (4 bytes) |
| FCS | = Frame Check Sequence |

# 4. Proxy ARP RFC1027 (1/4)

**Proxy ARP can be used to connect hosts that are unaware of subnets through a router that runs proxy ARP.**

**Example network scenario using proxy ARP:**



**Host A on subnet 1 wants to send a packet to Host D on subnet 2.**
**Host A has a 16 bit subnet mask while the router and all other hosts have 24 bit subnet masks.**

# 4. Proxy ARP RFC1027 (2/4)

**Step by step explanation:**

**1. Host A thinks that host D is on the same subnet because of the 16 bit subnet mask (host D address 172.16.20.200 is masked to 172.16.0.0 which is the network address of host A's interface).**

**2. Host A needs the MAC address of host D to send the packet to.**
**Host A therefore sends an ARP request packet with the following content:**

| Protocol | Content | | Comment |
|---|---|---|---|
| Ethernet | Ethernet source address | 00:00:0C:94:36:AA | Host A's MAC address |
| | Ethernet destination address | FF:FF:FF:FF:FF:FF | Ethernet broadcast address |
| ARP | Sender's MAC address | 00:00:0C:94:36:AA | Host A's MAC address |
| | Sender's IP address | 172.16.10.100 | Host A's IP address |
| | Target MAC address | 00:00:00:00:00:00 | Requested information |
| | Target IP address | 172.16.20.200 | Host D's IP address (target) |

**3. All devices on subnet 1 receive the ARP request including the router's E0 interface.**
**The router does not forward the ARP request to subnet 2 (MAC broadcasts are never routed).**

# 4. Proxy ARP RFC1027 (3/4)

**4. Since the router runs "Proxy ARP" it answers the ARP request with its own MAC address (the router answers the ARP request "on behalf" of host D):**

| Protocol | Content | | Comment |
|---|---|---|---|
| Ethernet | Ethernet source address | 00:00:0C:94:36:AB | The router's MAC address on E0 |
| | Ethernet destination address | 00:00:0C:94:36:AA | Host A's MAC address (ARP replies are unicast) |
| ARP | Sender's MAC address | 00:00:0C:94:36:AB | The router's MAC address |
| | Sender's IP address | 172.16.20.200 | Host D's IP address |
| | Target MAC address | 00:00:0C:94:36:AA | Host A's MAC address |
| | Target IP address | 172.16.10.100 | Host A's IP address |

**5. Host A updates its ARP cache with the entry 172.16.20.200 / 00:00:0C:94:36:AB**

**6. From now on host A forwards all packets destined to host D to the MAC address 00:00:0C:94:36:AB (the router's MAC address). Since the router is directly connected to subnet 2 it forwards (normal routing) the packets to host D.**

## 4. Proxy ARP RFC1027 (4/4)

➔ Packets from host D back to host A are routed normally without proxy ARP.

➔ Host A will forward all packets to hosts on subnet 2 to the router’s MAC address 00:00:0C:94:36:AB. This means that in host A’s ARP cache all entries with IP addresses from subnet 2 are mapped to the router’s MAC address 00:00:0c:94:36:AB.

➔ When host B on subnet 1 wants to send a packet to a host on subnet 2 it does not send an ARP request beforehand since host 2 has a 24 bit subnet mask. Host B will send the packet to the default gateway address which is again the router’s E0 interface.

Proxy ARP advantages / disadvantages:

☺ With proxy ARP a switched Ethernet network can be split into 2 subnets without the need to change IP addresses and routes on hosts and routers.
Only the router between the 2 subnets needs to run proxy ARP.

☹ Hosts need larger ARP tables.

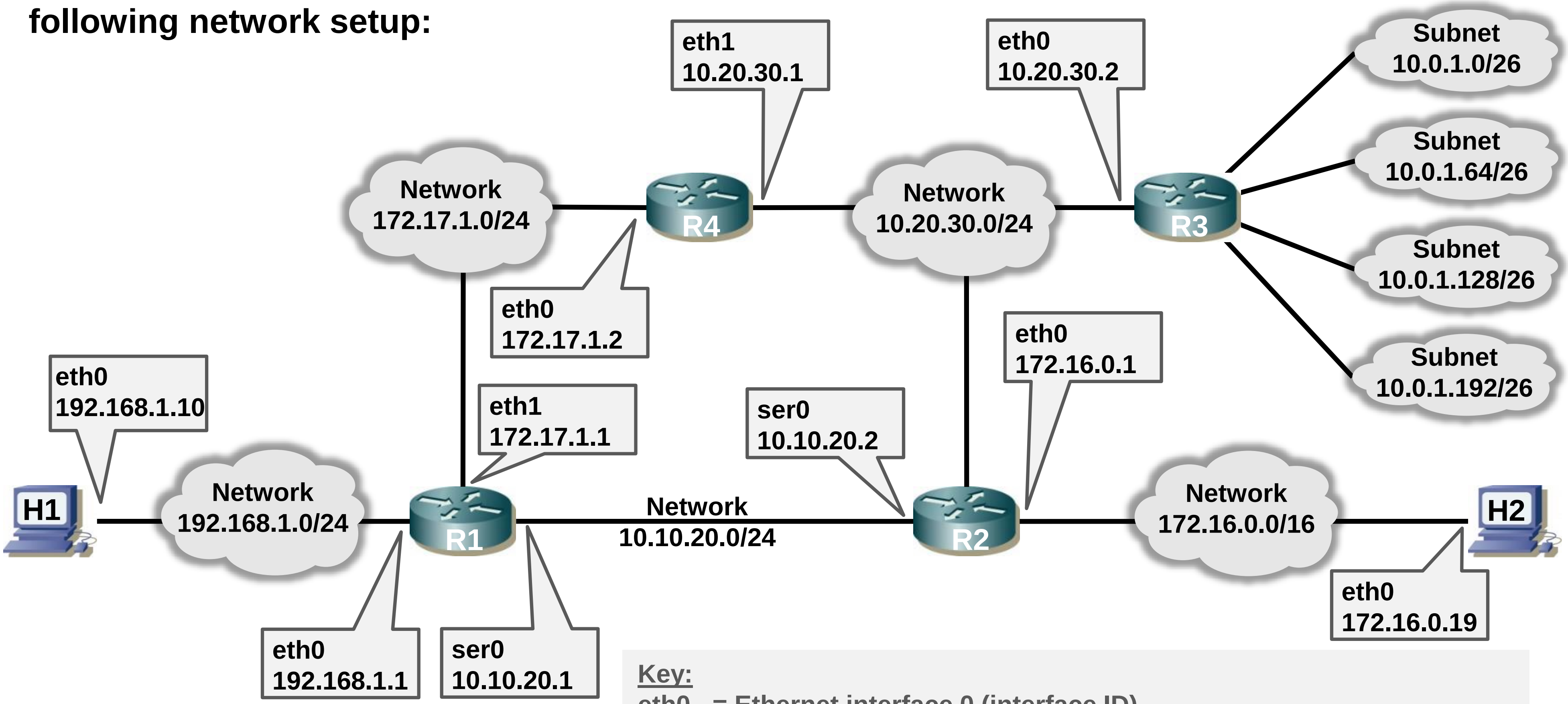
☹ Only works on networks that use ARP.

☹ Does not work when 2 routers connect 2 subnetworks.

☹ Reduced security (spoofing).

# 5. Routing and IP forwarding examples (1/13)

**The following pages exemplify various routing and IP forwarding scenarios based on the following network setup:**

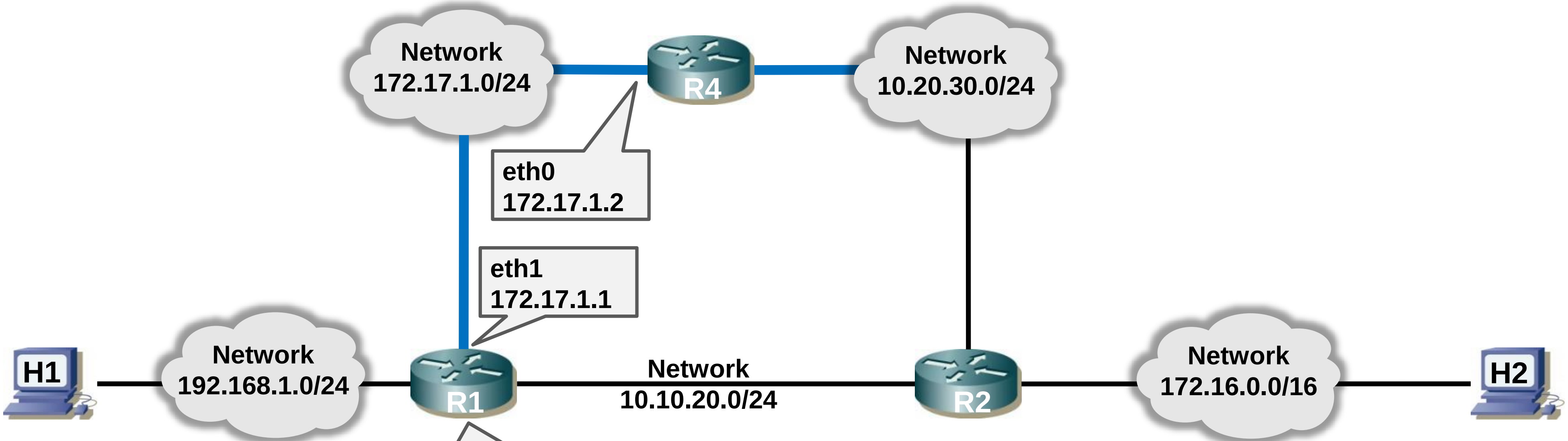


Key:
eth0 = Ethernet interface 0 (interface ID)
ser0 = Serial interface 0 (interface ID)
Callouts show interface ID (e.g. eth0) and IP address on that interface
/24 = Network mask expressed as number of ‘1’ bits from the left side
Hx = Host x
Rx = Router x

# 5. Routing and IP forwarding examples (2/13)

## A. Simple network route:

**In order to reach network 10.20.30.0/24, router R1 needs a route to that network as shown below.**



R1 routing table (excerpt)

| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| x.x.x.x | x.x.x.x | x.x.x.x | xxx |
| 10.20.30.0 | 255.255.255.0 | 172.17.1.2 | eth1 |
| x.x.x.x | x.x.x.x | x.x.x.x | xxx |

Route entry for network 10.20.30.0/24

# 5. Routing and IP forwarding examples (3/13)

## A. Simple network route:

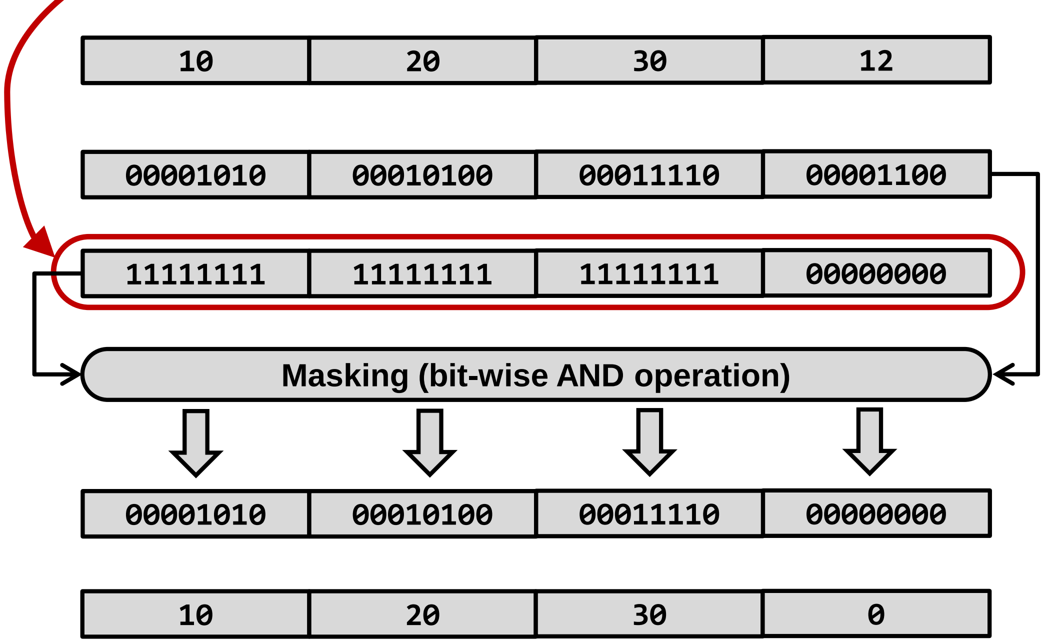
**Packet processing on R1:**

**1. R1 receives a packet with destination IP address = 10.20.30.12 (e.g. from H1).**

**2. R1 iterates through all routing table entries. For each entry it masks the packet's destination IP address with the network mask of the route entry (bit-wise AND operation) as follows:**

R1 routing table (excerpt)

| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| 10.20.30.0 | 255.255.255.0 | 172.17.1.2 | eth1 |



| | | | | |
|---|---|---|---|---|
| IP packet destination IP address in dotted-decimal notation | 10 | 20 | 30 | 12 |
| IP packet destination IP address in binary notation | 00001010 | 00010100 | 00011110 | 00001100 |
| Route entry network mask binary notation | 11111111 | 11111111 | 11111111 | 00000000 |
| Resulting IP address (prefix) in binary notation | 00001010 | 00010100 | 00011110 | 00000000 |
| Resulting IP address in dotted-decimal notation | 10 | 20 | 30 | 0 |

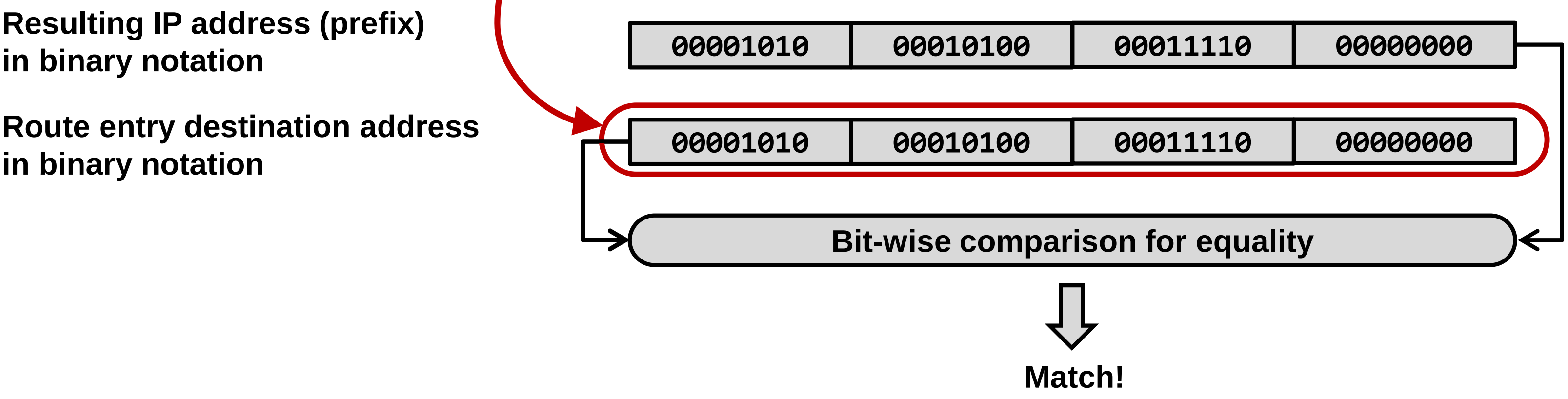
# 5. Routing and IP forwarding examples (4/13)

## A. Simple network route:

**Packet processing on R1:**

**3. R1 compares the resulting IP address (prefix) with the route entry destination IP address. In case of a full match (all 32 bits are identical), the route matches.**

| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| 10.20.30.0 | 255.255.255.0 | 172.17.1.2 | eth1 |


Resulting IP address (prefix) in binary notation
00001010
00010100
00011110
00000000
Route entry destination address in binary notation
00001010
00010100
00011110
00000000
Bit-wise comparison for equality
Match!

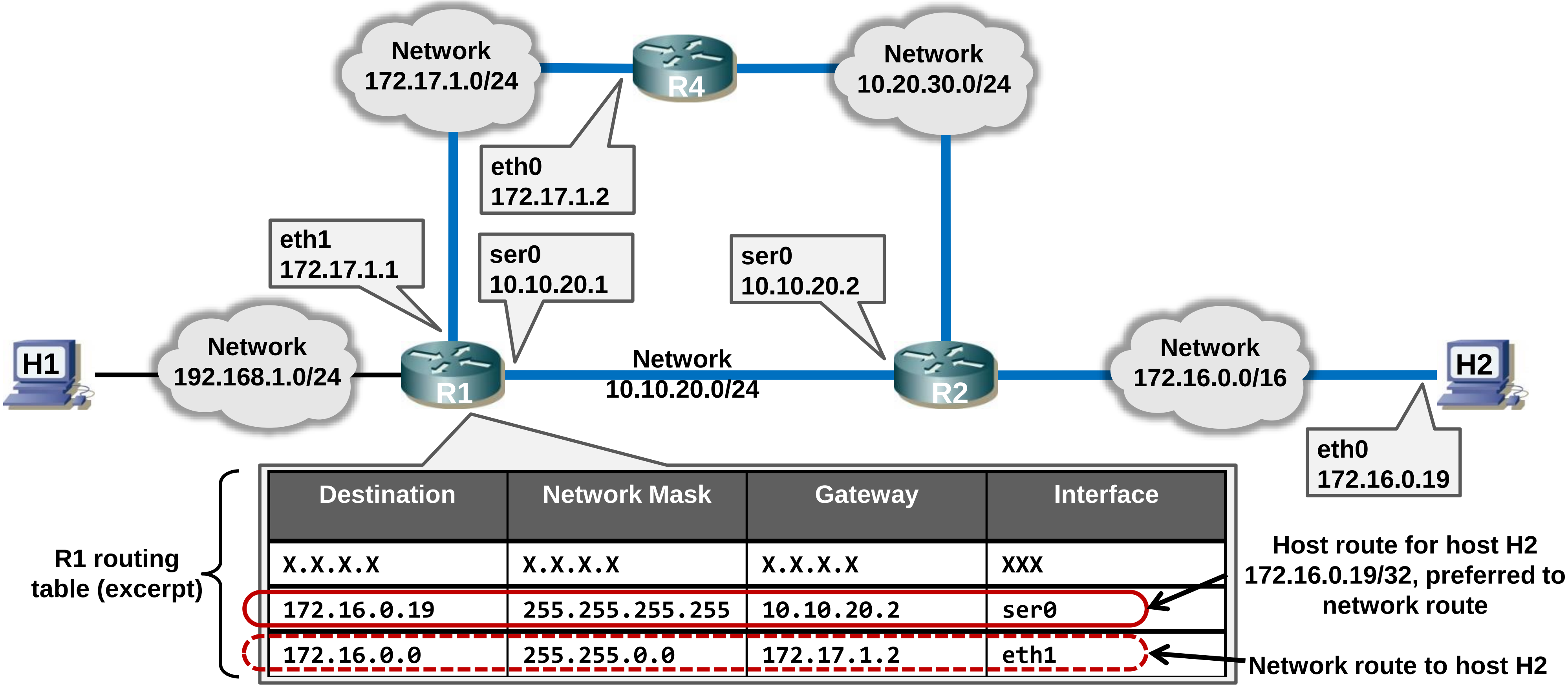
**4. In case of a route entry match, R1 forwards the IP packet over the interface and via the gateway defined by the route entry (172.17.1.2, interface eth1).**

# 5. Routing and IP forwarding examples (5/13)

## B. Host route:

**Host routes have a 32 bit network mask, i.e. 255.255.255.255.**

**Host routes take precedence over corresponding network routes since more bits (actually all 32 bits) of the IP packet's destination IP address match with the host route's destination IP address (longest prefix match rule).**



R1 routing table (excerpt)

| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| x.x.x.x | x.x.x.x | x.x.x.x | xxx |
| 172.16.0.19 | 255.255.255.255 | 10.10.20.2 | ser0 |
| 172.16.0.0 | 255.255.0.0 | 172.17.1.2 | eth1 |

Host route for host H2 172.16.0.19/32, preferred to network route

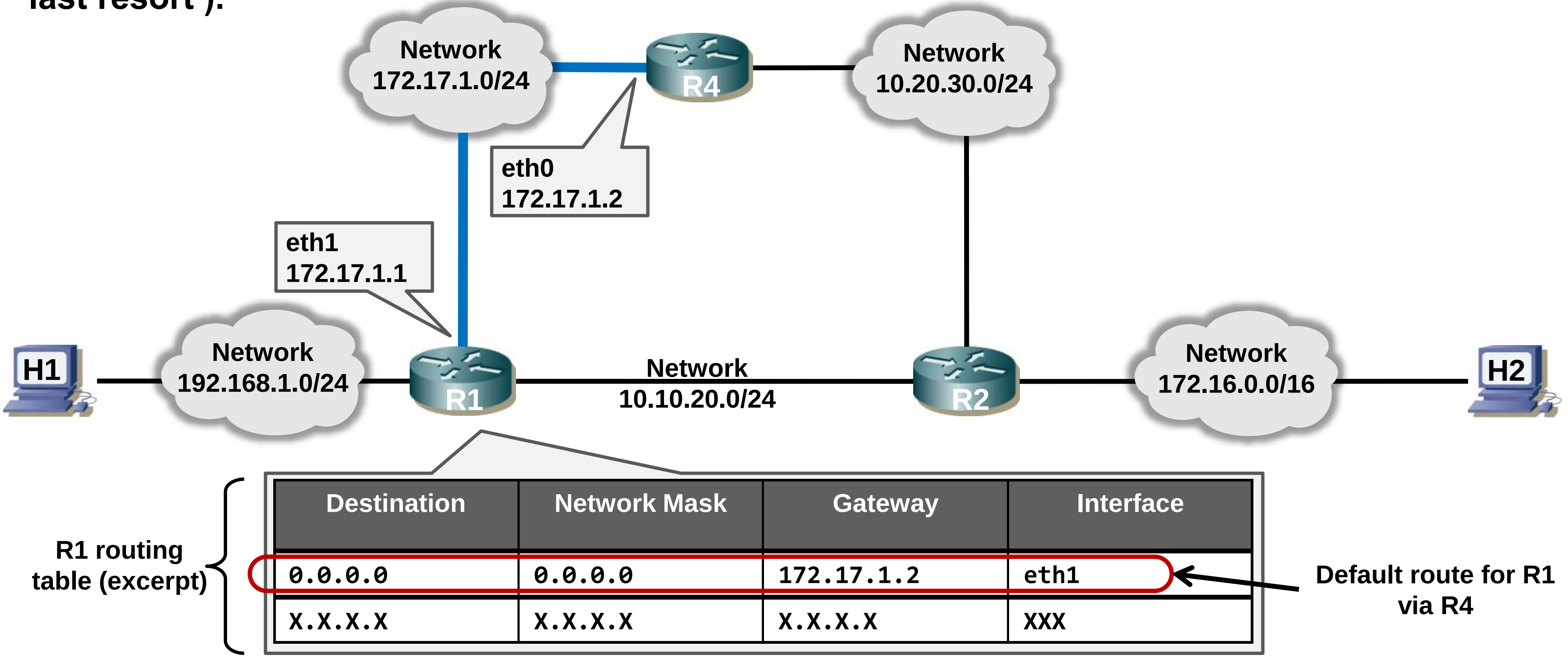
Network route to host H2

# 5. Routing and IP forwarding examples (6/13)

## C. Default route:

**Default routes have a destination IP address and network mask of 0.0.0.0. If present, default routes always match because masking with the network mask of 0.0.0.0 always results in a prefix 0.0.0.0 which matches the default route's destination 0.0.0.0.**

**Default routes have least preference since the match length is 0. Thus default routes are used for forwarding a packet if no other route (host or network) matches (default route = 'route of last resort').**



R1 routing table (excerpt)

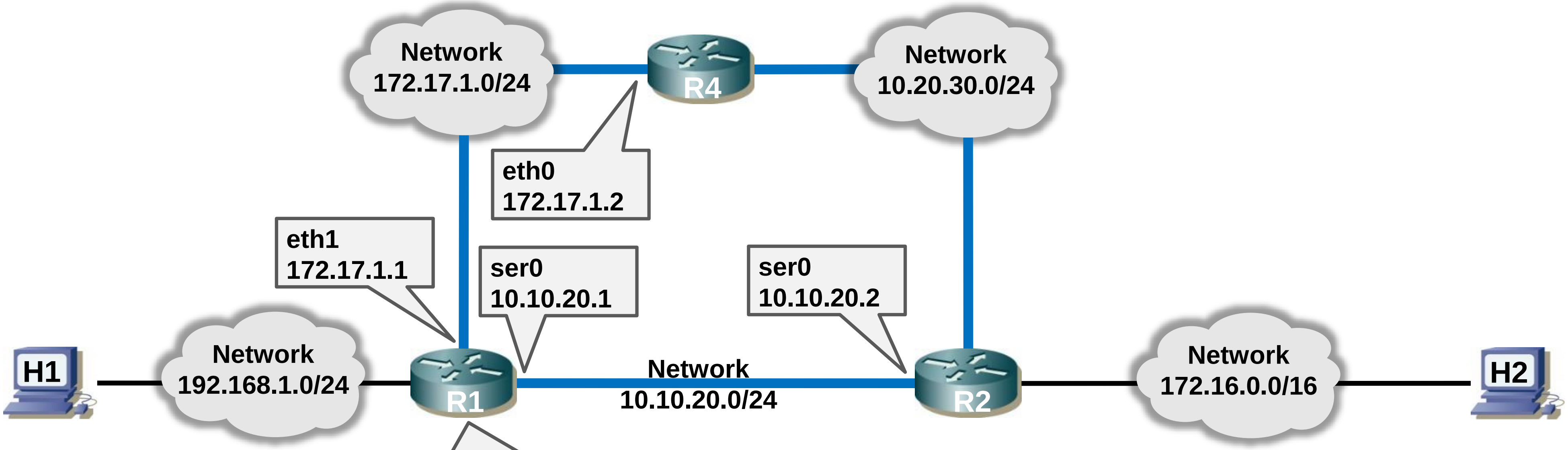
| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| 0.0.0.0 | 0.0.0.0 | 172.17.1.2 | eth1 |
| X.X.X.X | X.X.X.X | X.X.X.X | XXX |

Default route for R1 via R4

# 5. Routing and IP forwarding examples (7/13)

## D. Multiple routes to the same network (or host):

**A routing table may contain multiple routes to the same destination network. In the example below, 2 routes ‘point’ to the network 10.20.0.0/16.**



R1 routing table (excerpt)

| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| x.x.x.x | x.x.x.x | x.x.x.x | xxx |
| 10.20.0.0 | 255.255.0.0 | 10.10.20.2 | ser0 |
| 10.20.30.0 | 255.255.255.0 | 172.17.1.2 | eth1 |

Less specific route entry for network 10.20.0.0/16 via R2

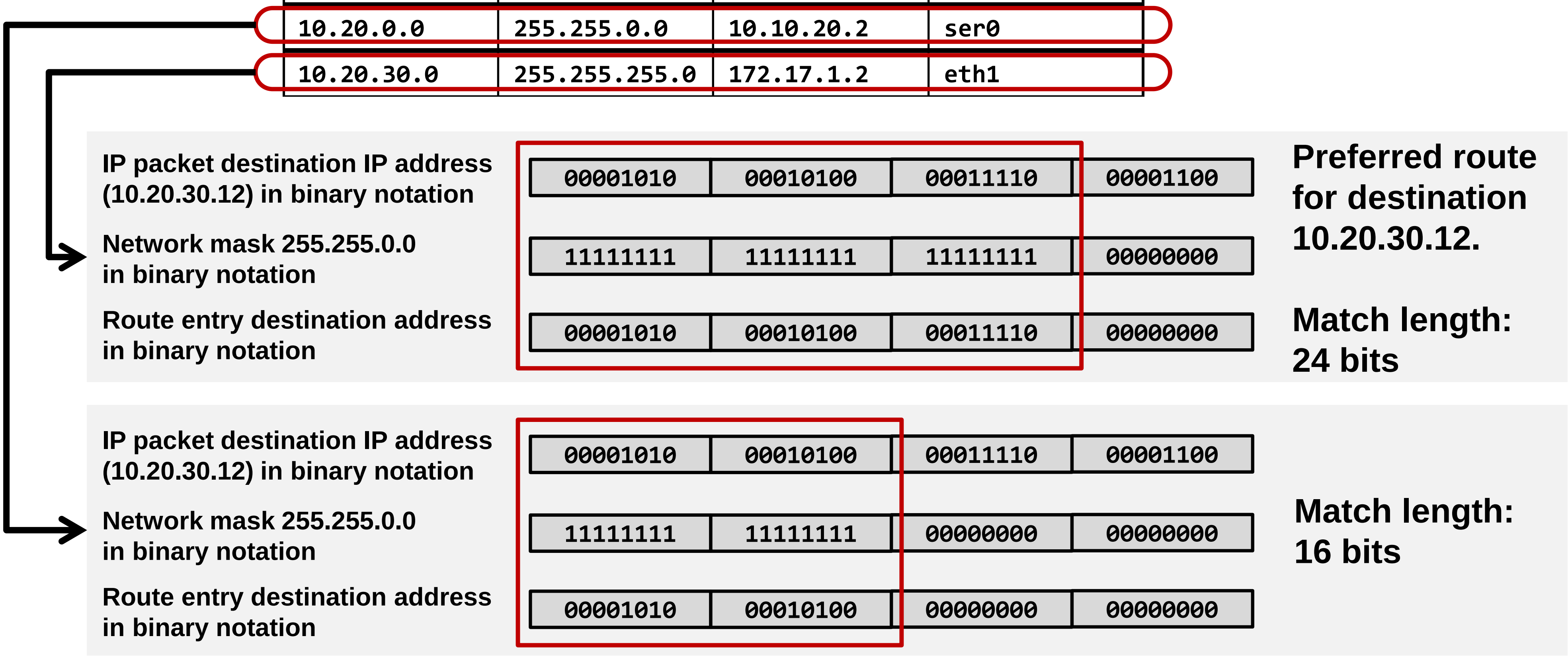
Route entry for network 10.20.30.0/24 via R4

# 5. Routing and IP forwarding examples (8/13)

## D. Multiple routes to the same network (or host):

**If 2 routes match for a given destination IP address, the route with the longer prefix match 'wins'.**

| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| X.X.X.X | X.X.X.X | X.X.X.X | XXX |
| 10.20.0.0 | 255.255.0.0 | 10.10.20.2 | ser0 |
| 10.20.30.0 | 255.255.255.0 | 172.17.1.2 | eth1 |



| | | | | |
|---|---|---|---|---|
| IP packet destination IP address (10.20.30.12) in binary notation | 00001010 | 00010100 | 00011110 | 00001100 |
| Network mask 255.255.0.0 in binary notation | 11111111 | 11111111 | 11111111 | 00000000 |
| Route entry destination address in binary notation | 00001010 | 00010100 | 00011110 | 00000000 |

**Preferred route for destination 10.20.30.12.**

**Match length: 24 bits**

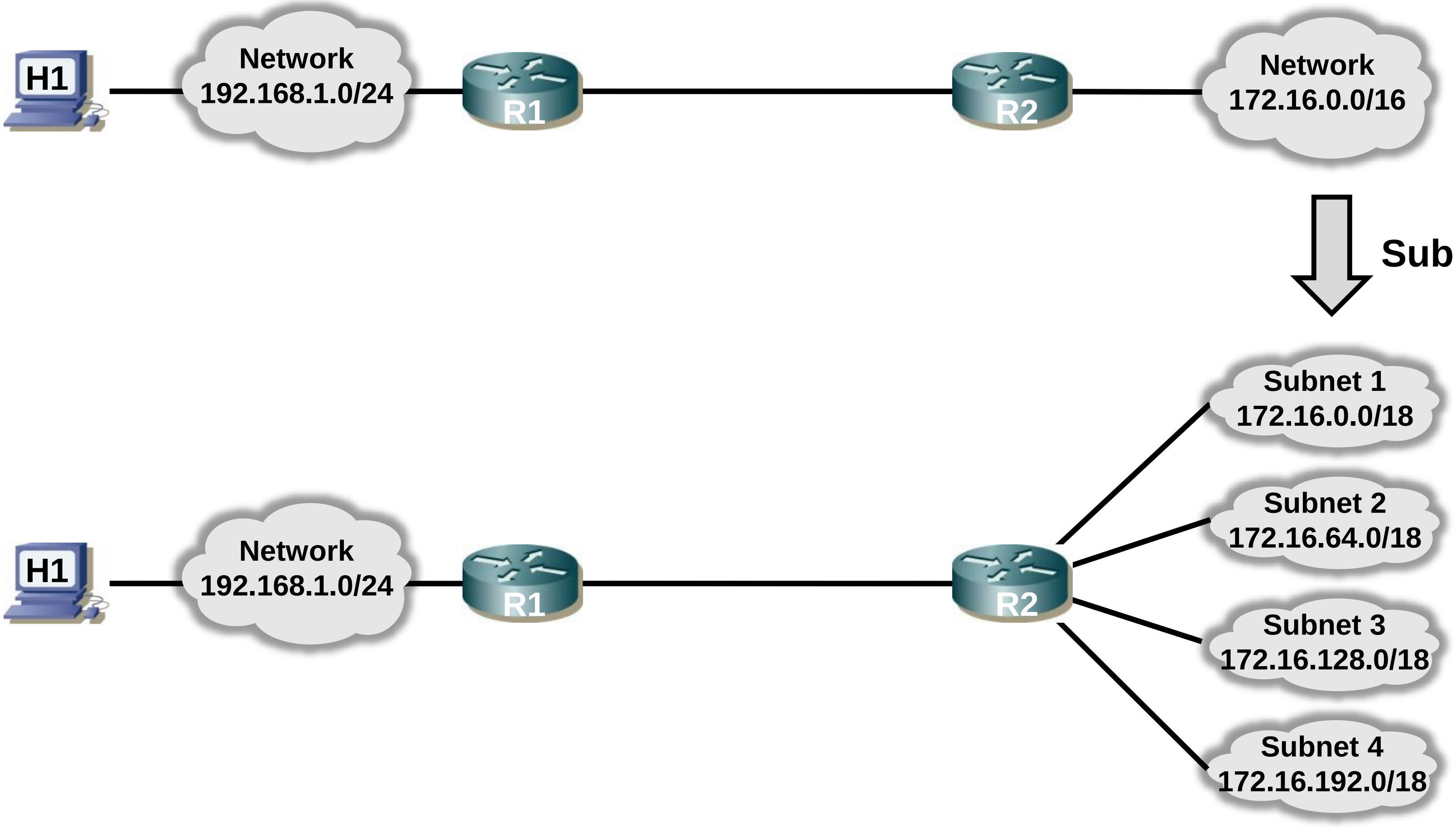
| | | | | |
|---|---|---|---|---|
| IP packet destination IP address (10.20.30.12) in binary notation | 00001010 | 00010100 | 00011110 | 00001100 |
| Network mask 255.255.0.0 in binary notation | 11111111 | 11111111 | 00000000 | 00000000 |
| Route entry destination address in binary notation | 00001010 | 00010100 | 00000000 | 00000000 |

**Match length: 16 bits**

# 5. Routing and IP forwarding examples (9/13)

## E. Subnetting:

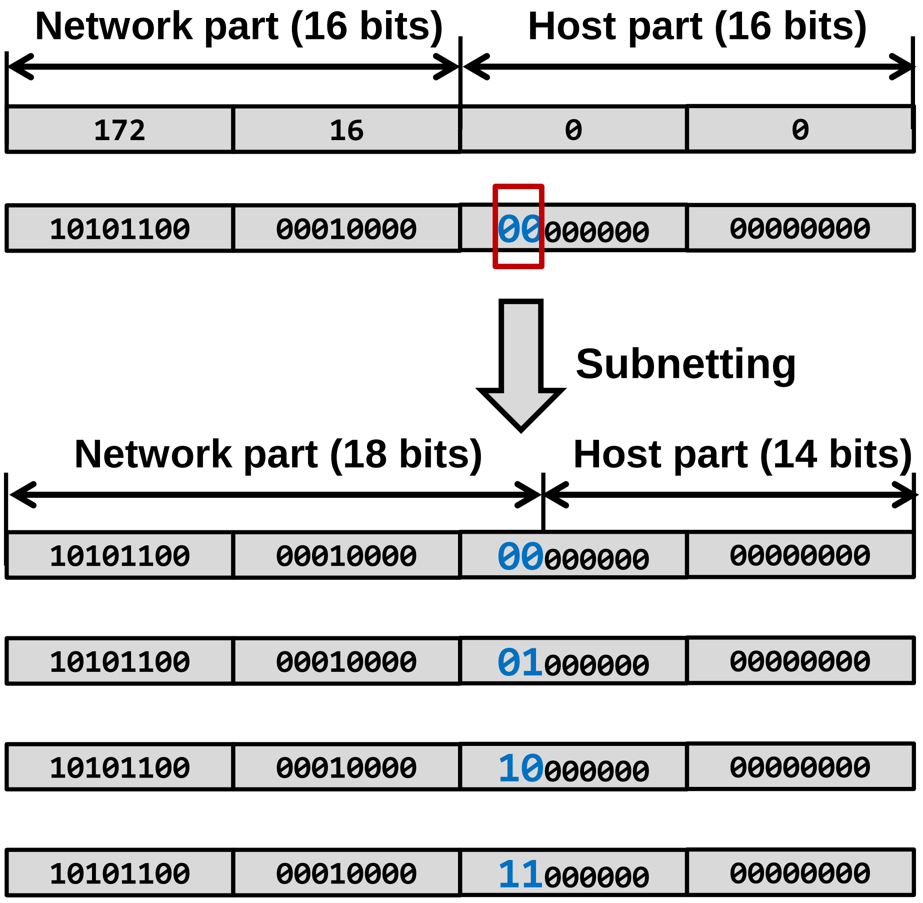
**Subnetting splits a network into a number of smaller networks for the purpose of logical separation. In the figure below, the network 172.16.0.0/16 is split into 4 subnets with equal size.**



# 5. Routing and IP forwarding examples (10/13)

## E. Subnetting:

**In subnetting, some of the leftmost bits of the host part of a network address are used for enumerating the subnets, thus increasing the length of the network part of the address.**

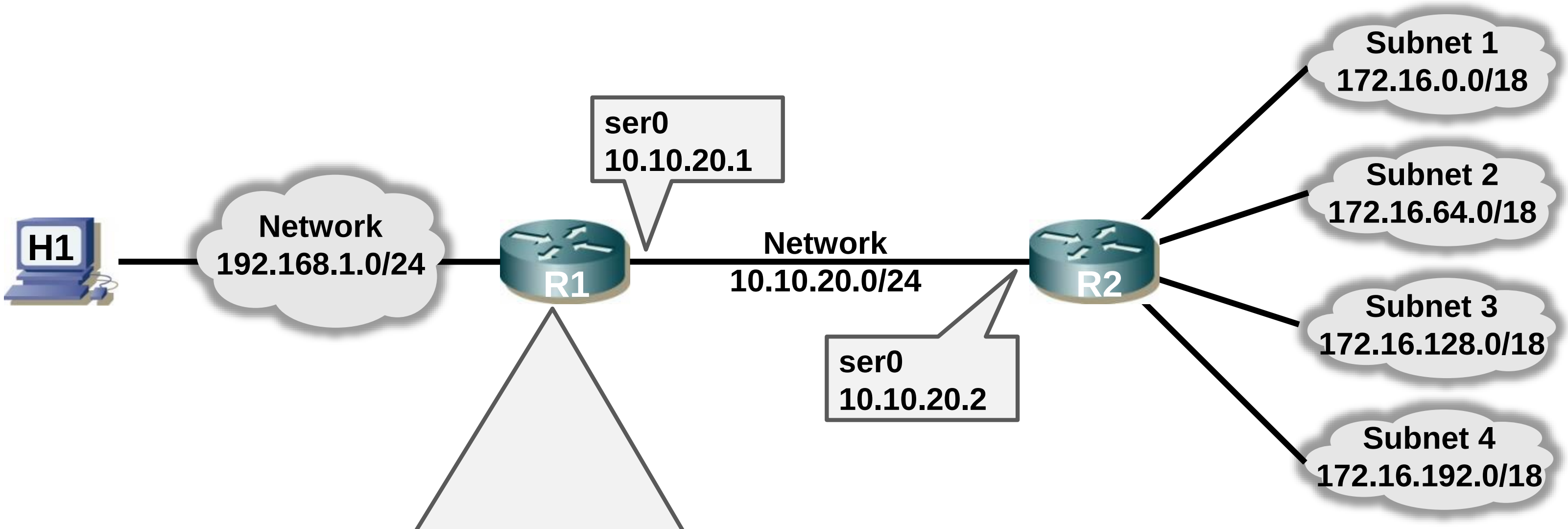


| | | | | |
|---|---|---|---|---|
| **Network 172.16.0.0/16 in dotted-decimal notation** | 172 | 16 | 0 | 0 |
| **Network 172.16.0.0/16 in binary notation** | 10101100 | 00010000 | 00000000 | 00000000 |
| **Subnet 172.16.0.0/18 in binary notation** | 10101100 | 00010000 | 00000000 | 00000000 |
| **Subnet 172.16.64.0/18 in binary notation** | 10101100 | 00010000 | 01000000 | 00000000 |
| **Subnet 172.16.128.0/18 in binary notation** | 10101100 | 00010000 | 10000000 | 00000000 |
| **Subnet 172.16.192.0/18 in binary notation** | 10101100 | 00010000 | 11000000 | 00000000 |

# 5. Routing and IP forwarding examples (11/13)

## E. Subnetting:

**Subnetting is transparent to other routers in the network.**
**In the example below, R1 can still reach all 4 subnets with the same routing table entry.**



| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| X.X.X.X | X.X.X.X | X.X.X.X | XXX |
| 172.16.0.0 | 255.255.0.0 | 10.10.20.2 | ser0 |
| X.X.X.X | X.X.X.X | X.X.X.X | XXX |

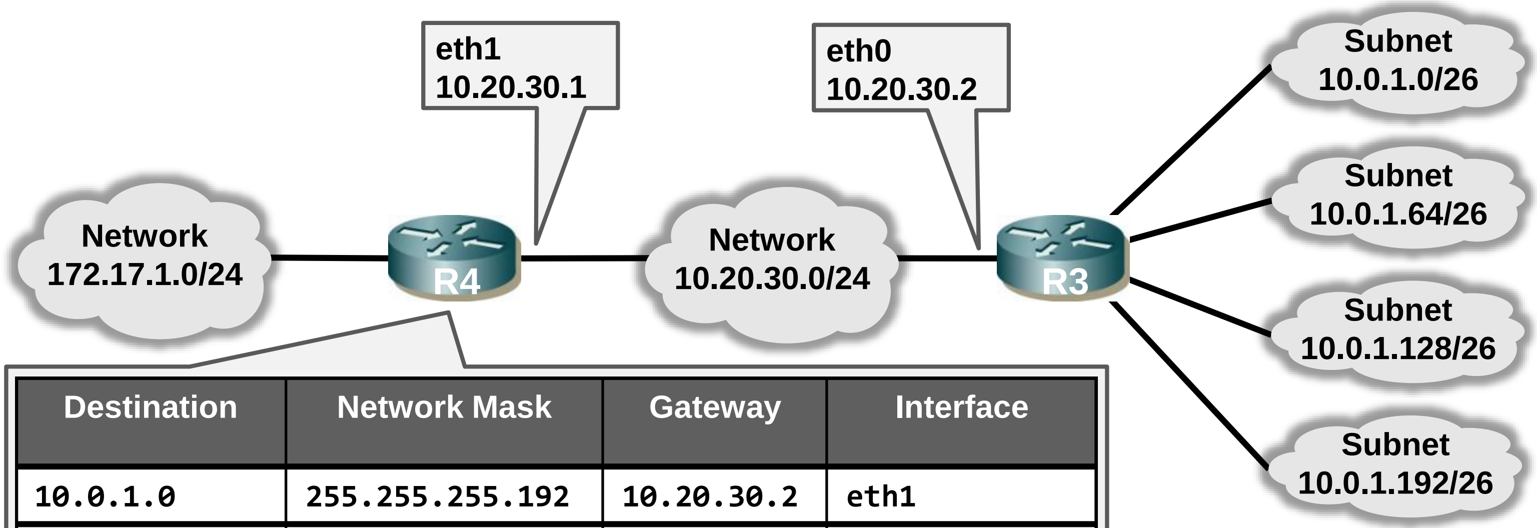
Summary route for subnets 1 through 4

# 5. Routing and IP forwarding examples (12/13)

## F. Supernetting:

**Supernetting aggregates multiple route entries into a single entry, thus reducing the number of route entries. Other terms for supernetting are "route aggregation", "prefix aggregation" or "route summarization".**

**In the example below, 4 route entries on R4 are aggregated into a single route entry.**



| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| 10.0.1.0 | 255.255.255.192 | 10.20.30.2 | eth1 |
| 10.0.1.64 | 255.255.255.192 | 10.20.30.2 | eth1 |
| 10.0.1.128 | 255.255.255.192 | 10.20.30.2 | eth1 |
| 10.0.1.192 | 255.255.255.192 | 10.20.30.2 | eth1 |

Route aggregation

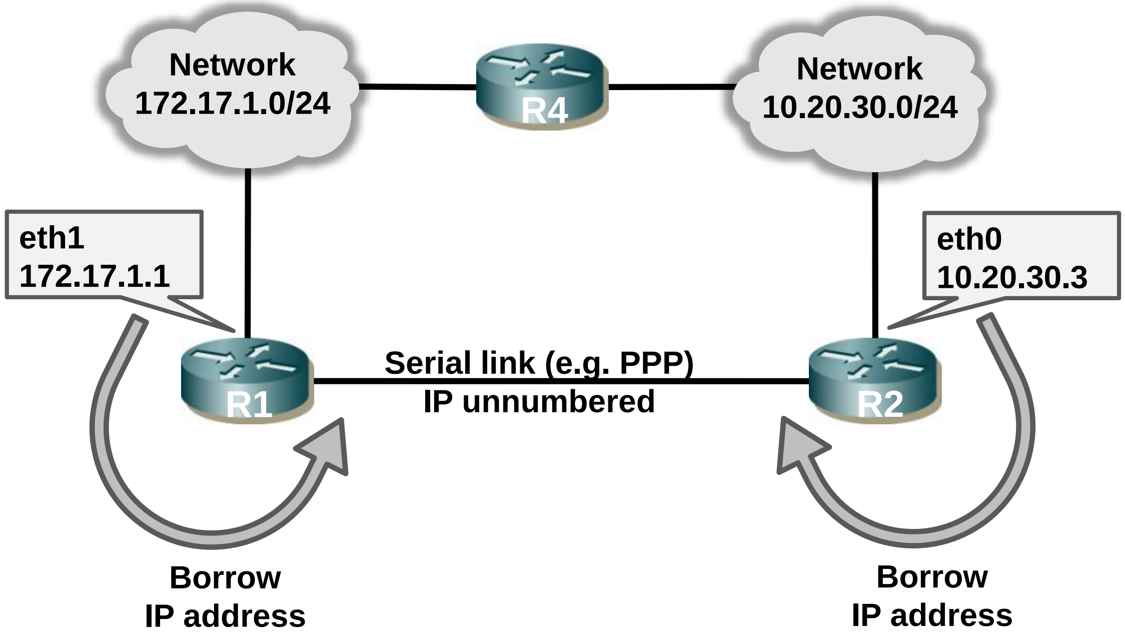
| Destination | Network Mask | Gateway | Interface |
|---|---|---|---|
| 10.0.1.0 | 255.255.0.0 | 10.20.30.2 | eth1 |

# 5. Routing and IP forwarding examples (13/13)

## G. Unnumbered link:

**IP unnumbered links are serial links without assigning unique IP addresses to the interfaces of the link.**
**The goal of unnumbered links (also known as 'IP unnumbered') is to conserve IP addresses.**
**IP unnumbered is restricted to serial links, i.e. point to point links without multiple access like Ethernet.**

**In order to enable IP processing on an interface without specific IP address, an IP address from another interface is 'borrowed' to the serial interface.**


Network
172.17.1.0/24
R4
Network
10.20.30.0/24
eth1
172.17.1.1
eth0
10.20.30.3
R1
Serial link (e.g. PPP)
IP unnumbered
R2
Borrow
IP address
Borrow
IP address