Internet Protocol Stack

- **Application**: supporting network applications
  - FTP, SMTP, HTTP
- **Transport**: data transfer between processes
  - TCP, UDP
- **Network**: routing of datagrams from source to destination
  - IP, routing protocols
- **Link**: data transfer between neighboring network elements
  - Ethernet, WiFi
- **Physical**: bits “on the wire”
  - Coaxial cable, optical fibers, radios
Network Layer and Mobile IP
Outline

- IP addresses
- Virtual circuit vs. datagrams
- Routing algorithms
  - Link state
  - Distance vector
- Mobile IP
  - Architecture
- Encapsulation
Network Layer in Internet: Big Picture

Host, router network layer functions:

- **Transport layer**: TCP, UDP
- **Network layer protocol (e.g., IP)**
  - addressing conventions
  - packet format
  - packet handling conventions
- **Control protocols (e.g., ICMP)**
  - error reporting
  - router “signaling”

- **Routing protocols**
  - path selection

Network layer

Link layer

physical layer
Network layer functions

- transport packet from sending to receiving hosts
- network layer protocols in every host, router

Three important functions:

- Path determination: route taken by packets from source to dest. Routing algorithms
- Switching: move packets from router’s input to appropriate router output
- Call setup: some network architectures require router call setup along path before data flows
**IP Datagram Format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP protocol version number</td>
<td>32 bits</td>
</tr>
<tr>
<td>header length (bytes)</td>
<td>16-bit identifier</td>
</tr>
<tr>
<td>&quot;type&quot; of data</td>
<td>time to live</td>
</tr>
<tr>
<td>max number remaining hops (decremented at each router)</td>
<td>upper layer</td>
</tr>
<tr>
<td>upper layer protocol to deliver payload to</td>
<td>Internet checksum</td>
</tr>
<tr>
<td>total datagram length (bytes)</td>
<td>fragment offset</td>
</tr>
<tr>
<td>for fragmentation/reassembly</td>
<td>32 bit source IP address</td>
</tr>
<tr>
<td>32 bit destination IP address</td>
<td>Options (if any)</td>
</tr>
<tr>
<td>data (variable length, typically a TCP or UDP segment)</td>
<td>E.g. timestamp, record route taken, specify list of routers to visit.</td>
</tr>
</tbody>
</table>
IP Addressing: introduction

- IP address: 32-bit identifier for host, router interface
- Interface: connection between host/router and physical link
  - Router’s typically have multiple interfaces
  - Host typically has one interface
  - IP addresses associated with each interface

223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1 1
Subnets

- **IP address:**
  - subnet part (high order bits)
  - host part (low order bits)

- **What’s a subnet?**
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router

network consisting of 3 subnets
Subnets

How many?
**IP addressing: CIDR**

**CIDR: Classless InterDomain Routing**

- subnet portion of address of arbitrary length
- address format: `a.b.c.d/x`, where `x` is # bits in subnet portion of address

```
11001000  00010111 00010000 00000000
```

200.23.16.0/23
IP addresses: how to get one?

Q: How does host get IP address?

- Hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from server
**IP addresses: how to get one?**

**Q:** How does network get subnet part of IP addr?

**A:** gets allocated portion of its provider ISP’s address space

<table>
<thead>
<tr>
<th>ISP's block</th>
<th>11001000 00010111 00010000 00000000</th>
<th>200.23.16.0/20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization 0</td>
<td>11001000 00010111 00010000 00000000</td>
<td>200.23.16.0/23</td>
</tr>
<tr>
<td>Organization 1</td>
<td>11001000 00010111 00010010 00000000</td>
<td>200.23.18.0/23</td>
</tr>
<tr>
<td>Organization 2</td>
<td>11001000 00010111 00010100 00000000</td>
<td>200.23.20.0/23</td>
</tr>
<tr>
<td>...</td>
<td>.....</td>
<td>....</td>
</tr>
<tr>
<td>Organization 7</td>
<td>11001000 00010111 00011110 00000000</td>
<td>200.23.30.0/23</td>
</tr>
</tbody>
</table>
Hierarchical addressing allows efficient advertisement of routing information:

**Hierarchical addressing:**

- **Organization 0**
  - 200.23.16.0/23
- **Organization 1**
  - 200.23.18.0/23
- **Organization 2**
  - 200.23.20.0/23
- **Organization 7**
  - 200.23.30.0/23

**Fly-By-Night-ISP**

- "Send me anything with addresses beginning 200.23.16.0/20"

**ISPs-R-Us**

- "Send me anything with addresses beginning 199.31.0.0/16"

**Internet**
Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1

Organization 0
200.23.16.0/23
Organization 2
200.23.20.0/23
Organization 7
200.23.30.0/23
Organization 1
200.23.18.0/23

Fly-By-Night-ISP

“Send me anything with addresses beginning 200.23.16.0/20”

ISPs-R-Us

“Send me anything with addresses beginning 199.31.0.0/16 or 200.23.18.0/23”

Internet
IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers
   - allocates addresses
   - manages DNS
   - assigns domain names, resolves disputes
Network service model

Q: What service model for “channel” transporting packets from sender to receiver?
- guaranteed bandwidth?
- preservation of inter-packet timing (no jitter)?
- loss-free delivery?
- in-order delivery?
- congestion feedback to sender?

The most important abstraction provided by network layer:
- virtual circuit or datagram?

CRUCIAL question!
Virtual circuits

“source-to-dest path behaves much like telephone circuit”
- performance-wise
- network actions along source-to-dest path

- call setup, teardown for each call before data can flow
- each packet carries VC identifier (not destination host ID)
- every router on source-dest path maintains “state” for each passing connection
  - transport-layer connection only involved two end systems
- link, router resources (bandwidth, buffers) may be allocated to VC
  - to get circuit-like perf.
Virtual circuits: signaling protocols

- used to setup, maintain teardown VC
- used in ATM, frame-relay, X.25
- not used in today’s Internet
- Pros and cons?

1. Initiate call
2. incoming call
3. Accept call
4. Call connected
5. Data flow begins
6. Receive data
Datagram networks: the Internet model

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of “connection”
- packets typically routed using destination host ID
  - packets between same source-dest pair may take different paths

1. Send data
2. Receive data
Datagram or VC network: why?

Internet
- data exchanged among computers
  - "elastic" service, no strict timing req.
- "smart" end systems (computers)
  - can adapt, perform cong. control, error recovery
  - simple inside network, complexity at "edge"
- many link types
  - different characteristics
  - uniform service difficult

ATM
- evolved from telephony
- human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- "dumb" end systems
  - telephones
  - complexity inside network
Outline

- IP addresses
- Virtual circuit vs. datagrams
- Routing algorithms
  - Link state
  - Distance vector
- Mobile IP
  - Architecture
- Encapsulation
Routing

Routing protocol:

Goal: determine “good” path (sequence of routers) thru network from source to dest.

Graph abstraction for routing algorithms:

- graph nodes are routers
- graph edges are physical links
  - What are possible link cost metrics?

- “good” path:
  - typically means minimum cost path
  - other def’s possible
Routing Algorithm classification

Global or decentralized information?

Global:
- all routers have complete topology, link cost info
- "link state" algorithms

Decentralized:
- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Static or dynamic?

Static:
- routes change slowly over time

Dynamic:
- routes change more quickly
  - periodic update
  - in response to link cost changes
A Link-State Routing Algorithm

Dijkstra’s algorithm

- net topology, link costs known to all nodes
  - accomplished via “link state broadcast”
  - all nodes have same info
- computes least cost paths from one node (“source”) to all other nodes
  - gives routing table for that node
- iterative: after k iterations, know least cost path to k dest.’s

Notation:

- \( c(i,j) \): link cost from node i to j. cost infinite if not direct neighbors
- \( D(v) \): current value of cost of path from source to dest. \( V \)
- \( p(v) \): predecessor node along path from source to v, that is next v
- \( N \): set of nodes whose least cost path definitively known
Dijkstra's Algorithm

1 **Initialization:**
2 \( N = \{A\} \)
3 for all nodes \( v \)
4 \( \quad \) if \( v \) adjacent to \( A \)
5 \( \quad \) then \( D(v) = c(A,v) \)
6 \( \quad \) else \( D(v) = \infty \)
7
8 **Loop**
9 find \( w \) not in \( N \) such that \( D(w) \) is a minimum
10 add \( w \) to \( N \)
11 update \( D(v) \) for all \( v \) adjacent to \( w \) and not in \( N \): 
12 \( D(v) = \min( D(v), D(w) + c(w,v) ) \)
13 /* new cost to \( v \) is either old cost to \( v \) or known 
14 shortest path cost to \( w \) plus cost from \( w \) to \( v \ */
15 **until all nodes in \( N \)**
Dijkstra's algorithm: example

<table>
<thead>
<tr>
<th>Step</th>
<th>start N</th>
<th>D(B),p(B)</th>
<th>D(C),p(C)</th>
<th>D(D),p(D)</th>
<th>D(E),p(E)</th>
<th>D(F),p(F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>2,A</td>
<td>5,A</td>
<td>1,A</td>
<td>infinity</td>
<td>infinity</td>
</tr>
<tr>
<td>1</td>
<td>AD</td>
<td>2,A</td>
<td>4,D</td>
<td>2,D</td>
<td>infinity</td>
<td>infinity</td>
</tr>
<tr>
<td>2</td>
<td>ADE</td>
<td>2,A</td>
<td>3,E</td>
<td></td>
<td>4,E</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ADEB</td>
<td></td>
<td>3,E</td>
<td></td>
<td>4,E</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ADEBC</td>
<td></td>
<td></td>
<td></td>
<td>4,E</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ADEBCF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
A        \\
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
B        \\
| 2  | 1  | 5  |
| 3  |    |    |
C        \\
| 3  |    |    |
D        \\
| 1  | 3  | 1  |
| 2  | 1  |    |
E        \\
| 1  | 3  | 2  |
| 1  |    |    |
F        \\
```
**Distance Vector Routing Algorithm**

**iterative:**
- continues until no nodes exchange info.
- *self-terminating:* no "signal" to stop

**asynchronous:**
- nodes need *not* exchange info/iterate in lock step!

**distributed:**
- each node communicates *only* with directly-attached neighbors

**Each node:**

1. **wait** for (change in local link cost of msg from neighbor)
2. **recompute** distance table
3. if least cost path to any dest has changed, **notify** neighbors
Distance Vector Algorithm: Data Structures

- Each node $x$ maintains:
  - For each neighbor $v$, cost $c(x,v)$
  - Node $x$'s distance vector: $D_x = [D_x(y): y \in N]$ containing $x$'s estimate of cost to all destinations
  - Distance vectors for each neighbor $v$: $D_v = [D_v(y): y \in N]$

- Basic operation: Bellman-Ford algorithm
  $$D_x(y) = \min_v \{c(x,v) + D_v(y)\} \quad y \in N$$
Distance Vector Algorithm:

At all nodes, X:

1 Initialization:
2 For all destinations $y \in N$:
3 \[ D_x(y) = c(x,y) \quad \text{/* if } y \text{ is not a neighbor, then } c(x,y) = \infty */ \]
4 For each neighbor $w$
5 \[ D_w(y) = \infty \text{ for all destinations } y \in N \]
6 For each neighbor $w$
7 Send distance vector $D_x = [D_x(y): y \in N]$ to $w$

8 Loop:
9 Wait (until communication from neighbor $w$)
10 For each $y \in N$:
11 \[ D_x(y) = \min_v \{ c(x,v) + D_v(y) \} \]
12 If $D_x(y)$ changes for any destination $y$
13 Send distance vector $D_x = [D_x(y): y \in N]$ to all neighbors
Distance Vector: link cost changes

Link cost changes:
- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors
Distance Vector: link cost changes

Link cost changes:
- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors

“good news travels fast”

At time \( t_0 \), \( y \) detects the link-cost change, updates its DV, and informs its neighbors.

At time \( t_1 \), \( z \) receives the update from \( y \) and updates its table. It computes a new least cost to \( x \) and sends its neighbors its DV.

At time \( t_2 \), \( y \) receives \( z \)'s update and updates its distance table. \( y \)'s least costs do not change and hence \( y \) does not send any message to \( z \).
Distance Vector: link cost changes
Distance Vector: link cost changes

\[ Dy(x) = \min \{c(y,x) + Dx(x), c(y,z) + Dz(x)\} \]
\[ = \min \{60 + 0, 1 + 5\} = 6 \]
Distance Vector: link cost changes

Link cost changes:
- good news travels fast
- bad news travels slow - "count to infinity" problem!
- 44 iterations before algorithm stabilizes

Poissoned reverse:
- If Z routes through Y to get to X:
  - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- will this completely solve count to infinity problem?
Comparison of LS and DV algorithms

Message complexity
- with $n$ nodes, $E$ links
- **LS:**
- **DV:**
Comparison of LS and DV algorithms II

Message complexity
- **LS:** with $n$ nodes, $E$ links, $O(nE)$ msgs sent each
- **DV:** exchange between neighbors only
  - convergence time varies

Speed of Convergence
- **LS:**
- **DV:**
Comparison of LS and DV algorithms

Message complexity

- **LS**: with \( n \) nodes, \( E \) links, \( O(nE) \) msgs sent each
- **DV**: exchange between neighbors only
  - convergence time varies

Speed of Convergence

- **LS**: \( O(n^2) \) algorithm requires \( O(nE) \) msgs
  - may have oscillations
- **DV**: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?
Comparison of LS and DV algorithms

Message complexity
- **LS**: with n nodes, E links, \( O(nE) \) msgs sent each
- **DV**: exchange between neighbors only
  - convergence time varies

Speed of Convergence
- **LS**: \( O(n^2) \) algorithm requires \( O(nE) \) msgs
  - may have oscillations
- **DV**: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?
- **LS**:
  - node can advertise incorrect link cost
  - each node computes only its own table
- **DV**:
  - DV node can advertise incorrect path cost
  - each node's table used by others
    - error propagate thru network
Hierarchical Routing

Our routing review thus far - idealization
- all routers identical
- network “flat”
... not true in practice

scale: with 200 million destinations:
- can’t store all dest’s in routing tables!
- routing table exchange would swamp links!

administrative autonomy
- internet = network of networks
- each network admin may want to control routing in its own network
Hierarchical Routing

- aggregate routers into regions, “autonomous systems” (AS)

- routers in same AS run same routing protocol
  - “intra-AS” routing protocol
  - routers in different AS can run different intra-AS routing protocol

- gateway routers
  - special routers in AS
  - run intra-AS routing protocol with all other routers in AS
  - also responsible for routing to destinations outside AS
    - run inter-AS routing protocol with other gateway routers
IntraAS and InterAS routing

Gateways:
• perform inter-AS routing amongst themselves
• perform intra-AS routers with other routers in their AS

inter-AS, intra-AS routing in gateway A.c
IntraAS and InterAS routing

Intra-AS routing within AS A

Inter-AS routing between A and B

Internet: OSPF, IS-IS, RIP

Internet: BGP

Intra-AS routing within AS B

Host h1

Host h2
Discussion

- **IP works fine for the Internet**
  - it has problems; but during vast majority of the time it gets its job done efficiently—moving a packet from a src. to a dest.

- **What problem can mobility cause?**
Motivation for Mobile IP

- Routing
  - based on IP destination address, network prefix (e.g. 129.13.42) determines physical subnet
  - change of physical subnet implies change of IP address to have a topological correct address (standard IP) or needs special entries in the routing tables
Motivation for Mobile IP

- **Routing**
  - based on IP destination address, network prefix (e.g. 129.13.42) determines physical subnet
  - change of physical subnet implies change of IP address to have a topological correct address (standard IP) or needs special entries in the routing tables

- **Keeping the IP address while moving**
  - Specific routes to end-systems
  - change of all routing table entries to forward packets to the right destination
  - does not scale with the number of mobile hosts and frequent changes in the location, security problems

- **Changing the IP address**
  - adjust the host IP address depending on the current location
  - almost impossible to find a mobile system, DNS updates take a long time
  - TCP connections break, security problems
Requirements to Mobile IP (RFC 3344, was: 3220, was: 2002)

- **Transparency**
  - mobile end-systems keep their IP address
  - continuation of communication after interruption of link possible
  - point of connection to the fixed network can be changed

- **Compatibility**
  - support of the same layer 2 protocols as IP
  - no changes to current end-systems and routers required
  - mobile end-systems can communicate with fixed systems

- **Security**
  - authentication of all registration messages

- **Efficiency and scalability**
  - only little additional messages to the mobile system required (connection typically via a low bandwidth radio link)
  - world-wide support of a large number of mobile systems in the whole Internet
Terminology

- **Mobile Node (MN)**
  - system (node) that can change the point of connection to the network without changing its IP address

- **Home Agent (HA)**
  - system in the home network of the MN, typically a router
  - registers the location of the MN, tunnels IP datagrams to the COA

- **Foreign Agent (FA)**
  - system in the current foreign network of the MN, typically a router
  - forwards the tunneled datagrams to the MN, typically also the default router for the MN

- **Care-of Address (COA)**
  - address of the current tunnel end-point for the MN (at FA or MN)
  - actual location of the MN from an IP point of view
  - can be chosen, e.g., via DHCP

- **Correspondent Node (CN)**
  - communication partner
Example network 1

- HA
- Internet
- home network (physical home network for the MN)
- CN
- end-system
- router
Example network 2

HA

router

home network

(physical home network for the MN)

Internet

router

Foreign

network

router

mobile end-system

CN

end-system

router

foreign network

(current physical network for the MN)
Data transfer to the mobile system

1. Sender sends to the IP address of MN, HA intercepts packet (proxy ARP)
2. HA tunnels packet to COA, here FA, by encapsulation
3. FA forwards the packet to the MN
Data transfer from the mobile system

1. Sender sends to the IP address of the receiver as usual, FA works as default router.
Overview

1. CN
2. home network
3. router HA
4. router FA
5. MN
6. foreign network
7. Internet
8. COA
9. 1.
10. 2.
11. 3.
12. 4.
13. CN
14. router
15. home network
16. router HA
17. router FA
18. MN
19. foreign network
Network integration

- Agent Advertisement
  - HA and FA periodically send advertisement messages into their physical subnets
  - MN listens to these messages and detects, if it is in the home or a foreign network (standard case for home network)
  - MN reads a COA from the FA advertisement messages

- Registration (always limited lifetime!)
  - MN signals COA to the HA via the FA, HA acknowledges via FA to MN
  - these actions have to be secured by authentication

- Advertisement
  - HA advertises the IP address of the MN (as for fixed systems), i.e. standard routing information
  - routers adjust their entries, these are stable for a longer time (HA responsible for a MN over a longer period of time)
  - packets to the MN are sent to the HA,
  - independent of changes in COA/FA
## Agent advertisement

**ICMP packets**
- **type = 16**
- **length = 6 + 4 * #COAs**
- **R**: registration required
- **B**: busy, no more registrations
- **H**: home agent
- **F**: foreign agent
- **M**: minimal encapsulation
- **G**: GRE encapsulation
- **r**: =0, ignored (former Van Jacobson compression)
- **T**: FA supports reverse tunneling
- **reserved**: =0, ignored

<table>
<thead>
<tr>
<th>0</th>
<th>7</th>
<th>8</th>
<th>15</th>
<th>16</th>
<th>23</th>
<th>24</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>code</td>
<td>checksum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#addresses</td>
<td>addr. size</td>
<td>lifetime</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>router address 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>preference level 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>router address 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>preference level 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>. . .</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>type = 16</th>
<th>length</th>
<th>sequence number</th>
</tr>
</thead>
<tbody>
<tr>
<td>registration lifetime</td>
<td>R</td>
<td>B</td>
</tr>
<tr>
<td>COA 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COA 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>. . .</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Registration

MN -> HA
registration request
registration reply

MN -> FA
registration request
registration reply

FA -> HA
registration request
registration reply
**Mobile IP registration request**

```
<table>
<thead>
<tr>
<th>0</th>
<th>7</th>
<th>8</th>
<th>15</th>
<th>16</th>
<th>23</th>
<th>24</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>type = 1</td>
<td>SB</td>
<td>DMG</td>
<td>r</td>
<td>T</td>
<td>x</td>
<td>lifetime</td>
<td></td>
</tr>
<tr>
<td>home address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>home agent</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>extensions . . .</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

UDP packets
S: simultaneous bindings
B: broadcast datagrams
D: decapsulation by MN
M minimal encapsulation
G: GRE encapsulation
r: =0, ignored
T: reverse tunneling requested
x: =0, ignored
Mobile IP registration reply

<table>
<thead>
<tr>
<th>0</th>
<th>7</th>
<th>8</th>
<th>15</th>
<th>16</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>type = 3</td>
<td>code</td>
<td>lifetime</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>home address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>home agent</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>extensions . . .</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example codes:

registration successful
0 registration accepted
1 registration accepted, but simultaneous mobility bindings unsupported

registration denied by FA
65 administratively prohibited
66 insufficient resources
67 mobile node failed authentication
68 home agent failed authentication
69 requested Lifetime too long

registration denied by HA
129 administratively prohibited
131 mobile node failed authentication
133 registration Identification mismatch
135 too many simultaneous mobility bindings
Encapsulation 1

Diagram:
- Home network
- Router HA
- Router FA
- Internet
- MN
- Foreign network

Steps:
1. CN -> router
2. Router HA -> Router FA
3. Router FA -> MN
4. MN -> Foreign network
Encapsulation II

<table>
<thead>
<tr>
<th>original IP header</th>
<th>original data</th>
</tr>
</thead>
<tbody>
<tr>
<td>new IP header</td>
<td>new data</td>
</tr>
<tr>
<td>outer header</td>
<td>inner header</td>
</tr>
</tbody>
</table>
Encapsulation I

- Encapsulation of one packet into another as payload
  - e.g. IPv6 in IPv4 (6Bone), Multicast in Unicast (Mbone)
  - here: e.g. IP-in-IP-encapsulation, minimal encapsulation or GRE (Generic Record Encapsulation)

  - tunnel between HA and COA

```
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ver.</td>
<td>IHL</td>
<td>DS (TOS)</td>
<td>length</td>
</tr>
<tr>
<td>IP identification</td>
<td>flags</td>
<td>fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>IP-in-IP</td>
<td>IP checksum</td>
<td></td>
</tr>
</tbody>
</table>

IP address of HA

Care-of address COA

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ver.</td>
<td>IHL</td>
<td>DS (TOS)</td>
<td>length</td>
</tr>
<tr>
<td>IP identification</td>
<td>flags</td>
<td>fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>lay. 4 prot.</td>
<td>IP checksum</td>
<td></td>
</tr>
</tbody>
</table>

IP address of CN

IP address of MN

TCP/UDP/...payload
```
Encapsulation II

- Minimal encapsulation (optional)
  - avoids repetition of identical fields
  - e.g. TTL, IHL, version, DS (RFC 2474, old: TOS)
  - only applicable for unfragmented packets, no space left for fragment identification

<table>
<thead>
<tr>
<th>ver.</th>
<th>IHL</th>
<th>DS (TOS)</th>
<th>length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP identification</td>
<td>flags</td>
<td>fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>min. encap.</td>
<td>IP checksum</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

IP address of HA

Care-of address COA

<table>
<thead>
<tr>
<th>lay. 4 protoc.</th>
<th>S</th>
<th>reserved</th>
<th>IP checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP address of MN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Original sender IP address (if S=1)

TCP/UDP/ ... payload
Generic Routing Encapsulation

RFC 1701

<table>
<thead>
<tr>
<th>ver.</th>
<th>IHL</th>
<th>DS (TOS)</th>
<th>length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP identification</td>
<td>flags</td>
<td>fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>GRE</td>
<td>IP checksum</td>
<td></td>
</tr>
<tr>
<td>IP address of HA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Care-of address COA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>rec.</td>
<td>rsv.</td>
<td>ver.</td>
</tr>
<tr>
<td>checksum (optional)</td>
<td>offset (optional)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>key (optional)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sequence number (optional)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>routing (optional)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ver.</td>
<td>IHL</td>
<td>DS (TOS)</td>
<td>length</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP identification</td>
<td>flags</td>
<td>fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>lay. 4 prot.</td>
<td>IP checksum</td>
<td></td>
</tr>
<tr>
<td>IP address of CN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP address of MN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP/UDP/ ... payload</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RFC 2784

<table>
<thead>
<tr>
<th>C</th>
<th>reserved0</th>
<th>ver.</th>
<th>protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>checksum (optional)</td>
<td>reserved1 (=0)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Optimization of packet forwarding

- **Triangular Routing**
  - sender sends all packets via HA to MN
  - higher latency and network load

- **“Solutions”**
  - sender learns the current location of MN
  - direct tunneling to this location
  - HA informs a sender about the location of MN
  - big security problems!

- **Change of FA**
  - packets on-the-fly during the change can be lost
  - new FA informs old FA to avoid packet loss, old FA now forwards remaining packets to new FA
  - this information also enables the old FA to release resources for the MN
Reverse tunneling (RFC 3024, was: 2344)

1. MN sends to FA
2. FA tunnels packets to HA by encapsulation
3. HA forwards the packet to the receiver (standard case)

1. MN sends to FA
2. FA tunnels packets to HA by encapsulation
3. HA forwards the packet to the receiver (standard case)
Mobile IP with reverse tunneling

- Router accept often only “topological correct“ addresses (firewall!)
  - a packet from the MN encapsulated by the FA is now topological correct
  - furthermore multicast and TTL problems solved (TTL in the home network correct, but MN is too far away from the receiver)

- Reverse tunneling does not solve
  - problems with firewalls, the reverse tunnel can be abused to circumvent security mechanisms (tunnel hijacking)
  - optimization of data paths, i.e. packets will be forwarded through the tunnel via the HA to a sender (double triangular routing)

- The standard is backwards compatible
  - the extensions can be implemented easily and cooperate with current implementations without these extensions
  - Agent Advertisements can carry requests for reverse tunneling
Mobile IP and IPv6

- Mobile IP was developed for IPv4, but IPv6 simplifies the protocols
  - security is integrated and not an add-on, authentication of registration is included
  - COA can be assigned via auto-configuration (DHCPv6 is one candidate), every node has address autoconfiguration
  - no need for a separate FA, all routers perform router advertisement which can be used instead of the special agent advertisement; addresses are always co-located
  - MN can signal a sender directly the COA, sending via HA not needed in this case (automatic path optimization)
  - “soft“ hand-over, i.e. without packet loss, between two subnets is supported
    - MN sends the new COA to its old router
    - the old router encapsulates all incoming packets for the MN and forwards them to the new COA
    - authentication is always granted
Problems with mobile IP

- **Security**
  - authentication with FA problematic, for the FA typically belongs to another organization
  - no protocol for key management and key distribution has been standardized in the Internet
  - patent and export restrictions

- **Firewalls**
  - typically mobile IP cannot be used together with firewalls, special set-ups are needed (such as reverse tunneling)

- **QoS**
  - many new reservations in case of RSVP
  - tunneling makes it hard to give a flow of packets a special treatment needed for the QoS

- **Security, firewalls, QoS etc. are topics of research and discussions!**
Security in Mobile IP

- Security requirements (Security Architecture for the Internet Protocol, RFC 1825)
Security in Mobile IP

- Security requirements (Security Architecture for the Internet Protocol, RFC 1825)
  - Integrity
    any changes to data between sender and receiver can be detected by the receiver
  - Authentication
    sender address is really the address of the sender and all data received is really data sent by this sender
  - Confidentiality
    only sender and receiver can read the data
  - Non-Repudiation
    sender cannot deny sending of data
  - Traffic Analysis
    creation of traffic and user profiles should not be possible
  - Replay Protection
    receivers can detect replay of messages
IP security architecture I

- Two or more partners have to negotiate security mechanisms to setup a security association
  - typically, all partners choose the same parameters and mechanisms

- Two headers have been defined for securing IP packets:
  - Authentication-Header
    - guarantees integrity and authenticity of IP packets
    - if asymmetric encryption schemes are used, non-repudiation can also be guaranteed

<table>
<thead>
<tr>
<th>IP header</th>
<th>authentication header</th>
<th>UDP/TCP data</th>
</tr>
</thead>
</table>

- Encapsulation Security Payload
  - protects confidentiality between communication partners

<table>
<thead>
<tr>
<th>IP header</th>
<th>ESP header</th>
<th>encrypted data</th>
</tr>
</thead>
<tbody>
<tr>
<td>not encrypted</td>
<td></td>
<td>encrypted</td>
</tr>
</tbody>
</table>
IP security architecture II

- Mobile Security Association for registrations
  - parameters for the mobile host (MH), home agent (HA), and foreign agent (FA)

- Extensions of the IP security architecture
  - extended authentication of registration

- prevention of replays of registrations
  - time stamps: 32 bit time stamps + 32 bit random number
  - nonces: 32 bit random number (MH) + 32 bit random number (HA)
Key distribution

- Home agent distributes session keys

- Foreign agent has a security association with the home agent
- Mobile host registers a new binding at the home agent
- Home agent answers with a new session key for foreign agent and mobile node

Diagram:
- HA (Home Agent) distributes session keys to FA (Foreign Agent)
- FA distributes session keys to MH (Mobile Host)
- HA sends response to FA: \( E_{HA-FA} \{\text{session key}\} \)
- HA sends response to MH: \( E_{HA-MH} \{\text{session key}\} \)
IP Micro-mobility support

- Micro-mobility support:
  - Efficient local handover inside a foreign domain without involving a home agent
  - Reduces control traffic on backbone
  - Especially needed in case of route optimization

- Example approaches:
  - Cellular IP
  - HAWAII
  - Hierarchical Mobile IP (HMIP)

- Important criteria: Efficiency, Security, Scalability, Transparency, Manageability
Recap

- Link state vs. distance vector
- Why do we need mobile IP?
- How does mobile IP work?
- What is encapsulation?
- How is encapsulation used in mobile IP?