Dynamic Host Configuration Protocol

- Application
  - simplification of installation and maintenance of networked computers
  - supplies systems with all necessary information, such as IP address, DNS server address, domain name, subnet mask, default router etc.
  - enables automatic integration of systems into an Intranet or the Internet, can be used to acquire a COA for Mobile IP

- Client/Server-Model
  - the client sends via a MAC broadcast a request to the DHCP server (might be via a DHCP relay)
DHCP – Protocol Mechanisms

DHCP Characteristics

- **Server**
  - several servers can be configured for DHCP, coordination not yet standardized (i.e., manual configuration)

- **Renewal of configurations**
  - IP addresses have to be requested periodically, simplified protocol

- **Options**
  - available for routers, subnet mask, NTP (network time protocol) timeserver, SLP (service location protocol) directory, DNS (domain name system)

- **Big security problems!**
  - no authentication of DHCP information specified
Mobility

**Home network**: permanent “home” of mobile (e.g., 128.119.40/24)

**Permanent address**: address in home network, can always be used to reach mobile (e.g., 128.119.40.186)

**Home agent**: entity that will perform mobility functions on behalf of mobile, when mobile is remote

**Visited network**: network in which mobile currently resides (e.g., 79.129.13/24)

**Care-of-address**: address in visited network (e.g., 79.129.13.2)

**Correspondent**: wants to communicate with mobile

**Permanent address**: remains constant (e.g., 128.119.40.186)
Finding Somebody

• Let routing handle it:
  • routers advertise permanent address of mobile-nodes-in-residence via usual routing table exchange
  • routing tables indicate where each mobile located
  • no changes to end-systems
  • NOT SCALABLE!

• Let end-systems handle it:
  • indirect routing: communication from correspondent to mobile goes through home agent, then forwarded to remote
  • direct routing: correspondent gets foreign address of mobile, sends directly to mobile

Mobility: Registration

End result:
• Foreign agent knows about mobile
• Home agent knows location of mobile
Mobility via Indirect Routing

Indirect Routing: Comments

- Mobile uses two addresses:
  - permanent address: used by correspondent (hence mobile location is *transparent* to correspondent)
  - care-of-address: used by home agent to forward datagrams to mobile
- foreign agent functions may be done by mobile itself
- triangle routing: correspondent-home-network-mobile
  - inefficient when correspondent, mobile are in same network
Indirect Routing: Moving Between Networks

- Suppose mobile user moves to another network
  - registers with new foreign agent
  - new foreign agent registers with home agent
  - home agent updates care-of-address for mobile
  - packets continue to be forwarded to mobile (but with new care-of-address)
- Mobility, changing foreign networks transparent: *ongoing connections can be maintained!*

Mobility via Direct Routing

1. Correspondent requests, receives foreign address of mobile
2. Correspondent forwards to foreign agent
3. Foreign agent receives packets, forwards to mobile
4. Mobile replies directly to correspondent
Mobility via Direct Routing: Comments

- Overcome triangle routing problem
- Non-transparent to correspondent: correspondent must get care-of-address from home agent
  - what if mobile changes visited network?

Accommodating Mobility with Direct Routing

- Anchor foreign agent: "anchor FA" in first visited network
- Data always routed first to anchor FA
- When mobile moves: new FA arranges to have data forwarded from old FA (chaining)
Mobile IP

- RFC 3220
- Has many features we’ve seen:
  - home agents, foreign agents, foreign-agent registration, care-of-addresses, encapsulation (packet-within-a-packet)
- Three components to standard:
  - indirect routing of datagrams
  - agent discovery
  - registration with home agent

Mobile IP: Indirect Routing

Permanent address: 128.119.40.186
Care-of address: 79.129.13.2

dest: 79.129.13.2
dest: 128.119.40.186

dest: 128.119.40.186

Packet sent by home agent to foreign agent: a packet within a packet

dest: 128.119.40.186

foreign-agent-to-mobile packet

dest: 128.119.40.186

Packet sent by correspondent

Mobile IP: Agent Discovery

- **Agent advertisement**: foreign/home agents advertise service by broadcasting ICMP messages (typefield = 9)

  - **H,F bits**: home and/or foreign agent
  - **R bit**: registration required

```
<table>
<thead>
<tr>
<th>0</th>
<th>8</th>
<th>16</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>code</td>
<td>checksum</td>
<td></td>
</tr>
<tr>
<td>router address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>length</td>
<td>sequence #</td>
<td></td>
</tr>
<tr>
<td>registration lifetime</td>
<td>RBHF/FGV reserved</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 or more care-of-addresses</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Mobile IP: Registration Example

- Home agent: HA: 128.119.40.7
- Foreign agent: COA: 79.129.13.2
- Visited network: 79.129.13/24
- ICMP agent adv.
- Mobile agent: MA: 128.119.40.186
Cell Network

- **MSC**
  - connects cells to wide area net
  - manages call setup
  - handles mobility

- **Cell**
  - covers geographical region
  - base station (BS) analogous to 802.11 AP
  - mobile users attach to network through BS
  - air-interface: physical and link layer protocol between mobile and BS

- **Public telephone network, and Internet**

- **Mobile Switching Center**

- **wired network**

---

Mobility Management

- Challenge: roaming message destination
  - Location management
  - Roaming management
  - Handoff management
Example: Cellular Networks

Mobility in Cellular Networks

- **Home network**: network of cellular provider you subscribe to (e.g., Sprint PCS, Verizon)
  - **home location register (HLR)**: database in home network containing permanent cell phone #, profile information (services, preferences, billing), information about current location (could be in another network)
- **Visited network**: network in which mobile currently resides
  - **visitor location register (VLR)**: database with entry for each user currently in network
  - could be home network
GSM: Indirect Routing

1. Call routed to home network
2. Home MSC consults HLR, gets roaming number of mobile in visited network
3. Home MSC sets up 2nd leg of call to MSC in visited network
4. MSC in visited network completes call through base station to mobile

GSM: Handoff with Common MSC

- Handoff goal: route call via new base station (without interruption)
- Reasons for handoff:
  - stronger signal to/from new BSS (continuing connectivity, less battery drain)
  - load balance: free up channel in current BSS
  - GSM doesn’t mandate why to perform handoff (policy), only how (mechanism)
- Handoff initiated by old BSS
GSM: Handoff with Common MSC

1. old BSS informs MSC of impending handoff, provides list of 1+ new BSSs
2. MSC sets up path (allocates resources) to new BSS
3. new BSS allocates radio channel for use by mobile
4. new BSS signals MSC, old BSS: ready
5. old BSS tells mobile: perform handoff to new BSS
6. mobile, new BSS signal to activate new channel
7. mobile signals via new BSS to MSC: handoff complete. MSC reroutes call
8. MSC-old-BSS resources released

GSM: Handoff Between MSCs

- **Anchor MSC**: first MSC visited during call
- call remains routed through anchor MSC
- new MSCs add on to end of MSC chain as mobile moves to new MSC
- IS-41 allows optional path minimization step to shorten multi-MSC chain
GSM: Handoff Between MSCs

- **Anchor MSC**: first MSC visited during call
- call remains routed through anchor MSC
- new MSCs add on to end of MSC chain as mobile moves to new MSC
- IS-41 allows optional path minimization step to shorten multi-MSC chain

Impact on Higher Layer Protocols

- logically, impact *should* be minimal …
  - best effort service model remains unchanged
  - TCP and UDP can (and do) run over wireless, mobile
- … but performance-wise:
  - packet loss/delay due to bit-errors (discarded packets, delays for link-layer retransmissions), and handoff
  - TCP interprets loss as congestion, will decrease congestion window unnecessarily
  - delay impairments for real-time traffic
  - limited bandwidth of wireless links
UDP – User Datagram Protocol

- Unreliable and unordered datagram service
- Adds multiplexing
- No flow control
- Endpoints identified by ports
  - servers have well-known ports
  - see /etc/services on Unix
- Header format
  - Optional checksum

TCP – Transmission Control Protocol

- Connection-oriented
- Byte-stream
  - app writes bytes
  - TCP sends segments
  - app reads bytes
- Full duplex
- Flow control: keep sender from overrunning receiver
- Congestion control: keep sender from overrunning network
TCP

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SrcPort</td>
<td></td>
</tr>
<tr>
<td>DstPort</td>
<td></td>
</tr>
<tr>
<td>SequenceNum</td>
<td></td>
</tr>
<tr>
<td>Acknowledgment</td>
<td></td>
</tr>
<tr>
<td>HdrLen</td>
<td>0</td>
</tr>
<tr>
<td>Flags</td>
<td></td>
</tr>
<tr>
<td>AdvertisedWindow</td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>UrgPtr</td>
<td></td>
</tr>
<tr>
<td>Options (variable)</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
</tr>
</tbody>
</table>

TCP’s 3-way handshake

Active participant (client)

- **SYN**, SequenceNum = \(x\)

Passive participant (server)

- **SYN+ACK**, SequenceNum = \(y\)
- **ACK**, Acknowledgment = \(x+1\)
- **ACK**, Acknowledgment = \(y+1\)
Connection Termination

First participant

\[\text{FIN, SequenceNum} = x\]

\[\text{ACK, Acknowledgment} = x+1\]

Second participant

\[\text{FIN, SequenceNum} = y\]

\[\text{Acknowledgment} = x+1\]

\[\text{ACK, Acknowledgment} = y+1\]

Motivation

- Transport protocols typically designed for
  - fixed end-systems
  - fixed, wired networks
- Research activities
  - performance
  - congestion control
  - efficient retransmissions
- TCP congestion control
  - packet loss in fixed networks typically due to (temporary) overload situations
  - router have to discard packets as soon as the buffers are full
  - TCP recognizes congestion only indirect via missing acknowledgements, retransmissions at full sending rate unwise, they would only contribute to the congestion and make it even worse
  - slow-start algorithm as reaction
Packet Loss

TCP Congestion Control

- Idea
  - assumes best-effort network (FIFO or FQ routers) each source determines network capacity for itself
  - uses implicit feedback
  - ACKs pace transmission (self-clocking)

- Challenge
  - determining the available capacity in the first place
  - adjusting to changes in the available capacity
AIMD

- Objective: adjust to changes in the available capacity
- New state variable per connection: \texttt{congestionWindow}
  - limits how much data source has in transit

\[
\text{MaxWin} = \text{MIN(CongestionWindow, AdvertisedWindow)}
\]
\[
\text{EffWin} = \text{MaxWin} - (\text{LastByteSent} - \text{LastByteAcked})
\]

- Idea:
  - increase \texttt{CongestionWindow} when congestion goes down
  - decrease \texttt{CongestionWindow} when congestion goes up

- Question: how does the source determine whether or not the network is congested?
- Answer: a timeout occurs
  - timeout signals that a packet was lost
  - packets are seldom lost due to transmission error
  - lost packet implies congestion

AIMD

- Algorithm
  - increment \texttt{CongestionWindow} by one packet per RTT (\textit{linear increase})
  - divide \texttt{CongestionWindow} by two whenever a timeout occurs (\textit{multiplicative decrease})
AIMD

- Sawtooth behavior

Slow Start

- Objective: determine the available capacity in the beginning
- Idea:
  - begin with $\text{CongestionWindow} = 1$ packet
  - double $\text{CongestionWindow}$ each RTT (increment by 1 packet for each ACK)
Slow Start

- Exponential growth, but slower than all at once
- Used…
  - when first starting connection
  - when connection goes dead waiting for timeout
- Trace

- Problem: lose up to half a Congestion Window’s worth of data

Fast Retransmit/Fast Recovery

- TCP sends an acknowledgement only after receiving a packet
- If a sender receives several acknowledgements for the same packet, this is due to a gap in received packets at the receiver
- However, the receiver got all packets up to the gap and is actually receiving packets
- Therefore, packet loss is not due to congestion, continue with current congestion window (do not use slow-start)
Fast Retransmit/Fast Recovery

- Fast recovery
  - skip the slow start phase
  - go directly to half the last successful CongestionWindow (ssthresh)

Mobility Affecting TCP-Mechanisms

- TCP assumes congestion if packets are dropped
  - typically wrong in wireless networks, here we often have packet loss due to transmission errors
  - furthermore, mobility itself can cause packet loss, if e.g. a mobile node roams from one access point (e.g., foreign agent in Mobile IP) to another while there are still packets in transit to the wrong access point and forwarding is not possible

- The performance of an unchanged TCP degrades severely
  - however, TCP cannot be changed fundamentally due to the large base of installation in the fixed network, TCP for mobility has to remain compatible
  - the basic TCP mechanisms keep the whole Internet together
Early Approach: Indirect TCP

- Indirect TCP or I-TCP segments the connection
  - no changes to the TCP protocol for hosts connected to the wired Internet, millions of computers use (variants of) this protocol
  - optimized TCP protocol for mobile hosts
  - splitting of the TCP connection at, e.g., the foreign agent into 2 TCP connections, no real end-to-end connection any longer
  - hosts in the fixed part of the net do not notice the characteristics of the wireless part

I-TCP Socket and State Migration
Indirect TCP

• Advantages
  • no changes in the fixed network necessary, no changes for the hosts (TCP protocol) necessary, all current optimizations to TCP still work
  • transmission errors on the wireless link do not propagate into the fixed network
  • simple to control, mobile TCP is used only for one hop between, e.g., a foreign agent and mobile host
  • therefore, a very fast retransmission of packets is possible, the short delay on the mobile hop is known

• Disadvantages
  • loss of end-to-end semantics, an acknowledgement to a sender does now not any longer mean that a receiver really got a packet, foreign agents might crash
  • higher latency possible due to buffering of data within the foreign agent and forwarding to a new foreign agent

Snooping TCP

• “Transparent” extension of TCP within the foreign agent
  • buffering of packets sent to the mobile host
  • lost packets on the wireless link (both directions!) will be retransmitted immediately by the mobile host or foreign agent, respectively (so called “local” retransmission)
  • the foreign agent therefore “snoops” the packet flow and recognizes acknowledgements in both directions, it also filters ACKs
  • changes of TCP only within the foreign agent
Snooping TCP

- Data transfer to the mobile host
  - FA buffers data until it receives ACK of the MH, FA detects packet loss via duplicated ACKs or time-out
  - fast retransmission possible, transparent for the fixed network
- Data transfer from the mobile host
  - FA detects packet loss on the wireless link via sequence numbers, FA answers directly with a NACK to the MH
  - MH can now retransmit data with only a very short delay
- Advantages
  - end-to-end semantics preserves, no changes to correspondent host (and few to mobile host)
  - no state handover (time-out and retransmission)
  - handover: next FA may not use this approach
- Problems
  - snooping TCP does not isolate the wireless link as good as I-TCP
  - snooping might be useless depending on encryption schemes

Mobile TCP

- Special handling of lengthy and/or frequent disconnections
- M-TCP splits as I-TCP does
  - unmodified TCP fixed network to supervisory host (SH)
  - optimized TCP SH to MH
- Supervisory host
  - no caching, no retransmission
  - monitors all packets, if disconnection detected
    - set sender window size to 0
    - sender automatically goes into persistent mode
  - old or new SH reopen the window
- Advantages
  - maintains semantics, supports disconnection, no buffer forwarding, no changes to sender’s TCP
- Disadvantages
  - loss on wireless link propagated into fixed network
  - adapted TCP on wireless link
Fast Retransmit/Fast Recovery

- Change of foreign agent often results in packet loss
  - TCP reacts with slow-start although there is no congestion
- Forced fast retransmit
  - as soon as the mobile host has registered with a new foreign agent, the MH sends duplicated acknowledgements on purpose
  - this forces the fast retransmit mode at the communication partners
  - additionally, the TCP on the MH is forced to continue sending with the actual window size and not to go into slow-start after registration
- Advantage
  - simple changes result in significant higher performance
- Disadvantage
  - focus on handover losses (not wireless link errors)

Transmission/Time-out Freezing

- Mobile hosts can be disconnected for a longer time
  - no packet exchange possible, e.g., in a tunnel, disconnection due to overloaded cells with higher priority traffic
  - TCP disconnects after time-out completely
- TCP freezing
  - MAC layer is often able to detect interruption in advance
  - MAC can inform TCP layer of upcoming loss of connection
  - TCP stops sending, but does not assume a congested link
  - MAC layer signals again if reconnected
- Advantage
  - scheme is independent of data
- Disadvantage
  - TCP on mobile host has to be changed, mechanism depends on MAC layer
Selective Retransmission

- TCP acknowledgements are often cumulative
  - ACK $n$ acknowledges correct and in-sequence receipt of packets up to $n$
  - if single packets are missing quite often a whole packet sequence beginning at the gap has to be retransmitted (go-back-$n$), thus wasting bandwidth
- Selective retransmission as one solution
  - RFC2018 allows for acknowledgements of single packets, not only acknowledgements of in-sequence packet streams without gaps
  - sender can now retransmit only the missing packets
- Advantage
  - much higher efficiency
- Disadvantage
  - more complex software in a receiver, more buffer needed at the receiver

Transaction Oriented TCP (T/TCP)

- TCP phases
  - connection setup, data transmission, connection release
  - using 3-way-handshake needs 3 packets for setup and release, respectively
  - thus, even short messages need a minimum of 7 packets!
- Transaction oriented TCP
  - RFC1644, T/TCP, describes a TCP version to avoid this overhead
  - connection setup, data transfer and connection release can be combined
  - thus, only 2 or 3 packets are needed
- Advantage
  - efficiency
- Disadvantage
  - requires changed TCP
  - mobility not longer transparent
Comparison of Different Approaches for a “Mobile” TCP

<table>
<thead>
<tr>
<th>Approach</th>
<th>Mechanism</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indirect TCP</td>
<td>splits TCP connection into two connections</td>
<td>isolation of wireless link, simple</td>
<td>loss of TCP semantics, higher latency at handover</td>
</tr>
<tr>
<td>Snooping TCP</td>
<td>“snoops” data and acknowledgements, local retransmission</td>
<td>transparent for end-to-end connection, MAC integration possible</td>
<td>problematic with encryption, bad isolation of wireless link</td>
</tr>
<tr>
<td>M-TCP</td>
<td>splits TCP connection, chokes sender via window size</td>
<td>maintains end-to-end semantics, handles long term and frequent disconnections</td>
<td>Bad isolation of wireless link, processing overhead due to bandwidth management</td>
</tr>
<tr>
<td>Fast retransmit/ fast recovery</td>
<td>avoids slow-start after roaming</td>
<td>simple and efficient</td>
<td>mixed layers, not transparent</td>
</tr>
<tr>
<td>Transmission/ time-out freezing</td>
<td>freezes TCP state at disconnect, resumes after reconnection</td>
<td>independent of content or encryption, works for longer interrupts</td>
<td>changes in TCP required, MAC dependant</td>
</tr>
<tr>
<td>Selective retransmission</td>
<td>retransmit only lost data</td>
<td>very efficient</td>
<td>slightly more complex receiver software, more buffer needed</td>
</tr>
<tr>
<td>Transaction oriented TCP</td>
<td>combine connection setup/release and data transmission</td>
<td>Efficient for certain applications</td>
<td>changes in TCP required, not transparent</td>
</tr>
</tbody>
</table>