Mobility
(and philosophical questions about names and identity)

David Andersen
CMU CS 15-744

The problem

• How to “support” mobile users
• What do we mean by support?
• Make it easy and convenient to effectively use the network while moving from location to location
The Solution Space

• Where can we address this problem?
  – Physical layer? (sure; very limited)
  – Link layer
  – Transport layer
  – “Something higher” (often called session)
  – Application layer

The questions

• What components are affected?
  – E.g., what needs to explicitly support mobility?
  – Is it incrementally deployable?
• What timescales does it support?
• What geographic/logical bounds does it place on mobility?
• What overhead does it impose?
• How does it affect or interact with other aspects of the architecture?
• How does it scale?
Who are we supporting?

• What kinds of mobility scenarios should we support?
  – Talking on a VoIP phone while walking down the street?
  – Navigating with a laptop in a car?
  – Using a laptop in an airplane?
  – Taking laptop from home to work?
  – Walking around lab or campus?
  – Something we haven’t thought of yet??

Try #1: No Network Support
(Applications are on their own)

• Let them disconnect and reconnect when they arrive at a new location.
  – Network support needed: None / DHCP
  – Your SSH sessions die. 😞 Your streaming media probably gets interrupted.
  – Some applications have already worked around this:
    • Your Web browser doesn’t care
    • Your IMAP mail reader probably doesn’t care
Dealing with disconnection

• Possible to code many applications to deal with disconnection
  – It’s all about trying to resume and managing state (we’ll come back to this)
  – But should the burden be placed on every application developer?

So – Application?

• What components are affected?
  – Any application that wants to work
• What timescales does it support?
  – End-to-end application communication. Seconds?
• What geographic/logical bounds does it place on mobility?
  – None
• What overhead does it impose?
  – Lots of programmer overhead
• How does it affect or interact with other aspects of the architecture?
  – Nothing’s changed
Try #2: Link-layer mobility

• Have the link layer mask mobility
  – E.g., the campus 802.11 wireless. You can move anywhere and keep the same MAC and IP address
• Completely transparent. No OS/App support needed. Brilliant!
• Fast & Local: Only switches near moving client must be updated.
• But – only local! Can’t move out of your subnet.

So – Link?

• What components are affected?
  – The local switching infrastructure
• What timescales does it support?
  – Pretty durned fast
• What geographic/logical bounds does it place on mobility?
  – Can only move within local subnet
• What overhead does it impose?
  – Little
• How does it affect or interact with other aspects of the architecture?
  – Could encourage ideas like making all of CMU a single broadcast domain. Oops, too late. 😊
IP Layer Mobility

• Allow hosts to take their “home” IP address with them wherever they go.

• Advantages:
  – Potentially global mobility scope (not limited to subnet like link layer)
  – Transparent to applications and layers above IP

• How can we do it?
  – (Many ways, each with own costs)

Brute Force: IP routing

• If node leaves home, send out (global?) routing announcement pointing to new location
  – In theory, “just works”
  – Example: Boeing’s “Connexion” announced a /24 into BGP for every supported airplane and moved the announcement to the gateway the plane was closest to
  – Why? Latency concerns over really long flights (start in SF, end in London)
  – Already have high latency from using satellites. Ow.
Brute force 2

- May be feasible for Boeing
- But wouldn’t scale for single IP addresses
  - Every AS in world would have routing entry for every mobile user in the world? Ouch!
- Problem: Having the whole world maintain state for every user
- Alternative: Keep state local, by…

Mobile IP (& others):

- Same as other problems in Computer Science
  - Add a level of indirection
- Keep some part of the network informed about current location
  - Need technique to route packets through this location (interception)
- Need to forward packets from this location to mobile host (delivery)
Interception

• Somewhere along normal forwarding path
  – At source
  – Any router along path
  – Router to home network
  – **Machine on home network** (masquerading as mobile host)

Delivery

• Get packet to mobile’s current location
• **Tunnels**
  – Tunnel endpoint = current location
  – Tunnel contents = original packets
• Source routing?
  – Loose source route through mobile current location (not widely supported)
• Network address translation (NAT)
  – What about packets from the mobile host?
Mobile IP (RFC 2290)

- **Interception**
  - Typically home agent – hosts on home network

- **Delivery**
  - Typically IP-in-IP tunneling
  - Endpoint – either temporary mobile address or foreign agent

- **Terminology**
  - Mobile host (MH), correspondent host (CH), home agent (HA), foreign agent (FA)
  - Care-of-address, home address

---

**Mobile IP (MH at Home)**

- Correspondent Host (CH)
- Mobile Host (MH)
- Home
- Visiting Location
- Internet
- Packet
Mobile IP (MH Moving)

Correspondent Host (CH)  
Home Agent (HA)  
Mobile Host (MH)

Packet

Internet

Home

Visiting Location

Mobile IP (MH Away – Foreign Agent)

Correspondent Host (CH)  
Mobile Host (MH)

Packet

Internet

Home

Visiting Location

Home Agent (HA)  
Foreign Agent (FA)
Mobile IP (MH Away - Collocated)

Other Mobile IP Issues

- Route optimality
  - Triangle routing
  - Can be improved with route optimization
    - Unsolicited binding cache update to sender
- Authentication
  - Registration messages
  - Binding cache updates
- Must send updates across network
  - Handoffs can be slow
- Problems with basic solution
  - Reverse path check for security
  - Do we really need it?
TCP Migrate

- Transport-layer solution
- Idea: No IP support; just have transport layer dynamically re-bind endpoints

The **Migrate** Approach

- Locate hosts through existing DNS
  - Secure, dynamic DNS is currently deployed and widely available (*RFC 2137*)
  - Maintains standard IP addressing model
    - IP address are topological addresses, not Ids
    - *Fundamental* to Internet scaling properties
- Ensure seamless connectivity through connection migration
  - Notify only the current set of correspondent hosts
  - Follows from the end-to-end argument

*Slide Credit: Alex Snoeren*
Migrate Architecture

Slide Credit: Alex Snoeren

Migrate

• Advantages:
  – (Mostly) transparent to applications
    • Unless they know their IP address and use it, e.g., peer-to-peer apps.
  – Keeps state and modifications entirely at endpoints
  – No triangle routing! All communication is direct

• But:
  – Requires TCP support / only works for TCP
    • Not true in general: “Host ID Protocol” – HIP – can work with both, but requires more invasive IP stack changes
  – Slower timescales than link-layer migration (several RTTs)
Complexities of e2e mobility

• Simultaneous movement
  – If only one host moves, easy
  – If both move, must be able to reconnect
  – Snoeren approach uses DNS with dynamic DNS updates – re-point your old name to your new IP when you move

• Security
  – How to prevent connection hijacking?

Mobility & Security

• Migrate principle: Equivalent security to TCP
  – TCP connections hard to hijack remotely if you can’t sniff because you must guess a 32-bit sequence # space. (mostly; we’ll talk about this more later)
  – Migrate approach: Establish a pretty secure session key on connection establishment
    • Resists snooping but not man-in-the-middle
    • But neither does normal TCP!

• Other options: HIP uses cryptographic host identification
  – Better idea
  – Less incrementally deployable
Names & Addresses & Bears, Oh My!

• Mobility raises good question:
  – *What is the identity of a host?*
    • MAC address? IP address? DNS name? Something else?

• Consider:
  – Hosts can have multiple MAC & IP addresses
  – IP address is a *topological* identifier – it points to a place in the local IP space and is awkward to move, as we’ve seen
  – DNS names? Maybe, but the binding between DNS/IP/hosts isn’t very strict

Host Identity

• Considerable recent work: Give each host a unique identity
  – Simplifies mobility
  – Also simplifies multi-homing! (Many related issues)
  – Me? I think it’s a great idea. Will it ever take off? 😊
What mobility do we need?

- Consider our scenarios and our techniques – what do we really need?
- Link layer mobility can deal with small-scale motion
- E2E mobility does a good job on “big”, less frequent movement
  - But if only a few apps matter, so does re-coding those apps to deal
  - Requires bilateral deployment! Boooo.
- Mobile IP (or VPNs, which is basically what mobile IP is) can be unilaterally deployed, but has triangle routing problems
  - But require more infrastructure
- Do most people care enough? Or would we have entire new classes of applications if mobility was easier?