



MSOCKS: An Architecture for Transport Layer Mobility

CSE291K Mobile Networking

Presenter: Erik Vandekieft



The story so far...

3rd major approach we've seen

- Mobile IP (and 4x4 extensions)
- I-TCP
- Now, MSOCKS

Similarities between the approaches

- ◆ All make use of proxies
- ◆ All seek to support an unbroken TCP session through some sort of “move”

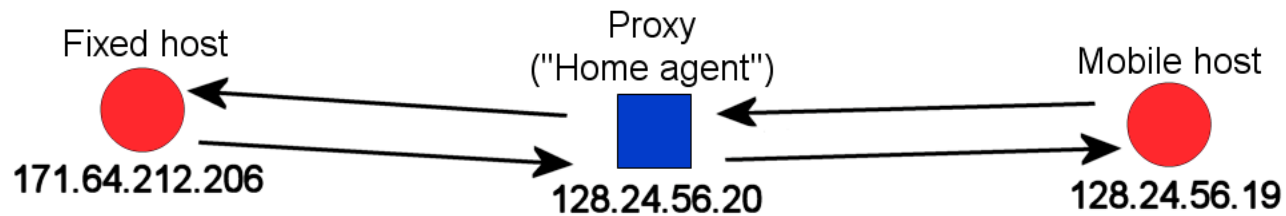
Differences between the approaches

Definition of "move":

- ◆ Mobile IP: switch between networks
- ◆ I-TCP: switch between proxies (i.e. MSRs, i.e. base stations) on same network
- ◆ MSOCKS: switch between network interfaces to proxy (switching between networks handled in same way)

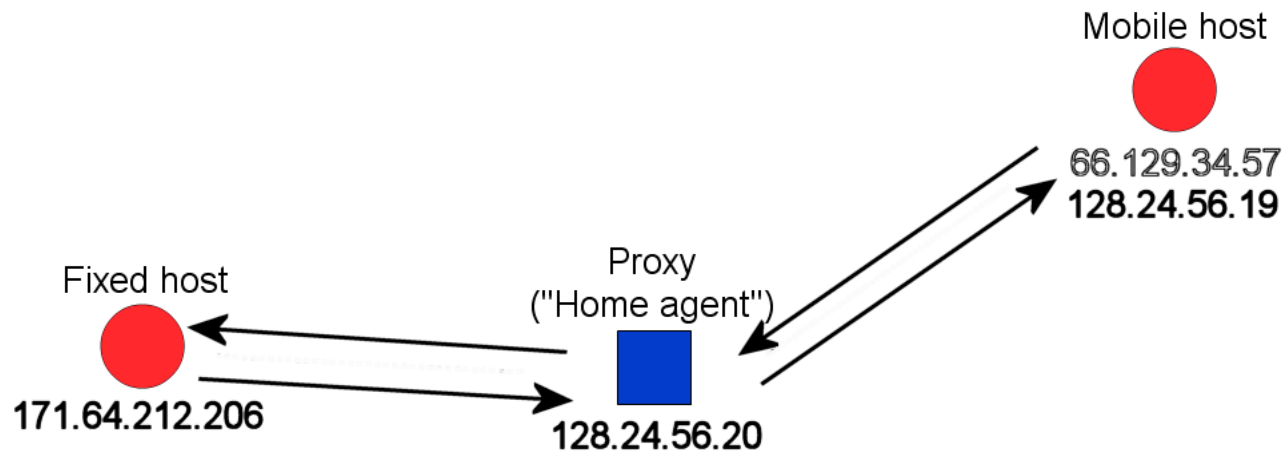
Mobile IP

- ◆ Network-level solution
- ◆ Packets are forwarded, TCP is supported indirectly
- ◆ "Movement" = switching between networks



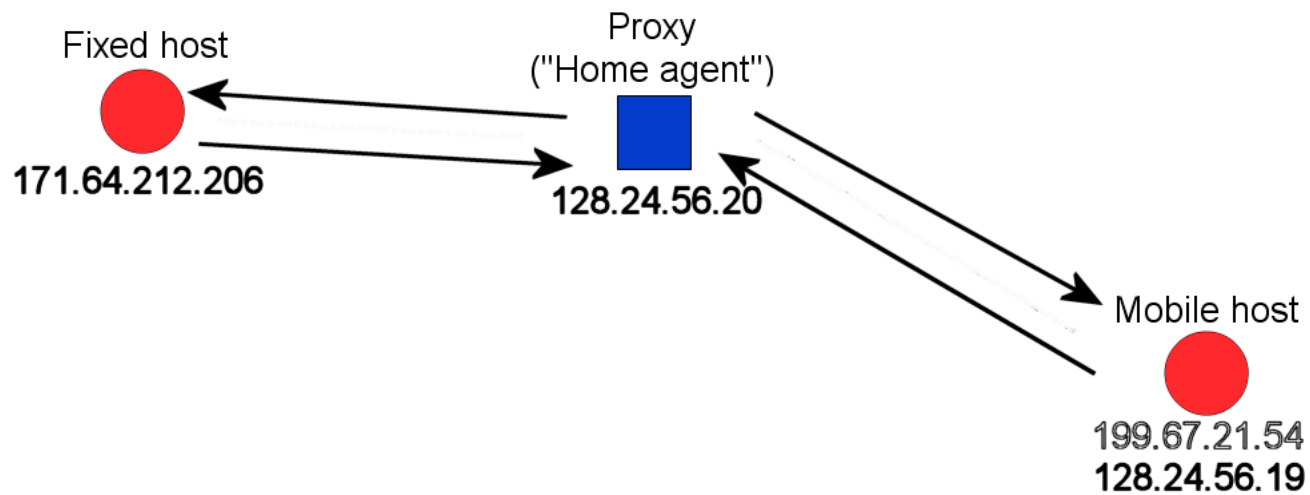
Mobile IP

- ◆ Can move anywhere
- ◆ Just need to tell home agent where you are



Mobile IP

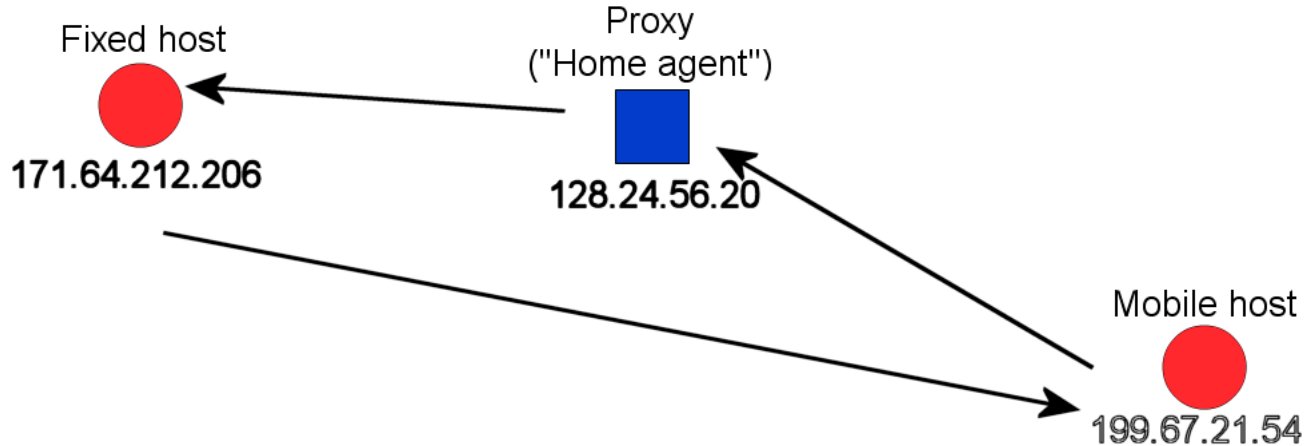
TCP connection remains unbroken because a fixed host can continue sending to your "home" IP address, and your home agent is in charge of making sure the packets get forwarded to your current location



Mobile IP with 4x4

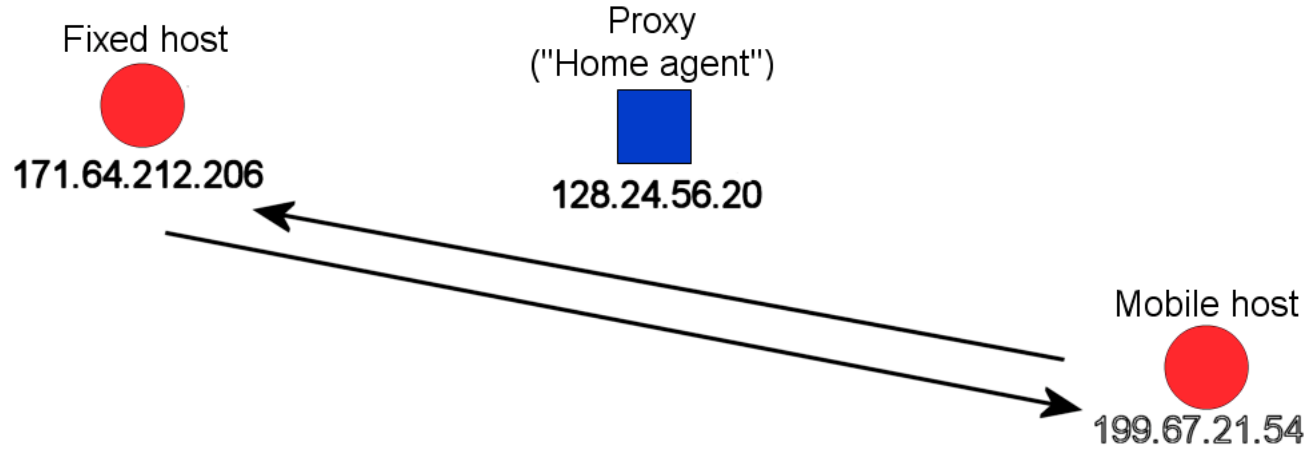
Just an extension of the ideas in the Mobile IP paper

If we have a mobility-aware fixed host, it can talk to mobile host directly



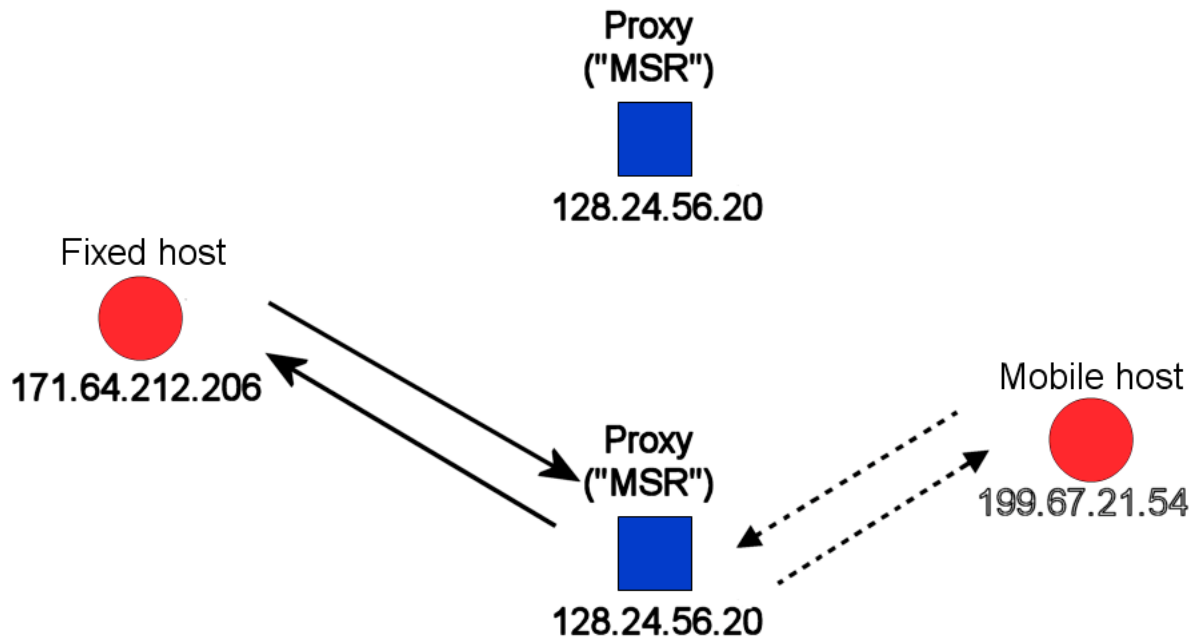
Mobile IP with 4x4

If we have permissive routers, mobile host can talk to fixed host directly



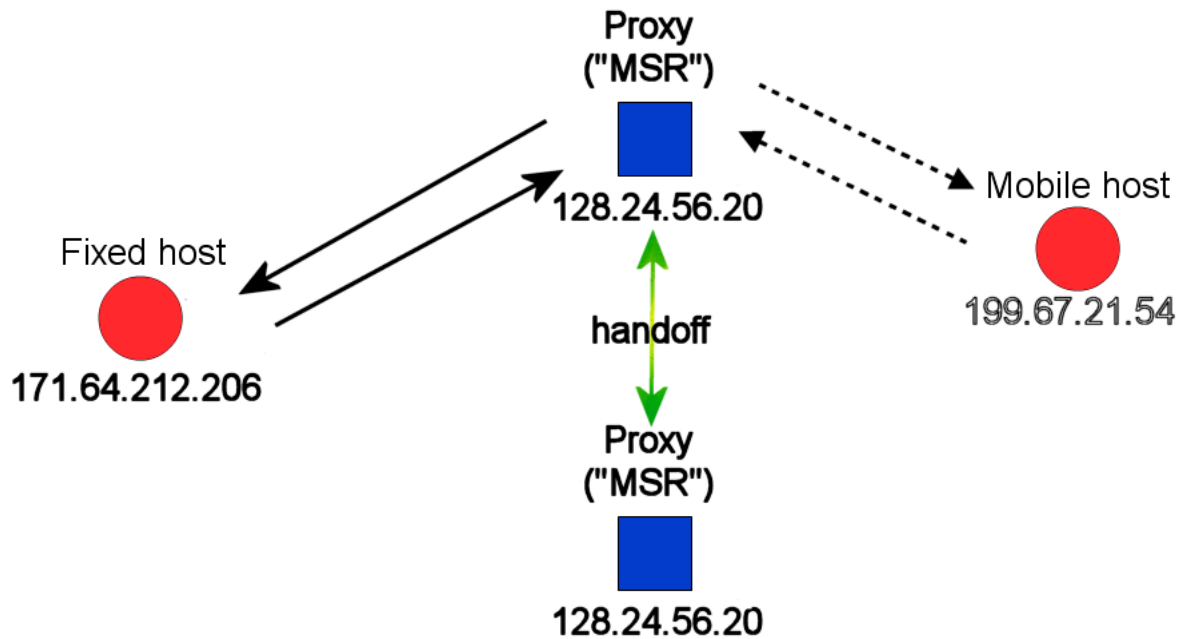
I-TCP

- Transport-level Solution
- Concerned with optimizing for special properties of wireless links
- "Movement" = switching between MSRs



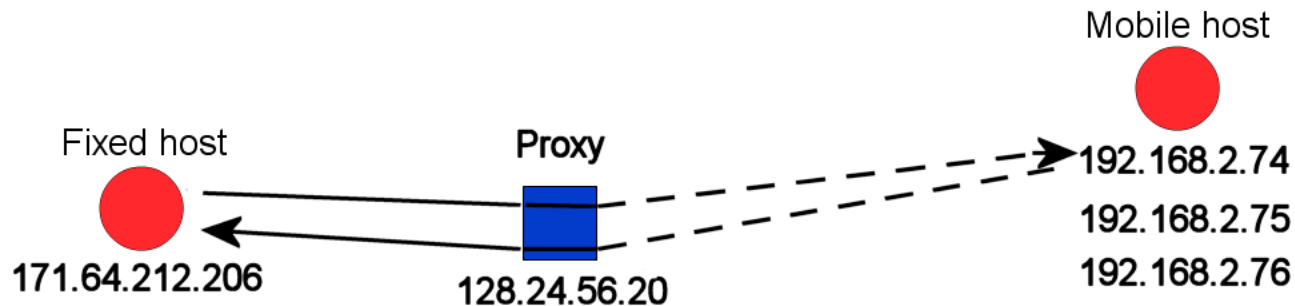
I-TCP

TCP connection remains unbroken because fixed host always sends to the same IP– the MSRs coordinate amongst themselves to decide who actually handles the communication. BUT, end-to-end TCP semantics are broken!



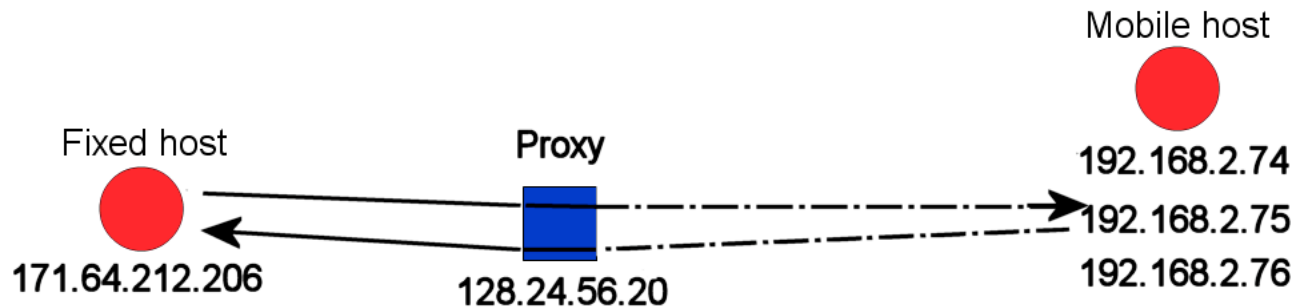
MSOCKS

- ◆ Transport-level Solution
- ◆ “Movement” = switching between interfaces to proxy (or switching networks; the point is that your IP address changes)



MSOCKS

- ◆ **TCP connection remains unbroken** because fixed hosts always just talk to the proxy over what appears to them as one regular TCP connection; proxy is in charge of sending the packets on the interface that the mobile host requested they be sent on.



MSOCKS achievements

Two main achievements:

- Mobile node has freedom to send *and receive* from network interface of its choice (only sending would be possible without a proxy)
- Preservation of TCP's end-to-end reliability and correctness semantics

MSOCKS implementation

- ◆ Basic idea: proxy will *modify TCP packets* as they pass through it to create the illusion of a single, direct TCP connection between the two parties.
- ◆ There are actually two TCP connections (fixed host <-> proxy, mobile host <-> proxy) which the proxy “splices” together, effectively creating a direct TCP connection from fixed host <-> mobile host

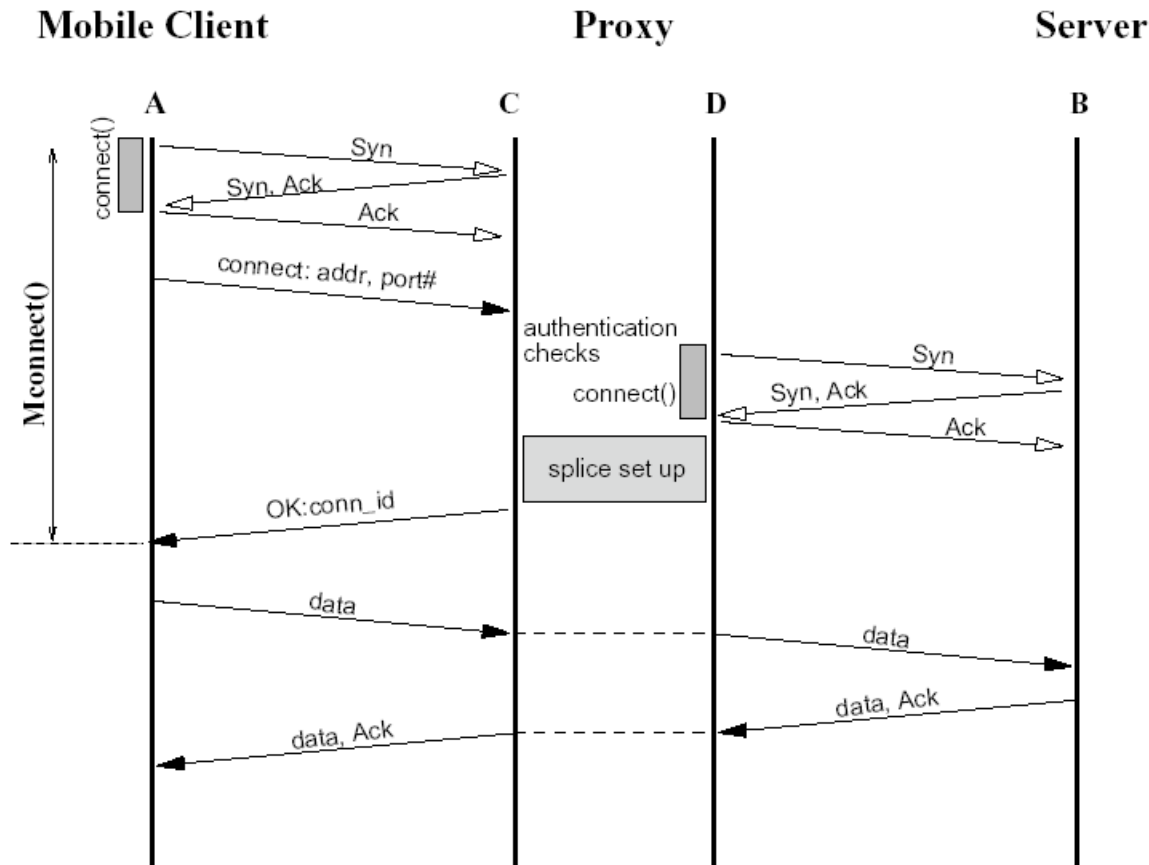
Implementation elements

- ◆ Mobile host machine must have MSOCKS library and MSOCKS-aware applications to utilize it
- ◆ Must have MSOCKS proxy machine (with TCP splice capability)
- ◆ MSOCKS protocol to allow mobile host and proxy to coordinate
- ◆ No changes whatsoever to fixed host

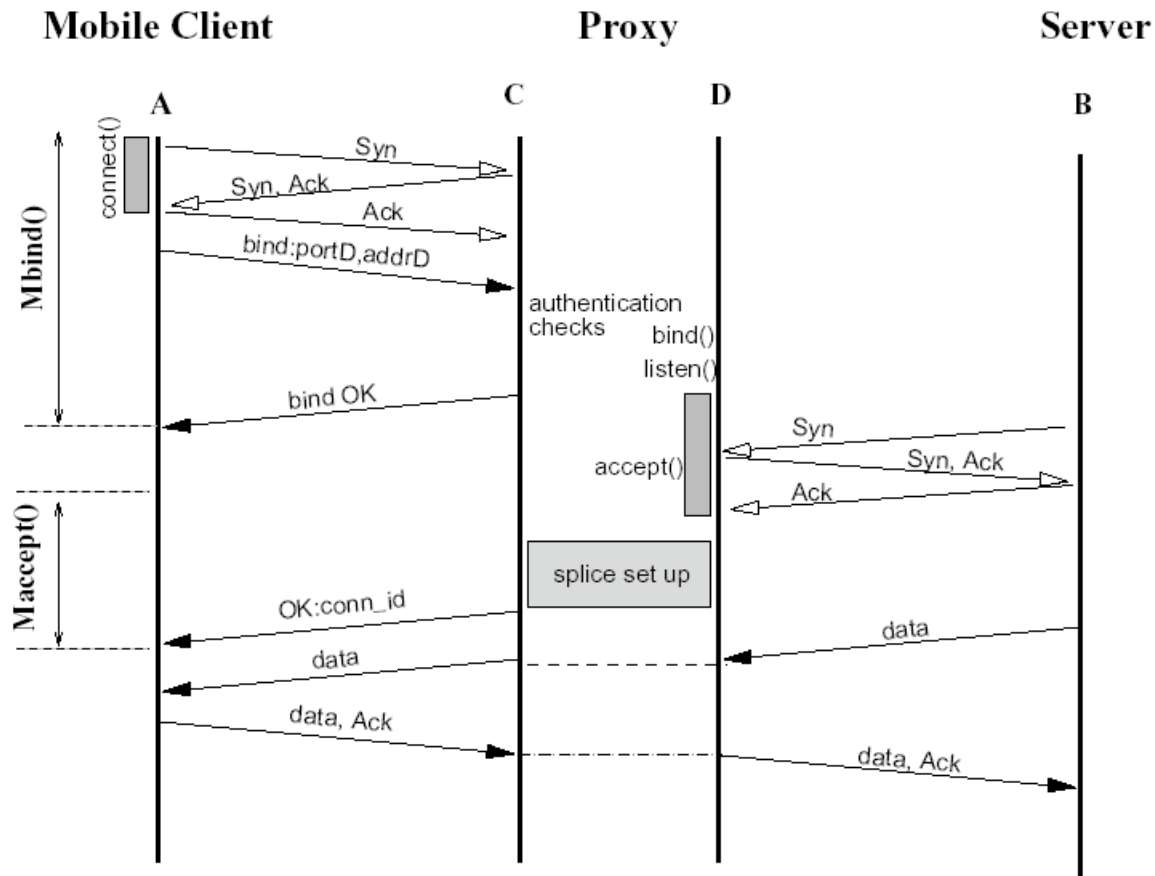
MSOCKS protocol

- ◆ Built on top of SOCKS, a protocol for firewall traversal
- ◆ Key difference is the ability to “reconnect” to an old session. This is how we switch between network interfaces; just break one and reconnect using another

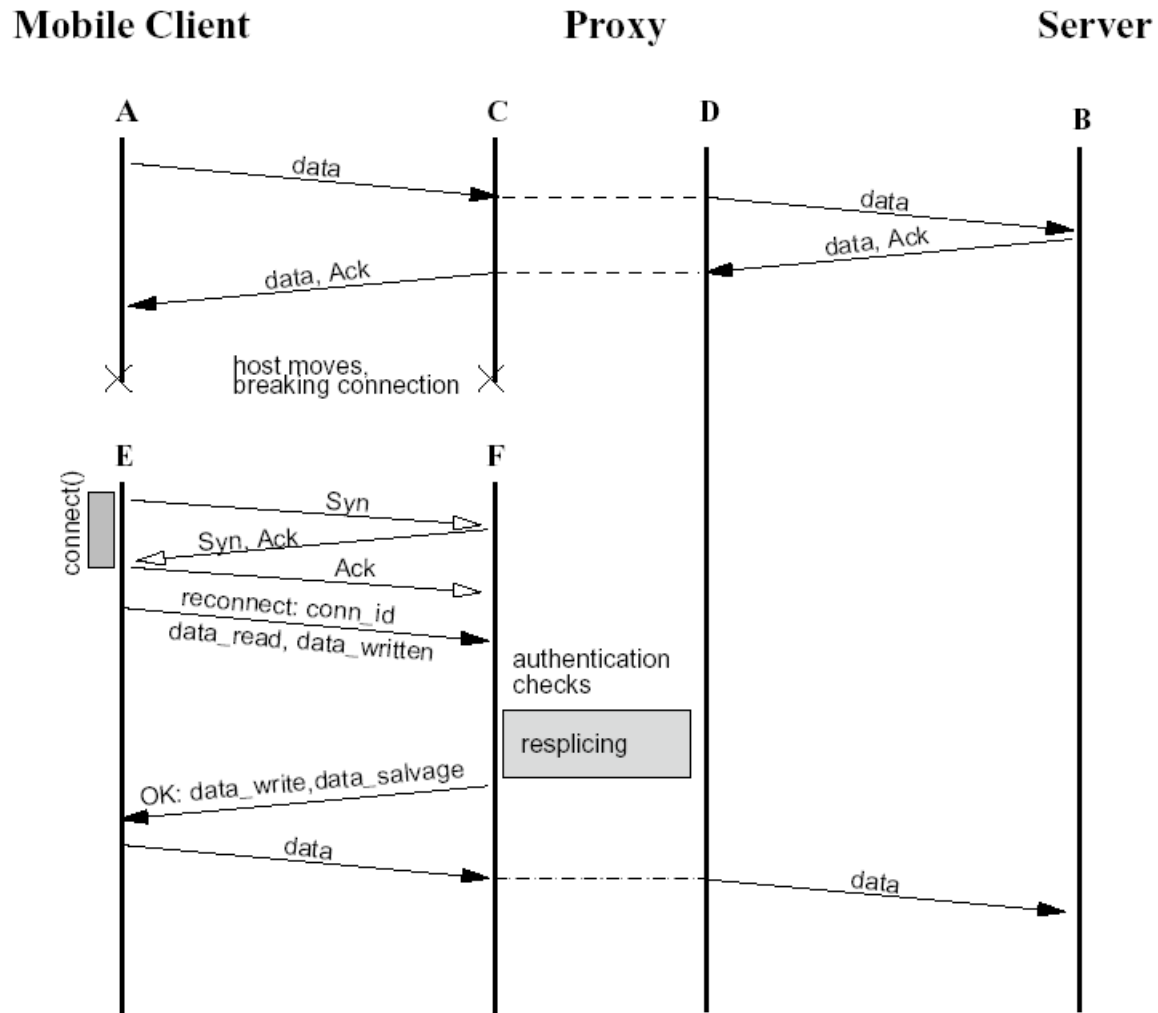
Connecting to an outside fixed host



Binding a port and accepting a connection



Reconnection



TCP splicing

- ◆ Maintain mapping between sequence number spaces of both connections
- ◆ As TCP packets pass through, alter sequence and ACK numbers (and a few other things) according to the mapping
- ◆ Must examine packets for indications that end systems are shutting down connection (FIN and then ACK, or RST)

TCP splicing (cont.)

Altering the packets:

◆ Alter IP header

- Change source/destination
- Update checksum

◆ Alter TCP header

- Change source/destination port
- Map sequence number from incoming space to outgoing space
- Map ACK number from incoming space to outgoing space
- Update checksum

Performance

- ◆ Their numbers show that this approach is indeed scalable
- ◆ Packet modification/forwarding operation is so cheap, the primary limitation on how many mobile nodes a proxy can handle is the link bandwidth into and out of the proxy

SUMMARY OF FORWARDING LATENCIES CREATED BY TCP SPLICE AND IP ROUTING

	mean (msecs)	median (msecs)
IP forwarding	0.4038	0.0960
TCP Splice forwarding	0.4444	0.1120

Goal comparison

- ◆ Mobile IP: supporting arbitrary movement (global mobility)
- ◆ I-TCP: optimize wireless link
- ◆ MSOCKS:
 - 1.) Allow switching between network interfaces
 - 2.) Support correct end-to-end TCP reliability semantics

Response to other approaches

- As to issue of global mobility: “we feel many mobile computer users, such as office workers, will not want to keep their connections up and valid during long moves... Given this environment, we focused on a design that allows individual data streams to be rerouted, rather than rerouting packets.”
- As to issue of wireless performance: “We see the issues of transport layer mobility and TCP’s wireless performance as largely orthogonal... Systems like Snoop TCP could be used underneath TCP Splice to improve wireless performance.”
(today: this is indeed solved at link layer: 802.11b)