ATTRIBUTION

• These slides are released under an Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0) Creative Commons license

• These slides incorporate material from:
  • Christo Wilson, NEU (used with permission)
  • Yashar Ganjali, Toronto (used with permission)
What are capes?

• **Only** source of feedback to UCSD about Professor & TA teaching

• **Anonymous**

• **Optional**

• **Extremely important**
  - Determines whether faculty get promoted, get tenure, keep their jobs
  - Determines whether TAs become TAs in the future

• **Please make your voice heard!**
  - Usually only students who *love* or *hate* the class fill them out
  - We appreciate the few minutes it takes to make your opinions/voice heard

*Thanks!*
Abstract View of the Internet

- A bunch of servers/virtual machines connected by point-to-point physical links
- Point-to-point links between routers are physically as direct as possible
Reality Check

- Fibers and wires limited by physical constraints
  - You can’t just dig up the ground everywhere
  - Most fiber laid along railroad tracks
- Physical fiber topology often far from ideal
- IP Internet is overlaid on top of the physical fiber topology
  - IP Internet topology is only logical
- Key concept: IP Internet is an overlay network
National Lambda Rail Project

IP Logical Link

Physical Circuit
Layering hides low level details from higher layers

- IP is a logical, point-to-point overlay
- ATM/SONET circuits on fibers
Overlays

- Overlay is a general concept
  - Networks are just about routing messages between named entities
- IP Internet overlays on top of physical topology
  - We assume that IP and IP addresses are the only names…
- Why stop there?
  - Overlay another network on top of IP
Overlay Networks
Overlay Networks

Focus at the application level
Overlay Networks

- A logical network built on top of a physical network
  - Overlay links are tunnels through the underlying network
- Many logical networks may coexist at once
  - Over the same underlying network
  - And providing its own particular service
- Nodes are often end hosts
  - Acting as intermediate nodes that forward traffic
  - Providing a service, such as access to files
- Who controls the nodes providing service?
  - The party providing the service (e.g., Akamai)
  - Distributed collection of end users (e.g., peer-to-peer)
Routing Overlays

- Alternative routing strategies
  - No application-level processing at the overlay nodes
  - Packet-delivery service with new routing strategies
- Incremental enhancements to IP
  - IPv6
  - Multicast
  - Mobility
  - Security
- Revisiting where a function belongs
  - End-system multicast: multicast distribution by end hosts
- Customized path selection
  - Resilient Overlay Networks: robust packet delivery
IP Tunneling

- IP tunnel is a virtual point-to-point link
  - Illusion of a direct link between two separated nodes

Logical view:

Physical view:

- Encapsulation of the packet inside an IP datagram
  - Node B sends a packet to node E
  - ... containing another packet as the payload
6Bone: Deploying IPv6 over IP4

Logical view:

Physical view:

Flow: X
Src: A
Dest: F
data

Flow: X
Src: A
Dest: F
data

A-to-B:
IPv6

B-to-C:
IPv6 inside IPv4

D-to-E:
IPv6 inside IPv4

E-to-F:
IPv6

Flow: X
Src: A
Dest: F
data

Flow: X
Src: A
Dest: F
data

Flow: X
Src: A
Dest: F
data
Communicating With Mobile Users

- A mobile user changes locations frequently
  - So, the IP address of the machine changes often
- The user wants applications to continue running
  - So, the change in IP address needs to be hidden
- Solution: fixed gateway forwards packets
  - Gateway has a fixed IP address
  - ... and keeps track of the mobile’s address changes
MBone: IP Multicast

- Multicast
  - Delivering the same data to many receivers
  - Avoiding sending the same data many times

- IP multicast
  - Special addressing, forwarding, and routing schemes
  - Not widely deployed, so MBone tunneled between nodes
End-System Multicast

- IP multicast still is not widely deployed
  - Technical and business challenges
  - Should multicast be a network-layer service?
- Multicast tree of end hosts
  - Allow end hosts to form their own multicast tree
  - Hosts receiving the data help forward to others
Unicast Streaming Video

This does not scale
IP Multicast Streaming Video

- Much better scalability
- IP multicast not deployed in reality
  - Good luck trying to make it work on the Internet
  - People have been trying for 20 years

Source only sends one stream

IP routers forward to multiple destinations
End System Multicast Overlay

How to build an efficient tree?

- Enlist the help of end-hosts to distribute stream
- Scalable
- Overlay implemented in the application layer
  - No IP-level support necessary

This does not scale
RON: Resilient Overlay Networks

Premise: by building application overlay network, can increase performance and reliability of routing
RON Can Outperform IP Routing

- IP routing does not adapt to congestion
  - But RON can reroute when the direct path is congested
- IP routing is sometimes slow to converge
  - But RON can quickly direct traffic through intermediary
- IP routing depends on AS routing policies
  - But RON may pick paths that circumvent policies

Then again, RON has its own overheads
- Packets go in and out at intermediate nodes
  - Performance degradation, load on hosts, and financial cost
- Probing overhead to monitor the virtual links
  - Limits RON to deployments with a small number of nodes
• Underlying network
• Internet connectivity (IP Routing)
• Potential overlay connectivity
  • SF as root
- Determine edge weights
- E.g., bandwidth, latency
• Build overlay connectivity
• An application-layer distribution tree
• “Tree” constructed using application-layer sockets

• Data flows along tree, not underlying network

• Why?
  • Can improve reliability
    • If link from B->G fails, can take few minutes for Internet to recover (meanwhile app can respond in milliseconds to create new path)

• Disseminate data in a scalable way

• Avoid censorship
**KEY CONCEPTS**

- **Link stress**
  - How often a packet transits a given link

- **Relative delay penalty (aka “Stretch”)**
  - Ratio of delay in overlay vs. underlying network
APP-LAYER OVERLAY EXAMPLE

- Network cost A -> F
  - 1
- Overlay cost A -> F
  - 4 + 2 + 2 = 8
- Relative delay penalty A -> F
  - 8/1
Secure Communication Over Insecure Links

- Encrypt packets at entry and decrypt at exit
- Eavesdropper cannot snoop the data
- ... or determine the real source and destination
Tor Project

- An overlay to enhance anonymity and privacy
  - Volunteer operated servers (?)

How Tor Works

- Obtain a list of Tor nodes from a directory
- Pick a random path to destination server
- Select a different path for other servers
WHAT IS CRYPTOGRAPHY?

• From Greek, meaning “secret writing”
• Confidentiality: encrypt data to hide content
• Include “signature” or “message authentication code”
  • Integrity: Message has not been modified
  • Authentication: Identify source of message

Modern encryption:
• Algorithm public, key secret and provides security
• Symmetric (shared secret) or asymmetric (public-private key)
SYMOMETRIC (SECRET KEY) CRYPTO

• Sender and recipient share common key
  • Main challenge: How to distribute the key?

• Provides dual use:
  • Confidentiality (encryption)
  • Message authentication + integrity (MAC)

• 1000x more computationally efficient than asymmetric
SYMMETRIC CIPHER MODEL

Hi Bob Alice

Encrypt

C = E(M, K)

Ciphertext

Symmetric key
(shared secret, known to A & B)

Decrypt

M = D(C, K)

Hi Bob Alice

C = Cipher text
M = Message (plaintext)
K = Secret Key
E = Encryption function
D = Decryption function
??!!
PUBLIC-KEY CRYPTOGRAPHY

- Each party has (public key, private key)

- Alice’s public key PK
  - Known by anybody
  - Bob uses PK to encrypt messages to Alice
  - Bob uses PK to verify signatures from Alice

- Alice’s private/secret key: sk
  - Known only by Alice
  - Alice uses sk to decrypt ciphertexts sent to her
  - Alice uses sk to generate new signatures on messages
(PK, sk) = generateKey(keysize)

Encryption API
- ciphertext = encrypt (message, PK)
- message = decrypt (ciphertext, sk)

Digital signatures API
- Signature = sign (message, sk)
- isValid = verify (signature, message, PK)
(SIMPLE) RSA ALGORITHM

- **Generating a key:**
  - Generate composite $n = p \times q$, where $p$ and $q$ are secret primes
  - Pick public exponent $e$
  - Solve for secret exponent $d$ in $d \cdot e \equiv 1 \pmod{(p-1)(q-1)}$
  - Public key = $(e, n)$, private key = $d$

- **Encrypting message $m$:** $c = m^e \mod n$
- **Decrypting ciphertext $c$:** $m = c^d \mod n$

- **Security** due to cost of factoring large numbers
  - Finding $(p, q)$ given $n$ takes $O(e^{\log n \log \log n})$ operations
  - $n$ chosen to be 2048 or 4096 bits long
IPSec

• Support for IPsec, as the architecture is called, is optional in IPv4 but mandatory in IPv6.
• IPsec is really a framework (as opposed to a single protocol or system) for providing all the security services discussed throughout this chapter.
• IPsec provides three degrees of freedom.
  – First, it is highly modular, allowing users (or more likely, system administrators) to select from a variety of cryptographic algorithms and specialized security protocols.
  – Second, IPsec allows users to select from a large menu of security properties, including access control, integrity, authentication, originality, and confidentiality.
  – Third, IPsec can be used to protect “narrow” streams (e.g., packets belonging to a particular TCP connection being sent between a pair of hosts) or “wide” streams (e.g., all packets flowing between a pair of routers).
Transport vs. tunnel mode

• Transport:
  – Host-to-host secure connection
  – Encrypted, authenticated, or both

• Tunnel
  – Host-to-network or network-to-network
  – Entire IP packet tunneled in secure IPSec “envelope” to recovered at destination
Security in IPSec

- **AH**: Authentication header
  - Access control, message integrity, authentication, and antireplay protection

- **ESP**: Encapsulating Security Payload
  - Like AH, but with encryption too

- **SA**: Security association
  - Selection of algorithms, crypto, hashes, etc

- **SPI**: Security Parameters Index (SPI)
  - Per-connection index into SA database

- **ISAKMP**: Internet Security Association and Key Management Protocol
Standard IPv4 Datagram

<table>
<thead>
<tr>
<th>ver</th>
<th>hlen</th>
<th>TOS</th>
<th>pkt len</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- ID
- flgs
- frag offset

**TTL**

proto=TCP -> header checksum

src IP address

dst IP address

IP Options (if present)

TCP header (proto = 6)

TCP payload

32 bits

Covered by header checksum

[http://wwwunixwiznet/techtips/iguidetipsec.html](http://wwwunixwiznet/techtips/iguidetipsec.html)
# IP “next” protocols

<table>
<thead>
<tr>
<th>Protocol code</th>
<th>Protocol Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ICMP — Internet Control Message Protocol</td>
</tr>
<tr>
<td>2</td>
<td>IGMP — Internet Group Management Protocol</td>
</tr>
<tr>
<td>4</td>
<td>IP within IP (a kind of encapsulation)</td>
</tr>
<tr>
<td>6</td>
<td>TCP — Transmission Control Protocol</td>
</tr>
<tr>
<td>17</td>
<td>UDP — User Datagram Protocol</td>
</tr>
<tr>
<td>41</td>
<td>IPv6 — next-generation TCP/IP</td>
</tr>
<tr>
<td>47</td>
<td>GRE — Generic Router Encapsulation (used by PPTP)</td>
</tr>
<tr>
<td>50</td>
<td>IPsec: ESP — Encapsulating Security Payload</td>
</tr>
<tr>
<td>51</td>
<td>IPsec: AH — Authentication Header</td>
</tr>
</tbody>
</table>
IPSec in AH Transport Mode

Original IPv4 Datagram

- ver
- hlen
- TOS
- pkt len
- ID
- flgs
- frag offset
- TTL
- proto-TCP
- header checksum
- src IP address
- dst IP address

TCP header (proto = 6)

TCP payload

Protected by AH Auth Data

New IPv4 Datagram

- ver
- hlen
- TOS
- pkt len + AH size
- ID
- flgs
- frag offset
- TTL
- proto-AH
- header checksum
- src IP address
- dst IP address
- next-TCP
- AH len
- Reserved
- SPI (Security Parameters Index)
- Sequence Number
- Authentication Data (usually MD5 or SHA-1 hash)

TCP header (proto = 6)

TCP payload
**IPSec in AH Tunnel Mode**

**Original IPv4 Datagram**

<table>
<thead>
<tr>
<th>ver</th>
<th>hlen</th>
<th>TOS</th>
<th>pkt len</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flgs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>frag offset</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>proto=TCP</td>
<td>header cksum</td>
<td></td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP header (proto = 6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP payload</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Protected by AH Auth Data**

**New IPv4 Datagram**

<table>
<thead>
<tr>
<th>ver</th>
<th>hlen</th>
<th>TOS</th>
<th>pkt len + AH + IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flgs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>frag offset</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>proto=AH</td>
<td>header cksum</td>
<td></td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>next-IP</td>
<td>AH len</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>SPI (Security Parameters Index)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence Number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authentication Data (usually MD5 or SHA-1 hash)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ver</td>
<td>hlen</td>
<td>TOS</td>
<td>pkt len</td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
<td>-----</td>
<td>---------</td>
</tr>
<tr>
<td>ID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flgs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>frag offset</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>proto=TCP</td>
<td>header cksum</td>
<td></td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP header (proto = 6)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP payload</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
UC San Diego