Last Time

- IPv6
- IPSec
This Time

- IPSec
- Differentiated Services
- MPLS

IPv6 Security

- IPv6 provides built-in mechanisms for security
  - not found in IPv4
- uses extension headers to specify
- some background info first...
IPSEC

- works at IP layer
- but is **connection-oriented**
- each connection is a **Security Association** (SA)
  - *simplex* connection
  - need 2 SAs for two-way communication
- 2 modes:
  1. transport: ‘seen’ by nodes between sender and dest
  2. tunnel: ‘seen’ only by sender and dest
- encryption “always on”
  - but IPsec doesn’t dictate particular algorithms
  - can use “null” algorithm for no encryption (RFC 2410)

Security Associations

- ‘secure’ connection from sender to receiver
- parameters negotiated at set-up
  - e.g., key
- database entry at receiver has parms for each SA
- SAs uniquely identified by:
  1. Security Parameters Index (SPI)
  2. IP destination address
  3. security protocol identifier
SA characteristics

- sequence number: 32-bit counter
- sequence counter o'flow: does counter overflow trigger “auditable” event and lock-out this SA?
- anti-replay window: range of allowed (expected) sequence numbers
- lifetime: time interval or byte-count after which SA is no longer valid
- IPSec protocol mode: transport or tunnel
- **Path MTU**: max size that doesn’t need fragmenting
-Hdr specific parms: see header descriptions

Components of IPsec

3 main components:

1. **AH: Authentication Header**
   - integrity: can't make undetected changes to msg
   - end receiver has ability to authenticate
     - eliminate spoof attacks
     - guard against replay attacks

2. **ESP: Encapsulating Security Payload**
   - confidentiality of msg contents
   - some traffic flow confidentiality

3. key management mechanism
AH Format

Authentication Data

- variable length
- ICV: integrity check value
- message authentication code (MAC):
  - sender & receiver share a secret key
  - sender computes a value from the message
  - encrypts value with secret key
- must be multiple of 32–bit size
Use of AH

ESP Packet Format
AH & ESP Operation

- both operate in
  - transport mode
  - tunnel mode

- IPsec allows one of AH or ESP but not both at a time

Transport Mode

- payload in packet is protected
  - in IPv4, payload immediately follows IP hdr
- ESP encrypts (+ optionally authenticates) IP payload but not header
- AH authenticates IP payload + selected parts of IP header
Tunnel Mode

- protects entire IP packet
- add AH or ESP fields to IP packet then re-wrap in new IP packet
  - can have different src/dst address (e.g., firewall to firewall)
  - “inner” packet is “tunneled” through network
  - NAT issues
- ESP encrypts (+ optionally authenticates) entire inner IP packet
- AH authenticates entire inner packet + selected parts of outer IP header
Private Network

- to protect networked assets, organizations use private networks

  ![Diagram of private network]

- can be real: lease dedicated lines between routers at sites
  - but leased lines expensive

Private Network

- to protect networked assets, organizations use private networks
- can be real
- can be virtual (VPN):
  - run over public network, often Internet itself
  - combine several elements we’ve seen to achieve effect of dedicated private network over public network
Virtual Private Network

use:

- **tunnels**: carry encrypted traffic between
- **firewalls** as single-access points
- use IPSEC’s ESP for tunneling then have single SA for each simplex ‘channel’: confidentiality + integrity

Far Reaching

- how do you reach a host in
  - London?
  - Osaka?
  - Melbourne?
  - Riyadh?
- can do with satellites
Very Long ‘Wire’

- can also do intercontinental via undersea cable
- first undersea cable in 1850
- first all-optical cable in 1996
- can go up to 400 km without repeater
  - current repeaters are pure optical
- cable includes power to run repeaters
  - about 40V per repeater
  - 7,500 km cable needs 10,000 V
- built to run 25 years without repair at depths down to 7 km

Undersea Cable: what’s it cost?

- costs of some undersea cables:

<table>
<thead>
<tr>
<th>Cable</th>
<th>BW</th>
<th>Length km</th>
<th>Cost: Gbps/Km</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAT-9</td>
<td>565 Mbps</td>
<td>9,310</td>
<td>$85,548</td>
</tr>
<tr>
<td>TPC-4</td>
<td>560 Mbps</td>
<td>9,860</td>
<td>$67,553</td>
</tr>
<tr>
<td>CIOS</td>
<td>622 Mbps</td>
<td>261</td>
<td>$61,598</td>
</tr>
<tr>
<td>TPC-5</td>
<td>5 Gps (max 20)</td>
<td>25,000</td>
<td>$8,960</td>
</tr>
<tr>
<td>FLAG</td>
<td>10 Gbps</td>
<td>27,000</td>
<td>$5,600</td>
</tr>
<tr>
<td>Columbus III</td>
<td>10 Gbps</td>
<td>11,000</td>
<td>$2,700</td>
</tr>
</tbody>
</table>

data from: [http://www-dsg.stanford.edu/holbrook/CableCosts.html](http://www-dsg.stanford.edu/holbrook/CableCosts.html)
Example: TPC–5CN

- usually do loop instead of point–to–point, like:

![Map of Undersea Cable](http://www.apricot.net/apricot97/apII/Presentations/KDDSubmarineFiber/sld017.htm)

**Undersea Cable**

- advantages:
  - low latency
  - high reliability
  - high bit rates

- disadvantages:
  - expensive to lay
  - time consuming to repair
    - hard to find breaks, hard to find cable!
Where Packets Go: Refresher

- network layer (IP datagrams) are *forwarded* at routers
  - but haven’t described how
  - start with a host that has a packet to send:

Forwarding Packets
Where Does The Packet Go?

- a **single host** on network needs to know where to send its packets in order to use network
- usually uses a **static routing** table:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>Genmask</th>
<th>Flags</th>
<th>Iface</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.0</td>
<td>0.0.0.0</td>
<td>255.255.255.0</td>
<td>U</td>
<td>eth0</td>
</tr>
<tr>
<td>127.0.0.0</td>
<td>0.0.0.0</td>
<td>255.0.0.0</td>
<td>U</td>
<td>lo</td>
</tr>
<tr>
<td>0.0.0.0</td>
<td>192.168.0.1</td>
<td>0.0.0.0</td>
<td>UG</td>
<td>eth0</td>
</tr>
</tbody>
</table>

Packet Switch Routing Table

- network may be drawn as graph:
Making Routing Tables

- for trivially small networks, do by hand and load into switches at boot time
- for everything else there’s software

Routing Tables

- routing info kept in tables
  - on hosts
  - in routers
- routes in tables may be
  - static
    - used by hosts
    - may also be used in routers

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>Genmask</th>
<th>Flags</th>
<th>Iface</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.0</td>
<td>*</td>
<td>255.255.255.0</td>
<td>U</td>
<td>eth0</td>
</tr>
<tr>
<td>default</td>
<td>192.168.0.1</td>
<td>0.0.0.0</td>
<td>UG</td>
<td>eth0</td>
</tr>
</tbody>
</table>
Routing Tables

- routing info kept in tables
- routes in tables may be
  - static
  - dynamic
    - initialize from 'static' table
    - adjust with new information while running
    - used by routers
    - generally not used by hosts
- relative pros and cons?

Making Routing Tables

- relative pros and cons?

<table>
<thead>
<tr>
<th>Feature</th>
<th>Static</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational Cost of determining routes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network traffic to perform table updates</td>
<td></td>
<td></td>
</tr>
<tr>
<td>handle congestion or link failure</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Simple Router Setting

- consider small company network:

```
R_1
h_1 h_2 h_3 ... h_n
```

to ISP and Internet

Scaling Up

- recall goal of *universal* service
  - any host can reach any other ⇒ must know route to everywhere
  - introduce a structure to sets of routers
AS: The Bigger Picture

- consider a collection of networks and related routers to be a ‘group’
  - e.g., a single corporation
  - e.g., a university
  - e.g., an ISP
- group is under a single ‘administrative authority’
- group is called an autonomous system (AS)
- traffic in AS is
  - local (originates or terminates in AS)
  - transit (originates and terminates outside AS)

In or Out

- a network consisting of two or more AS will have
  - routers internal to each AS using an IGP
  - routers external to (between) AS using an EGP
- each AS can choose a particular IGP for its internal routing traffic
  - so can be different in different AS
Optimal Routes

- routers should have the optimal route for any destination
- ‘optimal’ with respect to what criterion? (routing metric)
  - lowest cost?
  - lowest delay?
  - lowest hop count/distance?
  - lowest jitter?
- costs are assessed by admins of an AS
  - so who doesn’t use these routing metrics?
- can you really have true optimum path?

RIP

- Routing Information Protocol (RIP) [RFC 2453]
  - distance vector, uses hop count, via UDP
  - designed for use over LANs with hw broad/multi-cast capability
  - has messages to exchange routing tables
    - can include advisement of a default route
  - comes with UNIX
  - slow convergence
  - passive version for hosts
RIP-2 Packet Format

+---------------------+---------------------+---------------------+---------------------+
| Address Family Identifier (2) | Route Tag (2) |
|-------------------------------+-----------------|
+-------------------------------+-----------------|
<table>
<thead>
<tr>
<th>IP Address (4)</th>
</tr>
</thead>
</table>
+-------------------------------|
| Subnet Mask (4)               |
+-------------------------------|
| Next Hop (4)                  |
+-------------------------------|
| Metric (4)                    |
+-------------------------------|

RIP

- basic message contents:
  `<dest_network : distance>`
- distance measured in hops
  - number of networks pkt must travel over
- router receives RIP msg, for each dest_network:
  - if not in my table, add it
  - if lower cost than what I have, replace what I have
- simple to use, little config overhead
OSPF

- **Open Shortest Path First (OSPF)** [RFC 2328]
  - link state (not distance vector)
  - multiple metrics, via IP directly
  - type of service routing via TOS field in IP hdr
  - load balancing
  - information exchanged among OSPF routers by Link State Advertisement messages

OSPF and Areas

- OSPF allows partitioning of an AS into subregions called **areas**
  - AS admin determines layout
  - each AS has an “area 0” **backbone** allowing access from any area to any area in the AS
  - each area's routers use OSPF to communicate info about that area
  - designated ‘edge’ routers join separate areas
    - exchange summaries of what they can reach
    - reduces size of data set exchanged
OSPF and Network Topologies

• three kinds of connections and networks:
  1) point to point between two routers
  2) multi-access network with broadcasting
  3) multi-access networks without broadcasting

• above is achieved by abstracting collection of routers, networks and links into directed graph
  • vertices are routers or networks
  • edges are connections
  • see Stallings §11.4 for an example
    • esp. figures 11.12, 11.13, 11.14

OSPF Protocols

• hello protocol
  • checks that links are operational and elects designated routers and backups
  • hello packets sent every “hello interval”
  • run distributed elections

• exchange protocol
  • used to synchronize routing databases

• flooding protocol
  • distributes updates
  • includes acknowledgment mechanism
EGP

- what do routers ‘between’ AS say to each other?
  - info about routes offered in their respective AS
- how do they say it?
  - using an EGP
  - most widely used is **Border Gateway Protocol**, **BGP** (currently version 4) [RFC 1771]

Border Gateway Protocol (BGP)

- gateways exchange routing information among autonomous systems
- gateway “advertises” that it can reach certain IP networks and its distance to them
- distance-vector based
- distance metrics not standardized
- exterior routing through autonomous systems
  - BGP4 allows use of > 1 path for backup
  - to do this, advertise higher metric to undesirable paths
- uses TCP to exchange routing information
BGP, cont’d

- supports enforcement of AS policy
- important to distinguish between:
  - **advertising**: telling another AS what nets are reachable through this one
  - **routing**: setting up the tables to support forwarding

Multicast Routing Protocols

- distance vector:
  - each node maintains distance (multicast hops) from itself to each dest. (e.g., RIP)
Multicast Routing Protocols

- distance vector:
  - each node maintains distance (multicast hops) from itself to each dest. (e.g., RIP)

- multicast version: distance vector multicast routing protocol (DVMRP)
  - a tunnel counts as one hop

Multicast Routing Protocols, cont’d

- link state:
  - each router creates a ‘link state packet’ for each link to a neighbour containing cost
  - these packets xmit’ed throughout network whenever cost changes (e.g., OSPF)
Multicast Routing Protocols, cont’d

- link state:
  - each router creates a ‘link state packet’ for each link to a neighbour containing cost
  - these packets xmit'ed throughout network whenever cost changes (e.g., OSPF)

- multicast version: multicast open shortest path first (MOSPF)

Multicast Routing Protocols, cont’d

- border gateway:
  - exterior routers advertise distance to other networks they can reach (e.g., BGP)
Multicast Routing Protocols, cont’d

- border gateway:
  - exterior routers advertise distance to other networks they can reach (e.g., BGP)
  - multicast border gateway protocol (MBGP) for inter-domain routing

Multicast Routing Protocols, cont’d

- newest: **Protocol Independent Multicast (PIM)** routing
  - adds “relay points” for multicast-sparse portions of the Internet
  - PIM–Sparse Mode (PIM–SM) intended to avoid redundant, hand-provisioned tunnels
  - PIM–dense (PIM–DM) for multicast-dense portions is like DVMRP
  - for SSM, experimental PIM–SSM
  - primarily available in Cisco routers
Platform Choices

- implementing solutions requires choosing platforms to perform tasks
  - e.g., web servers, db engines
- many solutions possible
- two main categories:
  - proprietary
  - open-source

What is Open Source?

- licensed software
  - users must accept terms of license in order to use
  - several standard licenses, e.g., GPL
    http://www.gnu.org/licenses/gpl.html
- many developers
  - source code is openly available for anyone to work on
  - organized into code repositories, e.g., SourceForge
    http://sourceforge.net
  - scheduled updates and releases
  - developers distributed world-wide
What is Open Source?

- licensed software
- many developers
- delivered:
  - as source code: user compiles and installs
  - as compiled binaries: user copies and installs
  - as ‘packages’: user installs (e.g., RPM)
- free
  - no fee is paid to license the software
  - may pay to have ‘package’ with CDROM and documentation

What’s Available?

- what isn’t?
- operating systems, e.g., linux
- compilers, e.g., GNU gcc
- databases, e.g., MySQL, SAP
- graphical interfaces, e.g., X window
- network software
  - infrastructure, e.g., web servers, e.g., Apache
  - applications
- apps in all areas
The Bandwagon

- open source software obtained from its creator
  - e.g., Snort from snort.org
- major sites housing open source software
  - e.g., gnu.org
- major vendors
  - e.g., IBM (linux runs on z90 series mainframes)
  - e.g., HP now offers choice of XP or linux on new PC purchases

linux Operating System

- linux written by Linus Torvalds
  - adapted from minix written by Andrew Tanenbaum
- derived from UNIX
  - created by Bell Labs in 1970
- multi-user, multi-tasking OS
- runs on (nearly) everything
- current ‘versions’
  - Red Hat Fedora Core
  - Debian, SUSE, Mandrake
  - major vendors: IBM, HP
Popular Applications

- JBoss: java–based application server
  http://www.jboss.org
- Tomcat: java–based web server
  http://jakarta.apache.org/tomcat
- Apache: web server (written in C)
  http://httpd.apache.org
  - most widely–used web server on Internet

Is It Real?

- open source may be OK for a home desktop
- but not suited for ‘real’ work because...
  - not reliable, poor performance
  - it’s not really free (operating costs are higher)
  - no support
  - not fit for mission critical settings
  - not yet mature enough for desktop deployment
  - legal issues over licensing
- do these criticisms hold up?
Price vs Quality

- price not main attraction
  - companies will pay $ for best tech for the job
- Linux typically more reliable than NT for server settings
  - case study: Employease:
    - one of company’s NT servers fails each working day
    - at most two Linux failures a month, often no failures per month
    - faster: increased capacity 50 to 75%
    - “…we cannot risk choosing an inferior solution to save money.”

Not Really Free

- may be ‘free’ to get, but operating costs are purportedly higher for training, support, maintenance
- but:
  - no ‘vendor churn’: need newer versions with new licenses ($$)
  - case study: Sabre Group (includes Travelocity)
    - expects to save, over 5 years, “tens of millions”
  - scales: no additional license fees as installation grows
  - “expecting at least 80 % reduction in running cost”
No Support

- no customer service rep to yell at
- multiple sources means conflicting answers
- but:
  - world-wide, always-available population of developers
  - major software well documented, often in many languages
  - multiple sources can yield better answers than single vendor (who is sometimes wrong)
  - developers offer training programs
    - e.g., JBoss

No Support

- developers offer service using their software
  - e.g., Sourcefire (creators of Snort)
Not For Mission Critical

- risky in terms of security, reliability, maintenance
- case study: Banca Popolare di Milano:
  - had legacy system 90 million lines of COBOL
  - used legacy integration tool (Jacada) connecting COBOL code to IBM’s WebSphere in linux partition

Maturity

- still just for computer nerds
- Siemens Business Services study: linux suitable for desktop use by “nontechnical” workers
- case study: Baylis Distribution
  - moved major service to linux–based platform
  - then moved desktop users to linux environment
  - users see same ‘desktop’ no matter what PC they log on to; no personalized environment problems
  - “We've got better control, better upgradability, and better traceability” at “around half” the $
Legalese

- concern over licensing: who owns rights to code, what affect on right to use?
- SCO suing IBM over linux code purportedly proprietray UNIX code now owned by SCO
  - SCO’s claim to the linux code not established
  - almost no one expects the suit (or related suits) to have any impact on linux users
  - some sources provide indemnification to users
    - e.g., JBoss Group, HP, Red Hat and Novell, Sun