Network Security

Steven M. Bellovin

smb@research.att.com

http://www.research.att.com/~smb

Contents

- Principles of Network Security
- Classes of Attacks
- The Web
- Defenses
  - Firewalls
  - Configuring Firewalls
  - Authentication and Encryption
  - Measuring Vulnerability
- Conclusions

Principles of Network Security

Threats

- A *threat* is a capable and motivated adversary.
- Who are your enemies? Teenage “joy hackers”? Other companies? Disgruntled (ex-)employees? Foreign governments?
- The categories overlap!

Joy Hackers

- Many are “script kiddies”; some are very competent.

⇒ The scripts are very sophisticated.

- The hackers share tools more than the good guys do.

Are Joy Hackers a Problem?

- What would it cost you to rebuild a machine?
- What would your CEO say if you ended up on the front page of the NY Times?
- What if they're working for someone else?
- N.B. Their target selection has improved.

Industrial Espionage

- Less than 5% of attacks are detected. Professionals who are after you won't use your machine to attack other companies, and that's how successful penetrations are usually found.
- Professionals are more likely to use non-technical means, too: social engineering, bribery, wiretaps, etc.
- Professionals tend to know what they want.

Inside Jobs

- Insiders know what you have.
- Insiders often know the weak points.
- Insiders are on the inside of your firewall.
- Etc., etc., etc.

⇒ What if your system administrator turns to the Dark Side?

Spies

- Governments may want your technology.
- Some governments lend tangible support to companies in their own countries.
- Spies tend to be sophisticated, well-funded, etc.

Who Are the Targets?

- Popular organizations.
  Someone always wants to take them down.
- Unpopular organizations.
  The more enemies you have, the more trouble you're in. . .
- More or less anyone.
  New folks on the net have less experience, and are easier targets.

Robbing the Poor

- 2600 Magazine has already carried stories on how to eavesdrop on cable TV-based networks.
- @Home warns against sharing file systems and printers. (I saw several probes while typing this.)
- AOL hackers social-engineer passwords and credit card numbers from naive users.
- Trojan horses with back doors becoming popular (i.e., Back Orifice).

What's at Risk?

- Confidential information.
- Integrity of your system — someone can alter it, or wipe it out entirely.
- Your organization's reputation.
- Possible legal liability, if your system is abused to attack other sites.
- Usability of your network connection.

Should Sites Disconnect?

- Generally not — that's a denial of service attack on yourself.
- Connectivity brings benefits, often including morale.
- Risks should be evaluated and balanced against the benefits.

Attacks

Attack Types

- Authentication problems (often fixable by cryptography).
- Buggy code (most security problems can’t be fixed by cryptography).
- Denial of service.

Preauthenticated Connections

- Some services (i.e., `rlogin`) rely on name-based authentication.
- Users can grant trust via `.rhosts` files; only an administrator should be able to do that.
- Attack strategy: impersonate address or name of trusted machine.

Name-spoofing via the DNS

- An enemy who controls some part of the DNS can control connections to the hosts named by that zone.
- An enemy who controls the address-to-name mapping can fool most address-based authentication schemes, since they're really *name-based*. In other words, an attacker can succeed by spoofing either the source address *or* the DNS.
- Such attacks on the DNS are quite possible, especially by DNS *cache contamination*.

Routing

- IP routers learn the topology of the net by means of *routing protocols* such as OSPF, RIP, BGP, etc.
- An attacker who can inject fraudulent routing messages can subvert any address-based authentication mechanism.
- The threat is especially serious if the enemy can appear “closer” to the target than the site it is impersonating.
- Simply authenticating the source of the routing messages is not enough; checks must be done based on known-valid topologies, and a remote router can’t do even that much.

Source Routing

- Optionally, packets can contain explicit source routes.
- The inverse route is used for replies.
- This bypasses the routing system, making address impersonation easier.

TCP: Sequence Number Attacks

- Connection establishment requires a “3-way handshake”; each party must acknowledge the random initial sequence number used by the other.
- This implies that a fake source IP address won’t do much good, since the spoofer won’t see the other side’s sequence number, and hence can’t ACK it.

⇒ But under certain circumstances, an attacker can guess it—and therefore can spoof a TCP conversation, *without* ever seeing a return packet.

- This breaks address-based authentication.

Electronic Mail

- Email is one of the most important service on the Internet.
- However, source addresses cannot be trusted; mail-spoofing is a trivial exercise.
- Email in transit can be intercepted, especially if spooled on intermediate machines.
- In a totally different vein, the most popular mailer for UNIX systems (`sendmail`) has a long history of serious security problems.

Telnet: Remote Login

- Fundamental mechanism for remote login.
- No built-in authentication; most people use passwords.
- But passwords can be picked up by wiretappers—and this has happened, even on parts of the Internet backbone.
- Without cryptographic authentication, the usual alternative is `rlogin`, which uses address-based authentication. Choose your poison.
- One-time password technologies should be used in all high-threat environments (and that categorization applies more than you might think).

N.B. The same considerations apply to POP3 and IMAP.

NIS: Network Information Service

- Intended to distribute host tables, password file entries, etc., to client workstations on a LAN.
- NIS servers can often be persuaded to give outsiders copies of the password file—and on average, 20% of users pick guessable passwords.
- Under certain circumstances, clients can be told to trust an outside machine's NIS server. This allows the attacker to supply bogus password file entries.

Buffer Overflows

- Many C programs use fixed-length arrays for strings, and don't check inputs.
- Carefully crafted inputs can inject code, and overwrite the return address on the stack to point to this code.
- Publicized by the Internet Worm of 1988 — and responsible for 9 of 13 CERT advisories in 1998.

WWW: The World-Wide Web

- Structured information servers.
- The “killer app” of the Internet; growth is 20% *per month*.
- Clients (Internet Explorer, Netscape, etc.) are multiprotocol.
- Risks to servers: very powerful facilities are needed to process interesting queries.
- Risks to clients: receipt of interpreter instructions:

  Click here to infect your system.

The Web is a separate topic; we’ll return to it in much more detail later.

Some Other Bugs

- 6 of the 12 Microsoft Security advisories thus far this year pertain to buggy Web clients or servers.
- 2 CERT advisories last year concerned flaws in cryptographic protocols.
- Under certain circumstances, a popular firewall could allow access to nominally-protected hosts.
- Often, newer versions of code have fixed the bugs. Have your systems all been updated?

Denial of Service Attacks

- Deny you use of your own resources.
- Some people snap off car antennas; others write viruses.
- Types: memory eaters, bandwidth cloggers, system crashers.
- Hard to defend against; possible if it's cheaper for an attacker to send the message than for you to process it.

Memory Eaters

- Consumes some form of system memory.
- Example: SYN-flooding.
- Defenses: better algorithms, more memory, load-shedding.

Bandwidth Clogger

- Bombards you with messages. Example: “smurf”.
- Attacker sends “ping” to intermediate network’s broadcast address.
- Forged return address is target machine.
- All machines on intermediate network receive the “ping”, and reply, clogging their outgoing net and the target’s incoming net.
- Firewalls at target don’t help — the line is clogged before it reaches there.

System Crasher

- Exploits bugs in target system.
- Example: “teardrop” — overlapping IP fragments crash destination.
- Example: “ping of death” — very large packet crashes target.
- Example: “land” — connect socket back to itself.

The World-Wide Web — Threat or Menace?

The World-Wide Web

- By far the biggest service on the Internet — about 75% of backbone traffic.
- Traffic grows 20% per month.
- Potentially a serious security threat.

WWW: Problem Areas

- Complex administration: easy to get wrong (It took one site I know of three tries to get even simple access controls correct.)
- Complex structure: the servers try to validate source addresses; check passwords; parse file names; implement access restrictions; switch uids (which means they must run as root); etc.
- Scripts. . .

WWW Scripts

- Scripts are, in essence, programs that provide network services. Are they secure?
- Most such scripts are written by ordinary users. . . .
- The languages used to write these scripts are often inappropriate. Perl5, for example, raises security issues.
- The existence of these scripts implies the need for these interpreters (and for programs they invoke, especially for shell scripts) to be accessible to the Web servers.

The Web and Credit Cards

- Sniffing is easy:
  - ⇒ Web queries are short and easier to monitor than `telnet`.
  - ⇒ The number is probably in one packet.
  - ⇒ Credit card numbers are self-checking.
- Even if the number is protected in transit, is it safe then?

⇒ It's sitting on a Web server, in a file accessible to a Web script. . .

An Australian Case

```
From: cskinner@triode.apana.org.au (Christopher T Skinner)
Newsgroups: alt.security
Subject: Australian Penetration
Date: 18 Apr 1995 01:38:45 GMT
Message-ID: <3mv575$rh0@triode.apana.org.au>
Keywords: Credit Card numbers
```

From “The Australian” page 2, 18 Apr95:
AUSnet WWW server penetrated.. hackers claim to have stolen 1200 credit card numbers “yesterday afternoon”... [but] consultant Skeeve Stevens says “It was two-and-half weeks ago when it was first released [???].. The file was left in a public directory..Security was tightened but the technical directors password was public by then.. The files stolen by the hackers contained all the details - name and address card numbers and expirey dates..”
— Jeremy Horey

Wall Street Journal, 23 Apr. 1999

Expert Warns of Safety Glitch In Online-Shopping Software

LOS ANGELES – Internet surfers can tap online shoppers' personal data, including credit-card numbers, when common "shopping cart" software used by small retailers is improperly installed, an expert says.

. . . More than 100 sites on the Web with the vulnerability were found by Mr. Harris, who believes "it's only the tip of the iceberg" and there are hundreds more with similar problems with improperly installed shopping cart software.

Structural Issues

- Complex code is rarely correct.
- Complex administrative structures are rarely correct.
- Strong safeguards — separate uids, `chroot`, etc. — aren't used, or are set up very late in the game.
- Both WWW and system password files are often accessible to outsiders.
- Symbolic links make it very easy for ordinary inside users to give away the store.

Client Problems

- Clients are buggy, too.
- Some bogus URLs can exploit the poor code in the clients.
- It's very easy to download Trojan horses (though only the ease is attributable to the Web).
- Active content. . .

Active Content

- Outsiders supply code to be executed on your machine.
- Can this code be trusted?
- Can it be contained?
- How can we give active content enough power to be useful while still keeping it safe?
  $\Rightarrow$Can users administer fine-grained permissions?

More Client Problems

- Many clients leak information; most users are unaware of this.

```
GET / HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.08 [en] (X11; U; BSD/OS 4.0.1 i386)
Host: localhost:8000
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/pn
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
```

- This information can be used to tailor attacks:

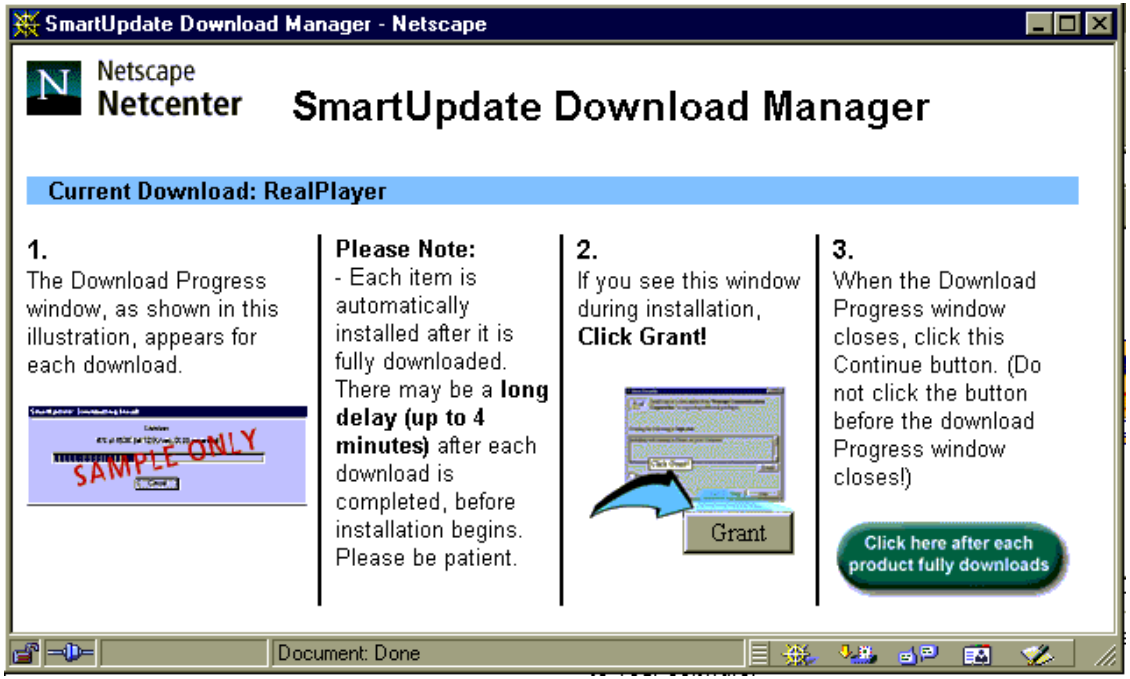
  Click here to infect your system.

- Denial of service attacks:

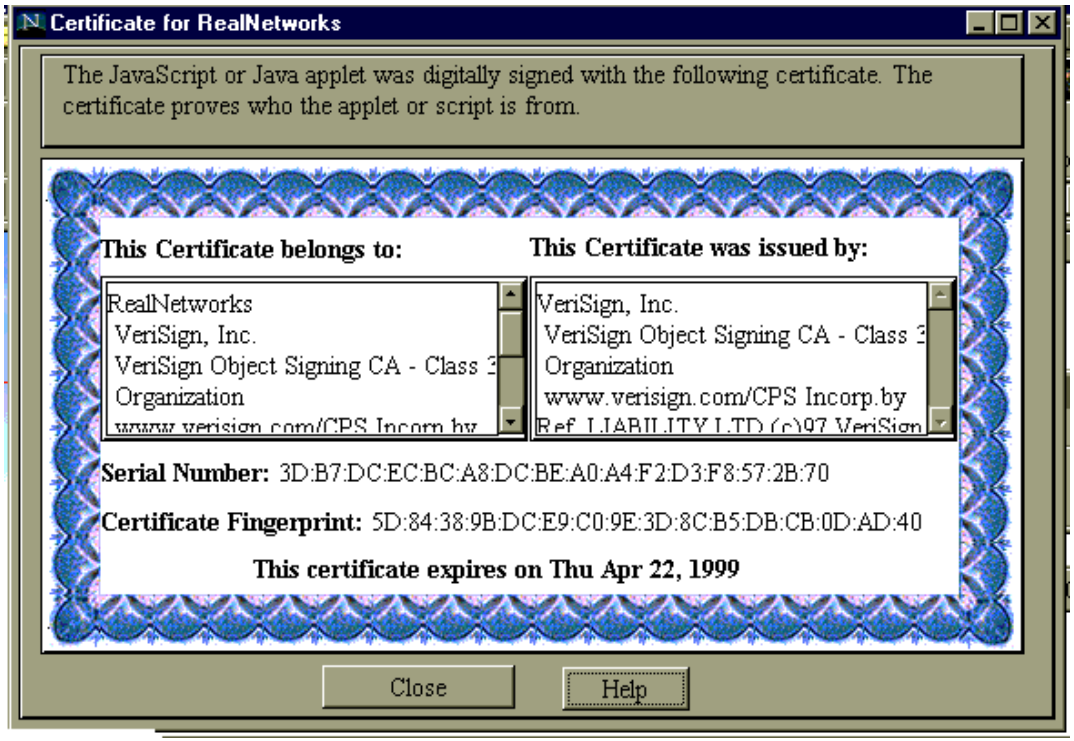
  `gopher://localhost:19`

- Bogus advice on client configuration can add unsafe viewing agents (i.e., `troff`, X).

Dangerous Instructions



A Certificate from April 25, 1999



WX-MAP order form

Welcome to WX-MAP's to go! Your fast order Surface Map store :). For this service to work you ***must be running the X window system***, and you need to issue a `xhost +rs560.cl.msu.edu` on your workstation. You can find a description of all the rendering options here and a list of all station id's here. Again, many thanks to Charley Kline for writing the fine weather mapping program that makes this all possible.

Please enter your display name (i.e. bob.msu.edu:0)

Please enter the weather station ID

Please enter the width of the gif image

Please enter the scale (2.3 covers the entire US)

Please select the number of stations to display (1 to 5, 5 is LOTS)

Please select output format: GIF Backdrop

Render image in Color Monochrome

If you only want your backdrop cleared, no maps drawn, select this

Please check all the options that you would like to see:

ars	clouds	dewpoint	fronts
isobars	isodrosotherms	isotherms	names
nolegend	pressure	radar	regions
selslog	temp	visibility	watches
weather	windbarb		
200 300	500 700	850	

WX-MAP Order Form, Charles Henrich

Firewalls

What's a Firewall

- Barrier between *us* and *them*.
- Limits communication to the outside world.

⇒ The outside world can be another part of the same organization.

- Only a very few machines exposed to attack.

Cyberspace is a Bad Neighborhood

- Many more security incidents are being reported.
- The sophistication of the attacks is growing.
- Some attacks (i.e., the recent “sniffer” incidents) are very far-reaching.
- But more and more organizations *need* to be on the Internet.

Why Use Firewalls?

- Most hosts have security holes.

  Proof: Most software is buggy. Therefore, most security software has security bugs.

- Firewalls run much less code, and hence have few bugs (and holes).
- Firewalls can be professionally (and hence better) administered.
- They enforce the partition of a network into separate security domains.
- *Without such a partition, a network acts as a giant virtual machine, with an unknown set of privileged and ordinary users.*

Firewalls by Analogy

- Passports are (generally) checked at the border.
- My office doesn't have a door direct to the outside.
- My bedroom doesn't have a real lock.
- But a bank still has a vault. . .

Should We Fix the Network Protocols Instead?

- Network security is not the problem.
- Firewalls are *not* a solution to network problems. They are a network response to a host security problem.
- More precisely, they are a response to the dismal state of software engineering; taken as a whole, the profession does not know how to produce software that is secure, correct, and easy to administer.
- Consequently, better network protocols will not obviate the need for firewalls. The best cryptography in the world will not guard against buggy code.
- That said, we need to engineer—and deploy—better security protocols.

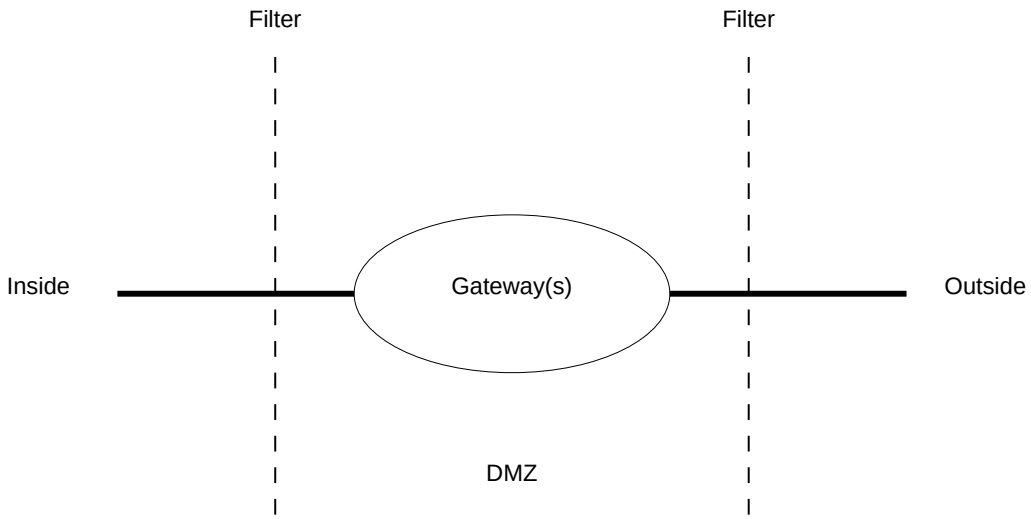
Firewall Advantages

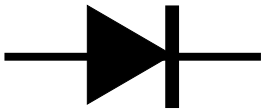
If you don't need it, get rid of it.

- No ordinary users, and hence no `/etc/passwd` entries.
- Run as few servers as possible (zap `rlogin`, `finger`, `www`, NetBIOS, etc.)
- Install conservative software (eliminate `sendmail`, don't get the latest fancy `ftpd`, etc.)
- Log everything, and monitor the log files.
- Keep copious backups, including a "Day 0" backup.

Ordinary machines cannot be run that way.

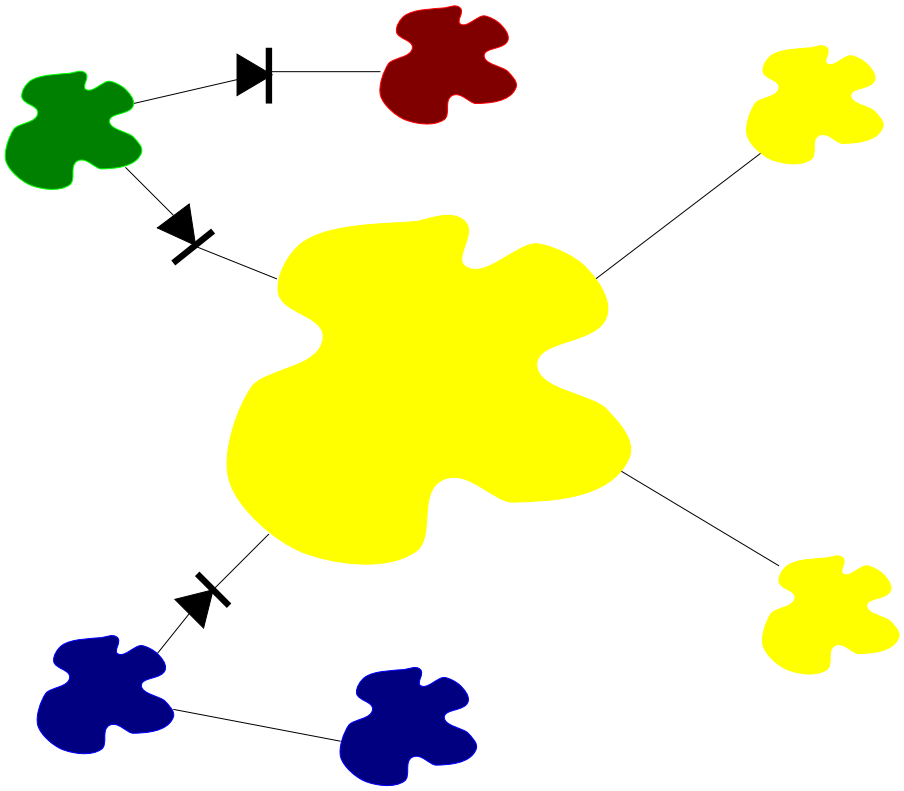
Schematic of a Firewall



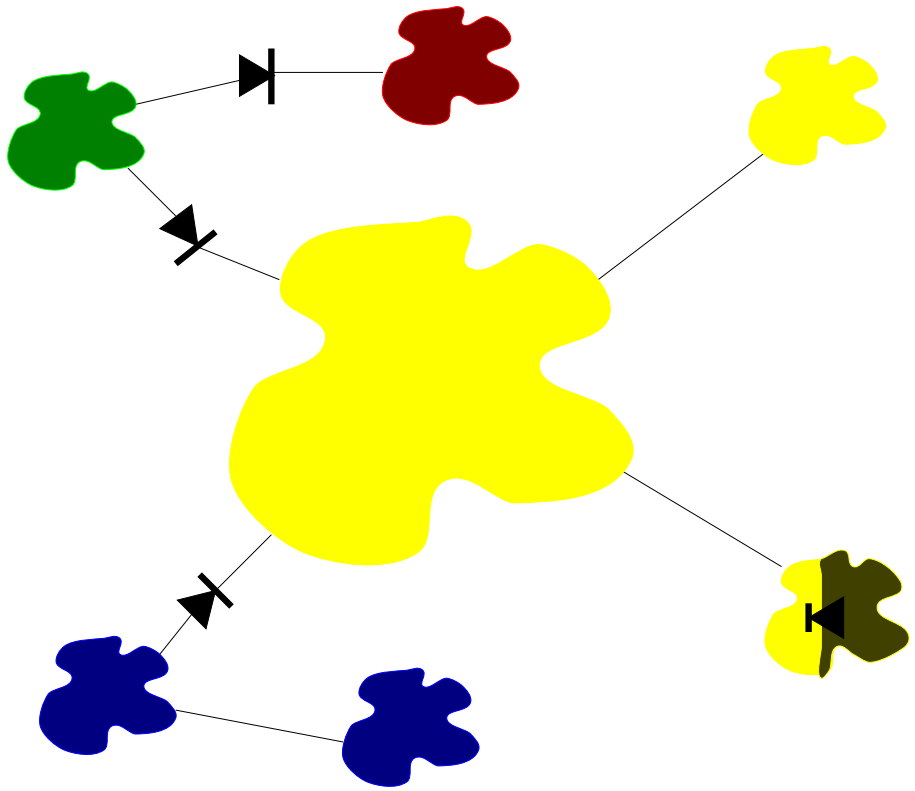


Positioning Firewalls

Firewalls protect *administrative* divisions.



Splitting a Location



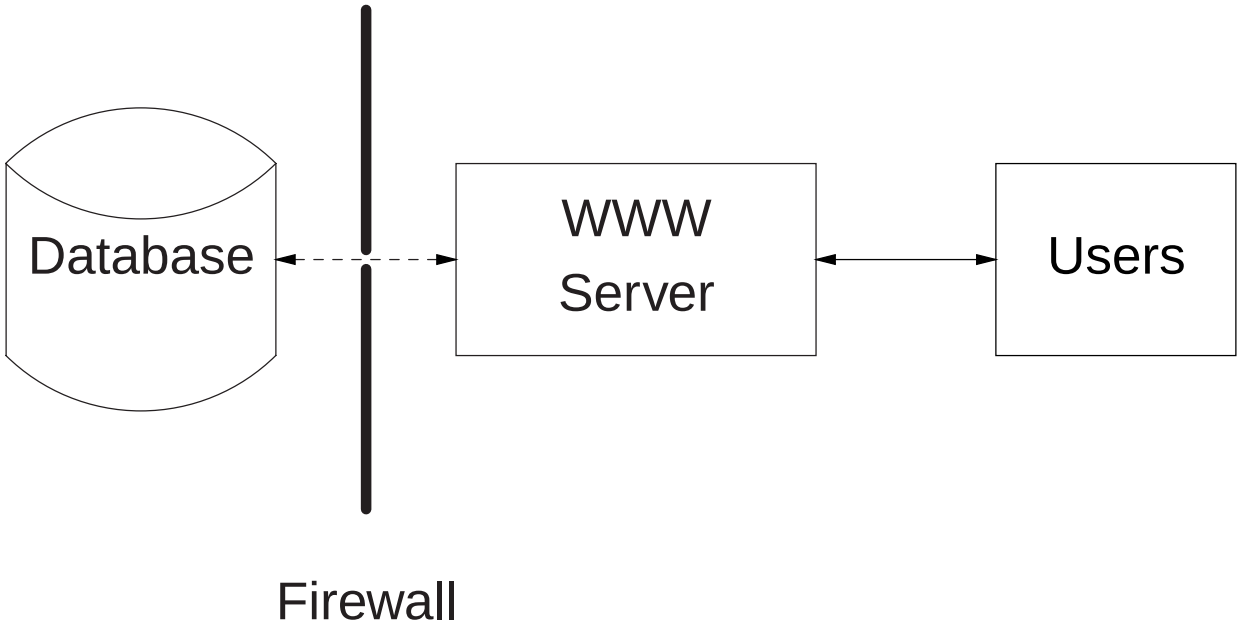
Do Firewalls Still Work?

- Connectivity is easy ($19.95/month. . .)
- Business units may have their own private links to joint venture partners, suppliers, customers, or the Internet.
- You don’t know who has breached your perimeter.
- New philosophy: local firewalls.

Firewalls as Components

- There are too many holes around and through most central corporate firewalls.
- Firewalls are most useful as an element protecting parts of any distributed system.
- A typical networked information system will have *many* small “point” firewalls.

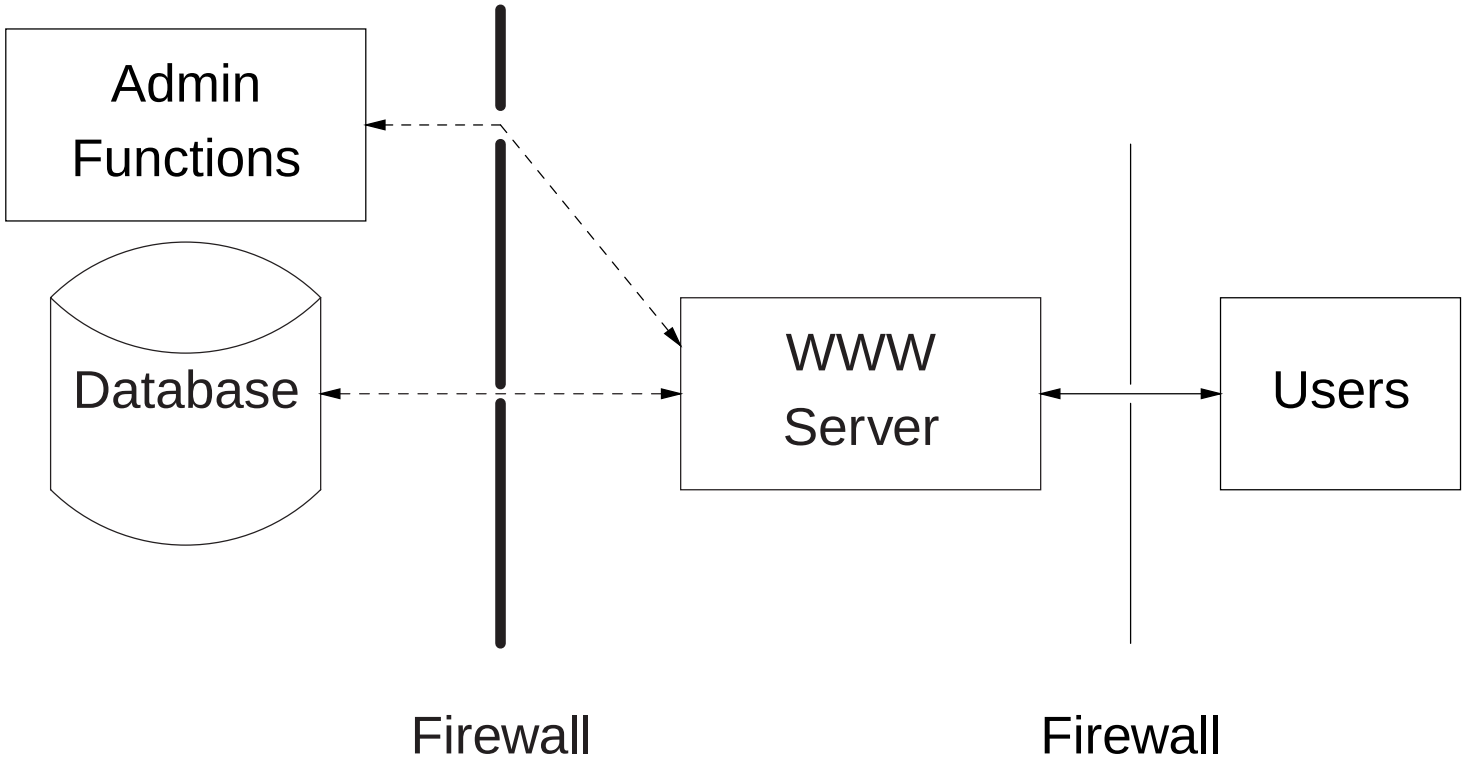
Firewalls — Dealing with Databases



Web Servers and Databases

- Why no firewall protecting the Web server?
  - Turn off everything but the Web server.
  - The Web server is very dangerous anyway.
  - But a firewall can’t protect it; it has to be accessible.
- The channel between the Web server and the database must be very narrow.
- Authentication must be end-to-end, between the users and the database.

Adding an Administrative Interface



Protecting the Administrative Interface

- The Web server may need other ports open for administrative access.
- A second point firewall may protect these ports.
- Often, though, those ports can be cryptographically protected instead.

Firewall Philosophies

1. Block all dangerous destinations.
2. Block everything; unblock things known to be both safe and necessary.

Option 1 gets you into an arms race with the attackers; you have to *know* everything that is dangerous, in all parts of your network. Option 2 is much safer.

Types of Firewalls

- Packet Filters
- Dynamic Packet Filters
- Application Gateways
- Circuit Relays

Many firewalls are combinations of these types.

Packet Filters

- Router-based (and hence cheap).
- Individual packets are accepted or rejected; no context is used.
- Filter rules are hard to set up; the primitives are often inadequate, and different rules can interact.
- Packet filters a poor fit for `ftp` and `X11`.
- Hard to manage access to RPC-based services.
- But often sufficient for point firewalls.

Sample Rule Set

block: theirhost = `spigot`
allow: theirhost = *any* **and**
theirport = *any* **and**
ourhost = `our-gw` **and**
ourport = 25.

Incorrect Rule Set

allow: theirhost = *any* **and**
theirport = 25 **and**
ourhost = *any* **and**
ourport = *any*.

Any remote process on port 25 can call in.

The Right Choice

allow:	theirhost	=	*any* **and**
	theirport	=	25 **and**
	ourhost	=	*any* **and**
	ourport	=	*any* **and**
	(bitset(ACK)	**or**	source = `inside`).

Permit *outgoing* calls. (But “ACK” rule opens the door to “stealth” scanning.)

It’s Worse Than That

- In general, each interface has its own access control list.
- Rules allow packets generally need a complementary rule on another interface to permit reply packets.
- There is no link between these rule pairs.
- It’s even worse if there are more than two interfaces.

Filtering UDP

- UDP has no notion of a connection. It is therefore impossible to distinguish a reply to a query—which should be permitted—from an intrusive packet.
- At best, one can try to block known-dangerous ports. But that's a risky game.
- The safe solution is to permit UDP packets through to known-safe servers only.
- Some systems monitor UDP traffic to learn query/response pairs, and modify the filter rules dynamically. But getting it right is tricky.

Filtering DNS

- How does one prevent DNS contamination?
- Mail can be rerouted, passwords captured, etc.
- Need separate DNS for inside versus outside.

Dynamic Packet Filters

- Standard technology for commercial firewalls.
- Look at query/response pairs; allows in responses to outgoing queries.
- One rule handles both directions; no interactions.
- Can use higher-level semantics to handle things like FTP's data channel.
- Virtually transparent to users and applications.
- Different implementation strategies possible; some are safer than others.

Application Gateways

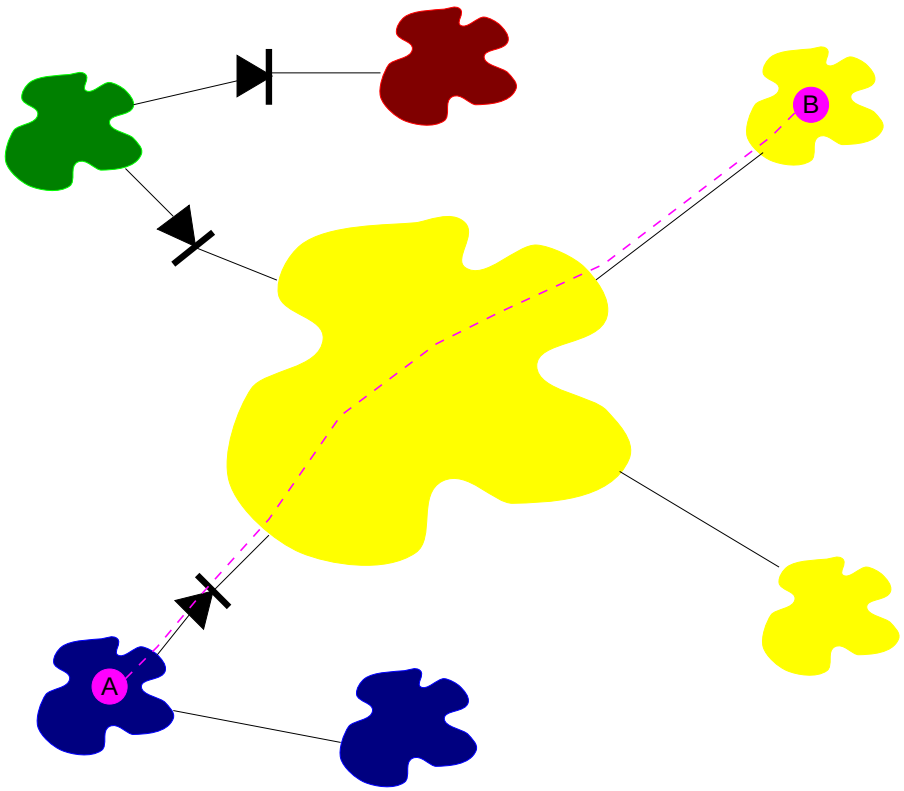
- Gateway machine has custom program for each application.
- Facilities sometimes needed anyway (i.e., mail gateways).
- A good choice for X11 relays or for controlling outbound traffic.

Circuit Relays

- Messages are passed at the TCP level.
- No semantic processing by the gateway.
- Applications must be converted (but this isn't hard, especially if you have the source).
- More flexible than application gateway, but can be subverted.

⇒ The free `socks` package implements circuit relays.

Creating Tunnels



But tunnels are often useful, especially if cryptographically protected.

Defending Against Tunnels

- Any channel can carry traffic; a pair of channels can be use for a tunnel.
- But traffic characteristics for a tunnel differ from the norm.
- Application-level gateways with authenticated connections are the first line of defense.
- Traffic analysis, plus cryptographic analysis of the content, could be applied to all connections.

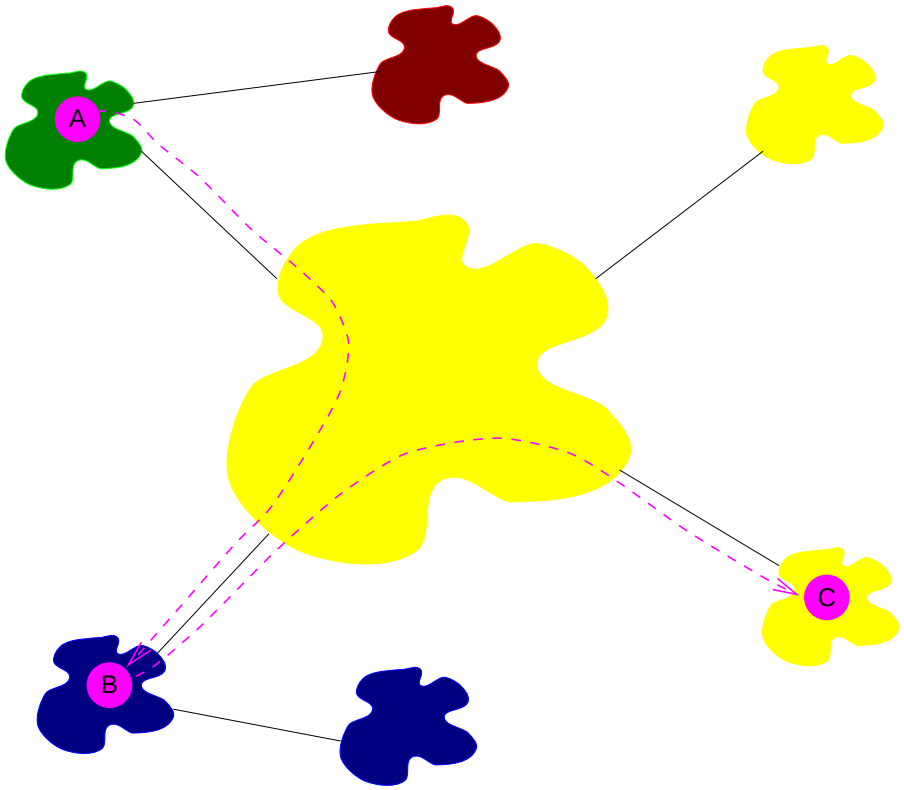
Providing Inbound Services

- Must allow some incoming traffic (mail, `ftp`, login, POP3, etc.)
- When possible, provide service for outsiders on an external machine (i.e., `www` repository).
- Use application gateway for pass-through services.
- High security solutions, such as smart card authentication and encrypted tunnels, are preferable.

How Break-ins Can Spread

- Inappropriate `.rhosts` files.
- Logins via cracked passwords.
- Booby-trapped commands and servers.
- Trojan horses and viruses that steal passwords.

Transitive Trust



If A trusts B and B trusts C, then A trusts C, whether it knows it or not.

Living With Firewalls

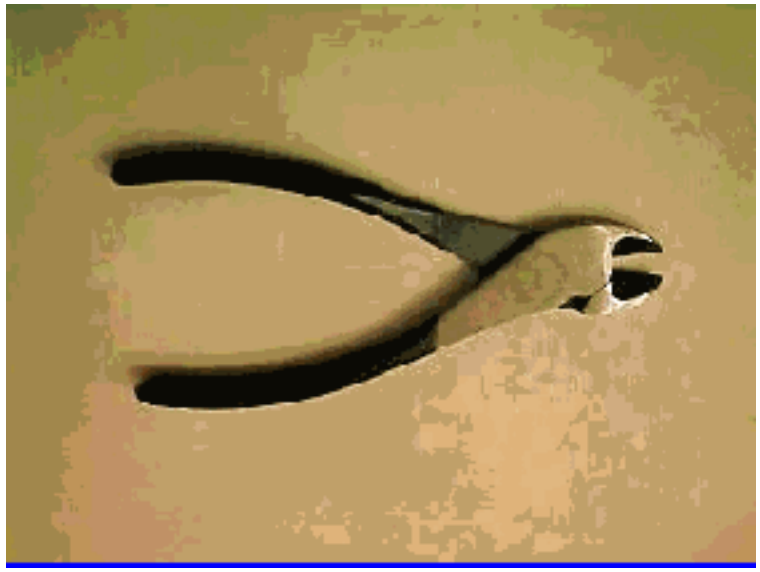
- Decide on a security policy.
- Decide which services fit that policy.
- Build/configure/tweak your firewall to permit those services.
- Evaluate new services using the same criteria.
- Block all others.

Picking a Security Policy

- Decide explicitly what is allowed.
  - Are incoming calls allowed? To what services?
  - Are outgoing calls allowed? Can users export data?
  - Can users import files from the Internet? Source code? Executables?
  - Can insiders use the Internet to dial in when traveling? Under what conditions?
- *You always have a security policy, if only by default.*

A Typical Firewall

A Overly-Strict Firewall



Picking a Platform for a Firewall

- No system is inherently better than any other.
- Use something you *know*, and know very well. “The devil is in the details”.
- Except for services offered on the firewall machine itself, Orange Book-type certification means little. There should be no local users, and the likely errors are of a different type.
- DOS PCs may be better starting points, since they don’t run lots of unnecessary daemons.
- Watch out for ordinary PC systems (Windows NT, Windows ’95, etc.); they run hard-to-disable servers.

What Firewalls Won't Do

- Guard against inside attacks.
- Block all tunnels.
- Secure other entry points, such as modems.
- Protect against security holes at higher levels, such as mail header problems.

Configuring Firewalls

Sample Packet Filters: A Small Business

- Permit incoming mail to a gateway and NTP (network time protocol) to a clock server.
- Permit DNS queries, but zone transfers only to a secondary server.
- Inbound logins go to the mail gateway.
- All outgoing calls are permitted.
- Everything else is blocked.

action	src	port	dest	port	flags	comment
allow	*	*	mail	25		inbound mail access
allow	*	*	mail	53	UDP	access to our DNS
allow	o-dns	*	mail	53		secondary nameserver access
allow	*	*	mail	23		incoming telnet access
allow	o-ntp	123	i-ntp	123	UDP	external time source
allow	in	*	*	*		outgoing TCP packets are OK
allow	*	*	in	*	ACK	return ACK packets are OK
block	*	*	*	*		nothing else is OK
block	*	*	*	*	UDP	block other UDP, too

Sample Packet Filters: A University

- Block DNS zone tranfers.
- Block `rsh` and `rlogin`; we can't trust them.
- Permit POP3 but not IMAP; the latter has been buggier.
- Block incoming X Windows.
- Only permit encrypted access to the transcript manager.
- Wall off an open PC lab; it's been the source of trouble.
- Allow other TCP, but not UDP.

action	src	port	dest	port	flags	comment
allow	o-dns	*	i-dns	53		allow our secondary nameserver
block	*	*	*	53		no other DNS zone transfers
allow	*	*	*	53	UDP	permit UDP DNS queries
allow	o-ntp	123	i-ntp	123	UDP	ntp time access
block	*	*	*	512		no incoming ”r” commands . . .
block	*	*	*	513		. . .
block	*	*	*	514		. . .
block	*	*	*	143		imap
block	*	*	*	6000		no incoming X
				⋮		
block	*	*	*	6031		
allow	*	*	admin	444		secure access to transcript mgr
block	*	*	admin	*		nothing else
block	pclab	*	*	*		students in pclab can’t go out
block	pclab	*	*	*	UDP	. . . not even with FSP!
allow	*	*	*	*		all other TCP is OK
block	*	*	*	*	UDP	suppress other UDP for now

The Problem with FTP

- The data channel uses an incoming call to a random port number.
- The actual port used is typically sent via a PORT command, but most packet filters don't (and can't) watch for them.
- PORT command can be abused to make your machine attack other sites.
- Solution: change the FTP clients to use the PASV command instead, so that the data channel is an outgoing call (see RFC 1579). Most Web browsers already do this.

Authentication and Encryption

Authentication

Authentication relies on one or more of

- Something you know, such as a password or PIN.
- Something you have (a smartcard or key).
- Something you are (biometrics).

Authentication Problems

Each of these have their weaknesses:

passwords Can be guessed, leaked, written down, stolen, wiretapped.

tokens Inconvenient; expensive.

biometrics Expensive; how do you change keys?

Authentication on the Internet

⇒ Given the problems with passwords, they are *not* acceptable for use on the Internet. Some form of one-time password is mandatory.

- We use challenge/response authenticators or smart cards.
- The Bellcore S/Key system is a useful alternative, since it does not require any special hardware. NRL has an improved and enhanced version (OPIE). *Warning*: there are tools that do password-guessing on S/Key strings.

Basic Cryptographic Terms

- *Symmetric* (or *conventional*) cipher uses one key for encryption and decryption.
- *Asymmetric* (or *public key*) cryptosystem uses separate keys for encryption and decryption.
- A *digital signature* uses a private key to sign something, and a publicly-known key to verify the signature.
- A *certificate* is a digitally-signed binding between an identity and a public key.

Key Lengths

- The length of keys is an upper bound on the strength of a cipher.
- 40-bit keys can be cracked by students in a few hours.
- Corporations can crack 56-bit DES.
- Governments are *probably* stopped by 90-bit keys.
- With modern ciphers, using more key bits has little or no effect on encryption speed.

N.B. Public key ciphers need much longer keys; the lengths are not directly comparable.

What is IPSEC, and Why?

- Network-layer security protocol for the Internet.
- Completely transparent to applications. (But causes trouble for NAT boxes.)
- TCP- or application-level retransmissions handle deleted or damaged packets.
  - Generally must modify protocol stack or kernel; out of reach of application writers or users.

Uses for IPSEC

- Virtual Private Networks.
- “Phone home” for laptops, telecommuters.
- General Internet security.

History

SP3 Layer 3 security protocol for SDNS.

NLSP OSIfied version of SP3, with an incomprehensible spec.

swIPe UNIX implementation by Ioannidis and Blaze.

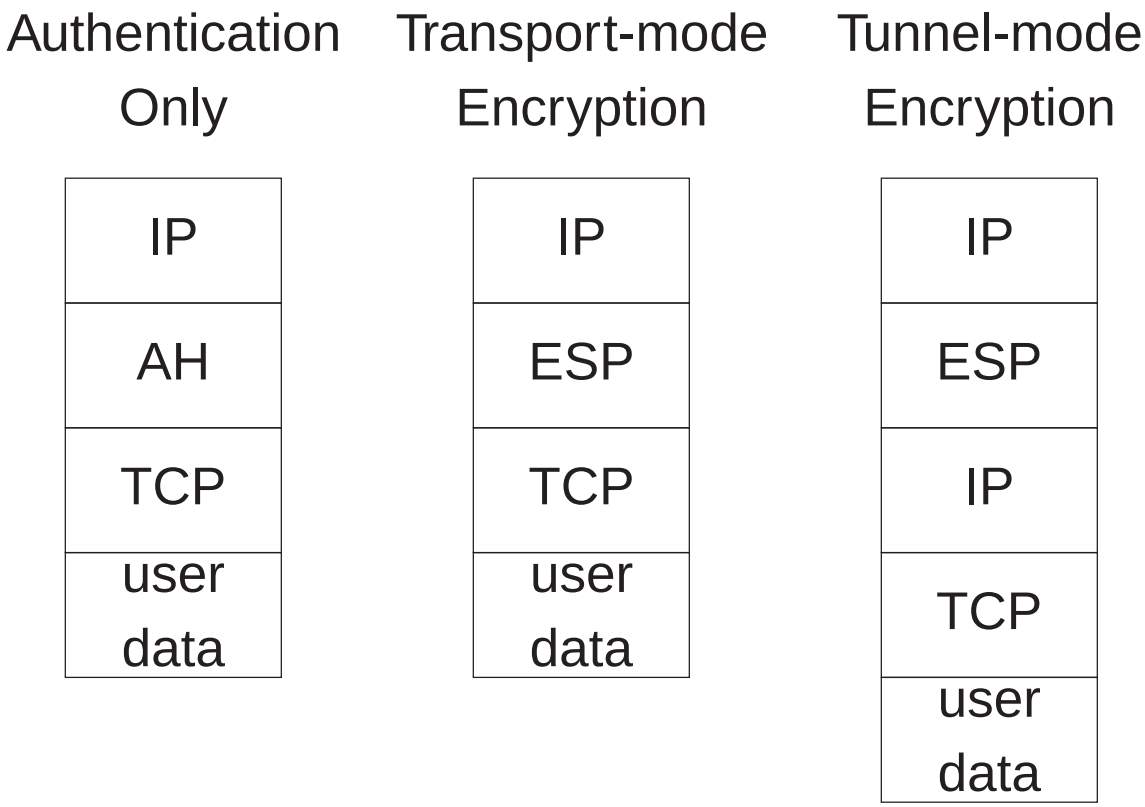
IPSEC First version of IPSEC standards (1995).

IPSEC Current version of IPSEC (1998), with key management and some policy constraints.

IPSEC Structure

- Nested headers: IP, ESP, maybe another IP, TCP or UDP, then data.
- Cryptographic protection can be host to host, host to firewall, or firewall to firewall.
- Option for user-granularity keying.
- Works with IPv4 and IPv6.

Packet Layout



Transport Mode

- Intended for host-to-host communication.
- Not yet widely available, but expected to be standard in Windows 2000.

Tunnel Mode

- Primarily used for gateway-to-gateway or host-to-gateway communication.
- Useful for virtual private networks, telecommuters, etc.
- Outer IP header stripped off by gateway; inner header contains internal IP addresses.
- Many interoperable implementations available today.

Authentication Header (AH)

- Based on keyed cryptographic hash function.
- Covers payload and portion of preceeding IP header.
- Uses *Security Parameter Index* (SPI) to identify security association, and hence key, algorithm, etc.

Encryption Header (ESP)

- Encrypts payload.
- Includes anti-replay counter, integrity check — no need to use AH with ESP.
- SPI identifies security association.

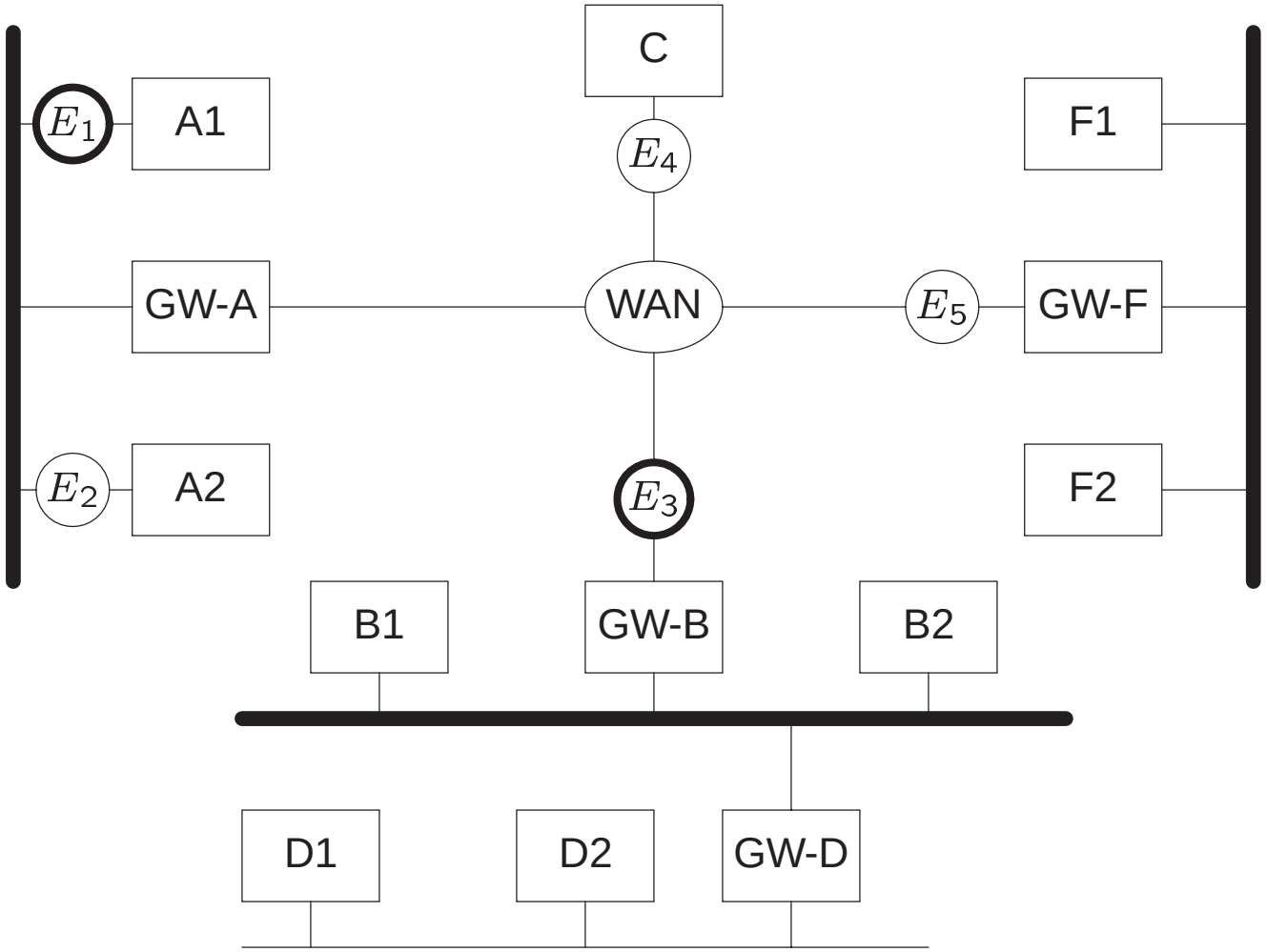
Key Management

- Public-key based negotiation, using IKE (Internet Key Exchange).
- Provides “perfect forward secrecy”.
- Can use either shared secrets or certificates.
  But — different implementations use different certificate formats.

Policy Table

- Says what packets to encrypt, based on IP addresses and port numbers.
- Inbound packets matched against the table, too.
- If no suitable security association exists, one must be negotiated.

Topologies



Paths

- A1 to F1, F2:
  Encryptors E_1, E_5
- B1, B2, D1, D2 to F1, F2:
  Encryptors E_3, E_5
- A2 to C:
  Encryptors E_2, E_4

IPSEC and Firewalls

- Encryption is not authentication.
- Access controls may need to be applied to encrypted traffic, depending on the source.
- The source IP address is only authenticated if it is somehow bound to the certificate.
  $\Rightarrow$Proper firewalls tie inbound permissions to certificate.
- Encrypted traffic can use a different firewall; however, co-ordination of policies may be needed.

IPSEC and the DNS

- IPSEC often relies on the DNS.
  - Users specify hostnames.
  - IPSEC operates at the IP layer, where IP addresses are used.
  - An attacker could try to subvert the mapping.
- DNSSEC — which isn't deployed yet, either — uses its own certificates, not X.509.

Implementation Issues

- How do applications request cryptographic protection? How do they verify its existence?
- How do adminstrators mandate cryptography between host or network pairs?
- We need to resolve authorization issues.
- What about export?

Secure Shell (SSH)

- Replacement for `rlogin`, `rsh`.
- Can forward other ports as well; often used to tunnel mail, web proxies, etc.
- Original UNIX version free for non-commercial use.
- Versions available for Windows, too.

Protecting Electronic Mail

- Cryptographic protection and authentication of email is important in some environments.
- Two different systems, S/MIME and PGP, can fill those roles.
- PGP is more popular among techies; corporations like S/MIME.
  $\Rightarrow$Popular Web browsers include S/MIME mailer.
- There are legal versions available in the U.S. and abroad.

Encryption: WWW

- SSL is widely used for purchases, to protect credit card numbers.
- Many browsers use encryption with 40-bit keys.
- As noted, these credit card numbers are often easily stolen from the servers.
- Users virtually never check certificates.

Measuring Vulnerability

Testing Your Security

- Configuring a firewall is tricky.
- You don't *know* if your setup is secure until it is tested.
- If you don't test it, someone else probably will. . .

The Hacker's Workbench

- All of these tools cut both ways.
- It is not worth taking extraordinary precautions to hide them; the hackers have their own set anyway. In fact, they share their hacking tools a lot more freely than administrators do.
- If the good guys don't help each other, they're helping the bad guys.

Target Selection

- The DNS is a fruitful source of data:

```
dig axfr zone @target.com +pfset=0x2020
```

- Blocking zone transfers doesn't help much:

```
for i in ‘seq 1 254‘
do
  dig +pfset=0x2020 -x  192.20.225.$i
done
```

Scanning a Network

- Ping each possible address.
- See who replies.
- Faster if you write your own tool, but simple shell scripts work, too:

```
for i in ‘seq 1 254‘
do
  ping -c 3 192.20.225.$i
done
```

Scanning a Host

- Attempt connection to all possible ports.
- See if any interesting services are running.
- Much faster if connections are done in parallel, but a simple loop will suffice.
- Newer tools do “stealth” scanning.
- For RPC, the standard `rpcinfo` command will ask the `portmapper`.

Probing for Weaknesses

- Attempt to grab a password file via TFTP.
- Check anonymous FTP permissions and files.
- Use `showmount` to check for NFS exports.
- Attempt to log in as *guest*. Use `rexecd`, since it doesn't log.
- Are any Windows file systems exported?
- Try to grab the password file via NIS.

Routing Games

- Use a recent version of `telnet` to experiment with source routing.
- SNMP can let you dump router tables. Are any unusual entries present on your inside net?
- Send bogus ICMP packets at your machines, and see what happens.

Monitoring the Net

- Buy a dedicated Ethernet monitor.
- Use a PC-based tool.
- Use `tcpdump` on a UNIX machine.

⇒ Caution: hackers *love* to find a machine that can monitor a net in promiscuous mode. But they bring their own software with them.

Tiger Teams

- Large sites should have formal “tiger teams”.
- Support from upper management is vital.
- Many former(?) hackers are now in the security business. Can you trust them?
- Don’t neglect non-technical adjuncts: social engineering, dumpster diving, etc. Again, the hackers do these things.
- User education is vital. *Don’t* hide details, *don’t* rely on security through obscurity. Most legitimate users are happy to co-operate if they’re told why.

Logging

- Exposed machines should do lots of logging.
- You want to know if you're under attack.
- Don't skimp on disk space; it's cheap, and you don't want hackers filling up the log partition before launching their real attack.
- Unfortunately, most systems do far too little logging.
- How many of your own attacks were detected by your logs?

Intrusion Detection Systems (IDS)

- Intended to alert you to hostile activity.
- Two basic types: known attack signatures and profile analysis.
- Advantages and disadvantages to each.

Attack Signatures

- Program in patterns of known attacks, such as overly-long file name requests.
- Almost no false positives.
- But — useless against new attacks.

Profile Analysis

- System is trained on “normal” behavior (users, network, etc.).
- Behavior that differs from the norm is flagged.
- False positives — users sometimes do different things.
- False negatives — some attacks aren’t unusual enough.
- But — can detect unknown attacks.

Placement of an IDS

- Outside vs. inside the firewall — the former only tells you if you're under attack; the latter can tell about successful attacks.
- Network vs. host — the former is less intrusive, and can correlate probes of different hosts; the latter can do a better job detecting certain attacks.
- One size does not fit all.

Lures and Honeypots

- If you have the energy, you can set up dummy services and accounts.
- Better yet, set up an interesting-sounding (but unused) internal machine, and look for *any* access to it.
- We monitor various attractive ports, etc.
- It is sometimes possible to attempt counter-intelligence.
- But make sure that the volume of messages doesn't get annoying; AI techniques may not be needed, but simple summary messages will often suffice.
- Machine-parseable log files are very convenient; it's easier to generate pretty stuff via a back-end program if you need it.

Conclusions

Recommendations: Firewalls

- Firewalls aren't a panacea, but they're a big help.
- Internal firewalls are needed, too.
- Firewalls are a perimeter defense; if you have no well-defined perimeter, you have no defense.

Recommendations: The Web

- “Ride the tiger”.
- Application-level filtering of Web traffic is a good idea, but isn’t foolproof; Web servers can live on many different ports. (`www.nsa.gov` lives on port 8080.)
- *Simple* Web proxy servers can help, but determined users can probably bypass them.
- Careful configuration of browsers and servers is important.

Recommendations: Cryptography

- Inbound traffic from the Internet, except for public services such as mail, should be encrypted.
- Even internally, address-based authentication should be replaced by cryptographic authentication.
- Use keylengths of at least 128 bits.

Recommendations: People

- It is extremely difficult to use technical means to enforce more security than the organizational culture accepts.
- Internet access is very easy; a PC, a modem, and $20/month will expose you to most of the dangers.
- The trick is to *manage* the risk.
- Reasonable Internet access and educated users are probably the ways to do that.

Useful Free Software

Wietse Venema's code: ftp://ftp.porcupine.org/pub/security/index.html

Cryptographic links: http://www.counterpane.com/sites.html

S/Key: ftp://ftp.bellcore.com/pub/nmh/skey/

IPSEC for Linux: http://www.xs4all.nl/~freeswan/

CERT's Tech Tips and tools: http://www.cert.org/tech_tips

SOCKS: http://www.socks.nec.com

Fortify http://www.fortify.net

Mailing Lists

firewall-wizards Subscribe at
http://www.nfr.net/forum/firewall-wizards.html

bugtraq To subscribe, send a message to listserv@netspace.org with
`subscribe bugtraq`
in the body of the message.

NTbugtraq To subscribe, send a message to
listserv@listserv.ntbugtraq.ntadvice.com with
`subscribe ntbugtraq firstname lastname`
in the body of the message.

Further Reading

A comprehensive (and growing) archive of security papers can be found at http://www.cs.purdue.edu/coast/archive.

Applied Cryptography, B. Schneier, John Wiley and Sons, 1996 (2nd Edition), ISBN 0-471-11709-9. Also see the archive of cryptographic papers at http://www.counterpane.com/biblio.

Firewalls and Internet Security: Repelling the Wily Hacker, W.R. Cheswick and S. Bellovin, Addison-Wesley, 1994, ISBN 0-201-63357-4.


HACKERS