Today

- network defenses

network attacks

- denial of service (DoS)
- account capture
- impersonation/spoofing
- scan and exploit
- advanced

security goals

- integrity
- privacy
- availability

How do we do it? – encryption, keyed-hash

No single solution.

defenses

Where to defend?

- at the network perimeter?
- at the host?
- at the application?

YES

encryption

Where to encrypt?

- at the physical layer?
- at the router?
- at the host network layer?
- at the application layer?

depends

host defenses

- logs/alarms
- tcpwrappers, limit services
- one-time passwords
- file checksums
- self scan (ISS)
- get latest patches
- backups

perimeter defenses

- filtering routers
  - allow all, block some
  - block tftp, 111, etc
  - block source routing
  - block local addr. from external
  - allow only local addr. out
- firewalls – deny all, allow a few
- intrusion detectors

cisco filters

complex, easy to mess up

```
! Permit outbound traffic from authorized nets
!
access-list 102 permit ip 128.219.0.0 0.0.255.255 0.0.0.0 255.255.25
access-list 102 permit ip 134.167.0.0 0.0.255.255 0.0.0.0 255.255.25
access-list 102 permit ip 192.12.68.0 0.0.0.255 0.0.0.0 255.255.255.
! Deny external SNMP traffic to LMES/LMERC nets
!
access-list 112 deny   udp 0.0.0.0 255.255.255.255 0.0.0.0 255.255.2
! Deny packets with source addresses that belong inside LMES/LMERC
access-list 112 deny ip 128.219.0.0   0.0.255.255 0.0.0.0 255.255.25
access-list 112 deny ip 134.167.0.0   0.0.255.255 0.0.0.0 255.255.25
access-list 112 deny ip 192.12.68.0   0.0.0.255   0.0.0.0 255.255.25
! Deny all other inbound TFTP access
!
access-list 112 deny   udp 0.0.0.0 255.255.255.255 0.0.0.0 255.255.2
```

firewalls

- better logging than routers
- looks deeper than network layer
- complex rule sets
- deny all, allow a few things
- use with proxy servers
- performance?

firewalls

- establish a perimeter
- single point of protection (rather than host by host)
  - always invoked
  - tamper resistant
  - small and simple (understandable)
- establish a policy
  - What's not denied is allowed
  - What's not allowed is denied
- security vs convenience

firewalls

What they don't do?

- don't do UDP very well
- don't prevent session hijacking
- don't provide privacy
- don't improve throughput
- don't prevent backdoors (modems, tunnels)

intrusion detection

- scan incoming/outgoing packets
- log and alarm, block?
- known attack patterns (port scans, host scans, smurfs)
- statistical profiles – "normal" usage
- content analysis (password guessing, su, uname)
- on each side of firewall
- commercial solutions
- develop your own with tcpdump/libpcap

encryption

- privacy
- integrity
- prevent spoofing
- prevent hijacking
- issues of key management, legality, exportable

application encryption

- RPC has authentication options
- email: PGP or S/MIME
- login/rsh: secure shell (ssh)
- web: ssl (https:)

Kerberos

- trusted, third-party authentication service
- part of MIT's project Athena (public domain)
- components: library, data base, authentication daemon, ticket-granting service
- uses authenticators (for users and servers) and tickets
- provides:
  1. authenticated messages
  2. safe messages (encrypted checksum)
  3. fully encrypted messages
- needs network time
- uses single-key encryption (DES) (keys)
- applications must be "kerberized"
- assumes hosts are not trustworthy

Kerberos

- no cleartext passwords
- users and servers are registered
- single-signon
- API available for safe and private messages, do it before you send and after you receive
- part of DCE and Windows 2000

session

1. user logs in, kerberized login sends < client name, TGS server name > to Kerberos KDC
2. KDC replies $\{< SK_{TGS}$, {client name, WS addr, TGS-name, $SK_{TGS}\}K_{TGS} > \}K_c$ user's password K_c is used to decrypt message
3. to get a ticket for another service, client sends a message to TGS, with authenticator (encrypted with SK_{TGS}), the sealed TGS ticket, and the server name
4. TGS replies with a $\{< SK_{server}$, {client name, WS addr,sever, $SK_{server}\}K_{server}$ all encrypted with SK_{TGS}
5. the client can send a request to the server consisting of the server's encrypted ticket, and an authenticator encrypted with SK_{server}
6. the server can decode the ticket and get the session key SK_{server} and decode and verify the client (check for replay)
7. server adds 1 to timestamp and sends to client encrypted with SK_{server} (mutual authentication)

kerberizing

- you can add Kerberos calls to your own client/servers
- need Kerberos data base, authenticator, ticket-granting server, and administrative programs
- can use **klogin**, but better if you have kerberized BSD utilities
- Kerberos calls added to login, r-utilities, NFS
- **rlogin -x** sets up encrypted session, every packet is encrypted

Why not?

- every network service must be modified
- Kerberos server must be physically secure
- doesn't protect against trojan horses
- if password is disclosed, eavesdropper can decrypt other tickets and spoof servers and users

Still, better than anything else.

Is IP secure?

- checksums?
- can you trust the source address?
- IP security option (RFC 760)
  - military security model: subject/object labels
  - label each IP datagram (secret, top secret, unclassified)
  - IP stack and routers enforce access controls
  - only effective in very controlled environment

IP security option

RFC 760

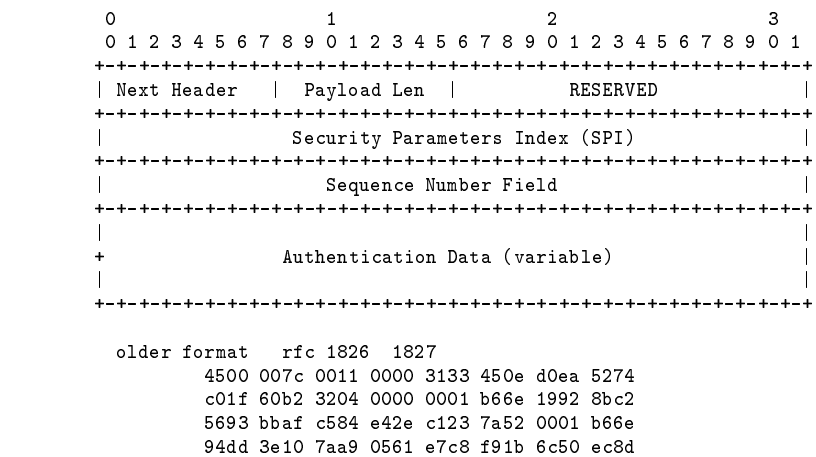
```
 Security

    This option provides a way for DOD hosts to send security and
    TCC (closed user groups) parameters through networks whose
    transport leader does not contain fields for this information.

      +--------+--------+---------+--------+
      |00000010|00000100|000000SS |  TCC   |
      +--------+--------+---------+--------+
        Type=2  Length=4

    Security:  2 bits

      Specifies one of 4 levels of security

        11-top secret
        10-secret
        01-confidential
        00-unclassified

    Transmission Control Code:  8 bits

      Provides a means to compartmentalize traffic and define
      controlled communities of interest among subscribers.
```


IPsec

- IPv6 and extensions to IPv4 for security
- IP protos 50 and 51
- authenticated and/or encrypted datagrams
- system manager sets "policy"
- no changes to application (or optional)
- key management (IKE)
- can tunnel IP datagrams (VPN)
- implementations available
- US export controls limiting

authentication header

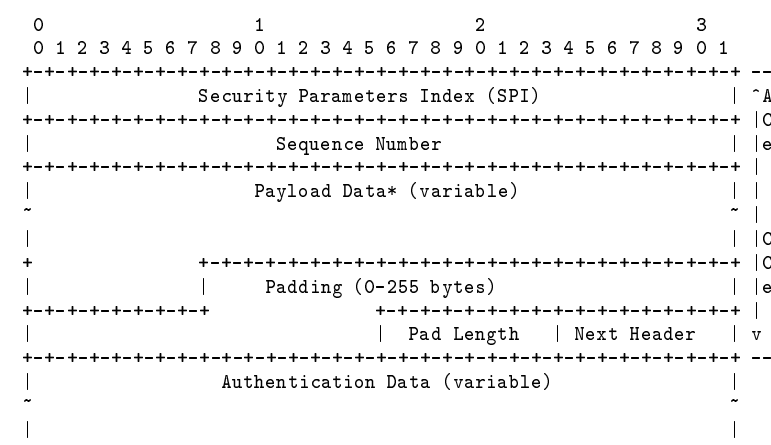
IP proto 51 RFC 2402




```
  older format   rfc 1826  1827
          4500 007c 0011 0000 3133 450e d0ea 5274
          c01f 60b2 3204 0000 0001 b66e 1992 8bc2
          5693 bbaf c584 e42e c123 7a52 0001 b66e
          94dd 3e10 7aa9 0561 e7c8 f91b 6c50 ec8d
```


encapsulating security payload

IP proto 50 RFC 2406




```
 4500 7000 daa5 0000 2f32 a0df c01f 60bc
 d0ea 52dc 0000 0102 0000 0002 4e84 b477
```

VPN

Virtual Private Network

- construct encrypted tunnels over the Internet
- shrink-wrapped IPsec (many vendors), or PPTP
- router-tunnels and standalone clients (roaming)
- clients for linux, max, win*
- no changes to client applications
- network address translation(NAT) makes client appear like its on local net

what is effect on performance?

net security

- there is a threat
- write good code
- use secure tools
- IPsec will help
- be suspicious

take a security course :-)

next time

perl, java, winsock