Network Security

Adolfo Rodriguez
CPS 214

DUKE Computer Science

Telco/Internet Comparison

- Telephone System
  - central authority
  - network in control
  - billing records per connection
  - legal issues well understood
  - provisions for law enforcement (wiretapping)
- Internet
  - no central authority
  - end systems in control
  - no central knowledge of connections
  - no per-packet billing
  - legal issues not well understood
  - anonymity is easy

DUKE Computer Science

Internet Security Stinks

- Hosts are hard to secure
- Bad defaults
- Poor software
- Fixes rarely applied
- Average user/administrator is clueless
- An overly secure system is not useful
- It's difficult to coordinate among sites

DUKE Computer Science

Security Goals

- Confidentiality
  - Snooping
  - Encryption
- Integrity
  - Deletion, changes
  - Backups
- Availability
  - Denial of service attacks
- Authentication
  - Are who you say you are?
- Nonrepudiation
  - No denying it
- Access Control
  - Don't touch that!
- Reputation
  - Ensure your good name

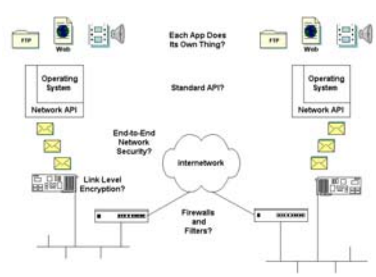
DUKE Computer Science

Challenges

- Increased overhead
- Complexity
- Performance!
- Is it really secure?
- Management

DUKE Computer Science

Where to Put the Protection?



DUKE Computer Science

Which Layer?

Function				Example
Ultimate data destination	Application	↔	Application	Web browser
Format conversion	Presentation	↔	Presentation	ASCII/XDR
Interaction across presentation	Session	↔	Session	Restartable file transfer
Reliable, ordered delivery	Transport	↔	Transport	TCP
Routing/ Internetworking	Network	↔	Network	IP
Data framing over links	Data link	↔	Data link	Ethernet, ATM
Bits on the wire	Physical	↔	Physical	SONET, 100BT

Which Layer?

Function			
Ultimate data destination	Application	↔	HTTPS/SSH
Format conversion	Presentation	↔	
Interaction across presentation	Session	↔	
Reliable, ordered delivery	Transport	↔	TLS/SSL
Routing/ Internetworking	Network	↔	IPSec
Data framing over links	Data link	↔	WEP
Bits on the wire	Physical	↔	Lock bldg!

Physical Security

- Trash bins
- Social engineering
  - Rubber hose attacks are the most dangerous
  - Disgruntled employee
  - Curious, but dangerous employee
  - Clueless and dangerous employee
- It's much easier to trust a face than a packet
- Protect from the *whoops*
  - power
  - spills
  - the clumsy
  - software really can kill hardware

Host Based Security

- Recall End-to-End Argument
- Security is ultimately a host problem
- Key idea: protect the *DATA*
- End hosts are in control of data
- Users are in control of end hosts
- Users can and often will do dumb things
  - Especially when others help them to!
- Result: very difficult to protect all hosts

Security by Obscurity

- Is no security at all.
- However
  - It's often best not to advertise unnecessarily
  - It's often the only layer used (e.g. passwords)
- Probably need more security

Password Cracking

- Very common today
- If attacker can get a hold of the password file, they can go offline and process it
- Recall
  - passwords are a form of obscurity
  - multiple defenses may be needed
- Given enough time, passwords alone are probably not safe

Viruses, Worms, and SpyBots

- Programs written with the intent to spread
- Worms are very common today
  - Often email based (e.g. ILOVEYOU)
- Viruses *infect* other programs
  - Code copied to other programs (e.g. macros)
- All require the code to be executed
  - Proves users continue to do dumb things
  - Sometimes software is at fault too

Network Based Security

- Should augment host based security
- Useful for
  - Protecting groups of users from others
  - Prohibiting certain types of network usage
  - Controlling traffic flow
- Difficult to inspect traffic
  - Encryption can hide bad things
  - Tunneling can mislead you

Layered Defenses

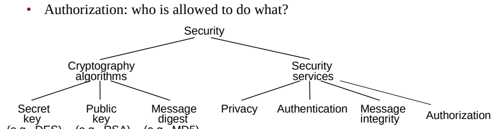
- The *belt and suspenders* approach
- Multiple layers make it harder to get through
- Multiple layers take longer to get through
- Basic statistics and probability apply
  - If Defense A stops 90% of all attacks and Defense B stops 90% of all attacks, you might be able to stop up to 99% of all attacks
- Trade-off in time, money, performance and convenience

Exploits Overview

- Passwords
  - hacking and sniffing
- System specific holes
  - NT, UNIX, NetWare, Linux
- Application (implementation) specific
  - web browser, ftp, email, finger
- Protocol specific
  - spoofing, TCP session hijacking, ICMP redirects, DNS
- Denial of Service
  - PING of death, SYN flood

Security Methods

- Cryptography functions
  - Secret key (e.g., DES)
  - Public key (e.g., RSA)
  - Message digest (e.g., MD5)
- Security services
  - Privacy: preventing unauthorized release of information
  - Authentication: verifying identity of the remote participant
  - Integrity: making sure message has not been altered
  - Authorization: who is allowed to do what?



Encryption

- Use a “secret” machine or algorithm
  - How do you know when it has been compromised?
  - German “Enigma”. First cracked in 1932 by Marian Rejewski, a Polish Mathematician. Then again in WW2 by British in 1939 by Alan Turing (founder of computer science)

Encryption

- Make a readable message unreadable
- Math intensive
- Plain text versus cipher text
- Algorithms and keys
  - public
  - private
  - key size

DUKE Computer Science

An unbreakable method

- One Time Pad – Hide message in noise!
  - Start with a sequence of random numb
    - r1, r2, r3, ….
  - Break message into number sequence
    - m1, m2, m3, ….
  - Compute x-or sum
    - c1= r1+m1, c2 = r2+m2, c3 = r3+m3, …
  - Recover message by
    - m1=c1+r1, m2=c2+r2, …
- Both parties must have copy of random sequence
  - Sequence must be truly random
    - Otherwise patterns can be detected

DUKE Computer Science

Shared Secret Key

- Each party knows a secret
- The secret is used to decrypt the cipher text
  - Book: Ulysses
  - Page: 7
  - Line: 23
  - Word: 4
- Must know the book and keep it a secret

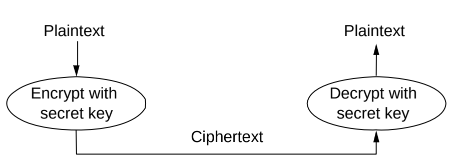
DUKE Computer Science

Shared Secret Key Illustrated

DUKE Computer Science

Secret Key (DES)

Data Encryption Standard uses a secret key.



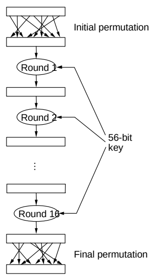
DUKE Computer Science

Main ideas of DES

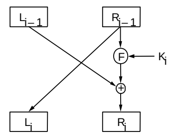
- **1972 - NBS issued a call for proposals:**
  - **Must provide high level of security.**
  - **Must be completely specified and easy to understand.**
  - **The algorithm itself must provide the security.**
  - **Must be available to all users.**
  - **Must be adaptable for use in diverse applications.**
  - **Must be economical to implement in electronic devices.**
  - **Must be efficient.**
  - **Must be able to be validated.**
  - **Must be exportable.**
- **1974 - IBM responded with "Lucifer"**
- **1976 - DES officially adopted.**

DUKE Computer Science

- 64-bit key (56-bits + 8-bit parity)
- 16 rounds

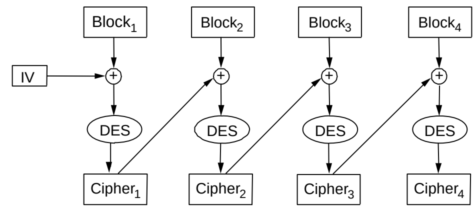


- Each Round



DUKE Computer Science

- Repeat for larger messages



DUKE Computer Science

Public Key Cryptography

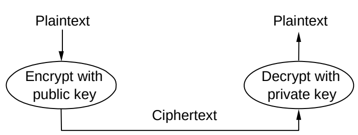
- Public Key
  - Everyone can use it to encrypt messages to you
- Private Key
  - Only you know this key and only it decrypts messages encrypted with your public key
- Keyring
  - Contains other people's public keys
  - How do you build this? Why is this hard?

DUKE Computer Science

Public Key Illustrated

DUKE Computer Science

Public Key (RSA)



- Encryption & Decryption
  - Let (e,n)=encryption key, (d,n) = decryption key
  - Let m = message, c = cipher text

$$c = m^e \bmod n$$
$$m = c^d \bmod n$$

DUKE Computer Science

How does this work?

- Every person x has a public key e(x) and a private key d(x)
- If I want to send a an encrypted message m to x, I compute $c = m^{e(x)} \bmod n$
  - X decripts it with his private key $m = c^{d(x)} \bmod n$
- Assumptions
  - Everybody that wants to send me a message must know my public key and n
  - I am the only person who has my private key
- How do we get d, e and n?

DUKE Computer Science

RSA in detail

- Choose two large prime numbers *p* and *q* (each 256 bits)
- Multiply *p* and *q* together to get *n*
- Choose the encryption key *e*, such that *e* and (p - 1) x (q - 1) are relatively prime.
  - Two numbers are relatively prime if they have no common factor greater than one
- Compute decryption key *d* such that

$$d = e^{-1} \bmod ((p - 1) \times (q - 1))$$

- Construct public key as (e, n)
- Construct private key as (d, n)
- Discard (do not disclose) original primes *p* and *q*

DUKE Computer Science

How can I break it?

- Suppose we have cipher text c and public key (e, n). We want m so we need d.
  - If $c = m^e$ then need to do $m = c^{(1/e)} = \sqrt[e]{c}$
  - Need to find d so that e*d = 1 mod (p-1)(q-1)
  - So find p and q!
  - n = p*q so just factor n.
    - Oh, that is hard!
  - Is there another function that can be used to get e given d and n?
    - Unknown.
    - Widely believed that any other method would be just as hard as factoring.

DUKE Computer Science

Performance Issues

- To protect the contents of a message, encrypt it!
  - Can use DES or RSA.
    - DES can do several hundred Mbps.
    - RSA is slow (100 Kbps)
  - Must use DES, but the key may be discovered.
    - Solution: only use it for a while.
    - Called a session key
  - How do we share the session key?
    - If we have RSA infrastructure, can exchange key with RSA and use DES for the session
    - Key distribution problem

DUKE Computer Science

Key Distribution

- Certificate
  - special type of digitally signed document:
    - *"I certify that the public key in this document belongs to the entity named in this document, signed X."*
  - the name of the entity being certified
  - the public key of the entity
  - the name of the certified authority
  - a digital signature
- Certified Authority (CA)
  - administrative entity that issues certificates
  - useful only to someone that already holds the CA's public key.

DUKE Computer Science

Key Distribution (cont)

- Chain of Trust
  - if *X* certifies that a certain public key belongs to *Y*, and *Y* certifies that another public key belongs to *Z*, then there exists a chain of certificates from *X* to *Z*
  - someone that wants to verify *Z*'s public key has to know *X*'s public key and follow the chain
- Certificate Revocation List

DUKE Computer Science

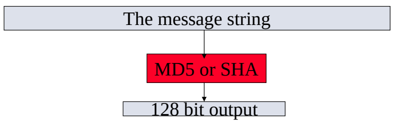
Message integrity

- I send a message M.
  - I don't care who sees the message but
    - I don't want it tampered with (no modifications)
    - I don't want anybody to forge messages from me.

DUKE Computer Science

Message Digest

- Cryptographic checksum
  - Like a regular checksum which protects eceiver from accidental changes to the message
  - A cryptographic checksum protects the receiver from malicious changes to the message.



DUKE Computer Science

Message Integrity Protocols

- Digital signature using RSA
  - special case of a message integrity where the code can only have been generated by one participant
  - compute signature with private key and verify with public key
- Keyed MD5
  - sender: *m* + MD5(*m* + *k*) + E(*k*, *senders private key*)
  - receiver
    - recovers random key using the sender's public key
    - applies MD5 to the concatenation of this random key message
- MD5 with RSA signature
  - sender: *m* + E(MD5(*m*), *senders private key*)
  - receiver
    - decrypts signature with sender's public key
    - compares result with MD5 checksum sent with message

DUKE Computer Science

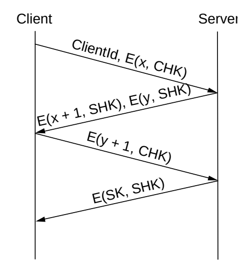
The important properties

- One-way function
  - given a cryptographic checksum for a message, it is virtually impossible to figure out what message produced it
  - it is not computationally feasible to find two messages that hash to the same cryptographic checksum.
- Relevance
  - if you are given a checksum for a message and are able to compute exactly the same checksum for that message, then it is highly likely this message produced the checksum you were given

DUKE Computer Science

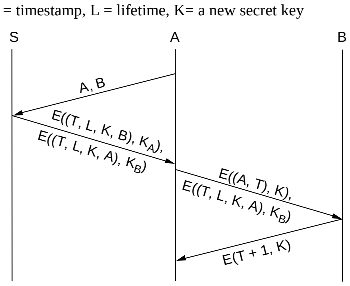
Authentication Protocols

- Three-way handshake
  - Assume client and server each know the others secret keys.
  - Client selects a random number x.
  - At end of handshake authentication is established?
- How did each side get the keys?



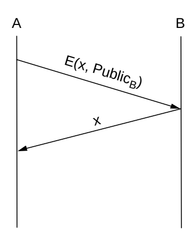
DUKE Computer Science

- Trusted third party (Kerberos)
  - K_A is a secret key shared between A and S. K_B similar
  - T = timestamp, L = lifetime, K= a new secret key



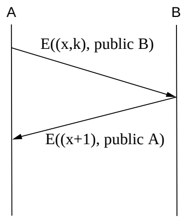
DUKE Computer Science

- Public key authentication :
  - One way: A wants to know if it is talking to B



DUKE Computer Science

- Using RSA to authenticate and establish a session Key :
  - Let x be random and k be a session key



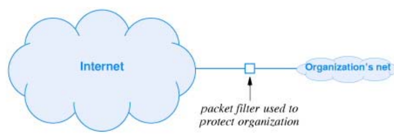
Firewall Solutions

- They help, but not a panacea
- A network response to a host problem
  - Packet by packet examination is tough
- Don't forget internal users
- Need well defined borders
- Can be a false sense of security
- Careful not to break standard protocol mechanisms!

Packet Filtering Firewalls

- Apply rules to incoming/outgoing packets
- Based on
  - Addresses
  - Protocols
  - Ports
  - Application
  - Other pattern match

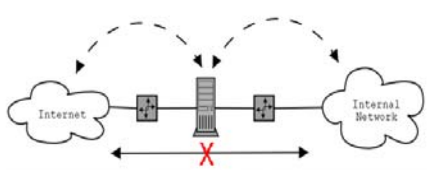
Packet Filtering Firewall Illustrated



Example Firewall: ipchains

```
-A input -s 192.168.0.0/255.255.0.0 -d 0.0.0.0/0.0.0.0 -j DENY

-A input -s 172.0.0.0/255.240.0.0 -d 0.0.0.0/0.0.0.0 -j DENY

-A input -s 10.0.0.0/255.0.0.0 -d 0.0.0.0/0.0.0.0 -j DENY

-A input -s 224.0.0.0/224.0.0.0 -d 0.0.0.0/0.0.0.0 -j DENY

-A input -s 0.0.0.0/0.0.0.0 -d a.b.c.d/255.255.255.255 22:22 -p 6 -j ACCEPT

-A input -s 0.0.0.0/0.0.0.0 -d a.b.c.d/255.255.255.255 1024:65535 -p 6 ! -y -j
   ACCEPT
```

Application Level Gateway

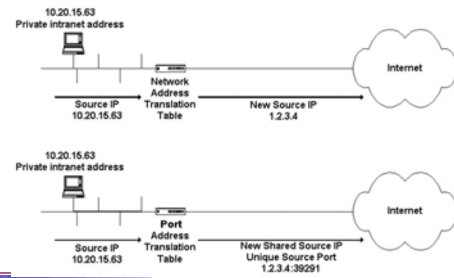


Network Address Translation

- Removes end-to-end addressing
- Standardized in RFC 1918
- NAT has been bad for the Internet
- Provides relatively no security with a great deal of cost - this slide shouldn't be here
- NAT has been required for sites with IP address allocation problems
- NAT may be used for IPv6 transition

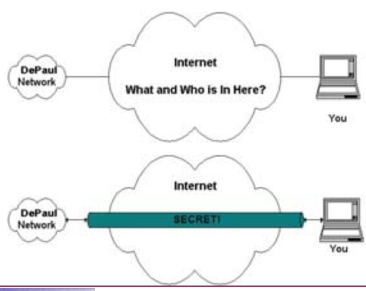
DUKE Computer Science

NAT Illustrated



DUKE Computer Science

Virtual Private Networks



DUKE Computer Science

Why VPNs?

- Cost, Cost, Cost!
- Ability to make use of a public, insecure network, rather than building your own private, secure network
- Connect business branches as if we had an expensive leased line

DUKE Computer Science

IPSec

- Authentication Header (AH)
  - Data Origin Authentication
  - Anti-replay service
  - Data Integrity
- Encapsulating Security Payload (ESP)
  - Confidentiality
  - Data Origin Authentication
  - Anti-replay service
  - Connectionless Integrity

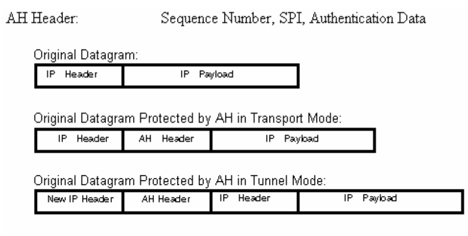
DUKE Computer Science

AH

- AH provides authentication for as much of the IP header as possible, as well as for upper level protocol data
- Tow modes: transport mode/tunnel mode

DUKE Computer Science

AH Location



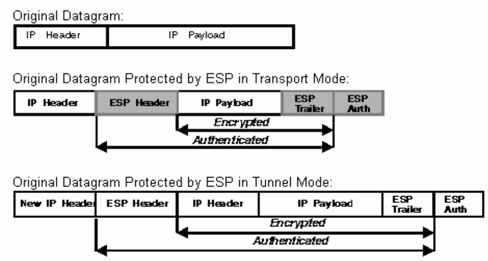
AH Algorithms

- Keyed Message Authentication Codes (MAC) based on Symmetric Key Encryption(DES)
- One-way hash function (MD5/SHA-1)

ESP

- Provides Data Confidentiality to IP payload using Encryption
- It can provide Data Integrity and connectionless Integrity, but the coverage is different from AH
- Two: transport Mode/Tunnel Mode

ESP Format



ESP Algorithms

- Encryption Algorithms
  - Symmetric Encryption Algorithms
- Authentication Algorithms
  - The same as AH