Introduction to Security Principles

Shambhu Upadhyaya
Wireless Network Security
CSE 566 (Lectures 3,4)

Outline

- Basic Encryption Methodology
- Message Authentication and Integrity
- Program Security
- Network Security
- Intrusion Detection
- Firewalls

Popular Encryption Methods

- Stream Ciphers
  - One Time Pad
  - RC4
- Block Ciphers
  - DES/AES
  - RSA
  - RC5

Stream Ciphers

- Processes the message bit by bit (as a stream)
- Typically has a (pseudo) random **stream key**
- Combined (XORed) with plaintext bit by bit
- Randomness of **stream key** completely destroys any statistically properties in the message
  - $C_i = M_i \text{ XOR } StreamKey_i$
- Concept is very simple!
- Stream key should not be reused
  - If reused the patterns can be used to re-identify the message

Stream Cipher Properties

- Some design considerations are:
  - Long period with no repetitions
  - Statistically random
  - Depends on large enough key
  - Large linear complexity
  - Correlation immunity
  - Confusion
  - Diffusion
  - Use of highly non-linear boolean functions

RC4

- A proprietary cipher owned by RSA DSI
- Another Ron Rivest design, simple but effective
- Variable key size, byte-oriented stream cipher
- Widely used (web SSL/TLS, wireless WEP)
- Key forms random permutation of all 8-bit values
- Uses that permutation to scramble input info. processed a byte at a time

RC4 Key Schedule

- Starts with an array S of numbers: 0..255
- Use key to well and truly shuffle
- S forms **internal state** of the cipher
- Given a key k of length l bytes

```
for i = 0 to 255 do
  S[i] = i
j = 0
for i = 0 to 255 do
  j = (j + S[i] + k[i mod l]) (mod 256)
  swap (S[i], S[j])
```


RC4 Encryption

- Encryption continues shuffling array values
- Sum of shuffled pair selects "stream key" value
- XOR with next byte of message to en/decrypt

```
i = j = 0
for each message byte Mi
  i = (i + 1) (mod 256)
  j = (j + S[i]) (mod 256)
  swap(S[i], S[j])
  t = (S[i] + S[j]) (mod 256)
  Ci = Mi XOR S[t]
```

RC4 Security

- Since RC4 is a stream cipher, must **never reuse a key**
- Used in **SSL** and **WEP**
- Claimed secure against known attacks
  - Falls short of the standards of a secure cipher in several ways, and thus is not recommended for use in new applications
- Fluhrer, Mantin and Shamir Attack can be used to break the cipher

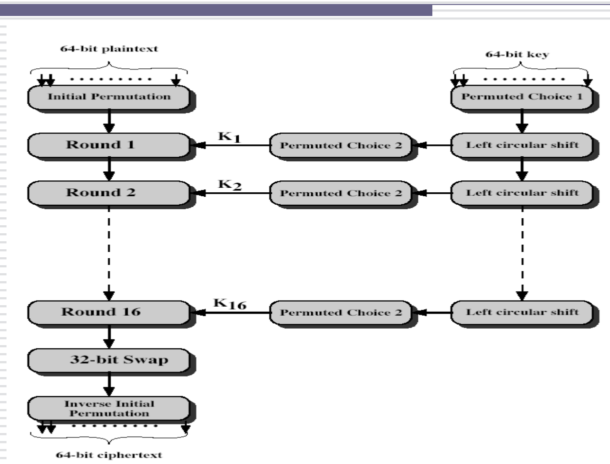
Block Cipher Characteristics

- Features seen in modern block ciphers are:
  - Variable key length/block size/no. of rounds
  - Mixed operators, data/key dependent rotation
  - Key dependent S-boxes
  - More complex key scheduling
  - Operation of full data in each round
  - Varying non-linear functions

DES Algorithm

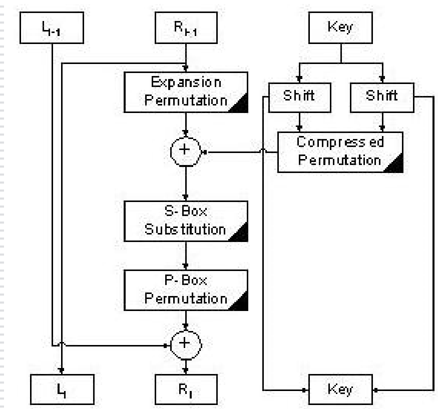
- Confusion and Diffusion
- 64 bit block cipher, plaintext is encrypted in blocks of 64 bits
- Substitution and Permutation (Transposition)
- 56 bit key + 8 bit parity = 64 bit key
- Repetitive nature – shift and xor
- Outline
  - Split data in half
  - Scramble each half independently
  - Combine key with one half
  - Swap the two halves
  - Repeat the process 16 times

DES Encryption



One Round of DES

- $L_i = R_{i-1}$
- $R_i = L_{i-1}$ xor $F(R_{i-1}, K_i)$



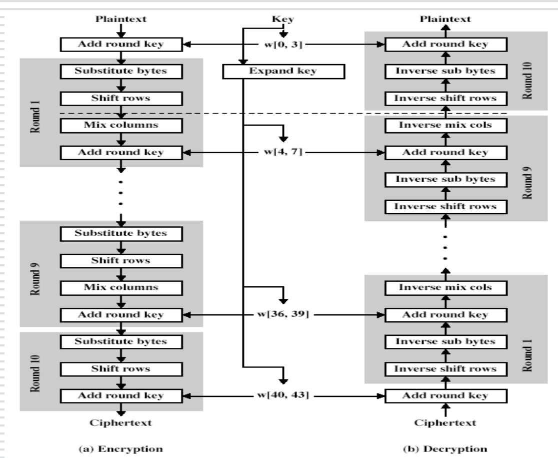
AES (Advanced Encryption Standard) Requirements

- Private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger & faster than Triple-DES
- Active life of 20-30 years (+ archival use)
- Provide full specification & design details
- Both C & Java implementations
- NIST have released all submissions & unclassified analyses

Rijndael

- Processes data as 4 groups of 4 bytes (state)
- Has 9/11/13 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multiply of groups)
  - add round key (XOR state with key material)
- Initial XOR key material & incomplete last round
- All operations can be combined into XOR and table lookups - hence very fast & efficient

Rijndael



Public Key Infrastructure

- Based on mathematical functions
- Asymmetric (two separate keys)
- Enhances confidentiality, key distribution and authentication
- Ingredients are
  - Plaintext
  - Encryption algorithm
  - Public and private keys
  - Ciphertext
  - Decryption algorithm

RSA

- RSA encryption and decryption are commutative, hence it may be used directly as a digital signature scheme
- Given an RSA scheme {(e,R), (d,p,q)}
- To **sign** a message, compute:
  - $S = M^d (\mathrm{mod}\ R)$
- To **verify** a signature, compute:
  - $M = S^e (\mathrm{mod}\ R) = M^{e.d} (\mathrm{mod}\ R) = M (\mathrm{mod}\ R)$
- Thus know the message was signed by the owner of the public-key

RSA

- Would seem obvious that a message may be encrypted, then signed using RSA without increasing its size
- But have blocking problem, since it is encrypted using the receivers modulus, but signed using the senders modulus (which may be smaller)
- Several approaches possible to overcome this
- More commonly use a hash function to create a separate message digest which is then signed

RC5

- A proprietary cipher owned by RSADSI
- Designed by Ronald Rivest (of RSA fame)
- Used in various RSADSI products
- Can vary key size / data size / no. of rounds
- Very clean and simple design
- Easy implementation on various CPUs
- Yet still regarded as secure

RC5 Ciphers

- RC5 is a family of ciphers RC5-w/r/b
  - w = word size in bits (16/32/64)

  data=2w
  - r = number of rounds (0..255)
  - b = number of bytes in key (0..255)
- Nominal version is RC5-32/12/16
  - i.e., 32-bit words, so encrypts 64-bit data blocks
  - using 12 rounds
  - with 16 bytes (128-bit) secret key

RC5 Key Expansion

- RC5 uses 2r+2 subkey words (w-bits)
- Subkeys are stored in array `S[i]`, i=0..t-1
- Then the key schedule consists of
  - Initializing S to a fixed pseudorandom value, based on constants e and phi
  - The byte key is copied (little-endian) into a c-word array L
  - A mixing operation then combines L and S to form the final S array

RC5 Encryption

- Split input into two halves A & B
  $L_0 = A + S[0];$
  $R_0 = B + S[1];$
  for i = 1 to r do
  $L_i = ((L_{i-1} \text{ XOR } R_{i-1}) <<< R_{i-1}) + S[2 \times i];$
  $R_i = ((R_{i-1} \text{ XOR } L_i) <<< L_i) + S[2 \times i + 1];$
- Each round is like 2 DES rounds
- Note rotation is main source of non-linearity
- Need reasonable number of rounds (e.g., 12-16)

RC5 Modes

- RFC2040 defines 4 modes used by RC5
  - RC5 Block Cipher, is ECB mode
  - RC5-CBC, is CBC mode
  - RC5-CBC-PAD, is CBC with padding by bytes with value being the number of padding bytes
  - RC5-CTS, a variant of CBC which is the same size as the original message, uses ciphertext stealing to keep size same as original

Message Authentication

- Message authentication is concerned with
  - Protecting the integrity of a message
  - Validating identity of originator
  - Non-repudiation of origin (dispute resolution)
- Can be provided by three methods
  - Message encryption
  - Message authentication code (MAC)
  - Hash function

Message Encryption

- Message encryption by itself also provides a measure of authentication
- If symmetric encryption is used then
  - Receiver knows sender must have created it
  - Since only sender and receiver know the key used
  - Content cannot be altered
  - If message has suitable structure, redundancy or a checksum to detect any changes

Message Encryption

- If public-key encryption is used
  - Encryption provides no confidence of sender at all
  - Since anyone potentially knows public-key
  - However if
    - Senders **sign** message using their private-key
    - Then encrypts with recipients public key
    - Have both secrecy and authentication
  - But at the cost of two public-key uses on message

Message Authentication Code (MAC)

- Generated by an algorithm that creates a small fixed-sized block
  - Depending on both message and some key
  - Like encryption, though need not be reversible
- Appended to message as a **signature**
- Receiver performs same computation on message and checks if it matches the MAC
- Provides assurance that message is unaltered and comes from sender

Digital Signatures

- Have looked at message authentication
  - But does not address issues of lack of trust
- Digital signatures provide the ability to
  - Verify author, date & time of signature
  - Authenticate message contents
  - Be verified by third parties to resolve disputes
- Hence include authentication function with additional capabilities

Digital Signature Properties

- Must depend on the message signed
- Must use information unique to sender
  - To prevent both forgery and denial
- Must be relatively easy to produce
- Must be relatively easy to recognize & verify
- Be computationally infeasible to forge
  - With new message for existing digital signature
  - With fraudulent digital signature for given message
- To be practical, save digital signature in storage

Direct Digital Signatures

- Involve only sender & receiver
- Assumes receiver has sender’s public-key
- Digital signature made by sender signing entire message or hash with private-key
- Can encrypt using receivers public-key
- Important that sign first then encrypt message & signature
- Security depends on sender’s private-key

Arbitrated Digital Signatures

- Involves use of arbiter A
  - Validates any signed message
  - Then dated and sent to recipient
- Requires suitable level of trust in arbiter
- Can be implemented with either private or public-key algorithms
- Arbiter may or may not see the message

Digital Signature Standard (DSS)

- U.S. Govt approved signature scheme FIPS 186
- Uses the SHA hash algorithm
- Designed by NIST & NSA in early 90's
- DSS is the standard, DSA is the algorithm
- A variant on ElGamal and Schnorr schemes
- Creates a 320 bit signature, but with 512-1024 bit security
- Security depends on difficulty of computing discrete logarithms

DSA Key Generation

- Have shared global public key values (p,q,g):
  - A large prime p such that $2^{L-1} < p < 2^L$
    - Where L= 512 to 1024 bits and is a multiple of 64
  - Choose q, a 160 bit prime factor of p-1
  - Choose $g = h^{(p-1)/q}$
    - Where $h<p-1$, $h^{(p-1)/q} \pmod{p} > 1$
- Users choose private & compute public key:
  - Choose x<q
  - Compute $y = g^x \pmod{p}$

DSA Signature Creation

- To **sign** a message `M` the sender:
  - Generates a random signature key `k`, `k<q`
  - `k` must be random, be destroyed after use, and never be reused
- Then computes signature pair:

  $r = (g^{k} (\mathrm{mod}\ p))(\mathrm{mod}\ q)$

  $s = (k^{-1}.SHA(M) + x.r)(\mathrm{mod}\ q)$
- Sends signature `(r,s)` with message `M`

DSA Signature Verification

- Having received M & signature `(r,s)`
- To **verify** a signature, recipient computes:

  $w = s^{-1}(\mathrm{mod}\ q)$

  $u1 = (SHA(M).w)(\mathrm{mod}\ q)$

  $u2 = (r.w)(\mathrm{mod}\ q)$

  $v = (g^{u1}.y^{u2}(\mathrm{mod}\ p))\ (\mathrm{mod}\ q)$
- If `v=r` then signature is verified

Program/Application Security

- Intrusion by misusing programs in clever ways to obtain unauthorized higher levels of privilege
- To prevent this a baseline behavior is established by observing the system call sequences
- This can be done by executing the program in isolation and generating huge amount of data to train the system
- Any malicious activity is captured through deviations from this baseline behavior
- Avoids attacks such as obtaining root privileges illegally, running malicious code, insider threats, etc.

Program/Application Security

- Advantages
  - Provides “true” end-to-end security
  - Flexibility
  - Protection against insider attacks
  - Secure audit trails
  - Mandate Use
- Disadvantages
  - Application dependence
  - Maintenance difficulties
  - Process speed degradation

Network Security

- The aim of network security is to protect networks from unauthorized modification, destruction, or disclosure, and provision of assurance that the network performs its critical functions correctly without harmful side-effects
- Detection based on packet patterns
- One of the methods for detecting intrusions is by using honeypots
- Avoids unauthorized access to network resources, Denial Of Service attacks, etc.

Network Security

- Advantages
  - Application independent
  - Reduced cost
  - Protection against external attack
  - Ease of upgrade and modification

Intruders

- Significant issue for networked systems is hostile or unwanted access
- Either via network or local
- Can identify classes of intruders:
  - Masquerader
  - Misfeasor
  - Clandestine user
- Varying levels of competence

Intruders

- Clearly a growing publicized problem
  - From "Wily Hacker" in 1986/87
  - Clearly escalating CERT stats
- May seem benign, but still cost resources
- May use compromised system to launch other attacks

Intrusion Techniques

- Aim to increase privileges on system
- Basic attack methodology
  - Target acquisition and information gathering
  - Initial access
  - Privilege escalation
  - Covering tracks
- Key goal often is to acquire passwords
- So then exercise access rights of owner

Password Guessing

- One of the most common attacks
- Attacker knows a login (from email/web page etc.)
- Then attempts to guess password for it
  - Try default passwords shipped with systems
  - Try all short passwords
  - Then try by searching dictionaries of common words
  - Intelligent searches try passwords associated with the user (variations on names, birthday, phone, common words/interests)
  - Before exhaustively searching all possible passwords
- Check by login attempt or against stolen password file
- Success depends on password chosen by user
- Surveys show many users choose poorly

Password Capture

- Another attack involves **password capture**
  - Watching over shoulder as password is entered
  - Using a Trojan horse program to collect
  - Monitoring an insecure network login (e.g., telnet, FTP, web, email)
  - Extracting recorded info after successful login (web history/cache, last number dialled, etc.)
- Using valid login/password can impersonate user
- Users need to be educated to use suitable precautions/countermeasures

Intrusion Detection

- Inevitably will have security failures
- So need also to detect intrusions so you can
  - Block if detected quickly
  - Act as deterrent
  - Collect info. to improve security
- Assume intruder will behave differently to a legitimate user
  - But will have imperfect distinction between

Approaches to Intrusion Detection

- Anomaly Detection Systems
  - Statistical Approaches
  - User Intent Identification
  - Predictive pattern generation
  - Neural Networks
- Misuse Detection Systems
  - Expert Systems
  - Signature Analysis
  - Colored Petri Nets

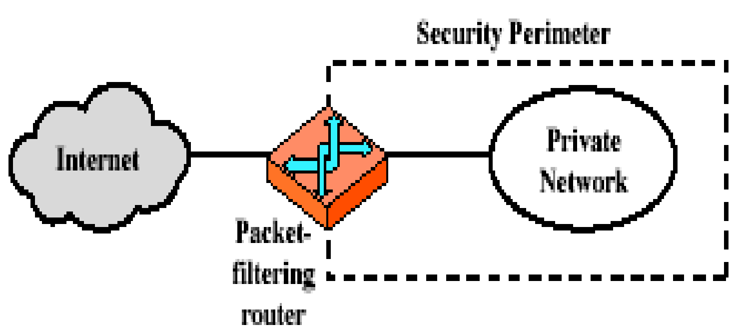
What is a Firewall?

- A **choke point** of control and monitoring
- Interconnects networks with differing trust
- Imposes restrictions on network services
  - Only authorized traffic is allowed
- Auditing and controlling access
  - Can implement alarms for abnormal behavior
- Is itself immune to penetration
- Provides **perimeter defense**

Firewall Limitations

- Cannot protect from attacks bypassing it
  - E.g., sneaker net, utility modems, trusted organizations, trusted services (e.g., SSL/SSH)
- Cannot protect against internal threats
  - E.g., disgruntled employee
- Cannot protect against transfer of all virus infected programs or files
  - Because of huge range of O/S & file types

Firewalls – Packet Filters



(a) Packet-filtering router

Firewalls – Packet Filters

- Simplest of components
- Foundation of any firewall system
- Examine each IP packet (not content) and permit or deny according to rules
- Hence restrict access to services (ports)
- Possible default policies
  - That not expressly permitted is prohibited
  - That not expressly prohibited is permitted

Attacks on Packet Filters

- IP address spoofing
  - Attack by spoofing trusted source addresses
  - Remedy: Configure filters to ignore external incoming packets with internal source IP addresses
- Source routing attacks
  - Source routing involves specifying the exact route of the packet in the network
  - Attacker sets a route other than default so as to attack a particular resource
  - Remedy: Block source routed packets until necessary

Attacks on Packet Filters

- Tiny fragment attacks
  - Split packet over several tiny packets (intentionally or due to underlying media requirements)
  - Generally packet filters reject the first packet and let others pass with assumption that without the first packet the whole message cannot be reassembled
  - To prevent an attack configure firewalls to keep a cache of recently seen first fragments and the filtering decision that was reached, and look up non-first fragments in this cache in order to apply the same decision