CSE331: Introduction to Networks and Security

Lecture 22
Fall 2006
Announcements

• Guest lecturer on Friday
  – Lecture will cover some ongoing research…
  – (Not part of the main course content)

• Homework 2 is due on Friday.

• Midterm 2 is next Friday, November 10th

• Project 3 has been assigned:
  – It's due November 20th
  – It covers basic cryptography / DES
Recap / Plan for today:

• Shared Key Cryptography
  – DES
  – AES, etc.

• Cryptographic Hashes
  – SHA1
  – MD5, etc.
Data Encryption Standard (DES)

• In 1973, National Bureau of Standards (now called NIST) issued a call for crypto algorithms:
  – Must provide a high level of security
  – Must be completely specified and easy to understand
  – Security of the algorithm must reside in the key; the security should not depend on secrecy of the algorithm
  – Must be available to all users
  – Must be adaptable for diverse applications
  – Must be economically implementable in electronic devices
  – Must be efficient
  – Must be validated
  – Must be exportable

• IBM was developing an algorithm called "Lucifer"
Data Encryption Standard (DES)

- Analyzed by the National Security Agency (NSA)
  - NSA reduced key length from 128 bits!
  - Workshops open for public debate (e.g. are there trapdoors?)
- Adopted as a standard in 1976
  - NBS published algorithm -- NSA admitted that certifying DES was a "big mistake" (they thought it was a hardware only spec)
- Key length is 56 bits
  - padded to 64 bits by using 8 parity bits
- Uses simple operators on (up to) 64 bit values
  - Simple to implement in software or hardware
  - Input is processed in 64 bit blocks
- Based on a series of 16 rounds
  - Each cycle uses permutation & substitution to combine plaintext with the key
DES Encryption
One Round of DES (f of previous slide)

- Expansion
- S-box
- Permuted choice of key
- Permutation
Types of Permutations in DES

- Permutation
- Permuted Choice
- Expansion Permutation
DES S-Boxes

• Substitution table
• 6 bits of input replaced by 4 bits of output
• Which substitution is applied depends on the input bits

• Implemented as a lookup table
  – 8 S-Boxes
  – Each S-Box has a table of 64 entries
  – Each entry specifies a 4-bit output
DES Decryption

- Use the same algorithm as encryption, but use $k_{16} \ldots k_1$ instead of $k_1 \ldots k_{16}$

- Proof that this works:
  - To obtain round $j$ from $j-1$:
    1. $L_j = R_{j-1}$
    2. $R_j = L_{j-1} \oplus f(R_{j-1}, k_j)$

  - Rewrite in terms of round $j-1$:
    1. $R_{j-1} = L_j$
    2. $L_{j-1} \oplus f(R_{j-1}, k_j) = R_j$
    $L_{j-1} \oplus f(R_{j-1}, k_j) \oplus f(R_{j-1}, k_j) = R_j \oplus f(R_{j-1}, k_j)$
    $L_{j-1} = R_j \oplus f(R_{j-1}, k_j)$
    $L_{j-1} = R_j \oplus f(L_j, k_j)$
Problems with DES

- Key length too short: 56 bits
  - [www.distributed.net](http://www.distributed.net) broke a DES challenge in 1999 in under 24 hours (parallel attack)

- Other problems
  - Bit-wise complementation of key produces bit-wise complemented ciphertext
  - Not all keys are good (specific patterns of roughly half 0’s and half 1’s)
  - Differential cryptanalysis: Carefully choose pairs of plaintext that differ in particular known ways (e.g. they are complements)
## Block Cipher Performance

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Key Length</th>
<th>Block Size</th>
<th>Rounds</th>
<th>Clks/Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twofish</td>
<td>variable</td>
<td>128</td>
<td>16</td>
<td>18.1</td>
</tr>
<tr>
<td>Blowfish</td>
<td>variable</td>
<td>64</td>
<td>16</td>
<td>19.8</td>
</tr>
<tr>
<td>Square</td>
<td>128</td>
<td>128</td>
<td>8</td>
<td>20.3</td>
</tr>
<tr>
<td>RC5-32/16</td>
<td>variable</td>
<td>64</td>
<td>32</td>
<td>24.8</td>
</tr>
<tr>
<td>CAST-128</td>
<td>128</td>
<td>64</td>
<td>16</td>
<td>29.5</td>
</tr>
<tr>
<td>DES</td>
<td>56</td>
<td>64</td>
<td>16</td>
<td>43</td>
</tr>
<tr>
<td>Serpent</td>
<td>128, 192, 256</td>
<td>128</td>
<td>32</td>
<td>45</td>
</tr>
<tr>
<td>SAFER (S)K-128</td>
<td>128</td>
<td>64</td>
<td>8</td>
<td>52</td>
</tr>
<tr>
<td>FEAL-32</td>
<td>64, 128</td>
<td>64</td>
<td>32</td>
<td>65</td>
</tr>
<tr>
<td>IDEA</td>
<td>128</td>
<td>64</td>
<td>8</td>
<td>74</td>
</tr>
<tr>
<td>Triple-DES</td>
<td>112</td>
<td>64</td>
<td>48</td>
<td>116</td>
</tr>
</tbody>
</table>
Advanced Encryption Standard (AES)

- National Institute of Standards & Technology (NIST)
  - Computer Security Research Center (CSRC)

- Uses the Rijndael algorithm
  - Invented by Belgium researchers Dr. Joan Daemen & Dr. Vincent Rijmen
  - Adopted May 26, 2002
  - Key length: 128, 192, or 256 bits
  - Block size: 128, 192, or 256 bits
Problems with Shared Key Crypto

• Compromised key means interceptors can decrypt any ciphertext they’ve acquired.
  – Change keys frequently to limit damage

• Distribution of keys is problematic
  – Keys must be transmitted securely
  – Use couriers?
  – Distribute in pieces over separate channels?

• Number of keys is $O(n^2)$ where $n$ is # of participants

• Potentially easier to break?
Hash Algorithms

- Take a variable length string
- Produce a fixed length digest
  - Typically 128-1024 bits

(Noncryptographic) Examples:
- Parity (or byte-wise XOR)
- CRC

Realistic Example
- The NIST Secure Hash Algorithm (SHA) takes a message of less than $2^{64}$ bits and produces a digest of 160 bits
Cryptographic Hashes

• Create a hard-to-invert summary of input data
• Useful for integrity properties
  – Sender computes the hash of the data, transmits data and hash
  – Receiver uses the same hash algorithm, checks the result
• Like a check-sum or error detection code
  – Uses a cryptographic algorithm internally
  – More expensive to compute
• Sometimes called a Message Digest
• Examples:
  – Secure Hash Algorithm (SHA)
  – Message Digest (MD4, MD5)
Uses of Hash Algorithms

- Hashes are used to protect *integrity* of data
  - Virus Scanners
  - Program fingerprinting in general
  - Modification Detection Codes (MDC)

- Message Authenticity Code (MAC)
  - Includes a cryptographic component
  - Send \((\text{msg}, \text{hash}(\text{msg}, \text{key}))\)
  - Attacker who doesn’t know the key can’t modify \text{msg} (or the hash)
  - Receiver who knows key can verify origin of message

- Make digital signatures more efficient
Desirable Properties

• The probability that a randomly chosen message maps to an n-bit hash should ideally be \((\frac{1}{2})^n\).
  – Attacker must spend a lot of effort to be able to modify the source message without altering the hash value

• Hash functions \(h\) for cryptographic use as MDC’s fall in one or both of the following classes.
  – **Collision Resistant Hash Function:** It should be computationally infeasible to find two distinct inputs that hash to a common value (i.e. \(h(x) = h(y)\)).
  – **One Way Hash Function:** Given a specific hash value \(y\), it should be computationally infeasible to find an input \(x\) such that \(h(x) = y\).
Secure Hash Algorithm (SHA)

- Pad message so it can be divided into 512-bit blocks, including a 64 bit value giving the length of the original message.
- Process each block as 16 32-bit words called $W(t)$ for $t$ from 0 to 15.
- Expand from these 16 words to 80 words by defining as follows for each $t$ from 16 to 79:
  - $W(t) := W(t-3) \oplus W(t-8) \oplus W(t-14) \oplus W(t-16)$
- Constants $H_0, \ldots, H_5$ are initialized to special constants
- Result is final contents of $H_0, \ldots, H_5$
for each 16-word block begin
  A := H0; B := H1; C := H2; D := H3; E := H4
  for I := 0 to 19 begin
    TEMP := S(5,A) + ((B \& C) \lor (\neg B \& D)) + E + W(I) + 5A827999;
    E := D; D := C; C := S(30,B); B := A; A := TEMP
  end
  for I := 20 to 39 begin
    TEMP := S(5,A) + (B \oplus C \oplus D) + E + W(I) + 6ED9EBA1;
    E := D; D := C; C := S(30,B); B := A; A := TEMP
  end
  for I := 40 to 59 begin
    TEMP := S(5,A) + ((B \& C) \lor (B \& D) \lor (C \& D)) + E + W(I) + 8F1BBCDC;
    E := D; D := C; C := S(30,B); B := A; A := TEMP
  end
  for I := 60 to 79 begin
    TEMP := S(5,A) + (B \oplus C \oplus D) + E + W(I) + CA62C1D6;
    E := D; D := C; C := S(30,B); B := A; A := TEMP
  end
  H0 := H0+A; H1 := H1+B; H2 := H2+C; H3 := H3+D; H4 := H4+E
end