SKIPJACK and Key Exchange Algorithm (KEA)

Carlton J. O’Riley
SKIPJACK Algorithm

- 64-bit Input/Output
- 80-bit Cryptovariable (Key)
- 32 Rounds
  - 8 Rounds of Rule A
  - 8 Rounds of Rule B
  - 8 Rounds of Rule A
  - 8 Rounds of Rule B
SKIPJACK Algorithm (Cont’d)

- G Permutation Function
  - 4 Round Feistel Structure
- F Byte Substitution Table
SKIPJACK Optimizations

• Precomputation of F-Table Key Values
  \[ F(\text{Key}[0][0..255]) \]
  \[ \ldots \]
  \[ F(\text{Key}[9][0\ldots255]) \]

• Requires 2560 bytes of a lookup table, but saves 4 XORs per round
SKIPJACK Optimizations (Cont’d)

• Not Shifting Words, But Operands
• Inline G Functions (not subroutines)
• Pointers To Buffer Data
• Compiler Optimizations
## SKIPJACK Optimizations (Cont’d)

<table>
<thead>
<tr>
<th></th>
<th>Encryption</th>
<th>Decryption</th>
</tr>
</thead>
<tbody>
<tr>
<td>SKIPJACK</td>
<td>34 ms</td>
<td>29 ms</td>
</tr>
<tr>
<td>DES</td>
<td>16 ms</td>
<td>16 ms</td>
</tr>
</tbody>
</table>
Key Exchange Algorithm
KEA Parameters

• Common Network Parameters
  – p: 1024-bit prime number
  – q: 160-bit prime divisor of p-1
  – g: 1024-bit base for exponentiation
• x: 160-bit Private Keys
• Y: 1024-bit Public Key (based on x)
• r: 160-bit Private Random Number
• R: 1024-bit Public Random Number (based on r)
Store and Forward KEA (Cont’d)

**Device A**
- \( p, q, g \) common to both devices
- \( x_A \) private key of each device
- \( Y_A = g^{x_A} \mod p \)
- Send \( Y_A \) in message
- \( Y_B \) A obtains B’s public from directory or local cache
- \( r_A \) A generates a random number
- \( R = g^{r_A} \mod p \)

**Device B**
- \( p, q, g \)
- \( x_B \)
- \( t_{AB} = (Y_B)^{x_A} \mod p \) check all values received
- \( u_{AB} = (Y_B)^{x_A} \mod p \) compute \( t = g^{e_A x_B} \mod p \)
- \( w = (t_{AB} + u_{AB}) \mod p \) compute \( u = g^{e_A x_B} \mod p \)
- \( v_1, v_2 \) extract \( v_1 \) and \( v_2 \) from \( w \)
- Key form Key from \( v_1, v_2, \) pad
- \( t_{BA} = (R_A)^{x_B} \mod p \)
- \( u_{BA} = (R_A)^{x_B} \mod p \)
- \( w = (t_{BA} + u_{BA}) \mod p \)
KEA Key Generation
FORTEZZA KEA
Wrapping the MEK

- Pad 80-bit MEK With Six Bytes of 06
- Encrypt Each 64-bits Using SKIPJACK With TEK As Key