Stream Ciphers

BIL 448/548
Internet Security Protocols
Ali Aydın Selçuk

Stream Ciphers

• Generate a \emph{pseudo-random} key stream \& xor to the plaintext.
• Key: The seed of the PRNG
• Traditional PRNGs (e.g. those used for simulations) are not secure.
  E.g., the linear congruential generator:
  \[ X_i = a \cdot X_{i-1} + b \mod m \]
  for some fixed \(a, b, m\).
• It passes the randomness tests, but it is predictable if previous output bytes are known.

Linear Feedback Shift Registers

• Feedback shift register:

  \[
  \begin{array}{cccc}
  X_0 & X_1 & \ldots & X_i \\
  \text{Feedback \ Fnc.} & & & \quad \text{output bits} \ldots \\
  \end{array}
  \]

  \begin{itemize}
  \item \text{“register”, “feedback”, “shift”}
  \end{itemize}

• LFSR: Feedback fnc. is linear over \(Z_2\) (i.e., an xor):

  \[
  \begin{array}{cccc}
  X_0 & X_1 & \ldots & X_i \\
  \quad \text{key stream} \\
  \end{array}
  \]

  \begin{itemize}
  \item Very compact \& efficient in hardware (e.g., SIM cards)
  \end{itemize}

Stream Ciphers from LFSRs

Desirable properties of \(f\):

\begin{itemize}
  \item high non-linearity
  \item long “cycle period” \((-2^{n_1+n_2+\ldots+n_k})\)
  \item low correlation with the input bits
\end{itemize}
The A5/1 stream cipher uses three LFSRs.
A register is clocked if its clocking bit (orange) agrees with one or both of the clocking bits of the other two registers. (majority match)

LFSRs are slow in software
Alternatives:
- Block ciphers (or hash functions) in CFB, OFB, CTR modes.
- Stream ciphers designed for software: RC4, SEAL, SALSA20, SOSEMANUK…

RC4
(Rivest, 1987)

Simple, byte-oriented, fast in s/w.
Popular: Google, MS-Windows, Apple, Oracle Secure SQL, WEP, WPA, etc.

Algorithm:
- Works on n-bit words. (typically, n = 8)
- State of the cipher: A permutation of {0,1,...,N-1}, where N = 2^n, stored at S[0,1,...,N-1].
- Key schedule: Expands the ℓ-byte key (typically 40-256 bits) into the initial state table S.

RC4 Key Schedule

The key schedule (i.e., initialization) algorithm:

```plaintext
// typically n = 8, ℓ = 16
for i = 1 to 2^n – 1 do:
    S[i] ← i
    i ← 0, j ← 0
for i = 1 to 2^n – 1 do:
    j ← j + S[i] + K[i mod ℓ]
    S[i] ↔ S[j]
```

Software-Oriented Stream Ciphers
**RC4 Encryption**

The encryption (i.e., the PRNG) algorithm:

```plaintext
i ← 0, j ← 0
loop: {
    i ← i + 1
    j ← j + S[i]
    S[i] ↔ S[j]
    output S[S[i] + S[j]]
}
```

**IV for Stream Ciphers**

- Use of an initialization vector is crucial in a stream cipher.
- Otherwise, the same stream will be produced each time the key is used (i.e., for each packet).
- The cipher may specify how to incorporate the IV. e.g., A5/1 mixes 22-bit frame no. into registers.
- Otherwise, ad hoc methods are used. e.g., WEP uses RC4 with 128-bit $K' = (IV || K)$ for a 24-bit IV and a 104-bit $K$.

**Speed of Software Stream Ciphers**

(Crypto++ 5.6 benchmarks, 2.2 GHz AMD Opteron 8354. March 2009.)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Speed (MiByte/s.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DES / CTR</td>
<td>17</td>
</tr>
<tr>
<td>AES-128 / CBC</td>
<td>148</td>
</tr>
<tr>
<td>AES-128 / CTR</td>
<td>198</td>
</tr>
<tr>
<td>RC4</td>
<td>124</td>
</tr>
<tr>
<td>SEAL</td>
<td>447</td>
</tr>
<tr>
<td>SOSEMANUK</td>
<td>767</td>
</tr>
<tr>
<td>SALSA20</td>
<td>953</td>
</tr>
</tbody>
</table>