Generic Attacks on Stream Ciphers

ERICSSON

KTH VETENSKAP OCH KONST

John Mattsson

Overview

- What is a stream cipher?
- Classification of attacks
- Different Attacks
  - Exhaustive Key Search
  - Time Memory Tradeoffs
  - Distinguishing Attacks
  - Guess-and-Determine attacks
  - Correlation Attacks
  - Algebraic Attacks
  - Sidechannel Attacks
- Summary

What is a stream cipher?

- **Input:** Secret key (*k* bits)
  Public IV (*v* bits).
- **Output:** Sequence z_1, z_2, … (keystream)

- The state (*s* bits) can informally be defined as the values of the set of variables that describes the current status of the cipher.
- For each new state, the cipher outputs some bits and then jumps to the next state where the process is repeated.
- The ciphertext is a function (usually XOR) of the keysteam and the plaintext.

Classification of attacks

- Assumed that the attacker has knowledge of the cryptographic algorithm but not the key.
- The aim of the attack
  - Key recovery
  - Prediction
  - Distinguishing
- The information available to the attacker.
  - Ciphertext-only
  - Known-plaintext
  - Chosen-plaintext
  - Chosen-chipertext

Exhaustive Key Search

- Can be used against any stream cipher. Given a keystream the attacker tries all different keys until the right one is found.
- If the key is k bits the attacker has to try 2^k keys in the worst case and 2^{k-1} keys on average.
- An attack with a higher computational complexity than exhaustive key search is not considered an attack at all.

Time Memory Tradeoffs (state)

- Large amounts of precomputed data is used to lower the computational complexity.
- Assume a key size of k bits and a state size of s bits. Generate keystream for 2^m different states and store them. Observe 2^d different keystreams. By the birthday paradox, we will on average be able to break one of these keystreams when

  $$m = d = s / 2.$$

  $\Rightarrow$ State size $\geq$ 2 * Key size
- **Example:** Attack on A5 used in GSM

Time Memory Tradeoffs (key/IV)

- Tradeoffs can work on key/IV pair instead of the state.
- Key size of k bits and an IV size of v bits. Generate keystream for 2^m different key/IV pairs and store them. Observe 2^d different keystreams. By the birthday paradox, we will be able to break one of these keystreams when

$$m = d = (k + v) / 2$$

$\Rightarrow$ IV size $\geq$ Key size

Distinguishing Attacks

- Method for distinguishing the keystream from a truly random sequence.
- A typical attack uses the fact that some part of the keystream, with a high probability, is a function of some other parts of the keystream.

$$z_i = f\,(z_{i-1}, z_{i-1}, \ldots, z_{i-n})$$

- **Example:** Attack on MAG (z_i = bytes)

$$z_{i+128} = z_i \oplus z_{i+127} \oplus z_{i+1} \oplus z_{i+2} \text{ with } p = 0.5$$

$$z_{i+128} = z_i \oplus z_{i+127} \oplus z_{i+1} \oplus {\sim}z_{i+2} \text{ with } p = 0.5$$

Generic Distinguishing Attacks

- Ordinary statistical tests were designed to evaluate PRNGs, only used for catching implementation errors.
  - Marsaglia's Diehard Battery of Tests
  - NIST Statistical Test Suite
- There exists generic distinguishing attacks on block ciphers in OFB or counter mode.
- More sofisticated generic distingushing attacks concentrate on the correlation between key, IV, and keystream.

Example: Saarinen's chosen-IV attack

- Able to distinguish 6/35 eStream candidates.
- The attack can be summarized as
  1. Choose *n* bits **x** = $(x^1, x^2, \ldots, x^n)$ in the IV as variables. The rest of the IV/key are given fixed values.
  2. Find the boolean function *f* from **x** to a single keystream bit (typically, the first).
  3. Check if the ANF (Algebraic Normal Form) expression of the Boolean function has the expected number of d-degree monomials. A monomial is a product of positive integer powers of a fixed sets of variables, for example, x^1, x^1x^3, or $x^2x^3x^7$.

Guess-and-Determine attacks

- Three steps
  1. Guess some parts of the key or state of the cipher.
  2. Determine other parts of the key/state under some assumption. The assumption is that the key/IV pair is of some subset of the total set that makes the cipher weak.
  3. By calculating keystream from the deduced values and compare with the known keystream we can check if the guess is right and the assumption holds.
- The attack is successful if

$$2^g \cdot (1/p) \cdot w < 2^k$$

- **Example:** My attack on Polar Bear.

Correlation Attacks

- For a correlation attack to be applicable, the keystream $z_1, z_2, \ldots$ must be correlated with the output sequence $a_1, a_2 \ldots$ of a much simpler internal device, such as a LFSR.
- The two sequences are correlated if the probalility $P(z_i = a_i) \neq 0.5$

Basic Correlation Attack

- Nonlinear combination generator with n LFSRs.
- For each possible initial state $\mathbf{u}_0 = (u_1, u_2, \ldots, u_l)$ an output sequence **a** of length N is generated. Define $\beta = N - d_H(\mathbf{a}, \mathbf{z})$.
- If we run through all 2^l possible initial states and if N is large enough, β will with high probability take its largest value when $\mathbf{u}_0$ is the correct initial state.
- Computational complexity is reduced from $\Pi_{i=1..n}(2^{l_i})$ to $\Sigma_{i=1..n}(2^{l_i})$ where l_i is the length of LFSR i.
- Applicable when the length of the shift registers are small and when the combining function leaks information about individual input variables.

Fast Correlation Attack

- Significantly faster than exhaustive search over the target LFSR, but requires received sequences of large length.
- Use certain parity check equations that are created from the feedback polynomial.
- Two phases
  - In the first, a set of parity check equations are found.
  - In the second these equations are used in a decoding algorithm to recover the transmitted codeword (the internal output sequence).

First phase

- Suppose that the feedback polynomial g(x) has *t* non-zero coefficients.

$$g(x) = 1 + c_1x + c_2x^2 + \ldots + c_lx^l$$

- From this we get *t* different parity check equations for the digit a_i. And by noting that

$$g(x)^{2k} = 1 + c_1x^{2k} + c_2x^{2k+1} + \ldots + c_lx^{l*2k}$$

  we get t more for each squaring.
- The total number of check equations that can be obtained by squaring the feedback polynomial is

$$m \approx t * \log(N / 2l)$$

Second phase

- The *m* parity check equations can be written as

$$a_i + s_j = 0 \quad j = 1..m$$

- If we substitute a_i with z_i we get the following expressions.

$$z_i + y_j = L_j \quad j = 1..m$$

- By counting the number of equations that hold we can calculate the probability

$$p^* = P(z_i = u_i \mid h \text{ equations hold})$$

- p^* is calculated for each observed symbol and the *l* positions with highest value of p^* are used to find the correct initial state

Example: Geffe's generator

- The combining function used in the Geffe's generator

$$f(x_1, x_2, x_3) = x_3 \oplus x_1x_2 \oplus x_2x_3$$

  is vulnerable to correlation attacks because

$$P(f(\mathbf{x}) = x_1) = P(f(\mathbf{x}) = x_3) = 0.75$$

Solution: Correlation immune combining function.

- But, there is a tradeoff between the correlation immunity m and the nonlinear order k. A m-th order correlation immune function can have at most nonlinear order $n - m$.

Algebraic Attacks

- Principle
  1. Find system of equations in keystream bits z_i and the unknown key bits k_i.
  2. Reduce the degree of the equations. (*fast algebraic attacks*)
  3. Insert the observed keystream bits z_i.
  4. Recover the key by solving the system of equations
- Have been used to attack for example: Toyocrypt, E0 (used in bluetooth), and a modified Snow

Finding Equations

- For a pure combiner we have that $z_i = f(\mathbf{x}_i)$ But $\mathbf{x}_i$ is a linear function of the secret key $\mathbf{k}$ (applied i times).
- So $z_i = f(L^t(\mathbf{k}))$ and our equation system is

$$z_i \oplus f(L^i(\mathbf{k})) = 0 \text{ for every } i$$

- For combiners with memory (E0) it is possible to cancel out the memory bits at the cost of more keystream.
- More output at a time gives equations of substantially lower degree $\Rightarrow$ much faster attacks.

Equation solving - Linearization (XL, XSL…)

- Use a over defined system of equations.
- Replace each monomial with a new variable.
- Solve as a linear system.

$$\begin{array}{rcr} x + y + z = 0 & & x + y + z = 0 \\ xyz + xy + z = 0 & \rightarrow & u + t + z = 0 \\ y + xyz = 0 & & y + u = 0 \end{array}$$

- But this is NP-complete in general case. Complexity $O(n^{3d})$ where d is the maximum degree of the equations, $d \leq n$
- Another option is Gröbner bases, but difficult to predict complexity

Sidechannel Attacks

- Uses information from the physical implementation instead of theoretic weaknesses
- Any information that can be measured and is dependant on the key, state or plaintext can potentially be used in a sidechannel attack.
- Examples of Sidechannel attacks are
  - Timing analysis
  - Power analysis
  - Electromagnetic radiation
  - Acoustic analysis

Summary

- Large number of different attacks to consider when designing stream ciphers.
- Most stream cipher proposals are broken, at least theoretical, (Distinguishing in $O(2^{100})$ time)
- Implementation is important.