Shared Stream Cipher
Encryption/Decryption
Magnus Öberg
Linköping University

Secret sharing is well known as a tool for storing secret information in a distributed fashion. Cipher systems are used in many instances to transmit secret data, authenticate data or provide integrity control. Here we combine these two areas by describing a system for shared encryption/decryption using stream ciphers.

Introduction

Unless you are doing encryption or decryption by hand, you have to trust the machine that makes the calculations for you. You have to trust that the output is correct and that no information have leaked to someone else. Are you sure that no one is eavesdropping on you when you feed the plaintext and encryption key into the machine?

Secret Sharing

Secret sharing deals with situations when one wants to store a secret, but don’t want to trust any single person. This is done by giving pieces of information, called shares or shadows, to a group of people. The pieces are created so that a certain number of pieces are needed in order to reconstruct the secret.

A simple secret sharing scheme is Shamir’s original scheme [1] from 1979, which is a so called threshold scheme where any subset of \( k \) or more (out of \( n \)) shares are enough to reconstruct the secret.

Shamir’s scheme has been shown to be equivalent to a Reed-Solomon encoding scheme and is constructed as follows. Let \( s \) be the secret, \( k \) be the threshold and \( n \) be the total number of shares. Then create a polynomial

\[
f(x) = s + a_1 \cdot x + a_2 \cdot x^2 + \ldots + a_{k-1} \cdot x^{k-1}
\]

over \( GF(q), q > n \) where \( a_i \) is coefficients drawn randomly from \( GF(q) \). Let \( \alpha \) be a primitive element. The shares are now the pairs \((i, f(\alpha^i))\).

Stream Ciphers

A stream cipher divides the plaintext into small entities called characters and encrypts one character at a time using a time-varying transformation. One common construction is to use a key stream generator and add the output of this to the plaintext at each instance, as in fig 1. The security of such a system depends on whether or not it is possible to predict the upcoming key characters.

![Figure 1: A common type of stream cipher.](image)

The key stream can be generated in many ways but since it has been proven that any finite sequence can be generated by a linear feedback shift register (LFSR) it is interesting to study these systems. Sequences produced by a LFSR obey the recursion formula:

\[
s_i = \sum_{m=1}^{L} c_m s_{i-m}
\]

Here \( L \) is the memory size of the register and \( c_i \) is the coefficient of tap \( i \). Commonly binary systems are considered but the formula is valid in any finite field \( GF(q) \).

Shared Encryption/Decryption

This is a simple scheme for shared encryption/decryption of a stream cipher based on a single linear feedback shift register (LFSR). Of course this is not practical since the shift register have to be very long to be secure, but since any other key stream generator can be transformed into an equivalent generator based on a single LFSR this scheme has interesting properties.

What we want to do is to replace the single cryptographic device with a group of devices that, without communication, transforms shares of the plaintext into shares of the ciphertext. The idea is that each letter in the plaintext is shared with normal secret sharing. The key stream sequence is shared using a modified scheme. The modification applies when the random coefficients are to be drawn. We then let these coefficients be generated by \((k-1)\) LFSR’s. All with the same recursion polynomial as the key stream generator, but different initial states. For each symbol in the key sequence we take one symbol from each of the LFSR’s sequences to construct the sharing polynomial. Then the shares are constructed as usual.

As a result of our construction each machine receives a share of one plaintext symbol and a share of one key symbol. With the encryption function just being an addition of plaintext and key, and with RS-codes being linear we see that adding the shares yields a share of a ciphertext symbol. This without any communication between the machines.

References