Semigroups in Cryptography

Ariel Clemmons, Shikha Kinariwala, Alexander Lee, and Julia Snyder
Allison Kohne, Scott Carson

Mason Experimental Geometry Lab

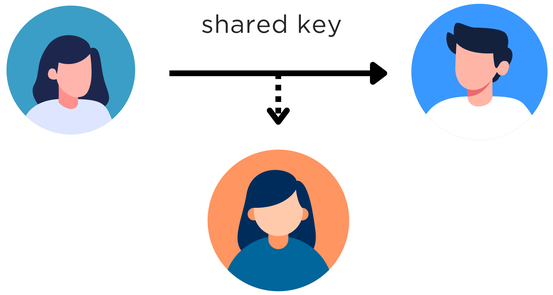
GEORGE MASON UNIVERSITY

Introduction

Most modern cryptosystems (such as Diffie-Hellman and RSA) rely on group theory, where every element has an inverse. By studying semigroups, a cryptosystem that is more general, harder to invert, and resistant to attacks which rely on exploiting inverses might be possible to develop. We explore and implement a proposed cryptosystem that utilizes simple semiring actions and matrix operations.

Diffie-Hellman Key Exchange

The Diffie-Hellman Key Exchange is a method which allows two parties to create a shared secret key over an insecure channel without sending the actual key.

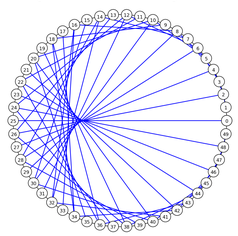


Diffie-Hellman is based on the difficulty of solving the Discrete Logarithm Problem.

Discrete Logarithm Problem

Given a base **g**, a modulus **n**, and a result **h**, find an **x** such that

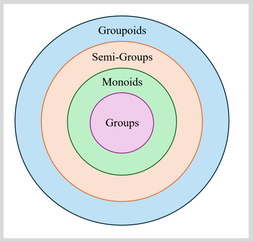
$$g^x \pmod{\mathrm{n}} = h$$



Discrete logarithm (base 2, modulo 50)

Semigroup & Semigroup Actions

Semigroups are sets which have an associative operation, and they do not require inverses or identity elements. A semigroup action is a way that elements of a semigroup can act on another set.



Semigroup Action Problem

Given a semigroup ***S***, a set ***X***, and an action $\cdot$ such that

$$S \times X \to X,$$

and two elements

$$x \in X \text{ and } y = s \cdot x,$$

find the semigroup element **s**.

Proposed Cryptosystem (Protocol 5.1 in [1])

1. Alice and Bob agree on a finite semiring ***R*** with a center **C**, then choose a matrix (size **n**) publicly.
2. Alice privately selects two polynomials, then sends her public message, ***A***.
   $p_a, q_a \in C[t]$ $A = p_a(M_1) \cdot S \cdot q_a(M_2)$
3. Bob privately picks two polynomials and sends ***B***.
   $p_b, q_b \in C[t]$ $B = p_b(M_1) \cdot S \cdot q_b(M_2).$
4. Alice and Bob compute their shared secret key, where
   $p_a(M_1)Bq_a(M_2) = p_a(M_1)p_b(M_1)Sq_b(M_2)q_a(M_2) = p_b(M_1)Aq_b(M_2).$

Implementation

//PRE-PROCESSING
>> get user input
  >> format values
  >> store values in a map
  >> assign matrices

//ENCRYPTION
>> read polynomial
  >> execute addition or multiplication operations
>> store polynomial result in p or q matrix
>> compute S matrix using p and q such that

$$X = p(M_1) \cdot S \cdot q(M_2)$$

>> return X

Moving Forward

We want to continue implementing our proposed implementation scheme into a working program, as well as develop our decryption scheme.

Currently, we have implemented a working example into a small program.

The proposed cryptosystem does not include decryption, so we will begin to develop our own decryption protocol, write that in our implementation above, then implement it within our working program.

Reference

[1] A. Grigoriev and V. Nikitin, "Public-key cryptography based on semigroup actions," arXiv:0501.017v4, 2015.