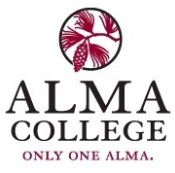
Generalizing the WHIRLPOOL Hash Function

D. Albertson, L. Babinkostova, H. de Kergorlay, A. Hegedus,
D. Nikolov, L. Wells

July 31, 2013



What is a hash function?

What is a hash function?

A hash function takes an arbitrary-length string (message) as input and gives back a fixed-length string (hash) as output.

What is a hash function?

A hash function takes an arbitrary-length string (message) as input and gives back a fixed-length string (hash) as output.
What makes a good hash function?

What is a hash function?

A hash function takes an arbitrary-length string (message) as input and gives back a fixed-length string (hash) as output.
What makes a good hash function?

- Given a hash h, it should be hard to find a message m whose hash is h.

What is a hash function?

A hash function takes an arbitrary-length string (message) as input and gives back a fixed-length string (hash) as output.
What makes a good hash function?

- Given a hash h, it should be hard to find a message m whose hash is h.
- Given a message m_1, it should be hard to find a message m_2 that has the same hash as m_1.

What is a hash function?

A hash function takes an arbitrary-length string (message) as input and gives back a fixed-length string (hash) as output.
What makes a good hash function?

- Given a hash h, it should be hard to find a message m whose hash is h.
- Given a message m_1, it should be hard to find a message m_2 that has the same hash as m_1.
- It should be hard to find *any* two messages m_1 and m_2 that have the same hash.

What are hash functions used for?

What are hash functions used for?

- **Authentication**
  - Password Storage

What are hash functions used for?

- **Authentication**
  - Password Storage
- **Data Integrity**
  - File integrity verification

What are hash functions used for?

- **Authentication**
  - Password Storage
- **Data Integrity**
  - File integrity verification
- **Non-repudiation**
  - Public Key Fingerprint

What are hash functions used for?

- **Authentication**
  - Password Storage
- **Data Integrity**
  - File integrity verification
- **Non-repudiation**
  - Public Key Fingerprint

Practical Hash Functions: MD4, MD5, SHA-1, SHA-2, WHIRLPOOL.

How Whirlpool Works - I

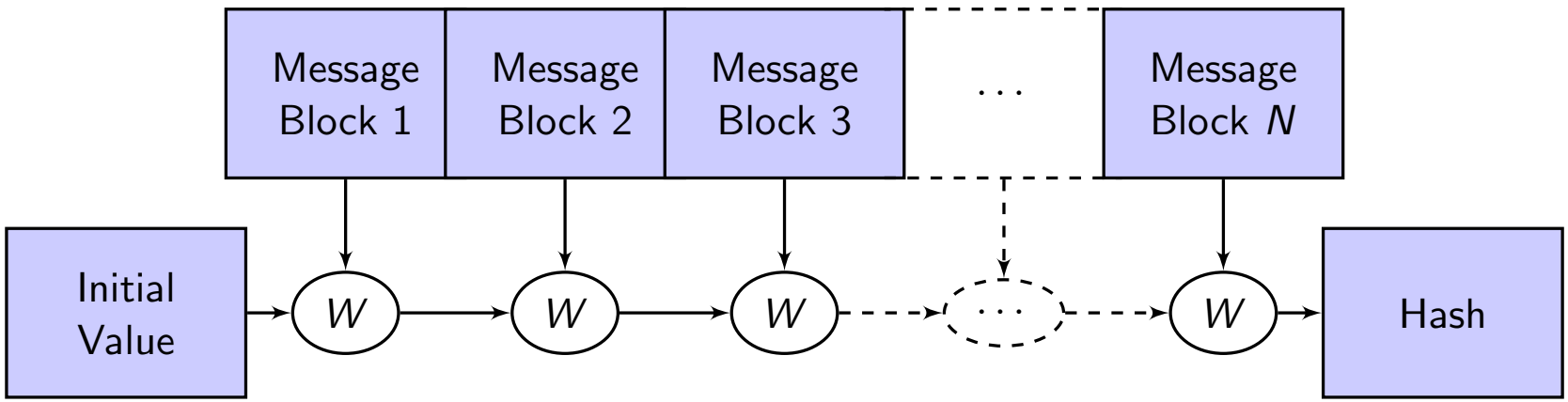


Figure : The Merkle-Damgård Construction

How Whirlpool Works - II

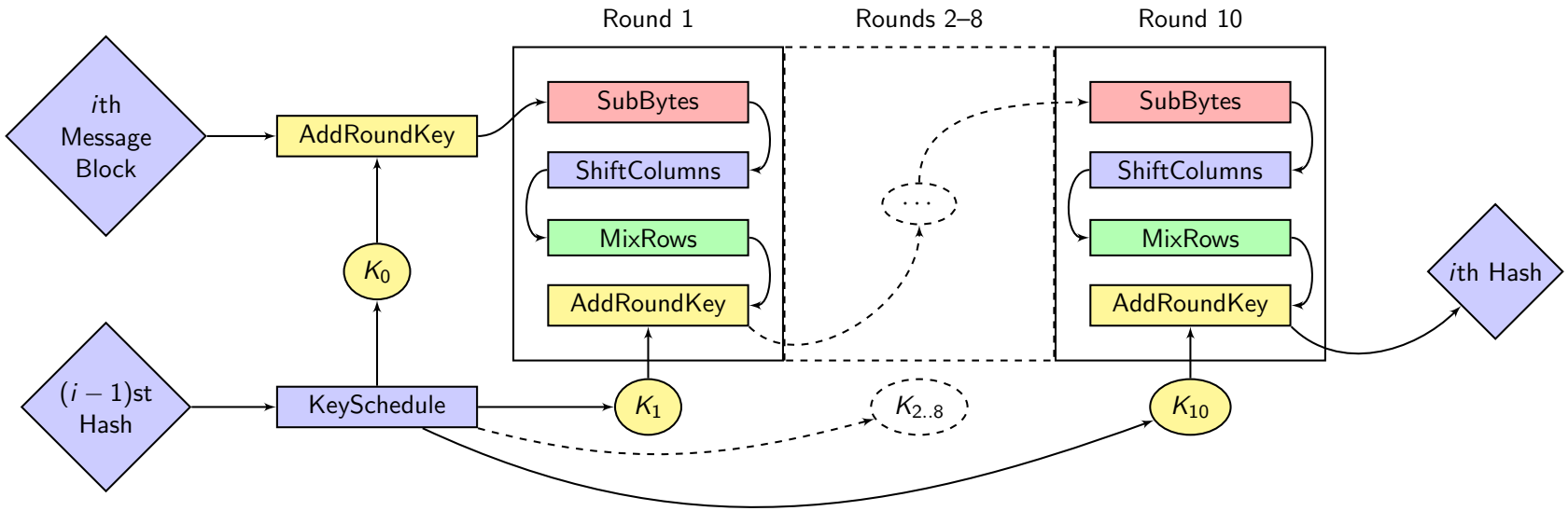


Figure : The Block Function W

The Security of Whirlpool

The Security of Whirlpool

If a hash function's behavior is predictable, then it is open to attack (forgery, message recovery, etc.).

The Security of Whirlpool

If a hash function's behavior is predictable, then it is open to attack (forgery, message recovery, etc.).The predictability of a hash function is closely related to its algebraic structure.

The Security of Whirlpool

If a hash function's behavior is predictable, then it is open to attack (forgery, message recovery, etc.).The predictability of a hash function is closely related to its algebraic structure.

- Since the set of all possible message blocks is the same as the set of all possible hashes, the Whirlpool hash function on one block is a *permutation*.

The Security of Whirlpool

If a hash function's behavior is predictable, then it is open to attack (forgery, message recovery, etc.).The predictability of a hash function is closely related to its algebraic structure.

- Since the set of all possible message blocks is the same as the set of all possible hashes, the Whirlpool hash function on one block is a *permutation*.
- Permutations may form *groups*. (bad)

The Security of Whirlpool

If a hash function's behavior is predictable, then it is open to attack (forgery, message recovery, etc.).The predictability of a hash function is closely related to its algebraic structure.

- Since the set of all possible message blocks is the same as the set of all possible hashes, the Whirlpool hash function on one block is a *permutation*.
- Permutations may form *groups*. (bad)
- Permutations *generate* groups. (bigger is better)

The Security of Whirlpool

If a hash function's behavior is predictable, then it is open to attack (forgery, message recovery, etc.).The predictability of a hash function is closely related to its algebraic structure.

- Since the set of all possible message blocks is the same as the set of all possible hashes, the Whirlpool hash function on one block is a *permutation*.
- Permutations may form *groups*. (bad)
- Permutations *generate* groups. (bigger is better)

To evaluate the security of Whirlpool, we need to know whether it *forms* a group, and what the size of the group that it *generates* is.

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)
- (odd) $\circ$ (odd) = (even)

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)
- (odd) $\circ$ (odd) = (even)
- (even) $\circ$ (odd) = (odd)

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)
- (odd) $\circ$ (odd) = (even)
- (even) $\circ$ (odd) = (odd)

Two possible groups Whirlpool could generate:

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)
- (odd) $\circ$ (odd) = (even)
- (even) $\circ$ (odd) = (odd)

Two possible groups Whirlpool could generate:

- Alternating Group (all even)

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)
- (odd) $\circ$ (odd) = (even)
- (even) $\circ$ (odd) = (odd)

Two possible groups Whirlpool could generate:

- Alternating Group (all even)
- Symmetric Group (half even, half odd)

The Security of Whirlpool

Since Whirlpool is a permutation, we look at its *parity*.

- (even) $\circ$ (even) = (even)
- (odd) $\circ$ (odd) = (even)
- (even) $\circ$ (odd) = (odd)

Two possible groups Whirlpool could generate:

- Alternating Group (all even)
- Symmetric Group (half even, half odd)

To find the parity of the complete Whirlpool function, we look at the parity of each sub-function.

Generalizing Whirlpool

Whirlpool depends on many parameters, which in classical Whirlpool are fixed.

Generalizing Whirlpool

Whirlpool depends on many parameters, which in classical Whirlpool are fixed. When we generalize Whirlpool, we find out what happens when these parameters are changed.

Generalizing Whirlpool

Whirlpool depends on many parameters, which in classical Whirlpool are fixed. When we generalize Whirlpool, we find out what happens when these parameters are changed. So by generalizing, we will know if small changes to the parameters can lead to large increases (or decreases) in security.

AddRoundKey ($\sigma[k]$) function

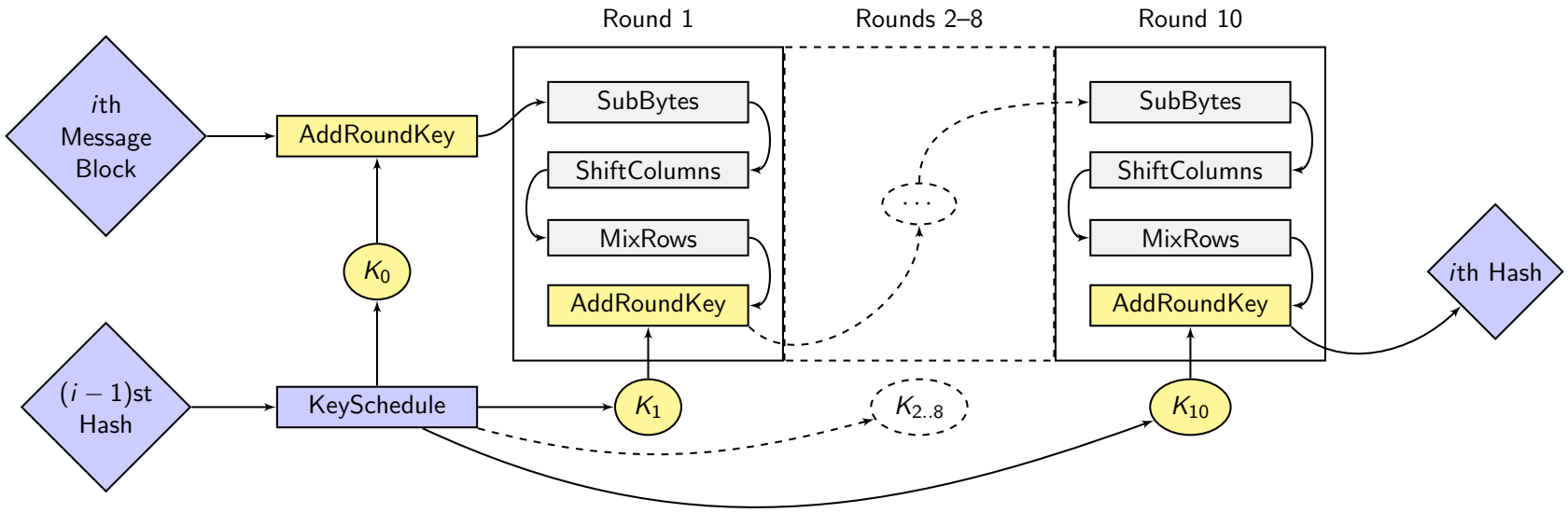


Figure : AddRoundKey

AddRoundKey ($\sigma[k]$) function

AddRoundKey ($\sigma[k]$) function

Definition

Let $\sigma[k] : M_{m,n}(\mathrm{GF}(p^r)) \rightarrow M_{m,n}(\mathrm{GF}(p^r))$

AddRoundKey ($\sigma[k]$) function

Definition

Let $\sigma[k] : M_{m,n}(\mathrm{GF}(p^r)) \to M_{m,n}(\mathrm{GF}(p^r))$ denote the mapping defined by $\sigma[k](a) = a'$ if and only if $a'_{i,j} = a_{i,j} + k_{i,j}$

AddRoundKey ($\sigma[k]$) function

Definition

Let $\sigma[k] : M_{m,n}(\mathrm{GF}(p^r)) \to M_{m,n}(\mathrm{GF}(p^r))$ denote the mapping defined by $\sigma[k](a) = a'$ if and only if $a'_{i,j} = a_{i,j} + k_{i,j}$ and $k \in M_{m,n}(\mathrm{GF}(p^r))$ for all $0 \leq i < m$, $0 \leq j < n$.

AddRoundKey ($\sigma[k]$) function

Definition

Let $\sigma[k] : M_{m,n}(\mathrm{GF}(p^r)) \to M_{m,n}(\mathrm{GF}(p^r))$ denote the mapping defined by $\sigma[k](a) = a'$ if and only if $a'_{i,j} = a_{i,j} + k_{i,j}$ and $k \in M_{m,n}(\mathrm{GF}(p^r))$ for all $0 \leq i < m$, $0 \leq j < n$. The function $\sigma[k]$ is called *Add Round Key* function.

AddRoundKey ($\sigma[k]$) function

Definition

Let $\sigma[k] : M_{m,n}(\mathrm{GF}(p^r)) \to M_{m,n}(\mathrm{GF}(p^r))$ denote the mapping defined by $\sigma[k](a) = a'$ if and only if $a'_{i,j} = a_{i,j} + k_{i,j}$ and $k \in M_{m,n}(\mathrm{GF}(p^r))$ for all $0 \leq i < m$, $0 \leq j < n$. The function $\sigma[k]$ is called *Add Round Key* function.

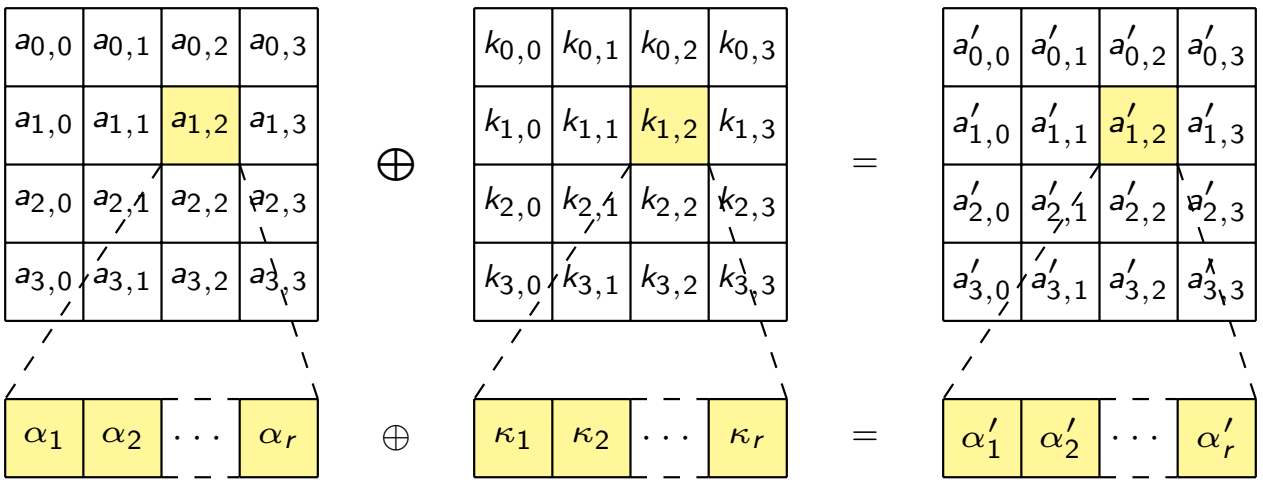


Figure : Example AddRoundKey Function

Parity of Add Round Key ($\sigma[k]$) function

Lemma

For all $k \in M_{m,n}(\mathrm{GF}(p^r))$ and $r, m, n > 1$, the function $\sigma[k]$ is an even permutation. [a]

---

[a] L. Babinkostova, K. Bombardier, M. Cole, T. Morrell, and C. Scott, *Algebraic Structure of generalized Rijndael-like SP networks*, (2012) (arXiv: 1210.7942)

ShiftColumn (π) Function

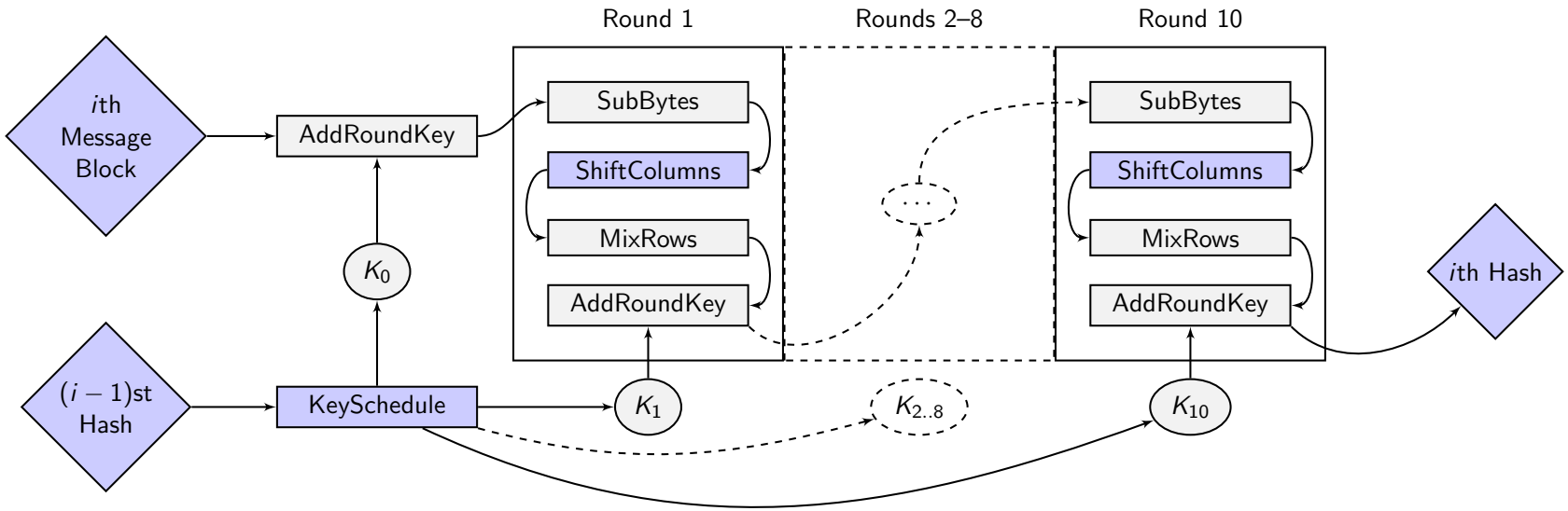


Figure : ShiftColumn

ShiftColumn (π) Function

ShiftColumn (π) Function

Definition

Let $\pi : M_{m,n}(\mathrm{GF}(p^r)) \rightarrow M_{m,n}(\mathrm{GF}(p^r))$ denote the mapping for which there is a function $s : \{0, \ldots, n-1\} \rightarrow \{0, \ldots, m-1\}$ such that $\pi(a) = a'$ if and only if $a'_{i,j} = a_{(i+s(j)) \bmod m,\ j}$ for all $0 \leq i < m$, $0 \leq j < n$. The mapping π is called a "Shift Column" function.

ShiftColumn (π) Function

Definition

Let $\pi : M_{m,n}(\mathrm{GF}(p^r)) \rightarrow M_{m,n}(\mathrm{GF}(p^r))$ denote the mapping for which there is a function $s : \{0, \ldots, n-1\} \rightarrow \{0, \ldots, m-1\}$ such that $\pi(a) = a'$ if and only if $a'_{i,j} = a_{(i+s(j)) \bmod m,\ j}$ for all $0 \leq i < m$, $0 \leq j < n$. The mapping π is called a "Shift Column" function.

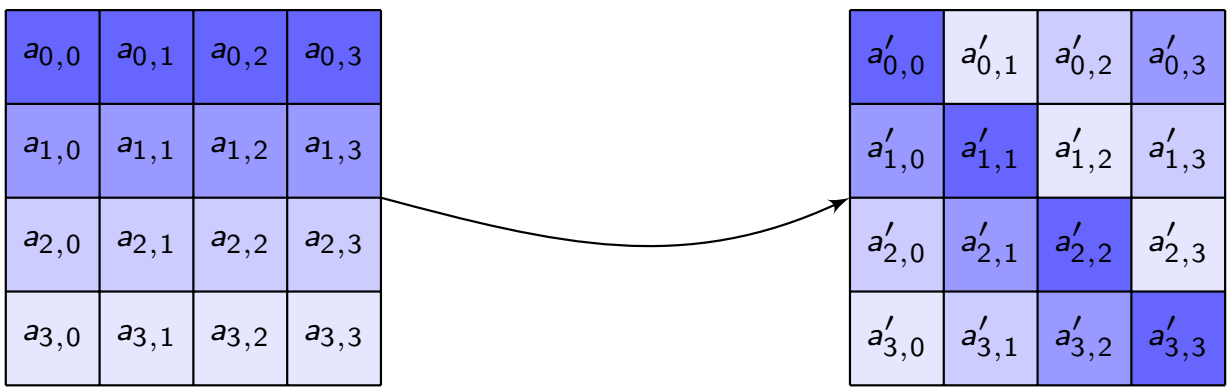


Figure : Example ShiftColumn Function

Counting Cycles of the WHIRLPOOL "Shift Column"

Counting Cycles of the WHIRLPOOL "Shift Column"

- In order to determine the parity of the ShiftColumn function, we count the cycles.

Counting Cycles of the WHIRLPOOL "Shift Column"

- In order to determine the parity of the ShiftColumn function, we count the cycles.
- Generalizing for an arbitrary matrix with arbitrary shifts $\rightarrow$ look at each column separately

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

- $n(c)$ = number of configurations in a c-cycle

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

- $n(c)$ = number of configurations in a c-cycle
- $N(c)$ = number of configurations in a c-cycle, overcounting every configuration in a d-cycle, where $d|c$.

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

- $n(c)$ = number of configurations in a c-cycle
- $N(c)$ = number of configurations in a c-cycle, overcounting every configuration in a d-cycle, where $d|c$.
- $\frac{n(c)}{c}$ = number of c-cycles.

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

- $n(c)$ = number of configurations in a c-cycle
- $N(c)$ = number of configurations in a c-cycle, overcounting every configuration in a d-cycle, where $d|c$.
- $\frac{n(c)}{c}$ = number of c-cycles.

$$N(c) = (p^r)^{c \cdot gcd(m,s(j))} \cdot (p^r)^{n(m-1)}$$

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

- $n(c)$ = number of configurations in a c-cycle
- $N(c)$ = number of configurations in a c-cycle, overcounting every configuration in a d-cycle, where $d|c$.
- $\frac{n(c)}{c}$ = number of c-cycles.

$$N(c) = (p^r)^{c \cdot gcd(m, s(j))} \cdot (p^r)^{n(m-1)}$$

$$n(c) = N(c) - \sum_{d||c} n(d)$$

Counting Cycles of the WHIRLPOOL "Shift Column"

Some definitions:

- $n(c)$ = number of configurations in a c-cycle
- $N(c)$ = number of configurations in a c-cycle, overcounting every configuration in a d-cycle, where $d|c$.
- $\frac{n(c)}{c}$ = number of c-cycles.

$$N(c) = (p^r)^{c \cdot gcd(m, s(j))} \cdot (p^r)^{n(m-1)}$$

$$n(c) = N(c) - \sum_{d||c} n(d)$$

$$n\left(2^k \prod_{l=1}^{N} p_\ell^{k_\ell}\right) = \sum_{\substack{i_0 \in \{0,\ldots,k-1\} \\ i_1 \in \{0,\ldots,k_1-1\} \\ \vdots \\ i_N \in \{0,\ldots,k_N-1\}}} (-1)^{\sum_{\ell=0}^{N} i_\ell} N\left(2^{k-i_0} \prod_{l=1}^{N} p_\ell^{k_\ell - i_\ell}\right)$$

Counting Cycles of the WHIRLPOOL "Shift Column"

The Möbius Inversion Theorem

If g and f are arithmetic functions, $g(c) = \sum_{d|c} f(d)$ for every integer $c > 1$, then $f(c) = \sum_{d|c} \mu(d) \cdot g\left(\frac{c}{d}\right)$ for every integer $c > 1$.

Counting Cycles of the WHIRLPOOL "Shift Column"

The Möbius Inversion Theorem

If g and f are arithmetic functions, $g(c) = \sum_{d|c} f(d)$ for every integer $c > 1$, then $f(c) = \sum_{d|c} \mu(d) \cdot g\left(\frac{c}{d}\right)$ for every integer $c > 1$.

Then, since $N(c) = \sum_{d|c} n(d)$, by the Möbius inversion theorem,

$$n(c) = \sum_{d|c} \mu(d) N\left(\frac{c}{d}\right)$$

Counting Cycles of the WHIRLPOOL "Shift Column"

The Möbius Inversion Theorem

If g and f are arithmetic functions, $g(c) = \sum_{d|c} f(d)$ for every integer $c > 1$, then $f(c) = \sum_{d|c} \mu(d) \cdot g\left(\frac{c}{d}\right)$ for every integer $c > 1$.

Then, since $N(c) = \sum_{d|c} n(d)$, by the Möbius inversion theorem,

$$n(c) = \sum_{d|c} \mu(d) N\left(\frac{c}{d}\right)$$

- $\mu(d) = 1$ if d is a square-free positive integer with an even number of prime factors.
- $\mu(d) = -1$ if d is a square-free positive integer with an odd number of prime factors.
- $\mu(d) = 0$ if d has a squared prime factor.

How the Möbius function simplifies

Without Möbius Inversion Formula:

$$n(72) = N(72) - N(36) - N(24) + N(18) + N(12) + N(8) \\ -N(9) - N(6) - N(4) + N(3) + N(2) - N(1)$$

How the Möbius function simplifies

Without Möbius Inversion Formula:

$$n(72) = N(72) - N(36) - N(24) + N(18) + N(12) + N(8) \\ -N(9) - N(6) - N(4) + N(3) + N(2) - N(1)$$

With Möbius Inversion Formula:

$$n(72) = N(72) - N(36) - N(24) + N(12)$$

Alternative ways of thinking about Shift Column

- The number of aperiodic necklaces that can be formed by arranging c beads, each of which can be one of p different colours.

Alternative ways of thinking about Shift Column

- The number of aperiodic necklaces that can be formed by arranging c beads, each of which can be one of p different colours.
  Moreau's necklace counting-function:

Alternative ways of thinking about Shift Column

- The number of aperiodic necklaces that can be formed by arranging c beads, each of which can be one of p different colours.
  Moreau's necklace counting-function:

$$M(p, c) = \frac{1}{c} \sum_{d|c} \mu\left(\frac{n}{c}\right) p^d$$

Alternative ways of thinking about Shift Column

- The number of aperiodic necklaces that can be formed by arranging c beads, each of which can be one of p different colours.
  Moreau's necklace counting-function:

$$M(p, c) = \frac{1}{c} \sum_{d|c} \mu\left(\frac{n}{c}\right) p^d$$

- The number of Lyndon words of a given length c, from an alphabet of p letters.

Alternative ways of thinking about Shift Column

- The number of aperiodic necklaces that can be formed by arranging c beads, each of which can be one of p different colours.
  Moreau's necklace counting-function:

$$M(p, c) = \frac{1}{c} \sum_{d|c} \mu\left(\frac{n}{c}\right) p^d$$

- The number of Lyndon words of a given length c, from an alphabet of p letters.
- The number of monic irreducible polynomials of degree c in $\mathrm{GF}(p)$

The connection with monic irreducible polynomials

The connection with monic irreducible polynomials

Preliminary results on finite fields

The connection with monic irreducible polynomials

Preliminary results on finite fields

Theorem

For each divisor c of m, $\mathrm{GF}(p^m)$ has a unique subfield of order p^c. Moreover, these are the only subfields of $\mathrm{GF}(p^m)$.[a]

The connection with monic irreducible polynomials

Preliminary results on finite fields

Theorem

For each divisor c of m, $\mathrm{GF}(p^m)$ has a unique subfield of order p^c. Moreover, these are the only subfields of $\mathrm{GF}(p^m)$.[a]

Theorem

If $p(x)$ is an irreducible polynomial over a finite field $\mathrm{GF}(p^m)$, then $p(x)$ has no multiple roots.[a]

[a] J. A. Gallian, *Contemporary Abstract Algebra*, **Huston Mifflan Company**, (1992).

The connection with monic irreducible polynomials

Every monic irreducible polynomial $p(x) \in \mathrm{GF}(p)[x]$ of degree $\leq m$ has a root in $\mathrm{GF}(p^m)$.

The connection with monic irreducible polynomials

Every monic irreducible polynomial $p(x) \in \mathrm{GF}(p)[x]$ of degree $\leq m$ has a root in $\mathrm{GF}(p^m)$.

This implies that there is a mapping from the set of elements of $\mathrm{GF}(p^m)$ to the set of monic irreducible polynomials of degree m over $\mathrm{GF}(p)$ that are not in any field $\mathbb{F}$ with $\mathrm{GF}(p) \subseteq \mathbb{F} \subset \mathrm{GF}(p^m)$.

The connection with monic irreducible polynomials

Every subfield of $\mathrm{GF}(p^m)$ is of the form $\mathrm{GF}(p^c)$ where $c|m$. So, the number of elements in $\mathrm{GF}(p^m)$ that are not in any proper subfield of $\mathrm{GF}(p^m)$ is

$$\sum_{c|m} \mu(c) p^{\left(\frac{m}{c}\right)}$$

The connection with monic irreducible polynomials

The connection with monic irreducible polynomials

We count the number of monic irreducible polynomials of degree c over $\mathrm{GF}(p)$.

The connection with monic irreducible polynomials

We count the number of monic irreducible polynomials of degree c over $\mathrm{GF}(p)$.

Definition

Let A be an algebraic closure of $\mathrm{GF}(p)$. The mapping $f(x) : A \longrightarrow A$ defined by $f(a) = a^p$ is called the Frobenius mapping.

The connection with monic irreducible polynomials

We count the number of monic irreducible polynomials of degree c over $\mathrm{GF}(p)$.

Definition

Let A be an algebraic closure of $\mathrm{GF}(p)$. The mapping $f(x) : A \longrightarrow A$ defined by $f(a) = a^p$ is called the Frobenius mapping.

If $p(x) \in \mathrm{GF}(p)[x]$ has root α, then it also has $\alpha^p = f(\alpha)$ as a root.

The connection with monic irreducible polynomials

We count the number of monic irreducible polynomials of degree c over $\mathrm{GF}(p)$.

Definition

Let A be an algebraic closure of $\mathrm{GF}(p)$. The mapping $f(x) : A \longrightarrow A$ defined by $f(a) = a^p$ is called the Frobenius mapping.

If $p(x) \in \mathrm{GF}(p)[x]$ has root α, then it also has $\alpha^p = f(\alpha)$ as a root.

Thus, a monic irreducible polynomial of degree c has factorization

$$(x - \alpha)(x - \alpha^p)(x - \alpha^{p^2}) \ldots (x - \alpha^{p^{c-1}}).$$

The connection with monic irreducible polynomials

There is a bijection between the set of all monic irreducible polynomials over $\mathrm{GF}(p^c)$ and the set of c – *element* sets

$$\{\alpha, \alpha^{p}, \alpha^{p^2}, \ldots \alpha^{p^{c-1}}\}.$$

The connection with monic irreducible polynomials

There is a bijection between the set of all monic irreducible polynomials over $\mathrm{GF}(p^c)$ and the set of c – *element* sets

$$\{\alpha, \alpha^{p}, \alpha^{p^2}, \ldots \alpha^{p^{c-1}}\}.$$

Any two irreducible polynomials have distinct roots, their corresponding sets are disjoint.

The connection with monic irreducible polynomials

There is a bijection between the set of all monic irreducible polynomials over $\mathrm{GF}(p^c)$ and the set of $c - \textit{element}$ sets

$$\{\alpha, \alpha^p, \alpha^{p^2}, \ldots \alpha^{p^{c-1}}\}.$$

Any two irreducible polynomials have distinct roots, their corresponding sets are disjoint.

Therefore, we have that the number of monic irreducible polynomials of degree c is

$$\frac{1}{c} \sum_{d|c} \mu(d) p^{\frac{c}{d}}.$$

The connection with monic irreducible polynomials

Theorem

For every prime power $p > 1$ and every positive integer c there exists a primitive normal basis of $\mathrm{GF}(p^c)$ over $\mathrm{GF}(p)$.[a]

[a] H.W.Lenstra, R.J. Schoof, *Primitive Normal Bases for Finite Fields*, **Mathematics of Computation Vol. 48 No. 177**, (1987).

The connection with monic irreducible polynomials

Theorem

For every prime power $p > 1$ and every positive integer c there exists a primitive normal basis of $\mathrm{GF}(p^c)$ over $\mathrm{GF}(p)$.[a]

[a] H.W.Lenstra, R.J. Schoof, *Primitive Normal Bases for Finite Fields*, **Mathematics of Computation Vol. 48 No. 177**, (1987).

Let $\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{c-1}}$ be the primitive normal basis, where α is a generator of the multiplicative group of $\mathrm{GF}(p^c)$.

The connection with monic irreducible polynomials

Theorem

For every prime power $p > 1$ and every positive integer c there exists a primitive normal basis of $\mathrm{GF}(p^c)$ over $\mathrm{GF}(p)$.[a]

[a] H.W.Lenstra, R.J. Schoof, *Primitive Normal Bases for Finite Fields*, **Mathematics of Computation Vol. 48 No. 177**, (1987).

Let $\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{c-1}}$ be the primitive normal basis, where α is a generator of the multiplicative group of $\mathrm{GF}(p^c)$.

Thus, each element $x \in \mathrm{GF}(p^c)$ can be uniquely written as

$$x = a_0\alpha + a_1\alpha^p + \ldots a_{c-1}\alpha^{p^{c-1}}$$

where $a_0, a_1, \ldots a_{c-1} \in \mathrm{GF}(p)$.

The Frobenius Map

The Frobenius map:

$$f^d : GF(p^c) \longrightarrow GF(p^c)$$
$$x \longmapsto x^{p^d}$$

The Frobenius Map

The Frobenius map:

$$f^d : GF(p^c) \longrightarrow GF(p^c)$$
$$x \longmapsto x^{p^d}$$

is a permutation on $GF(p^c)$, which fixes exactly the elements of $GF(p^d)$, where $d|c$.

The Frobenius Map, cont'd

The elements of $GF(p^d)$ thus satisfy:

$$f^d(x) = x$$

The Frobenius Map, cont'd

The elements of $GF(p^d)$ thus satisfy:

$$f^d(x) = x$$

and:

$$x = \sum_{j=0}^{c-1} a_j \alpha^{p^j}$$

$$f^d(x) = \sum_{j=0}^{p^c-1} a_j \alpha^{p^{j+d}}$$

The Frobenius Map, cont'd

The elements of $GF(p^d)$ thus satisfy:

$$f^d(x) = x$$

and:

$$x = \sum_{j=0}^{c-1} a_j \alpha^{p^j}$$

$$f^d(x) = \sum_{j=0}^{p^c-1} a_j \alpha^{p^{j+d}}$$

so: $\forall j \in \{0, ..., c-1\}, a_j = a_{j+d}$

The Frobenius Map, cont'd

The elements of $GF(p^d)$ thus satisfy:

$$f^d(x) = x$$

and:

$$x = \sum_{j=0}^{c-1} a_j \alpha^{p^j}$$

$$f^d(x) = \sum_{j=0}^{p^c-1} a_j \alpha^{p^{j+d}}$$

so: $\forall j \in \{0, ..., c-1\}, a_j = a_{j+d}$

Note that applying f^d to $x = (a_0, a_1, ..., a_{c-1})$ corresponds to applying a shift of d on the entries of x.

The Frobenius Map, cont'd

So the fixed points of f^d are in bijection with the lists $(a_0, a_1, ..., a_{c-1}) \in (GF(p))^c$ that are fixed after a shift of d on its entries.

The Frobenius Map, cont'd

So the fixed points of f^d are in bijection with the lists $(a_0, a_1, ..., a_{c-1}) \in (GF(p))^c$ that are fixed after a shift of d on its entries.

Definition

$Fix(f^d) := \{x \in GF(p^c) : f^d(x) = x\}$

The Frobenius Map, cont'd

So the fixed points of f^d are in bijection with the lists $(a_0, a_1, ..., a_{c-1}) \in (GF(p))^c$ that are fixed after a shift of d on its entries.

Definition

$Fix(f^d) := \{x \in GF(p^c) : f^d(x) = x\}$

Definition

$$C(f, c) = |Fix(f^c) \setminus \bigcup_{d||c} Fix(f^d)|$$

The Frobenius Map, cont'd

So the fixed points of f^d are in bijection with the lists $(a_0, a_1, ..., a_{c-1}) \in (GF(p))^c$ that are fixed after a shift of d on its entries.

Definition

$Fix(f^d) := \{x \in GF(p^c) : f^d(x) = x\}$

Definition

$$C(f, c) = |Fix(f^c) \setminus \bigcup_{d||c} Fix(f^d)|$$

$\frac{C(f,c)}{c}$ is the number of c-cycles.

The Frobenius Map, cont'd

$$|Fix(f^c)| = \sum_{d|c} C(f, c) = p^c$$

The Frobenius Map, cont'd

$$|Fix(f^c)| = \sum_{d|c} C(f, c) = p^c$$

By Möbius inversion formula:

$$C(f, c) = \sum_{d|c} \mu(d) p^{\frac{c}{d}}$$

The Frobenius Map, cont'd

$$|Fix(f^c)| = \sum_{d|c} C(f,c) = p^c$$

By Möbius inversion formula:

$$C(f,c) = \sum_{d|c} \mu(d) p^{\frac{c}{d}}$$

$$\Rightarrow \frac{1}{c} C(f,c) = \frac{1}{c} \sum_{d|c} \mu(d) p^{\frac{c}{d}}$$

Conclusions about the Parity of ShiftColumn

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \frac{m}{gcd(k, m)}$.

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \frac{m}{gcd(k, m)}$.
- The disjoint cycles in the disjoint cycle decomposition have length $c|\frac{m}{gcd(k, m)}$.

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \frac{m}{gcd(k, m)}$.
- The disjoint cycles in the disjoint cycle decomposition have length $c|\frac{m}{gcd(k, m)}$.

Recall:

- Goal: to determine the parity of f^k (view as a permutation).

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \frac{m}{gcd(k, m)}$.
- The disjoint cycles in the disjoint cycle decomposition have length $c|\frac{m}{gcd(k, m)}$.

Recall:

- Goal: to determine the parity of f^k (view as a permutation).
- To do this, determine the number of cycles of even length.

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \frac{m}{gcd(k,m)}$.
- The disjoint cycles in the disjoint cycle decomposition have length $c|\frac{m}{gcd(k,m)}$.

Recall:

- Goal: to determine the parity of f^k (view as a permutation).
- To do this, determine the number of cycles of even length.
- So we only need to regard the case: $\frac{m}{gcd(k,m)}$ even.

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \dfrac{m}{gcd(k,m)}$.
- The disjoint cycles in the disjoint cycle decomposition have length $c|\dfrac{m}{gcd(k,m)}$.

Recall:

- Goal: to determine the parity of f^k (view as a permutation).
- To do this, determine the number of cycles of even length.
- So we only need to regard the case: $\dfrac{m}{gcd(k,m)}$ even.

Proposition

For 2-cycles:

$$\begin{aligned} &\text{If } p = 1 \pmod 4, && \text{then the parity is even,} \\ &\text{If } p = 3 \pmod 4, && \text{if } gcd(k,m) \text{ is even, the parity is even,} \\ & && \text{if } gcd(k,m) \text{ is odd, the parity is odd.} \end{aligned}$$

Conclusions about the Parity of ShiftColumn

- For a shift of k, $order(f^k) = \dfrac{m}{gcd(k,m)}$.
- The disjoint cycles in the disjoint cycle decomposition have length $c|\dfrac{m}{gcd(k,m)}$.

Recall:

- Goal: to determine the parity of f^k (view as a permutation).
- To do this, determine the number of cycles of even length.
- So we only need to regard the case: $\dfrac{m}{gcd(k,m)}$ even.

Proposition

For 2-cycles:

$$\begin{aligned} &\text{If } p = 1 \pmod 4, && \text{then the parity is even,} \\ &\text{If } p = 3 \pmod 4, && \text{if } gcd(k,m) \text{ is even, the parity is even,} \\ & && \text{if } gcd(k,m) \text{ is odd, the parity is odd.} \end{aligned}$$

MixRow (ρ) Function

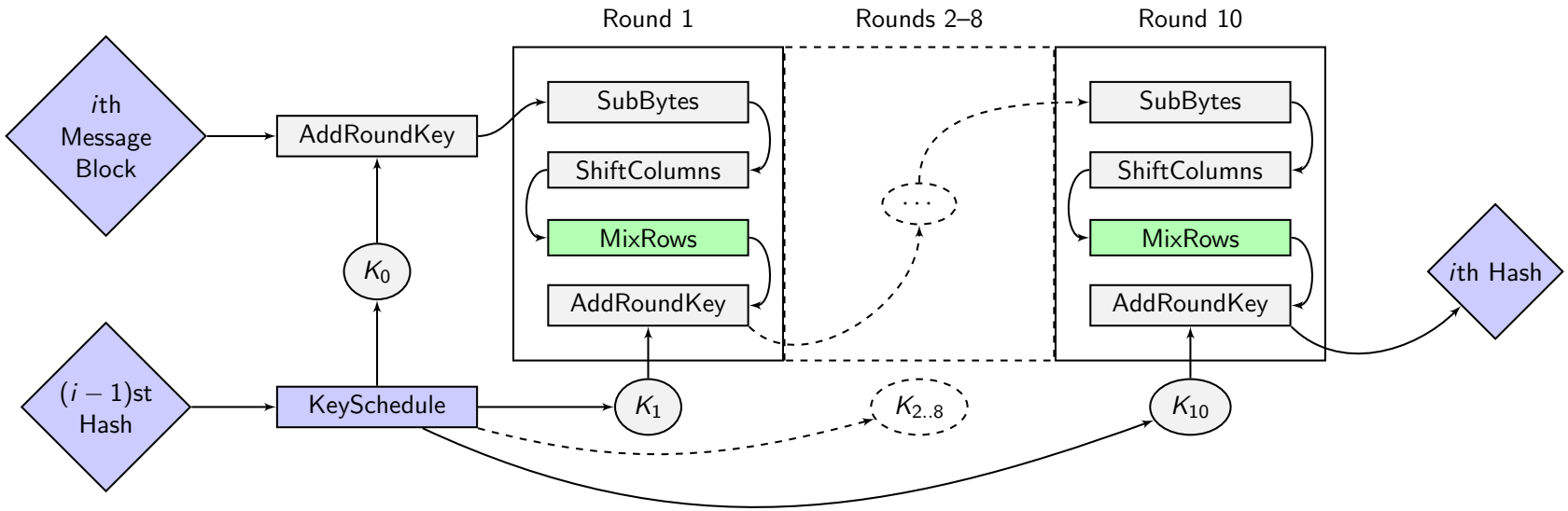


Figure : MixRow

MixRow (ρ) Function

Definition

Let $\rho : M_{m,n}(\mathrm{GF}(p^r)) \to M_{m,n}(\mathrm{GF}(p^r))$ is a mapping defined as the parallel application of n "row" mappings $\rho_i : M_{1,n}(\mathrm{GF}(p^r)) \to M_{1,n}(\mathrm{GF}(p^r))$ defined by $\rho(a) = a'$ if and only if $a'_i = \rho_i(a_i)$ for all $0 \leq i < m$, where each ρ_i is given by $\rho_i(x) = x \cdot C$ for all $x \in M_{1,n}(\mathrm{GF}(p^r))$, where $C \in M_{n,n}(\mathrm{GF}(p^r))$ is an invertible diffusion matrix.

MixRow (ρ) Function

Definition

Let $\rho : M_{m,n}(\mathrm{GF}(p^r)) \to M_{m,n}(\mathrm{GF}(p^r))$ is a mapping defined as the parallel application of n "row" mappings $\rho_i : M_{1,n}(\mathrm{GF}(p^r)) \to M_{1,n}(\mathrm{GF}(p^r))$ defined by $\rho(a) = a'$ if and only if $a'_i = \rho_i(a_i)$ for all $0 \leq i < m$, where each ρ_i is given by $\rho_i(x) = x \cdot C$ for all $x \in M_{1,n}(\mathrm{GF}(p^r))$, where $C \in M_{n,n}(\mathrm{GF}(p^r))$ is an invertible diffusion matrix.

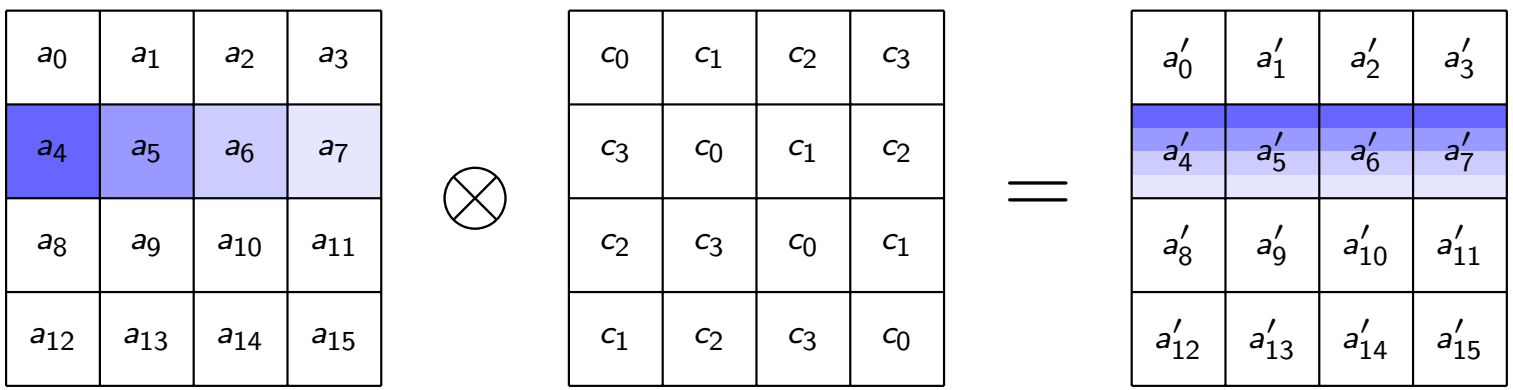


Figure : Example of MixRow Function

Previous Results

Theorem

Let C be an invertible MDS matrix

MixRow is an odd permutation if and only if p, m and $\frac{p^{rn}-1}{|\langle C \rangle|}$ are odd. [a]

[a] L. Babinkostova, K. Bombardier, M. Cole, T. Morrell, and C. Scott, *Algebraic Structure of generalized Rijndael-like SP networks*, (2012) (arXiv: 1210.7942)

Previous Results cont.

Theorem

Let C be an invertible circulant MDS matrix

If $\sum_{i=1}^{n} c_i \not\equiv 1$, then MixRow is an odd permutation if and only if

p, m and $\frac{p^{rn}-1}{|\langle C \rangle|}$ are odd.

If $\sum_{i=1}^{n} c_i \equiv 1$, then MixRow is an odd permutation if and only if

p, m and $\frac{p^{rn}-p^{r}}{|\langle C \rangle|}$ are odd. [a]

[a] L. Babinkostova, K. Bombardier, M. Cole, T. Morrell, and C. Scott, *Algebraic Structure of generalized Rijndael-like SP networks*, (2012) (arXiv: 1210.7942)

The Isomorphism

[1] Z. Zhang, *Construction of the Orthogonal Groups of $n \times n$ Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism

We want to construct an isomorphism between the set of circulant matrices and a set of polynomials:

[1] Z. Zhang, *Construction of the Orthogonal Groups of $n \times n$ Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism

We want to construct an isomorphism between the set of circulant matrices and a set of polynomials:

- Let F be the finite field with p^r elements where p is prime.

[1] Z. Zhang, *Construction of the Orthogonal Groups of n × n Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism

We want to construct an isomorphism between the set of circulant matrices and a set of polynomials:

- Let F be the finite field with p^r elements where p is prime.
- Let $\mathcal{M}$ be the algebra of $n \times n$ circulant matrices with elements from F.

[1] Z. Zhang, *Construction of the Orthogonal Groups of n × n Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism

We want to construct an isomorphism between the set of circulant matrices and a set of polynomials:

- Let F be the finite field with p^r elements where p is prime.
- Let $\mathcal{M}$ be the algebra of $n \times n$ circulant matrices with elements from F.
- Let $R_n = \frac{F[x]}{\langle (x^n - 1) \rangle}$ be the algebra of polynomials mod $(x^n - 1)$ over F.

[1] Z. Zhang, *Construction of the Orthogonal Groups of n × n Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism

We want to construct an isomorphism between the set of circulant matrices and a set of polynomials:

- Let F be the finite field with p^r elements where p is prime.
- Let $\mathcal{M}$ be the algebra of $n \times n$ circulant matrices with elements from F.
- Let $R_n = \frac{F[x]}{\langle (x^n - 1) \rangle}$ be the algebra of polynomials mod $(x^n - 1)$ over F.

$$\begin{aligned} \phi : \quad \mathcal{M} \quad &\mapsto \quad R_n \\ \sum a_i T^i &\mapsto \sum a_i x^i \end{aligned}$$

is an isomorphism.

[1] Z. Zhang, *Construction of the Orthogonal Groups of n × n Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism

We want to construct an isomorphism between the set of circulant matrices and a set of polynomials:

- Let F be the finite field with p^r elements where p is prime.
- Let $\mathcal{M}$ be the algebra of $n \times n$ circulant matrices with elements from F.
- Let $R_n = \frac{F[x]}{\langle (x^n - 1) \rangle}$ be the algebra of polynomials mod $(x^n - 1)$ over F.

$$\begin{aligned} \phi : \quad \mathcal{M} \quad &\mapsto \quad R_n \\ \sum a_i T^i &\mapsto \sum a_i x^i \end{aligned}$$

is an isomorphism.
ϕ maps a matrix A to its corresponding representative polynomial (the polynomial whose coefficients are the entries in the matrix). [1]

[1] Z. Zhang, *Construction of the Orthogonal Groups of n × n Circulant Matrices Over Finite Fields*, Masters thesis, Concordia University, National Library of Canada, (1997).

The Isomorphism, cont'd

We factor the polynomial $(x^n - 1)$ as $x^n - 1 = f_1^{n_1} \cdot f_2^{n_2} \cdots f_k^{n_k}$, where $f_i (1 \leq i \leq k)$ are distinct monic irreducible polynomials of $F[x]$.

The Isomorphism, cont'd

We factor the polynomial $(x^n - 1)$ as $x^n - 1 = f_1^{n_1} \cdot f_2^{n_2} \cdots f_k^{n_k}$, where $f_i (1 \leq i \leq k)$ are distinct monic irreducible polynomials of $F[x]$.

Then:

$$\mathcal{M} \cong \frac{F[x]}{\langle (x^n - 1) \rangle} \cong \frac{F[x]}{\langle f_1^{n_1} \rangle} \times \frac{F[x]}{\langle f_2^{n_2} \rangle} \times \cdots \times \frac{F[x]}{\langle f_r^{n_k} \rangle}$$

Chinese Remainder Theorem

By the Chinese Remainder Theorem,

Chinese Remainder Theorem

By the Chinese Remainder Theorem, if we take $a_1, a_2, \ldots, a_k$ in F, then we can find a $f(x)$ such that

$$\begin{aligned} f(x) &= a_1 \mod (f_1^{n_1}) \\ f(x) &= a_2 \mod (f_2^{n_2}) \\ &\vdots \\ f(x) &= a_k \mod (f_r^{n_k}) \end{aligned}$$

This $f(x)$ will be of the form

$$f(x) = c_{n-1}x^{n-1} + \cdots c_1 x^1 + c_0$$

Chinese Remainder Theorem

By the Chinese Remainder Theorem, if we take $a_1, a_2, \ldots, a_k$ in F, then we can find a $f(x)$ such that

$$\begin{aligned} f(x) &= a_1 \mod (f_1^{n_1}) \\ f(x) &= a_2 \mod (f_2^{n_2}) \\ &\vdots \\ f(x) &= a_k \mod (f_r^{n_k}) \end{aligned}$$

This $f(x)$ will be of the form

$$f(x) = c_{n-1}x^{n-1} + \cdots c_1 x^1 + c_0$$

and will generate a circulant matrix C,

Chinese Remainder Theorem

By the Chinese Remainder Theorem, if we take $a_1, a_2, \ldots, a_k$ in F, then we can find a $f(x)$ such that

$$\begin{aligned} f(x) &= a_1 \mod (f_1^{n_1}) \\ f(x) &= a_2 \mod (f_2^{n_2}) \\ &\vdots \\ f(x) &= a_k \mod (f_r^{n_k}) \end{aligned}$$

This $f(x)$ will be of the form

$$f(x) = c_{n-1}x^{n-1} + \cdots c_1 x^1 + c_0$$

and will generate a circulant matrix C, where

$$C = circ\{c_0, c_1, \cdots, c_{n-1}\}$$

Improved Result

Theorem

Let C be an invertible circulant matrix where $C \leftrightarrow (a_1, a_2, \cdots, a_k)$ and $a_i \in \frac{F[x]}{\langle f_i^{n_i} \rangle}$.

If $\sum_{i=1}^{n} c_i \not\equiv 1$, then MixRow is an odd permutation if and only if p, m and $\frac{p^{rn}-1}{\text{lcm}(\{\text{ord}(a_i)\}_1^k)}$ are odd.

If $\sum_{i=1}^{n} c_i \equiv 1$, then MixRow is an odd permutation if and only if p, m and $\frac{p^{rn}-p^r}{\text{lcm}(\{\text{ord}(a_i)\}_1^k)}$ are odd.

$|C|$ when n is prime

Theorem

Assume n is prime and $q = p^r$. Then the multiplicative group of $n \times n$ circulant matrices

$$C_n^*(GF(q)) \cong \mathbb{Z}_q^* \otimes \mathbb{Z}_{q^w}^{*\ \frac{n-1}{w}},$$

where $w = ord_n(q)$. [a]

[a] Alun Wyn-jones, *C*irculants, **http://circulants.org/circ/index.html** (2008)

$|C|$ when n is prime

Theorem

Assume n is prime and $q = p^r$. Then the multiplicative group of $n \times n$ circulant matrices

$$\mathcal{C}_n^*(GF(q)) \cong \mathbb{Z}_q^* \otimes \mathbb{Z}_{q^w}^{*\ \frac{n-1}{w}},$$

where $w = ord_n(q)$. [a]

[a] Alun Wyn-jones, *C*irculants, **http://circulants.org/circ/index.html** (2008)

If q is a primitive root of n, then $\mathcal{C}_n^*(GF(q)) \cong \mathbb{Z}_q^* \otimes \mathbb{Z}_{q^{n-1}}^*$

When q is a primitive root of n

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y) = (g_1^{\ell}, g_2^{k})$ for $x \in \mathbb{Z}_q^*$ and $y \in \mathbb{Z}_{q^{n-1}}^*$

When q is a primitive root of n

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y) = (g_1^\ell, g_2^k)$ for $x \in \mathbb{Z}_q^*$ and $y \in \mathbb{Z}_{q^{n-1}}^*$

$$ord(x) = ord(g_1^\ell) = \frac{ord(g_1)}{\gcd(\ell, ord(g_1))} = \frac{q-1}{\gcd(\ell, q-1)}$$

$$ord(y) = ord(g_2^k) = \frac{ord(g_2)}{\gcd(k, ord(g_2))} = \frac{q^{n-1}-1}{\gcd(k, q^{n-1}-1)}$$

When q is a primitive root of n

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y) = (g_1^\ell, g_2^k)$ for $x \in \mathbb{Z}_q^*$ and $y \in \mathbb{Z}_{q^{n-1}}^*$

$$ord(x) = ord(g_1^\ell) = \frac{ord(g_1)}{\gcd(\ell, ord(g_1))} = \frac{q-1}{\gcd(\ell, q-1)}$$

$$ord(y) = ord(g_2^k) = \frac{ord(g_2)}{\gcd(k, ord(g_2))} = \frac{q^{n-1}-1}{\gcd(k, q^{n-1}-1)}$$

$$|C| = \mathrm{lcm}(ord(x), ord(y))$$

When q is a primitive root of n

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y) = (g_1^\ell, g_2^k)$ for $x \in \mathbb{Z}_q^*$ and $y \in \mathbb{Z}_{q^{n-1}}^*$

$$ord(x) = ord(g_1^\ell) = \frac{ord(g_1)}{\gcd(\ell, ord(g_1))} = \frac{q-1}{\gcd(\ell, q-1)}$$

$$ord(y) = ord(g_2^k) = \frac{ord(g_2)}{\gcd(k, ord(g_2))} = \frac{q^{n-1}-1}{\gcd(k, q^{n-1}-1)}$$

$$|C| = \text{lcm}(ord(x), ord(y))$$

But since $\text{lcm}(s, r) = \frac{s \cdot r}{\gcd(s,r)}$,

When q is a primitive root of n

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y) = (g_1^\ell, g_2^k)$ for $x \in \mathbb{Z}_q^*$ and $y \in \mathbb{Z}_{q^{n-1}}^*$

$$ord(x) = ord(g_1^\ell) = \frac{ord(g_1)}{\gcd(\ell, ord(g_1))} = \frac{q-1}{\gcd(\ell, q-1)}$$

$$ord(y) = ord(g_2^k) = \frac{ord(g_2)}{\gcd(k, ord(g_2))} = \frac{q^{n-1}-1}{\gcd(k, q^{n-1}-1)}$$

$$|C| = \text{lcm}(ord(x), ord(y))$$

But since $\text{lcm}(s, r) = \frac{s \cdot r}{\gcd(s,r)}$,

$$|C| = \frac{(q-1)(q^{n-1}-1)}{\gcd(\ell, q-1) \cdot \gcd(k, q^{n-1}-1) \cdot \gcd\left(\frac{q-1}{\gcd(\ell,q-1)}, \frac{q^{n-1}-1}{\gcd(k,q^{n-1}-1)}\right)}$$

Parity of $|C|$

Define a, b, c, d, e such that

$$\begin{aligned} q^{n-1} - 1 &= 2^a \cdot P_1 \\ q - 1 &= 2^b \cdot P_2 \\ k &= 2^c \cdot P_3 \\ \ell &= 2^d \cdot P_4 \\ |C| &= 2^e \cdot P_5 \end{aligned}$$

Parity of $|C|$

Define a, b, c, d, e such that

$$q^{n-1} - 1 = 2^a \cdot P_1$$
$$q - 1 = 2^b \cdot P_2$$
$$k = 2^c \cdot P_3$$
$$\ell = 2^d \cdot P_4$$
$$|C| = 2^e \cdot P_5$$

Using the formula for $|C|$,

$$e = a+b-min(a,c)-min(b,d)-min((a-min(a,c)),(b-min(b,d)))$$

$$e = max((a - min(a,c)),(b - min(b,d)))$$

Parity of $|C|$

Define a, b, c, d, e such that

$$q^{n-1} - 1 = 2^a \cdot P_1$$
$$q - 1 = 2^b \cdot P_2$$
$$k = 2^c \cdot P_3$$
$$\ell = 2^d \cdot P_4$$
$$|C| = 2^e \cdot P_5$$

Using the formula for $|C|$,

$$e = a+b-min(a,c)-min(b,d)-min((a-min(a,c)),(b-min(b,d)))$$

$$e = max((a - min(a,c)),(b - min(b,d)))$$

$|C|$ is odd if and only if $c \geq a$ and $d \geq b$

Parity of MixRow

Theorem

Let C be an invertible circulant MDS matrix.

If $\sum_{i=1}^{n} c_i \not\equiv 1$, then MixRow is an odd permutation if and only if p, m and $\frac{p^{rn}-1}{|\langle C \rangle|}$ are odd. If $\sum_{i=1}^{n} c_i \equiv 1$, then MixRow is an odd permutation if and only if p, m and $\frac{p^{rn}-p^r}{|\langle C \rangle|}$ are odd. [a]

[a] L. Babinkostova, K. Bombardier, M. Cole, T. Morrell, and C. Scott, *Algebraic Structure of generalized Rijndael-like SP networks*, (2012) (arXiv: 1210.7942)

Proposition

If $c \geq a$ and $d \geq b$, then MixRow is an even permutation.

Min and Max of $|C|$

Let C be an invertible circulant MDS matrix.

$$|C| = \frac{(q-1)(q^{n-1}-1)}{\gcd(\ell,q-1)\cdot\gcd(k,q^{n-1}-1)\cdot\gcd\left(\frac{q-1}{\gcd(\ell,q-1)},\frac{q^{n-1}-1}{\gcd(k,q^{n-1}-1)}\right)}$$

Min and Max of $|C|$

Let C be an invertible circulant MDS matrix.

$$|C| = \frac{(q-1)(q^{n-1}-1)}{\gcd(\ell,q-1)\cdot\gcd(k,q^{n-1}-1)\cdot\gcd\left(\frac{q-1}{\gcd(\ell,q-1)},\frac{q^{n-1}-1}{\gcd(k,q^{n-1}-1)}\right)}$$

Proposition

If $C \leftrightarrow (x, y)$, $|C|$ attains its minimum of 1 if and only if $x = y = 1$

Min and Max of $|C|$

Let C be an invertible circulant MDS matrix.

$$|C| = \frac{(q-1)(q^{n-1}-1)}{\gcd(\ell,q-1)\cdot\gcd(k,q^{n-1}-1)\cdot\gcd\left(\frac{q-1}{\gcd(\ell,q-1)},\frac{q^{n-1}-1}{\gcd(k,q^{n-1}-1)}\right)}$$

Proposition

If $C \leftrightarrow (x, y)$, $|C|$ attains its minimum of 1 if and only if $x = y = 1$

Proposition

$|C|$ attains its maximum of $q^{n-1} - 1$ if and only if $gcd(k, q^{n-1} - 1)|q - 1$ and $\gcd(\gcd(\ell, q - 1), \gcd(k, q^{n-1} - 1)) = 1$

Generalizing for any w

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y_1, y_2, \cdots, y_{\frac{n-1}{w}}) = (g_1^{\ell}, g_2^{k_1}, g_2^{k_2}, \cdots, g_2^{k_{\frac{n-1}{w}}})$ for $x \in \mathbb{Z}_q^*$ and $y_i \in \mathbb{Z}_{q^w}^*$ and where w is the order of q mod n

Generalizing for any w

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y_1, y_2, \cdots, y_{\frac{n-1}{w}}) = (g_1^{\ell}, g_2^{k_1}, g_2^{k_2}, \cdots, g_2^{k_{\frac{n-1}{w}}})$ for $x \in \mathbb{Z}_q^*$ and $y_i \in \mathbb{Z}_{q^w}^*$ and where w is the order of q mod n

We define a, b, c_i, d, e for $(1 \leq i \leq \frac{n-1}{w})$ such that:

$$q^w - 1 = 2^a \cdot P_1$$
$$q - 1 = 2^b \cdot P_2$$
$$k_i = 2^{c_i} \cdot P_{3_i}$$
$$\ell = 2^d \cdot P_4$$
$$|C| = 2^e \cdot P_5$$

Generalizing for any w

For $C \in \mathcal{C}_n^*(GF(q))$,

$C \leftrightarrow (x, y_1, y_2, \cdots, y_{\frac{n-1}{w}}) = (g_1^{\ell}, g_2^{k_1}, g_2^{k_2}, \cdots, g_2^{k_{\frac{n-1}{w}}})$ for $x \in \mathbb{Z}_q^*$ and $y_i \in \mathbb{Z}_{q^w}^*$ and where w is the order of q mod n

We define a, b, c_i, d, e for $(1 \leq i \leq \frac{n-1}{w})$ such that:

$$q^w - 1 = 2^a \cdot P_1$$
$$q - 1 = 2^b \cdot P_2$$
$$k_i = 2^{c_i} \cdot P_{3_i}$$
$$\ell = 2^d \cdot P_4$$
$$|C| = 2^e \cdot P_5$$

Proposition

$|C|$ is odd if and only if $c_i \geq a$ and $d \geq b$

Generalizing cont.

Proposition

If $c_i \geq a$ and $d \geq b$, then MixRow is an even permutation.

Generalizing cont.

Proposition

If $c_i \geq a$ and $d \geq b$, then MixRow is an even permutation.

Proposition

$|C|$ attains its minimum of 1 if and only if $y_i = 1$ and $x = 1$

Generalizing cont.

Proposition

If $c_i \geq a$ and $d \geq b$, then MixRow is an even permutation.

Proposition

$|C|$ attains its minimum of 1 if and only if $y_i = 1$ and $x = 1$

Proposition

$|C|$ attains its maximum of $q^w - 1$ if and only if $\exists i \in [1, \frac{n-1}{w}]$ such that $\gcd(k_i, q^w - 1) = a, a|q - 1$ and $\gcd(a, \gcd(\ell, q - 1)) = 1$

Data Collected

q	$q-1$	n	$\max\{\lvert C\rvert\}$	$\lvert\mathcal{C}_n^*(\mathrm{GF}(q))\rvert$
2^4	15	2	30	240
		3	15	3375
		4	60	61440
		5	15	759375
		7	?	251535375
3^2	8	2	8	64
		3	24	648
		4	8	4096
		5	40	51200
		7	?	4239872

SubBytes (λ) Function

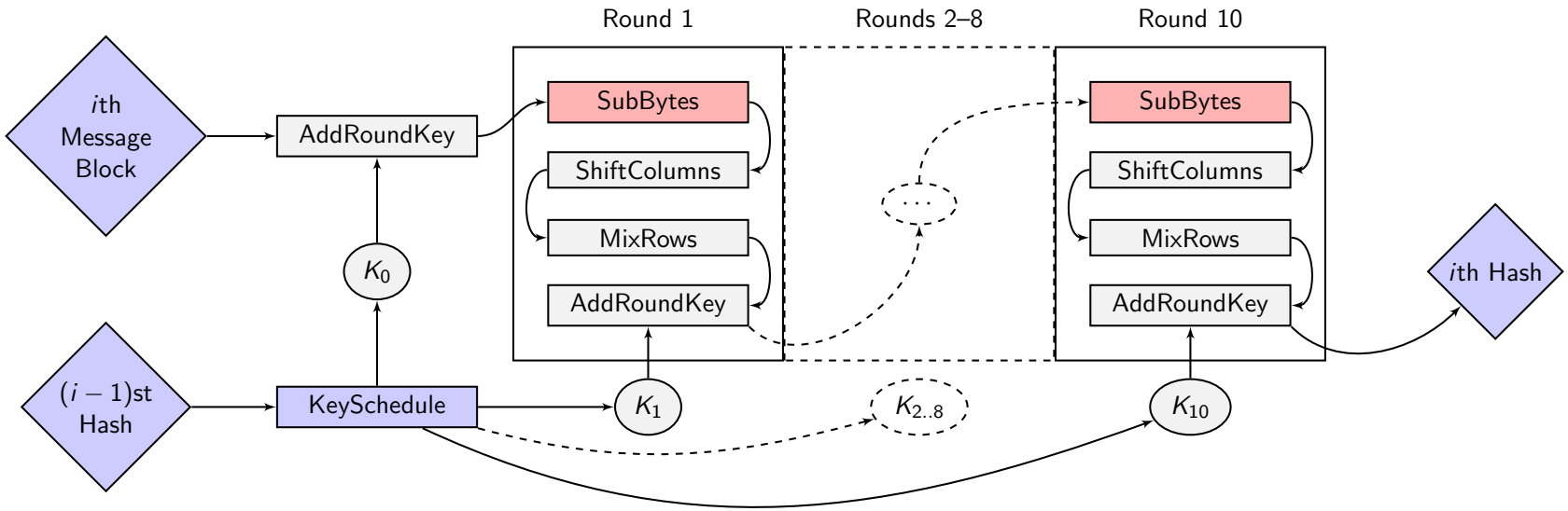
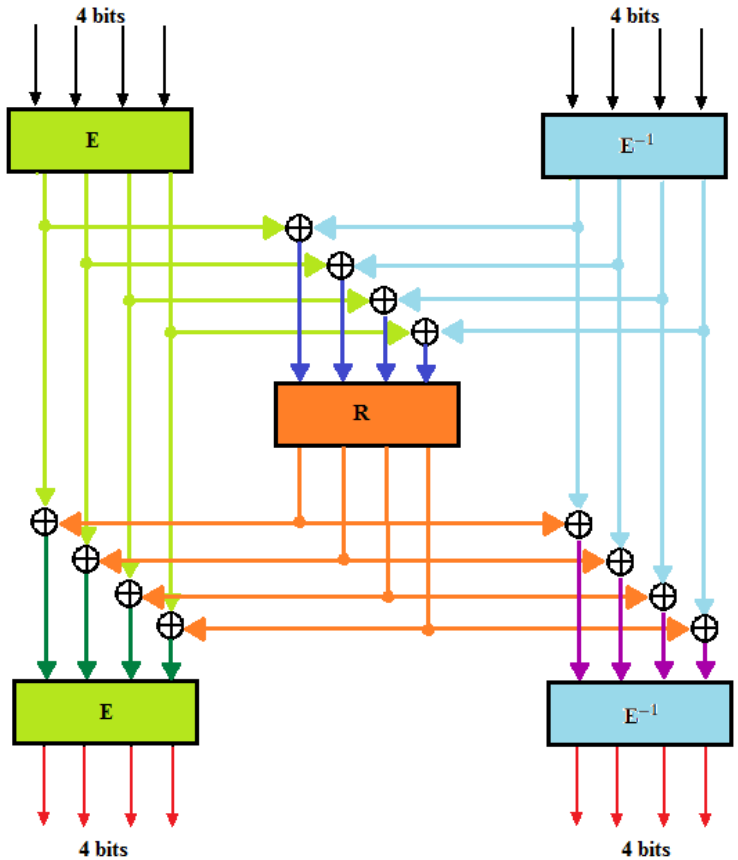


Figure : SubBytes

The Whirlpool S-Box



$E =$

u	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$E(u)$	1	B	9	C	D	6	F	3	E	8	7	4	A	2	5	0

$R =$

u	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$R(u)$	7	C	B	D	E	4	9	F	6	3	8	A	2	5	1	0

The E Mini-Box

The underlying mapping of the WHIRLPOOL mini E-box is exponentiation in a finite field.

The E Mini-Box

The underlying mapping of the WHIRLPOOL mini E-box is exponentiation in a finite field.

Let g be a generator of the multiplicative group $GF(p^r) \setminus \{0\}$ and $q(x)$ be a primitive polynomial in $\mathrm{GF}(p^r)$.

The E Mini-Box

The underlying mapping of the WHIRLPOOL mini E-box is exponentiation in a finite field.

Let g be a generator of the multiplicative group $GF(p^r) \setminus \{0\}$ and $q(x)$ be a primitive polynomial in $\mathrm{GF}(p^r)$.

We consider the map $E : \mathrm{GF}(p^r) \longrightarrow \mathrm{GF}(p^r)$ with respect to the polynomial $q(x)$ and map $u \in GF(p^r) \setminus \{0\}$ to g^u, where $u \in \mathrm{GF}(p^r)$ is taken to be its numerical value

$$u = \sum_{i=0}^{r-1} u_i . p^i .$$

The E Mini-Box

Definition

$$E(u) = \begin{cases} 0 & \text{if } u = p^r - 1 \\ g^u & \text{otherwise} \end{cases},$$

where $u \in GF(p^r)$ and g is a generator of $\mathrm{GF}(p^r)^*$.

The E Mini-Box

Definition

$$E(u) = \begin{cases} 0 & \text{if } u = p^r - 1 \\ g^u & \text{otherwise} \end{cases},$$

where $u \in GF(p^r)$ and g is a generator of $\mathrm{GF}(p^r)^*$.

Standard WHIRLPOOL encryption uses $g = B$ as the generator.

The E Mini-Box

Definition

$$E(u) = \begin{cases} 0 & \text{if } u = p^r - 1 \\ g^u & \text{otherwise} \end{cases},$$

where $u \in GF(p^r)$ and g is a generator of $\mathrm{GF}(p^r)^*$.

Standard WHIRLPOOL encryption uses $g = B$ as the generator.

$E =$

u	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$E(u)$	1	B	9	C	D	6	F	3	E	8	7	4	A	2	5	0

The E Mini-Box

Definition

We define the permutation induced by the generator g as the mapping $u \longrightarrow g^u$, where u runs through all the elements $\{1, 2, \ldots, p^r - 1\}$ of the finite field.

$$\sigma_g = \begin{pmatrix} 0 & 1 & 2 & \cdots & p^r - 2 & p^r - 1 \\ 1 & g^1 & g^2 & \cdots & g^{p^r - 2} & 0 \end{pmatrix}$$

The E Mini-Box

Definition

We define the permutation induced by the generator g as the mapping $u \longrightarrow g^u$, where u runs through all the elements $\{1, 2, \ldots, p^r - 1\}$ of the finite field.

$$\sigma_g = \begin{pmatrix} 0 & 1 & 2 & \cdots & p^r - 2 & p^r - 1 \\ 1 & g^1 & g^2 & \cdots & g^{p^r - 2} & 0 \end{pmatrix}$$

In terms of this, we shall now call g a "good" generator if the permutation induced by it, written in disjoint cycle notation, contains no $1-$ or $2-$ *cycles*.

The E Mini-Box

Definition

We define the permutation induced by the generator g as the mapping $u \longrightarrow g^u$, where u runs through all the elements $\{1, 2, \ldots, p^r - 1\}$ of the finite field.

$$\sigma_g = \begin{pmatrix} 0 & 1 & 2 & \cdots & p^r - 2 & p^r - 1 \\ 1 & g^1 & g^2 & \cdots & g^{p^r-2} & 0 \end{pmatrix}$$

In terms of this, we shall now call g a "good" generator if the permutation induced by it, written in disjoint cycle notation, contains no $1-$ or $2-$ *cycles*.

Otherwise, we call g a "bad" generator.

Facts about Generators

Theorem

Let g be a generator $(\mathrm{mod}\ p)$, *σ_g be its induced permutation, and* $T = \{\tau_j = \{j, (p-1)-j\} | 1 \leq j < p-1, j \neq \frac{p-1}{2}\}$.

- $(\sigma_{g^{-1}}$ *has* $1-$ *cycle* $(a))$ *iff* $(\sigma_g((p-1)-a) = a)$
- $(\sigma_{g^{-1}}$ *has* $2-$ *cycle* $(a, b))$ *iff* $((a, (p-1)-a) = \tau_i, (b, (p-1)-b)\tau_j$, $i \neq j$, *and* $\sigma_g((p-1)-a) = b$, *and* $\sigma_g((p-1)-b) = a)$

Facts about Generators

Theorem

Let g be a generator $(\mathrm{mod}\ p)$, *σ_g be its induced permutation, and* $T = \{\tau_j = \{j, (p-1)-j\} | 1 \leq j < p-1, j \neq \frac{p-1}{2}\}$.

- $(\sigma_{g^{-1}}$ *has* $1-$ *cycle* $(a))$ *iff* $(\sigma_g((p-1)-a) = a)$
- $(\sigma_{g^{-1}}$ *has* $2-$ *cycle* $(a, b))$ *iff* $((a, (p-1)-a) = \tau_i, (b, (p-1)-b)\tau_j$, $i \neq j$, *and* $\sigma_g((p-1)-a) = b$, *and* $\sigma_g((p-1)-b) = a)$

Theorem

If $p \equiv 1 \pmod 4$, *and g is a generator then* $(p-g)^{\frac{(p+1)}{2}} \equiv g \pmod p$.

Facts about Generators

Theorem

Let g be a generator (mod p), *σ_g be its induced permutation, and* $T = \{\tau_j = \{j, (p-1)-j\} | 1 \leq j < p-1, j \neq \frac{p-1}{2}\}$.

- ($\sigma_{g^{-1}}$ *has* $1-$ *cycle* (a)) *iff* ($\sigma_g((p-1)-a) = a$)
- ($\sigma_{g^{-1}}$ *has* $2-$ *cycle* (a, b)) *iff*
  $((a, (p-1)-a) = \tau_i, (b, (p-1)-b)\tau_j$, $i \neq j$, *and*
  $\sigma_g((p-1)-a) = b$, *and* $\sigma_g((p-1)-b) = a$)

Theorem

If $p \equiv 1$ (mod 4), *and g is a generator then* $(p-g)^{\frac{(p+1)}{2}} \equiv g$ (mod p).
i.e., $\sigma_{p-g}(\frac{p+1}{2}) = g$.

Facts about Generators

Proposition

Let $v \in \mathbb{Z}_p^*$ *and* $d = \frac{p-1}{ord_p(v)}$, *then*

$\{u \in \mathbb{Z}_p \mid g^u = v \text{ for some } g\} = \{u'd \mid gcd(u', \frac{p-1}{d}) = 1\}$.

Furthermore, every u *is an pre-image of* v *exactly* $\frac{\phi(p-1)}{\phi(\frac{p-1}{d})}$ *times.*

Facts about Generators

Proposition

Let $v \in \mathbb{Z}_p^*$ *and* $d = \frac{p-1}{ord_p(v)}$, *then*

$\{u \in \mathbb{Z}_p \mid g^u = v \text{ for some } g\} = \{u'd \mid gcd(u', \frac{p-1}{d}) = 1\}$.

Furthermore, every u *is an pre-image of* v *exactly* $\frac{\phi(p-1)}{\phi(\frac{p-1}{d})}$ *times.*

Corollary

v *is its own pre-image iff* $gcd(v, p-1) = d$.

Facts about Generators

So given a prime p, there will be a fixed point for some generator if and only if there exists an element $v \in \mathbb{Z}_p^*$ where
$d = \frac{p-1}{ord_p(v)}$
and
$gcd(v, p-1) = d$.

Corollary

If v is a generator and is relatively prime to $p-1$, then v will be a fixed point for some generator.

Facts about Generators

Proposition

Let p be a prime, g be a generator dividing $p-1$, let $k_0 \in Z_p$ such that $gcd(g^{k_0}, p-1) = g^{k_0-1}$, then $g^{g^{k_0-1}}$ is a fixed point for some generator.

Facts about Generators

Proposition

Let p be a prime, g be a generator dividing $p-1$, let $k_0 \in Z_p$ such that $gcd(g^{k_0}, p-1) = g^{k_0-1}$, then $g^{g^{k_0-1}}$ is a fixed point for some generator.

Note that this is assuming the existence of a generator dividing $p-1$ and that $gcd(g, \frac{p-1}{g^{k_0-1}}) = 1$.

Facts about Generators

Theorem

Let p be a prime and g a generator that is also a prime number,

Facts about Generators

Theorem

Let p be a prime and g a generator that is also a prime number, if $g|p-1$, then $g^{g^{k_0-1}}$ is a fixed point,

Facts about Generators

Theorem

Let p be a prime and g a generator that is also a prime number,
if $g|p-1$, then $g^{g^{k_0-1}}$ is a fixed point,
else g is relatively prime with $p-1$, then g will be a fixed point.

Facts about Generators

Theorem

Let p be a prime and g a generator that is also a prime number, if $g|p-1$, then $g^{g^{k_0-1}}$ is a fixed point, else g is relatively prime with $p-1$, then g will be a fixed point.

So our question of the existence of fixed points is reduced to the existence of a prime generator for every finite cyclic group $\mathbb{Z}_p^*$!

Example for $p = 31$

Example for $p = 31$

$$ord_p\left(\frac{p-1}{2}\right) = 10 = \frac{p-1}{d}.$$

Example for $p = 31$

$ord_p\left(\frac{p-1}{2}\right) = 10 = \frac{p-1}{d}$. So $v = \frac{p-1}{2}$ and $d = 3$. Furthermore, an example of a generator $\pmod{31}$ which is also a prime is $g = 3$. $\gcd(3^2, 30) = 3$, thus $k_0 = 2$.

Example for $p = 31$

$ord_p\left(\frac{p-1}{2}\right) = 10 = \frac{p-1}{d}$. So $v = \frac{p-1}{2}$ and $d = 3$. Furthermore, an example of a generator $\pmod{31}$ which is also a prime is $g = 3$. $\gcd(3^2, 30) = 3$, thus $k_0 = 2$.

Thus, we expect the numbers preceding $\frac{p-1}{2}$ to be multiples of 3, and each to appear for $\phi(3) = 2$ generators, since $3^2 \nmid 30$.

Example for $p = 31$

$ord_p\left(\frac{p-1}{2}\right) = 10 = \frac{p-1}{d}$. So $v = \frac{p-1}{2}$ and $d = 3$. Furthermore, an example of a generator $\pmod{31}$ which is also a prime is $g = 3$. $\gcd(3^2, 30) = 3$, thus $k_0 = 2$.

Thus, we expect the numbers preceding $\frac{p-1}{2}$ to be multiples of 3, and each to appear for $\phi(3) = 2$ generators, since $3^2 \nmid 30$.

31 with primitive root
3 : (0 1 3 27 23 11 13 24 2 9 29 21 15 30)(4 19 12 8 20 5 26 18)
(6 16 28 7 17 22 14 10 25)
11 : (0 1 11 24 8 19 22 18 2 28 10 5 6 4 9 23 12 16 20 25 26 7 13 21
27 15 30)(3 29 17)(14)
12 : (0 1 12 4 28 14 18 8 9 15 30)(2 20 5 26 10 25 6)
(3 23 22 7 24 16 19)(11 21 29 13 17)(27)
13 : (0 1 13 11 3 27 23 24 2 14 19 21 15 30)(4 10 5 6 16 18)
(7 22 9 29 12 8)(17)(20 25 26 28)

Example for $p = 31$

17 : (0 1 17 21 23 13 3 15 30)
(2 10 25 6 8 18 16 14 20 5 26 9 27 29 11 22 19 24 4 7 12)(28)
21 : (0 1 21 29 3 23 17 24 16 10 5 6 2 7 11 12 4 18 8 14 28 9 15 30)
(13 22 20 25 26 19)(27)
22 : (0 1 22 10 5 6 8 28 18 16 9 27 29 24 4 20 25 26 14 7 21 23 3
15 30)(2 19 11 17 12)(13)
24 : (0 1 24 8 10 25 6 4 14 9 23 21 27 15 30)(2 18)
(3 29 22 28 19 17 13 12 16 7)(5 26 20)(11)

Results

- **AddRoundKey:** Even for all $k \in M_{m,n}(\mathrm{GF}(p^r))$ and $r, m, n > 1$.

Results

- **AddRoundKey:** Even for all $k \in M_{m,n}(\mathrm{GF}(p^r))$ and $r, m, n > 1$.
- **ShiftColumn:** For 2-cycles:

  If $p = 1 \pmod 4$, then the parity is even,

  If $p = 3 \pmod 4$, if $gcd(k, m)$ is even, the parity is even,

  if $gcd(k, m)$ is odd, the parity is odd.

Results

- **AddRoundKey:** Even for all $k \in M_{m,n}(\mathrm{GF}(p^r))$ and $r, m, n > 1$.
- **ShiftColumn:** For 2-cycles:

  If $p = 1 \pmod 4$, then the parity is even,

  If $p = 3 \pmod 4$, if $gcd(k, m)$ is even, the parity is even,

  if $gcd(k, m)$ is odd, the parity is odd.

- **MixRow:** Let $\mathcal{C} = \sum_{i=1}^{n} c_i$.
  - Odd permutation if and only if p, m and $\dfrac{p^{rn} - 1}{\mathrm{lcm}(\{\mathrm{ord}(a_i)\}_1^r)}$ are odd when $\mathcal{C} \not\equiv 1$.
  - Odd permutation if and only if p, m and $\dfrac{p^{rn} - p^r}{\mathrm{lcm}(\{\mathrm{ord}(a_i)\}_1^r)}$ are odd when $\mathcal{C} \equiv 1$.

Results

- **AddRoundKey:** Even for all $k \in M_{m,n}(\mathrm{GF}(p^r))$ and $r, m, n > 1$.
- **ShiftColumn:** For 2-cycles:

  If $p = 1 \pmod 4$, then the parity is even,

  If $p = 3 \pmod 4$, if $gcd(k, m)$ is even, the parity is even,

  if $gcd(k, m)$ is odd, the parity is odd.

- **MixRow:** Let $\mathcal{C} = \sum_{i=1}^{n} c_i$.
  - Odd permutation if and only if p, m and $\dfrac{p^{rn} - 1}{\mathrm{lcm}(\{\mathrm{ord}(a_i)\}_1^r)}$ are odd when $\mathcal{C} \not\equiv 1$.
  - Odd permutation if and only if p, m and $\dfrac{p^{rn} - p^r}{\mathrm{lcm}(\{\mathrm{ord}(a_i)\}_1^r)}$ are odd when $\mathcal{C} \equiv 1$.
- **SubBytes:** The standard Whirlpool SubBytes in $\mathrm{GF}(2^8)$ is even.

Future Work

- Determine the parity of the generalized WHIRLPOOL block cipher.

Future Work

- Determine the parity of the generalized WHIRLPOOL block cipher.
- For SubBytes: Determine when we can define the E function for the generalized WHIRLPOOL function in $GF(p^r)$.

Future Work

- Determine the parity of the generalized WHIRLPOOL block cipher.
- For SubBytes: Determine when we can define the E function for the generalized WHIRLPOOL function in $GF(p^r)$.
- Find the group generated by the generalized WHIRLPOOL function.

Acknowledgments

Acknowledgments

- Boise State Mathematics Department

Acknowledgments

- Boise State Mathematics Department
- National Science Foundation grant DMS 1062857

Questions?