Funções de Resumo

Universidade Federal do Ceará - Campus Quixadá

Roberto Cabral
rbcabral@ufc.br

05 de Maio de 2023

Auditoria e Segurança de SI

Funções de resumo criptográfico

- Uma função de resumo h mapeia uma string de bits de tamanho arbitrário em uma string de n bits.

$$h : \{0,1\}^* \rightarrow \{0,1\}^n$$

Funções de resumo criptográfico

- Uma função de resumo h mapeia uma string de bits de tamanho arbitrário em uma string de n bits.

$$h : \{0,1\}^* \rightarrow \{0,1\}^n$$

- A string de bits gerada é conhecida por *digest* ou valor de resumo.

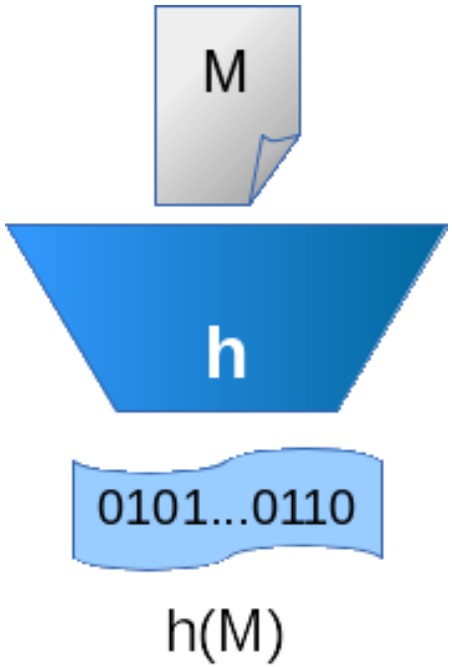
Funções de resumo criptográfico

- Uma função de resumo h mapeia uma string de bits de tamanho arbitrário em uma string de n bits.

$$h : \{0,1\}^* \rightarrow \{0,1\}^n$$

- A string de bits gerada é conhecida por *digest* ou valor de resumo.
- Funções de resumo são usadas em várias aplicações criptográficas, como assinatura digital, derivação de chaves, geração de números pseudoaleatórios, dentre outras.

Funções de resumo criptográfico



Propriedades de uma função de resumo

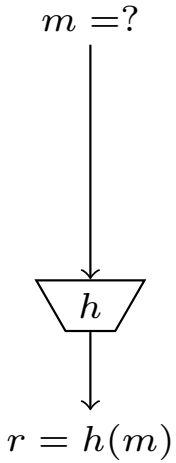
- **Resistência à primeira pré-imagem:** Dada uma função $h : M \to R$ e um valor de resumo $r \in R$, é computacionalmente inviável encontrar uma mensagem $m \in M$ tal que $h(m) = r$.

Propriedades de uma função de resumo

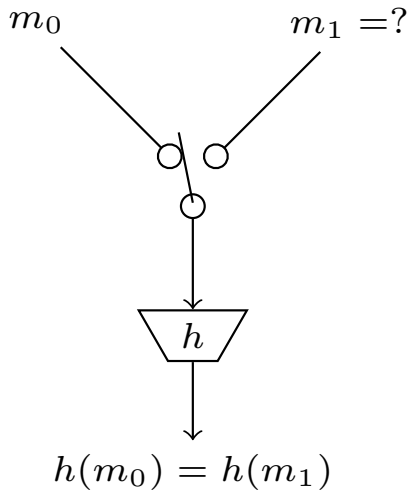
- **Resistência à primeira pré-imagem:** Dada uma função $h : M \to R$ e um valor de resumo $r \in R$, é computacionalmente inviável encontrar uma mensagem $m \in M$ tal que $h(m) = r$.
- **Resistência à segunda pré-imagem:** Dada uma função $h : M \to R$ e uma mensagem $m_0 \in M$, é computacionalmente inviável encontrar uma mensagem $m_1 \in M$ tal que $m_0 \neq m_1$ e $h(m_0) = h(m_1)$.

Propriedades de uma função de resumo

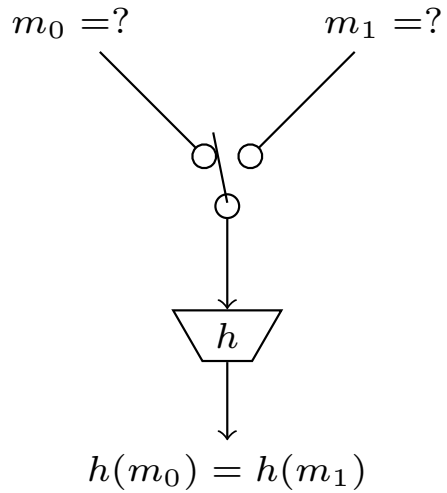
- **Resistência à primeira pré-imagem:** Dada uma função $h : M \to R$ e um valor de resumo $r \in R$, é computacionalmente inviável encontrar uma mensagem $m \in M$ tal que $h(m) = r$.
- **Resistência à segunda pré-imagem:** Dada uma função $h : M \to R$ e uma mensagem $m_0 \in M$, é computacionalmente inviável encontrar uma mensagem $m_1 \in M$ tal que $m_0 \neq m_1$ e $h(m_0) = h(m_1)$.
- **Resistência à colisão:** Dada uma função $h : M \to R$, é computacionalmente inviável encontrar duas mensagens $m_0, m_1 \in M$ tal que $m_0 \neq m_1$ e $h(m_0) = h(m_1)$.



(a) Pré-imagem.



(b) Segunda pré-imagem.



(c) Colisões.

Uma função de resumo resistente a colisões também é resistente à segunda pré-imagem! Entretanto, uma função de resumo ser resistente à colisão não garante que ela seja resistente à primeira pré-imagem

Oráculo Aleatório

- O oráculo aleatório pode ser visto como uma caixa preta.

Oráculo Aleatório

- O oráculo aleatório pode ser visto como uma caixa preta.
- Ele recebe como entrada uma cadeia de *bits* de tamanho variado e retorna uma cadeia de *bits* de tamanho arbitrário.

Oráculo Aleatório

- O oráculo aleatório pode ser visto como uma caixa preta.
- Ele recebe como entrada uma cadeia de *bits* de tamanho variado e retorna uma cadeia de *bits* de tamanho arbitrário.
- Qualquer um pode interagir com ele, seja uma entidade honesta ou um atacante.

Oráculo Aleatório

- O oráculo aleatório pode ser visto como uma caixa preta.
- Ele recebe como entrada uma cadeia de *bits* de tamanho variado e retorna uma cadeia de *bits* de tamanho arbitrário.
- Qualquer um pode interagir com ele, seja uma entidade honesta ou um atacante.
- O modelo do oráculo aleatório foi introduzido em 1993 por Mihir Bellare e Phillip Rogaway e fornece uma metodologia formal que pode ser usada no projeto e validação de esquemas criptográficos.

Paradoxo do Aniversário

- É um problema clássico que demonstra como resultados em probabilidade podem ser contra-intuitivos para o cérebro humano.

Paradoxo do Aniversário

- É um problema clássico que demonstra como resultados em probabilidade podem ser contra-intuitivos para o cérebro humano.

Definição

Qual é o menor valor de n para que a probabilidade de duas pessoas em uma sala com n pessoas compartilhem a mesma data de aniversário com probabilidade maior que 50%?

Paradoxo do Aniversário

- É um problema clássico que demonstra como resultados em probabilidade podem ser contra-intuitivos para o cérebro humano.

Definição

Qual é o menor valor de n para que a probabilidade de duas pessoas em uma sala com n pessoas compartilhem a mesma data de aniversário com probabilidade maior que 50%?

- Fazendo umas continhas, é possível ver que serão 'aproximadamente' $1,17 * \sqrt{365}$.

Paradoxo do Aniversário

- É um problema clássico que demonstra como resultados em probabilidade podem ser contra-intuitivos para o cérebro humano.

Definição

Qual é o menor valor de n para que a probabilidade de duas pessoas em uma sala com n pessoas compartilhem a mesma data de aniversário com probabilidade maior que 50%?

- Fazendo umas continhas, é possível ver que serão 'aproximadamente' $1,17 * \sqrt{365}$.
- Se estendermos o conceito para funções de resumo, temos que a probabilidade de encontrarmos uma colisão em uma função de resumo gira em torno de raiz quadrada do tamanho da saída da função.

Nível de segurança atingível pelas funções de resumo.

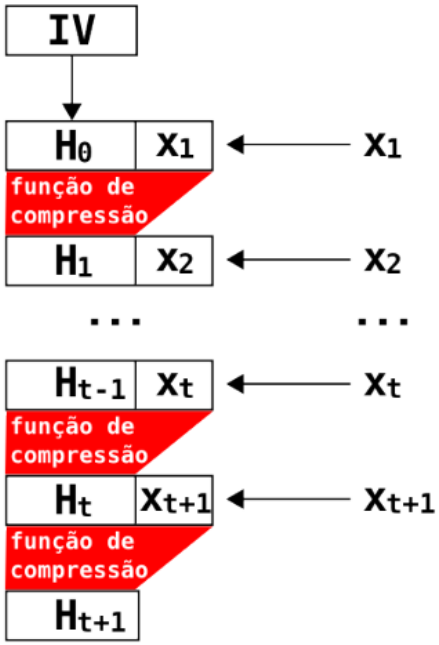
Propriedades	Nível de segurança em *bits*
Primeira pré-imagem	n
Segunda pré-imagem	n
Colisão	$\frac{n}{2}$

Construção Merkle-Damgård

- Uma forma de construir funções de resumo é projetar uma função de compressão resistente à colisão com tamanho de entrada fixo e então usar extensão de domínio para criar uma função com tamanho de entrada arbitrário.

Construção Merkle-Damgård

- Uma forma de construir funções de resumo é projetar uma função de compressão resistente à colisão com tamanho de entrada fixo e então usar extensão de domínio para criar uma função com tamanho de entrada arbitrário.
- Esse método foi usado no projeto do MD5, SHA-1 e SHA-2.



Construção Esponja

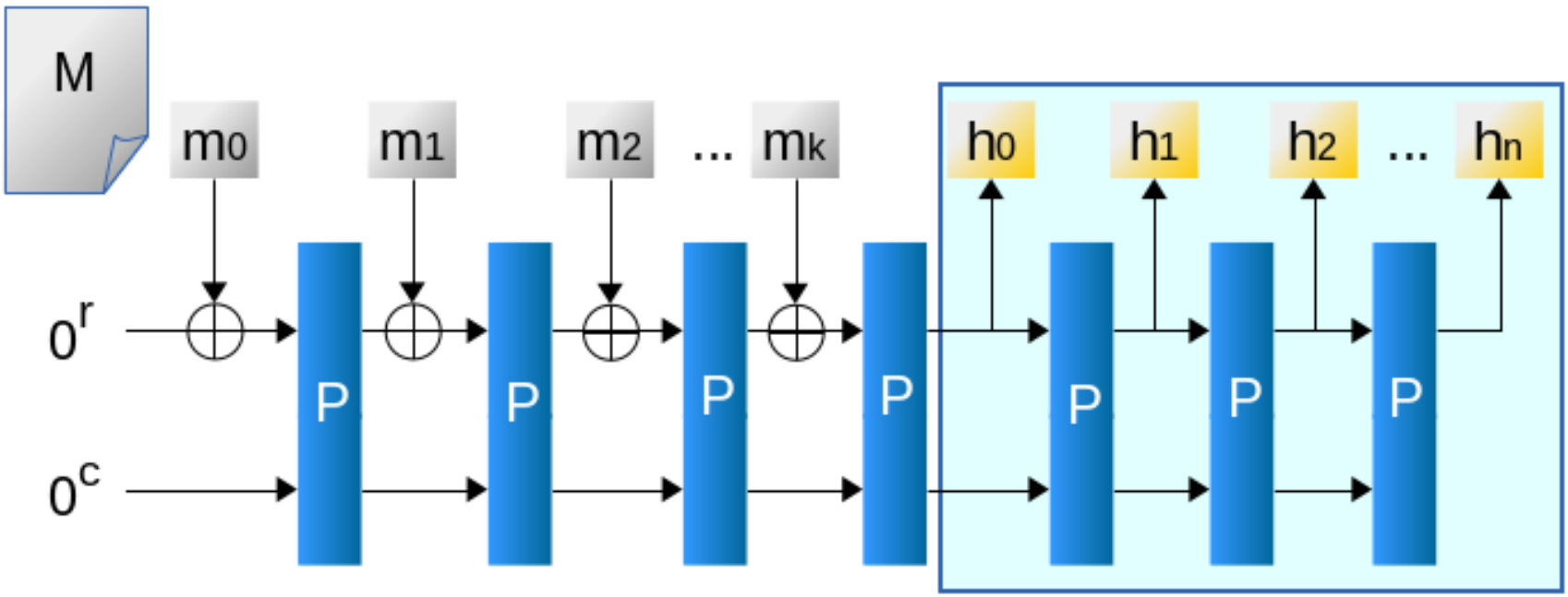
Definição

A construção esponja é um modo de operação baseado em uma permutação de tamanho fixo e uma regra de preenchimento que constrói uma função que mapeia entradas de tamanho arbitrário em saídas de tamanho variado.

Construção Esponja

Definição

A construção esponja é um modo de operação baseado em uma permutação de tamanho fixo e uma regra de preenchimento que constrói uma função que mapeia entradas de tamanho arbitrário em saídas de tamanho variado.



SHA-1

- E 1993, o NIST anuncia um algoritmo padrão para funções de resumo criptográfico denominado *Standart Hash Algorithm*.

SHA-1

- E 1993, o NIST anuncia um algoritmo padrão para funções de resumo criptográfico denominado *Standart Hash Algorithm*.
- O algoritmo foi projetado pela NSA e usa a construção Merkle-Damgård.

SHA-1

- E 1993, o NIST anuncia um algoritmo padrão para funções de resumo criptográfico denominado *Standart Hash Algorithm*.
- O algoritmo foi projetado pela NSA e usa a construção Merkle-Damgård.
- Pouco tempo depois, foi anunciada a anulação do algoritmo e, em 1995, outro algoritmo é publicado, o SHA-1.

SHA-1

- E 1993, o NIST anuncia um algoritmo padrão para funções de resumo criptográfico denominado *Standart Hash Algorithm*.
- O algoritmo foi projetado pela NSA e usa a construção Merkle-Damgård.
- Pouco tempo depois, foi anunciada a anulação do algoritmo e, em 1995, outro algoritmo é publicado, o SHA-1.
- O primeiro algoritmo passou a ser chamado pela comunidade como SHA-0.

SHA-1

- E 1993, o NIST anuncia um algoritmo padrão para funções de resumo criptográfico denominado *Standart Hash Algorithm*.
- O algoritmo foi projetado pela NSA e usa a construção Merkle-Damgård.
- Pouco tempo depois, foi anunciada a anulação do algoritmo e, em 1995, outro algoritmo é publicado, o SHA-1.
- O primeiro algoritmo passou a ser chamado pela comunidade como SHA-0.
- A Diferença do SHA-1 para o SHA-0 é apenas uma rotação na etapa inicial do processamento das palavras que compõem um bloco da mensagem.

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224
  - SHA-256

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224
  - SHA-256
  - SHA-384

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- O SHA-2 pode ser descrito como duas etapas:

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- O SHA-2 pode ser descrito como duas etapas:
  - Pré-processamento, onde a mensagem é preenchida e dividida em blocos de 512 bits (ou 1024, dependendo do algoritmo).

SHA-2

- Com suspeitas cada vez maiores de vulnerabilidade no SHA-1 e com o advento de novas tecnologias, a NSA projetou em 2001 a versão seguinte da família SHA.
- O SHA-2 é composto por quatro algoritmos:
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- O SHA-2 pode ser descrito como duas etapas:
  - Pré-processamento, onde a mensagem é preenchida e dividida em blocos de 512 bits (ou 1024, dependendo do algoritmo).
  - Calculo do valor de resumo.

Concurso SHA-3

- Foram recebidas 64 propostas de algoritmos, dentre as quais 51 foram selecionadas para a primeira etapa, em Outubro de 2008.

Concurso SHA-3

- Foram recebidas 64 propostas de algoritmos, dentre as quais 51 foram selecionadas para a primeira etapa, em Outubro de 2008.
- Para a segunda etapa, que teve seu início em Julho de 2009, foram selecionados 14 candidatos.

Concurso SHA-3

- Foram recebidas 64 propostas de algoritmos, dentre as quais 51 foram selecionadas para a primeira etapa, em Outubro de 2008.
- Para a segunda etapa, que teve seu início em Julho de 2009, foram selecionados 14 candidatos.
- Para a terceira e última etapa, iniciada em 2011, foram selecionados cinco candidatos: BLAKE, Grøstl, JH, Keccak e Skein.

Concurso SHA-3

- Foram recebidas 64 propostas de algoritmos, dentre as quais 51 foram selecionadas para a primeira etapa, em Outubro de 2008.
- Para a segunda etapa, que teve seu início em Julho de 2009, foram selecionados 14 candidatos.
- Para a terceira e última etapa, iniciada em 2011, foram selecionados cinco candidatos: BLAKE, Grøstl, JH, Keccak e Skein.
- Em 2012, Keccak foi escolhido como o vencedor.

Concurso SHA-3

- Foram recebidas 64 propostas de algoritmos, dentre as quais 51 foram selecionadas para a primeira etapa, em Outubro de 2008.
- Para a segunda etapa, que teve seu início em Julho de 2009, foram selecionados 14 candidatos.
- Para a terceira e última etapa, iniciada em 2011, foram selecionados cinco candidatos: BLAKE, Grøstl, JH, Keccak e Skein.
- Em 2012, Keccak foi escolhido como o vencedor.
- O Keccak baseia-se na construção esponja e foi criado por Guido Bertoni *et al.* [**?**].

Concurso SHA-3

- Foram recebidas 64 propostas de algoritmos, dentre as quais 51 foram selecionadas para a primeira etapa, em Outubro de 2008.
- Para a segunda etapa, que teve seu início em Julho de 2009, foram selecionados 14 candidatos.
- Para a terceira e última etapa, iniciada em 2011, foram selecionados cinco candidatos: BLAKE, Grøstl, JH, Keccak e Skein.
- Em 2012, Keccak foi escolhido como o vencedor.
- O Keccak baseia-se na construção esponja e foi criado por Guido Bertoni *et al.* [**?**].
- O processo de padronização teve fim em Agosto de 2015, culminando no documento FIPS 202 [**?**].

0.1

Construção Esponja

Construção Esponja

- A Construção Esponja foi criada por Guido Bertoni, Joan Daemen, Michaël Peeters e Gilles Van Assche.

Construção Esponja

- A Construção Esponja foi criada por Guido Bertoni, Joan Daemen, Michaël Peeters e Gilles Van Assche.
- É uma função de mapeamento que recebe como entrada dados de comprimento variável e gera como saída uma mensagem também de comprimento variável.

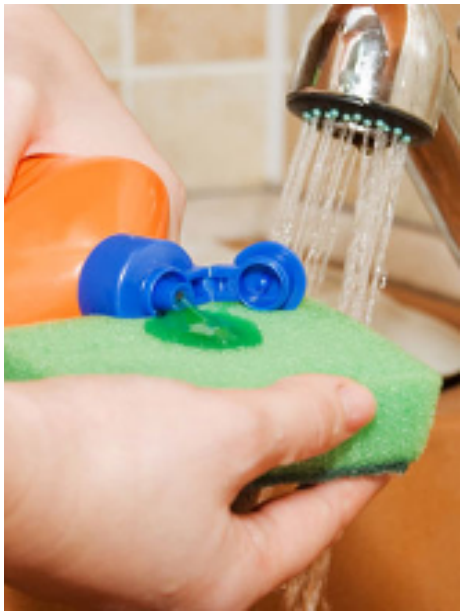
Construção Esponja

- A Construção Esponja foi criada por Guido Bertoni, Joan Daemen, Michaël Peeters e Gilles Van Assche.
- É uma função de mapeamento que recebe como entrada dados de comprimento variável e gera como saída uma mensagem também de comprimento variável.
- Baseia-se em uma permutação de comprimento fixo, uma regra de preenchimento e uma taxa de bits r.

Construção Esponja

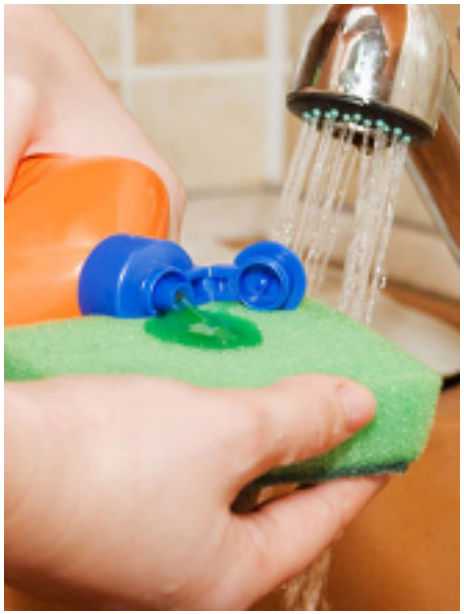
- A Construção Esponja foi criada por Guido Bertoni, Joan Daemen, Michaël Peeters e Gilles Van Assche.
- É uma função de mapeamento que recebe como entrada dados de comprimento variável e gera como saída uma mensagem também de comprimento variável.
- Baseia-se em uma permutação de comprimento fixo, uma regra de preenchimento e uma taxa de bits r.
- A função esponja pode ser usada para modelar ou implementar a maioria das primitivas criptográficas, incluindo função hash, códigos de autenticação de mensagens, cifra de fluxo, cifra de bloco, gerador de números pseudoaleatórios e autenticação.

Construção Esponja



Fase de Inicialização

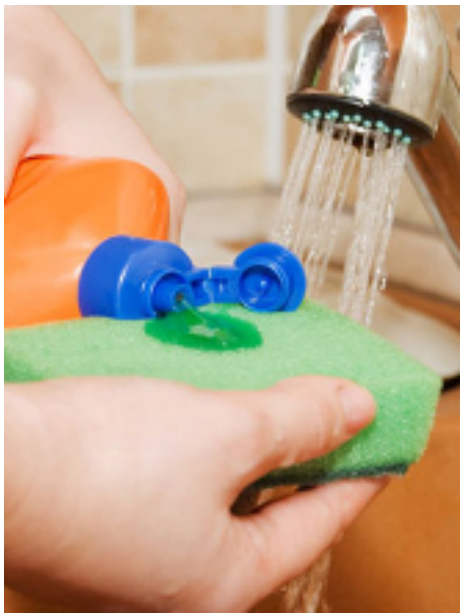
Construção Esponja



Fase de Inicialização

Fase de Absorção

Construção Esponja

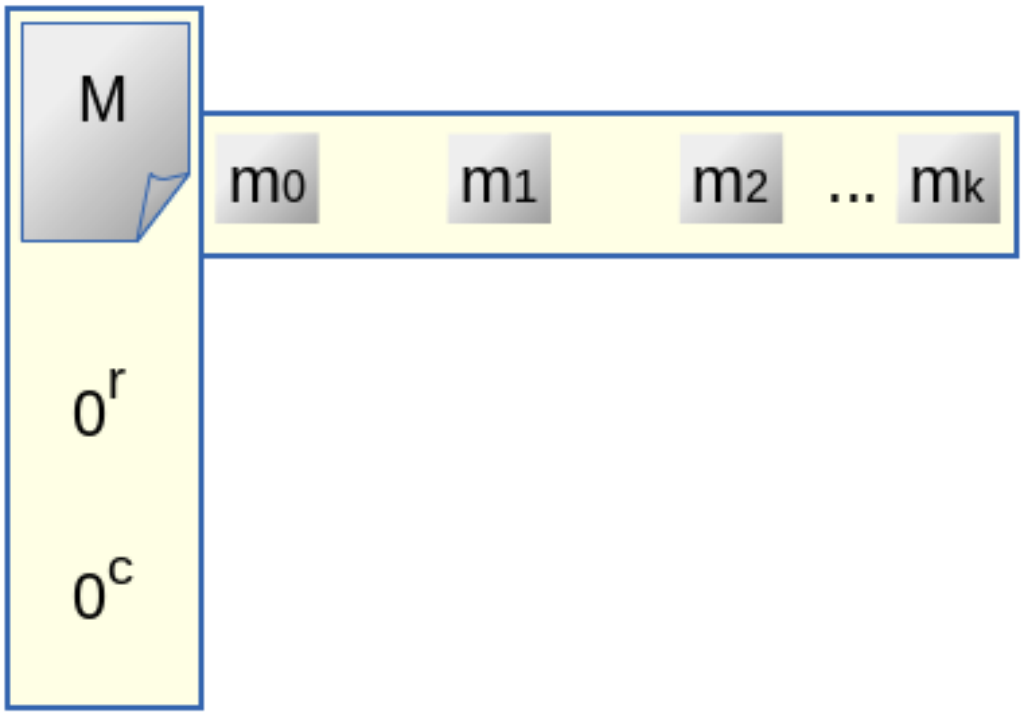


Fase de Inicialização

Fase de Absorção

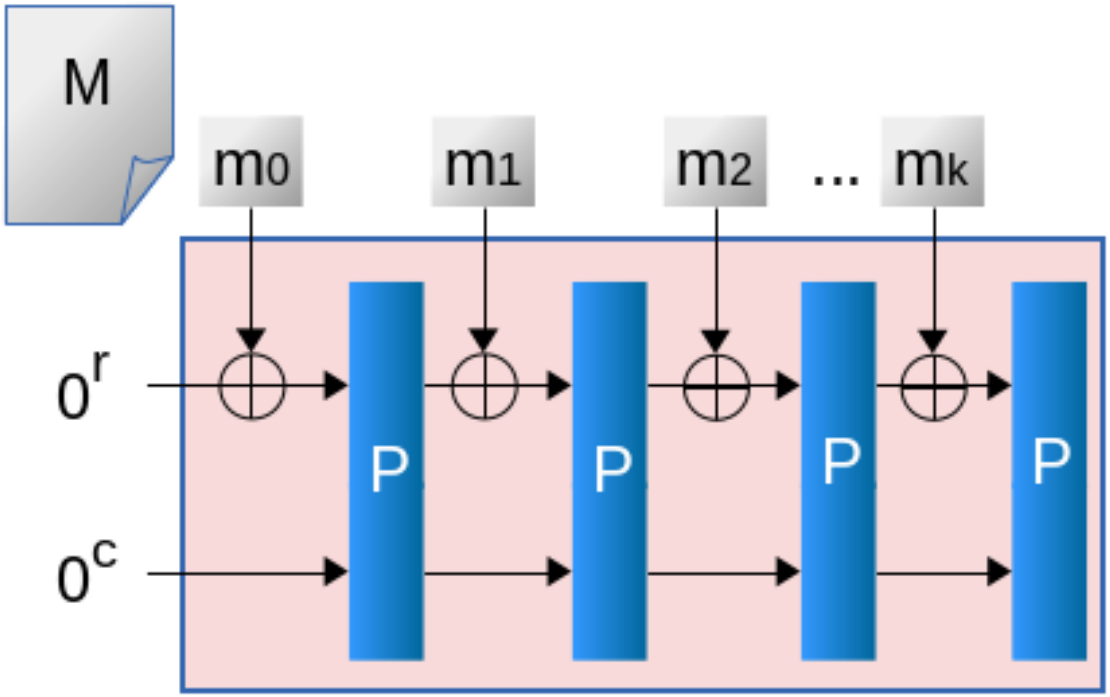
Fase de extração

Construção Esponja



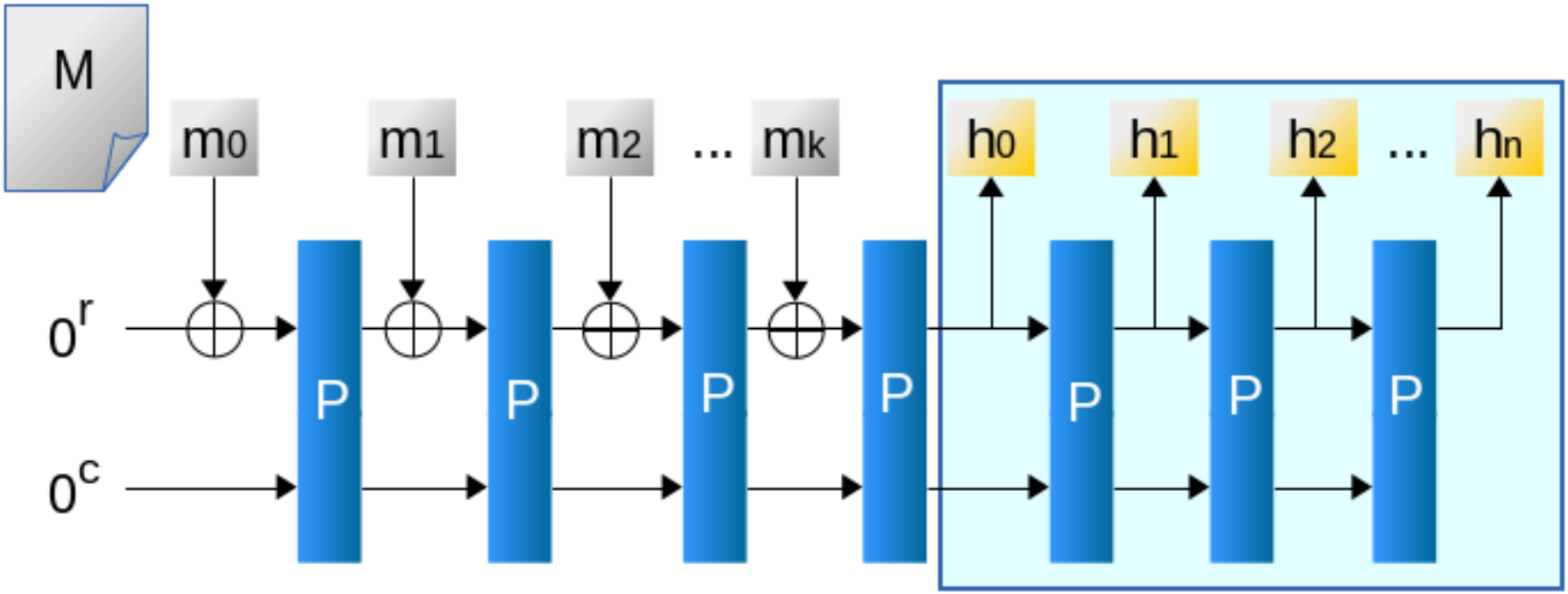
Fase de inicialização: O estado é inicializado com zeros e a mensagem é dividida em k blocos de r bits.

Construção Esponja



Fase de Absorção: é aplicada uma operação XOR entre um bloco m_i com os primeiros r bits do estado atual. Então, o estado é atualizado por meio de uma permutação P.

Construção Esponja



Fase de extração: os primeiros r bits do estado são usados como saída;

0.2

Família SHA-3

Família SHA-3

- A família SHA-3 é composta por quatro funções de resumo e duas funções com saída variável (XOF - *Extendable-Output Function*).

Família SHA-3

- A família SHA-3 é composta por quatro funções de resumo e duas funções com saída variável (XOF - *Extendable-Output Function*).

Funções	Taxa de bits (r)	Capacidade (c)	Nível de segurança
SHA-3$_{224}$	1,152	448	112
SHA-3$_{256}$	1,088	512	128
SHA-3$_{384}$	832	768	192
SHA-3$_{512}$	576	1,024	256
SHAKE$_{128}$	1,344	256	$\min(d/2, 128)$
SHAKE$_{256}$	1,088	512	$\min(d/2, 256)$

Nas funções XOF, d representa o número de bits produzidos.

Funções com saída variável (XOF)

- Uma função com saída variável mapeia uma cadeia de bits de tamanho arbitrário em uma cadeia de bits de tamanho variável d.

Funções com saída variável (XOF)

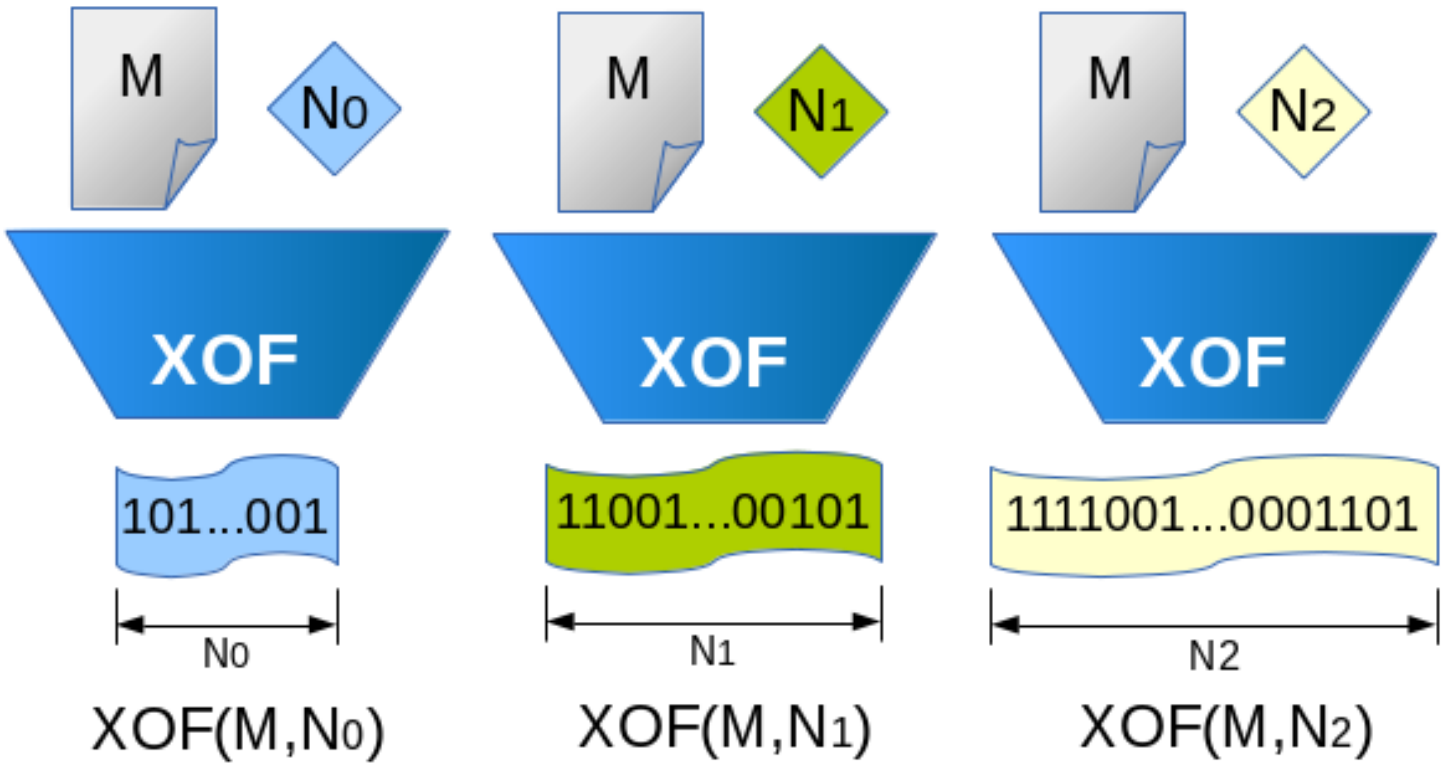
- Uma função com saída variável mapeia uma cadeia de bits de tamanho arbitrário em uma cadeia de bits de tamanho variável d.

$$\text{XOF} : \{0,1\}^* \times d \in \mathbb{N} \longrightarrow \{0,1\}^d$$

Funções com saída variável (XOF)

- Uma função com saída variável mapeia uma cadeia de bits de tamanho arbitrário em uma cadeia de bits de tamanho variável d.

$$\text{XOF} : \{0,1\}^* \times d \in \mathbb{N} \longrightarrow \{0,1\}^d$$



0.3

Função de Permutação P

Função de Permutação P

O estado S é composto por $r + c = 1600$ bits e pode ser representado por uma matriz de $5 \times 5 \times 64$ bits.

Função de Permutação P

O estado S é composto por $r + c = 1600$ bits e pode ser representado por uma matriz de 5 $\times$ 5 $\times$ 64 bits.

$$S = \begin{bmatrix} s_0 & s_1 & s_2 & s_3 & s_4 \\ s_5 & s_6 & s_7 & s_8 & s_9 \\ s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \end{bmatrix}; \; S[x, y] = s_{5x+y} \text{ para } 0 \leq x, y < 5.$$

Função de Permutação P

$$P : S \longrightarrow S'$$

Função de Permutação P

$$P : S \longrightarrow S'$$

$$\theta$$

$$\iota \qquad\qquad \rho$$

$$\chi \qquad\qquad \pi$$

Etapas de mapeamento

Entrada: Estado S e i_r o índice da rodada.
Salida: Estado S atualizado.

Etapa de mapeamento θ

Etapas de mapeamento ρ e π

Etapa de mapeamento χ

Etapa de mapeamento ι

Etapa de mapeamento ι

1: **return** S

Etapa de mapeamento θ

- Neste etapa, é aplicada uma operação XOR de cada elemento i com a paridade da coluna $i-1$ e a paridade da coluna $i+1$ rotada um bit.

Etapa de mapeamento θ

- Neste etapa, é aplicada uma operação XOR de cada elemento i com a paridade da coluna $i-1$ e a paridade da coluna $i+1$ rotada um bit.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

Etapa de mapeamento θ

- Neste etapa, é aplicada uma operação XOR de cada elemento i com a paridade da coluna $i-1$ e a paridade da coluna $i+1$ rotada um bit.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$P_{i-1}$$

Etapa de mapeamento θ

- Neste etapa, é aplicada uma operação XOR de cada elemento i com a paridade da coluna $i-1$ e a paridade da coluna $i+1$ rotada um bit.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{P_{i-1}} \oplus \boxed{P'_{i+1}}$$

$$\boxed{P'_{i+1}} = \mathsf{ROT}(P_{i+1}, 1)$$

Etapa de mapeamento θ

- Neste etapa, é aplicada uma operação XOR de cada elemento i com a paridade da coluna $i-1$ e a paridade da coluna $i+1$ rotada um bit.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$s_{12} \oplus P_{i-1} \oplus P'_{i+1} = s_{12'}$$

$$P'_{i+1} = \mathsf{ROT}(P_{i+1}, 1)$$

Etapa de mapeamento θ

- Neste etapa, é aplicada uma operação XOR de cada elemento i com a paridade da coluna $i-1$ e a paridade da coluna $i+1$ rotada um bit.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12'} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$s_{12} \oplus P_{i-1} \oplus P'_{i+1} = s_{12'}$$

$$P'_{i+1} = \text{ROT}(P_{i+1}, 1)$$

Etapas de mapeamento

Entrada: Estado S e i_r o índice da rodada.
Salida: Estado S atualizado.

Etapa de mapeamento θ

1: $C_y = s_{0+y} \oplus s_{5+y} \oplus s_{10+y} \oplus s_{15+y} \oplus s_{20+y}$ para $0 \leq y < 5$
2: $D_x = C_{(x-1) \bmod 5} \oplus (\mathsf{ROT}(C_{(x+1) \bmod 5}, 1))$ para $0 \leq x < 5$
3: $s_{5x+y} = s_{5x+y} \oplus D_x$ para $0 \leq x, y < 5$

Etapas de mapeamento ρ e π

Etapa de mapeamento χ

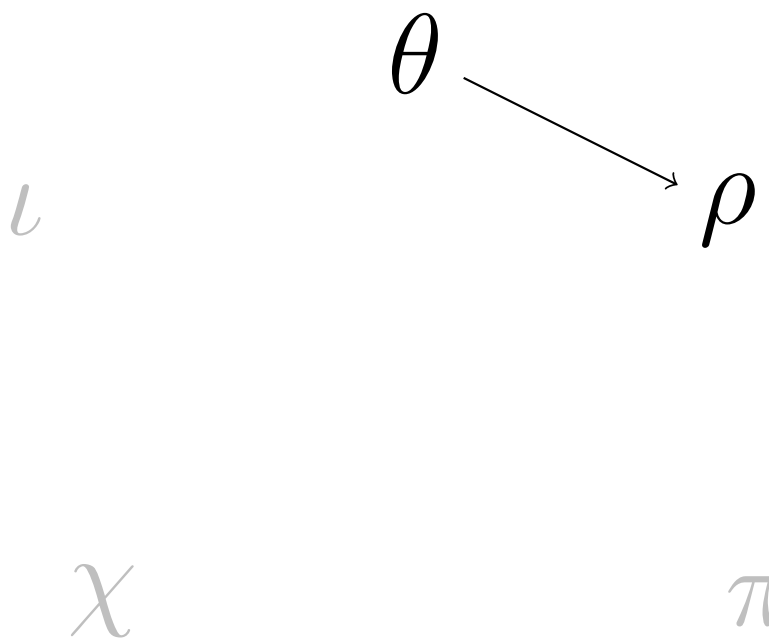
Etapa de mapeamento ι

Etapa de mapeamento ι

4: **return** S

Etapa de mapeamento ρ

$$P : S \longrightarrow S'$$



Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{0'}} = \mathsf{ROT}(s_0, 0)$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline s_{0'} \\ \hline \end{array}$$

$$S' =$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$s_{1'} = \mathsf{ROT}(s_1, 1)$$

$$s_{0'}$$

$$S' =$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|} \hline s_{0'} & s_{1'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|}\hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{2'}} = \mathsf{ROT}(s_2, 62)$$

$$S' = \begin{array}{|c|c|}\hline s_{0'} & s_{1'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|c|} \hline s_{0'} & s_{1'} & s_{2'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|}\hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{3'}} = \mathsf{ROT}(s_3, 28)$$

$$S' = \begin{array}{|c|c|c|}\hline s_{0'} & s_{1'} & s_{2'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|c|c|} \hline s_{0'} & s_{1'} & s_{2'} & s_{3'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{4'}} = \mathsf{ROT}(s_4, 27)$$

$$S' = \begin{array}{|c|c|c|c|} \hline s_{0'} & s_{1'} & s_{2'} & s_{3'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|c|c|c|} \hline s_{0'} & s_{1'} & s_{2'} & s_{3'} & s_{4'} \\ \hline \end{array}$$

Etapa de mapeamento ρ

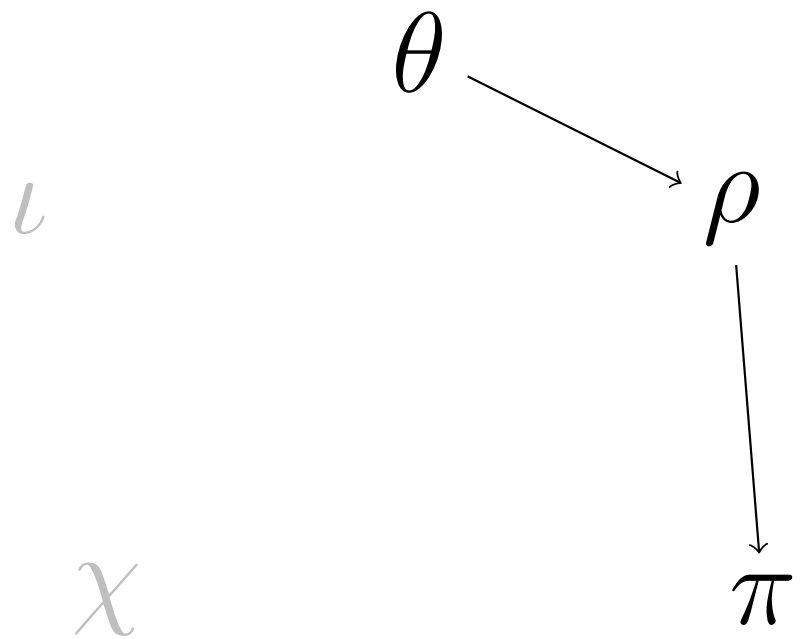
- Nesta etapa, cada palavra do estado será rodada uma quantidade fixa de bits.

$$S = \begin{array}{|c|c|c|c|c|}\hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|c|c|c|}\hline s_{0'} & s_{1'} & s_{2'} & s_{3'} & s_{4'} \\ \hline s_{5'} & s_{6'} & s_{7'} & s_{8'} & s_{9'} \\ \hline s_{10'} & s_{11'} & s_{12'} & s_{13'} & s_{14'} \\ \hline s_{15'} & s_{16'} & s_{17'} & s_{18'} & s_{19'} \\ \hline s_{20'} & s_{21'} & s_{22'} & s_{23'} & s_{24'} \\ \hline \end{array}$$

Etapa de mapeamento π

$$P : S \longrightarrow S'$$



Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{0'}} = \pi(s_0) = \boxed{s_0}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline s_0 \\ \hline \end{array}$$

$$S' =$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{10'}} = \pi(s_1) = \boxed{s_1}$$

$$\boxed{s_0}$$

$$S' =$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|} \hline s_0 \\ \hline \end{array} \quad \begin{array}{|c|} \hline s_1 \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{20'}} = \pi(s_2) = \boxed{s_2}$$

$$\boxed{s_0}$$

$$S' = \boxed{s_1}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|} \hline s_0 \\ \hline \end{array} \quad \begin{array}{|c|} \hline s_1 \\ \hline \end{array} \quad \begin{array}{|c|} \hline s_2 \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$s_{5'} = \pi(s_3) = s_3$$

$$S' = \begin{array}{|c|} \hline s_0 \\ \hline s_1 \\ \hline s_2 \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|} \hline s_0 \\ \hline s_3 \\ \hline s_1 \\ \hline \end{array}$$

$$\begin{array}{|c|} \hline s_2 \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$\boxed{s_{15'}} = \pi(s_4) = \boxed{s_4}$$

$$S' = \begin{array}{|c|} \hline s_0 \\ \hline s_3 \\ \hline s_1 \\ \hline \end{array}$$

$$\boxed{s_2}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|} \hline s_0 \\ \hline s_3 \\ \hline s_1 \\ \hline s_4 \\ \hline s_2 \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|c|c|c|} \hline s_0 & s_6 & s_{12} & s_{18} & s_{24} \\ \hline s_3 & s_9 & s_{10} & s_{16} & s_{22} \\ \hline s_1 & s_7 & s_{13} & s_{19} & s_{20} \\ \hline s_4 & s_5 & s_{11} & s_{17} & s_{23} \\ \hline s_2 & s_8 & s_{14} & s_{15} & s_{21} \\ \hline \end{array}$$

Etapa de mapeamento π

- Esta etapa consiste em embaralhar as palavras no estado.

$$S = \begin{array}{|c|c|c|c|c|}\hline s_0 & s_1 & s_2 & s_3 & s_4 \\ \hline s_5 & s_6 & s_7 & s_8 & s_9 \\ \hline s_{10} & s_{11} & s_{12} & s_{13} & s_{14} \\ \hline s_{15} & s_{16} & s_{17} & s_{18} & s_{19} \\ \hline s_{20} & s_{21} & s_{22} & s_{23} & s_{24} \\ \hline \end{array}$$

$$S' = \begin{array}{|c|c|c|c|c|}\hline s_{0'} & s_{1'} & s_{2'} & s_{3'} & s_{4'} \\ \hline s_{5'} & s_{6'} & s_{7'} & s_{8'} & s_{9'} \\ \hline s_{10'} & s_{11'} & s_{12'} & s_{13'} & s_{14'} \\ \hline s_{15'} & s_{16'} & s_{17'} & s_{18'} & s_{19'} \\ \hline s_{20'} & s_{21'} & s_{22'} & s_{23'} & s_{24'} \\ \hline \end{array}$$

Etapas de mapeamento

Entrada: Estado S e i_r o índice da rodada.
Salida: Estado S atualizado.

Etapa de mapeamento θ

1: $C_y = s_{0+y} \oplus s_{5+y} \oplus s_{10+y} \oplus s_{15+y} \oplus s_{20+y}$ para $0 \leq y < 5$
2: $D_x = C_{(x-1) \bmod 5} \oplus (\mathsf{ROT}(C_{(x+1) \bmod 5}, 1))$ para $0 \leq x < 5$
3: $s_{5x+y} = s_{5x+y} \oplus D_x$ para $0 \leq x, y < 5$

Etapas de mapeamento ρ e π

4: $B_{(16x+10y) \bmod 25} = (\mathsf{ROT}(s_{5x+y}, r_{5x+y}))$ para $0 \leq x, y < 5$

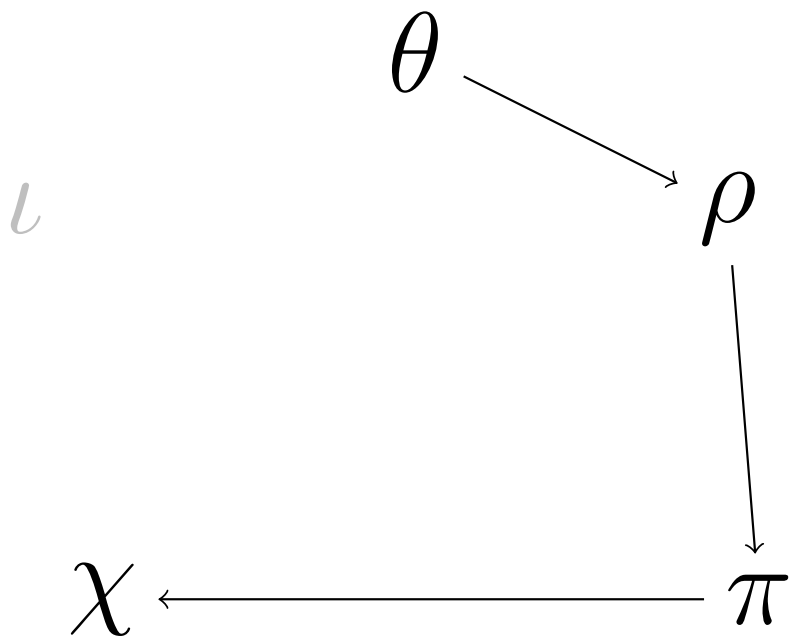
Etapa de mapeamento χ

Etapa de mapeamento ι

5: **return** S

Etapa de mapeamento χ

$$P : S \longrightarrow S'$$



Etapa de mapeamento χ

- Nesta etapa uma função não linear é aplicada entre elementos da mesma linha.

Etapa de mapeamento χ

- Nesta etapa uma função não linear é aplicada entre elementos da mesma linha.

$$
\begin{aligned}
s'_0 &= s_0 \oplus (\neg s_1 \wedge s_2) \\
s'_1 &= s_1 \oplus (\neg s_2 \wedge s_3) \\
s'_2 &= s_2 \oplus (\neg s_3 \wedge s_4) \\
s'_3 &= s_3 \oplus (\neg s_4 \wedge s_0) \\
s'_4 &= s_4 \oplus (\neg s_0 \wedge s_1)
\end{aligned}
$$

Etapas de mapeamento

Entrada: Estado S e i_r o índice da rodada.
Salida: Estado S atualizado.

Etapa de mapeamento θ

1: $C_y = s_{0+y} \oplus s_{5+y} \oplus s_{10+y} \oplus s_{15+y} \oplus s_{20+y}$ para $0 \leq y < 5$
2: $D_x = C_{(x-1) \bmod 5} \oplus (\mathsf{ROT}(C_{(x+1) \bmod 5}, 1))$ para $0 \leq x < 5$
3: $s_{5x+y} = s_{5x+y} \oplus D_x$ para $0 \leq x, y < 5$

Etapas de mapeamento ρ e π

4: $B_{(16x+10y) \bmod 25} = (\mathsf{ROT}(s_{5x+y}, r_{5x+y}))$ para $0 \leq x, y < 5$

Etapa de mapeamento χ

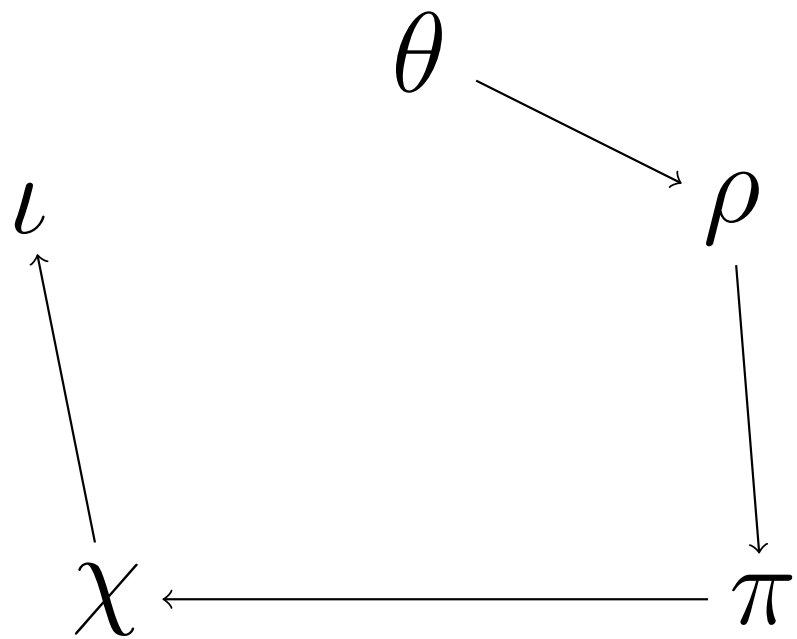
5: **for all** (x, y) tal que $0 \leq x, y < 5$ **do**
6: $\quad T = (B_{5x+(y+2) \bmod 5}) \wedge (\neg B_{5x+(y+1) \bmod 5})$
7: $\quad s_{5x+y} = B_{5x+y} \oplus T$

Etapa de mapeamento ι

8: **return** S

Etapa de mapeamento ι

$$P : S \longrightarrow S'$$



Etapa de mapeamento ι

- Nesta etapa, uma operação XOR é aplicada entre o primeiro elemento do estado e uma constante.

Etapa de mapeamento ι

- Nesta etapa, uma operação XOR é aplicada entre o primeiro elemento do estado e uma constante.

$$\boxed{s'_0} = \boxed{s_0} \oplus \boxed{\text{const}}$$

Pseudocódigo da permutação P

Entrada: Estado S e i_r o índice da rodada.
Salida: Estado S atualizado.

Etapa de mapeamento θ

1: $C_y = s_{0+y} \oplus s_{5+y} \oplus s_{10+y} \oplus s_{15+y} \oplus s_{20+y}$ para $0 \leq y < 5$
2: $D_x = C_{(x-1) \bmod 5} \oplus (\mathsf{ROT}(C_{(x+1) \bmod 5}, 1))$ para $0 \leq x < 5$
3: $s_{5x+y} = s_{5x+y} \oplus D_x$ para $0 \leq x, y < 5$

Etapas de mapeamento ρ e π

4: $B_{(16x+10y) \bmod 25} = (\mathsf{ROT}(s_{5x+y}, r_{5x+y}))$ para $0 \leq x, y < 5$

Etapa de mapeamento χ

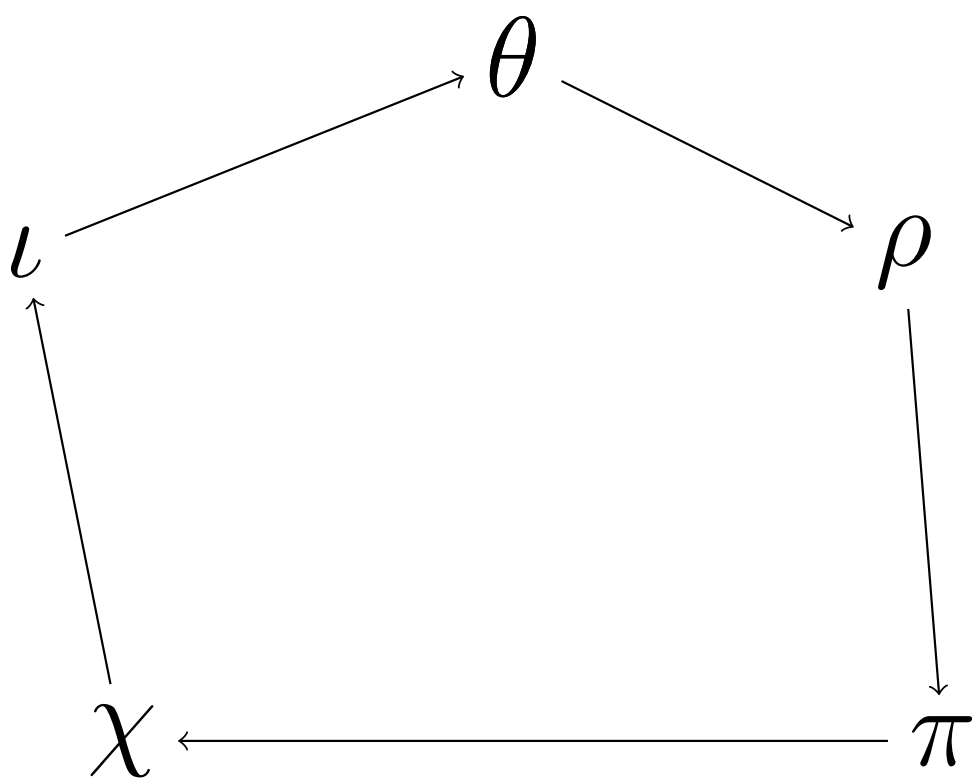
5: **for all** (x, y) tal que $0 \leq x, y < 5$ **do**
6: $\quad T = (B_{5x+(y+2) \bmod 5}) \wedge (\neg B_{5x+(y+1) \bmod 5})$
7: $\quad s_{5x+y} = B_{5x+y} \oplus T$

Etapa de mapeamento ι

8: $s_0 = s_0 \oplus rc(i_r)$
9: **return** A

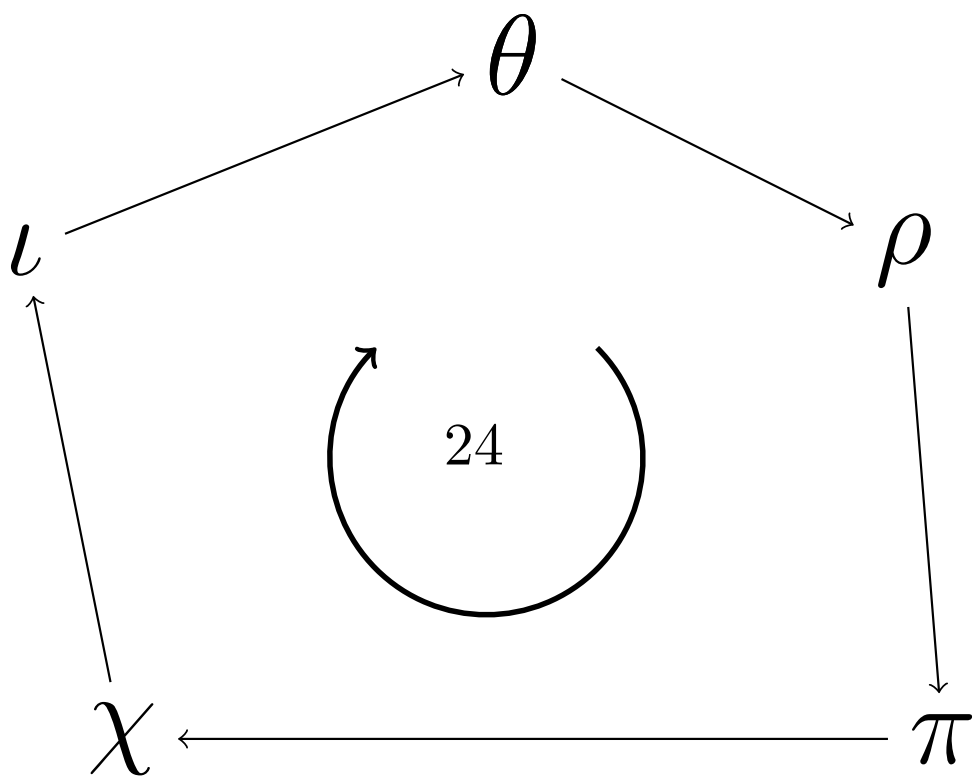
Permutação P

$$P : S \longrightarrow S'$$



Permutação P

$$P : S \longrightarrow S'$$



Oportunidades de otimização

- No pseudocódigo apresentado, além das 25 palavras que representam o estado S, são usadas outras 25 palavras para armazenar o estado após as etapas de mapeamento ρ e π.

Oportunidades de otimização

- No pseudocódigo apresentado, além das 25 palavras que representam o estado S, são usadas outras 25 palavras para armazenar o estado após as etapas de mapeamento ρ e π.
- A maioria das arquiteturas de 64 bits possuem apenas 16 registradores de propósito geral.

Oportunidades de otimização

- No pseudocódigo apresentado, além das 25 palavras que representam o estado S, são usadas outras 25 palavras para armazenar o estado após as etapas de mapeamento ρ e π.
- A maioria das arquiteturas de 64 bits possuem apenas 16 registradores de propósito geral.
- Uma estratégia para acelerar a execução dessa função é processar as etapas de ρ, π e χ de forma modular.

Pseudocódigo modular

Entrada: Estado S depois da etapa de mapeamento θ.
Salida: Estado S depois de processado pelas etapas ρ, π e χ .

1: $T'_0 = s_0 \lll 0$
2: $T'_1 = s_6 \lll 44$
3: $T'_2 = s_{12} \lll 43$
4: $T'_3 = s_{18} \lll 21$
5: $T'_4 = s_{24} \lll 14$
6: $s_0 = T'_0 \oplus (\neg T'_1 \wedge T'_2)$
7: $s_{12} = T'_1 \oplus (\neg T'_2 \wedge T'_3)$
8: $s_{24} = T'_2 \oplus (\neg T'_3 \wedge T'_4)$
9: $s_6 = T'_3 \oplus (\neg T'_4 \wedge T'_0)$
10: $s_{18} = T'_4 \oplus (\neg T'_0 \wedge T'_1)$
11: **return** A

Pseudocódigo modular

Entrada: Estado S depois da etapa de mapeamento θ.
Salida: Estado S depois de processado pelas etapas ρ, π e χ .

1:	$T'_0 = s_0 \lll 0$	$T''_0 = s_3 \lll 28$
2:	$T'_1 = s_6 \lll 44$	$T''_1 = s_9 \lll 20$
3:	$T'_2 = s_{12} \lll 43$	$T''_2 = s_{10} \lll 3$
4:	$T'_3 = s_{18} \lll 21$	$T''_3 = s_{16} \lll 45$
5:	$T'_4 = s_{24} \lll 14$	$T''_4 = s_{22} \lll 61$
6:	$s_0 = T'_0 \oplus (\neg T'_1 \wedge T'_2)$	$s_{16} = T''_0 \oplus (\neg T''_1 \wedge T''_2)$
7:	$s_{12} = T'_1 \oplus (\neg T'_2 \wedge T'_3)$	$s_3 = T''_1 \oplus (\neg T''_2 \wedge T''_3)$
8:	$s_{24} = T'_2 \oplus (\neg T'_3 \wedge T'_4)$	$s_{10} = T''_2 \oplus (\neg T''_3 \wedge T''_4)$
9:	$s_6 = T'_3 \oplus (\neg T'_4 \wedge T'_0)$	$s_{22} = T''_3 \oplus (\neg T''_4 \wedge T''_0)$
10:	$s_{18} = T'_4 \oplus (\neg T'_0 \wedge T'_1)$	$s_9 = T''_4 \oplus (\neg T''_0 \wedge T''_1)$
11:	**return** A	

Pseudocódigo modular

Entrada: Estado S depois da etapa de mapeamento θ.
Salida: Estado S depois de processado pelas etapas ρ, π e χ .

1:	$T'_0 = s_0 \lll 0$	$T''_0 = s_3 \lll 28$	$T'''_0 = s_1 \lll 1 \ldots$
2:	$T'_1 = s_6 \lll 44$	$T''_1 = s_9 \lll 20$	$T'''_1 = s_7 \lll 6 \ldots$
3:	$T'_2 = s_{12} \lll 43$	$T''_2 = s_{10} \lll 3$	$T'''_2 = s_{13} \lll 25 \ldots$
4:	$T'_3 = s_{18} \lll 21$	$T''_3 = s_{16} \lll 45$	$T'''_3 = s_{19} \lll 8 \ldots$
5:	$T'_4 = s_{24} \lll 14$	$T''_4 = s_{22} \lll 61$	$T'''_4 = s_{20} \lll 18 \ldots$
6:	$s_0 = T'_0 \oplus (\neg T'_1 \wedge T'_2)$	$s_{16} = T''_0 \oplus (\neg T''_1 \wedge T''_2)$	$s_7 = T'''_0 \oplus (\neg T'''_1 \wedge T'''_2) \ldots$
7:	$s_{12} = T'_1 \oplus (\neg T'_2 \wedge T'_3)$	$s_3 = T''_1 \oplus (\neg T''_2 \wedge T''_3)$	$s_{19} = T'''_1 \oplus (\neg T'''_2 \wedge T'''_3) \ldots$
8:	$s_{24} = T'_2 \oplus (\neg T'_3 \wedge T'_4)$	$s_{10} = T''_2 \oplus (\neg T''_3 \wedge T''_4)$	$s_1 = T'''_2 \oplus (\neg T'''_3 \wedge T'''_4) \ldots$
9:	$s_6 = T'_3 \oplus (\neg T'_4 \wedge T'_0)$	$s_{22} = T''_3 \oplus (\neg T''_4 \wedge T''_0)$	$s_{13} = T'''_3 \oplus (\neg T'''_4 \wedge T'''_0) \ldots$
10:	$s_{18} = T'_4 \oplus (\neg T'_0 \wedge T'_1)$	$s_9 = T''_4 \oplus (\neg T''_0 \wedge T''_1)$	$s_{20} = T'''_4 \oplus (\neg T'''_0 \wedge T'''_1) \ldots$
11:	**return** A		

Vantagens da implementação modular

- Redução do número de registradores usados.

Vantagens da implementação modular

- Redução do número de registradores usados.
- Melhora significativa na localidade temporal dos dados.

Vantagens da implementação modular

- Redução do número de registradores usados.
- Melhora significativa na localidade temporal dos dados.
- Aumento do paralelismo.

Vantagens da implementação modular

- Redução do número de registradores usados.
- Melhora significativa na localidade temporal dos dados.
- Aumento do paralelismo.

Vantagens da implementação modular

- Redução do número de registradores usados.
- Melhora significativa na localidade temporal dos dados.
- Aumento do paralelismo.

Comparação de desempenho entre as duas versões

Implementação	Ciclos por bytes
Não Otimizada	24,09
Versão Modular	8,99

Funções de Resumo

Universidade Federal do Ceará - Campus Quixadá

Roberto Cabral
rbcabral@ufc.br

05 de Maio de 2023

Auditoria e Segurança de SI