Modes of Operations for Wide-Block Encryption

Palash Sarkar

Indian Statistical Institute, Kolkata

Structure of Presentation

- From block cipher to wide block encryption.
- Construction Ideas.
- Sketches of several constructions.
- Comparative study.

Block Cipher

Definition. $E_K : \{0,1\}^n \to \{0,1\}^n$.

- $K \in \mathcal{K}$;
- for each K, E_K is a permutation of $\{0,1\}^n$;
- good practical examples are known, e.g. AES.

Block Cipher

Definition. $E_K : \{0,1\}^n \to \{0,1\}^n$.

- $K \in \mathcal{K}$;
- for each K, E_K is a permutation of $\{0,1\}^n$;
- good practical examples are known, e.g. AES.

Security.

- Resists "known" attacks.
- Confidence in security grows with time.

Block Cipher

Definition. $E_K : \{0,1\}^n \to \{0,1\}^n$.

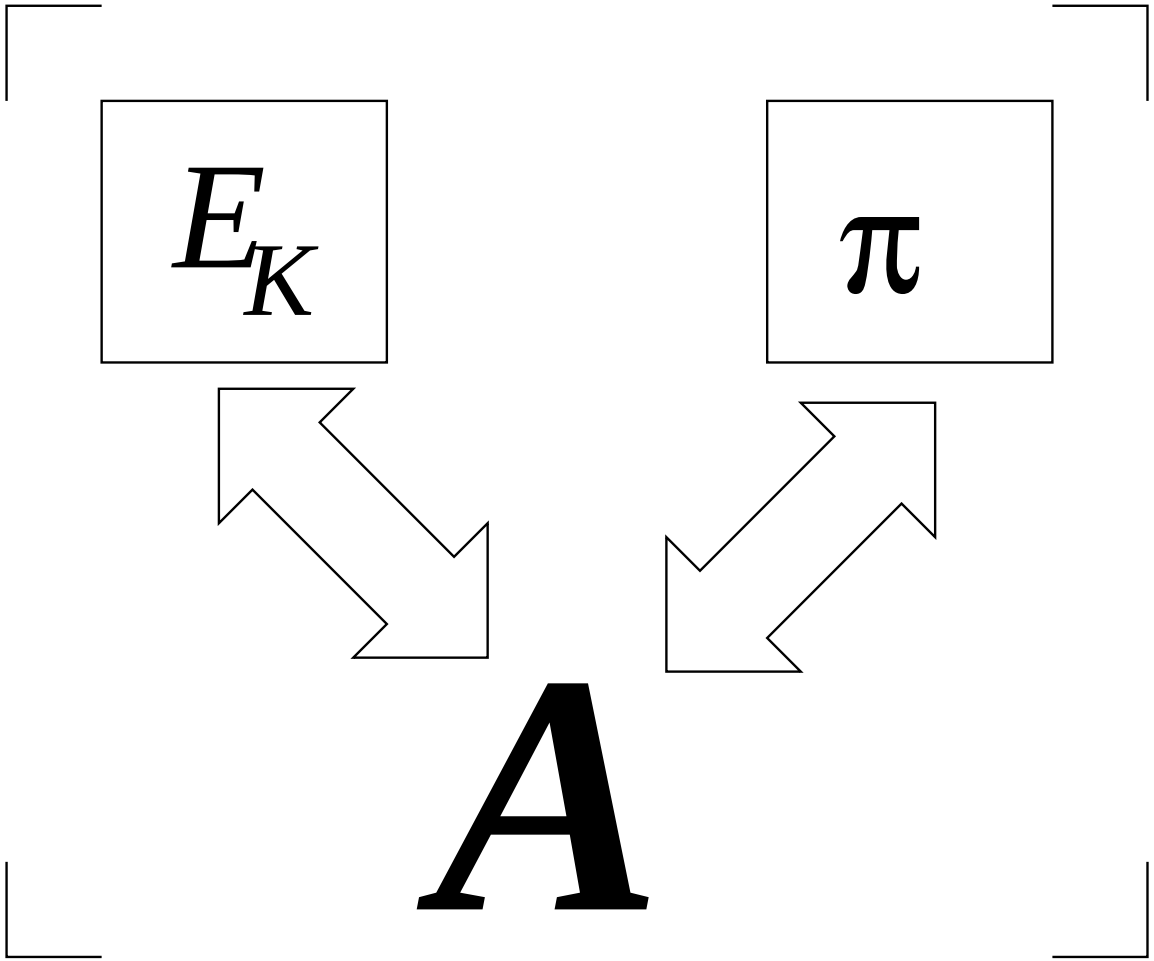
- $K \in \mathcal{K}$;
- for each K, E_K is a permutation of $\{0,1\}^n$;
- good practical examples are known, e.g. AES.

Security.

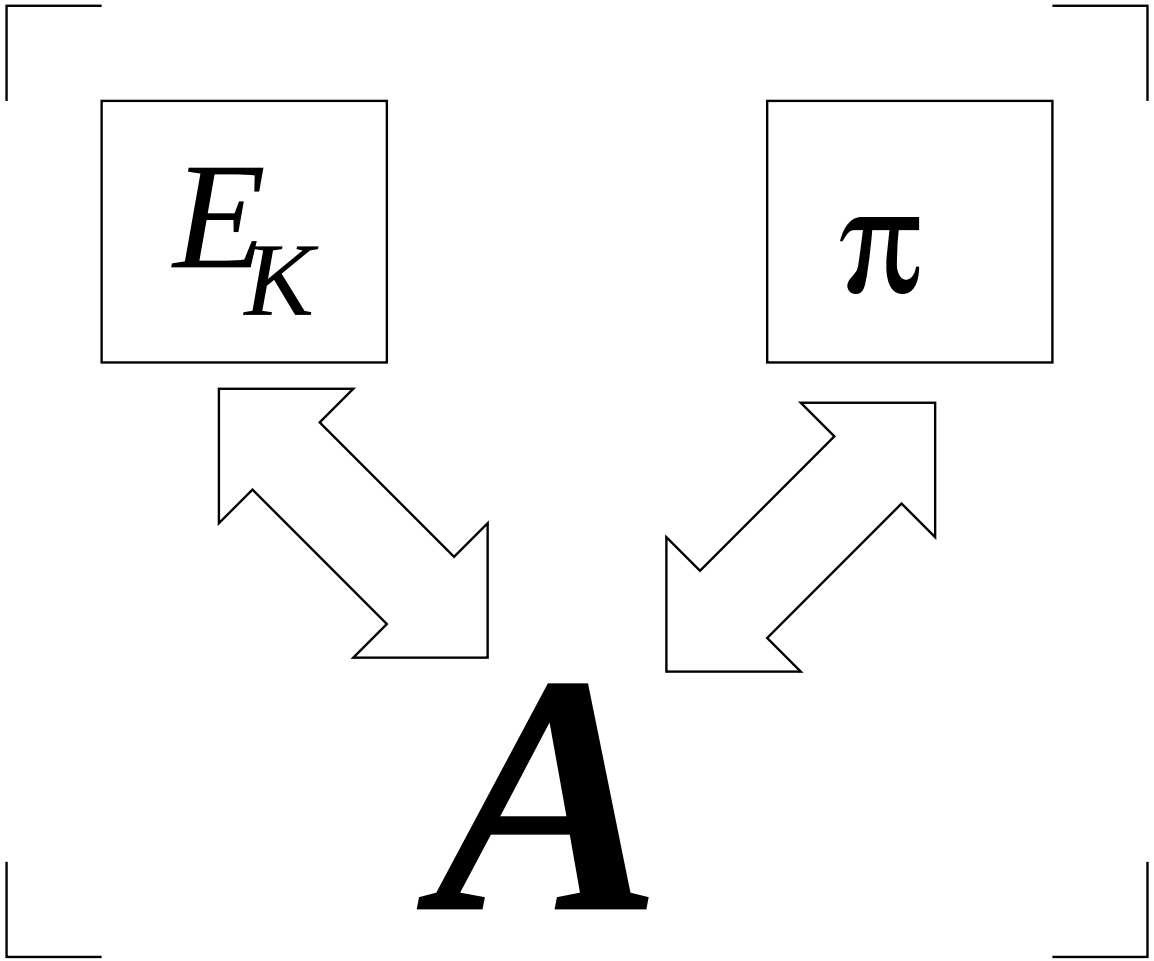
- Resists "known" attacks.
- Confidence in security grows with time.

Formal model. A pseudo-random (unpredictable) permutation (Luby-Rackoff, 1985, 1988).

Pseudo-Random Permutation

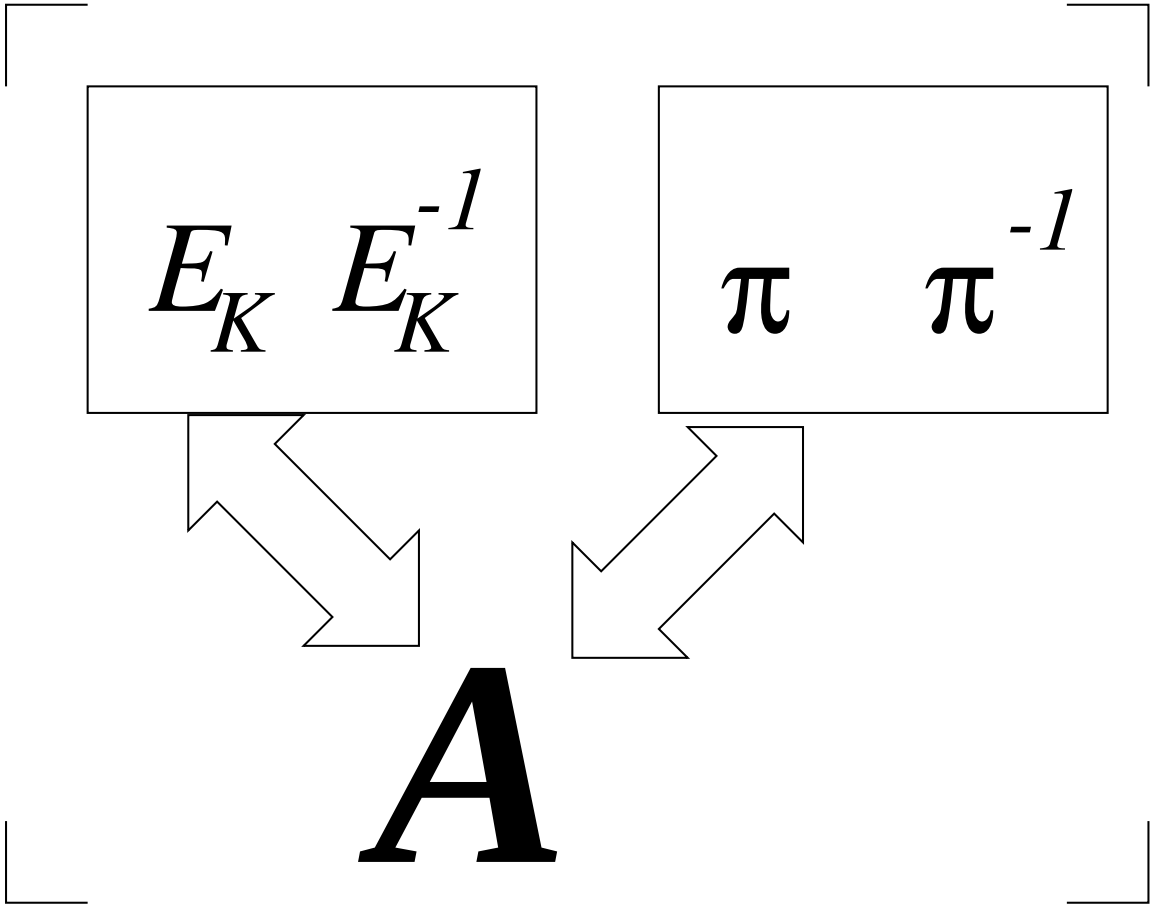


Pseudo-Random Permutation

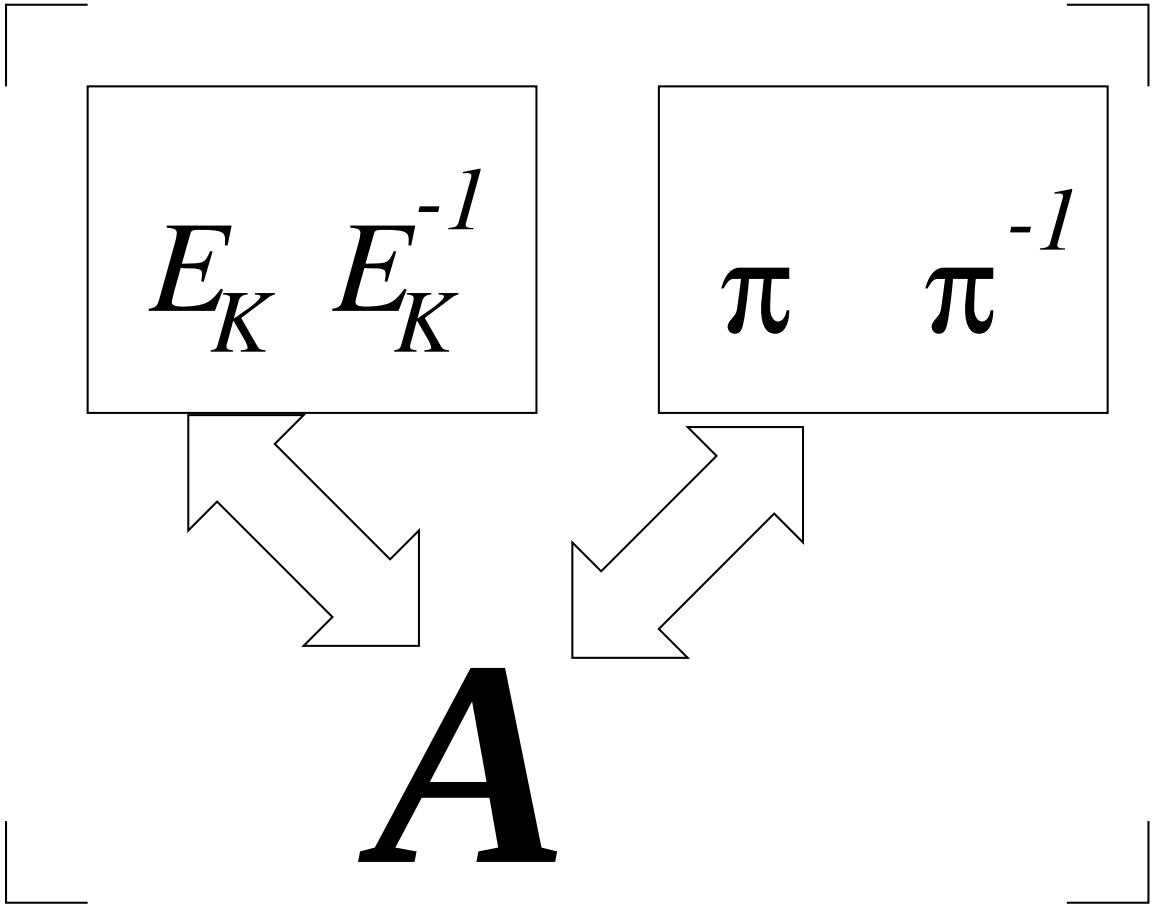


$$\mathsf{Adv}_E^{\mathrm{prp}}(\mathcal{A}) = \Pr_K[\mathcal{A}^{E_K} \to 1] - \Pr_\pi[\mathcal{A}^{\pi} \to 1].$$

Strong PRP



Strong PRP



$$\mathsf{Adv}_E^{\pm\mathrm{prp}}(\mathcal{A}) = \Pr_K[\mathcal{A}^{E_K, E_K^{-1}} \to 1] - \Pr_\pi[\mathcal{A}^{\pi, \pi^{-1}} \to 1].$$

Tweakable Block Cipher

Liskov-Rivest-Wagner (2002).

Tweakable Block Cipher

Liskov-Rivest-Wagner (2002).

$E_K^T : \{0,1\}^n \to \{0,1\}^n$.

- K is the secret key;
- $T \in \mathcal{T}$ is the tweak;
- for each (K, T) pair E_K^T is a permutation of $\{0,1\}^n$;

Tweakable Block Cipher

Liskov-Rivest-Wagner (2002).

$E_K^T : \{0,1\}^n \to \{0,1\}^n$.

- K is the secret key;
- $T \in \mathcal{T}$ is the tweak;
- for each (K, T) pair E_K^T is a permutation of $\{0,1\}^n$;

Tweakable (S)PRP.

- The tweak (unlike the key) is assumed to be known to the adversary.
- In its queries, the adversary can
  - choose a tweak;
  - reuse a tweak (with another message).

Modes of Operations

A block cipher can encrypt short and fixed length strings.

Modes of Operations

A block cipher can encrypt short and fixed length strings.

Applications require the encryption of long strings of possibly different lengths.

Modes of Operations

A block cipher can encrypt short and fixed length strings.

Applications require the encryption of long strings of possibly different lengths.

Applications have different goals.

- Privacy.
- Authentication.

Modes of Operations

A block cipher can encrypt short and fixed length strings.

Applications require the encryption of long strings of possibly different lengths.

Applications have different goals.

- Privacy.
- Authentication.

A mode of operation is used to extend the capabilities of a block cipher to achieve a desired goal.

Length Preserving Encryption

The set of messages $\mathcal{M}$ consists of strings of different lengths.

Length Preserving Encryption

The set of messages $\mathcal{M}$ consists of strings of different lengths.

The length of the ciphertext should be equal to the length of the message.

Length Preserving Encryption

The set of messages $\mathcal{M}$ consists of strings of different lengths.

The length of the ciphertext should be equal to the length of the message.

Security: a natural extension of the notion of (S)PRP defined for a block cipher.

- Real oracle: is the actual encryption.
- Random oracle: is a uniform random length preserving permutation of $\mathcal{M}$.

Wide Block Encryption

- A length preserving mode of operation.
- Supports a tweak.
- Other names.
  - Tweakable enciphering scheme.
  - Tweakable strong pseudo-random permutation.

Wide Block Encryption

- A length preserving mode of operation.
- Supports a tweak.
- Other names.
  - Tweakable enciphering scheme.
  - Tweakable strong pseudo-random permutation.

Disk encryption.

- Encryption is done sector-wise.
- Sector address is the tweak.

Construction Ideas

Types of Constructions

Hash-Encrypt-Hash.

- Introduced by Naor-Reingold (1999);
- later work done by several other authors.

Encrypt-Mix-Encrypt

- Introduced by Halevi-Rogaway (2003);
- followed up by Halevi-Rogaway (2004);
- and Halevi (2004).

Hash-Encrypt-Hash

Encryption layer is ECB.

- NRmode (Naor-Reingold 1999);
- PEP (Chakraborty-Sarkar 2005);
- TET (Halevi 2007);
- HEH (Sarkar 2007).

Hash-Encrypt-Hash

Encryption layer is Ctr.

- XCB (McGrew-Fluhrer 2004);
- HCTR (Wang-Feng-Wu 2005);
- HCH (Chakraborty-Sarkar 2006);
- iHCH (Sarkar 2008).

Hash-Encrypt-Hash

Encryption layer is Ctr.

- XCB (McGrew-Fluhrer 2004);
- HCTR (Wang-Feng-Wu 2005);
- HCH (Chakraborty-Sarkar 2006);
- iHCH (Sarkar 2008).

Encryption layer is OFB.

- HOH (Sarkar 2008).

Encrypt-Mix-Encrypt

Encryption layers are CBC.

- CMC (Halevi-Rogaway 2003)

Encrypt-Mix-Encrypt

Encryption layers are CBC.

- CMC (Halevi-Rogaway 2003)

Encryption layers are ECB.

- EME/EME$^+$ (Halevi-Rogaway 2004);
- EME* (Halevi 2004)
- EMME (Sarkar 2008): generalises the masking operations of EME and its variants; no change in structure of the constructions.

Overview of Constructions

Assume that the block cipher is a uniform random permutation π;

Overview of Constructions

Assume that the block cipher is a uniform random permutation π;

From π construct the mode of operation Π;

Overview of Constructions

Assume that the block cipher is a uniform random permutation π;

From π construct the mode of operation Π;

Prove that Π, Π^{-1} are indistinguishable from oracles which return uniform random strings.

Overview of Constructions

Assume that the block cipher is a uniform random permutation π;

From π construct the mode of operation Π;

Prove that Π, Π^{-1} are indistinguishable from oracles which return uniform random strings.

Prove an upper bound on the advantage of the adversary.

Overview of Constructions

Assume that the block cipher is a uniform random permutation π;

From π construct the mode of operation Π;

Prove that Π, Π^{-1} are indistinguishable from oracles which return uniform random strings.

Prove an upper bound on the advantage of the adversary.

Analysis is information theoretic. Consequently, adversary can be assumed to be deterministic.

Overview of Constructions

Assume that the block cipher is a uniform random permutation π;

From π construct the mode of operation Π;

Prove that Π, Π^{-1} are indistinguishable from oracles which return uniform random strings.

Prove an upper bound on the advantage of the adversary.

Analysis is information theoretic. Consequently, adversary can be assumed to be deterministic.

Time bounded probabilistic adversary and computational complexity theoretic analysis required to tackle the replacement of E_K by π.

Overview of Constructions

If the inputs to π and π^{-1} are distinct, then their outputs are almost uniformly distributed.

Overview of Constructions

If the inputs to π and π^{-1} are distinct, then their outputs are almost uniformly distributed.

Crux. Prove that with high probability the inputs to π and π^{-1} are distinct.

Overview of Constructions

If the inputs to π and π^{-1} are distinct, then their outputs are almost uniformly distributed.

Crux. Prove that with high probability the inputs to π and π^{-1} are distinct.

- In the hash-encrypt-hash approach, a universal hash function is required to ensure this.
- In the encrypt-mix-encrypt approach, an implicit universal hash is built.

Universal Hash Definition

Function family.

Fix integer $m \geq 1$ and let $\mathbb{F}$ be a finite field.

$$\mathcal{F} : \mathcal{K} \times \mathbb{F}^m \rightarrow \mathbb{F}.$$

Universal Hash Definition

Function family.

Fix integer $m \geq 1$ and let $\mathbb{F}$ be a finite field.

$$\mathcal{F} : \mathcal{K} \times \mathbb{F}^m \rightarrow \mathbb{F}.$$

ϵ-almost universal (ϵ-AU):

For $x, x' \in \mathbb{F}^m$, $x \neq x'$,

$$\Pr_K[\mathcal{F}_K(x) = \mathcal{F}_K(x')] \leq \epsilon.$$

In other words, the collision probabilities are small.

Universal Hashing

$$\mathcal{F}_K(P_1, \ldots, P_m) = P_m + K P_{m-1} + \cdots + K^{m-1} P_1.$$

Universal Hashing

$$\mathcal{F}_K(P_1, \ldots, P_m) = P_m + KP_{m-1} + \cdots + K^{m-1}P_1.$$

Requires $(m-1)$ multplications; $m/|\mathbb{F}|$-AU; $K\mathcal{F}_K$ satisfies XOR universality.

Universal Hashing

$$\mathcal{F}_K(P_1, \ldots, P_m) = P_m + KP_{m-1} + \cdots + K^{m-1}P_1.$$

Requires $(m-1)$ multplications; $m/|\mathbb{F}|$-AU; $K\mathcal{F}_K$ satisfies XOR universality.

Bernstein (2007): introduces a class of polynomials which builds upon earlier work by Rabin-Winograd;

Universal Hashing

$$\mathcal{F}_K(P_1, \ldots, P_m) = P_m + KP_{m-1} + \cdots + K^{m-1}P_1.$$

Requires $(m-1)$ multplications; $m/|\mathbb{F}|$-AU; $K\mathcal{F}_K$ satisfies XOR universality.

Bernstein (2007): introduces a class of polynomials which builds upon earlier work by Rabin-Winograd; BRW polynomials can be computed using $m/2$ multiplications; $2m/|\mathbb{F}|$-AU;

Universal Hashing

$$\mathcal{F}_K(P_1, \ldots, P_m) = P_m + KP_{m-1} + \cdots + K^{m-1}P_1.$$

Requires $(m-1)$ multplications; $m/|\mathbb{F}|$-AU; $K\mathcal{F}_K$ satisfies XOR universality.

Bernstein (2007): introduces a class of polynomials which builds upon earlier work by Rabin-Winograd; BRW polynomials can be computed using $m/2$ multiplications; $2m/|\mathbb{F}|$-AU;

Sarkar (2008): universal hashing from word oriented LFSRs;

Universal Hashing

$$\mathcal{F}_K(P_1, \ldots, P_m) = P_m + KP_{m-1} + \cdots + K^{m-1}P_1.$$

Requires $(m-1)$ multplications; $m/|\mathbb{F}|$-AU; $K\mathcal{F}_K$ satisfies XOR universality.

Bernstein (2007): introduces a class of polynomials which builds upon earlier work by Rabin-Winograd; BRW polynomials can be computed using $m/2$ multiplications; $2m/|\mathbb{F}|$-AU;

Sarkar (2008): universal hashing from word oriented LFSRs;
no multiplications; bitwise processing – slow in software; good for resource constrained devices.

Invertible Blockwise Universal

Fix integer $m \geq 1$ and let $\mathbb{F}$ be a finite field.

$$\mathcal{F} : \mathcal{K} \times \mathbb{F}^m \rightarrow \mathbb{F}^m$$

- For each $K \in \mathcal{K}$, $\mathcal{F}_K$ is invertible.
- $(\mathbf{x}, i) \neq (\mathbf{x}', i')$, $1 \leq i, i' \leq m$.

Invertible Blockwise Universal

Fix integer $m \geq 1$ and let $\mathbb{F}$ be a finite field.

$$\mathcal{F} : \mathcal{K} \times \mathbb{F}^m \rightarrow \mathbb{F}^m$$

- For each $K \in \mathcal{K}$, $\mathcal{F}_K$ is invertible.
- $(\mathbf{x}, i) \neq (\mathbf{x}', i')$, $1 \leq i, i' \leq m$.

ϵ-BAU.

$$\begin{aligned} \mathcal{F}_K : \ & (X_1, \ldots, X_m) \ \mapsto \ (Y_1, \ldots, Y_m) \\ \mathcal{F}_K : \ & (X'_1, \ldots, X'_m) \ \mapsto \ (Y'_1, \ldots, Y'_m) \end{aligned}$$

$$\Pr_K[Y_i = Y'_{i'}] \leq \epsilon.$$

Constructions

$(X_1, \ldots, X_m) \mapsto$

- $(X_1 \oplus u(Y), \ldots, X_{m-1} \oplus u(Y), Y) \oplus$
  $(\phi_\beta(0), \ldots, \phi_\beta(m-1))$ (NR 99)
- $(X_1 \oplus Y, \ldots, X_{m-1} \oplus Y, Y) \oplus$
  $(\phi_\beta(0), \ldots, \phi_\beta(m-1))$ (Sarkar 08)

$Y = X_m \oplus \psi_\tau(X_1, \ldots, X_{m-1})$; ϕ, ψ and u are AXU;

Constructions

$(X_1, \ldots, X_m) \mapsto$

- $(X_1 \oplus u(Y), \ldots, X_{m-1} \oplus u(Y), Y) \oplus$
  $(\phi_\beta(0), \ldots, \phi_\beta(m-1))$ (NR 99)
- $(X_1 \oplus Y, \ldots, X_{m-1} \oplus Y, Y) \oplus$
  $(\phi_\beta(0), \ldots, \phi_\beta(m-1))$ (Sarkar 08)

$Y = X_m \oplus \psi_\tau(X_1, \ldots, X_{m-1})$; ϕ, ψ and u are AXU;

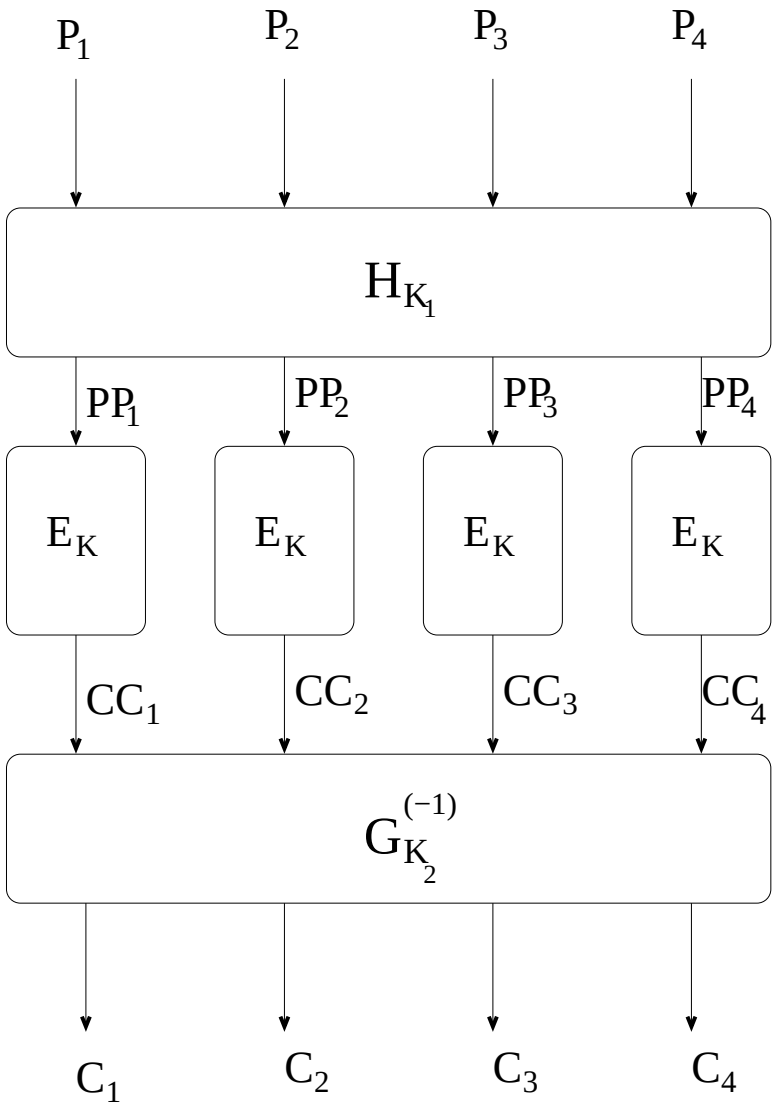
$(X_1, \ldots, X_m) \mapsto$

- $(X_1 \oplus Z, \ldots, X_m \oplus Z) \oplus (\beta, \alpha\beta, \ldots, \alpha^{m-1}\beta)$
  (Halevi 07)

$Z = \sigma^{-1}(X_1\tau^m \oplus \cdots \oplus X_{m-1}\tau^2 \oplus X_m\tau)$;
$\sigma = 1 \oplus \tau \oplus \cdots \oplus \tau^m$.

Hash-ECB-Hash Constructions

Hash-ECB-Hash



H and G are invertible blockwise universal hash functions.

Hash-ECB-Hash Constructions

NRmode. Predates the notion of tweaks; did not specify handling of partial blocks.

Hash-ECB-Hash Constructions

NRmode. Predates the notion of tweaks; did not specify handling of partial blocks.

TET. Incorporates tweaks and handles partial blocks. But, more complicated and hence less efficient.

Hash-ECB-Hash Constructions

NRmode. Predates the notion of tweaks; did not specify handling of partial blocks.

TET. Incorporates tweaks and handles partial blocks. But, more complicated and hence less efficient.

HEH. Incorporates tweaks and handles partial blocks. Simplifies NRmode construction; more efficient than TET.

Hash-ECB-Hash Constructions

NRmode. Predates the notion of tweaks; did not specify handling of partial blocks.

TET. Incorporates tweaks and handles partial blocks. But, more complicated and hence less efficient.

HEH. Incorporates tweaks and handles partial blocks. Simplifies NRmode construction; more efficient than TET.

Other less efficient constructions known.

HEH (Full Blocks)

$$H_{K_1} = \Psi_{\tau,\beta_1}, \quad G_{K_2} = \Psi_{\tau,\beta_2}.$$

HEH (Full Blocks)

$$H_{K_1} = \Psi_{\tau,\beta_1}, \quad G_{K_2} = \Psi_{\tau,\beta_2}.$$

$$\begin{aligned}
&\Psi_{\tau,\beta}(X_1, \ldots, X_m) \\
&\quad = (X_1 \oplus Y, \ldots, X_{m-1} \oplus Y, Y) \oplus \\
&\qquad (\phi_\beta(1), \ldots, \phi_\beta(m-1), \phi_\beta(0)) \\
&Y = X_m \oplus \psi_\tau(X_1, \ldots, X_{m-1}).
\end{aligned}$$

HEH (Full Blocks)

$$H_{K_1} = \Psi_{\tau,\beta_1}, \quad G_{K_2} = \Psi_{\tau,\beta_2}.$$

$$\begin{aligned}
&\Psi_{\tau,\beta}(X_1, \ldots, X_m) \\
&\quad = (X_1 \oplus Y, \ldots, X_{m-1} \oplus Y, Y) \oplus \\
&\qquad (\phi_\beta(1), \ldots, \phi_\beta(m-1), \phi_\beta(0)) \\
&Y = X_m \oplus \psi_\tau(X_1, \ldots, X_{m-1}).
\end{aligned}$$

$\phi_\beta : i \mapsto \alpha^i \beta$; α is a primitive element of $GF(2^n)$.

HEH (Full Blocks)

$$H_{K_1} = \Psi_{\tau,\beta_1}, \quad G_{K_2} = \Psi_{\tau,\beta_2}.$$

$$\begin{aligned}\Psi_{\tau,\beta}(X_1,\ldots,X_m) &= (X_1 \oplus Y,\ldots,X_{m-1}\oplus Y, Y)\oplus \\ &\quad (\phi_\beta(1),\ldots,\phi_\beta(m-1),\phi_\beta(0))\end{aligned}$$

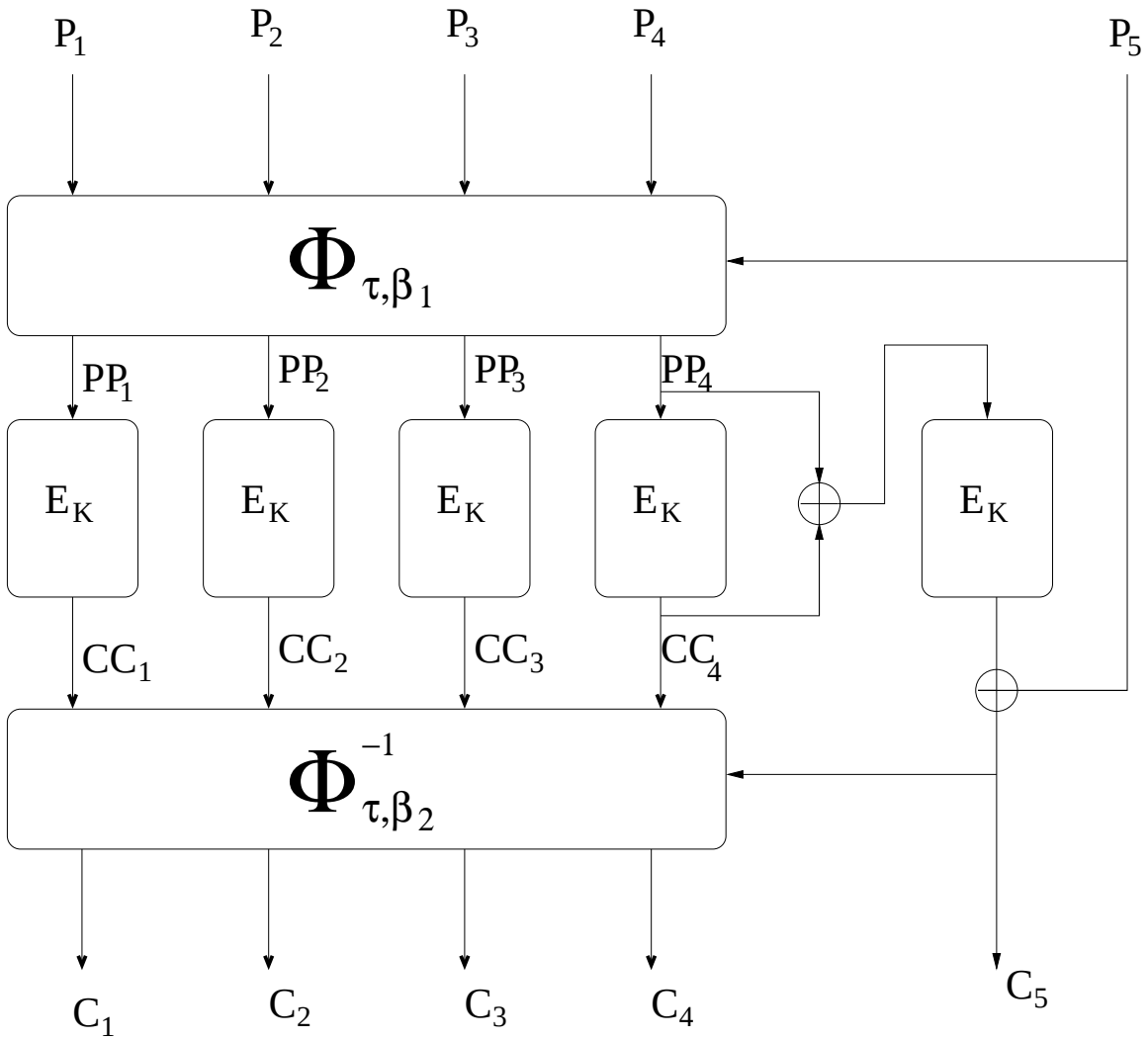
$$Y = X_m \oplus \psi_\tau(X_1,\ldots,X_{m-1}).$$

$\phi_\beta : i \mapsto \alpha^i\beta$; α is a primitive element of $GF(2^n)$.

ψ_τ can be instantiated using

- usual polynomial hashing;
- hashing using BRW polynomials;
- hashing using word oriented LFSRs.

HEH (Partial Block)



Handling partial blocks.

HEH (Partial Block)

$$\begin{aligned}\Phi_{\tau,\beta}(X_1,\ldots,X_{m-1},X_m)\\ =&(X_1\oplus Y,\ldots,X_{m-2}\oplus Y,Y,X_m)\\ &\oplus(\phi_\beta(1),\ldots,\phi_\beta(m-2),\phi_\beta(0),0^r)\end{aligned}$$

$$Y = X_{m-1}\oplus\psi_\tau(X_1,\ldots,X_{m-2},X_m||0^{n-r}).$$

Note.

- X_{m-1} is the last full block;
- ϕ_β and ψ_τ defined as for full blocks.

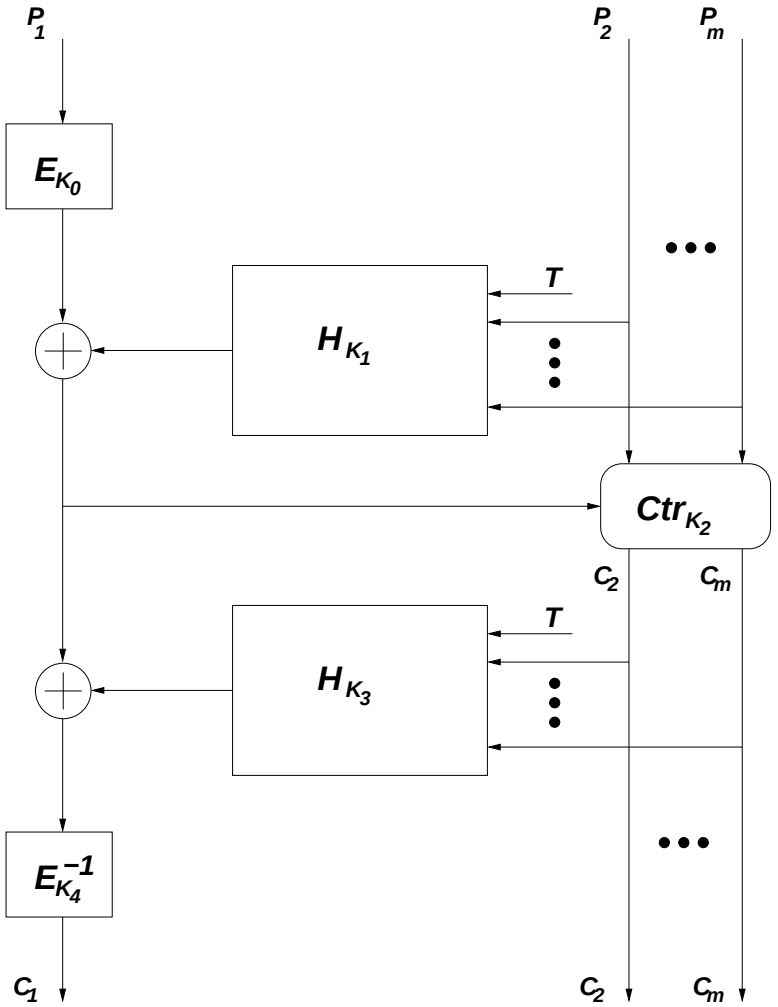
Definition of τ, β_1, β_2

KeyDef1	KeyDef2
$\gamma = E_K(T)$; $\beta_1 = E_K(\gamma \oplus \mathsf{bin}_n(\ell))$; $\beta_2 = \alpha\beta_1$; $\tau = \gamma$.	$\gamma = E_K(T)$; $\beta_1 = E_K(\gamma \oplus \mathsf{bin}_n(\ell))$; $\beta_2 = \alpha\beta_1$; choose τ randomly from $GF(2^n)$.
KeyDef3	
$\beta_1 = E_K(T)$; $\beta_2 = \alpha\beta_1$; choose τ randomly from $GF(2^n)$.	

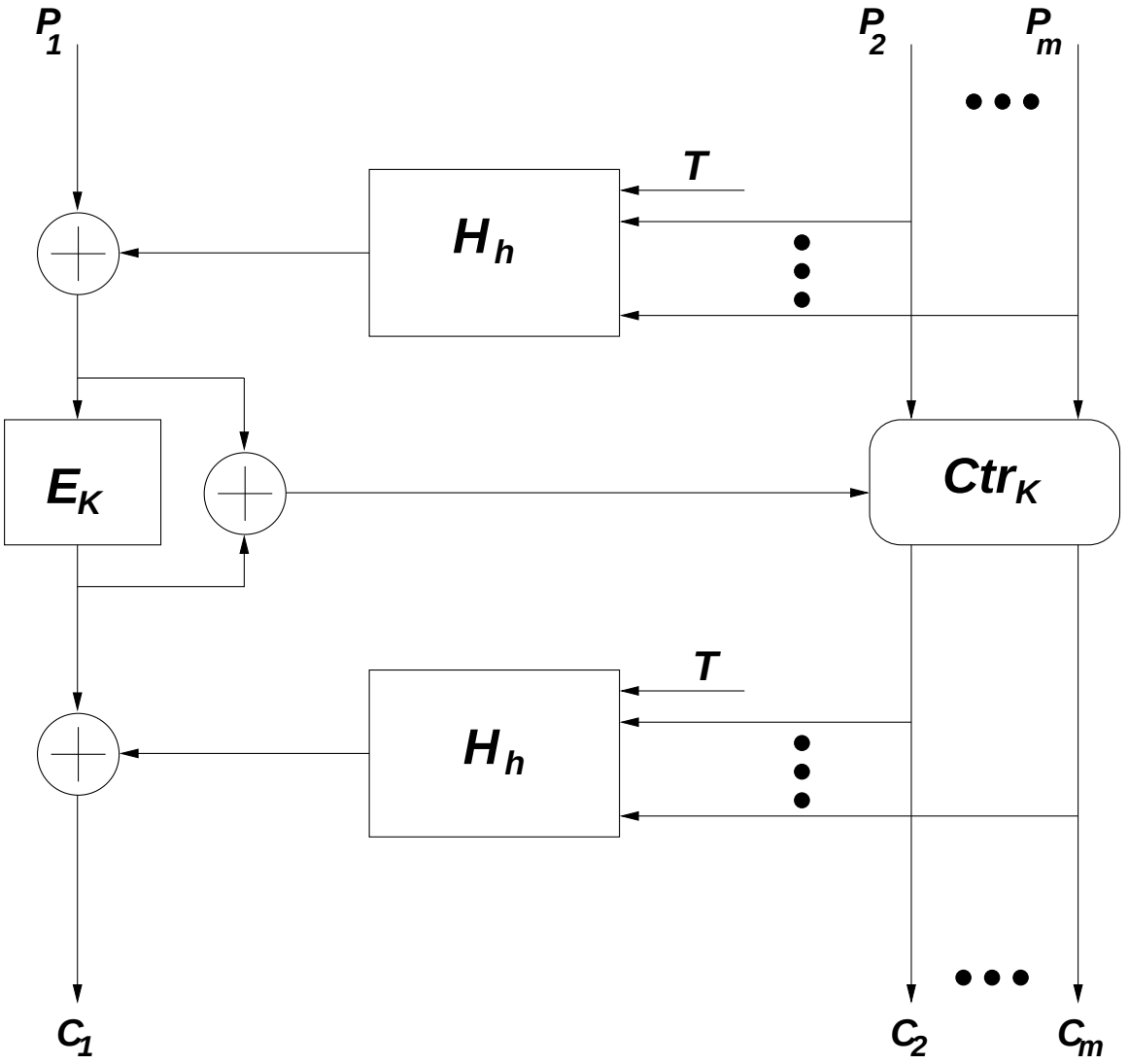
KeyDef1: single key; no pre-comp;
KeyDef2: separate hash key; allows pre-comp;
KeyDef3: fixed length; saves one call.

Counter/OFB Based Constructions

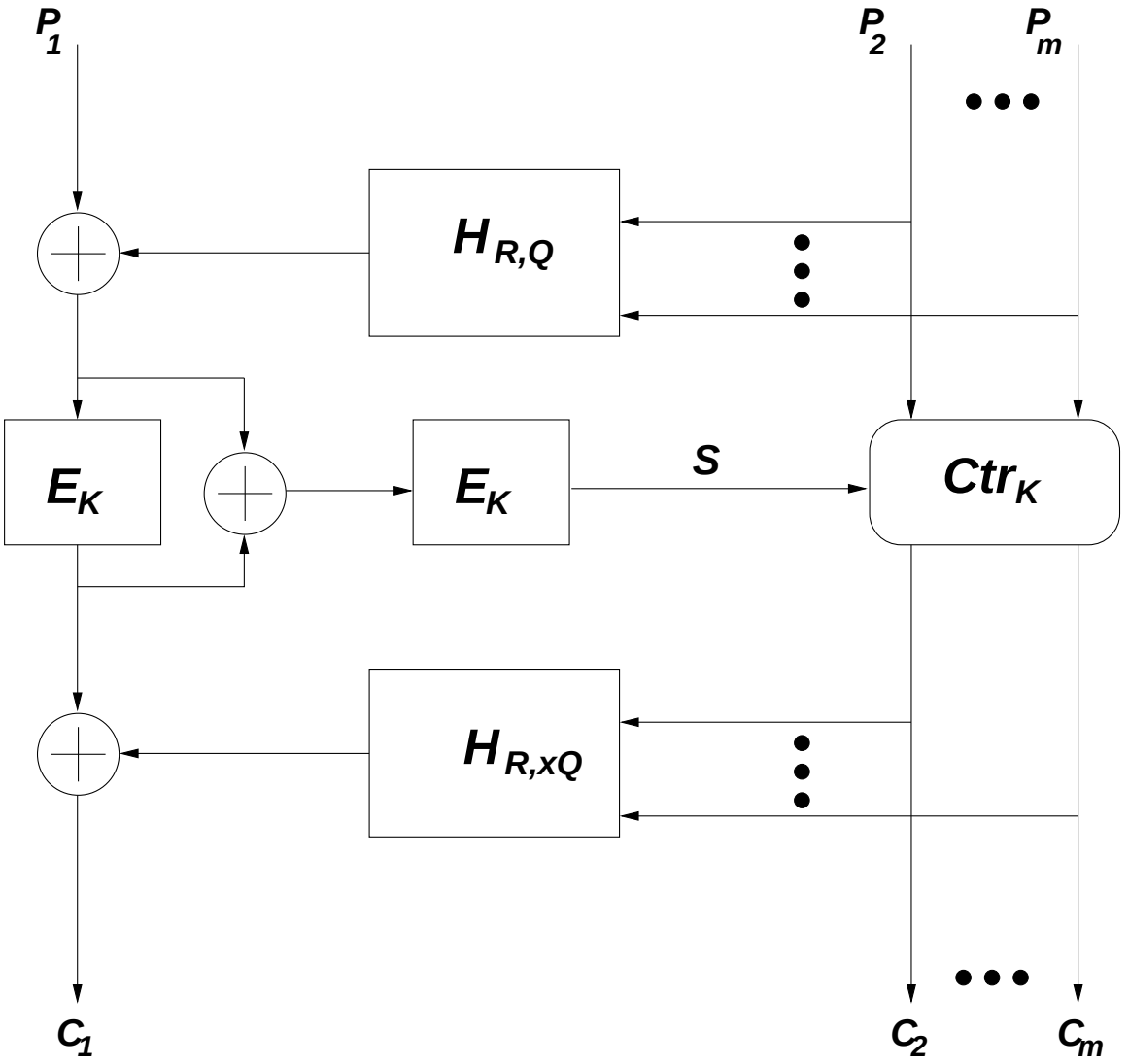
XCB Mode of Operation



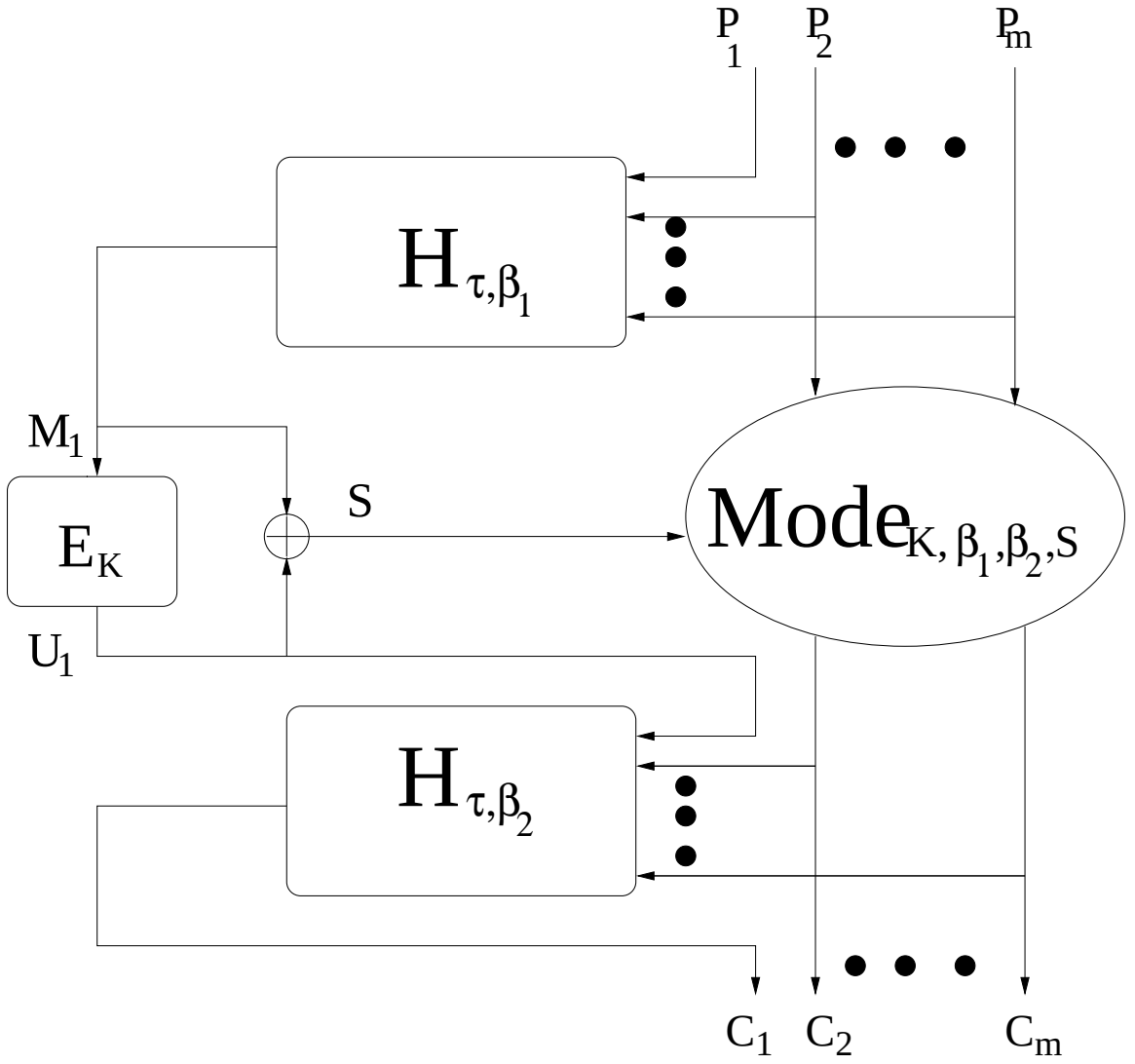
HCtr Mode of Operation



HCH Mode of Operation



iHCH/HOH Modes



iHCH/HOH Modes

$$H_{\tau,\beta}(X_1,\ldots,X_m) = \beta \oplus X_1 \oplus \psi_\tau(X_2,\ldots,X_m).$$

Mode$_{K,\beta_1,\beta_2,S}$ as Counter:

$$\mathsf{Ctr}_{K,\beta_1,R}(X_1,\ldots,X_m) = \mathrm{ECB}_K(R_1,\ldots,R_m)$$

where $R = S \oplus \beta_1 \oplus \beta_2$, $R_i = \phi_\beta(i-1) \oplus R$.

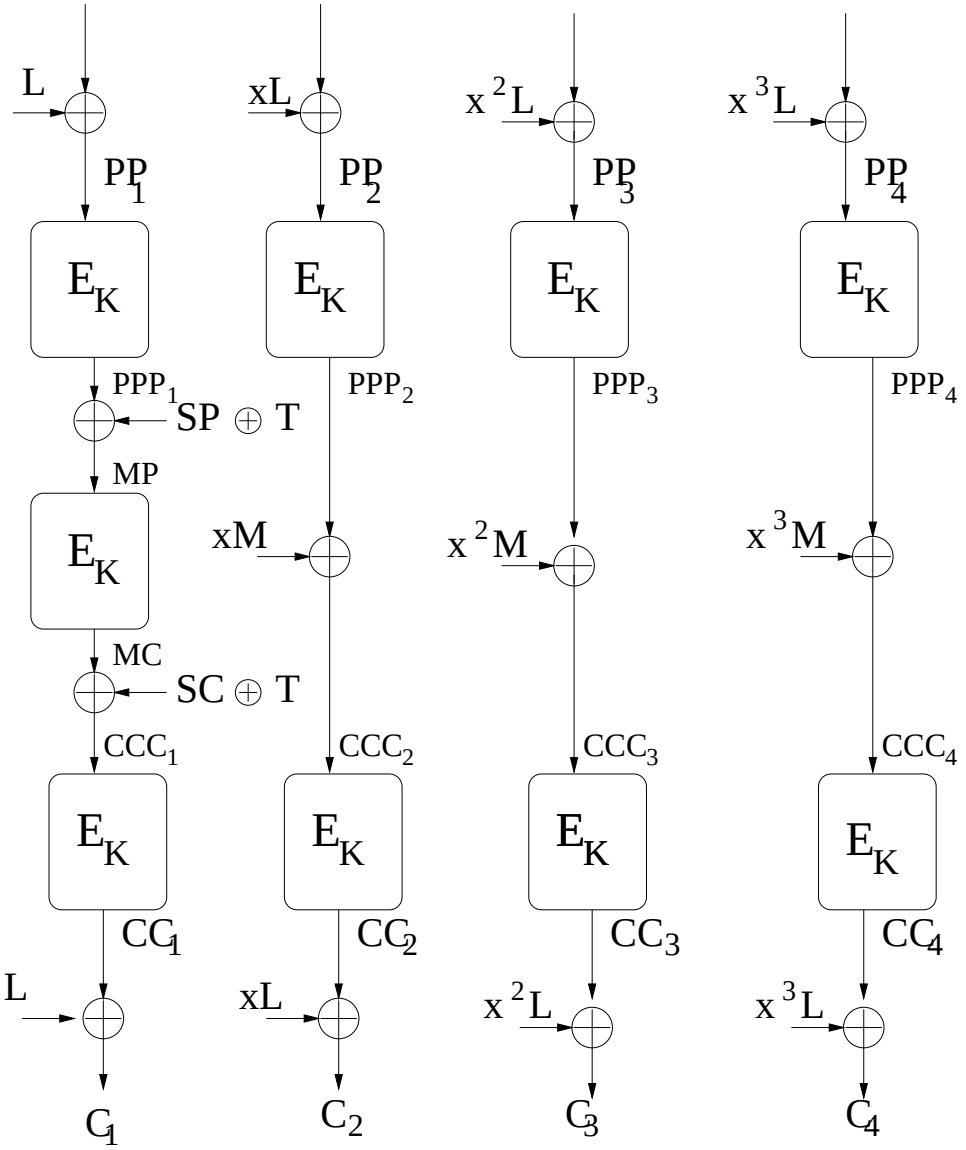
Mode$_{K,\beta_1,\beta_2,S}$ as OFB:

$$\mathsf{OFB}_{K,S}(X_1,\ldots,X_m) = (X_1,\ldots,X_m) \oplus (S_1,\ldots,S_m)$$

where $S_i = E_K^i(S)$; β_1 and β_2 are not used.

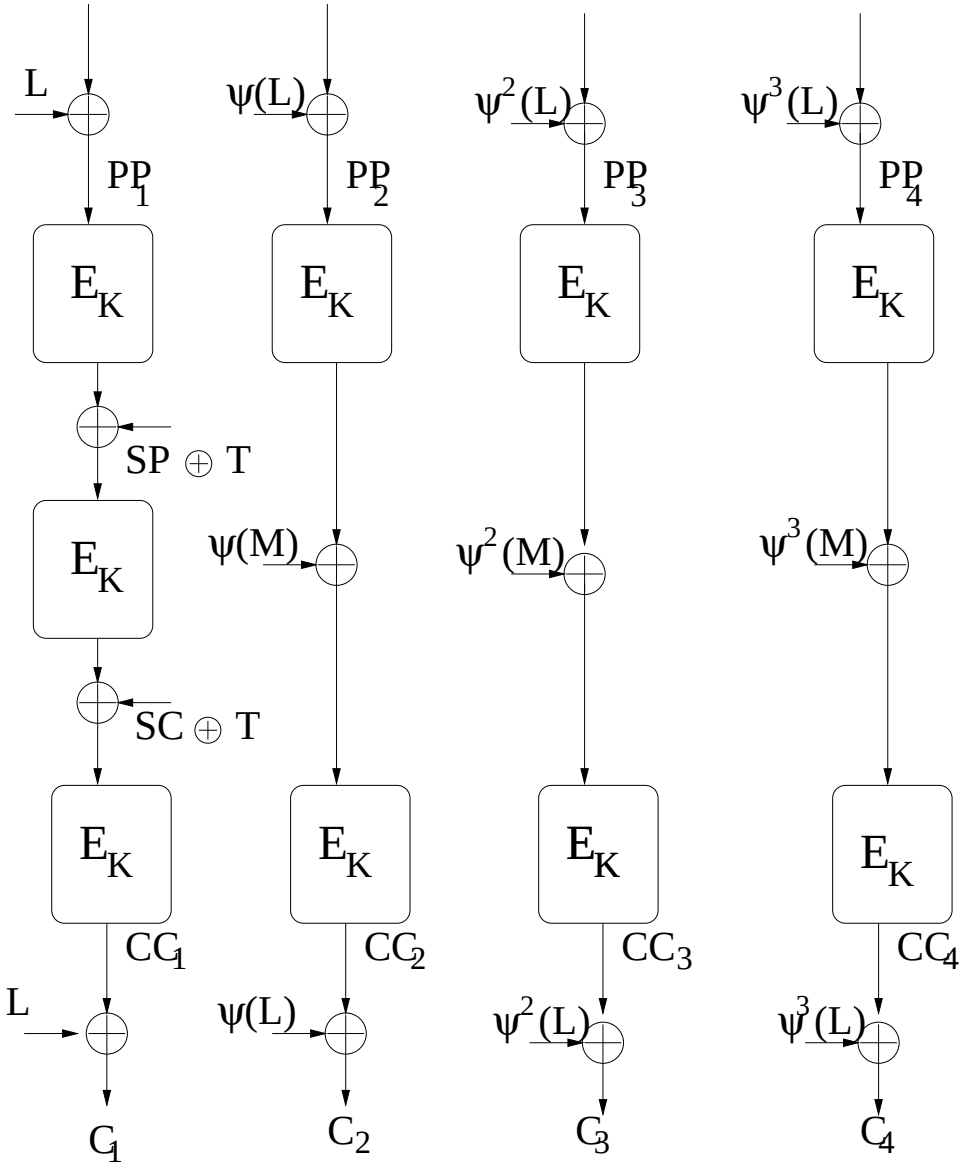
Encrypt-Mix-Encrypt Constructions

EME Mode of Operation



$$SP = PPP_2 \oplus PPP_3 \oplus PPP_4; SC = CCC_2 \oplus CCC_3 \oplus CCC_4; M = MP \oplus MC.$$

EMME Mode of Operation



$$SP = PPP_2 \oplus PPP_3 \oplus PPP_4; SC = CCC_2 \oplus CCC_3 \oplus CCC_4; M = MP \oplus MC.$$

EMME Mode of Operation

EME: "Multiplication by x": $S \mapsto xS \bmod \tau(x)$
$\tau(x)$: primitive, degree n polynomial over $GF(2)$.

EMME Mode of Operation

EME: "Multiplication by x": $S \mapsto xS \bmod \tau(x)$
$\tau(x)$: primitive, degree n polynomial over $GF(2)$.

EMME: Generalization: $\psi : GF(2^n) \to GF(2^n)$; linear map whose minimal polynomial over $GF(2)$ is primitive and of degree n.

EMME Mode of Operation

EME: "Multiplication by x": $S \mapsto xS \bmod \tau(x)$
$\tau(x)$: primitive, degree n polynomial over $GF(2)$.

EMME: Generalization: $\psi : GF(2^n) \to GF(2^n)$; linear map whose minimal polynomial over $GF(2)$ is primitive and of degree n.

- ψ can be instantiated using a tower field representation of $GF(2^n)$;
- "word oriented LFSR";
- software implementation is faster than "multiplication by x".

Comparison Issues

Security: all modes provide the same security level.
Efficiency:

- Number of block cipher calls.

Comparison Issues

Security: all modes provide the same security level.

Efficiency:

- Number of block cipher calls.
- Number of multiplications (if any).

Comparison Issues

Security: all modes provide the same security level.
Efficiency:

- Number of block cipher calls.
- Number of multiplications (if any).
- Number of block cipher keys.

Comparison Issues

Security: all modes provide the same security level.
Efficiency:

- Number of block cipher calls.
- Number of multiplications (if any).
- Number of block cipher keys.
- Number of other key material (if any).

Comparison Issues

Security: all modes provide the same security level.
Efficiency:

- Number of block cipher calls.
- Number of multiplications (if any).
- Number of block cipher keys.
- Number of other key material (if any).
- Use of pre-computed multiplication tables (if relevant).

Comparison Issues

Security: all modes provide the same security level.
Efficiency:

- Number of block cipher calls.
- Number of multiplications (if any).
- Number of block cipher keys.
- Number of other key material (if any).
- Use of pre-computed multiplication tables (if relevant).
- Parallelism, simplicity, ...

Comparison Issues

Security: all modes provide the same security level.
Efficiency:

- Number of block cipher calls.
- Number of multiplications (if any).
- Number of block cipher keys.
- Number of other key material (if any).
- Use of pre-computed multiplication tables (if relevant).
- Parallelism, simplicity, ...
- An important issue: Variable/arbitrary versus fixed length messages.

Variable Length – Efficiency

Mode	[BC]	[M]	[I]
EME*	$2m + m/n + 1$	–	–
XCB	$(m + 6)$	$2(m + 1)$	–
HCTR	m	$2(m + 1)$	–
HCH	$(m + 3)$	$2(m - 1)$	–
TET	$2\imath + m + 2$	$\imath(m - 1) + 2m$	1
XXX	$(m + 2)$	$2(m - 1)$	–
XXX	$(m + 2)$	m (with BRW)	–

XXX: HEH, iHCH, HOH; for TET $\imath \geq 1$.

Variable Length – # Keys

Mode	# keys
EME*	1[BCK] + 2[AK]
XCB	1[BCK]
HCTR	1[BCK]+1[AK]
HCH	1[BCK]
TET	2[BCK]
XXX	1[BCK]

XXX: HEH, iHCH, HOH.

Fixed Length Messages

Mode	[BC]	[M]	[BCK]	[AK]	tab
EME*	$2m+1+m/n$	–	1	2	–
XCB	$m+1$	$2(m+3)$	3	2	yes
HCTR	m	$2(m+1)$	1	1	yes
HCH	$m+2$	$2(m-1)$	1	1	yes
TET	$m+1$	$2m$	2	3	yes
XXX	$m+1$	$2(m-1)$	1	1	yes
XXX	$m+1$	m (BRW)	1	1	no

XXX: HEH, iHCH, HOH.

Key Agility

Mode	comp. cost	key sch.	mult. tab.
EME*	–	1	–
XCB	5[BC]	3	2
HCTR	–	1	1
HCH	–	1	1
TET	$\imath((m-1)$[M] 2[BC])+ 1[BC]+1[I]	2	1
XXX	–	1	1

XXX: HEH, iHCH, HOH.

Comparison Summary

Software implementation:

- HOH will be the fastest: there are no masking operations; HEH, iHCH are also good choices;
- so are HCTR and HCH;
- XCB has good efficiency but less efficient key agilitiy;
- TET: inefficient for variable lengths; bad key agility for fixed lengths.
- EME is good; EMME is sligtly better.
- No pre-computation: use BRW polynomials.

Comparison Summary

Software implementation:

- HOH will be the fastest: there are no masking operations; HEH, iHCH are also good choices;
- so are HCTR and HCH;
- XCB has good efficiency but less efficient key agilitiy;
- TET: inefficient for variable lengths; bad key agility for fixed lengths.
- EME is good; EMME is sligtly better.
- No pre-computation: use BRW polynomials.

Hardware implementation:

- Issues: chip size, memory, ...
- HEH, iHCH, HOH offer the best features.

Patents and Standards

Patented: EME/EME*, XCB.

Unpatented:

- NRmode, TET, HEH;
- HCtr, HCH, iHCH (uses Ctr mode; as does XCB).
- HOH.

IEEE P1619.2 standard: decision (as of Nov 2008) to standardize

- EME2 (of the encrypt-mix-encrypt type)
- XCB (of the hash-encrypt-hash type)

`https://siswg.net/`

Thank you for your attention!