Disk Encryption

Cuauhtemoc Mancillas López

Department of Computer Science
Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
(CINVESTAV-IPN)
Mexico City, Mexico 07360

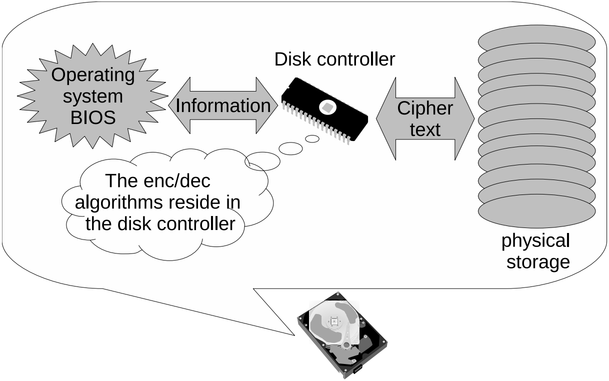
cuauhtemoc.mancillas.lopez@univ-st-etienne.fr

June 30, 2014

Disk Encryption Problem

- The problem of disk encryption is to encrypt bulk information stored in a storage media like hard disk, flash memory, CD or DVD.
- The nature of storage media dictates the type of encryption required. **We are primarily interested in hard disks**.
- A well accepted proposal for encrypting hard disks is to encrypt individual sectors.

Low Level Disk Encryption



The Solution

Encrypt all data present in the disk!

The Solution

Encrypt all data present in the disk!

Important questions

- Which scheme to use?
- How to use it?

The Solution

Encrypt all data present in the disk!

Important questions

- Which scheme to use?
- How to use it?

In this presentation we would explore answers to these questions.

Requirements

- **Length preserving:** Length of plaintext and ciphertext should be the same. (May not be an important requirement, we shall explore this later)

Requirements

- **Length preserving:** Length of plaintext and ciphertext should be the same. (May not be an important requirement, we shall explore this later)
- **Ciphertext Variability:** Same data stored in two different sectors should look different.

Requirements

- **Length preserving:** Length of plaintext and ciphertext should be the same. (May not be an important requirement, we shall explore this later)
- **Ciphertext Variability:** Same data stored in two different sectors should look different.
- **Security**
  - An adversary should not be able to infer "anything" regarding the plaintext by looking at the ciphertext.
  - An adversary should not be able to tamper the ciphertext such that it gets decrypted to something meaningful.

Requirements

- **Length preserving:** Length of plaintext and ciphertext should be the same. (May not be an important requirement, we shall explore this later)
- **Ciphertext Variability:** Same data stored in two different sectors should look different.
- **Security**
  - An adversary should not be able to infer "anything" regarding the plaintext by looking at the ciphertext.
  - An adversary should not be able to tamper the ciphertext such that it gets decrypted to something meaningful.

Tweakable enciphering schemes satisfy all these requirements

Outline

Preliminaries

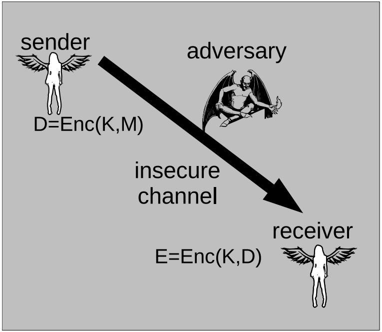
Tweakable Enciphering Schemes
- Implementations
- BRW Polynomials

Lightweight Disk Encryption
- STES
- Implementation

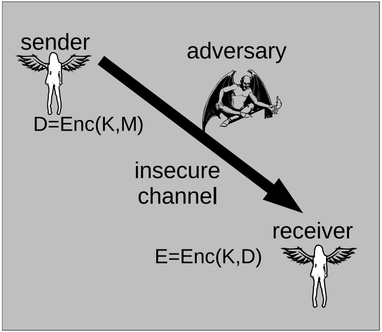
New Model for Disk Encryption
- BCTR
- Implementation

Open Problems

Adversary


sender
adversary
D=Enc(K,M)
insecure
channel
receiver
E=Enc(K,D)

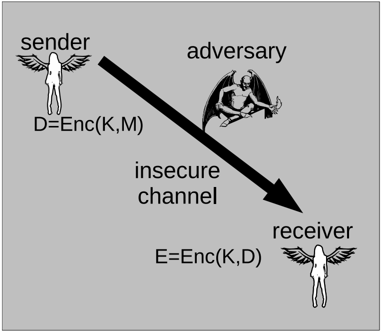
Adversary



Adversarial Goals

- Key Recovery
- Plaintext Recovery
- Create Ciphertext
- Distinguishing

Adversary



Adversarial Goals

- Key Recovery
- Plaintext Recovery
- Create Ciphertext
- Distinguishing

Adversarial Resources

- Ciphertext only
- Known Plaintext
- Chosen Plaintext
- Chosen Ciphertext
- Adaptive Chosen Plaintext
- Adaptive Chosen Ciphertext

The Adversary

- The adversary is considered to be a probabilistic algorithm.
- It has oracle access to the functions and can output either a 0 or 1.
- It can interact with the function through valid queries.
- An adversary $\mathcal{A}$ interacting with an oracle $\mathcal{O}$ outputing 1 will be denoted by

$$\mathcal{A}^{\mathcal{O}} \Rightarrow 1$$

Block Ciphers

- A function $E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$.
- Usually written as $E_K(M)$ instead of $E(K, M)$
- k the key length
- n the block length
- For every $K \in \{0,1\}^k$, $E_K()$ must be a permutation. Thus, for every $K \in \{0,1\}^k$, $E_K^{-1}()$, is defined and $E_K^{-1}(E_K(M)) = M$.

Examples: AES, DES, IDEA, SERPENT, TWOFISH, PRESENT ....

Block Ciphers

- A function $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$.
- Usually written as $E_K(M)$ instead of $E(K, M)$
- k the key length
- n the block length
- For every $K \in \{0,1\}^k$, $E_K()$ must be a permutation. Thus, for every $K \in \{0,1\}^k$, $E_K^{-1}()$, is defined and $E_K^{-1}(E_K(M)) = M$.

Examples: AES, DES, IDEA, SERPENT, TWOFISH, PRESENT ....

When is a block cipher secure?

Block Cipher Security

When is a block cipher secure?

Block Cipher Security

When is a block cipher secure?

- Difficult to recover the key

Block Cipher Security

When is a block cipher secure?

- Difficult to recover the key
- Difficult to recover part of the key

Block Cipher Security

When is a block cipher secure?

- Difficult to recover the key
- Difficult to recover part of the key
- Difficult to obtain the plaintext

Block Cipher Security

When is a block cipher secure?

- Difficult to recover the key
- Difficult to recover part of the key
- Difficult to obtain the plaintext
- Difficult to obtain a part of the plaintext

Block Cipher Security

When is a block cipher secure?

- Difficult to recover the key
- Difficult to recover part of the key
- Difficult to obtain the plaintext
- Difficult to obtain a part of the plaintext
- Difficult to say if the i-th bit of the plaintext is 0.

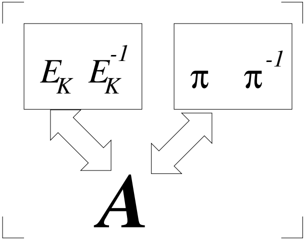
  ......

Strong Pseudorandom Permutations

It is assumed that a secure block cipher is a strong pseudorandom permutation

Strong Pseudorandom Permutations

It is assumed that a secure block cipher is a strong pseudorandom permutation



$$\mathbf{Adv}_E^{\mathrm{prp}}(\mathcal{A}) = \Pr\left[K \xleftarrow{\$} \mathcal{K} : A^{E_K()E_K^{-1}()} \Rightarrow 1\right] - \Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}(n) : \mathcal{A}^{\pi()\pi^{-1}()} \Rightarrow 1\right].$$

Perm(n) is the set of all permutations from n bits to n bits.

Pseudorandom Functions

Given a function family $F : \{0,1\}^m \rightarrow \{0,1\}^n$ and an adversary $\mathcal{A}$, define the PRF advantage of $\mathcal{A}$ in breaking F as

$$\begin{aligned} \mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) &= \Pr\left[K \stackrel{\$}{\leftarrow} \mathcal{K} : \mathcal{A}^{F_K(.)} \Rightarrow 1\right] \\ &\quad - \Pr\left[\rho \stackrel{\$}{\leftarrow} \mathsf{Func}(\mathsf{m},\mathsf{n}) : \mathcal{A}^{\rho(.)} \Rightarrow 1\right]. \end{aligned}$$

$\mathsf{Func}(\mathsf{m},\mathsf{n})$ is the set of all functions that maps from m bits to n bits.

Pseudorandom Functions

Given a function family $F : \{0,1\}^m \to \{0,1\}^n$ and an adversary $\mathcal{A}$, define the PRF advantage of $\mathcal{A}$ in breaking F as

$$\begin{aligned}\textbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) &= \Pr\left[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{F_K(\cdot)} \Rightarrow 1\right] \\ &\quad - \Pr\left[\rho \xleftarrow{\$} \mathsf{Func(m,n)} : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1\right].\end{aligned}$$

$\mathsf{Func(m,n)}$ is the set of all functions that maps from m bits to n bits.

***F* is called a pseudorandom function family if for all adversaries $\mathcal{A}$ using reasonable resources, $\textbf{Adv}_F^{\mathrm{prf}}(\mathcal{A})$ is small.**

Finite Fields

We shall often treat n bit binary strings as elements of $GF(2^n)$.

Elements in $\{0,1\}^n$ can be seen as polynomials of the form

$$a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}.$$

Finite Fields

We shall often treat n bit binary strings as elements of $GF(2^n)$.

Elements in $\{0,1\}^n$ can be seen as polynomials of the form

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}.$$

For $X, Y \in \{0,1\}^n$,

- Addition in the field: $X \oplus Y$, realized by bitwise xor.
- Multiplication: XY, realized by ordinary polynomial multiplication followed by reduction using a fixed n degree irreducible polynomial.

Finite Fields

An important operation on finite fields is *xtimes*.

For $A \in GF(2^n)$, by xA, we mean the multiplication of the monomial x with the polynomial A followed by a reduction using the irreducible polynomial.

This does not amount to a multiplication, can be easily done using a shift and a conditional xor.

Polynomial Hash

Informally a hash function maps a big string into a small one. We shall use a specific type of hash called the polynomial hash

$$H : \{0,1\}^n \times \{0,1\}^{nm} \rightarrow \{0,1\}^n$$

defined as

$$H_h(P_1||...||P_m) = P_1 h^m \oplus P_2 h^{m-1} \oplus ... \oplus P_m h$$

All operations are in $GF(2^n)$,$h, P_i \in \{0,1\}^n$
This type of functions are AXU (almost xor universal hash), because for any $G \in \{0,1\}^n$, and $P \neq P'$.

$$\Pr[h \xleftarrow{\$} \{0,1\}^n : H_h(P) \oplus H_h(P') = G] \leq \frac{maxdegree(P, P')}{2^n}$$

Block Ciphers Modes of Operation

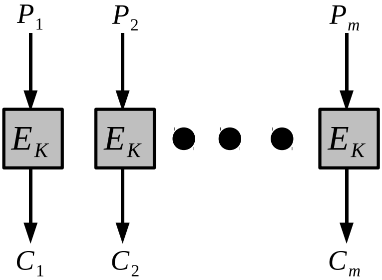
We have a secure $n - bit$ to $n - bit$ block-cipher. How can we encrypt a M ($M > n$) bit message?

Block Ciphers Modes of Operation

We have a secure $n - bit$ to $n - bit$ block-cipher. How can we encrypt a M $(M > n)$ bit message?

Using a block cipher mode of operation like:

- Electronic Code Book (ECB).

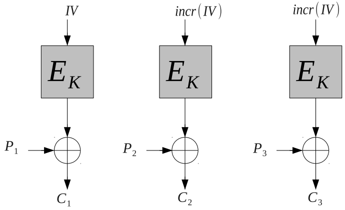


Block Ciphers Modes of Operation

We have a secure $n - bit$ to $n - bit$ block-cipher. How can we encrypt a M ($M > n$) bit message?

Using a block cipher mode of operation like:

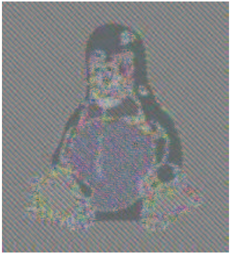
- Counter Mode (CTR).



Security of Modes of Operation

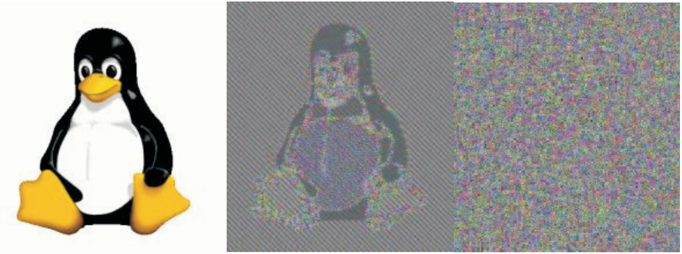
Security of Modes of Operation

Insecure: The encryption algorithm gives information about plaintext.



Security of Modes of Operation

Secure: The ciphertext looks like a random output.



*Images taken from wikipedia.

Types of Modes

Modes can be classified according to the type of security service they provide

- Privacy only: Ctr, CBC, OFB.
- Authenticated encryption: GCM, CCM, OCB.
- Authenticated encryption with associated data.
- Message Authentication Codes: PMAC, OMAC, CMAC.
- Tweakable enciphering schemes (Modes for Disk Encryption).
- Deterministic authenticated encryption.

Outline

Preliminaries

Tweakable Enciphering Schemes
- Implementations
- BRW Polynomials

Lightweight Disk Encryption
- STES
- Implementation

New Model for Disk Encryption
- BCTR
- Implementation

Open Problems

Tweakable Enciphering Schemes

A length preserving encryption scheme which provides security as of a SPRP.

Tweakable Enciphering Schemes

A length preserving encryption scheme which provides security as of a SPRP.

$$\mathbf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$$

- $\mathcal{K} \neq \emptyset$ is the key space.
- $\mathcal{T} \neq \emptyset$ is the tweak space.
- The message and the cipher spaces are $\mathcal{M}$.
  Ideally $\mathcal{M} = \cup_{i>1}\{0,1\}^i$
  For most practical purposes $\mathcal{M} = \{0,1\}^{mn}$.

Generally written as $\mathbf{E}_K^T(.)$.
A TES is supposed to behave like a block-cipher on a big block.

Security of TES

- Let $\mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ denote the set of all functions $\boldsymbol{\pi} : \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ where $\boldsymbol{\pi}(\mathcal{T}, .)$ is a length preserving permutation.
- Such a $\boldsymbol{\pi} \in \mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ is called a tweak indexed permutation.
- Let $\mathbf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ be a TES.
- We define the advantage an adversary $\mathcal{A}$ has in distinguishing $\mathbf{E}$ and its inverse from a random tweak indexed permutation and its inverse in the following manner.

$$\mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(A) = \Pr\left[K \stackrel{\$}{\leftarrow} \mathcal{K} : A^{\mathbf{E}_K(.,.),\mathbf{E}_K^{-1}(.,.)} \Rightarrow 1\right] - \Pr\left[\boldsymbol{\pi} \stackrel{\$}{\leftarrow} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M}) : A^{\boldsymbol{\pi}(.,.),\boldsymbol{\pi}^{-1}(.,.)} \Rightarrow 1\right].$$

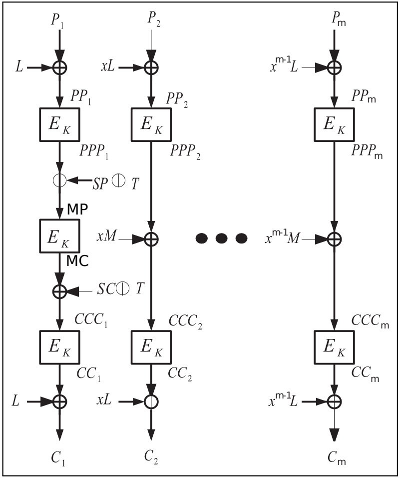
Existing TES

Depending of their structure TES are classified as follows

- ECB-Mask-ECB.
- Hash-Counter-Hash.
- Hash-ECB-Hash.

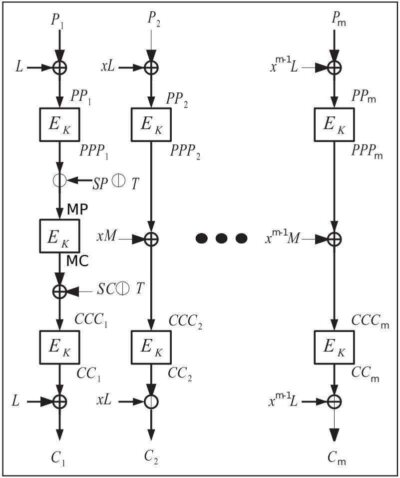
ECB-mask-ECB

- EME (Halevi and Rogaway,2003).



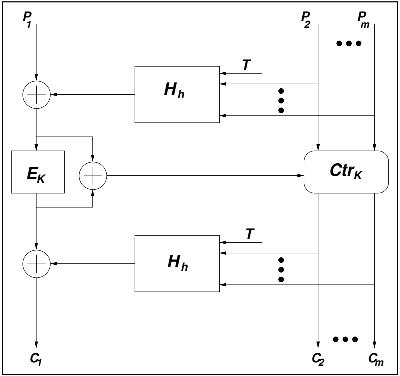
ECB-mask-ECB

- EME (Halevi and Rogaway,2003).
- CMC (Halevi and Rogaway,2003).
- EME* (Halevi,2004).
- EME2 (Halevi,2007).



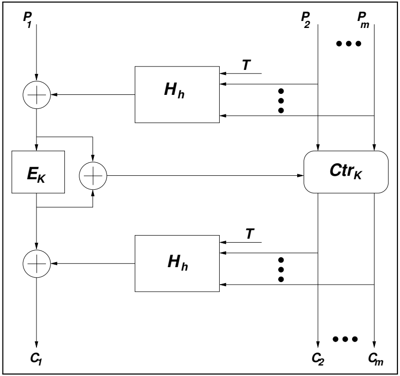
Hash-Counter-Hash

- HCTR (Wang,et. al,2005).



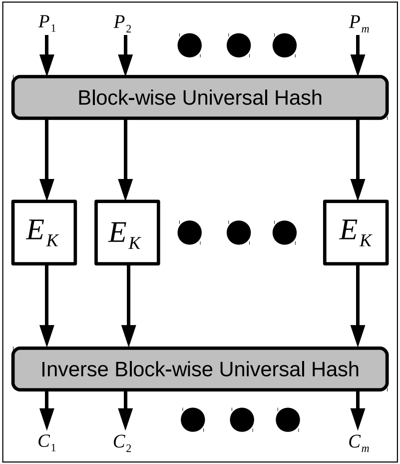
Hash-Counter-Hash

- HCTR (Wang,et. al,2005).
- ABL (McGrew and Viega,2004).
- XCB (McGrew and Flurer,2004).
- HCH (Chakraborty and Sarkar,2006).



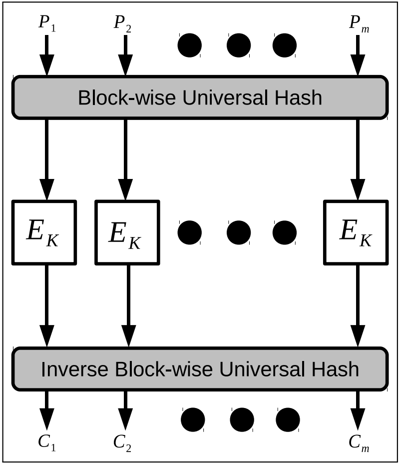
Hash-ECB-Hash

- TET (Halevi,2007).



Hash-ECB-Hash

- TET (Halevi,2007).
- HEH (Sarkar,2007).
- PEP (Chakraborty and Sarkar,2006).



IEEE SISW and P1619

- IEEE security in storage working group has been working towards standardization of cryptographic algorithms for various storage media.
- For sector wise storage media they have divided the task into two categories:
  - **Wide block modes:** A tweakable block cipher on the whole disk sector
  - **Narrow block modes**: An ECB mode of tweakable block ciphers.

Wide Block Modes

Technically same as a TES.

- Length Preserving: **Yes**
- Ciphertext Variability: **Yes**
- Security : **Satisfactory**

Wide Block Modes

Technically same as a TES.

- Length Preserving: **Yes**
- Ciphertext Variability: **Yes**
- Security : **Satisfactory**

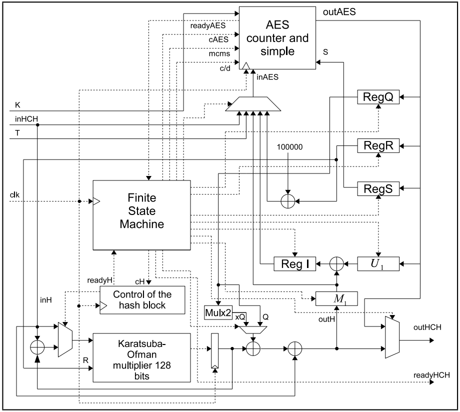
Current Status

- EME2 and XCB are described in the standard IEEE 1619.2-2010, which recommends use of these algorithms for encrypting random access block oriented storage devices
- The reason for the choice is not very clear.
  - Both XCB and EME2 are un-ambiguously covered under some existing patent claims.
  - Performance of XCB in hardware is poor compared to (many) other modes.

Efficient Implementations

Some TES were implemented using a 128 bits pipelined AES and fully parallel Karatsuba Ofman multiplier, on Virtex 2 pro, Virtex 4 and Virtex 5 FPGAs.

- EME.
- HCTR,HCH,XCB.
- TET,HEH.



Efficient Implementations

Mode	Slices	B-RAM	Frequency (MHz)	Clock Cycles	Time (μS)	Latency (μS)	Throughput GBits/Sec
HCTR	12068	85	79.65	89	1.117	0.703	**3.665**
HCH	13622	85	65.94	107	1.623	0.801	2.524
HCHfp	12970	85	66.50	96	1.443	0.990	2.837
XCB	13418	85	54.02	116	2.147	1.114	1.907
EME	10120	87	67.84	107	1.577	1.123	2.597
TET	12072	87	60.51	111	1.834	1.301	2.232
HEH	11545	85	72.44	75	1.035	0.591	**3.956**

Table: Hardware costs of the modes with an underlying full 10-stage pipelined 128-bit AES core when processing one sector of 32 AES blocks: Virtex 4 Implementation

The objective was to reach the speed of SATA hard disks 3 GBits/Sec.

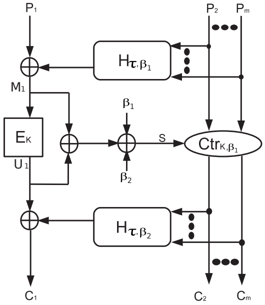
This work was published in:

- C. Mancillas-López, D. Chakraborty, and F. Rodríguez-Henríquez. **Reconfigurable Hardware Implementations of Tweakable Enciphering Schemes**, IEEE Transactions on Computers, vol. 59, no. 11, pp. 1547-1561, November 2010.

HMCH

Sarkar at 2009 proposed to use BRW as hash function to construct efficient tweakable enciphering schemes.

Algorithm $Encrypt^{\tau}_{K,\beta_1,\beta_2,}(P_1, \ldots P_m)$
2. $M_1 \leftarrow H_{\tau,\beta_1}(P_1, \ldots, P_m)$
3. $U_1 \leftarrow E_K(M_1); S \leftarrow M_1 \oplus U_1 \oplus (\beta_1 \oplus \beta_2)$
4. $(C_2, \ldots, C_m) \leftarrow Ctr_{K,\beta_1,S}(P_2, \ldots P_m)$
5. $C_1 \leftarrow H_{\tau,\beta_2}(C_1, \ldots, C_m)$



$H_{R,\beta_1}(P_2, \ldots, P_{m-1}, P_m) = P_2\tau^{m-1} \oplus P_2\tau^{m-2} \oplus \ldots \oplus P_m\tau \oplus P_1 \oplus \beta_1$

$\mathrm{Ctr}_{K,S}(P_2, \ldots, P_{m-1}, P_m) = (P_2 \oplus E_K(S \oplus \beta_1), E_K(S \oplus x\beta_1), \ldots, P_m \oplus E_K(S \oplus x^{m-1}\beta_1))$

Efficient Implementations

The BRW polynomial BRW is defined recursively as follows:

$$
\begin{aligned}
\mathrm{BRW}_h() &= 0 \\
\mathrm{BRW}_h(P_1) &= P_1 \\
\mathrm{BRW}_h(P_1, P_2) &= P_1 + P_2 h \\
\mathrm{BRW}_h(P_1, P_2, P_3) &= (h + P_1)(h^2 + P_2) + P_3 \\
\mathrm{BRW}_h(P_1, P_2, \ldots, P_m) &= \mathrm{BRW}_h(P_1, P_2, \ldots, P_{t-1})(h^t + P_t) + \\
&\quad \mathrm{BRW}_h(P_{t+1}, \ldots, P_m)
\end{aligned}
$$

where $t \in \{4, 8, 16, \ldots\}$ and $t \leq m < 2t$

The number of multiplications is given by $\lfloor \frac{m}{2} \rfloor$.

Additions: $m + \lfloor \frac{m-3}{2} \rfloor$.

Squarings: $\lfloor lgm \rfloor$.

BRW-Polynomials

- We propose a framework to construct an efficient circuit to compute BRW polynomials using a pipelined multiplier.
- To achieve a good performance in the implementations of BRW polynomial, there are two important aspects:
  - Scheduling of the blocks of information, trying to have the pipeline always full.
  - The number of accumulators or registers required.

BRW-Polynomials Representation

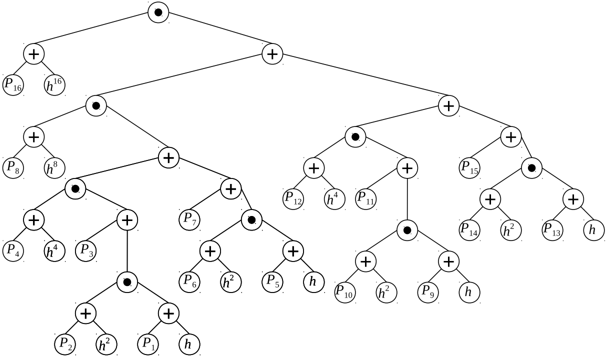
Let's see the BRW-Polynomial with 16 coefficients

$$\begin{aligned}
\mathrm{BRW}_h(P_1, ..., P_{16}) \quad = \quad & ((((h + P_1)(h^2 + P_2) + P_3)(h^4 + P_4) \\
& +(h + P_5)(h^2 + P_6) + P_7)(h^8 + P_8) \\
& +((h + P_9)(h^2 + P_{10}) + P_{11})(h^4 + P_{12}) \\
& +(h + P_{13})(h^2 + P_{14}) + P_{15})(h^{16} + P_{16})
\end{aligned}$$

The total number of operations are 8 multiplications, 4 squarings and 19 additions.

BRW-Polynomials Representation

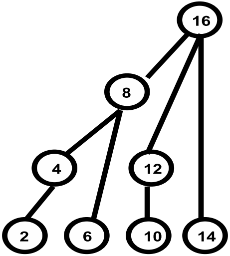
It can be represented as a tree T_m.



$$\mathrm{BRW}_h(P_1, \ldots, P_{16}) = ((((h+P_1)(h^2+P_2)+P_3)(h^4+P_4)+(h+P_5)(h^2+P_6)+P_7)(h^8+P_8)$$
$$+((h+P_9)(h^2+P_{10})+P_{11})(h^4+P_{12})+(h+P_{13})(h^2+P_{14})+P_{15})(h^{16}+P_{16})$$

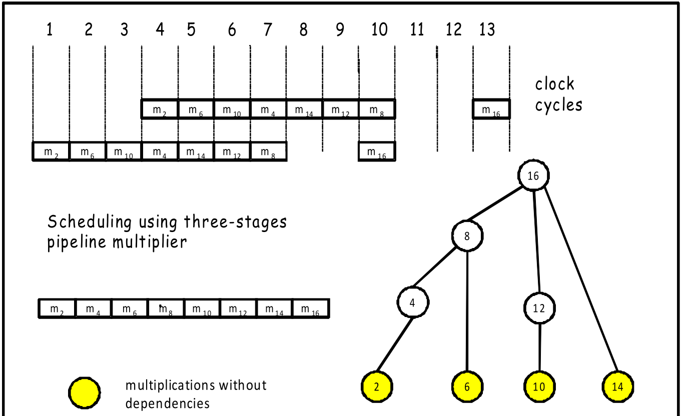
BRW-Polynomials Representation

It can be represented as a tree T_m.



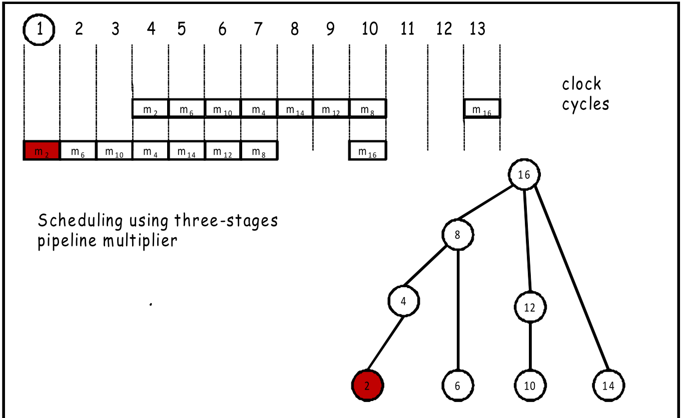
$$\begin{aligned}\mathrm{BRW}_h(P_1, \ldots, P_{16}) \quad &= \quad ((((h + P_1)(h^2 + P_2) + P_3)(h^4 + P_4) + (h + P_5)(h^2 + P_6) + P_7)(h^8 + P_8) \\ &\quad +((h + P_9)(h^2 + P_{10}) + P_{11})(h^4 + P_{12}) + (h + P_{13})(h^2 + P_{14}) + P_{15})(h^{16} + P_{16})\end{aligned}$$

Scheduling of the blocks



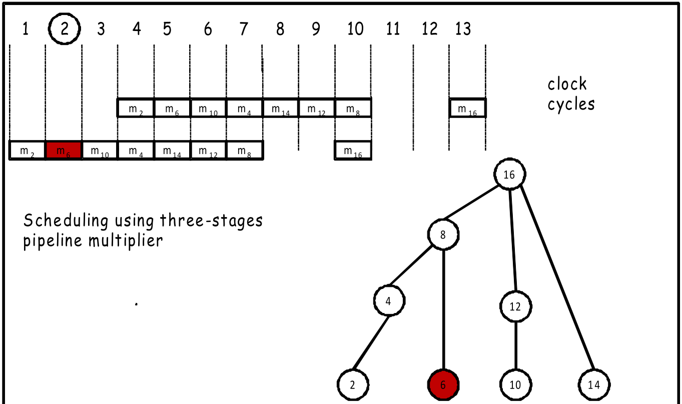
$$H(P_1,\ldots,P_{16}) = \left(\begin{array}{l} \left(\left(\left((h+P_1)(h^2+P_2)+P_3\right)(h^4+P_4)+(h+P_5)(h^2+P_6)+m_7\right)(h^8+P_8)+ \right. \\ \left. +\left(\left((h+P_9)(h^2+P_{10})+P_{11}\right)(h^4+P_{12})+(h+P_{13})(h^2+P_{14})+P_{15}\right)\right) \end{array} \right)(h^{16}+m_{16})$$

Scheduling of the blocks



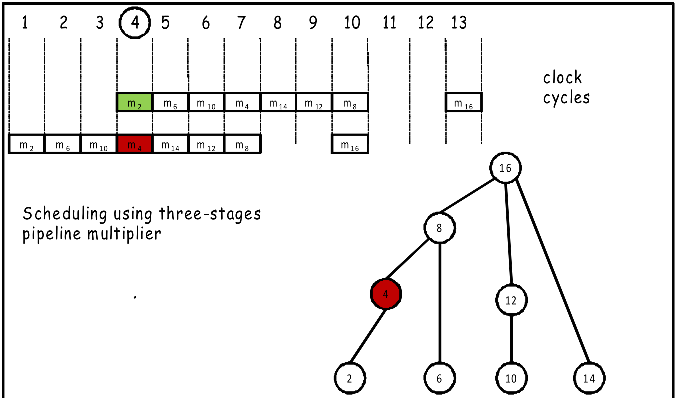
$$H(P_1,\ldots,P_{16}) = \left(\begin{array}{l} \left(\left(\left(h+P_1\right)\left(h^2+P_2\right)+P_3\right)\left(h^4+P_4\right)+\left(h+P_5\right)\left(h^2+P_6\right)+m_7\right)\left(h^8+P_8\right)+ \\ +\left(\left(h+P_9\right)\left(h^2+P_{10}\right)+P_{11}\right)\left(h^4+P_{12}\right)+\left(h+P_{13}\right)\left(h^2+P_{14}\right)+P_{15} \end{array} \right)\left(h^{16}+m_{16}\right)$$

Scheduling of the blocks



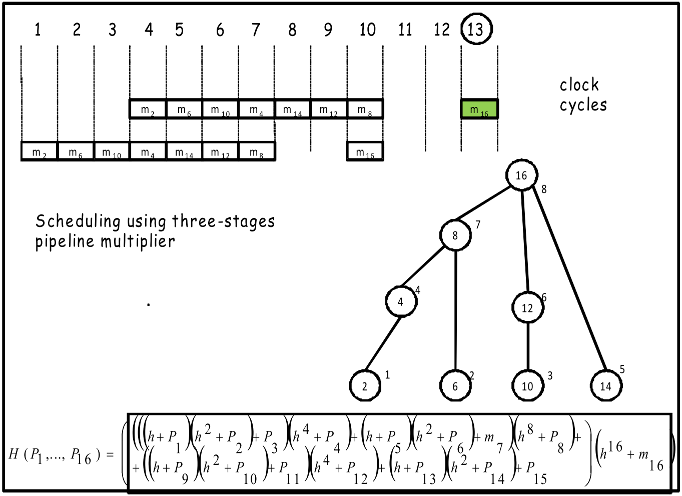
$$H(P_1,\ldots,P_{16}) = \left(\begin{array}{l} \left(\left(\left(\left(h+P_1\right)\left(h^2+P_2\right)+P_3\right)\left(h^4+P_4\right)+\left(h+P_5\right)\left(h^2+P_6\right)+m_7\right)\left(h^8+P_8\right)+ \\ +\left(\left(h+P_9\right)\left(h^2+P_{10}\right)+P_{11}\right)\left(h^4+P_{12}\right)+\left(h+P_{13}\right)\left(h^2+P_{14}\right)+P_{15} \end{array} \right)\left(h^{16}+m_{16}\right)$$

Scheduling of the blocks



$$H(P_1, \ldots, P_{16}) = \left(\begin{array}{l} \left(\left(\left((h+P_1)h^2+P_2\right)+P_3\right)h^4+P_4\right)+(h+P_5)(h^2+P_6)+m_7\,(h^8+P_8)+ \\ +\left(\left((h+P_9)(h^2+P_{10})+P_{11}\right)(h^4+P_{12})+(h+P_{13})(h^2+P_{14})+P_{15}\right) \end{array} \right)(h^{16}+m_{16})$$

Scheduling of the blocks



Optimal Scheduling

Theorem

Let $H_h(X_1, X_2, \ldots, X_m)$ *be a BRW polynomial and let* $p = \lfloor m/2 \rfloor$ *be the number of nodes in the corresponding collapsed tree. Let clks be the number of clock cycles taken by* Schedule *to schedule all nodes, then,*

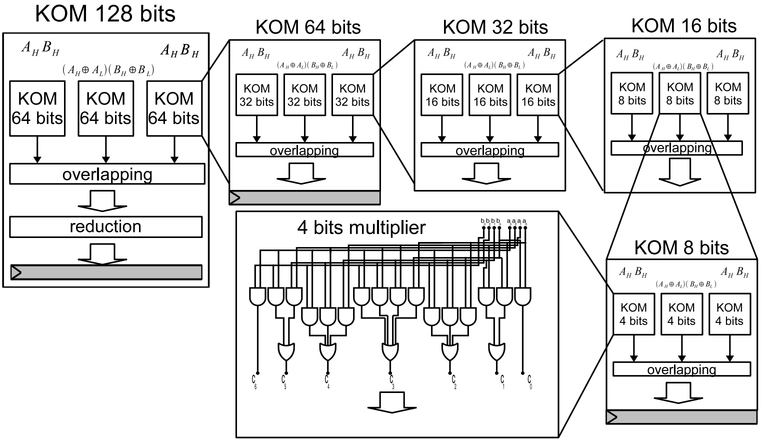
1. *If* NS $= 2$, *and* $p \geq 3$, $clks = p + 1$ *if* $p \equiv 0 \bmod 4$*; and* $clks = p$ *otherwise.*
2. *If* NS $= 3$ *and* $p \geq 7$, *then*

$$clks = \begin{cases} p+2 & \text{if } p \equiv 0 \bmod 4 \\ p+1 & \text{if } p \equiv 1 \bmod 4 \\ p+1 & \text{if } p \equiv 2 \bmod 4 \\ p & \text{if } p \equiv 3 \bmod 4 \end{cases}$$

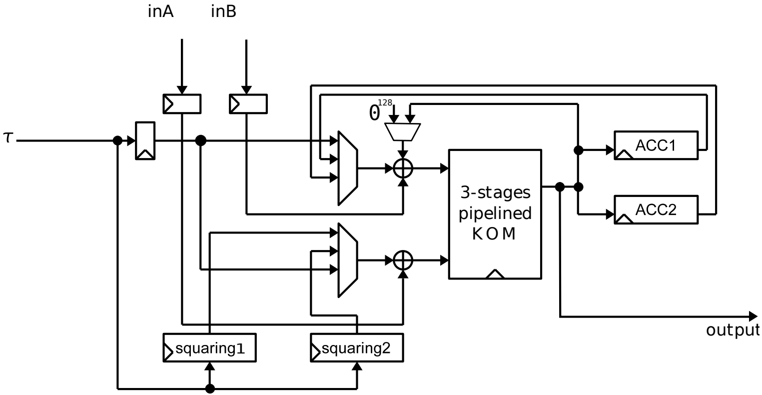
Karatsuba-Ofman Multiplier

To multiply $C = A * B$, we can write it as follows
$C = (A^L + x^{\frac{m}{2}} A^H) * (B^L + x^{\frac{m}{2}} B^H)$
$C = x^m A^H B^H + (A^H B^L + A^L B^H) x^{\frac{m}{2}} + A^L B^L$
$C = x^m A^H B^H + A^L B^L + (A^H B^H + A^L B^L + (A^H + A^L)(B^L + B^H)) x^{\frac{m}{2}} = x^m C^H + C^L$
The last equation has three multiplications with half of the initial bits. We can construct a multiplier recursively.

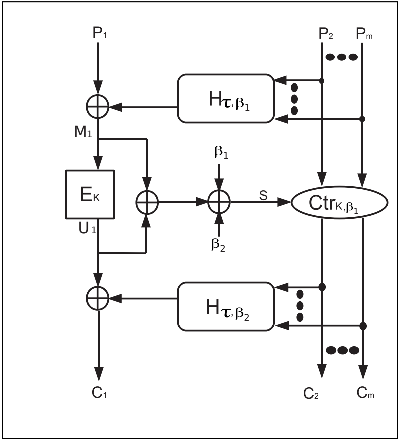
3 Stages Pipelined Karatusuba-Ofman Multiplier

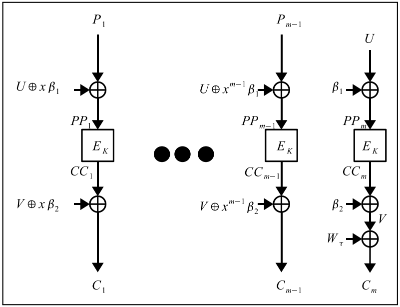


Architecture to BRW Polynomial Evaluation



HMCH and HEH


P1
P2
Pm
Hτ,β1
M1
β1
EK
S
CtrK,β1
U1
β2
Hτ,β2
C1
C2
Cm


P1
Pm−1
U
U ⊕ x β1
U ⊕ x^{m−1} β1
β1
PP1
PPm−1
PPm
EK
EK
EK
CC1
CCm−1
CCm
V ⊕ x β2
V ⊕ x^{m−1} β2
β2
V
Wτ
C1
Cm−1
Cm

Experimental Results

Table: Modes of operation on Virtex-5 device. AES-PEC: AES pipelined encryption core, AES-PDC: AES pipelined decryption core, AES-SDC: AES sequential decryption core, SOF : squares computed on the fly, SPC: squares pre-computed

Mode	Implementation Details	Slices	Frequency (MHz)	Clock Cycles	Time (nS)	Throughput (Gbits/Sec)
HMCH[BRW]-1	2 AES-PEC, 1 AES-SDC, SOF	8040	211.785	66	311.637	13.143
HMCH[BRW]-2	2 AES-PEC, 1 AES-SDC, SPC	8140	212.589	66	310.458	13.193
HMCH[BRW]-3	1 AES-PEC, 1 AES-SDC, SOF	6112	223.364	80	358.160	11.436
HEH[BRW]-1	2 AES-PEC, 2 AES-PDC, SOF	11850	202.856	55	271.128	15.170
HEH[BRW]-2	2 AES-PEC, 2 AES-PDC, SPC	12002	203.894	55	269.748	**15.184**
HEH[BRW]-3	1 AES-PEC, 1 AES-PDC, SOF	8012	218.384	69	315.957	12.964
HMCH[Poly]	1 AES-PEC, 1 AES-SDC	**5345**	225.485	94	416.879	9.825
HEH[Poly]	1 AES-PEC, 1 AES-PDC	6962	218.198	83	380.388	10.768

Outline

Preliminaries

Tweakable Enciphering Schemes
- Implementations
- BRW Polynomials

Lightweight Disk Encryption
- STES
- Implementation

New Model for Disk Encryption
- BCTR
- Implementation

Open Problems

The Case of Small Devices

- Small devices like mobile phones, cameras etc. have non trivial amount of storage.
- These devices generally have flash memories as storage.
- Constrained in terms of power utilization and area.

The Case of Small Devices

- Small devices like mobile phones, cameras etc. have non trivial amount of storage.
- These devices generally have flash memories as storage.
- Constrained in terms of power utilization and area.

Questions?

- Are the current schemes suitable for small devices?
- Can light-weight crypto be used for designing storage encryption?
- Light weight block ciphers may not be suitable for the $\frac{\sigma^2}{2^n}$ bounds of the existing TES.
  - Can the security bounds be improved?
  - Can we use pseudorandom generators (stream ciphers)?

STES

- A light weight tweakable enciphering scheme.
- We use special type of hash functions which can be implemented using multipliers with varying data paths.
  - Multilinear Universal Hash (MLUH)
  - Pseudo Dot Product (PD)
- Additionally STES uses stream ciphers with low hardware footprints.

Multilinear Universal Hash

A MLUH (Multilinear Universal Hash) with data path d takes in as input:

- A message $M = M_1||M_2||\cdots||M_m$, where each $|M_i| = d$.
- A key $K = K_1||K_2||\ldots||K_{m+b-1}$, where $|K_i| = d$ and $b \geq 1$.

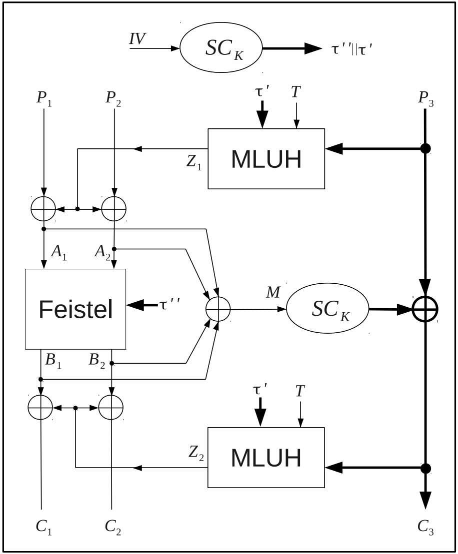
With these inputs MLUH produces a bd bit output. We define

$$\mathrm{MLUH}_K^{d,b}(M) = h_1||h_2||\cdots||h_b,$$

where

$$\begin{aligned}
h_1 &= M_1 \cdot K_1 \oplus M_2 \cdot K_2 \oplus ... \oplus M_m \cdot K_m \\
h_2 &= M_1 \cdot K_2 \oplus M_2 \cdot K_3 \oplus ... \oplus M_m \cdot K_{m+1} \\
&. \\
&. \\
h_b &= M_1 \cdot K_b \oplus M_2 \cdot K_{b+1} \oplus .... \oplus M_m \cdot K_{b+m-1},
\end{aligned}$$

STES



$\text{Feistel}^{\ell,d}_{K,\tau''}(A_1, A_2)$

1. $b \leftarrow \lceil \frac{\ell}{d} \rceil$
2. $H_1 \leftarrow \text{MLUH}^{d,b}_{\tau''}(A_1)$;
3. $F_1 \leftarrow H_1 \oplus A_2$;
4. $G_1 \leftarrow \text{SC}^{\ell}_{K}(F_1)$;
5. $F_2 \leftarrow A_1 \oplus G_1$;
6. $G_2 \leftarrow \text{SC}^{\ell}_{K}(F_2)$;
7. $B_2 \leftarrow F_1 \oplus G_2$;
8. $H_2 \leftarrow \text{MLUH}^{d,b}_{\tau''}(B_2)$;
9. $B_1 \leftarrow H_2 \oplus F_2$;
10. **return**(B_1, B_2);

Security of STES

The following theorem specifies the security of STES.

Theorem

Let $\delta \xleftarrow{\$} \mathsf{Func}(\ell, L)$ and $\mathsf{STES}[\delta]$ be STES instantiated with the function δ in place of the stream cipher. Then, for any arbitrary adversary $\mathcal{A}$ which asks at most q queries we have

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathsf{STES}[\delta]}(\mathcal{A}) \leq \frac{10q^2 + 3q}{2^{\ell}}.$$

Security of STES

The following theorem specifies the security of STES.

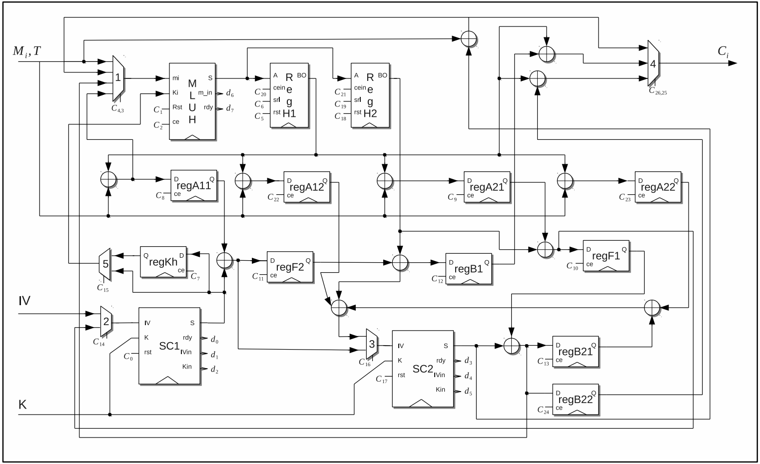
Theorem

Let $\delta \xleftarrow{\$} \mathsf{Func}(\ell, L)$ and $\mathsf{STES}[\delta]$ be STES instantiated with the function δ in place of the stream cipher. Then, for any arbitrary adversary $\mathcal{A}$ which asks at most q queries we have

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathsf{STES}[\delta]}(\mathcal{A}) \leq \frac{10q^2 + 3q}{2^\ell}.$$

If the underlying stream cipher $\mathsf{SC} : 0, 1^\ell \rightarrow \{0, 1\}^L$ is a random function then STES is secure.

General Architecure for STES



Results

Mode	Logic cells	Cycles	Frequency (MHz)	Throughput (Mbps)	TPA	Static power (mW)	Dynamic power (mW)	Tolal power (mW)
SCTES-T-1b	2013	13765	140.29	41.75	5.06	0.16	38.49	38.65
SCTES-T-4b	2379	3449	138.15	164.07	16.84	0.16	49.60	49.76
SCTES-T-8b	2676	1729	165.78	321.66	29.35	0.16	58.45	58.61
SCTES-T-16b	3402	871	133.07	625.78	44.91	0.16	95.17	95.33
SCTES-T-40b	5252	355	128.08	1477.79	68.65	0.16	156.27	156.42
SCTES-G-1b	2165	10501	135.26	52.76	5.95	0.16	42.55	42.91
SCTES-G-4b	2708	2633	130.87	203.59	18.35	0.16	49.63	49.78
SCTES-G-8b	3242	1321	128.59	398.71	30.03	0.16	67.26	67.42
SCTES-G-16b	4204	667	120.76	741.58	43.07	0.16	92.77	92.93
SCTES-G-32b	6092	339	118.66	1434.81	57.50	0.16	119.19	119.35
SCTES-M-1b	1720	10117	130.75	52.94	7.51	0.16	42.49	42.65

Table: TES Lattice ICE40.

Outline

Preliminaries

Tweakable Enciphering Schemes
- Implementations
- BRW Polynomials

Lightweight Disk Encryption
- STES
- Implementation

New Model for Disk Encryption
- BCTR
- Implementation

Open Problems

New Model for Disk Encryption

- Till now the accepted proposal for disk encryption are TES.
- Why? Mainly because of the length preserving requirement.
- We ask the question:

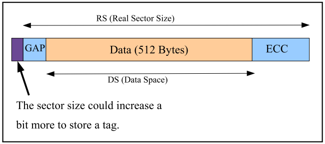
  Is length preserving that important?.

A real sector:

A sector storing 512 bytes user data is not 512 bytes long.

Table: Extra format overhead

Sector size (in bytes)	Tag size (in bits) 64	96	128
512	1.56%	2.34%	3.13%
4096	0.19%	0.29%	0.39%
8192	0.09%	0.14%	0.19 %



Which Encryption Scheme?

Authenticated Encryption

$AE(N, H, M) = N, \tau, C$

AEs need extra space to store N and τ.

Which Encryption Scheme?

Authenticated Encryption

$AE(N, H, M) = N, \tau, C$

AEs need extra space to store N and τ.

Deterministic Authenticated Encryption

$DAE(H, M) = \tau, C$

DAEs need extra space to store only τ.

Which Encryption Scheme?

We propose to use Deterministic Authenticated Encryption modes (DAEs).

Definition

A DAE is a tuple $\Psi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$

- They provide authentication and privacy.
- Authentication is on message and associated data.
- A pseudorandom function and an IV based encryption scheme are required in order to construct a DAE.
- DAEs are not length preserving. Ciphertext is a pair τ, C where τ is a tag for authentication.

Security of DAEs

Let's Ψ be a DAE, it offers privacy:

$$\textbf{Adv}_{\Psi}^{DAE-priv}(\mathcal{A}) = \left|\Pr\left[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_K(.,.)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(.,.)} \Rightarrow 1\right]\right|$$

DAE is secure when $\textbf{Adv}_{\Psi}^{DAE-priv}(\mathcal{A})$ is small for all efficient adversaries. It offers authentication:

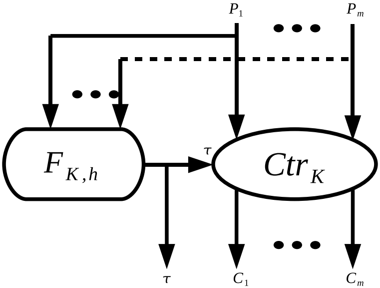
$$\textbf{Adv}_{\Psi}^{DAE-auth}(\mathcal{A}) = \Pr[\mathcal{A}^{\mathcal{E}_K(.,.,.)} \text{ forges }]$$

If $\textbf{Adv}_{\Psi}^{DAE-auth}(\mathcal{A})$ is small, this signify that it must hard for an adversary to create a valid ciphertext.

BCTR: A Novel Disk Cipher

Encrypt.$BCTR^{T}_{K,h}(P_1||P_2||...||P_m)$

1. $\alpha = E_K(0)$; $\beta = E_K(1)$;
2. $\gamma \leftarrow h \cdot \mathrm{BRW}_h(P_1||P_2||\ldots||P_m||T)$
3. $\tau \leftarrow E_K(\gamma \oplus \alpha)$;
4. **for** $j = 1$ to m
5. $\quad R_j \leftarrow E_K(\tau \oplus x^j \beta)$
6. $\quad C_j \leftarrow R_j \oplus P_j$
7. **endfor**
8. **return** $(C_1||C_2||\ldots||C_m||\tau)$



Computational Cost: $m + 3$ Block Cipher Calls and $1 + \lfloor (m+1)/2 \rfloor$ Multiplications. It increases the ciphertext in 128 bits.

Efficiency of BCTR

Mode	[BC]	[M]	[BCK]	[OK]
CMC	$2m+1$	–	1	–
EME	$2m+2$	–	1	–
XCB	$m+1$	$2(m+3)$	3	2
HCTR	m	$(2m+1)$	1	1
HCHfp	$m+2$	$2(m-1)$	1	1
TET	$m+1$	$2m$	2	3
Constructions Sarkar's proposals using normal polynomials	$m+1$	$2(m-1)$	1	1
Constructions Sarkar's proposals using BRW polynomials	$m+1$	$2+2\lfloor(m-1)/2\rfloor$	1	
BCTR	$m+3$	$1+\lfloor(m+1)/2\rfloor$	1	1
SIV [Rogaway and Srimptom]	$2m+3$	–	2	–
HBS [Iwata and Yasuda]	$m+2$	$m+3$	1	–
BTM [Iwata and Yasuda]	$m+3$	m	1	–

[BC]: Number of block-cipher calls; [M]: Number of multiplications, [BCK]: Number of blockcipher keys, [OK]: Other keys, including hash keys.

Security of BCTR

Theorem

Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ *be a block-cipher secure in the PRP sense. Let* $\mathcal{A}$ *be an adversary attacking* $\mathrm{BCTR}[E]$ *who asks* q *queries, then there exist an adversary* $\mathcal{A}'$ *such that*

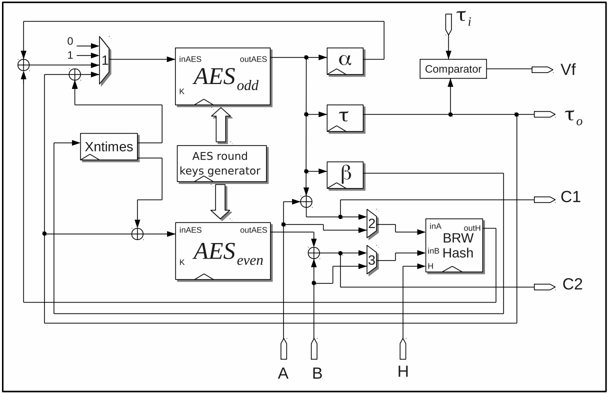
$$\mathbf{Adv}^{DAE-priv}_{\mathrm{BCTR}[E]}(\mathcal{A}) \leq \frac{14m^2q^2}{2^n} + \mathbf{Adv}^{\mathrm{prp}}_{E}(\mathcal{A}') \quad (1)$$

$$\mathbf{Adv}^{DAE-auth}_{\mathrm{BCTR}[E]}(\mathcal{A}) \leq \frac{1}{2^n} + \frac{18m^2q^2}{2^n} + 2\mathbf{Adv}^{\mathrm{prp}}_{E}(\mathcal{A}') \quad (2)$$

where $\mathcal{A}'$ *asks* $O(q)$ *queries and run for time* $t + O(q)$ *where* t *is the running time of* $\mathcal{A}$*.*

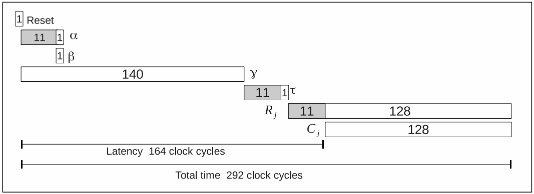
Implementation of BCTR

We generate the following architecture for BCTR using Virtex 5 FPGAs. The message length is 4 KB.

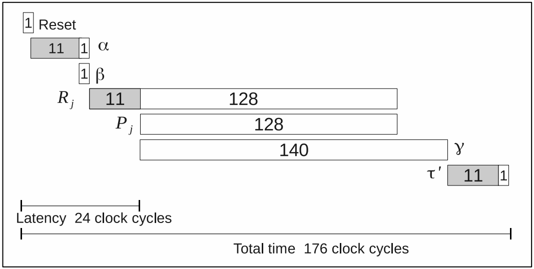


Timing Analysis

Encryption



Decryption



Results

Mode	Implementation Details	Slices	Frequency (MHz)	Throughput (Gbits/Sec)	$\frac{1}{(\text{Slice}*\text{Time})}$
BCTR-1	2 AES-PEC, SOF	7876	291.29	**32.69/54.23**	94.69
BCTR-2	1 AES-PEC, SOF	5517	292.56	**17.12/30.34**	126.66
HMCH-1	2 AES-PEC, 1 AES-SDC, SOF	8040	211.79	13.14	399.11
HMCH-2	1 AES-PEC, 1 AES-SDC, SOF	6112	223.36	11.44	456.81
HEH-1	2 AES-PEC, 2 AES-PDC, SOF	11850	202.86	15.17	311.25
HEH-2	1 AES-PEC, 1 AES-PDC, SOF	8012	218.38	12.96	395.02
BTM*	-	6421	291.715	16.865	-
HBS*	-	8285	246.430	14.34	-

Table: Modes of operation on Virtex-5 device. AES-PEC: AES pipelined encryption core, AES-PDC: AES pipelined decryption core, AES-SDC: AES sequential decryption core, SOF : squares computed on the fly, SPC: squares pre-computed

*The performance for BTM and HBS was taken from the master thesis of Alejandro García Luna.

Outline

Preliminaries

Tweakable Enciphering Schemes
- Implementations
- BRW Polynomials

Lightweight Disk Encryption
- STES
- Implementation

New Model for Disk Encryption
- BCTR
- Implementation

Open Problems

Open Problems

- Development of prototypes for disk encryption.
- More implementations.
- Key management.
- Counter measures against side channel attacks.
- Stronger security notions.

Credits

Most of the work presented here was done jointly with:

Debrup Chakraborty
CINVESTAV-IPN, Mexico

Francisco Rodríguez-Henríquez
CINVESTAV-IPN, Mexico

Palash Sarkar
Indian Statistical Institute, Kolkata

Thanks for your Attention

Questions?