Minimum Number of Multiplications of ΔU Hash Functions

Mridul Nandi

Indian Statistical Institute, Kolkata

mridul@isical.ac.in

March 4, FSE-2014, London

Authentication: The Popular Story

1. Alice and Bob share a secret key K.
2. Data Integrity: Alice sends M along with tag $T = Tag_K(M)$ to Bob. Bob can verify.

Authentication: The Popular Story

1. Alice and Bob share a secret key K.
2. Data Integrity: Alice sends M along with tag $T = Tag_K(M)$ to Bob. Bob can verify.

Examples from Scratch.

3. Fixed Input-Length (FIL) and Fixed Output-Length (FOL) Prf (or Mac) f
   - *Blockcipher*
   - *compression function* of a hash (key is injected through chain or message block).

Authentication: The Popular Story

1. Alice and Bob share a secret key K.
2. Data Integrity: Alice sends M along with tag $T = Tag_K(M)$ to Bob. Bob can verify.

Examples from Scratch.

3. Fixed Input-Length (FIL) and Fixed Output-Length (FOL) Prf (or Mac) f
   - *Blockcipher*
   - *compression function* of a hash (key is injected through chain or message block).
4. Domain extensions (construction of VIL) based on
   1. blockcipher (variants of CBC, PMAC etc.) and
   2. compression functions (HMAC, EMD, sandwich, MDP etc.).

VIL-FOL Authentication from FIL-FOL

1. Composition Method: Let H be an n-bit (unkeyed) collision resistant hash function then $f \circ H$ is Prf (also Mac).

   $\boxed{\textbf{Question}.\ \text{Is } f(N) \oplus H(M) \text{ Nonce-based Mac?}}$ (nonce can repeat only for forging message)

VIL-FOL Authentication from FIL-FOL

1. Composition Method: Let H be an n-bit (unkeyed) collision resistant hash function then $f \circ H$ is Prf (also Mac).

   Question. Is $f(N) \oplus H(M)$ Nonce-based Mac? (nonce can repeat only for forging message)

2. NO, given $T = f(N) \oplus H(M) \Rightarrow T' = T \oplus H(M) \oplus H(M')$ is also tag. So we need keyed hash H_k.

   Question. Is $f(N) \oplus H_k(M)$ Nonce-based Mac?

VIL-FOL Authentication from FIL-FOL

1. Composition Method: Let H be an n-bit (unkeyed) collision resistant hash function then $f \circ H$ is Prf (also Mac).

   Question. Is $f(N) \oplus H(M)$ Nonce-based Mac? (nonce can repeat only for forging message)

2. NO, given $T = f(N) \oplus H(M) \Rightarrow T' = T \oplus H(M) \oplus H(M')$ is also tag. So we need keyed hash H_k.

   Question. Is $f(N) \oplus H_k(M)$ Nonce-based Mac?

3. Not always, if $\Pr[H_k(M) \oplus H_k(M') = \delta]$ is high then

$$T = f(N) \oplus H_k(M) \Rightarrow \Pr[f(N) \oplus M' = T \oplus \delta] \text{ is high.}$$

Definitions of ΔU and Universal hash.

1. **Differential probability**: For all $M \neq M'$ and for all δ, H_k is called ϵ-ΔU if
   differential probability $\Pr[H_k(M) \oplus H_k(M') = \delta] \leq \epsilon$.
   - Denote the event $\Delta H_k(M) = \delta$. $(\Delta f(x) := f(x) - f(x'))$
   - For "small" ϵ, $f(N) \oplus H_k(M)$ is Mac (nonce-based).

Definitions of ΔU and Universal hash.

1. **Differential probability**: For all $M \neq M'$ and for all δ, H_k is called ϵ-ΔU if

   differential probability $\Pr[H_k(M) \oplus H_k(M') = \delta] \leq \epsilon$.

   - Denote the event $\Delta H_k(M) = \delta$. $(\Delta f(x) := f(x) - f(x'))$
   - For "small" ϵ, $f(N) \oplus H_k(M)$ is Mac (nonce-based).

2. **Collision probability**: When we restrict to $\delta = 0$, i.e., collision probability $\Pr[H_k(M) = H_k(M')] \leq \epsilon$ we say that H_k is ϵ-U hash.
   - For "small" ϵ, $f \circ H_k$ is Prf and so Mac.

3. Main object of the talk - **On optimum complexity of ΔU (or Universal) hash functions**.

Example. Multi-Linear (ML) Hash

Convention. Galois field $\mathbb{F}_{2^n}$ (elements are called **blocks**).
$K_1, K_2, \ldots \xleftarrow{\$} \mathbb{F}_{2^n}$ and $\mathbf{K}$ to denote vector of keys.

1. $\forall m_1, m_2 \in \mathbb{F}_{2^n}$, $(m_1, m_2) \mapsto \boxed{m_1 K_1 + m_2 K_2}$.

Example. Multi-Linear (ML) Hash

Convention. Galois field $\mathbb{F}_{2^n}$ (elements are called **blocks**).
$K_1, K_2, \ldots \xleftarrow{\$} \mathbb{F}_{2^n}$ and $\mathbf{K}$ to denote vector of keys.

1. $\forall m_1, m_2 \in \mathbb{F}_{2^n}$, $(m_1, m_2) \mapsto \boxed{m_1 K_1 + m_2 K_2}$.
2. Differential property: For any $(m_1, m_2) \neq (m'_1, m'_2)$, $\delta \in \mathbb{F}_{2^n}$,

$$\Pr[\underbrace{m_1 K_1 + m_2 K_2 = m'_1 K_1 + m'_2 K_2 + \delta}_{\text{differential event } E}] = \frac{1}{2^n}.$$

Example. Multi-Linear (ML) Hash

Convention. Galois field $\mathbb{F}_{2^n}$ (elements are called **blocks**).
$K_1, K_2, \ldots \xleftarrow{\$} \mathbb{F}_{2^n}$ and $\mathbf{K}$ to denote vector of keys.

1. $\forall m_1, m_2 \in \mathbb{F}_{2^n}$, $(m_1, m_2) \mapsto \boxed{m_1 K_1 + m_2 K_2}$.
2. Differential property: For any $(m_1, m_2) \neq (m'_1, m'_2)$, $\delta \in \mathbb{F}_{2^n}$,

$$\Pr[\underbrace{m_1 K_1 + m_2 K_2 = m'_1 K_1 + m'_2 K_2 + \delta}_{\text{differential event } E}] = \frac{1}{2^n}.$$

3. **Proof**. If $m_1 \neq m'_1$ (i.e., $\Delta m_1 \neq 0$) then result follows conditioning K_2.

Example: **Pseudo dot-product** (PDP) Hash

1. $\forall m_1, m_2 \in \mathbb{F}_{2^n}, (m_1, m_2) \mapsto \boxed{(m_1 + K_1)(m_2 + K_2)}$.
2. Differential property: PDP $=$ ML $+ K_1 K_2 + m_1 m_2$. Function of key gets canceled and messages goes to δ.

Example: **Pseudo dot-product** (PDP) Hash

1. $\forall m_1, m_2 \in \mathbb{F}_{2^n}, (m_1, m_2) \mapsto \boxed{(m_1 + K_1)(m_2 + K_2)}$.
2. Differential property: PDP $=$ ML $+K_1K_2 + m_1m_2$. Function of key gets canceled and messages goes to δ.
3. 1 (or $\ell/2$) mult for 2 (or ℓ even) blocks (compare with ML).

$$(m_1 + K_1)(m_2 + K_2) + \cdots + (m_{\ell-1} + K_{\ell-1})(m_\ell + K_\ell).$$

Question 1. Can we have ΔU hash for ℓ message blocks requiring less than $\ell/2$ multiplications?

Linear function (in message and keys) has no mult and can not be universal. Note # multiplicands is $2c$ for c mult and these behave like linear, so **due to entropy should not hope**.

Multi-block Hash

1. d-block hash $H = (H_1, \ldots, H_d)$ outputs $\mathbb{F}_{2^n}^d$ (nd bits) We need it possibly for
   - larger hash output or
   - work with smaller field size might lead to better performance. For example, 64 bit system wants to produce 128 bits.

   Examples.
2. d-independent hash: $H = (H_{\mathbf{K}_1}, \ldots, H_{\mathbf{K}_d})$ where H is ΔU and $\mathbf{K}_i$'s are independent.
   - Larger keys,
   - parallel.
3. Toeplitz hash (applied to ML and PDP): Less keys and parallel. requires about $d \times \ell$ or $d \times \ell/2$ multiplications.

Toeplitz Hash for ML

$$\begin{bmatrix} m_1 & m_2 & \dots & m_\ell & 0 & \dots & 0 & 0 \\ 0 & m_1 & \dots & m_{\ell-1} & m_\ell & \dots & 0 & 0 \\ 0 & 0 & \dots & m_{\ell-2} & m_{\ell-1} & \dots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \dots & m_{\ell-d+1} & & \dots & m_{\ell-1} & m_\ell \end{bmatrix} . \begin{pmatrix} K_1 \\ K_2 \\ K_3 \\ \vdots \\ K_{\ell+d-1} \end{pmatrix}$$

- Can be computed in $d \times \ell$ multiplications.
- Winograd showed that it can not be computed in "**less than**" $d \times \ell$ mult.

Toeplitz Hash for PDP

$$\begin{bmatrix} (m_1, m_2) & (m_3, m_4) & \ldots & (m_{\ell-1}, m_\ell) & 0 & \ldots & 0 \\ 0 & (m_1, m_2) & \ldots & (m_{\ell-3}, m_{\ell-2}) & (m_{\ell-1}, m_\ell) & \ldots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \end{bmatrix} \bullet \begin{pmatrix} (K_1, K_2) \\ (K_3, K_4) \\ \vdots \end{pmatrix}$$

- Here, $(m, m') \bullet (K, K') = (m + K) \cdot (m' + K')$.
- It can be computed in $d \times \ell/2$ multiplications for computing d-block hash.
- No known better algorithm.

Question 1-d. Can we have d-block ΔU hash for ℓ message blocks requiring less than $d \times \ell/2$ multiplications?

Question 1-d. Can we have d-block ΔU hash for ℓ message blocks requiring less than $d \times \ell/2$ multiplications?

1. **Try-1**: $(m_1 K_1 + m_2 K_2, m_1 K_2 + m_2 K_1) \rightarrow 3$ mult instead of 4.

   However, 2^{-n} differential probability. Expect 2^{-2n} and **about 2^{-nd} for d-blk hash**. We always have $(H_1, \ldots, H_1)$.

Question 1-d. Can we have d-block ΔU hash for ℓ message blocks requiring less than $d \times \ell/2$ multiplications?

1. **Try-1**: $(m_1K_1 + m_2K_2, m_1K_2 + m_2K_1) \rightarrow$ 3 mult instead of 4.

   However, 2^{-n} differential probability. Expect 2^{-2n} and **about 2^{-nd} for d-blk hash**. We always have $(H_1, \ldots, H_1)$.

2. **Try-2**: Let α be a primitive element of $\mathbb{F}_{2^n}$.

$$\boxed{(m_1K_1 + m_2K_2 + m_3K_3,\ \alpha^2 m_1K_1 + \alpha m_2K_2 + m_3K_3)}$$

where $m_3 = m_1 + m_2$.

- 2^{-2n} differential probability,
- 3 mult (mult by α is efficient) for 4 blocks with PDP.
- Our construction EHC requires less than $d \times \ell/2$ mult.

Final Question: Multiplication Complexity.

1. Minimum how much mult is necessary for d-blk hash?

1. Minimum how much mult is necessary for d-blk hash?
2. Need to define a complexity metric for hash.
   - Multiplication complexity (MC) for a polynomial (or d polynomials) - Minimum # mult to compute a polynomial (or d polynomials).
   - MC for $H_1 := m_1K_1 + m_2K_2$ and $H_2 := m_1K_2 + m_2K_1$ are individually 2 and for (H_1, H_2) is 3.

Final-Question. Minimum MC for a good ΔU hash function.

Results and Outline of Rest of the Talk.

1. Definition of Multiplication Complexity (MC).

Results and Outline of Rest of the Talk.

1. Definition of Multiplication Complexity (MC).
2. Answer 1. The MC for any “good” ΔU hash function for ℓ block messages is $\ell/2$.

Results and Outline of Rest of the Talk.

1. Definition of Multiplication Complexity (MC).
2. Answer 1. The MC for any "good" ΔU hash function for ℓ block messages is $\ell/2$.
3. Answer 1-d. The MC for any "good" d-blk ΔU hash function for ℓ block messages is $(d-1)+\ell/2$.

Results and Outline of Rest of the Talk.

1. Definition of Multiplication Complexity (MC).
2. Answer 1. The MC for any "good" ΔU hash function for ℓ block messages is $\ell/2$.
3. Answer 1-d. The MC for any "good" d-blk ΔU hash function for ℓ block messages is $(d-1)+\ell/2$.
4. A new construction ECH (Encode-then-Hash-then-Combine). Requires matching $(d-1)+\ell/2$ mult for $d \leq 4$.

Results and Outline of Rest of the Talk.

1. Definition of Multiplication Complexity (MC).
2. Answer 1. The MC for any "good" ΔU hash function for ℓ block messages is $\ell/2$.
3. Answer 1-d. The MC for any "good" d-blk ΔU hash function for ℓ block messages is $(d-1)+\ell/2$.
4. A new construction ECH (Encode-then-Hash-then-Combine). Requires matching $(d-1)+\ell/2$ mult for $d \leq 4$.
5. Future scope and Conclusion.

1. **Algebraic computation** C over variables $\mathbf{x} = (x_1, \ldots, x_s)$: sequence of addition and multiplications.
   - All consecutive additions $\rightarrow$ Linear function.
   - multiplicands are linear functions of $\mathbf{x}$ and v_j's (result of previous multiplications).

1. **Algebraic computation** C over variables $\mathbf{x} = (x_1, \ldots, x_s)$: sequence of addition and multiplications.
   - All consecutive additions $\rightarrow$ Linear function.
   - multiplicands are linear functions of $\mathbf{x}$ and v_j's (result of previous multiplications).
2. Want to compute PDP $(m_1 + K_1)(m_2 + K_2) + (m_3 + K_3)(m_4 + K_4)$.
   1. $L_1 = (m_1 + K_1), L_2 = (m_2 + K_2),\ v_1 = L_1 \cdot L_2$.

1. **Algebraic computation** C over variables $\mathbf{x} = (x_1, \ldots, x_s)$: sequence of addition and multiplications.
   - All consecutive additions $\rightarrow$ Linear function.
   - multiplicands are linear functions of $\mathbf{x}$ and v_j's (result of previous multiplications).
2. Want to compute PDP $(m_1 + K_1)(m_2 + K_2) + (m_3 + K_3)(m_4 + K_4)$.
   1. $L_1 = (m_1 + K_1), L_2 = (m_2 + K_2),\ v_1 = L_1 \cdot L_2$.
   2. $L_3 = (m_3 + K_3), L_4 = (m_4 + K_4)$, (these do not use v_1).
   3. $v_2 = L_3 \cdot L_4$.

1. **Algebraic computation** C over variables $\mathbf{x} = (x_1, \ldots, x_s)$: sequence of addition and multiplications.
   - All consecutive additions $\rightarrow$ Linear function.
   - multiplicands are linear functions of $\mathbf{x}$ and v_j's (result of previous multiplications).
2. Want to compute PDP
   $(m_1 + K_1)(m_2 + K_2) + (m_3 + K_3)(m_4 + K_4)$.
   1. $L_1 = (m_1 + K_1), L_2 = (m_2 + K_2),\ v_1 = L_1 \cdot L_2$.
   2. $L_3 = (m_3 + K_3), L_4 = (m_4 + K_4)$, (these do not use v_1).
   3. $v_2 = L_3 \cdot L_4$.
   4. $L_5 = v_1 + v_2$.

1. Want to compute Poly-hash $m_1 K + m_2 K^2 + m_3 K^3$.
   - $L_1 = m_3, L_2 = K,\ v_1 = L_1 \cdot L_2$.

1. Want to compute Poly-hash $m_1K + m_2K^2 + m_3K^3$.
   - $L_1 = m_3, L_2 = K$, $v_1 = L_1 \cdot L_2$.
   - $L_3 = v_1 + m_2$, (here we use v_1), $L_4 = K$.
   - $v_2 = L_3 \cdot L_4$.

Multiplication Complexity: **Algebraic Computation**

1. Want to compute Poly-hash $m_1 K + m_2 K^2 + m_3 K^3$.
   - $L_1 = m_3, L_2 = K,\ v_1 = L_1 \cdot L_2$.
   - $L_3 = v_1 + m_2$, (here we use v_1), $L_4 = K$.
   - $v_2 = L_3 \cdot L_4$.
   - $L_5 = v_2 + m_1,\ L_6 = K,\ v_3 = L_5 \cdot L_6$.
   - $L_7 = v_3$.

Multiplication Complexity: **Algebraic Computation**

1. Want to compute Poly-hash $m_1K + m_2K^2 + m_3K^3$.
   - $L_1 = m_3, L_2 = K,\ v_1 = L_1 \cdot L_2$.
   - $L_3 = v_1 + m_2$, (here we use v_1), $L_4 = K$.
   - $v_2 = L_3 \cdot L_4$.
   - $L_5 = v_2 + m_1,\ L_6 = K,\ v_3 = L_5 \cdot L_6$.
   - $L_7 = v_3$.
2. C with t mult can be described by $2t + 1$ linear functions: $L_1, \ldots, L_{2t+1}$ mapping to $\mathbb{F}_{2^n}$.
3. L_{2i-1} and L_{2i} are linear in $\mathbf{x}$ and $v_j := L_{2j-1} \cdot L_{2j}$, $1 \leq j < i$.
4. x_i's will be key and message blocks.
5. Constant multiplications. Efficient and linear.

Multiplication Complexity.

Algebraic computation: $C(x_1, \ldots, x_s)$.

1. For $j = 1$ to t
2. $v_j := L_{2j-1}(x_1, \ldots, x_s, v_1, \ldots, v_{j-1}) \cdot L_{2j}(x_1, \ldots, x_s, v_1, \ldots, v_{j-1})$;
3. Return $L_{2t+1}(x_1, \ldots, x_s, v_1, \ldots, v_t)$;

We say that $C(x_1, \ldots, x_s)$ computes the polynomial $P(x_1, \ldots, x_s)$ if $L_{2t+1}(x_1, \ldots, x_s, v_1, \ldots, v_t) = P$.

Definition (Multiplication complexity)

Multiplication complexity of a polynomial P is the **minimum** number of mult. over **all algebraic computations** computing P.

Multiplication Complexity for vector of Polynomials.

Algebraic computation: $C(x_1, \ldots, x_s)$ computing d polynomials.

1. For $j = 1$ to t
2. $v_j := L_{2j-1}(x_1, \ldots, x_s, v_1, \ldots, v_{j-1}) \cdot L_{2j}(x_1, \ldots, x_s, v_1, \ldots, v_{j-1})$;
3. Return $(L_{2t+1}(\mathbf{x}, \mathbf{v}), \ldots, L_{2t+d}(\mathbf{x}, \mathbf{v}))$; where $\mathbf{v} = (v_1, \ldots, v_t)$

We say that C computes the polynomial $(P_1, \ldots, P_d)$ if $L_{2t+i}(\mathbf{x}, \mathbf{v}) = P_i$, $1 \leq i \leq d$.

Definition (Multiplication complexity)

Multiplication complexity of a vector of polynomial $(P_1, \ldots, P_d)$ is the minimum number of mult. over all algebraic computations computing $(P_1, \ldots, P_d)$.

Some Examples of Multiplication Complexity.

1. Upper bound of MC: Construct an algebraic computation.
2. Lower bound of MC: requires some tricks, not obvious.

Examples.

Some Examples of Multiplication Complexity.

1. Upper bound of MC: Construct an algebraic computation.
2. Lower bound of MC: requires some tricks, not obvious.

Examples.

1. MC for x^n is $\log_2 n$. Note that by multiplying c times we can get degree at most 2^c.
2. Winograd had shown that MC for $m_1 K_1 + \ldots + m_\ell K_\ell$ is ℓ.
3. MC for Topelitz construction based on ML is ℓd.

1. Lower bound of $MC(p)$ for any fixed polynomial p is not obvious.

Lower Bound of MC.

1. Lower bound of $MC(p)$ for any fixed polynomial p is not obvious.

2. Here we target apparently more harder questions.

   What is $\min\{MC(p) : p \in \mathcal{H}\}$ where $\mathcal{H}$ is a family of polynomials having ΔU property?

Answer to Question-1.

Theorem

Let $t < \ell/2$. Let C compute $H(K_1, \ldots, K_r, m_1, \ldots, m_\ell)$ with t multiplications (i.e., $MC(H) \leq t$) then $\exists \mathbf{m} \neq \mathbf{m}' \in \mathbb{F}_{2^n}^{\ell}, \delta \in \mathbb{F}_{2^n}$,

$$Pr[H_{\mathbf{K}}(\mathbf{m}) \oplus H_{\mathbf{K}}(\mathbf{m}') = \delta] = 1.$$

Corollary

$MC(PDP) = \ell/2$, and it is optimum.

BRW (Bernstein-Rabin-Winograd) is also optimum (single key, but about $\ell 2^{-n}$-ΔU.

Theorem

Let $t < \ell/2$. Let C compute $H(K_1, \ldots, K_r, m_1, \ldots, m_\ell)$ with t multiplications (i.e., $MC(H) \leq t$) then $\exists \mathbf{m} \neq \mathbf{m}' \in \mathbb{F}_{2^n}^{\ell}, \delta \in \mathbb{F}_{2^n}$,

$$Pr[H_{\mathbf{K}}(\mathbf{m}) \oplus H_{\mathbf{K}}(\mathbf{m}') = \delta] = 1.$$

Proof Sketch.

1. We define a function V maps $\mathbf{m}, \mathbf{K}$ to $(v_1, \ldots, v_{2t})$.
2. Using linearity and m has more than $2t$ choices we find a differential pair of V with probability 1.
3. The same pair leads differential pair for H (possibly with different difference).

Answer to Question 1-d.

Theorem

Let $t < \ell/2 + r$, $r \leq d$. Let C compute a vector of d polynomials $H = (H_1, \ldots, H_d)$ with t multiplications then

$$\exists \mathbf{m} \neq \mathbf{m}' \in \mathbb{F}_{2^n}^{\ell}, \delta \in \mathbb{F}_{2^n}, Pr[H_{\mathbf{K}}(\mathbf{m}) \oplus H_{\mathbf{K}}(\mathbf{m}') = \delta] \geq 2^{-nr}.$$

1. If $r = d - 1$ (or $t = \ell/2 + d - 2$), we say that we only get differential probability about $2^{-n(d-1)}$ instead of 2^{-nd}.
2. $r = d \Rightarrow t \geq d - 1 + \ell/2$ is the minimum number of mult (in $\mathbb{F}_{2^n}$) to get about 2^{-nd}-ΔU hash which outputs $\mathbb{F}_{2^n}^{d}$.

Proof of Theorem 1-d.

1. Can apply previous idea to find a differential pair for the first $v_1, \ldots, v_{t-r}$ (as $2(t-r) < \ell$).
2. For remaining v_i's (r such, i.e., $v_{t-r+1}, \ldots, v_t$) we claim that there must exist a difference with probability at least 2^{-nr} (the best difference, existential).
3. This will eventually leads to differential pair for H with same probability.

Answer to the Final Question.

Encode-then-Hash-then-Combine:

1. **error correcting code**: $e : D \to A^\ell$ with the minimum distance d.

   MDS with systematic form such as $[I : V]$ where V is a Vandermonde Matrix.

2. **ΔU hash**: $h_K : A \to \mathbb{F}_{2^n}$ be an ϵ-ΔU.

   $A = \mathbb{F}_{2^n}^2$ and $(m_1, m_2) \mapsto (m_1 + K_1)(m_2 + K_2)$.

3. **Combiner**: Let V be a matrix of dimension $d \times \ell$ whose entries are from $\mathbb{F}_{2^n}$ such that any d **columns are linearly independent**.

   Vandermonde Matrix, again.

Encode-then-Hash-then-Combine or EHC.

Input: $M \in D$.
Output: $(H_1, \ldots, H_d) \in \mathbb{F}_{2^n}^d$.

1. $e(M) = (m_1, \ldots, m_\ell) \in A^\ell$.
2. $h_i = h_{K_i}(m_i)$ for ℓ independent keys K_i's, $1 \leq i \leq \ell$.
3. $(H_1, \ldots, H_d) = (h_1, \ldots, h_\ell) \cdot V$, i.e.

$$\begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ \alpha^{\ell-1} & \alpha^{\ell-2} & \cdots & \alpha & 1 \\ \vdots & \vdots & \vdots & \vdots & \\ \alpha^{(\ell-1)(d-1)} & \alpha^{(\ell-2)(d-1)} & \cdots & \alpha^{d-1} & 1 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_\ell \end{pmatrix} = \begin{pmatrix} H_1 \\ H_2 \\ \vdots \\ H_d \end{pmatrix}$$

Differential property of EHC.

- If $M \neq M'$, then $(m_1, \ldots, m_\ell)$ and $(m_1, \ldots, m_\ell)$ differ at least in d positions (for simplicity assume the first d positions).
- Conditions all keys $K_{d+1}, \ldots, K_\ell$.
- The differential event implies that $(\Delta h_{K_1}(m_1), \ldots, \Delta h_{K_d}(m_d)) \cdot V' = \delta'$ where V' is the first d columns of V and non-singular.
- Thus differential probability is at most ϵ^d.

Specific Choices of EHC for $d = 2$, $\ell + 2 = 2\ell'$.

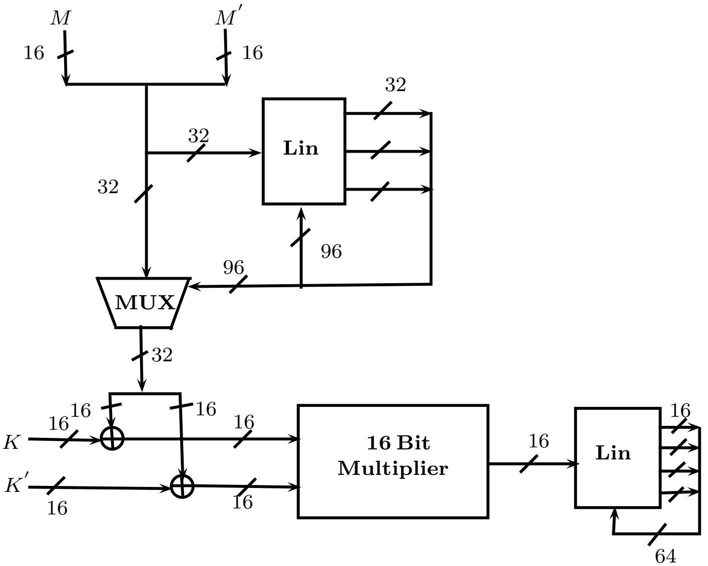
1. $M = (x_1, \ldots, x_{\ell'}) \in \mathbb{F}_{2^{2n}}^{\ell'}$. We write $x_i = (m_{2i-1}, m_{2i}) \in \mathbb{F}_{2^n}^2$.
2. $x_{\ell'} = \oplus_i x_i = (m_{\ell'-1}, m_{\ell'})$.
3. $h_{K,K'}(m, m') = (m \oplus K) \cdot (m' \oplus K') \in \mathbb{F}_{2^n}$ (PDP).
4. V is Vandermonde matrix with entries from $\mathbb{F}_{2^n}$.

$$\begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ \alpha^{\ell-1} & \alpha^{\ell-2} & \cdots & \alpha & 1 \end{pmatrix}$$

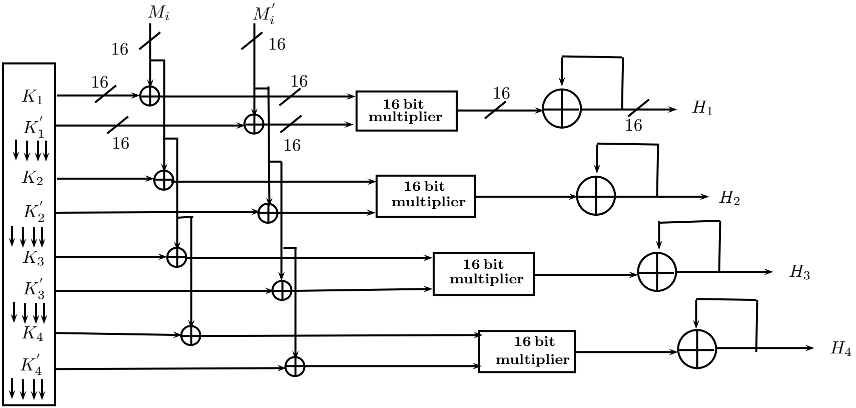
5. $H_1 = (m_1 \oplus K_1)(m_2 \oplus K_2) \oplus \cdots \oplus (m_{\ell-1} \oplus K_{\ell-1})(m_\ell \oplus K_\ell)$
6. $H_2 = \alpha^{\ell'-1}(m_1 \oplus K_1)(m_2 \oplus K_2) \oplus \cdots \oplus (m_{\ell-1} \oplus K_{\ell-1})(m_\ell \oplus K_\ell)$

Variable Length. Can be taken care by hashing length.

Specific Choices of EHC for $d = 4$.



Comparison with Toeplitz, $d = 4$ for PDP



Future Work and Conclusion.

1. Provide tight matching bounds on multiplications for ΔU hash functions, even for multi-block hash.
2. A practical construction (hardware friendly, less area). Actual hardware performace yet to observe.
3. Here we consider multiplication vs. message blocks. One can include error probability and study the relationship among these.

Thank You