Pipelineable On-Line Encryption (POE)

FSE 2014

Farzaneh Abed\textsuperscript{2} Scott Fluhrer\textsuperscript{1} John Foley\textsuperscript{1}
Christian Forler\textsuperscript{2} Eik List\textsuperscript{2} Stefan Lucks\textsuperscript{2} David McGrew\textsuperscript{1}
Jakob Wenzel\textsuperscript{2}

\textsuperscript{1} Cisco Systems, \textsuperscript{2} Bauhaus-Universität Weimar

March 3, 2014

London, UK
Section 1

Scenario
Case Study: Optical Transport Network (OTN)

Task:

- secure network traffic
Task:

- secure network traffic
- of real-time applications
Case Study: Optical Transport Network (OTN)

Task:

- *secure* network traffic
- of real-time applications
- in an Optical Transport Network (OTN)
Case Study: Optical Transport Network (OTN)

Task:

- secure network traffic
- of real-time applications
- in an Optical Transport Network (OTN)
  - High throughput (40 - 100 Gbit/s)
  - Low latency (few clock cycles)
  - Large message frames (64 KB)
    (usually consist of multiple TCP/IP or UDP/IP packages)
Task:
- *secure* network traffic
- of real-time applications
- in an Optical Transport Network (OTN)
  - High throughput (40 - 100 Gbit/s)
  - Low latency (few clock cycles)
  - Large message frames (64 KB)
    (usually consist of multiple TCP/IP or UDP/IP packages)

Security requirements:
- Data privacy (IND-CPA), and
- Data integrity (INT-CTXT)
Scenario

Case Study: Optical Transport Network (OTN)

Task:

- secure network traffic
- of real-time applications
- in an Optical Transport Network (OTN)
  - High throughput (40 - 100 Gbit/s)
  - Low latency (few clock cycles)
  - Large message frames (64 KB)
    (usually consist of multiple TCP/IP or UDP/IP packages)

Security requirements:

- Data privacy (IND-CPA), and
- Data integrity (INT-CTXT)

Functional requirements:

- On-line encryption/decryption
Problem and Workarounds

Problem: High Latency of Authenticated Decryption

1. Decryption of the entire message
2. Verification of the authentication tag

For 64-kB frames we have 4,096 ciphertext blocks (128 bits)

Workarounds:
- Decrypt-then-mask? [Fouque et al. 03] \(\Rightarrow\) latency again
- Pass plaintext beforehand and hope...
Scenario

Problem and Workarounds

Problem: High Latency of Authenticated Decryption

1. Decryption of the entire message
2. Verification of the authentication tag

For 64-kB frames we have 4,096 ciphertext blocks (128 bits)

- Workarounds:
  - Decrypt-then-mask? [Fouque et al. 03] ⇒ latency again
  - Pass plaintext beforehand and hope. . .

- Drawbacks:
  - Plaintext information would leak if authentication tag invalid
  - Literature calls this setting decryption-misuse [Fleischmann, Forler, and Lucks 12]
How Severe is Decryption-Misuse?

- Puts security at high risk
- CCA-adversary may inject controlled manipulations
- Particularly, CTR-mode based encryption schemes
How Severe is Decryption-Misuse?

- Puts security at high risk
- CCA-adversary may inject controlled manipulations
- Particularly, CTR-mode based encryption schemes

Decryption-misuse is *not* covered by existing CCA3-security proofs
Decryption Misuse Resistance

- Best to wish for:
  - Manipulation of ciphertext block $C_i$
    $\Rightarrow$ completely random plaintext
  - Contradiction to on-line requirement
Scenario

Decryption Misuse Resistance

- Best to wish for:
  - Manipulation of ciphertext block $C_i$  
    $\Rightarrow$ completely random plaintext
  - Contradiction to on-line requirement

- What can we achieve with an on-line encryption scheme?
  - Manipulation of $C_i$  $\Rightarrow$ random $(M_i, M_{i+1}, \ldots)$
  - Adversary sees at best common message prefixes
Decryption Misuse Resistance

Best to wish for:
- Manipulation of ciphertext block $C_i$ ⇒ completely random plaintext
- Contradiction to on-line requirement

What can we achieve with an on-line encryption scheme?
- Manipulation of $C_i$ ⇒ random $(M_i, M_{i+1}, \ldots)$
- Adversary sees at best common message prefixes

The security notion of OPERM-CCA covers this behaviour
Definition (OPERM-CCA Advantage)

Let $P$ be a random on-line permutation, $\Pi = (K, E, D)$ an encryption scheme, and $A$ be an adversary. Then we have

$$\text{Adv}^\text{OPERM-CCA}_{\Pi}(A) = \left| \Pr \left[ k \xleftarrow{\$} K() : A^{E_k(\cdot), D_k(\cdot)} \right] - \left[ A^{P(\cdot), P^{-1}(\cdot)} \right] \right|$$
On-Line Permutation

On-Line Permutation (OPerm)

Like a PRP with the following property:
Plaintexts with common prefix $\rightarrow$ ciphertexts with common prefix

(Bellare et al.; “Online Ciphers and the Hash-CBC Construction”; CRYPTO’01)
Intermediate (Authentication) Tags

Assume an OPERM-CCA secure encryption scheme

- Recap: Modifying $C_i \Rightarrow M_i, M_{i+1}, \ldots, M_M$ random garbage
- Redundancy in the plaintext (e.g., checksum)
Intermediate (Authentication) Tags

Assume an OPERM-CCA secure encryption scheme

- Recap: Modifying $C_i \implies M_i, M_{i+1}, \ldots, M_M$ random garbage
- Redundancy in the plaintext (e.g., checksum) $\implies$ intermediate authentication tags
Intermediate (Authentication) Tags

Assume an OPERM-CCA secure encryption scheme

- Recap: Modifying $C_i \implies M_i, M_{i+1}, \ldots, M_M$ random garbage
- Redundancy in the plaintext (e.g., checksum) \implies intermediate authentication tags

- Common network packets (TCP/IP, UDP/IP) have a checksum \implies OTN: 16-bit integrity for free (per packet)
Promising Candidate: TC3

- TC3 [Rogaway & Zhang 11] is IND-CCA
- McOE [Fleischmann et al. 12] is based on TC3
Promising Candidate: TC3

- TC3 [Rogaway & Zhang 11] is IND-CCA
- McOE [Fleischmann et al. 12] is based on TC3
- Why not using TC3?
Promising Candidate: TC3

- TC3 [Rogaway & Zhang 11] is IND-CCA
- McOE [Fleischmann et al. 12] is based on TC3
- Why not using TC3?
  - Inherently sequential
## Comparison of Common On-line Encryption Schemes

<table>
<thead>
<tr>
<th></th>
<th>Sequential</th>
<th>Non-Sequential</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CCA-insecure</strong></td>
<td>ABC, CBC, CFB, HCBC1, IGE, OFB, TC1</td>
<td>COPE, CTR, ECB, TIE, XTS</td>
</tr>
<tr>
<td><strong>CCA-secure</strong></td>
<td>APE, CMC, HCBC2, MCBC, MHCBC, TC2/3</td>
<td></td>
</tr>
</tbody>
</table>

It seems that there is still some place for a new encryption scheme.
Comparison of Common On-line Encryption Schemes

<table>
<thead>
<tr>
<th>Sequential</th>
<th>Non-Sequential</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CCA-insecure</strong></td>
<td><strong>COPE, CTR, ECB, TIE, XTS</strong></td>
</tr>
<tr>
<td>ABC, CBC, CFB, HCBC1, IGE, OFB, TC1</td>
<td></td>
</tr>
<tr>
<td><strong>CCA-secure</strong></td>
<td></td>
</tr>
<tr>
<td>APE, CMC, HCBC2, MCBC, MHCBC, TC2/3</td>
<td></td>
</tr>
</tbody>
</table>

It seems that there is still some place for a new encryption scheme.
Section 2

POE/POET
Pipelineable On-Line Encryption (POE)

- Well pipelineable
- OPERM-CCA-secure
- $1 \text{ BC} + 2 \epsilon$-AXU hash-function ($F$) calls per block
Instantiations of the $\epsilon$-AXU Hash Function $F$

4-Round-AES

- $10 + 4 + 4 = 18$ AES rounds/block
- $\epsilon$-AXU with $\epsilon \approx 1.88 \cdot 2^{-114}$ [Daemen & Rijmen 98]
Instantiations of the $\epsilon$-AXU Hash Function $F$

4-Round-AES
- $10 + 4 + 4 = 18$ AES rounds/block
- $\epsilon$-AXU with $\epsilon \approx 1.88 \cdot 2^{-114}$ [Daemen & Rijmen 98]

GF($2^{128}$)-multiplication
- 1 BC call + 2 multiplications with $\epsilon \approx 2^{-128}$
- POE can be parallelized
  - Given $p^i = K^i + K^{i-1} \cdot M_1 + \ldots + K \cdot M_{i-1} + M_i$
  - Core 1: $K \cdot p^i + M_{i+1}$
  - Core 2: $K^2 \cdot p^i + K \cdot M_{i+1} + M_{i+2}$
  - ...
Instantiations of the $\epsilon$-AXU Hash Function $F$

4-Round-AES
- $10 + 4 + 4 = 18$ AES rounds/block
- $\epsilon$-AXU with $\epsilon \approx 1.88 \cdot 2^{-114}$ [Daemen & Rijmen 98]

$\text{GF}(2^{128})$-multiplication
- 1 BC call + 2 multiplications with $\epsilon \approx 2^{-128}$
- POE can be parallelized
  - Given $p^i = K^i + K^{i-1} \cdot M_1 + \ldots + K \cdot M_{i-1} + M_i$
  - Core 1: $K \cdot p^i + M_{i+1}$
  - Core 2: $K^2 \cdot p^i + K \cdot M_{i+1} + M_{i+2}$
  - $\ldots$
  - Increases number of multiplications
  - Decreases latency ($O(c) \rightarrow O(\log c)$)
Key Derivation

- 3 keys: $K$ for $E$ and $K_1, K_2$ for $F$ in the top and bottom row
- $K = E_L(0)$, $K_1 = E_L(1)$, $K_2 = E_L(2)$
POE with Tag (POET)

- Prepends $H$
- CCA3-secure
- Borrows tag-splitting procedure from McOE
- Robust against **nonce- and decryption-misuse**
Section 3

Security of POE/POET
For an adversary $A$, asking at most $q$ messages, consisting of at most $\ell$ total blocks, which runs in time at most $t$, it holds that

$$\text{Adv}_\Pi^{\text{OCCA3}}(A) \leq \text{Adv}_\Pi^{\text{OPERM-CCA}}(q, \ell, t) + \text{Adv}_\Pi^{\text{INT-CTXT}}(q, \ell, t).$$

---

**OCCA3**

Cisco Systems, Bauhaus-Universität Weimar

Pipelineable On-Line Encryption (POE)

FSE 2014
A instantly wins if a **bad event** occurs

1. $A$ can distinguish $E$ from random permutation
$\mathcal{A}$ instantly wins if a **bad event** occurs

1. $\mathcal{A}$ can distinguish $E$ from random permutation
2. Collision in top row
A instantly wins if a **bad event** occurs

1. $A$ can distinguish $E$ from random permutation
2. Collision in top row
3. Collision in bottom row
1. Assume $E$ is secure:

$$\text{Adv}^{\text{IND-SPRP}}_{E,E^{-1}}(\ell, O(t))$$
1. Assume $E$ is secure:

$$\text{Adv}_{E,E^{-1}}^{\text{IND-SPRP}}(\ell, O(t))$$

2. Collision in top row

$$\frac{\epsilon \cdot \ell(\ell - 1)}{2} \leq \frac{\epsilon \cdot \ell^2}{2}$$
1. Assume $E$ is secure:

$$\text{Adv}_{E,E^{-1}}^{\text{IND-SPRP}}(\ell, O(t))$$

2. Collision in top row

$$\epsilon \cdot \ell (\ell - 1) \leq \epsilon \cdot \ell^2$$

3. Collision in bottom row (see 2.)
If no **bad event occurs** we have

\[
\frac{\ell^2}{2^n - \ell}
\]

The total probability is given by the sum

**OPERM-CCA Advantage**

\[
\text{Adv}^\text{OPERM-CCA}_{\text{POET}}(q, \ell, t) \leq \epsilon \ell^2 + \frac{\ell^2}{2^n - \ell} + \text{Adv}^\text{IND-SPRP}_{E,E^{-1}}(\ell, O(t))
\]
# Filling the Gap

<table>
<thead>
<tr>
<th>Sequential</th>
<th>Non-Sequential</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CCA-insecure</strong></td>
<td><strong>CCA-secure</strong></td>
</tr>
<tr>
<td>ABC, CBC, CFB, HCBC1, IGE, OFB, TC1</td>
<td>COPE, CTR, ECB, TIE, XTS</td>
</tr>
<tr>
<td>APE, CMC, HCBC2, MCBC, MHCBC, TC2/3</td>
<td><strong>POE</strong></td>
</tr>
</tbody>
</table>
POET: INT-CTXT-Security

- INT-CTXT proof is game-based
- Combines the ideas from its OPERM-CCA proof and the INT-CTXT proof from McOE
- Details (→ Paper)
\[
\text{Adv}^{\text{INT-CTXT}}_{\text{POET}}(q, \ell, t) \leq (\ell + 2q)^2 \epsilon + \frac{q}{2^n - (\ell + 2q)} + \text{Adv}^{\text{IND-SPRP}}_{E, E^{-1}}(\ell + 2q, O(t))
\]
POE: Non-sequential on-line cipher
  - Simple design
  - Support for intermediate tags
  - Provably OPERM-CCA-secure
  - High throughput: non-sequential, on-line
  - Robust against nonce- and decryption-misuse
Conclusion

- **POE**: Non-sequential on-line cipher
  - Simple design
  - Support for intermediate tags
  - Provably OPERM-CCA-secure
  - High throughput: non-sequential, on-line
  - Robust against nonce- and decryption-misuse

- **POET**: On-line AE built on POE
  - Security: Provably OCCA3-secure
  - Fulfills the demanding requirements of high-speed networks
Thank you

Questions?
OPERM-CCA Attack Against COPE (1)

\[ Y_a = E_K(M_a \oplus 3L) \oplus L \] and \[ Y_b = E_K(M_b \oplus 3L) \oplus L \]

- Query: \((M_a, M_c)\); Result: \((C_a, C_{(a,c)})\)
- Query: \((M_b, M_c)\); Result: \((C_b, C_{(b,c)})\)
OPERM-CCA Attack Against COPE (2)

\[ Y_a = E_K(M_a \oplus 3L) \oplus L \] and \[ Y_b = E_K(M_b \oplus 3L) \oplus L \]

- Query: \((C_a, C_{(b,c)})\); Result: \((M_a, M_{(a,bc)})\)
- Query: \((C_b, C_{(a,c)})\); Result \((M_b, M_{(b,ac)})\)

\[ Y_{(a,c)} = E_K^{-1}(C_{(a,c)} \oplus 4L), \quad X_{(b,ac)} = Y_{(a,c)} \oplus Y_b = X_{(a,bc)} \]

\[ \implies M_{(a,bc)} = M_{(b,ac)} \]