Rugged Pseudorandom Permutations and Their Applications

Jean Paul Degabriele and Vukašin Karadžić
Outline

• The Security Definition

• Transforming Rugged PRPs into AEAD

• Nonce-Set AEAD and Order-Resilient Channels

• Application to Onion Encryption in Tor

• RPRP Constructions
The Security Definition
The Security of Variable-Length Tweakable Ciphers

PRP

E vs Π

RPRP

E, f(Π) vs Π, f(Π⁻¹)

SPRP

E, D vs Π, Π⁻¹

Is there something useful in between?

- Not very useful as most applications require deciphering
- Intuitively f( ) limits access to D and Π⁻¹
- Goal: Notion permitting more efficient constructions that have practical applications.
- Strong security, but
- Heavy Constructions
- Or require stronger assumptions (AEZ)
- Sometimes Overkill
The notion of a Rugged PRP requires a slightly more stringent syntax.

Namely the (VIL) tweakable cipher must operate over a split domain, such as \(\{0,1\}^n \times \{0,1\}^*\), where \(n\) is in the range 128-256 bits.
Rugged Pseudorandom Permutations

Deciphering can be accessed via two separate oracles:

- **De** - restricted queries, full output.
- **Gu** – unrestricted queries, 1-bit output.
Rugged Pseudorandom Permutations

• Replace $E_K$ with an ideal cipher $\Pi$.

• $Gu$ always returns false.

• For satisfiability $Gu$ queries must not be trivial to guess.
Transforming RPRPs into AEAD
The EtE Transform

- We revisit and adapt the **Encode-then-Encipher paradigm** [BelRog00, ShrTer13] in the context of RPRPs.

- EtE is slightly more general, the above is a specific instantiation of it.

- \((E_K, D_K)\) is RPRP secure \(\Rightarrow\) EtE is **Misuse-Resistant AEAD**.
The EtD Transform

- \((E_K, D_K)\) is RPRP secure \(\Rightarrow\) EtD yields a **RUPAE nonce-hiding AEAD**.

- However we can instantiate it differently to reduce the ciphertext expansion by using the **nonce to authenticate** the ciphertext.

- \((E_K, D_K)\) is RPRP secure \(\Rightarrow\) EtD is a (standard) **AEAD** that is **RUPAE** secure.
Nonce-Set AEAD and Order-Resilient Channels
The AwN Transform

- We can also use the **nonce to authenticate** in the EtE transform and obtain a nonce-hiding AEAD.

- We can generalize this further by **testing the nonce for set membership** instead of equality, yielding the **AwN** transform.

- **AwN** transforms an RPRP into a **Nonce-Set AEAD** that is **Misuse-Resistant**.
Why Nonce-Set AEAD?

• Nonce-Set AEAD serves as a **stepping stone** for realizing **order-resilient channels** such as **QUIC** and **DTLS**.

• Several possibilities arise for handling **reorderings**, **replays**, **modifications**, and **deletions**, and how much of each to tolerate.

• Typical constructions employ one or more **window mechanisms**, which add complexity—making them **hard to understand and analyze**.

• In general, it is unclear how these **additional mechanisms** interact with AEAD and what the **overall security** of the channel is.
The Support Predicate

• The various functionalities of such channels can be formally characterized by a support predicate:

\[
\text{accept/reject} \leftarrow \text{supp}(C, C_S, D_{C_R})
\]

• It was developed in [Bac19, FGJ20] as a generalization of the silencing approach by [RogZha18].

• The support predicate permeates into all aspects of the secure channel correctness, security, and robustness [FGJ20].
Order-Resilient Channels from NS-AEAD

- We present a **universal** and **generic** channel construction from Nonce-Set AEAD for any desired support predicate!

- The construction consists of a **Nonce-Set AEAD (blue)** scheme and a tuple of **Nonce-Set Processing (NSP) scheme (red)**.

<table>
<thead>
<tr>
<th>Init()</th>
<th>Send(stk_s, A, M)</th>
<th>Recv(stk_r, A, C)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$(st_s, st_r) \leftarrow $\text{StInit()}$</td>
<td>$(st_s, K) \leftarrow stk_s$</td>
</tr>
<tr>
<td></td>
<td>$K \leftarrow ${0, 1}^k$</td>
<td>$(st'_s, N) \leftarrow \text{NonceExtract(st_s)}$</td>
</tr>
<tr>
<td></td>
<td>stk_s $\leftarrow (st_s, K)$</td>
<td>if $N = \bot$ then</td>
</tr>
<tr>
<td></td>
<td>stk_r $\leftarrow (st_r, K)$</td>
<td>return $(st'_s, \bot)$</td>
</tr>
<tr>
<td>return $(stk_s, stk_r)$</td>
<td>C $\leftarrow \text{Enc}(K, N, A, M)$</td>
<td>else</td>
</tr>
<tr>
<td></td>
<td>stk'_s $\leftarrow (st'_s, K)$</td>
<td>stk'_r $\leftarrow (st'_r, K)$</td>
</tr>
</tbody>
</table>
Order-Resilient Channels from NS-AEAD

• We prove this channel construction correct, robust, and secure.

• We only require that the Nonce-Set AEAD is secure and that the NSP scheme satisfy a functionality property called faithfulness.

• Informally, faithfulness says that the NSP scheme accurately reproduces the support predicate logic over the nonces.

• One can simply tune the NSP to the desired functionality and plug in their favourite Nonce-Set AEAD and security/robustness will be automatic.
Application to Onion Encryption in Tor
Onion Encryption from RPRPs

• Tor is susceptible to **tagging attacks** which undermine privacy by exploiting the **malleability** in its encryption layers.

• To address this, it has been proposed to replace each layer with a **wide-block Strong PRP**, but a **Rugged PRP** turns out to be sufficient.

• With other co-authors we have a proposal for an RPRP-based onion encryption scheme which adds **forward security**, protects against **tagging attacks**, and provides **competitive performance**.
Onion Encryption in Tor with an RPRP

Forward Direction

\[
\begin{align*}
&\text{M} \\
&\downarrow \text{D}_K \\
&\text{OPD}(\text{enc}) \\
&\downarrow \text{D}_K \\
&\text{M} \\
&\uparrow \text{C}_3 \\
&\text{OR}(\text{dec}) \\
&\text{OR}(\text{dec}) \\
&\text{OR}(\text{dec}) \\
&\text{OR}(\text{dec}) \\
&\text{M}
\end{align*}
\]

Backward Direction

\[
\begin{align*}
&\text{M} \\
&\downarrow \text{D}_K \\
&\text{OPD}(\text{dec}) \\
&\downarrow \text{D}_K \\
&\text{M} \\
&\uparrow \text{C}_3 \\
&\text{OR}(\text{enc}) \\
&\text{OR}(\text{enc}) \\
&\text{OR}(\text{enc}) \\
&\text{OR}(\text{enc}) \\
&\text{M}
\end{align*}
\]
RPRP Constructions
Unilaterally-Protected IV (UIV)

• UIV is obtained from PIV [ShrTer13] by dropping the third layer and is RPRP secure.

• It can be instantiated with GCM components leading to a performance similar to GCM-SIV.

• It is closely related to GCM-RUP [ADL17] and MiniCTR [Min15].
Hash-Encipher-Counter (HEC)

- HEC is inspired by HCTR where the second AXU Hash is replaced with a lighter XOR operation.
- HEC is an RPRP but not an SPRP.
- When instantiated with GCM components HEC requires less key material than UIV.
Unbalanced Three-Round Feistel

• Security proof typically require either access to Guess or Decipher but not both simultaneously.

• As such it makes sense to consider the notions: \textbf{RPRPd} (Enc+Dec) and \textbf{RPRPg} (Enc+Gue).

• Then the \textbf{Expand-Compress-Expand (ECE)} construction shown on the left is \textbf{RPRd} secure.

• The analogous \textbf{Compress-Expand-Compress (CEC)} construction is \textbf{RPRPg} secure.
Concluding Remarks
Summary

• Rugged PRPs strike a **new tradeoff** between **security** and **performance**.

• An RPRP is a rather **versatile primitive** to have in a crypto library as it can easily be turned into **MRAE (n/nh)**, **RUPAE (n/nh)**, **Nonce-Set AEAD/ Order-Resilient Channels**, **Onion Encryption**.

• We are currently working on RPRP constructions with **BBB security**.

• Intrinsically a variable-length cipher is **not key-committing**. Identifying efficient ways to add this property to our constructions is an interesting open problem.