Modifying an Enciphering Scheme after Deployment

Paul Grubbs, Thomas Ristenpart, Yuval Yarom
Format-Preserving Encryption (FPE)
Format-Preserving Encryption (FPE)
Format-Preserving Encryption (FPE)

This work: Backwards-compatible FPE

Academic and industry work on FPE:
- **Tokenization**
- Cycle walking [BR]
- FE1, FE2 constructions [BRRS]
- Thorp shuffle [MRS]
- NIST standard FFX
- Support for arbitrary formats [DCRS,LDJRS,LSRJ]
Format-Preserving Encryption (FPE)

This work: Backwards-compatible FPE

Encryption service

New encryption service

• Includes new features
• Decrypts old ciphertexts properly
• Not just key rotation, FPE scheme changes
Example: Upgrading from tokenization

Tokenization: implement FPE using look-up table of random ciphertexts (encryption key is the table)
Example: Upgrading from tokenization

Tokenization: implement FPE using look-up table of random ciphertexts (encryption key is the table)

Does not scale well! Practitioners want to use modern FPE instead (e.g., FFX)

Frequent problem in practice:
Old ciphertexts can’t be retrieved & re-encrypted
Example: Upgrading from tokenization

Tokenization: implement FPE using look-up table of random ciphertexts (encryption key is the table)
Does not scale well! Practitioners want to use modern FPE instead (e.g., FFX)

Need a backwards-compatible FPE:
- New plaintexts encrypted with compact key
- Old ciphertexts decrypted using tokenization
- Preserve permutivity

Frequent problem in practice:
Old ciphertexts can’t be retrieved & re-encrypted
Example: Expanding format

Problem: deployed with one format in mind (e.g., just 16 digit CCN’s) but need to support others as well (e.g., also 15 digit CCN’s)
Example: Expanding format

Problem: deployed with one format in mind (e.g., just 16 digit CCN’s) but need to support others as well (e.g., also 15 digit CCN’s)

Need a backwards-compatible FPE:
- New plaintexts (15 or 16 digit CCNs) encrypted
- Old 16-digit ciphertexts can be decrypted
- Preserve permutivity

Frequent problem in practice:
Old ciphertexts can’t be retrieved & re-encrypted
Our contributions

Give generic algorithm (Zig-Zag) for backwards-compatible FPE

Domain completion (tokenization upgrade example)
- Prove “natural” security
- Analyze runtime

Domain extension (expanding format example)
- “Natural” security is impossible
- Give new security goal, analyze
Domain completion (formally)

Need a backwards-compatible FPE:
- New plaintexts encrypted with compact key
- Old ciphertexts decrypted using tokenization
- Preserve permutivity

An FPE scheme $\text{FPE}_k : D \rightarrow D$ with key $K$ is a permutation of $D$ for every $K$

Call old FPE (partial permutation) $F_{k^*} : D \rightarrow D$ and $T = \text{Dom}(F_{k^*})$.

Need new FPE $\text{ZZ}_{k'} : D \rightarrow D$ so that

$$\forall t \in T, \text{ZZ}_{k'}(t) = F_{k^*}(t)$$

Security goal is

**Strong Pseudorandom Permutation:**
indistinguishable from random permutation **even if adversary knows** $T$
The obvious approach doesn’t work

What about simply using a tokenization scheme and a new FPE in parallel?

- table $\text{Tok}[\ ]$ ($F_{k^*}$)
- $\text{FFX E}$ with secret key $K$

Encrypt($\text{Tok}[, \text{K}]$, M):

If M in T then:
  Return $\text{Tok}[M]$
Else:
  Return $E_K(M)$

This doesn’t define a permutation for every (T,K)!
The Zig-Zag Construction

Uses a form of cycle walking to "repair" permutation on colliding points

= table Tok[ ] ( F_k* )

= FFX E with secret key K

Encrypt( (Tok[], K) , M):

If M in T then:
   Return Tok[M]

Else:
   C = E_k(M)
   while (Tok^{-1}[C] != null):

   Return C
The Zig-Zag Construction

Uses a form of cycle walking to "repair" permutation on colliding points

= table Tok[] (F_k*)

= FFX E with secret key K

Encrypt((Tok[], K), M):
If M in T then:
   Return Tok[M]
Else:
   C = E_K(M)
   while (Tok^{-1}[C] != null):
      M' = Tok^{-1}[C]
      C = E_K(M')
   Return C
Zig-Zag analysis

Theorem (informal): If $|T| \leq |D|/2$, the Zig-Zag algorithm runs in amortized constant time, except with negligible probability.

Key intuition: With random permutations, can use hypergeometric tail bound to upper-bound drawing many collisions in a row.

Theorem (informal): The Zig-Zag algorithm is as secure as the underlying permutations (E) even if the adversary knows T.
Domain extension (formally)

Need a backwards-compatible FPE:
• New plaintexts (15 or 16 digit CCNs) encrypted
• Old 16-digit ciphertexts can be decrypted
• Preserve permutivity

Call old FPE (partial permutation)
$F_{k^*} : D \rightarrow D$, $T = \text{Dom}(F_{k^*})$,
and new domain $M$ ($D \subseteq M$).
Need FPE $ZZ_{k'} : M \rightarrow M$ so that
$\forall t \in T, ZZ_{k'}(t) = F_{k^*}(t)$
Zig-Zag works for domain extension

= Old secret key $K^*$ for $F_{k^*} : D \to D$

= FFX secret key $K$ for $E_k : M \to M$

Encrypt( ($K^*$, $K$), $M$):
If $M$ in $T$ then:
    Return $F_{k^*}(M)$
Else:
    $C = E_k(M)$
while ($F_{k^*}^{-1}(C) \in T$):
    $C = E_k(F_{k^*}^{-1}(C))$
return $C$
Zig-Zag works for domain extension

What security does this achieve?

Encrypt((K*, K), M):
If M in T then:
  Return $F_{k^*}(M)$
Else:
  $C = E_K(M)$
  while $(F_{k^*}^{-1}(C) \in T)$:
    $C = E_K(F_{k^*}^{-1}(C))$
  return C

$F_{k^*}(T)$

$M$ $M$

$T$ $D$

$D$
SPRP security is impossible

When adversary knows $T = \{t_1 \ldots t_{|T|}\}$, there is a trivial distinguisher for any DE cipher

for $i$ in $[1 \ldots q]$:
  if $ZZ_{k'}(t_i) \notin D$:
    return “ideal”
  return “real”

Key intuition: Unlikely for random permutation that all queries fall in $D$.

Can we prove any meaningful security?

$$\text{Advantage} = 1 - \frac{|D|!(|M|-q)!}{|M|!(|D|-q)!}$$
Can we achieve any meaningful security?

Weaken SPRP security notion, target indistinguishability from different ideal object

“Strong extended pseudorandom permutation”
SEPRP security

A permutation is an SEPRP if indistinguishable from permutation sampled uniformly subject to $\forall t \in T, ZZ_{k'}(t) = F_{k*}(t)$

**Theorem (informal):** Zig-Zag is an SEPRP.

**Theorem (informal):** SEPRP gives at most a factor-of-2 speedup in message recovery game from [BRRS].

Key intuition: Generalize message recovery notion from [BRRS]. One hidden bit (membership in $T$), so 2x queries.
Other considerations

- If adversary only knows $|T|$, modified Zig-Zag can meet SPRP (see paper)
- Variable timing for some inputs
  + Timing side channel only leaks membership in $T$
- Rank-encipher-unrank construction
  + Fast in worst case
    - High storage overhead, cache side channels
Summary

Introduce backwards-compatible crypto

We give generic algorithm (Zig-Zag) for backwards-compatible FPE

Achieved *domain completion* and *domain extension* for FPE using the Zig-Zag algorithm. Our techniques are efficient, provably secure, and solve real problems for practitioners

Thanks for listening! Any questions?