Feistel Finite Set Encryption Mode

Terence Spies
Voltage Security, Inc.
`terence@voltage.com`

Abstract

In many applications, such as encryption of credit card numbers, it is desirable to encrypt items from an arbitrarily sized set onto that same set. Unfortunately, conventional cipher modes such as ECB, CBC, or CTR are unsuitable for this purpose. Feistel Finite Set Encryption Mode (FFSEM) allows encryption of a value ranging from 0..n with resultant ciphertext in that same range. This mode can be used to encrypt fields where the expansion associated with a block cipher is undesirable or the format of the data must be preserved.

1 Introduction

Conventional block ciphers are fundamentally algorithmic permutations from a specific range (typically 2^{64} or 2^{128}) onto values in that same range. In some applications, it is valuable to be able to encrypt values in a smaller range in the same way. For example, many financial or e-commerce databases contain credit card numbers or social security numbers. For both practical and regulatory reasons, it is important to encrypt these values, however encrypting the entire database is impractical. A better approach is to encrypt only the sensitive values, but in many cases, the fields to be encrypted are fixed format and naively encrypting (for instance, with AES-ECB) produces ciphertexts which violate the format constraints. An encryption algorithm which works for these applications must be a permutation from valid values onto other valid values.

In 2002, Black and Rogaway [2] described a practical way of building such a construction. Their *Cycle Following* scheme uses a block cipher to produce a secure permutation for sets of size less than but near to 2^{2m} where $2m$ is the size of the cipher block. Unfortunately, cycle following becomes progressively more expensive as the set gets smaller and thus is not practical for use with AES in cases such as credit cards where there are approximately 2^{40} possible values. The effect is that cycle following is only practical if there is a block cipher with a block size of approximately the same size as the set to be encrypted, which is generally not the case.

The Luby-Rackoff construction [7] (LR) can be used to produce a block cipher with any even bit width $2m$. Black and Rogaway describe *Generalized*

Feistel Mode which combines LR with cycle following to allow the the encryption of sets of any size. Black and Rogaway show a reduction to the base block cipher when the attacker has less than $2^{m/2}$ plaintext/ciphertext pairs. This reduction is sufficient for the encryption of large sets, but provides an insufficient level of confidence for some cases, such as credit card numbers (CCNs) where $2^{2m} \approx 10^{16}$.

In 2004, Patarin [4] proved that a straightforward extension of Black and Rogaway's method (running six or more rounds instead of the three rounds used originally) reduces to the underlying block cipher for a computationally unbounded attacker with less than 2^m ciphertext/plaintext pairs. This allows use of this construction for a much wider range of sets. In particular, it allows encryption of sets around the size of the CCN space, in addition to enabling a number of other applications.

This paper specifies *Feistel Finite Set Encryption Mode* (FFSEM), a concrete instantiation of the Black and Rogaway method with an increased round count as suggested by Patarin, using AES as the underlying cipher. This mode is used to encrypt items smaller than the block size of the underlying cipher to ciphertext of the same size. We also describe appropriate parameter choices for implementation of the mode.

2 Overview of FFSEM

FFSEM consists of two basic components:

Cycle Following used to encrypt sets of approximately the same size as a given cipher's block size.

Feistel Method used to produce a block cipher of approximately the right size.

When used together, these components allow the construction of efficient pseudo-random permutations over sufficiently large arbitrary sets.

2.1 Cycle Following

Cycle following is a general method for using an q-bit block cipher to encrypt and decrypt sets of size n where $n < 2^q$. We treat the cipher block as an q-bit integer with the first n values mapped directly onto our set $0..n-1$ and and the remaining $2^q - n$ values marked invalid.

In order to encrypt a value i in $0..n-1$, we simply construct the corresponding cipher block, padding with zeroes as appropriate. If the resulting ciphertext is valid (in the range $0..n-1$) we output that as our result. Otherwise, we encrypt again using the block cipher, as shown in Figure 1. Black and Rogaway show that this repetition, which they call cycle-following, will always terminate and does not degrade the security of the underlying cipher.

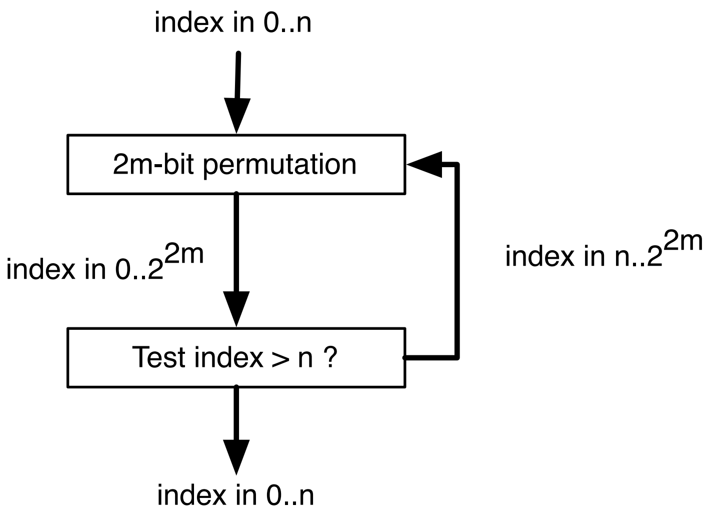

Figure 1: Overall cycle structure

Decryption follows the same procedure, starting with a ciphertext which (by construction) is in the range $0..n-1$ and iterating the underlying block cipher until a valid value is returned.

Because the underlying block cipher is a PRP over the space 2^q, the probability that any given cycle will produce a valid value is $\frac{n}{2^q}$. If $n << 2^q$, the cycle following algorithm will have very poor average behavior. It is therefore important to start with a block cipher of approximately the right size. Although standard block ciphers only come in a few inconvenient sizes, we can use the-Luby Rackoff construction to produce such a block cipher.

2.2 Feistel Mode

As described by Black and Rogaway, we can use a Luby-Rackoff construction to turn a standard, fixed-width block cipher into a block cipher of arbitrary width using the block cipher as the basis for the round-specific PRF.

2.2.1 Luby-Rackoff

Luby and Rackoff [7] show how to construct a block cipher using a specified PRF in a repeated Feistel network, as shown in Figure 2.

The Feistel round structure, when repeated some number of times, yields a Pseudo-Random Permutation. A single round simply divides the input bitvector into a right half and a left half, runs the right half through the PRF, XORs it with the left half, then swaps the right with the left.

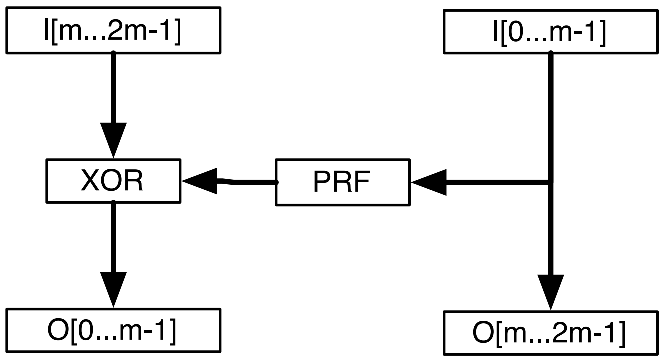

Figure 2: A single Feistel round

2.2.2 Forming the PRF

The L-R construction simply assumes the existence of a PRF, but any concrete instantiation requires actually instantiating one. The L-R PRF operates on half of the block size and so we want a PRF of width m bits for a cipher $2m$ bits wide. The natural PRF to use here is simply the base block cipher. Hall et al. [6] show that bitwise truncation of a PRP yields a PRF as long as the resultant PRF is significantly smaller than the PRP. So we can simply use a truncated version of the base block cipher as our PRF provided that the block size of the resultant L-R cipher is smaller than the base cipher, since then the PRF is less than half the width of the base cipher.

It is desirable to use a different PRF at each round of the L-R construction. A "tweakable" PRF [3] will give a different PRF at each round if the round count is used as the tweak. We accomplish this tweak by incorporating the round number into the input of the block cipher. This tweaking method is also used in the OMAC procedure used in EAX mode [5]. Bellare and Impagliazzo [8] and Lucks [1] show a construction that uses two invocations of the block cipher to derive a PRF that would be suitable if the needed PRF was closer to the size of the underlying block cipher. Since we only need a PRF of maximum one half the block size, the truncation construction suffices.

2.2.3 Security of this Construction

The Luby-Rackoff construction was first published in [7], with security bounds established for the three and four round versions. The security bounds in [7] and subsequent papers show a reduction to the strength of the underlying PRF by showing that an attacker with unbounded computational ability cannot distinguish the L-R permutation from a random permutation with less than some number of plaintext/ciphertext pairs, assuming that a fully random function is used for the PRF. Any attack more efficent than this would then need to exploit properties of the underlying PRF, not the overall L-R structure.

The best bounds for the L-R construction at the time of the Black-Rogaway paper were only for three and four rounds. They established that an $2m$ bit L-R cipher using a random function in place of the PRF could be distinguished from a random permutation after $2^{m/2}$ queries. These were hard limits, as there are demonstrated practical attacks against the construction at these limits. With these limits, the L-R construction has questionable security for sets significantly smaller than the AES blocksize. In their paper, Black and Rogaway conjectured that extending the construction to more rounds would improve security, but these bounds were unknown at that time.

In 2005, Patarin [4] showed that increasing the round count to 5 against CPA-2 attacks, and 6 against CPCA-2 attacks, increases the required queries to 2^m. Significantly, the practical attack against the 3 and 4 round versions no longer works. The security bounds shown are for a computationally unbounded attacker capable of guessing the inputs and outputs of the internal PRF. Patarin shows the best known realizable attack against these constructions requires more plaintext/ciphertext pairs than are generated by a single permutation.

3 FFSEM Encryption and Decryption

Using the building blocks described in the previous section, we can now provide a concrete description of FFSEM using AES as our base block cipher.

To specify FFSEM, we define two subfunctions, FFSEM-PRF, which is a Pseudo-Random Function based on some block cipher, and FFSEM-ROUND, which is an individual Feistel round. The FFSEM-ENCRYPT and FFSEM-DECRYPT functions are defined to take a maximum value n, a value i that is between 0 and n-1, and a round count.

3.1 FFSEM PRF Specification

An FFSEM encryption of 2m bits requires a PRF over m bits. Since FFSEM is limited to encryptions of less than block size of the underlying cipher, we will always need a PRF that is less than half the size of the underlying cipher. Bellare and Impagliazzo show [8] a more general method of PRF generation using the sum of two PRPs, but this is not needed for the size of PRF we require.

A tweak is used to insure that a different PRF is used at each round. We introduce the tweak as the upper byte of the input to the block cipher. For a k-bit wide PRF using an m-bit wide block cipher, the input to the block cipher then looks like the following:

$0..k-1$	$k..m-9$	$m-8..m-1$
input	0	round count

The output of the blockcipher is then trimmed to k bits.

```
Function FFSEM-PRF(
  bitvector a,
  integer length,
  key k,
  integer tweak)

  bitvector b[0...block_length]

  b[0..length(a)-1] = a[0..length(a)-1]
  b[length(a)..length(b)-8] = 0
  b[length(b)-8...length(b)-1] = byte_representation(tweak)

  return E(k, b) & 2^(length+1)-1;
```

3.2 FFSEM Round

The FFSEM-ROUND function describes a single Feistel round, as shown in Figure 1. The Feistel round structure, when repeated some number of times, yields a Pseudo-Random Permutation. Here the FFSEM-PRF function is used as the required PRF. THe FFSEM-ROUND function simply divides the input bitvector into a right half and a left half, runs the right half through the FFSEM-PRF function, XORs it with the left half, then swaps the right with the left.

```
Function FFSEM-ROUND(
  bitvector b,
  integer length,
  key k,
  integer tweak)

  right = b[0..length/2-1]
  left = b[length/2..length]

  return right << length/2 | left
            XOR FFSEM-PRF(right, length/2, k, tweak)
```

3.3 FFSEM Encryption

FFSEM-ENCRYPT uses FFSEM-ROUND and FFSEM-PRF to encrypt a value in the range 0..n onto the same range. The function takes the following inputs:

- A key for the underlying block cipher
- The maximum value of the range of plaintexts
- A round count

The minimum value of the round count is six, which provides security bounds shown by Patarin. The security considerations section contains more details on the relationship between round count and values of n.

The initial step in FFSEM-ENCRYPT and FFSEM-DECRYPT is to find the smallest bitvector of even length that will take an encoding of n-1. The input value is then encoded into a bitvector of that length, and the cycling Feistel structure is used to encrypt that bitvector.

```
Function FFSEM-ENCRYPT(
  index i,
  integer n,
  key k,
  integer rounds)

  width = (lg(n) + 1) & 0xfffffffe
  do
     t = bit-encode(i)
     for j = 1 to rounds
        t = FFSEM-ROUND(t, width, k, j)
     i = bit-decode(t)
  while(i > n-1)

  return i
```

Note that encrypting any specific type of data requires a function to map it onto the set $0..n-1$. For some sets, this is straightforward, but others are sparse. For instance, it may be desirable to limit credit card numbers to only strings that start with a five or a six. In these cases it may be necessary to first produce a function to map the sparse valid values onto the non-sparse set of sequential integers.

3.4 FFSEM Decryption

Decryption is similar to encryption, using the Feistel rounds and terminating when the Feistel output represents an integer smaller than n. It uses the same cycling construction, terminating and returning its output when the output of the Feistel rounds is in the proper range.

```
Function FFSEM-DECRYPT(
  index i,
  integer n,
  key k,
  integer rounds)

  width = (lg(n) + 1) & 0xfffffffe
  do
     t = bit-encode(i)
```

```
    for j = 1 to rounds
       t = FFSEM-ROUND(t, width, k, rounds-j)
    i = bit-decode(t)
while(i > n-1)

return i
```

4 Summary of Properties

FFSEM is a somewhat unusual mode in that it does not encrypt multiple blocks of data. However, the standard tools for reasoning about the security of a cipher mode work in this case. The mode is not a substitute for a standard mode, but allows for AES encryption of data in applications where data expansion is not acceptable, or protocol security requirements dictate that plaintext contains no redundancy.

4.1 Advantages of FFSEM

No Ciphertext Expansion FFSEM encrypts values from a given range back into that same range. In applications where this is required, FFSEM allows use of an established block cipher in a mode with proven security bounds.

4.2 Disadvantages of FFSEM

Performance FFSEM requires multiple invocations of the block cipher to encrypt a single data item. Depending on the size of the item to be encrypted, anywhere from six to several hundred invocations may be required.

Non-deterministic Performance Due to the cycling construction of FFSEM, different data items can take more or less time to encrypt or decrypt. Note that in some applications, it is conceivable that this may require countermeasures to prevent timing attacks.

Encrypt only FFSEM does not provide integrity or authentication properties, only encryption.

5 Security Considerations

FFSEM is used to encrypt values smaller than the block size of the underlying cipher, and this creates the need to use some care when designing protocols with this mode. While there are no known practical attacks against the FFSEM permutation, encrypting small ranges creates the risk of exploitable attacks if an encrypt or decrypt oracle is available to the attacker. Small ranges ($2^{10} <$

$n < 2^{40}$) appear to be safe from the best known attack against the construction, but applications can specify a larger round count to provide a safety margin if desired.

5.1 Exhaustion Attacks

Like any cipher, if FFSEM is used in a deterministic mode (no tweak or key change between items), an attacker with access to an encryption or decryption oracle can potentially build a dictionary which maps ciphertexts to plaintexts. If the space of possible plaintexts is small, this attack may result in a dictionary that spans some significant fraction of the the entire plaintext space.

There are two defenses against this attack: randomization and eliminating oracle access. To randomize the encryption process, keys can be changed per data item or FFSEM can be utilized in conjunction with a tweaking process like Liskov-Rivest XEX [3]. This document does not describe a full tweaked mode specific to FFSEM. In some applications, it may be desirable to have a deterministic permutation, which preserves the identity of plaintexts in the corresponding ciphertexts. In this case, protocol or operational mechanisms should be used to prevent arbitrary attacker access to keyed encryption or decryption operations.

5.2 Determining Round Count

Patarin [4] shows that for a computationally unbounded attacker, distinguishing the FFSEM permutation from a random permutation requires a minimum of $O(2^{lg(n)/2})$ plaintext/ciphertext pairs. Due to the fairly small set sizes involved, the required number of pairs is not necessarily prohibitive. For example, if $n = 40$, 2^{20} plaintext-ciphertext pairs would be required.

While $O(2^{lg(n)/2})$ represents the theoretical bound on the number of plaintext/ciphertext pairs, the best known actual attack, due to Patarin, requires $O(k2^m)$ ciphertext/plaintext pairs, and $O(2^{km2^m})$ computations for a cipher with block width $2m$ and k rounds. For even small values of m, these values get large quite quickly. For instance, an 8 round and 10 bit wide permutation would require $O(2^{1280})$ computational steps. Note that increasing the number of rounds provides a defense here, which restores $lg(k)$ bits of security against this attack.

We do not recommend the use of FFSEM on sets smaller than 2^{32}. The prefix technique shown by Black and Rogaway [2] is practical for sets slightly smaller than this, and is indistinguishable from a random permutation (modulo flaws in the underlying block cipher used) even with the entire permutation available to the attacker.

While using six rounds satisfies the Patarin security bounds, implementors may desire to use additional rounds to insure that the theoretic attack is not conceivably implementable. We do not specify a defined set of allowable rounds, other than requring that six or more rounds are used. As a general guideline, for values of $n > 40$ bits, the standard six rounds should be sufficient. For

$32 < n < 40$, additional rounds should be used to compensate for the small number of ciphertext/plaintext pairs required for the theoretic attack. Using 128 rounds adds 7 bits of work to the theoretic attack, providing an ample security margin.

6 Acknowledgements

Thanks to Hovav Shacham, Luther Martin, Xavier Boyen and Dan Boneh for valuable technical input and review during the development of this paper. Also, thanks to Eric Rescorla and Steve Haas for extensive advice on the presentation of the material and editorial comments.

References

[1] Stefan Lucks. The sum of PRPs is a secure PRF. *Lecture Notes in Computer Science*, 1807:470–??, 2000.

[2] John Black and Phillip Rogaway. Ciphers with arbitrary finite domains. In *CT-RSA*, pages 114–130, 2002.

[3] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 31–46, London, UK, 2002. Springer-Verlag.

[4] Jacques Patarin. Security of random feistel schemes with 5 or more rounds. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 106–122. Springer, 2004.

[5] M. Bellare, P. Rogaway, and D. Wagner. A conventional authenticated-encryption mode, 2003.

[6] Chris Hall, David Wagner, John Kelsey, and Bruce Schneier. Building PRFs from PRPs. *Lecture Notes in Computer Science*, 1462:370–??, 1998.

[7] M. Luby and C. Rackoff. How to construct psuedorandom permutations from psuedorandom functions. *SIAM J. Computing*, 17(2):373–386, 1988.

[8] M. Bellare and R. Impagliazzo. A tool for obtaining tighter security analyses of pseudorandom function based constructions, with applications to prp to prf conversion. Cryptology ePrint Archive, Report 1999/024, 1999. `http://eprint.iacr.org/`.