An enciphering scheme based on a card shuffle

Ben Morris
Mathematics, UC Davis

Joint work with Viet Tung Hoang (Computer Science, UC Davis) and Phil Rogaway (Computer Science, UC Davis).

Setting

Blockcipher construction

pseudorandom function $\longrightarrow$ pseudorandom permutation

Most current methods rely on either:

Feistel networks, or

SP networks

New method: Swap-or-not shuffle. Stronger provable-security results.

Contribution: Swap-or-not

- A new method to construct a blockcipher
- A proof that it works, and with much better bounds than with Feistel

Security of Swap-or-not : Numerical Examples

Domain	size	# rounds	$\mathrm{Adv}^{\mathrm{CCA}}$	# queries
64-bit strings	2^{64}	1200	$< 10^{-10}$	2^{63}
social security numbers	10^9	340	$< 10^{-10}$	10^8
credit card numbers	10^{16}	500	$< 10^{-10}$	10^{15}

Flexible domain

Our cipher works directly on nonbinary domains such as credit card numbers and social security numbers.

The Problem

PRF $\longrightarrow$ PRP

Luby, Rackoff 88
Patarin 90, 03, 10
Maurer 92
Maurer, Pietrzak 03
M, Rogaway, Stegers 09

Proven upper bounds for enciphering n-bit strings:

method	# rounds	# queries	
Balanced Feistel	3	$q \approx 2^{n/4}$	Luby, Rackoff
	r	$q \approx 2^{n/2-1/r}$	Maurer, Pietrzak
	6	$q \approx 2^{n/2}$	Patarin
Thorp shuffle	$O(n)$	$q \approx 2^{(1-\epsilon)n}$	M, Rogaway, Stegers
Swap-or-not	$O(n)$	$q \approx (1-\epsilon)2^n$	today's talk

Format-preserving Encryption

Finite set $\mathcal{M}$ of messages.

$$\begin{array}{lll} \text{Eg} & \mathcal{M} = & \{\text{social security numbers}\} \\ & \mathcal{M} = & \{\text{credit card numbers}\} \end{array}$$

Want PRP $\pi : \mathcal{M} \to \mathcal{M}$.

It's not clear how to do this using AES.

Format-preserving Encryption

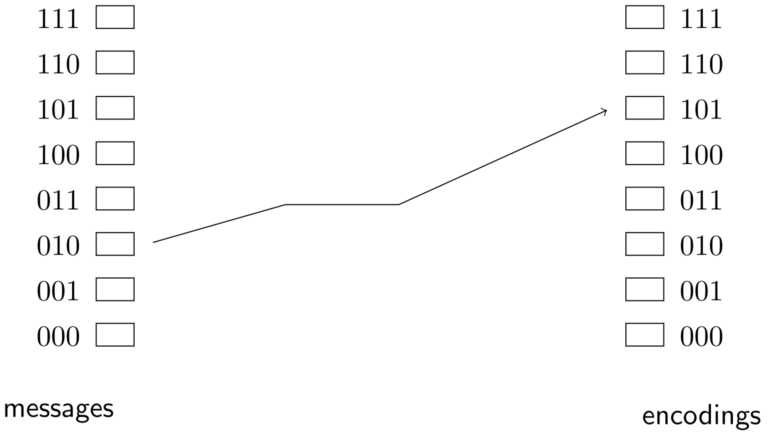
Bounds on balanced Feistel give security up to roughly $\sqrt{|\mathcal{M}|}$ queries.

Problem. $\mathcal{M} = \{\text{social security numbers}\}$

$$|\mathcal{M}| = 10^9$$
$$\sqrt{|\mathcal{M}|} \approx 32,000 \qquad \text{not too big}$$

Swap-or-not provides a practical solution to FPE on domains of troublesome size.

Enciphering scheme $\longleftrightarrow$ Card shuffle



Oblivious shuffle (Naor): you can follow the trajectory of one card without attending to the others.

Swap-or-not shuffle

111

110

101

100

011

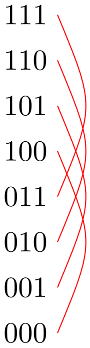
010

001

000

At step t, choose K_t uniformly at random from $\{0, 1\}^n$. Pair each x with $K_t \oplus x$. For each pair, flip a coin. If the coin lands heads, swap the cards at those locations.

Swap-or-not shuffle



K_t induces a random matching.

(Pictured is the case $K_t = 100$.)

At step t, choose K_t uniformly at random from $\{0,1\}^n$. Pair each x with $K_t \oplus x$. For each pair, flip a coin. If the coin lands heads, swap the cards at those locations.

Alternative view

```
function E_KF(x)        //swap-or-not
for t ← 1 to r do
    x̂ ← max(x, K_t ⊕ x)
    b ← F_t(x̂)
    if b = 1 then x ← K_t ⊕ x
return x
```

function $E_{KF}(x)$ //**swap-or-not**
for $t \leftarrow 1$ to r do
  $\widehat{x} \leftarrow \max(x, K_t \oplus x)$
  $b \leftarrow F_t(\widehat{x})$
  if $b = 1$ then $x \leftarrow K_t \oplus x$
return x

Cipher E encrypts $x \in \{0,1\}^n$ using a key KF naming $K_1, \ldots, K_r \in \{0,1\}^n$ and round functions $F_1, \ldots, F_r : \{0,1\}^n \to \{0,1\}$.

Decryption: same, except run from r down to 1.

Why this works: Each round is its own inverse. To reverse the effect of the final round, run it again. Then run the next-to-last round, and so on.

Alternative view

Note that $\pi(x)$ is of the form $x \oplus \sum_{i \in S_x} K_i$.

But this is not linear. S_x is adaptively constructed.

Quantifying the advantage of an adversary

Random permutation π.

Adversary A queries π and π^{-1}, then outputs a bit b. His advantage is $\mathbf{P}(b=1) - \mathbf{P}_u(b=1)$.

$$\begin{aligned} \mathbf{Adv}^{\mathrm{cca}}(q) &= \text{maximum advantage when } A \text{ is limited to } q \text{ queries} \\ \mathbf{Adv}^{\mathrm{ncpa}}(q) &= \text{maximum advantage when } A \text{ is limited to } q \\ & \text{nonadaptive queries of } \pi \end{aligned}$$

Theorem (Maurer, Pietrzak, Renner 2007)

If F and G are blockciphers on the same message space, then, for any q,

$$\mathbf{Adv}^{\mathrm{cca}}_{F \circ G^{-1}}(q) \le \mathbf{Adv}^{\mathrm{ncpa}}_{F}(q) + \mathbf{Adv}^{\mathrm{ncpa}}_{G}(q).$$

Quantitative bound

Theorem

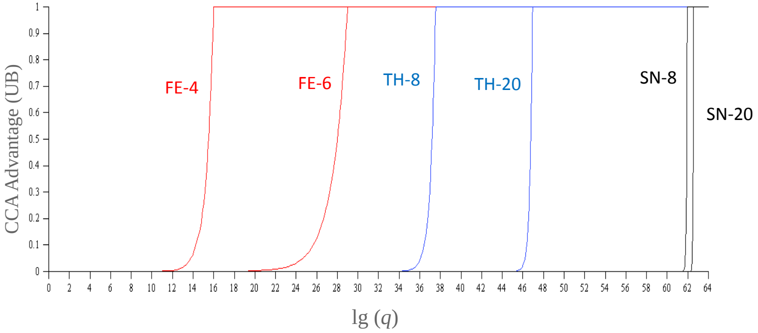
For r rounds of swap-or-not on $\{0,1\}^n$,

$$\mathbf{Adv}^{\mathrm{cca}}(q) \leq \frac{2^{2+3n/2}}{r+4}\left(\frac{q+2^n}{2^{n+1}}\right)^{r/4+1}.$$

If $q \leq (1-\epsilon)2^n$ then the advantage is small after $O(n)$ rounds.

Feistel, Thorp, Swap-or-Not

on $\mathcal{M} = \{0,1\}^{64}$



Proof sketch

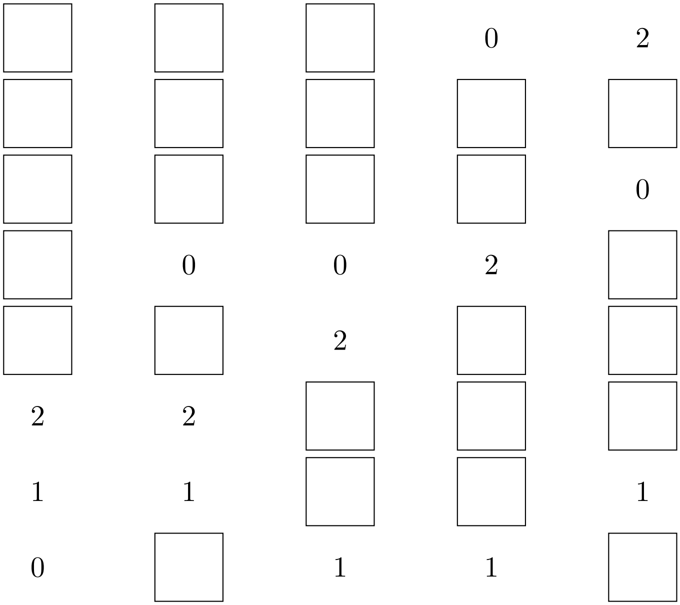
By MPR07, we may assume a non-adaptive adversary who queries only π. For simplicity, suppose the queries are $\pi(0), \ldots, \pi(q-1)$.

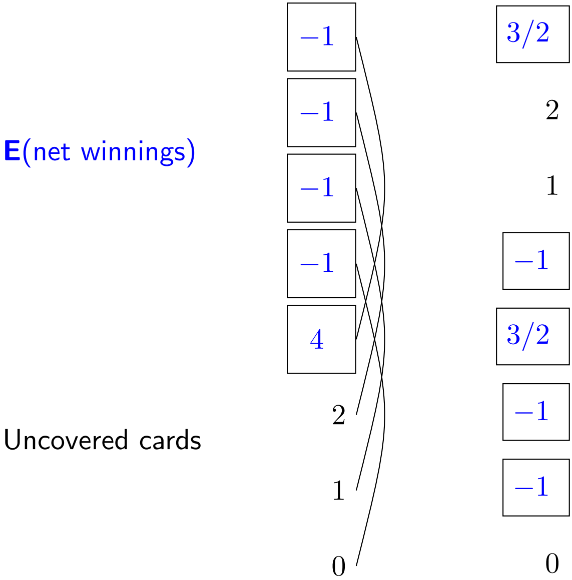
Game: Do r swap-or-not shuffles. Now turn over the cards labeled $0, 1, 2, \ldots$ (reveal $\pi(0), \pi(1), \ldots$).

Before each step, the adversary pays $\$1$. If he guesses the next card's location correctly, he wins $\$k$ if k cards were face down.

Claim: If expected net winnings ≈ 0, then the adversary has small advantage.

It remains to show that the expected winnings are small. This is true even if when we turn over a card we reveal its whole trajectory!


0 2
0
0 0 2
2
2 2
1 1 1
0 1 1



E(net winnings)
Uncovered cards
−1
−1
−1
−1
4
2
1
0
3/2
2
1
−1
3/2
−1
−1
0

Let $w_i(t)$ be the expected net winnings if the adversary guesses i.

Note: the adversary can expect to win $\max_i w_i(t)$.

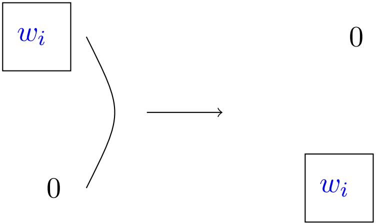
Let $W(t) = \sum_i w_i(t)^2$.

Claim: If $q \leq (1-\epsilon)2^n$ then

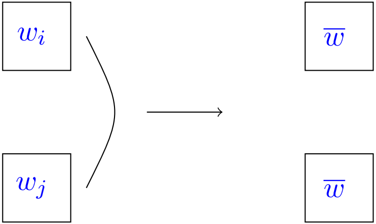
$$\mathbf{E}\left(W(t+1)\right) \leq (1-\epsilon/2)\mathbf{E}(W(t)).$$

Say an covered card is *good* if it is matched to another covered card.

Not good:



Good:



$$\overline{w}^2 + \overline{w}^2 = \tfrac{1}{2}(w_i^2 + w_j^2) + w_i w_j$$

cross terms
are 0 on the
average

Recall that $W(t) = \sum_i w_i(t)^2$.

Good cards are expected to contribute $\frac{1}{2} w_i^2(t)$ to $W(t+1)$. Not good cards contribute $w_i^2(t)$ to $W(t+1)$. It follows that

$$
\begin{aligned}
\mathbf{E}\left(W(t+1) \mid W_t\right) &= \mathbf{P}(\text{good})\tfrac{1}{2}W(t) + \mathbf{P}(\text{not good})W(t) \\
&= \left(1 - \tfrac{1}{2}\mathbf{P}(\text{good})\right) W(t) \\
&\leq (1 - \epsilon/2)W(t),
\end{aligned}
$$

since $\mathbf{P}(\text{good}) \geq \epsilon$. $\square$

Using swap-or-not to make confusion/diffusion ciphers

Example: Specify F_t by an n-bit string L_t and let $F_t(\widehat{x}) = L_t \odot \widehat{x}$ be the inner product of L_t and $\widehat{x}$.

function $E_{KL}(x)$ //**inner product realization**
for $t \leftarrow 1$ to r do
    $\widehat{x} \leftarrow \max(x, K_t \oplus x)$
    $b \leftarrow L_t \odot \widehat{x}$
    if $b = 1$ then $x \leftarrow K_t \oplus x$
return x

Cipher E encrypts $x \in \{0,1\}^n$ using a key KL that specifies $K_1, \ldots, K_r, L_1, \ldots, L_r \in \{0,1\}^n$.

We don't know how many rounds to suggest.

More general domain

If the domain is a finite, abelian group $(G, +)$, the cipher is the same as before, except

- Choose K_t uniformly at random from G.
- Pair x with $K_t - x$.

function $E_{KF}(x)$ //**generalized domain**
for $t \leftarrow 1$ to r do

$\widehat{x} \leftarrow \max(x, K_t - x)$

$b \leftarrow F_t(\widehat{x})$

if $b = 1$ then $x \leftarrow K_t - x$

return x

Cipher E encrypts $x \in G$ using a key KF naming $K_1, \ldots, K_r \in G$ and round functions $F_1, \ldots, F_r : G \to \{0, 1\}$.