A Formal Treatment of Remotely Keyed Encryption

Matt Blaze
AT&T Labs

Joan Feigenbaum
AT&T Labs

Moni Naor
Weizmann Institute

Who will guard the guards?

No cryptographic protocol is stronger than the mechanism protecting its secret keys.

Almost any computer connected to the world will be corrupted (at least partly) at some point in time.

However: in most systems no safe place for storing the keys.

Idea: add a special purpose device for encryption (Smart-Card, PCMCIA).

Where should I put it....?

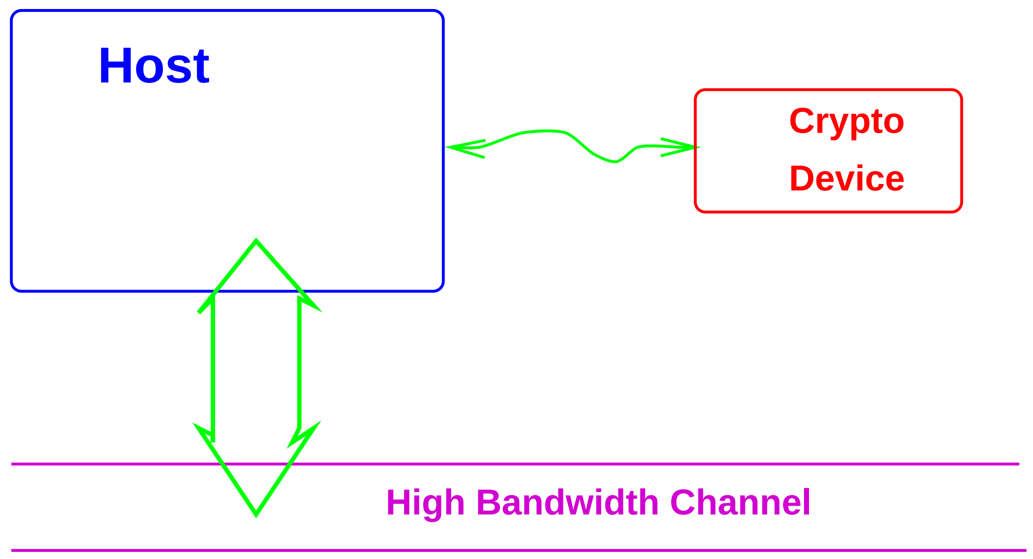
Special purpose device

Advantages:

- Limited functionality, fewer places to err, easier to design
- Can design once and for all. Should work with all systems.
- Can be cheap - smart-card (even with reader)

Problems:

- Bandwidth from device to host. Should be as high as any link.
- Does not grow with the host. Keys/card may live many years.



Remotely keyed encryption problem

How to do encryption/decryption in a high-bandwidth environment taking advantage of:

- The power (bandwidth, computing) of the host.
- Superior security of the smart-card.

Security risk: host is completely controlled by attacker for certain periods of time.

History

- Lots of work on host helping card with *computation* - Public-key.

- Complexity Theoretical work

- Blaze - first to call attention to bandwidth problem. Weakness - adversary can generate more encryptions after it ceases control

- Lucks - attempt to define security formally. Suggested 3 desirable properties. We claim - intuitively insecure! Reveals a lot of information on plaintexts.

- Lots of of work of session key exchange. Shoup and Rubin - similar adversary. Requires interaction between hosts. Irrelevant to problems such as disk traffic encryption.

Model

Two parties: $Host$, $Card$. To encrypt/decrypt ($Host$, $Card$) interact.

Desirable: low communication.

An adversary $\mathcal{A}$ attacks the system:

- Host Phase - Adversary $\mathcal{A}$ controls the $Host$ and all its communication links. However, $\mathcal{A}$ *cannot* see internal computation of the $Card$. (No gentle physical pressure!)
- Challenge Phase - Adversary $\mathcal{A}$ ceases control of the internal communication. Can still attack the pair ($Host$,$Card$) externally.

Goal: Adversary $\mathcal{A}$ should not learn anything *new in the Challenge Phase.*

Problem: Characterize what $\mathcal{A}$ learned in Host Phase.

Our results

- Definition of Security for Remotely-Keyed Encryption Schemes (RKES).

  Length Preserving Encryption and
  Length Increasing Encryption

- Construction of a Length-Increasing RKES.

- Construction of a Length-Preserving RKES

All construction encrypt n blocks with fixed communication and computation at the card.

Rigorous Specification of Security

To define security of a system must specify:

1. The power of the adversary - both computational *and* access to the system.

2. What constitute a failure of the system - what it means to break the system.

A security proof of a system implementation: no adversary with limitation as (1) can break the system as in (2) (assuming some hardness assumptions).

Length Preserving and Increasing Encryption

Length Increasing Encryption:

- Less restrictive
- May provide better security.

Length Preserving Encryption:

- Saves on memory and communication bandwidth
- Easy to embed in existing systems - doesn't destroy formats (sectors, packets)

Problem: what to do with repeated blocks?

Solutions:

- Chaining (CFB,CBC) - reveals prefix information.
- Permutation on very large blocks - our approach.

Definition - Length Preserving RKES

Input $X = (X_1, \ldots, X_n)$, Output $Y = (Y_1, \ldots, Y_n)$. Each $X_i, Y_i \in \{0, 1\}^b$.

Non-RKES security: Encryption function should be a pseudo-random permutation $\prod$.

Even if adversary $\mathcal{A}$ can access $\prod$ and $\prod^{-1}$, $\mathcal{A}$ cannot distinguish it from a random permutation.

Too strong for RKES: $\prod$ is not random for $\mathcal{A}$ since it has short description of $\prod(X^1), \prod(X^2), \ldots$

Idea: secure if $\mathcal{A}$ cannot distinguish a switch to a random permutation after host-phase.

What about $X^1, \ldots X^m$ from Host Phase? Well, except them...

Problem: they are not well defined!

Definition - the Arbiter

To prove security should provide an arbiter B to filter the message of the Challenge Phase.

The arbiter $\mathcal{B}$ acts as a *simple* function of the communication of the Host Phase.

The number of messages filtered by B in the Challenge Phase should be bounded by m, the number of interactions in the Host Phase.

Definition - Summary

A length preserving RKES consists of protocols for ($Host$,$Card$) and an arbiter B such that any PPT adversary $\mathcal{A}$ cannot distinguish with non-negligible prob. whether there was a switch to a random permutation between the Host Phase and the Challenge Phase. Where:

- $\mathcal{A}$ completely controls the $Host$ during the Host Phase. Can send any message to $Card$. Number of interactions - m.
- During Challenge Phase $\mathcal{A}$ can ask $Host$ to encrypt or decrypt any message.
- Arbiter B can reject up to m encryption queries and m decryption queries.

Tools

- A pseudo-random function $F_k : \{0,1\}^b \mapsto \{0,1\}^b$
- A pseudo-random permutation

$$E_k, D_k : \{0,1\}^b \mapsto \{0,1\}^b.$$

  E, D should be a *strong* pseudo-random permutation (and its inverse).

  E and F may be implemented by "common" block ciphers.

- A length preserving encryption scheme

$$G_S : \{0,1\}^{nb} \mapsto \{0,1\}^{nb}.$$

  If S is random, then $G_S(X_1, \ldots X_n)$ is pseudo-random for all $X_1, \ldots X_n$.

  Possible realizations: a pseudo-random generator, permutation on large or small blocks.

- A collision intractable hash function

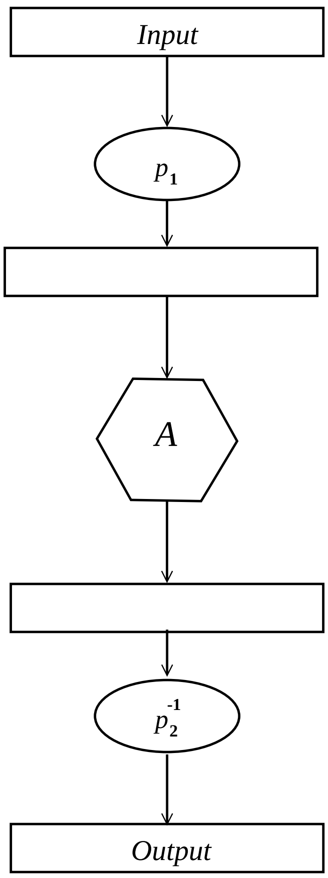
$$H : \{0,1\}^* \mapsto \{0,1\}^b.$$

  Should be infeasible to come up with $X \neq Y$ such that $H(X) = H(Y)$.

The NR-Framework

Compose $\prod = p_1 \circ A \circ p_2^{-1}$ where:

- $p_1, p_2, A : \{0,1\}^\ell \mapsto \{0,1\}^\ell$ are permutations.
- p_1 and p_2 are lightweight — mostly $Card$.
- A is heavy — mostly $Host$.



. . . The Adapted Framework

A should be indistinguishable from a truly random permutation on "most" sequences.

Role of p_1 and p_2 to ensure that A sees only the good inputs.

Key point: Identify A-Good sequences

$$\{\langle X^1, Y^1\rangle, \ldots, \langle X^m, Y^m\rangle\}$$

for p_1 and p_2 s.t.

- No adversary $\mathcal{A}$ can find with non-neg. prob. a sequence that is *not* A-Good for p_1 and p_2, even if A is known.
- A switch to a random permutation $\tilde{A}$ is indistinguishable, given that the sequence was A-Good.

Theorem: If $\prod = p_1 \circ A \circ p_2^{-1}$ and p_1, p_2 and A are as above, then $\prod$ is a pseudo-random permutation.

Our Construction

p_1 and p_2 change only the first block

- $p_1 : (X_1, X_2, \ldots, X_n) \mapsto (W, X_2, \ldots, X_n)$,
  W is a function of X_1 and $h_x = H(X_2, \ldots, X_n)$
- $p_2 : (Y_1, Y_2, \ldots, Y_n) \mapsto (Z, Y_2, \ldots, Y_n)$,
  Z is a function of Y_1 and $h_y = H(Y_2, \ldots, Y_n)$

The permutation A is defined by two keys (k_3, k_4) so that $A : (W, X_2, \ldots X_n) \mapsto (Z, Y_2, \ldots Y_n)$ where

$$Z = E_{k_3}(W)$$

and

$$Y_2, \ldots Y_n = G_{F_{k_4}(W)}(X_2, \ldots, X_n)$$

Evaluation

Evaluation of p_1 by ($Host$,$Card$):

- $Host$ computes $h_x = H(X_2, \ldots, X_n)$ and sends (X_1, h_x).
- $Card$ computes W based on its secret keys.

Evaluation of A by ($Host$,$Card$):

- $Card$ computes $S = F_{k_4}(W)$ and $Z = E_{k_3}(W)$ and sends (S, Z).
- $Host$ computes $Y_2, \ldots Y_n = G_S(X_2, \ldots, X_n)$

Evaluation of p_2^{-1} by ($Host$,$Card$):

- $Host$ computes $h_y = H(Y_2, \ldots, Y_n)$ and sends it.
- $Card$ computes Y_1 as a function of (Z, h_y) based on its secret keys.

Inversion is done the same!

Non-Colliding Encryption

A-Good sequences - different X's have different Z's.

To specify p_1 and p_2 want *non-colliding encryption*: a pair $(C_k, \widehat{C}_k)$ where

$$C_k : \{0,1\}^b \times \{0,1\}^b \mapsto \{0,1\}^b$$

and

$$\widehat{C}_k : \{0,1\}^b \times \{0,1\}^b \mapsto \{0,1\}^b$$

such that

- $\forall V \in \{0,1\}^b$ and $h \in \{0,1\}^b$
  $\widehat{C}_k(C_k(V,h),h) = V$ and $C_k(\widehat{C}_k(V,h),h) = V$.
- It is hard to find $(V,h) \neq (V',h')$ such that

  $$C_k(V,h) = C_k(V',h').$$

  and where (V,h) *and* (V',h') did *not* appear in a query.

(V,h) *appeared in a query* if $C_k(V,h)$ was a query or if V was the reply to query $\widehat{C}_k(U,h)$.

...Non-Colliding Encryption

Construction 1:

Idea: apply F on h and to obtain a "session-key" to encrypt V.

$C_k(V,h) = E_{F_k(h)}(V)$ and $\widehat{C}_k(V,h) = D_{F_k(h)}(V)$.

Claim: $(C_k, \widehat{C}_k)$ as defined above is a non-colliding encryption scheme.

Proof based on comparison with process that has access to many random permutations $E_1, E_2, \ldots$

...Non-Colliding Encryption

Construction 2:

Idea: apply F on h and to obtain a "mask" for encrypting V.

$$C_k(V,h) = E_{k_1}(F_{k_2}(h) \oplus V) \oplus F_{k_3}(h) \text{ and}$$
$$\widehat{C}_k(V,h) = D_{k_1}(F_{k_3}(h) \oplus V) \oplus F_{k_2}(h).$$

Claim: $(C_k, \widehat{C}_k)$ as defined above is a non-colliding encryption scheme.

Proof based on Even-Mansour.

Encryption Protocol

Encryption : input $X_1, \ldots, X_n$**; output** $Y_1, \ldots, Y_n$

P1 Host: $h_x \leftarrow H(X_2, \ldots, X_n)$

P2 Host $\rightarrow$ Card: h_x, X_1

P3 Card: $W \leftarrow E_{F_{k_1}(h_x)}(X_1)$

P4 Card: $Z \leftarrow E_{k_2}(W)$

P5 Card: $S \leftarrow F_{k_3}(W)$

P6 Card $\rightarrow$ Host: S

P7 Host: For $j \leftarrow 2$ to n, $Y_j \leftarrow G_S^j(X_2, \ldots X_n)$.

P8 Host: $h_y \leftarrow H(Y_2, \ldots, Y_n)$

P9 Host $\rightarrow$ Card: h_y

P10 Card: $Y_1 \leftarrow E_{F_{k_4}(h_y)}(Z)$

P11 Card $\rightarrow$ Host: Y_1

Decryption Protocol

Decryption : **input** $Y_1, \ldots, Y_n$; **output** $X_1, \ldots, X_n$

P12 Host: $h_y \leftarrow H(Y_2, \ldots, Y_n)$

P13 Host $\rightarrow$ Card: h_y, Y_1

P14 Card: $Z \leftarrow D_{F_{k_4}(h_y)}(Y_1)$

P15 Card: $W \leftarrow D_{k_2}(Z)$

P16 Card: $S \leftarrow F_{k_3}(W)$

P17 Card $\rightarrow$ Host: S

P18 Host: For $j \leftarrow 2$ to n, $X_j \leftarrow \widehat{G}_S^j(Y_2, \ldots, Y_n)$.

P19 Host: $h_x \leftarrow H(X_2, \ldots, X_n)$

P20 Host $\rightarrow$ Card: h_x

P21 Card: $X_1 \leftarrow D_{F_{k_1}(h_x)}(Z)$

P22 Card $\rightarrow$ Host: X_1

The Arbiter and Security Theorem

Arbiter $\mathcal{B}$: On encryption query $(X_1, \ldots, X_n)$, compute

$$h = H(X_2, \ldots, X_n)$$

and check whether (h, X_1) occured in the transcript of the host phase.

On decryption query $(Y_1, \ldots, Y_n)$, compute

$$h = H(Y_1, \ldots, Y_n)$$

and check whether (h, Y_1) occured in the transcript of the host phase.

Theorem 1 A Scheme $\prod = p_1 \circ A \circ p_2^{-1}$ where p_1 and p_2 are based on non-colliding encryption and A is based on secure G_S and E is a secure length preserving RKES.

A-Good Sequences

The Host Phase transcript:

$$(X_1^1, h_x^1, Y_1^1, h_y^1), \ldots (X_1^{m_1}, h_x^{m_1}, Y_1^{m_1}, h_y^{m_1})$$

The Challenge Phase transcript: from the host phase and

$$(X^{m_1+1}, Y^{m_1+1}), \ldots (X^{m_1+m_2}, Y^{m_1+m_2})$$

The sequence is A-good for p_1 and p_2 if the W's and Z's of the Challenge phase are different from one another and from those of the host phase, I.e.

1. For all $1 \leq i < j \leq m_2$ if $X^{m_1+i} \neq X^{m_1+j}$ and X^{m_1+i} *and* X^{m_1+j} are not filtered by B, then $W^i \neq W^j$. (Similarly for the Y's.)

2. For all $1 \leq i \leq m_2$ if X^{m_1+i} is not filtered by B, then $W^{m_1+i} \neq W^j$ for all $1 \leq j \leq m_1$. (Similarly for the Y's.)

Proof

Claim: The adversary $\mathcal{A}$ cannot find a sequence which is not A-Good for p_1 and p_2, even if it knows A.

Proof is based on non-colliding encryption.

...Proof

Replace A with a "more" random process.

1. $\tilde{A}_1$ - E_{k_3} is replaced with a random permutation and G_S is replaced with a process $\tilde{G}$ that on input S produces a random string of length $(n-1) \cdot b$.

2. $\tilde{A}_2$ - a random permutation.

Claim: Adversary $\mathcal{A}$ cannot distinguish when A is switch with $\tilde{A}_1$ following Host Phase.

Claim: Adversary $\mathcal{A}$ cannot distinguish between a switch to $\tilde{A}_1$ and a switch to $\tilde{A}_2$.

...Open Problems

- Are the two rounds in encryption/decryption necessary?

- Is the use of a collision intractable function (and not UOWHF) necessary.