Revised: 11 Feb 99.

ECS 227 — Modern Cryptography — Winter 99
Lecture 5

Symmetric Encryption
(Part II)

4 February 1999

Today we practice reductions. We show equivalences among the definitions we gave last time. We will also pay some attention to the efficiency of our reductions and look at how you show that a reduction is tight.

Review the last set of lecture notes to see the definitions for real-or-random security, find-then-guess security, and semantic security. In today's lecture will be establishing the following:

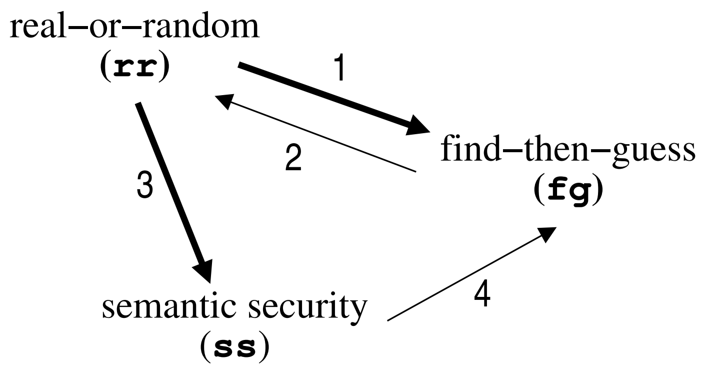

Figure 1: *Relations among the notions. We draw an arrow from notion A to notion B when security in the sense of A implies security in the sense of B. The dark arrows indicate tight reductions; the ordinary arrows indicate that there is a loss of security in the reduction.*

One of these theorems, the one numbered 1, was done last lecture. So we have three more reductions to give today. Let us begin.

1 Find-then-Guess $\Rightarrow$ Real-or-Random

Theorem 1.1 Let $\Pi = (\mathcal{K}, \mathcal{E}, D)$ be a symmetric encryption scheme. Suppose there exists an adversary $A_{\rm rr}$ that $(t, q, \mu; \delta)$-breaks$^{\rm rr}$ Π. Then there exists an adversary $A_{\rm fg}$ that $(t', q', \mu'; \delta')$-breaks$^{\rm fg}$ Π, where

$$\begin{aligned} t' &= t + O(\mu), \\ q' &= q, \\ \mu' &= \mu, \text{ and} \\ \delta' &= \delta/q\ . \end{aligned}$$

Proof. This results seems harder to prove than the last one we did. We have an adversary $A_{\rm rr}$ that does a good job breaking Π in the real-or-random sense. We need to construct an adversary that does a good job breaking Π in the find-then-guess sense. This is a tall order—how is the adversary $A_{\rm fg}$ going to behave? The operating environment for a real-or-random adversary is very different from a find-then-guess adversary. The first has a single oracle, h, that we imagine to be a proper encryption oracle or else a bogus encryption oracle. The adversary experiments with it to try to figure out which. But a find-then-guess adversary always has a proper encryption oracle. It is trying to find a pair of messages that it can distinguish the encryptions of.

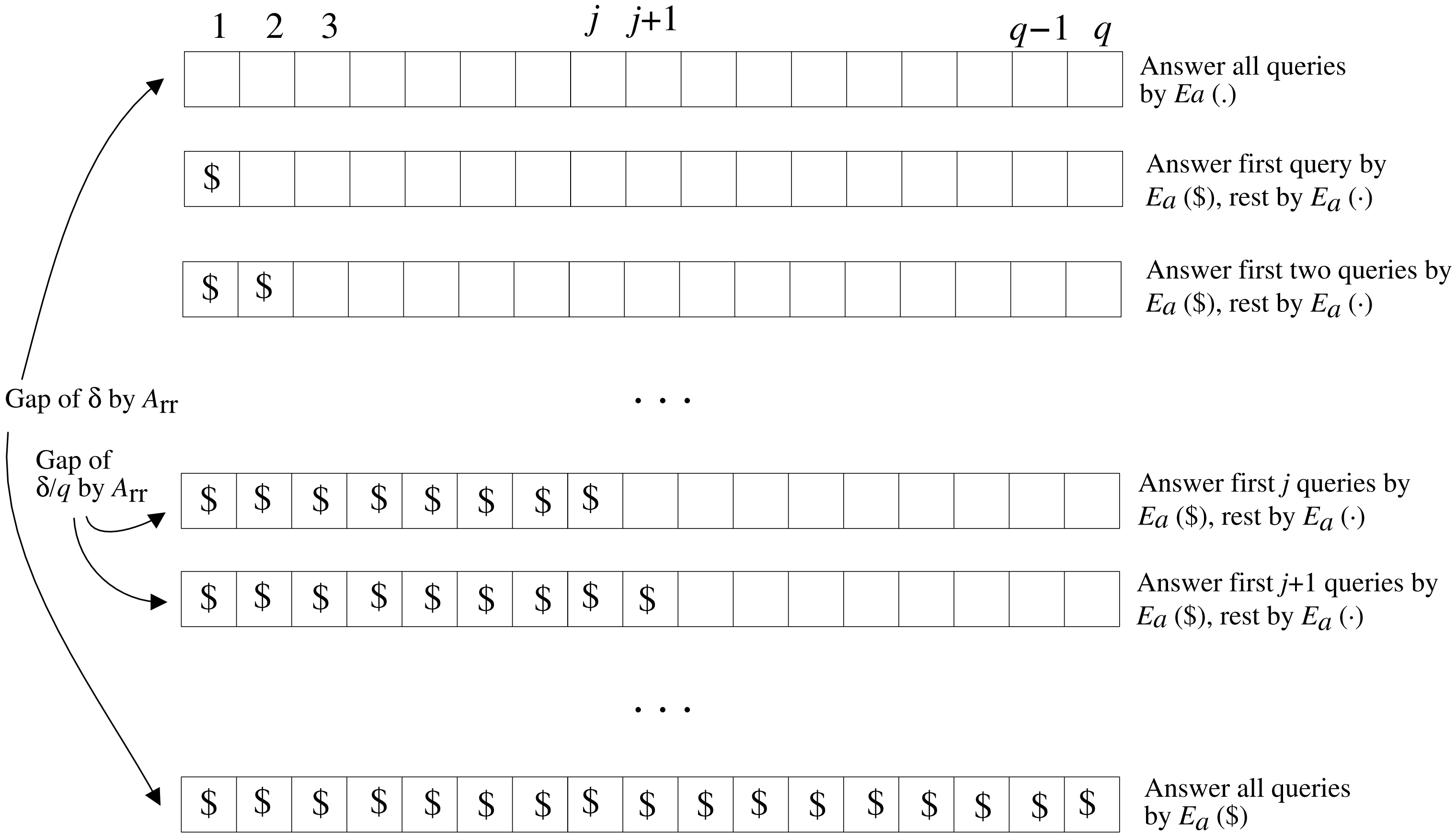

Figure 2: *Hybrid argument used for fg⇒rr.*

We will use a hybrid argument. Here is the adversary $A_{\rm fg}$, parameterized by some number j.

Algorithm $A^g_{\rm fg}(\textsf{find})$:
  Run $A^h_{\rm rr}$
    When $A^h_{\rm rr}$ makes its i^{th} oracle query, x_i, answer as follows:
      If $i < j$, then return $y_i \leftarrow g(\$^{|x_i|})$.
      If $i = j$, then we are done: save the current state in s and return
        $(M_0, M_1, s) \leftarrow (\$^{|x_j|}, x_j, s)$.

Algorithm $A^g_{\rm fg}(\textsf{guess}, C, s)$
  Continue running $A^h_{\rm rr}$, starting in state s.
    Oracle query x_j was just asked by $A^h_{\rm rr}$. Answer it by C.

When $A_{\rm rr}^h$ makes additional oracle queries, x_i, answer with $y_i \leftarrow g(x_i)$.
Eventually, $A_{\rm rr}$ outputs a bit, b'.
Return b'. That is, if $A_{\rm rr}^h$ thinks that h is a good encryption oracle (indicated by output of 1) then $A_{\rm fg}$ guesses that C is the encryption of $M_1 = x_j$; else $A_{\rm fg}$ guesses that C is the encryption of the random string M_0.

What is going on here? We know that if we run $A_{\rm rr}$ answering all of its questions with a real encryption oracle then we get a prediction from $A_{\rm rr}$ that is δ-different from what we get if we answer all of the questions with a random encryption oracle. Consider blending the one extreme into the other: answer the first $0, 1, 2, \ldots$ questions with a random encryption oracle, and the remaining with a real encryption oracle. See Figure 2.

For $0 \leq i \leq q$, let P_i be the probability that $A_{\rm rr}^h$ outputs 1 if we answer the first i oracle queries with a $\mathcal{E}_a(\$^{|\cdot|})$-oracle, and the remaining queries with a $\mathcal{E}_a(\cdot)$-oracle. (In both cases, we initially choose a random $a \in \mathcal{K}$.) We know that

$$p_0 - p_q \geq \delta.$$

By the triangle inequality there exists a $0 \leq j < q$ such that

$$P_j - P_{j+1} \geq \delta/q.$$

Fix such a j and consider the adversary $A_{\rm fg}$ described earlier as specific to this particular j.

Now consider the environment in which $A_{\rm rr}$ finds itself when g is a (good) encryption oracle. And either C is a random encryption of M_0 ($b = 0$) or else C is a random encryption of M_1 ($b = 1$). In the former case we have provided for $A_{\rm rr}$ an environment which is identical to that which defines P_{i+1}: the first $i + 1$ oracles queries have been answered with a random encryption oracle, the rest with a proper encryption oracle. In the latter case we have provided for $A_{\rm rr}$ an environment which is identical to that which defines P_i. We have, therefore, that $A_{\rm fg}$'s advantage δ' is precisely $P_j - P_{j+1} \geq \delta/q$.

The other resource bounds for $A_{\rm fg}$ are easy to check. ▮

2 The Bound Can Not Be Improved

The above proof gives up a factor of q in the security. It is interesting to ask if the loss essential, or if it is just an artifact of the proof.

We argue that the loss is essential; there is no proof that gives a better bound. Thus the real-or-random definition is in some quantitative sense *stronger* than the find-then-guess definition. Let me give the proof idea.

We'd like to construct an encryption scheme with a "gap"—an encryption scheme that is q times "more secure" in the find-then-guess sense than in the real-or-random sense. We'll make a scheme which is *terrible* in the real-or-random sense—some rr-adversary has advantage which is nearly 1—but which for which no adversary has advantage much more than $1/q$ when measured in the find-then-guess sense. As usual, q is the number of queries the adversaries ask, and it is a parameter of our example.

Suppose you have a good encryption scheme, $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Think of this scheme as being very strong in the rr-sense (and therefore the fg-sense, too): the rr-advantage is ≈ 0 for any adversary that spends a reasonable amount of time. We're going to modify Π to make an encryption scheme $\Pi' = (\mathcal{K}, \mathcal{E}', \mathcal{D}')$ which is disproportionately weaker in the rr-sense than in the fg-sense. For simplicity, let's make Π' stateful. At the beginning, $\mathcal{E}'$ chooses a random (secret) $j \in [1..q]$. Then we define $\mathcal{E}'_a(x)$ to return a random $0 \parallel \mathcal{E}_a(x)$ for all queries except the j^{th}, and to return $1 \parallel x$ for the j^{th} query.

Now the scheme Π' is terrible in the rr-sense: a simple adversary can get advantage which is very nearly 1. That adversary repeatedly asks for the encryptions of some particular message M. If the response $1M$ is ever returned, the adversary can be pretty sure that this was a real encryption oracle. If no such response ever comes back from the encryption oracle, then the adversary can be sure that it was a random encryption oracle that she was provided.

The scheme $|\Pi'|$ may not be great in the fg-sense, but it is not nearly so bad as in the rr-sense; no adversary will have a chance more than $1/q$ plus the advantage of an adversary of comparable resources in attacking the scheme Π (which we think of as 0). Why? Consider a find-then-guess adversary attacking Π'. It will be seeing q encryptions, all together: $q-1$ directly asked of its encryption oracle, and one encryption of either M_0 or M_1, as asked by the adversary at the end of its find-stage. Now if the M_b query is the j^{th} query, then we will give up—just assume that the adversary wins. But if it is any of the other $q-1$ queries in which we encrypt the query x as $1x$, we claim that we have not helped the adversary to accomplish her goal (identifying which M_b the challenge C correspond to). For the only new information we have given to the adversary is $1x$ in response to one of the queries x. The adversary could have computed this on her own, and it has nothing to do with whether C is the encryption of M_0 or of M_1.

The above is of course not a formal argument. See [BDJR97] for a proper theorem statement and proof.

3 Real-or-Random $\Rightarrow$ Semantic

This proof won't be a hybrid argument. The proof method is, well, just slight of hand!

Theorem 3.1 Let $\Pi = (\mathcal{K}, \mathcal{E}, D)$ be a symmetric encryption scheme. Suppose there exists an adversary $A_{\rm sm}$ that $(t, q, \mu; \delta)$-breaks$^{\rm sm}$ Π. Then there exists an adversary $A_{\rm rr}$ that $(t', q', \mu'; \delta')$-breaks$^{\rm rr}$ Π, where

$$\begin{aligned} t' &= t + O(\mu), \\ q' &= q + 1, \\ \mu' &= \mu, \text{ and} \\ \delta' &= \delta/2\ . \end{aligned}$$

Proof. Let

$$p = \Pr[a \leftarrow \mathcal{K}\,; (\mathcal{M}, f, s) \leftarrow A_{\rm sm}^{\mathcal{E}_a(\cdot)}(\mathsf{select})\,; M \leftarrow \mathcal{M}\,; C \leftarrow \mathcal{E}_a(M) : A_{\rm sm}^{\mathcal{E}_a(\cdot)}(\mathsf{predict}, C, s) = f(x)]$$

and let

$$q = \Pr[a \leftarrow \mathcal{K}\,; (\mathcal{M}, f, s) \leftarrow A_{\rm sm}^{\mathcal{E}_a(\cdot)}(\mathsf{select})\,; M, M' \leftarrow \mathcal{M}\,; C \leftarrow \mathcal{E}_a(M') : A_{\rm sm}^{\mathcal{E}_a(\cdot)}(\mathsf{predict}, C, s) = f(x)]\ .$$

Now define

$$p_* = \Pr[a \leftarrow \mathcal{K}\,; (\mathcal{M}, f, s) \leftarrow A_{\rm sm}^{\mathcal{E}_a(\$^{|\cdot|})}(\mathsf{select})\,; M \leftarrow \mathcal{M}\,; C \leftarrow \mathcal{E}_a(\$^{|M|}) : A_{\rm sm}^{\mathcal{E}_a(\$^{|\cdot|})}(\mathsf{predict}, C, s) = f(x)]$$

and

$$q_* \quad = \quad \Pr[a \leftarrow \mathcal{K} \,;\, (\mathcal{M}, f, s) \leftarrow A_{\mathrm{sm}}^{\mathcal{E}_a(\$^{|\cdot|})}(\textsf{select}) \,;\, M, M' \leftarrow \mathcal{M} \,;\, C \leftarrow \mathcal{E}_a(\$^{|M'|}) : \\ A_{\mathrm{sm}}^{\mathcal{E}_a(\$^{|\cdot|})}(\textsf{predict}, C, s) = f(x)]$$

That is, the experiment which defines p_* is obtained from the experiment which defines p by replacing all of the "real" encryption oracles by "random" encryption oracles, and similarly for q and q_*. Now we know that

$$p - q \quad \geq \quad \delta.$$

We also observe that

$$p_* \quad = \quad q_*.$$

This is because $\mathcal{M}$ only outputs strings of some one particular length, so if $M, M' \leftarrow \mathcal{M}$ then a random string of length $|M|$ is the same thing as a random string of length M'. We thus have that

$$(p - p_*) + (q_* - q) \quad \geq \quad \delta$$

and so

$$p - p_* \geq \delta/2 \qquad \text{or} \qquad q_* - q \geq \delta/2\ .$$

Either way, we can construct an adversary A_{rr} with the success probability and resource bounds asserted. The adversary is the obvious one. Suppose that $p - p_* \geq \delta/2$. Then consider the Boolean experiment by:

$$[(\mathcal{M}, f, s) \leftarrow A_{\mathrm{sm}}^{g}(\textsf{select}) \,;\, M \leftarrow \mathcal{M} \,;\, C \leftarrow g(M) : A_{\mathrm{sm}}^{g}(\textsf{predict}, C, s) = f(x)]\ .$$

If $g = \mathcal{E}_a(\cdot)$ for a random $a \in \mathcal{K}$ then the experiment exactly coincides with the one defining p, while if $g = \mathcal{E}_a(\$^{|\cdot|})$ for a random $a \in \mathcal{K}$ then the experiment exactly coincides with the one defining p_*. This immediately gives us an adversary A_{rr}^{g}. The other case is analogous. ▮

4 Semantic ⇒ Find-then-Guess

The last reduction that we shall do today is interesting insofar as we use very little of our available power.

Theorem 4.1 Let $\Pi = (\mathcal{K}, \mathcal{E}, D)$ be a symmetric encryption scheme. Suppose there exists an adversary A_{fg} that $(t, q, \mu; \delta)$-breaks$^{\mathrm{fg}}$ Π. Then there exists an adversary A_{sm} that $(t', q', \mu'; \delta')$-breaks$^{\mathrm{sm}}$ Π, where

$$\begin{aligned} t' &= t + O(\mu), \\ q' &= q, \\ \mu' &= \mu, \text{ and} \\ \delta' &= \delta\ . \end{aligned}$$

Proof. We have an adversary attacking the fg-security of Π. We want an sm-adversary. What should this adversary use as its information function f? And what should it use as its message space $\mathcal{M}$? We make very simple choices. Here is the adversary's definition:

Algorithm $A_{\rm sm}^{g}(\textsf{select})$:
  Let $f = \mathrm{id}$ be the identity function
  Run $A_{\rm fg}^{h}(\textsf{find})$
    When $A_{\rm fg}$ makes an oracle query, x, answer it with $y \leftarrow g(x)$.
    Eventually, $A_{\rm fg}$ halts, outputting a triple (M_0, M_1, s), where $|M_0| = |M_1|$.
  Let $\mathcal{M}$ be the message space with two points: M_0 and M_1, each with probability .5.
  Return (f, M, s).

Algorithm $A_{\rm sm}^{g}(\textsf{predict}, C, s)$
  Run $A_{\rm fg}^{h}(\textsf{guess}, C, s)$
    When $A_{\rm fg}$ makes an oracle query, x, answer it with $y \leftarrow g(x)$.
  Eventually, $A_{\rm fg}$ outputs a bit, b.
  If $b = 0$ then output M_0 else output M_1.

With g an encryption oracle $\mathcal{E}_a(\cdot)$, for a random $a \in \mathcal{K}$, we provide $A_{\rm fg}$ an environment which exactly mimics the one which defines adversarial success for an fg-adversary. Adversary $A_{\rm fg}$ answering 0 corresponds to its prediction that C is the encryption of M_0, while outputting 1 corresponds to the prediction that C is the encryption of M_1. Thus by outputting M_0 and M_1 in these two cases we will be correctly predicting the identity function of the plaintext whenever Afg makes a correct prediction. The result follows. ∎

5 Methods for Symmetric Encryption

In the closing minutes of class we looked at some methods for symmetric encryption, starting with a block cipher. We looked at CBC encryption with zero initialization vector (IV) (this doesn't work); CBC encryption with random IV; and counter mode encryption. We explained (as we did in lecture 1) why the first is no good, and we stated a bound for the security of the second. See [BDJR97]. We'll pick up the (very easy) proof for the security of counter-mode encryption in the next lecture. See Figures 3 and 4.

References

Symmetric encryption. Again, the closest match for this lecture is

[BDJR97] "A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation," Mihir Bellare, Anand Desai, Eron Jokipii and Phillip Rogaway, FOCS '97. Available from `http://www.cs.ucdavis.edu/~rogaway/papers/sym-enc-abstract.html`

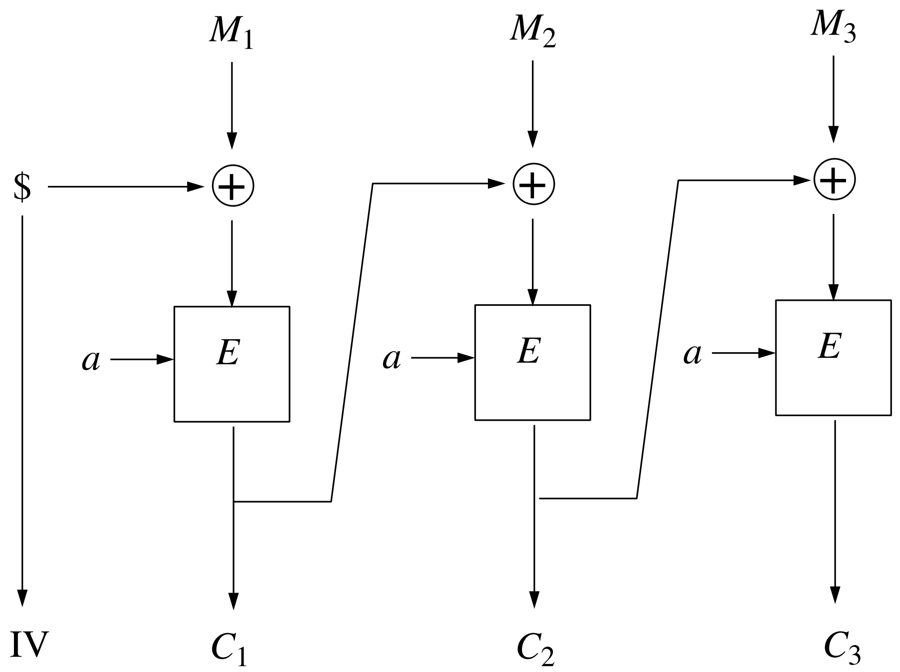

Figure 3: *CBC Encryption with a random IV. Assume the plaintext* $M = M_1M_2M_3$ *is three blocks long. We use the underlying block cipher* E*.*

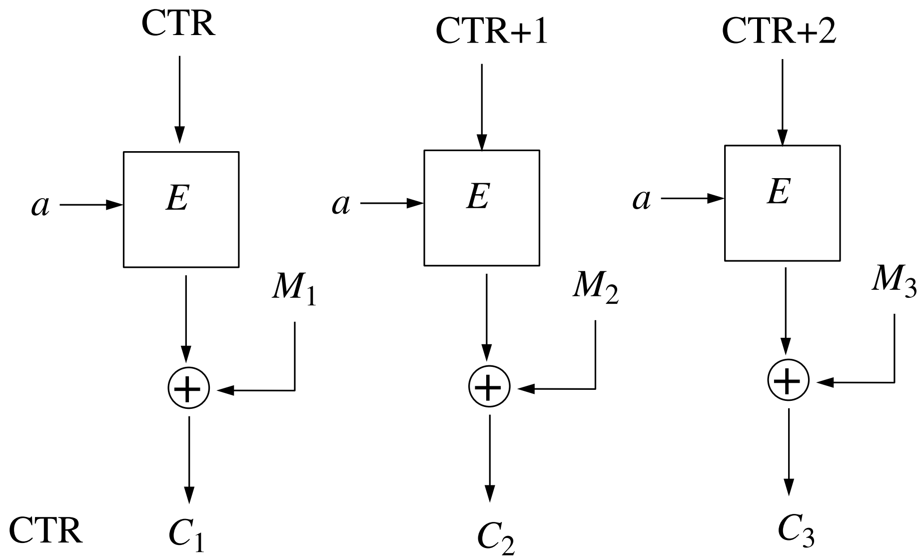

Figure 4: *CTR mode encryption. The value of CTR is increased by* n *following the encryption of an* n*-block message. At most* 2^β *blocks can be encrypted with a block cipher of* β*-bits.*