Revised: 4 Feb 99.

ECS 227 — Modern Cryptography — Winter 99
Lecture 4

Symmetric Encryption
(Part I)

4 February 1999

These notes will be brief. Mostly I will recap the definitions we developed. See [BDJR97] for further information.

1 Odds and Ends

I began this lecture by giving a solution to Homework Problem 2. I also encouraged people to do a conscientious job on the homeworks.

2 Syntax

Before we even attempt to describe when a symmetric encryption scheme is "good," let us just write down what a symmetric encryption scheme *is*. You might suspect that this is just a pair consisting of an encryption algorithm and a decryption algorithm. If so, you've forgotten that the ability to generate a key is an important part of the process, too!

Definition 2.1 A (probabilistic) symmetric encryption scheme Π is a triple of algorithms $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where

- $\mathcal{K}$ is a probabilistic algorithm which takes no input and produces a string ("the key") $a \in \mathsf{Key}$. We write $a \leftarrow \mathcal{K}$ to indicate sampling from that space (running that algorithm).
- $\mathcal{E}$ is a probabilistic algorithm that takes a string $a \in \mathsf{Key}$ and a string $M \in \{0,1\}^*$ and produces a string in $C \in \mathsf{Ciphertext} \cup \{\perp\}$. If presented with a string $M \in \mathsf{Message}$ the algorithm must produce a string $C \in \mathsf{Ciphertext}$; and if presented with a string $M \notin \mathsf{Message}$ the algorithm must return the distinguished symbol $\perp$.
- $\mathcal{D}$ is a deterministic algorithm that takes a key $a \in \mathsf{Key}$ and a string $C \in \{0,1\}^*$. It produces either a string $M \in \mathsf{Message}$, or else the distinguished symbol $\perp$.

We require that for any $a \in \mathsf{Key}$ and any $M \in \mathsf{Message}$, if $C \leftarrow \mathcal{E}_a(M)$ then $\mathcal{D}_a(C) = M$. ▮

The sets Key, $\mathsf{Message}$ and $\mathsf{Ciphertext}$ are part of the definition of $\mathcal{K}$, $\mathcal{E}$, and $\mathcal{D}$; they are intrinsic elements of Π. The set $\mathsf{Message}$ may be finite or infinite, but we always demand that if $M \in \mathsf{Message}$ then $M' \in \mathsf{Message}$ for any M' of the same length as M.

It is important $\mathcal{D}$ should be defined on any string, not just legitimate ciphertexts. We'd hate that the adversary could make $\mathcal{D}$ explode by feeding it an evil string!

Sometimes our encryption schemes will be *stateful* instead of probabilistic. This means that $\mathcal{E}$ has some hidden variable, the "state," which is initialized in some specified manner and which is updated with each application of $\mathcal{E}$. For stateful encryption schemes, we permit that $\mathcal{E}$ outputs $\perp$ even when provided $a \in \mathsf{Key}$ and $M \in \mathsf{Message}$. This might happen when, for example, the encryption scheme has encrypted as much as it is willing to encrypt. The condition saying that it must be possible to decrypt after encryption only applies to "proper" ciphertexts—if $M \in \mathsf{Message}$, $a \in \mathsf{Key}$, and $C \leftarrow \mathcal{E}_a(M)$, then if $C \neq \perp$ we must have that $\mathcal{D}_a(C) = M$.

3 Model

Definitions are hard, sometimes to the point that it is better to think about doing a *model* even before one begins to think about a definition proper. The model describes the adversary's capabilities but is silent about what it means for the adversary to win. In the context of symmetric encryption, a model captures the type of attack you imagine the adversary capable of mounting. In the formal definition, the model may completely vanish, and so it is probably not very useful to be formal about the model.

Here are four natural models —attack scenarios— for symmetric encryption.

(1) In a **known-distribution attack** one assumes that there is a certain distribution, $\mathcal{M}$, on plaintexts. The adversary knows $\mathcal{M}$. The adversary sees samples of ciphertexts where the plaintexts have been drawn according to $\mathcal{M}$. The model amounts to giving the adversary a button, and each time she pushes it a random $M \leftarrow \mathcal{M}$ is selected and the adversary is handed C, a random ciphertext for M.

(2) In a **known-plaintext attack** there is, once again, some distribution $\mathcal{M}$ on plaintexts. But now the adversary sees (M, C) pairs, where the plaintexts are generated by sampling in $\mathcal{M}$. The model amounts to giving the adversary a button, and each time she pushes it a random $M \leftarrow \mathcal{M}$ is selected and the adversary is presented with (M, C), where C is a random ciphertext for M.

(3) In an (adaptive) **chosen-plaintext attack** there is *no* distribution on the inputs. Instead, the adversary is given *complete* control on what is encrypted. The adversary makes up messages $M_1, M_2, \ldots$ and she is provided an encryption for each. The attack is adaptive in the sense that M_i might depend on the earlier ciphertexts the adversary has learned. The model amounts to providing the adversary with an encryption oracle. When presented with a message M the oracle responds with $C \leftarrow \mathcal{E}_a(M)$.

(4) In an (adaptive) **chosen-ciphertext attack** we give the adversary the ability of a chosen-plaintext attack, and also the ability to learn the plaintexts for ciphertexts of her choice. It's like having (perhaps temporarily) access to encryption/decryption equipment (for some particular hidden key). Requests to encipher M_i and decipher C_j might be arbitrarily interleaved. The model amounts to providing the adversary with an encryption oracle and a decryption oracle: the first, on input M, responds with a $C \in \mathcal{E}_a(M)$; the second, on input C, responds with $M = \mathcal{D}_a(C)$.

In coming up with concrete attacks on encryption schemes it is certainly important to have the adversary use the least capabilities possible. But in coming up with schemes one wants to assume the adversary has considerable power. So we tend to favor the strong notions. In addition, assuming at least a chosen-plaintext attack gives rise to a robust theory, which has not been demonstrated with weaker notions.

In what follows, we will assume a chosen-plaintext attack.

4 Definitional Ideas

Now we've described *what* the adversary can do, but we've said absolutely nothing about what she is trying to accomplish. We've described the model but have not yet begun to describe the definition. How can we get at a definition?

Perhaps the first thing that comes to mind is to try and capture the following intuition: that no adversary can recover the key a from using the encryption scheme $\mathcal{E}_a(\cdot)$. But this notion is no good. For example, suppose the encryption of M under 100-bit key a is M itself. The key a is supposed to be 100 random bits. Now the adversary has only a tiny tiny chance to recover the key a (just 2^{-100}), no matter how much work the adversary does and no matter how much enciphered text the adversary obtains. But the scheme is clearly worthless.

A second idea might be this: formalize that the encryption of a string should look like random garbage. We would be able to formalize this notion quite easily, drawing on what we've learned from formalizing pseudorandom generators. But the notion would not be good: it would be *too strong.* Suppose you have a good encryption scheme— the encryption of every message looks random. Now modify the encryption scheme so that the encryption of each message happens to start with the bit "0". The scheme ought still be a good encryption —it is doing a fine job of hiding the plaintext— but it would no longer come close to meeting the definition. Looking random is basically *irrelevant.*

A third idea might be the following: that an adversary can't recover M from $\mathcal{E}_a(M)$. Such intuition can certainly be lifted to a definition, but the problem is that the resulting definition would end up being too weak. Let us see why.

In an encryption scheme in which all we know is that A can't recover M from $\mathcal{E}_a(M)$, there is always the danger that the adversary can still recover a lot of *partial information* about M. For example, maybe the adversary can compute some particular bit of M. And maybe that bit of M is all the adversary cares about! For example, if the adversary knows that the plaintext includes in a certain position either the ASCII encoding of "$500,000" or "$800,000," then knowing a certain bit of plaintext is all that the adversary requires to distinguish the one plaintext from the other.

Partial information can take many different forms. For example, maybe the adversary can't recover M, nor any particular bit of M, but still she can recover the XOR of all of its bits. Once again, this might be exactly the information the adversary is interested in.

We want our encryption schemes not just to hide the plaintext, but to leak *no* partial information about it. This idea —properly formalized— is what we will call *semantic security.* It is one of the notions which we will define.

In this phrase "properly formalized" we hide a lot. One immediate concern is that there seems to be some information about M which is *inherently* leaked by $\mathcal{E}_a(M)$. One such piece of information is the existence of some plaintext M. A second piece of information is length of M. And a third piece of information is an encryption of the message M under the (secret) key a.

Ultimately, we deal with the first two concerns by saying that it is not the function of *encryption* to hide a message's existence or its length. It is not that these goals are not "legitimate" or even desirable; it is simply that we don't consider them to be the domain of techniques which we call by "encryption." We deal with the third objection by saying that $C \leftarrow \mathcal{E}_a(M)$ is really *not* information about M— we regard information about M as being realized by some *deterministic* function f applied to M.

Goldwasser and Micali (1984) first captured the intuition above (in the domain of asymmetric encryption), defining and showing how to achieve semantic security (in an asymptotic sense, under a complexity-theoretic assumption). As they say it, an encryption scheme is semantically secure if "whatever is efficiently computable about the plaintext given the ciphertext is also efficiently computable in the absence of the ciphertext."

5 Four Definitions

We went through four candidate definitions. We will now restate these.

I'm not going to "talk through" any of these definitions, but you should definitely think about them and make sure that they all make good sense to you!

BRAD'S SUGGESTION. The first definition was suggested by Brad Ballinger. He said (more or less) that an encryption scheme could be regarded as secure if an adversary can't find a message M such that she can tell apart an encryption of M from an encryption of a random string of junk (that random string having the same length as M). We could formalize this as follows.

Definition 5.1 Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and let A be an adversary. Then

$$\begin{aligned}\mathrm{Adv}_{\Pi}^{\mathrm{bb}}(A) \stackrel{\mathrm{def}}{=} \; & 2 \cdot \Pr[a \leftarrow \mathcal{K} \,;\, (M, s) \leftarrow A^{\mathcal{E}_a(\cdot)}(\mathsf{find}) \,;\, b \leftarrow \{0,1\} \,; \\ & \text{if } b = 0 \text{ then } C \leftarrow \mathcal{E}_a(M) \text{ else } C \leftarrow \mathcal{E}_a(\$^{|M|}) : A^{\mathcal{E}_a(\cdot)}(\mathsf{guess}, s, C) = b] - 1 \,.\end{aligned}$$ ▮

Sometimes we prefer to talk about security instead of adversarial advantage. There is a simple way to make the connection. We illustrate with an example.

Definition 5.2 Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and let A be an adversary. We say that A $(t, q, \mu; \epsilon)$-attacks$^{\mathrm{bb}}$ Π if A runs in time at most t, asks at most q oracle queries, the sum of the lengths of these queries, plus the length of the string M output in the first stage of A, is at most μ, and $\mathrm{Adv}_{\Pi}^{\mathrm{bb}}(A) \geq \epsilon$.

We write $\mathrm{Adv}_{\Pi}^{\mathrm{bb}}(t, q, \mu)$ for the maximal value of ϵ such that there is an adversary A that $(t, q, \mu; \epsilon)$-attacks$^{\mathrm{bb}}$ Π. (Note the overloaded notation.) ▮

You can do this for any of our definitions. Define some "version" of advantage. Identify the computational resources of interest. Then speak of the advantage that an adversary gets, where the adversary uses at most the specified resources. And speak of the maximal advantage any adversary can obtain, just so long as she is limited to the specified resources.

I am going to try to stick with the attack-centric notation, always naming adversaries and talking about their advantage. I hope that is easier to understand. I am interested in your ideas about what is the easiest-to-understand way to handle this.

FIND-THEN-GUESS. Here's a slightly different idea. We capture the following: that an adversary can't come up with a pair of equal-length messages (M_0, M_1) such that if she gets a random encryption of one of them, she'll know which one it is.

Definition 5.3 Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and let A be an adversary. Then

$$\begin{aligned}\mathrm{Adv}_{\Pi}^{\mathrm{fg}}(A) \stackrel{\mathrm{def}}{=} \; & 2 \cdot \Pr[a \leftarrow \mathcal{K} \,;\, (M_0, M_1, s) \leftarrow A^{\mathcal{E}_a(\cdot)}(\mathsf{find}) \,;\, b \leftarrow \{0,1\} \,; \\ & C \leftarrow \mathcal{E}_a(M_b) : A^{\mathcal{E}_a(\cdot)}(\mathsf{guess}, s, C) = b] - 1 \,.\end{aligned}$$ ▮

It is understood that in the experiment above the adversary must output M_0 and M_1 of equal length. Some convention is established so that every output encodes a pair of equal length strings (and the string s).

When we speak of complexity, let communication complexity, μ, include the length of M_0.

REAL-OR-RANDOM. Here's a different idea. It seems to stray a bit from the modeling of a chosen-plaintext attack, but it will turn out to be irrelevant. We capture the following: that an adversary can't tell if she has a "real" encryption oracle or a "bogus" encryption oracle. The latter, on query M, ignores M (except for its length), flips $|M|$ random coins, and encrypts them.

Definition 5.4 Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and let A be an adversary. Then

$$\mathrm{Adv}_{\Pi}^{\mathrm{rr}}(A) \stackrel{\mathrm{def}}{=} \Pr[a \leftarrow \mathcal{K} : A^{\mathcal{E}_a(\cdot)} = 1] - \Pr[a \leftarrow \mathcal{K} : A^{\mathcal{E}_a(\$^{|\cdot|})} = 1] \ . \quad \blacksquare$$

It wouldn't have mattered if we had encrypted $0^{|\cdot|}$ instead of $\$^{|\cdot|}$.

SEMANTIC SECURITY. Here is a version of semantic security. This is an unpublished definition which builds on our CRYPTO '98 paper on strong notions for asymmetric encryption.

Definition 5.5 Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme and let A be an adversary. Then

$$\begin{aligned}\mathrm{Adv}_{\Pi}^{\mathrm{sem}}(A) \stackrel{\mathrm{def}}{=} \; & \Pr[a \leftarrow \mathcal{K} \, ; (\mathcal{M}, f, s) \leftarrow A^{\mathcal{E}_a(\cdot)}(\mathsf{select})] \, ; M \leftarrow \mathcal{M} \, ; C \leftarrow \mathcal{E}_a(M) : \\ & \quad A^{\mathcal{E}_a(\cdot)}(\mathsf{predict}, C, s) = f(M)] - \\ & \Pr[a \leftarrow \mathcal{K} \, ; (\mathcal{M}, f, s) \leftarrow A^{\mathcal{E}_a(\cdot)}(\mathsf{select})] \, ; M, M' \leftarrow \mathcal{M} \, ; C \leftarrow \mathcal{E}_a(M') : \\ & \quad A^{\mathcal{E}_a(\cdot)}(\mathsf{predict}, C, s) = f(M)] \ . \quad \blacksquare\end{aligned}$$

In the definition above, when the adversary outputs $\mathcal{M}$ that means to output the encoding of a message-space sampling algorithm. We require that the sampling algorithm only output strings of some particular length. (Why?) When the adversary outputs f that means that she outputs the description of a program that computes the function f. We don't distinguish $\mathcal{M}$ as a distribution from $\mathcal{M}$ as an algorithm, nor do we distinguish f as a function from f as a program that computes it.

When we speak of complexity, let time, t, be understood to include the time for computing f (once) and for sampling from M (once). Let communication complexity, μ, include not only the sum of the lengths of the oracle queries, but also the length of (one) sample from $\mathcal{M}$. These conventions make theorem statements simpler.

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

We are now going to start showing some relationships. Here is one of the simplest.

Proposition 6.1 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_{rr} that $(t', q', \mu'; \delta')$-breaks$^{\mathrm{rr}}$ Π, where

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

Proof. We must construct the indicated adversary. First, ignore the bounds. One figures those out from looking at the efficiency of the constructed adversary.

How will the adversary $A_{\rm rr}$ work? Well, we can follow our noses!

Algorithm $A_{\rm rr}^g$:
  Run $A_{\rm fg}^h(\mathsf{find})$
    When $A_{\rm fg}^h$ asks its h-oracle some query x, answer using our own oracle, g.
  Eventually $A_{\rm fg}$ outputs some (M_0, M_1, s)
  Choose $b \leftarrow \{0,1\}$.
  $C \leftarrow g(M_b)$.
  Run $A_{\rm fg}^h(\mathsf{guess}, C, s)$
    When $A_{\rm fg}^h$ asks its h-oracle some query x, answer using our own oracle, g.
  Eventually, $A_{\rm fg}$ outputs a bit, b'.
  If $b = b'$ then output 1 ("real") else output 0 ("random").

It is easy to check that $A_{\rm fg}$ runs in the claimed time bound and has the claimed bit complexity and query complexity. But we have to check that the advantage of $A_{\rm fg}$ is as indicated.

First think about what happens if we run $A_{\rm rr}^g$ where g is a realized by a "real" encryption oracle. Then $A_{\rm rr}$ perfectly simulates for $A_{\rm fg}$ the "native" environment which defines $A_{\rm fg}$'s success:

$$\begin{aligned}\Pr[a \leftarrow \mathcal{K} : A_{\rm rr}^{\mathcal{E}_a(\cdot)}] &= \Pr[a \leftarrow \mathcal{K} \,;\, (M_0, M_1, s) \leftarrow A_{\rm fg}^{\mathcal{E}_a(\cdot)}(\mathsf{find}) \,;\, b \leftarrow \{0,1\} \,;\, C \leftarrow \mathcal{E}_a(M_b) : \\ &\qquad A_{\rm fg}^{\mathcal{E}_a(\cdot)}(\mathsf{guess}, C, s) = b]\end{aligned}$$

Now think about what happens if $A_{\rm rr}^g$ if g is realized by a "random-encrypting" encryption oracle. Then

$$\begin{aligned}\Pr[a \leftarrow \mathcal{K} : A_{\rm rr}^{\mathcal{E}_a(\$^{|\cdot|})}] &= \Pr[a \leftarrow \mathcal{K} \,;\, (M_0, M_1, s) \leftarrow A_{\rm fg}^{\mathcal{E}_a(\$^{|\cdot|})}(\mathsf{find}) \,;\, b \leftarrow \{0,1\} \,;\, C \leftarrow \mathcal{E}_a(\$^{|M_b|}) : \\ &\qquad A_{\rm fg}^{\mathcal{E}_a(\$^{|\cdot|})}(\mathsf{guess}, C, s) = b] \\ &= \frac{1}{2}\,.\end{aligned}$$

Why is this 1/2? Because we've given $A_{\rm fg}$, in her guess stage, absolutely no information about b. It doesn't matter how much time she spends or what strategy she uses, she can't predict a bit about which we've given her no information any better (or worse!) than 1/2.

But then

$$\begin{aligned}\mathrm{Adv}_{A_{\rm rr}}^{\rm rr} &= \Pr[a \leftarrow \mathcal{K} : A_{\rm rr}^{\mathcal{E}_a(\cdot)}] - \Pr[a \leftarrow \mathcal{K} : A_{\rm rr}^{\mathcal{E}_a(\$^{|\cdot|})}] \\ &= \Pr[a \leftarrow \mathcal{K} \,;\, (M_0, M_1, s) \leftarrow A_{\rm fg}^{\mathcal{E}_a(\cdot)}(\mathsf{find}) \,;\, b \leftarrow \{0,1\} \,;\, C \leftarrow \mathcal{E}_a(M_b) : \\ &\qquad A_{\rm fg}^{\mathcal{E}_a(\cdot)}(\mathsf{guess}, C, s) = b] - \frac{1}{2}\end{aligned}$$

And so

$$\begin{aligned}2 \cdot \mathrm{Adv}_{A_{\rm rr}}^{\rm rr} &= 2\Pr[a \leftarrow \mathcal{K} \,;\, (M_0, M_1, s) \leftarrow A_{\rm fg}^{\mathcal{E}_a(\cdot)}(\mathsf{find}) \,;\, b \leftarrow \{0,1\} \,;\, C \leftarrow \mathcal{E}_a(M_b) : \\ &\qquad A_{\rm fg}^{\mathcal{E}_a(\cdot)}(\mathsf{guess}, C, s) = b] - 1 \\ &= \mathrm{Adv}_{A_{\rm fg}}^{\rm fg}\end{aligned}$$

and so $\mathrm{Adv}_{A_{\rm rr}}^{\rm rr} = \frac{1}{2}\mathrm{Adv}_{A_{\rm fg}}^{\rm fg} \geq \delta/2$. ∎

References

Symmetric encryption. The closest treatment is that in:

[**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`