Examples of reductions with interactive games

Noah Stephens-Davidowitz

June 9, 2023

These are supplemental notes in which I provide some examples of security proofs via a reduction for cryptographic primitives whose security is defined in terms of an interactive game. They should (hopefully) be helpful in showing you how to do the homework and in filling in some of the details that I do not have the time to cover in lectures. In particular, the reductions that I typically do in lecture and in the lecture notes tend to be rather complicated. The reductions below are meant to be simple, to just illustrate the basics of how such reductions should look without having to worry about additional subtleties.

For example, the multi-message semantic security game for secret-key encryption is an *interactive* game because it involves back-and-forth communication between the adversary and the challenger. In particular, in this game, the adversary (takes as input 1^n and) sends two lists of plaintexts $m_{1,0}, \ldots, m_{\ell,0}$ and $m_{1,1}, \ldots, m_{\ell,1}$. Then, the challenger samples a secret key $k \leftarrow \mathsf{Gen}(1^n)$, flips a coin $b \sim \{0,1\}$, and encrypts the messages from list b, i.e., sets $c_1 \leftarrow \mathsf{Enc}(k, m_{1,b}), \ldots, c_\ell \leftarrow \mathsf{Enc}(k, m_{\ell,b})$. It then sends the ciphertexts $c_1, \ldots, c_\ell$ to the adversary, and the adversary responds with a bit b'. The adversary wins if $b = b'$.

The basic idea of reductions in this setting is still the same. In order to prove that some construction X is secure under the assumption that Y is secure, we show that an adversary that breaks X could be used to break Y. (Always make sure that you have the direction correct!) However, when we wish to use an adversary $\mathcal{A}$ that is playing one of these interactive games in order to break some other scheme, we can't just write something like $b' \leftarrow \mathcal{A}(c_1, \ldots, c_\ell)$, because $\mathcal{A}$ doesn't just take an input and return an output. Instead, $\mathcal{A}$ *interacts*—it first sends some stuff, then it receives some stuff, and then it sends some stuff, etc. I.e., $\mathcal{A}$ likes to have conversations (the formal terminology is a protocol, as we will see later), in which whatever it says now depends on everything that's been said to it and everything it's said to us (and any coins flipped by $\mathcal{A}$, including coins flipped earlier in the conversation).

We typically describe reductions $\mathcal{B}$ that use such an adversary $\mathcal{A}$ with language like "$\mathcal{B}$ first receives the first message sent by $\mathcal{A}$, consisting of two lists of plaintexts $m_{1,0}, \ldots, m_{\ell,0}$ and $m_{1,1}, \ldots, m_{\ell,1}$. $\mathcal{B}$ then [does some crazy stuff in order to really cleverly choose some ciphertexts $c_1, \ldots, c_\ell$] and sends $c_1, \ldots, c_\ell$ to $\mathcal{A}$, receiving in response $b' \ldots$" In other words, we describe how $\mathcal{B}$ will interact in a conversation with $\mathcal{A}$. If $\mathcal{B}$ is itself playing an interactive game, then we will also describe any messages that $\mathcal{B}$ sends to the challenger. E.g., $\mathcal{B}$ might send the lists $m_{1,0}, \ldots, m_{\ell,0}$ and $m_{1,1}, \ldots, m_{\ell,1}$ to its challenger and use the response from the challenger to choose $c_1, \ldots, c_\ell$. We therefore end up with a conversation between three parties, with $\mathcal{B}$ in the middle.

(If one wants to be super-duper formal and describe such interactive adversaries $\mathcal{A}$ as a plain old algorithm, then one can do so. For example, one can define the adversary in the many-message semantic security game as an algorithm $\mathcal{A}$ that, when it takes as input $(0, 1^n)$ (here, the zero

represents the fact that $\mathcal{A}$ has sent zero messages so far) outputs two lists of plaintexts $m_{1,0}, \ldots, m_{\ell,0}$ and $m_{1,1}, \ldots, m_{\ell,1}$, *AND* a state σ. (Think of σ as the memory of $\mathcal{A}$. I.e., $\mathcal{A}$ might have performed some crazy computations to choose his plaintexts $m_{i,b}$, and σ can record some information about that.) On input $(1, (c_1, \ldots, c_\ell), \sigma)$, $\mathcal{A}$ outputs a bit b'. One can then define multi-message semantic security in terms of an experiment in which this algorithm is called twice, first on input $(0, 1^n)$ and then on input $(1, (c_1, \ldots, c_\ell), \sigma)$, where the c_i are sampled as in the security game. But this is not very intuitive.)

When interactive games are defined in terms of oracles (such as in the security game for signatures), we may view the oracle queries made by $\mathcal{A}$ as messages passed to $\mathcal{B}$. So, we write something like "For each signing query m made by $\mathcal{A}$, $\mathcal{B}$ [does some crazy computation to compute some signature σ, possibly involving interaction with its own oracles], and responds with σ.

Below, I do some examples.

Claim 1. *Let* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a many-message semantically secure* secret-key *encryption scheme, and define* $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ *as follows.*

- $\mathsf{Gen}' = \mathsf{Gen}$.
- $\mathsf{Enc}'(k, m)$*: Set* $c \leftarrow \mathsf{Enc}(k, m)$*, and output* $c|c$*. (I.e., a ciphertext of* Enc' *is just a ciphertext of* Enc *written twice.)*
- $\mathsf{Dec}'(k, c|c) = \mathsf{Dec}(k, c)$.

Then, $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ *is many-message semantically secure.*

Proof. Suppose for contradiction that $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ is not semantically secure, so that there exists an adversary $\mathcal{A}$ and non-negligible $\varepsilon(n)$ such that $\mathcal{A}$ wins the semantic security game against $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ with probability $1/2 + \varepsilon(n)$. Then, we construct an adversary $\mathcal{B}$ that breaks $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ as follows.

$\mathcal{B}$ takes as input 1^n and passes this to $\mathcal{A}$. $\mathcal{A}$ responds with two lists of plaintexts $M_0 := (m_{1,0}, \ldots, m_{\ell,0})$ and $M_1 := (m_{1,1}, \ldots, m_{\ell,1})$. $\mathcal{B}$ simply sends M_0 and M_1 to its challenger, receiving in response $c_1, \ldots, c_\ell$, where $c_i \leftarrow \mathsf{Enc}(k, m_{i,b})$ for $k \leftarrow \mathsf{Gen}(1^n)$ and $b \sim \{0, 1\}$. $\mathcal{B}$ then sends $c_1|c_1, \ldots, c_\ell|c_\ell$ to $\mathcal{A}$, receiving in response some bit b'. Finally, $\mathcal{B}$ simply outputs b'.

Clearly $\mathcal{B}$ is efficient. And, notice that the ciphertexts $c_1|c_1, c_2|c_2, \ldots, c_\ell|c_\ell$ sent to $\mathcal{A}$ in the above reduction are distributed exactly as $\mathsf{Enc}'(k, m_{i,b})$, and that k is distributed exactly as $k \leftarrow \mathsf{Gen}'(1^n)$ (since the two key-generation algorithms are exactly the same). It follows that

$$\Pr[b' = b] = 1/2 + \varepsilon(n) \ .$$

Since $\varepsilon(n)$ is non-negligible by assumption, this is a contradiction, as needed. □

Claim 2. *If* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *is a secure signature scheme where the length of a signature always equals the length of the plaintext, then* $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ *defined as follows is also a secure signature scheme.*

- $\mathsf{Gen}'(1^n)$*:* $(sk, vk) \leftarrow \mathsf{Gen}(1^{n/2})$*. (I.e., we run* Gen*, but with the security parameter divided by two for whatever reason.)*
- $\mathsf{Sign}'(sk, m)$*: Set* $\sigma \leftarrow \mathsf{Sign}(sk, m)$*, output* $\sigma' := m \oplus \sigma$.

- $\mathsf{Ver}'(vk, m, \sigma')$*: Output* $\mathsf{Ver}(vk, m, \sigma' \oplus m)$.

Proof. Suppose that there is an efficient adversary $\mathcal{A}$ that breaks $(\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$. I.e., $\mathcal{A}$ wins the signature security game with probability $\varepsilon(n)$ for some non-negligible $\varepsilon(n)$. We construct an adversary $\mathcal{B}$ that breaks the signature security game against $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ as follows.

$\mathcal{B}$ takes as input 1^n and a verification key vk and sends 1^{2n} and vk to $\mathcal{A}$. Then, each time that $\mathcal{A}$ makes a signing query m_i, $\mathcal{B}$ passes the signing query m_i to its own signing oracle, receiving in response a signature σ_i. $\mathcal{B}$ computes $\sigma_i' := \sigma_i \oplus m$ and sends σ_i' to $\mathcal{A}$. Eventually, $\mathcal{A}$ gets tired of making signing queries and outputs (m', σ'). $\mathcal{B}$ outputs $(m', \sigma' \oplus m')$.

Clearly $\mathcal{B}$ is efficient. We claim that $\mathcal{B}$ wins the signature game against $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ with probability exactly $\varepsilon(2n)$, which is non-negligible. Indeed, notice that (vk, sk) is distributed exactly as the output of $\mathsf{Gen}'(1^{2n})$ and that the signatures σ_i' are distributed exactly as $\mathsf{Sign}'(sk, m_i)$. Therefore, the probability that $\mathsf{Ver}'(vk, m', \sigma') = 1$ and $m' \notin \{m_1, \ldots, m_q\}$ is exactly $\varepsilon(2n)$, which is non-negligible. Finally, notice that $\mathsf{Ver}'(vk, m', \sigma') = \mathsf{Ver}(vk, m', \sigma' \oplus m')$. The result follows. □

Claim 3. *Let* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a semantically secure* public-key *encryption scheme, and define* $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ *as follows.*

- $\mathsf{Gen}' = \mathsf{Gen}$.
- $\mathsf{Enc}'(pk, m)$*: Set* $c \leftarrow \mathsf{Enc}(pk, m)$*, and output* $pk|c$*. (I.e., a ciphertext of* Enc' *is just a ciphertext of* Enc *with the public key attached. Notice that if this were a secret-key encryption scheme, then attaching the whole secret key to the ciphertext would be a bad idea :), but with public-key schemes, this is fine.)*
- $\mathsf{Dec}'(sk, pk|c) = \mathsf{Dec}(sk, c)$.

Then, $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ *is also a semantically secure public-key encryption scheme.*

Proof. Suppose for contradiction that there is an efficient adversary $\mathcal{A}$ that wins the (public-key) semantic security game against $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ with probability $1/2 + \varepsilon(n)$ for non-negligible $\varepsilon(n)$. We construct an adversary $\mathcal{B}$ in the (public-key) semantic security game against $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ as follows.

$\mathcal{B}$ takes as input 1^n and pk, where $(sk, pk) \leftarrow \mathsf{Gen}(1^n)$. It simply passes 1^n and pk to $\mathcal{A}$. $\mathcal{A}$ responds with two plaintexts m_0, m_1. $\mathcal{B}$ simply passes m_0 and m_1 to its challenger, receiving in response $c \leftarrow \mathsf{Enc}(pk, m_b)$ for $b \sim \{0, 1\}$. $\mathcal{B}$ sends $pk|c$ to $\mathcal{A}$, receiving in response a bit b', and $\mathcal{B}$ simply outputs b'.

Clearly, $\mathcal{B}$ is efficient. Furthermore, it is clear that $\Pr[b = b']$ is exactly $1/2 + \varepsilon(n)$ (since the public key pk and the ciphertext $pk|c$ sent to $\mathcal{A}$ are distributed exactly as they are in the semantic security game against $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$). □