Proofs by Reduction

Florian Weber

I wrote the first version of this article in 2019 in response to an exercise sheet with suboptimal results when it came to reductions. This is the 2020 revision that removes some stuff that was specific for 2019 and adds a terse version that is closer to what I would expect from you in homework as well as a note on aborts.

Note that (except for the last two sections) I'm focusing on *explaining* the techniques used for the proofs here and merely use the provided examples to demonstrate aspects of that. A proof that just tries to prove a statement can (and should) be much shorter, similar to the one presented in Section 6.

Also, before we get to anything else, a note on collisions: In cryptography we rarely care about the *existence* of a collision. Collisions *exist* for every non-injective function, what we usually care about in cryptography is how hard it is to *find* them. Please make sure that the wording in your homework reflects that.

1 Games and Security Definitions

When working with reductions, we usually define our security notions as so called **games**. A game (usually) consists of two parties: A *challenger* who, as the name suggests, creates a challenge and an *adversary* or *attacker* who tries to break it. If you don't know where to start with a proof it is always a good idea to make sure that you fully understand the games that you will have to work with.

A very simple example for a game would be this:

- The challenger samples a random bit $b \leftarrow \mathbb{B} := \{0,1\}$ and a random bitstring r of length λ.
- The challenger gives $H(b||r)$ to the adversary.
- The adversary gives a bit b' to the challenger.
- The challenger returns 1 if $b = b'$ and 0 otherwise.

We then define that the adversary wins this game if the challenger returns 1 in the end.

Obviously the job of the adversary is to win whatever game it is being presented. Sometimes (such as here) a certain probability of winning is considered to be trivially achievable ($\frac{1}{2}$ by simpling making a random guess) in which case we demand that attackers have a success probability that is non-negligibly different from that value. We call the absolute value of this value the *adversarial advantage* $\mathsf{Adv}_{\mathcal{A}} =: \left|\Pr[b = b'] - \frac{1}{2}\right|$ (often abreveated as just "advantage") and say that a scheme satisfies a notion X, if there is no attacker that has both a non negligible advantage **and** satisfies the restrictions placed upon attackers.

To justify the reason why we define the adversarial advantage as the absolute value, assume that there is an adversary $\mathcal{A}$ for which the success probability is non negligibly below the trivial value. Then there exists also an adversary $\mathcal{A}'$ that outputs the inverse of $\mathcal{A}$, meaning that the existence of one implies the existence of the other. Also note that the advantage can only be positive in the first place if the trivial success rate is 0, as there are no negative probabilities.

The other point about restrictions on the adversary is also very important: Most of the time an exponential-time adversary can break our schemes with high probability, it is just that we don't necessarily consider them to be valid. Similarly we usually don't consider adversaries with quantum computers to be valid when talking about pre quantum crypto. That doesn't mean that they don't exist though and they should be considered in practice.

(A small ungraded exercise for you: Think about whether the security property that we used as an example in this section is in any correlation to any of those that you already know from the lecture or is in fact independent.)

2 Receiver Indistinguishability

Let $\mathbb{G} =: \langle g \rangle$ be a cyclic group of prime order q for which the DDH assumption holds (i.e. that breaking DDH is hard) and offers λ bits of security.

Let ElGamal refer to the ElGamal encryption scheme with $\mathbb{G}$ as group as presented in the lecture.

ElGamal has countless properties that go beyond it being just a secure encryption scheme (for a certain definition of secure). The one we are going to prove here, is receiver indistinguishability (RI). Intuitively it means that an attacker should be unable to learn any information about the intended receiver just from looking at the ciphertext, *even if he himself chooses the plaintext that is encrypted.*

This property is not self-evident: Consider any version of RSA with a keylength of λ bits and let n_0 and n_1 be different public keys with $\frac{n_1 - n_0}{n_1}$ being non negligible. Then there is a non negligible chance that encrypting a random value under n_1 results in a value c such that $c > n_0$. This means however, that c is not a ciphertext under n_0, thus even a passive attacker can learn information about who the receiver might be.

There are ways around this even with RSA, but ElGamal with a standardised group that is shared by all users has that kind of protection from the very start. Intuitively this is because the first element of an ElGamal ciphertext is truly random, and the second one is the product of a known plaintext and something that is indistinguishable from randomness, meaning that from an attacker's view, given the security argument behind one-time pads, the second element is essentially random as well. But two random elements are obviously not related to any public-key, thus ElGamal offers receiver indistinguishability.

Of course the above explanation is not a formal security argument and the above „definition" of receiver indistinguishability also leaves a lot of precision to be desired; I provide them here not as an example for how to write a proof, but as an aid for understanding what is going on in the actual proof.

For our purposes we define the receiver-indistinguishability game as follows:

- The challenger samples a random bit b.
- The challenger generates to keypairs $(\mathsf{pk}_0,\ \mathsf{sk}_0)$ and $(\mathsf{pk}_1,\ \mathsf{sk}_1)$.
- The challenger gives pk_0 and pk_1 to the adversary.
- The adversary gives a plaintext m to the challenger.
- The challenger encrypts m with the public-key pk_b, giving the ciphertext c.
- The challenger gives c to the adversary.
- The adversary gives a bit b' to the challenger.
- The challenger returns 1 if $b = b'$ and 0 otherwise.

We say that the adversary wins the game if the challenger outputs 1.

We say that a scheme offers receiver indistinguishability if there is no probabilistic polynomial-time (non quantum) adversary that wins with a probability that is non negligibly different than $\frac{1}{2}$.

Much more formal:

$$\forall \mathcal{A} \in \mathrm{PPT}, \lambda \in \mathbb{N}:$$
$$\left| \Pr\left[b = b' \;\middle|\; \begin{array}{l} b \xleftarrow{\$} \mathbb{B} \\ \mathsf{pk}_0,\ \mathsf{sk}_0 := \mathsf{Gen}(1^\lambda) \\ \mathsf{pk}_1,\ \mathsf{sk}_1 := \mathsf{Gen}(1^\lambda) \\ m,\ s := \mathcal{A}(\mathsf{pk}_0, \mathsf{pk}_1) \\ c := \mathsf{Enc}(\mathsf{pk}_b, m) \\ b' := \mathcal{A}(s, c) \end{array} \right] - \frac{1}{2} \right| \leq \mathrm{negl}(\lambda)$$

Here PPT means the set of all probabilistic polynomial-time Turing machines, s refers to arbitrary state that the adversary $\mathcal{A}$ is allowed to retain between its invocations and negl is a negligible function.

3 DDH

For the purposes of this text we use the following definition of DDH:

- The challenger samples a random bit b.
- The challenger samples random values x, y, z from $\mathbb{Z}/q\mathbb{Z}$.
- If $b = 0$, the challenger gives $g^x,\ g^y,\ g^z$ to the adversary, otherwise $g^x,\ g^y,\ g^{xy}$.
- The adversary gives a bit b' to the challenger.
- The challenger returns 1 if $b = b'$ and 0 otherwise.

We say that the adversary wins the game if the challenger outputs 1.

We say that the DDH assumption holds in $\mathbb{G}$, if there is no probabilistic polynomial time (non quantum) adversary that wins with a probability that is non negligibly different from $\frac{1}{2}$.

Much more formal:

$$\forall \mathcal{A} \in \mathrm{PPT}, \lambda \in \mathbb{N}:$$
$$\left| \Pr\left[b = b' \;\middle|\; \begin{array}{l} b \xleftarrow{\$} \mathbb{B} \\ x,\ y,\ z_0 \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z} \\ z_1 := x \cdot y \\ b' := \mathcal{A}(g^x, g^y, g^{z_b}) \end{array} \right] - \frac{1}{2} \right| \leq \mathrm{negl}(\lambda)$$

4 The Actual Reduction

Using this, we can now finally look at the actual reduction. To prove that ElGamal offers receiver indistinguishability under the DDH assumption, we need to prove that we can turn every attacker against the receiver indistinguishability into an attacker against the DDH assumption.

The important part here is that we **must use the interfaces as provided**. To perform the reduction we need to take on the role of a translator between these two problems. In the one direction we need to take the role of an DDH adversary that interfaces with a DDH challenger. In the other direction we need to take on the role of a receiver-indistinguishability challenger. For reasons that would lead to far at this point[1] the translator is usually called a "simulator" which I will use from here onwards.

Our approach will now work as follows (see also figure 1):

- Let $g^x,\ g^y,\ g^z$ be the values that we receive from the DDH challenger.
- We sample a bit b at random.
- We sample an exponent r from $\mathbb{Z}/q\mathbb{Z}$.
- If $b = 0$ then we define $U := g^x$ and $V := g^r$.

  Otherwise ($b = 1$) we define $U := g^r$ and $V := g^x$.
- We give U and V to the RI attacker.
- The RI attacker gives us a message m.
- We give $g^y,\ g^z \cdot m$ to the RI attacker

[1] The choice of this term becomes clearer in the context of so called simulation-based security notions, which have many great advantages but also quickly get very messy and are therefore not covered in this lecture.

- The RI attacker gives us a bit b'.
- If $b = b'$, we give 1 to the DDH challenger, otherwise 0.

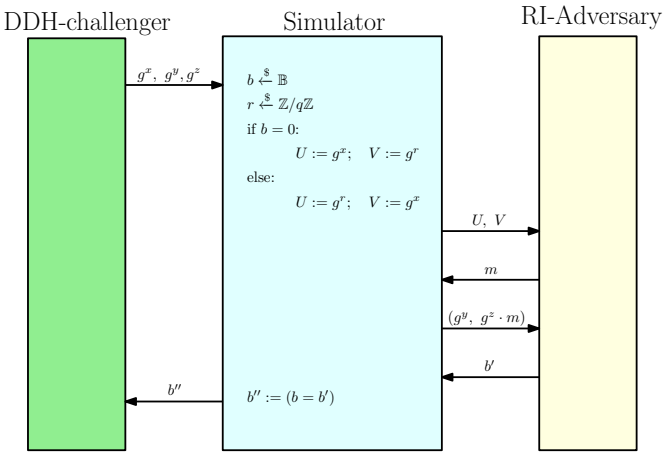

Figure 1: Graphical representation of our simulator.

Note that we always gave the exact kind of input that they were expecting to all parties. An important trick for this was, that we sampled the second public-key ourselves: Since we were only provided one useful value by the DDH challenger (as we need the other two values later), but the RI adversary needed two, we had to generate that value ourselves. This is an important technique that is applicable very often.

We now need to analyze why our replacement (of the RI-Challenger with the simulator and the DDH-challenger) works. For this we have to consider two cases: The one in which $x \cdot y = z$ and the one in which it does not.

In the first case we simulate the RI game perfectly (jargon for: indistinguishable even for adversaries with infinite computational power): U and V are both random group elements that are sampled from the same distribution as they would be if they were both generated using the key-generation algorithm. Thus the inputs follow the expected distribution.

Since y is random by the definition of the DDH game and $z = x \cdot y$ by the assumption of this case, $(g^y,\ g^z \cdot m)$ is not just a valid encryption of m under g^x but also is also perfectly indistinguishable from the one that would be generated by a real RI-challenger. Given that we assigned the public keys so that g^x is in a random position (first or second), it is furthermore encrypted under a randomly selected public-key. (Again: The keys themselves are drawn from the same distribution so the order in which we arrange them doesn't matter.) This means that the second reply to the adversary also follows the expected distribution.

Note that the randomization of the position of the actual public-key is **important**: If you wouldn't do it, adversaries could win by always outputting 0 or 1 respectively. In other context different things could happen. Imagine for example that you have to break a property of a certain value but the adversary wants multiple such values. If you don't put the value into a random position, the adversary might always break the first/second/third/... value, but never the one that you want it to break.

Finally the RI adversary will output a bit b'. Let the probability that this bit is equal to b, be $\frac{1}{2} + \epsilon$. Note now that by the definition of this case, we win in the DDH game, if we output 1 to the DDH challenger. This is the case if $b = b'$. Since that is the case with probability $\frac{1}{2} + \epsilon$ we win with the same probability in this case.

Otherwise, if z was sampled at random, the situation changes: The first message that we send to the RI challenger is still indistinguishable from a real one for the very same reason as before. Now the second part of the ciphertext is however the result of multiplying m with a truly random and independent group element, resulting in a truly random group element over all. This means that it is not an encryption of m under either key. (Technically it could be, but the probability for each key is $\frac{1}{q}$, which is negligible and furthermore cancels out.) This however means that the ciphertextas a whole is in no correlation to either of the public-keys. Given that whatever bit b' the RI adversary outputs, it is independent of b. Since b is truly random, this means that the event $b = b'$ is random as well.

This means that in this case we end up outputting 0 and 1 with both probability $\frac{1}{2}$ to the DDH challenger. Since we, by definition of the case, only win the DDH game if we give 0 to the DDH challenger, this means that we win the DDH game with probability $\frac{1}{2}$ in this case.

Since either of these cases happens with probability $\frac{1}{2}$, we can compute our overall probability of winning the DDH game by simply computing the average of our success probabilities:

$$\frac{1}{2}\left(\left(\frac{1}{2} + \epsilon\right) + \frac{1}{2}\right) = \frac{1}{2} + \frac{\epsilon}{2}$$

Thus: If there is an attacker that can break the receiver-indistinguishability of ElGamal with probability $\frac{1}{2} + \epsilon$, then there is also an attacker against the DDH assumption with success probability $\frac{1}{2} + \frac{\epsilon}{2}$.

However: If ϵ is non negligible, than neither is $\frac{\epsilon}{2}$. Furthermore: If the RI adversary runs in classical probabilistic polynomial time, then so does our derived DDH attacker (technically we have to prove this, but most of the time (including here), it is so obvious that in practice nobody cares if you just claim that it is the case).

Therefore: If there is an efficient attacker with high success probability against the receiver-indistinguishability of ElGamal (statement A), then there is also an efficient attacker with high success probability against the DDH assumption (statement B).

AKA: $A \implies B$

This implies $\neg B \implies \neg A$. In other words:

If there is no efficient attacker [...] against the DDH assumption, then there is also no efficient attacker [...] against the receiver-indistinguishability of ElGamal.

Therefore: The DDH assumption implies the receiver-indistinguishability of ElGamal.

We also say that the problem of breaking the DDH assumption can be reduced to the problem of breaking the receiver-indistinguishability, since breaking the later implies breaking both of them.

5 Compact proof

(The following is how I would write this proof if I were solving it as an exercise.)

We define our simulator $\mathcal{S}$ as follows:

- Let $g^x,\ g^y,\ g^z$ be the values that we receive from the DDH challenger $\mathcal{C}$.
- Sample $b \leftarrow \mathbb{B}$, $r \leftarrow \mathbb{Z}/q\mathbb{Z}$.
- If $b = 0$: Let $U := g^x$ and $V := g^r$.

  If $b = 1$: Let $U := g^r$ and $V := g^x$.
- Give U and V to the RI attacker $\mathcal{A}$.
- Let m be the output of $\mathcal{A}$.
- Give $(g^y,\ g^z \cdot m) =: c$ to $\mathcal{A}$
- Let b' be the output of $\mathcal{A}$.
- If $b = b'$, give 1 to $\mathcal{C}$, otherwise 0.

(Always describe your simulator!)

U and V are truly random group elements and thus distributed exactly like the public keys in the real RI-game, therefore this part of the simulation is sound.

Following that there are two cases:

- $xy = z$: In this case c is distributed exactly like the ciphertext in the real RI-game. The simulation is therefore perfect and $\mathcal{S}$ wins exactly then when $\mathcal{A}$ wins.
- $xy \neq z$: In this case c consists only of random group elements that are independent of b. $\mathcal{A}$ can therefore not do better than guessing a random bit that it knows nothing about, giving it a success probability of exactly $\frac{1}{2}$.

Let $\frac{1}{2} + \epsilon$ be the success probability of $\mathcal{A}$ in the RI game. Then we find that $\mathcal{S}$ has a sucess probability of $\frac{1}{2}\left(\left(\frac{1}{2} + \epsilon\right) + \frac{1}{2}\right) = \frac{1}{2} + \frac{\epsilon}{2}$ with essentially the same runtime. Therefore any efficient adversary against the RI-security of ElGamal can be converted into an efficient adversary against the DDH assumption of the underlying group and we find that ElGamal is RI-secure if the used group offers DDH-security.

(You can do this part slightly less wordy, but always explain why your simulator works and why it wins!)

6 Aborts

All of the above assumes that the adversary does not abort execution or output clearly nonsensical values (such as the string "hello world" when the game demands an integer). This assumption is dangerous and there are protocols that can break completely because of something like this. For the notions that we deal with in this lecture and the games that are associated with them there are however simple workarounds such as having the simulator sample b' in the previous proof if the adversary misbehaves.

As such it is perfectly fine if you don't consider aborts in this lecture (unless explicitly asked), but please remember that they are a thing and that they have to be considered if you want to use proofs in practice.