Guided Solution - Handout 4, Exercise 1

Boaz Barak

October 18, 2005

Exercise 1 (20 points)**.** Consider the following variant of CMA-security for MACs: instead of giving the adversary black boxes for both the signing and verification algorithms, give it only a black box for the signing algorithm. Let's call this definition CMA'-security. That is,

> **Definition 1.** A pair of algorithms $(\mathsf{Sign}, \mathsf{Ver})$ (with $\mathsf{Sign} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^t$, $\mathsf{Ver} : \{0,1\}^n \times \{0,1\}^m \times \{0,1\}^t \to \{0,1\}$) is a (T, ϵ)-CMA'-secure MAC if for every x, k, $\mathsf{Ver}_k(x, \mathsf{Sign}_k(x)) = 1$ andf or every T-time Adv, if we run the following experiment:
>
> - Choose $k \leftarrow_{\mathrm{R}} \{0,1\}^n$
> - Give adversary access to black box for $\mathsf{Sign}_k(\cdot)$
> - Adversary *wins* if it comes up with a pair $\langle x', s' \rangle$ such that **(a)** x' is *not* one of the messages that the adversary gave to the black box $\mathsf{Sign}_k(\cdot)$ and **(b)** $\mathsf{Ver}_k(x', s') = 1$.
>
> Then the probability Adv wins is at most ϵ.
>
> $(\mathsf{Sign}, \mathsf{Ver})$ is CMA'-secure if there are super-polynomial functions T, ϵ such that for every n, $(\mathsf{Sign}, \mathsf{Ver})$ is $(T(n), \epsilon(n))$-CMA'-secure. In other words, there is no polynomial-time Adv that succeeds with polynomial probability to break it.

A MAC scheme has *unique tags* if for every message there is only one tag that passes verification. An equivalent way of stating this property is that the verification algorithm on input x and t outputs 1 if and only if $S_k(x) = 1$. Note that the MAC scheme we saw in class has this property. Prove that for MACs with unique tags, CMA security and CMA' security are *equivalent* (e.g., such a scheme is (T, ϵ)-CMA secure if and only if it is (T', ϵ')-CMA' secure for some T',ϵ' polynomially related to T, ϵ. (The condition of unique tags is important — if a MAC scheme does *not* have unique tags then these notions may not be equivalent.)

If you think about it for a while, it's sort of obvious that verification box should not help to break the MAC. In this case the obvious intuition is right (assuming we have unique signatures). However, there are many things in crypto that seem obvious but turn out to be false (for example, it was "obvious" that an encryption should solve the login problem). Therefore, the way to check our intuitions is to try to translate them into formal proofs. Once you get used to it, this translation is actually often not very hard. This is the case here and so this question is an excellent example for how we go about transforming an intuition into a proof.

1 General form of the question.

First, note that the question we're dealing with here is of the following general form: Suppose that S is a scheme that satisfies security definition D. Now let S' be some scheme that depends on S. Prove that S' satisfies security definition D'.

We deal with questions of this form all the time in crypto. In this case both S and S' are the same scheme $(\mathsf{Sign}, \mathsf{Ver})$ when D is the CMA' definition of security (signing-box only) and D' is the standard CMA definition of security (signing and verification boxes).

Another case is the one we saw in class where S was a one-way permutation f and the hard-core bit h (and D was the security definition of a hard-core bit) while S' was the function $G(x) = f(x), h(x)$ and D' was the definition of a pseudorandom generator.

Whenever we have a question of this general form, the statement we need to prove will be the following:

> Let A' be an adversary of size T that breaks the scheme S' (where "breaking" is defined according to the definition D') with probability at least ϵ. Then, there is an adversary A of size polynomial in T and n that breaks the original scheme S (where breaking is defined according to the definition D) with probability at least polynomial in ϵ, T and n.

This will imply that if S was (T, ϵ)-secure according to D for some super-polynomial T, ϵ then S' will be (T', ϵ')-secure according to D' for some super-polynomial T', ϵ'. However, it's much easier to think of this in the other direction: make A not much slower than A' and with success probability not much worse than A' and you're done.

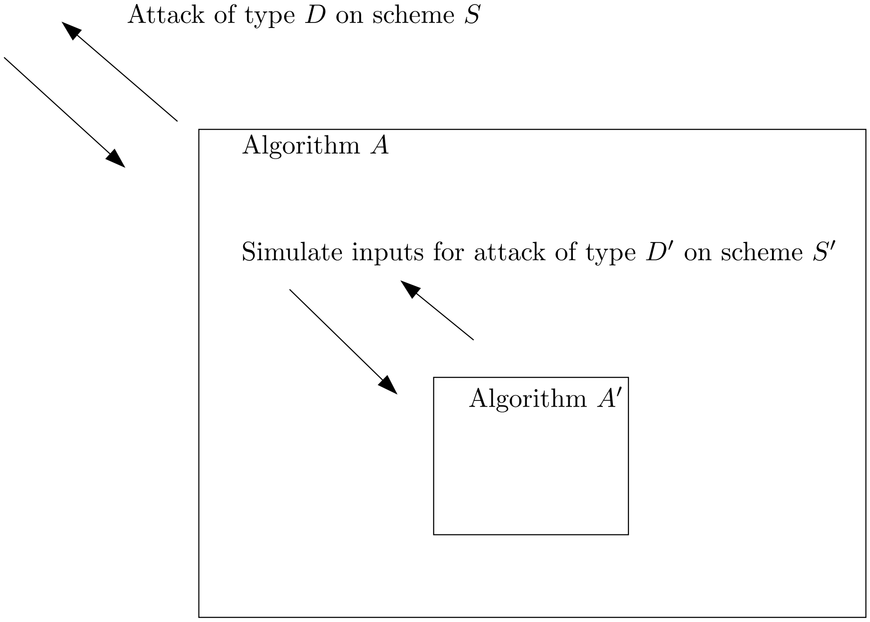

Attack of type D on scheme S
Algorithm A
Simulate inputs for attack of type D' on scheme S'
Algorithm A'

Figure 1: General form of Algorithm A based on Algorithm A'.

How do we come up with A? To prove such a statement, we assume that we're given some A' breaking the scheme S' and we need to come up with A. Again, even before we go into the specifics of the question, it is clear what the general form of A has to be: it will need to use A' in some sort, so will A will be an algorithm that has A' in "its belly' and runs A' on various inputs. It's also clear that if we want to use what we know about the success of A' in breaking S', then our inputs must be the same (or at least indistinguishable) to the inputs that A sees when attacking the scheme S'. This general form is depicted in Figure 1.

2 The specific case of this question.

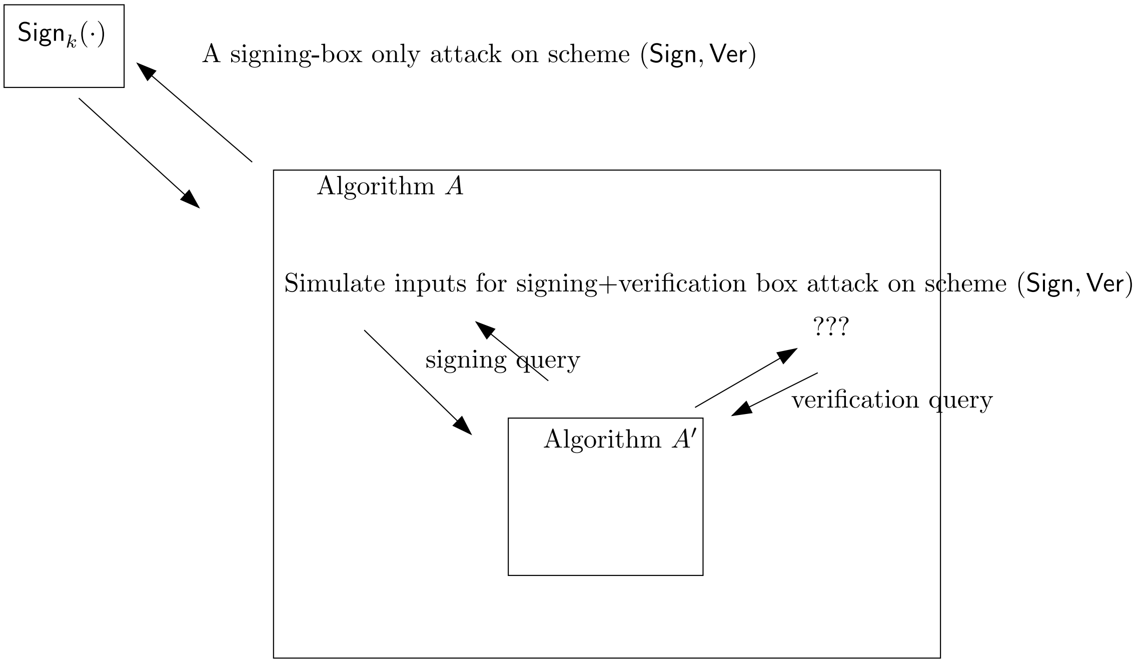
In this case, we are given a scheme $(\mathsf{Sign}, \mathsf{Ver})$ that we know two things about:

- It has the unique signatures property: that is, for every x and k there is at most a single t such that $\mathsf{Ver}_k(x,t) = 1$.
- It is secure against an attacker A with only access to a signing box.

We want to prove that $(\mathsf{Sign}, \mathsf{Ver})$ is also secure against an attacker A' with access to both a signing box and a verification box.

As in the general form, to prove something like that, we'll assume that we have a T-time A that succeeds in breaking $(\mathsf{Sign}, \mathsf{Ver})$ with probability at least ϵ when given access to $\mathsf{Sign}_k(\cdot)$ and $\mathsf{Ver}_k(\cdot)$ for a random $k \leftarrow_{\text{R}} \{0,1\}^n$. We'll try to construct A' that gets only access to $\mathsf{Sign}_k(\cdot)$ but still breaks the scheme with probability related to ϵ and is not much slower than A.

Let's try to construct such an algorithm A. Algorithm A will run A' in its belly and will need to simulate for A' a signing+verification attack on $(\mathsf{Sign}, \mathsf{Ver})$. It is easy to simulate signing queries that A' makes since A gets access to a signing box. The only question (which we'll resolve shortly) is how to simulate verification queries for A. The form of Algorithm A for our specific case is depicted in Figure 2.



$\mathsf{Sign}_k(\cdot)$
A signing-box only attack on scheme $(\mathsf{Sign}, \mathsf{Ver})$
Algorithm A
Simulate inputs for signing+verification box attack on scheme $(\mathsf{Sign}, \mathsf{Ver})$
???
signing query
verification query
Algorithm A'

Figure 2: Specific form of Algorithm A in our case.

Handling verification queries. It is clear that if we managed to simulate for A' perfectly a signing+verification attack on $(\mathsf{Sign}, \mathsf{Ver})$ then by having A output the output of A' we'll succeed with the same probability ϵ. The only question that remains is how do we answer verification-queries that A' makes. For this it seems reasonable to try to use what we have not yet used before: that $(\mathsf{Sign}, \mathsf{Ver})$ (like our PRF-based scheme) has the unique signatures property.

The unique signature property means that for every x and k there exist at most a single t such that $\mathsf{Ver}_k(x,t) = 1$. On the other hand by the validity condition for MACs (that for all k, x $\mathsf{Ver}_K(x, \mathsf{Sign}_k(x)) = 1$) we know that there also must exist at least one t such that $\mathsf{Ver}_k(x,t) = 1$ and this is $t = \mathsf{Sign}_k(x)$.

This means that the verification algorithm can be described in the following way: given x and t, output 1 if and only if $t = \mathsf{Sign}_k(x)$. However, this description immediately shows us how we can simulate a verification query using a signing box: if A' gives out a verification query (x,t) then A will query x to its signing box to obtain $t' = \mathsf{Sign}_k(x)$ and then we'll return 1 to A if and only if $t = t'$.

This completes the description of A. Since A' gets in this execution exactly the same responses it gets in a CMA-attack on $(\mathsf{Sign}, \mathsf{Ver})$, we get that A outputs a successful forgery with the same probability as A' (i.e. ϵ). Since A runs in roughly the same time as A' this means we're done.