CS195-7: Introduction to Cryptography September 11, 2002

Solution Set 1

Instructor: Anna Lysyanskaya

Problem 1

In Lecture 1, we gave a definition of a perfectly secure cryptosystem, and saw some limitations arising while using it. In this problem, we will look at perfectly secure message authentication schemes, and show their limitations.

Suppose we are given the following toy scenario: Alice and Bob are communicating over a channel controlled by Eve. Eve may delete or alter messages sent over the channel, and may inject messages of her own. Our goal is to program Alice and Bob in such a way that Bob will only accept a message if it indeed came from Alice.

To that end, we want to design algorithms (G, A, V) for message space M such that

1. Algorithm G is a randomized setup procedure that generates the public parameters P of the system, as well as private inputs to each party, denoted s_{name}, where "name" is the name of the player to which this input was given; for example s_{Alice}.
2. Algorithm A is a procedure that computes an *authentication code* on a message $m \in M$: $c = A(m, \text{"Alice"}, \text{"Bob"}, P, s_{Alice})$.
3. Algorithm V checks that the authentication code c corresponds to message m: for all $m \in M$, for all s_{Alice}, for all values of the public parameters P, if $c = A(m, \text{"Alice"}, \text{"Bob"}, P, s_{Alice})$, then $V(m, c, Alice, Bob, s_{Bob}) =$ "Accept".
4. ϵ-security for n uses: Even after Eve sees the authentication codes for at most n messages *of its own choice*, the probability that she can produce (using any amount of time and space) a pair (m, c), where m is a new message, and c is an authentication code for m, is bounded by ϵ. To make any sense, this is only defined when $n < |M|$, where M is the size of the message space.

(a) Show that $\epsilon = 0$ cannot be achieved for any $n \geq 0$, for any non-empty message space.

Solution:

Let m be a message for which Alice has not issued an authentication code. Suppose that there exists a string c such that $V(m, c, Alice, Bob, s_{Bob}) =$ "Accept". Consider any probability distribution that assigns to each binary string a non-zero probability. (For example, it outputs a string of length u with probability 2^{-u}, and the distribution given that the length is u is uniform.) If Eve outputs a string according to this distribution, this string will equal c with non-zero probability.

(b) Suppose that all messages in M can be represented as ℓ-bit strings. Consider the following authentication mechanism: the initialization procedure G picks an ℓ-bit shared secret s and gives s to Alice and to Bob. $A(m, Alice, Bob, s) = m \oplus s$. For what values of n and ϵ is this authentication mechanism secure? (In other words, how many messages can Alice send before Eve can authenticate a message on her own?)

Solution:

Suppose that the distribution from which s is chosen is uniform over strings of length ℓ. When $n = 0$, this authentication mechanism is secure, with $\epsilon = 2^{-\ell}$, for the following reason. Suppose Eve outputs (m, c). Since she outputs it without seeing authentications for any other messages, we can imagine that the common shared string s is picked after Eve has output (m, c). Therefore, the event that Eve guesses correctly is the same as the event that $s = m \oplus c$ was picked for the common shared string. This event has the probability $2^{-\ell}$.

Otherwise, if nothing is given about the distribution from which s is selected, we cannot claim any security even for the (useless) case of $n = 0$. This is because if, for example, s is always the same string, then Eve can compute the authentication code the same as Alice.

When $n \geq 1$, this scheme is insecure. From observing just one pair (m, c), Eve computes $s = m \oplus c$, and can forge authentication codes for all other messages.

(c) Suppose that all messages in M can be represented as ℓ-bit strings. Consider the following authentication mechanism:

The initialization procedure G picks an $\ell + 1$-bit prime number p, and two random integers a and b, $0 \leq a, b < p$. The pair of values a and b are given to Alice and Bob, that is their shared secret. The value p is a public parameter.

An authentication code on message m is $c = am + b \bmod p$, where m is treated as an ℓ-bit integer (and so $m < p$).

For what values of n and ϵ is this authentication mechanism secure?

Solution:

We will show that for $n = 1$, this authentication mechanism is secure, while for $n \geq 2$, it is completely broken.

Suppose $n \geq 2$. Then Eve is given (m, c) and (m', c'), where c is the authentication code for m, while c' is the authentication code for m'. Then Eve derives a and b by solving a system of linear equations modulo p: $c = am + b$, $c' = am' + b$.

Suppose $n = 1$. Suppose Eve is given (m, c) where c is the authentication code for m. Suppose Eve then outputs (m', c'), where $m' \neq m$. Since Eve outputs it without seeing anything except the fact that $c = am + b \bmod p$, we can think of a and b as chosen as follows: First, $0 \leq c < p$ is chosen at random. Then, when Eve requests message m authenticated, (m, c) is output. Note that c is distributed correctly, because a uniform (over integers in $[0, p-1]$) distribution on a and b induces a uniform distribution on c. Then, after Eve's output, c'' is chosen uniformly at random from integers in $[0, p-1]$, and a and b are solved for using linear equations modulo p: $c = am + b$, $c' = am' + b$.

a and b are chosen correctly, namely, the probability that a given (a, b) pair was chosen is exactly $1/p^2$. However, the event that Eve guesses correctly is the event that $c' = c''$, and since c'' is chosen at random *after* c'' was fixed, the probability of this event is $2^{-\ell}$.

(d) Show that if $\epsilon = 2^{-k}$ and an authentication mechanism is ϵ-secure for n uses, then s_{Alice} and s_{Bob} must both be at least $k(n+1)$ bits long.

Solution:

We will show this by induction on n. The base case, $n = 0$, is as follows: suppose the value s_{Alice} or s_{Bob} is fewer than $k - 1$ bits. Then Eve can guess it with probability $2^{-(k-1)} > 2^{-k} = \epsilon$, and compute an authentication code for any message correctly.

Let us therefore make the induction hypothesis that kn bits are required for $(n-1)$-security with $\epsilon - 2^{-k}$.

Now suppose we have an n-secure system (G, A, V) that uses secrets with L bits. Let us derive an $(n-1)$-secure system (G', A', V'). This system will be on a smaller message space that misses one of the messages, namely the message m that has the lowest value lexicographically. Our new system will use a secrets of $L - k$ bits.

The setup algorithm G' runs G and generates a shared string of L bits. It then publishes the value c which is the authentication code on the special message m. It lists, in lexicographic order, all the possible secrets s_{Alice} that could cause the authentication code on m to be c. Call this array S_A.

It also lists all the possible secrets s_{Bob} that could cause Bob to accept c as the authentication code on m. Call this array S_B.

There are at most 2^{L-k} possibilities in the array S_A, since otherwise, c would have been easier to guess. Namely, we know that there are 2^L possible values for s_{Alice}. Suppose that more than 2^{L-k} of them resulted in the same value c being an authentication for m. Then the probability, over the choice of s_{Alice}, that c is the authentication code for m, is greater than 2^{-k}, since at least $2^{L-k} + 1$ out of 2^L strings result cause c to be the authentication code for m. Therefore, if this were the case, then (G, A, V) would not be secure.

Similarly, there are at most 2^{L-k} possibilities in the array S_B.

The string s_{Alice} must be in the array S_A. G' computes the index i_{Alice} of s_{Alice} on this list S_A (i.e., the value i such that $s_{Alice} = S_A[i]$). Since there are at most 2^{L-k} strings in the array, the value i_{Alice} is at most $L - k$ bits long. This value i_{Alice} is then sent to Alice as her secret.

On input i_{Alice}, using the public value c, Alice reconstructs the list S_A and sets $s_{Alice} = S_A[i_{Alice}]$. Similarly, on input i_{Bob}, using the public value c, Bob reconstructs the list S_B and sets $s_{Bob} = S_A[i_{Bob}]$.

In order to authenticate and verify any message $m' \neq m$, they then use the procedures A and V, respectively.

Therefore, if for any n, fewer than $k(n+1)$ bits were sufficient, then for $n-1$, kn bits would have been sufficient, violating the induction hypothesis.