Lecture 7 - CCA security

Boaz Barak

February 24, 2010

Reading BS Section 9.3, KL Section 3.7 (pages 103–104), Chapter 4 (although we follow a somewhat different order than both books).

The online shopping problem Let's abstract the online shopping problem as follows: Alice and Bob share a key k, and Alice also has a credit card number c which Bob can validate against a database. (For simplicity, assume that there is only one valid credit card, and so the validation is simply checking that the card equals c.)

Consider the following protocol for an online transaction:

Protocol 1

Alice		Bob
(k, c)		(k, c)
	$\underline{y = \mathsf{E}_k(c)} \longrightarrow$	
		Check $\mathsf{D}_k(y) = c$
	Either:	
	$\underline{\longleftarrow QK}$	
	or	
	$\underline{\longleftarrow GTH}$	

Security What does it mean for this protocol to be secure. A security definition always has three components:

- The capabilities of the adversary: we'll always model it as a polynomial-time algorithm.
- The attack model: in this case we'll start with a passive attack - Eve just sits on the line and listens to polynomially many executions of Protocol 1 (all using the same key and credit card).
- The definition of a break / "win" of the adversary: We'll make the following definition: Assume that $k \leftarrow_R \{0,1\}^n$ and $c \leftarrow_R [10^8]$. We say

that the protocol is secure if for every poly-time Eve, poly-bounded ϵ and large enough n,

$$\Pr[\text{Eve guesses } c\ ] \leq 10^{-8} + \epsilon(n)$$

We say that a protocol satisfying the above definition is a *passively secure authentication protocol*

Theorem 1 If (E, D) is CPA-secure then Protocol 1 is a passively secure authentication protocol.

Proof We'll do the proof by reduction: assume that A can break the passive security of Protocol 1, we'll build an adversary B that can break the CPA security of the encryption scheme.

In an actual attack, A sees T repetitions of the following messages:

$$\overrightarrow{\mathsf{E}_k(c)}$$
$$\overleftarrow{OK}$$

(of course each occurrence of $\mathsf{E}_k(c)$ uses fresh randomness)

Consider the following "mental experiment" we run the attack of A, but instead choose a junk message $J \leftarrow_{\text{R}} \{0,1\}^n$ (e.g., $J = 0^n$) and send an encryption of J instead of c. That is, A will see T repetitions of:

$$\overrightarrow{\mathsf{E}_k(J)}$$
$$\overleftarrow{OK}$$

Clearly in this case A gets no information on c and so will guess it successfully with probability at most 10^{-8}. Hence under our assumption A can distinguish between the two cases with advantage ϵ. Using the hybrid method it follows that there is an $i \in [T]$ such that A can distinguish between the case that the first i messages are encryptions of c and the last $T - i$ messages are encryption of J and the case that the first $i + 1$ messages are encryptions of c and the last $T - i - 1$ messages are encryptions of J. Specifically, if we let p be the probability that A guesses correctly in the first case then in the second case A guesses correctly with probability at least $p + \epsilon/T$.

We can now mount a CPA attack against the encryption scheme. B will do the following:

1. Choose a number $c \leftarrow_{\text{R}} [10^8]$, $J \leftarrow_{\text{R}} \{0,1\}^n$.
2. Ask for i encryptions of c.

3. Choose $x_0 = J, x_1 = c$ and give x_0, x_1 to the sender to obtain $y = \mathsf{E}_k(x_i)$ for $i \leftarrow_{\text{R}} \{0,1\}$.
4. Ask for $T - i - 1$ encryptions of J.
5. Feed all the encryptions plus "OK" messages to A and get a guess g from A.
6. If $g = c$ then output 1; otherwise output a random bit in $\{0,1\}$.

We have that if $i = 0$ then we output 1 with probability $p + (1-p)\frac{1}{2} = \frac{1}{2} + \frac{p}{2}$, while if $i = 1$ we output 1 with probability at least $\frac{1}{2} + \frac{p}{+}\frac{\epsilon}{2T}$. Hence, the probability we make the correct guess is at least

$$\tfrac{1}{2}\big(\tfrac{1}{2} + \tfrac{p}{2} + \tfrac{\epsilon}{2T}\big) + \tfrac{1}{2}\big(1 - \tfrac{1}{2} - \tfrac{p}{2}\big) = \tfrac{1}{2} + \tfrac{\epsilon}{4T}$$

□

Active attacks In an active attack we allow Eve to the drop, insert and modify messages between Alice and Bob. (Of course, she can completely cut off the communication between them, but then she won't learn the credit card number.) Alice and Bob can always check that messages are of the proper length etc.., and so we'll assume that Eve only sends well formed messages (though this is a huge issue in practice!).

Does the proof go through? To simulate Eve in this case, we'd like to change Step 5 above to the following: For every generated encryption y_i, feed y_i to A and obtain the modified version y_i'. Then, check if y_i' is an encryption of c and if so, feed "OK" to A; otherwise feed it with GTH. But of course, we cannot check if y_i' is an encryption of c in the course of running an attack on the encryption scheme!

Is it just the proof? It seems "obvious" that the problem is with the proof method and that Protocol 1 is secure even against active attacks - after all how can Eve guess the credit card when it's never sent in the clear??

Counterexample It turns out the intuition is false - there is a CPA secure encryption which makes Protocol 1 *insecure* against active attacks! The counterexample below may seem contrived (and it is) but it illustrates an idea that was used in actual attacks on real-life protocols (specifically SSL V3.0).

Consider the unary encoding of decimal digits as bit strings: the digit i is encoded a string of 11 bits starting with i zeros and then a 1. To be more specific, the encoding algorithm will encode i as $0^i 1^{11-i}$ but the

decoding algorithm will only check for existence of a single 1 following the first i zeroes.

Now suppose that the credit card number c is encoded in this form to a string $\hat{c}$, and then we use the PRF-based CPA-secure encryption we saw in class ($\mathsf{E}_k(x) = (r, f_k(r) \oplus x)$) to encrypt $\hat{c}$.

The crucial observation is that if Eve gets a ciphertext $\mathsf{E}_k(x) = (r, y)$ even without knowing what is the plaintext, she can still flip the i^{th} bit of y and that will change the ciphertext to the encryption of x with its i^{th} bit flipped.

This means that Eve to guess a particular digit of the card, Eve can try to flip the last location in the encoding of that digit, then one before last, and so on. If i is the first location where Eve got a "GTH" message then the digit was $i - 1$.

Thus, after at most 88 executions Eve would be able to learn the entire credit card number.

On the importance of contrived counterexamples: Why is this meaningful? On a first encounter a natural reaction to such a counterexample is that this is cheating. This is an obviously contrived encryption scheme which no designer in his right mind would use in a login protocol. How can such an example teach us something about security? There are several answers to this concern (in some sense these are all different ways to state the same answer):

1. Although this example is contrived, its only a simplified presentation of attacks which worked for real-world protocols such as WEP, IPSEC, SSH etc.
2. Such examples teach us what we need to assume about the underlying components that we use. If there is a credit-card payment protocol that uses an encryption this example tells us that if the protocol is secure at all, its security is not based solely on the fact that the encryption scheme is CPA secure, but rather the protocol needs some additional property from the encryption scheme. This is important because even if the protocol is secure now, at some future date someone might decide to use a different encryption scheme for the protocol, and so it is crucial to explicitly state the security requirements from the encryption scheme used.
3. The fact that this example does not immediately imply that protocol X is insecure does not mean that protocol X is secure . The onus

is always on the protocol designer to demonstrate that the protocol is secure. If the designer claims that his login protocol is secure, he should state under what conditions on the encryption scheme this will be the case.

Stronger notion of encryption: It turns out there is a stronger notion of encryption scheme, that suffices to imply security of Protocol 1 and in many other places. This notion might seem "crazy strong" but it turns out that:

- It is needed for many applications.
- There are pretty efficient constructions conjectured to satisfy it.

Chosen ciphertext attack The definition of chosen ciphertext attack is a variation of chosen plaintext attack but this time the attacker is given oracle access even to the decryption box!

- Adversary gets oracle access to $x \mapsto \mathsf{E}_k(x)$ and $y \mapsto \mathsf{D}_k(y)$.
- Adversary chooses x_0, x_1.
- Sender chooses $k \leftarrow_{\mathrm{R}} \{0,1\}^n$, $i \leftarrow_{\mathrm{R}} \{0,1\}$ and sends $y^* = \mathsf{E}_k(x_i)$ to the adversary.
- Adversary gets access to $x \mapsto \mathsf{E}_k(x)$ and $y \mapsto \mathsf{D}_k(y)$ with one restriction - it cannot ask the decryiption box the exact string y^*.
- Adversary comes up with a guess j. She *wins* if $i = j$.

(E, D) is CCA-secure if for every poly Adv, poly-bounded $\epsilon : \mathbb{N} \to [0,1]$, and large enough n, $\Pr[\mathsf{Adv} \text{ wins }] \leq 1/2 + \epsilon(n)$.

Theorem 2 If (E, D) is CCA-secure then Protocol 1 is an actively secure authentication protocol.

Proof We modify the proof of Theorem 1 in the following way: first, we assume that Bob sends an OK message if the encryption is either of c or the junk message J. We can do that because the probability that Eve sends an encryption of the junk message J is at most $\frac{T}{2^n}$.

Then, we can simulate Eve - in Step 5 we do the following: for every generated encryption y_i, feed y_i to A and obtain the modified version y_i'. Then, if $y_i' = y^*$ then feed "OK" to A; otherwise, ask the decryption box to decrypt y_i' and answer OK iff it decrypts to either c or J.

An example of a non CCA secure encryption The basic CPA secure encryption we showed in class: given PRFs $\{f_k\}$, encrypt x by choosing $r \leftarrow_{\mathrm{R}} \{0,1\}^n$ and outputting $(r, f_k(x) \oplus r)$ is not CCA secure.

Construction of CCA secure encryption scheme The following encryption scheme is CCA secure: let $\{p_k\}$ be a collection of pseudorandom permutation mapping $\{0,1\}^{3n}$ to $\{0,1\}^{3n}$:

- To encrypt $x \in \{0,1\}^n$ do the following: choose $r \leftarrow_R \{0,1\}^n$, and send $p_k(x\|r\|0^n)$ (were $\|$ denotes concatenation).
- To decrypt $y \in \{0,1\}^{3n}$, compute $x\|r\|w = p_k^{-1}(y)$. if $w \neq 0^n$ then output $\perp$. Otherwise, output x.

Proving that this is CCA secure is left as exercise.

Another construction of a CCA secure encryption scheme Here's another construction: Suppose that (E, D) is CPA secure, and let $\{f_k\}$ be a collection of PRF's. The following scheme (E', D') is a CCA secure encryption:

Key (k, k') where k is a key for (E, D) and k' is a key for the PRF collection.

Encrypt $E'_{k,k'}(x) = (y, t)$ where $y = E_k(x)$ and $t = f_{k'}(y)$.

Decrypt $D_{k,k'}(y, t) = \perp$ if $f_{k'}(y) \neq t$ and $D_k(y)$ otherwise.

Exercise: prove that this is CCA secure. Is it still CCA secure even if you choose the same $k' = k$?

Something to think about Is there a CPA secure encryption where an active Eve can set every bit in the encrypted to text to an arbitrary value of her choice + flip every bit she wants to?