Lecture 8 - Message Authentication Codes II

Boaz Barak

October 12, 2005

Order of Encryption and Authentication Recall the login problem:

The login problem. Assume that client and server share a secret PIN PIN that was chosen at random between 0 and 10000 (e.g. a 13 bit number). Suppose in addition they have a shared key $k \leftarrow_{\mathrm{R}} \{0,1\}^n$ between it and the server, which an adversary does not know. Now, suppose they run the following protocol: the client sends the server an encryption of PIN, and the server decrypts and verifies that this is the correct PIN. If not, the server aborts the communication.

We saw that CPA-secure encryption alone can't be use to solve this. A natural approach is to add authentication. There are three natural constructions:

- Encrypt and then Authenticate (EtA): Compute $y = \mathsf{E}_k(PIN)$ and $t_y = \mathsf{Sign}_{k'}(y)$ and send (y, t_y). (IPSec-style)
- Authenticate and then Encrypt (AtE): Compute $t_x = \mathsf{Sign}_{k'}(PIN)$ and then $\mathsf{E}_k(x, t)$. (SSL style)
- Encrypt and Authenticate (E& A): Compute $y = \mathsf{E}_k(PIN)$ and $t_x = \mathsf{Sign}_{k'}(PIN)$ and send (y, t_x). (SSH style)

(Don't encrypt is WEP-style) Note that in all these methods we use independent keys for encryption and authentication.

It turns out that generically there is only one right choice.

Theorem 1.

1. *If (E, D) is CPA-secure and $(\mathsf{Sign}, \mathsf{Ver})$ is CMA-secure then the probability that poly-time adversary guesses PIN after seeing poly many interactions of the EtA protocol is less than $1/9999$.*
2. *There is a CPA-secure encryption such that for every CMA-secure MAC the adversary can find out the PIN after 13 sessions of AtE protocol.*
3. *There is a CMA-secure MAC such that for every CPA-secure encryption, the adversary can find out the PIN after just one session of the E& A protocol.*

Note: This does not by itself mean that, say, SSL is not secure. But it does mean that it is not *generically secure.* That is, the SSL protocol relies on specific (and not explicitly stated) properties of the encryption scheme used.

This theorem and its proof can be found in Hugo Krawczyk's CRYPTO 2001 paper "The order of encryption and authentication for protecting communications (Or: how secure is SSL?)", see `http://eprint.iacr.org/2001/045`.

CCA security Intuitively, we want an encryption scheme to behave as a "sealed digital envelope". That is, given an encryption of x we should not be able to find out x, but we should also not be able to transform this to an encryption of x'. The examples we saw in this and the last class showed us that CPA-security does not suffice to obtain "envelope-like" behavior which leads us to consider a stronger notion of security called *chosen cipher text attack* (CCA).

Recall that CPA security means that we preserve security even if we let the adversary play with an encryption box (with the secret key inside the box). Loosely speaking, CCA security means that we maintain security even if we let the adversary not only play with an encryption box but also with a *decryption* box.

A story sometimes told is to imagine that the adversary found a decryption box in the desert, some other stories are the "lunchtime" "midnight" attack (you left your decryption box on the table and went out to lunch, etc..). However these stories seem unrealistic and do not provide a good motivation in my view. This is the same way that the motivation for CPA-security is not that the adversary may find an encryption box in the desert. Rather it is that the adversary may influence the sort of messages that the sender encrypts (e.g., the British made military maneuvers in WWII that caused the German to send certain encrypted messages). In a similar way, it is sometimes possible for an attacker to influence the sort of messages that a receiver *decrypts*. Indeed, this is exactly the sort of attack that we had against the login protocol. Thus CCA security comes to prevent such attacks. While in some sense it is an overkill (and there are some slightly weaker variants), CCA security is currently the most standard formalism of what it means for an encryption scheme to be a "digital envelope". Indeed, in 1988 Daniel Bleichenbacher found an ingenious attack on the widely used SSL protocol that relied on the fact that this protocol used a non CCA secure encryption scheme.

See Victor Shoup's article "Why chosen ciphertext security matters?" (`http://shoup.net/papers/expo.pdf`) for more on the motivation behind CCA security (although he talks about public-key encryption, the main ideas are the same).

Definition 1 (CCA security — take 1)**.** An encryption (E, D) is said to be (T, ϵ)-*CCA secure* if it's valid ($\mathsf{D}_k(\mathsf{E}_k(x)) = x$) and for every T-time A if we consider the following game:

- Sender and receiver choose shared $k \leftarrow_{\mathrm{R}} \{0,1\}^n$.
- A gets access to black boxes for $\mathsf{E}_k(\cdot)$ and $\mathsf{D}_k(\cdot)$.
- A chooses x_1, x_2.
- Sender chooses $i \leftarrow_{\mathrm{R}} \{1,2\}$ and gives A $y = \mathsf{E}_k(x_i)$.
- A gets more access to black boxes for $\mathsf{E}_k(\cdot)$ and $\mathsf{D}_k(\cdot)$.
- A outputs $j \in \{1,2\}$.

A is successful if $j = i$, the scheme is (T, ϵ) secure if the probability that A is successful is at most $\frac{1}{2} + \epsilon$.

This definition is strong enough such that an encryption scheme satisfying it would be useful in many applications, including the login protocol (even without authentication). However, it is a little bit too strong... there does not exist any encryption scheme satisfying it.

Indeed, after getting the challenge y, A can always feed y to the decryption box and get back x_i. Thus we need to slightly relax the definition of CCA security to get something that is not impossible to satisfy. We'll make the minimum relaxation: we'll say that in the second stage A can ask the decryption box anything it wants *except* for y. The resulting definition:

Definition 2 (CCA security — actual definition)**.** An encryption (E, D) is said to be (T, ϵ)-*CCA secure* if it's valid $(\mathsf{D}_k(\mathsf{E}_k(x)) = x)$ and for every T-time A if we consider the following game:

- Sender and receiver choose shared $k \leftarrow_{\text{R}} \{0,1\}^n$.
- A gets access to black boxes for $\mathsf{E}_k(\cdot)$ and $\mathsf{D}_k(\cdot)$.
- A chooses x_1, x_2.
- Sender chooses $i \leftarrow_{\text{R}} \{1,2\}$ and gives A $y = \mathsf{E}_k(x_i)$.
- A gets more access to black boxes for $\mathsf{E}_k(\cdot)$ and $\mathsf{D}_k(\cdot)$ but is restricted not to ask y to the decryption box. More formally, A gets access to the following function $D'_k(\cdot)$ instead of $\mathsf{D}_k(\cdot)$

  $$D'_k(y') = \begin{cases} D_k(y') & y' \neq y \\ \perp & y' = y \end{cases}$$

  ($\perp$ is a symbol that signifies "failure" or "invalid input")
- A outputs $j \in \{1,2\}$.

A is successful if $j = i$, the scheme is (T, ϵ) secure if the probability that A is successful is at most $\frac{1}{2} + \epsilon$.

Construction of a CCA secure scheme. We'll now show how to construct a CCA-secure encryption scheme. That is, we prove the following theorem:

Theorem 2. *Assuming Axiom 1, there exists a CCA secure (private key) encryption scheme.*

Proof. The proof is actually to use the EtA construction, assuming some extra condition on the MAC (which is satisfied by the PRF-based construction). We say that a MAC has *unique signatures* if for every x there's at most one tag t such that $\mathsf{Ver}_k(x,t) = 1$. This is equivalent to saying that $\mathsf{Ver}_k(x,t)$ outputs 1 if and only if $t = \mathsf{Sign}_k(x,t)$ (note that this is how Ver worked in the PRF-based construction). Let $(\mathsf{Sign}, \mathsf{Ver})$ be such a MAC and let $(\mathsf{E}', \mathsf{D}')$ be a CPA-secure scheme. Our CCA-secure scheme (E, D) will be the following:

- Key: $\langle k, k' \rangle$ with $k, k' \leftarrow_{\text{R}} \{0,1\}^n$.
- Encryption: To encrypt x compute $y = \mathsf{E}_k(x)$, $t = \mathsf{Sign}_{k'}(y)$ and send $\langle y, t \rangle$.
- Decryption: To decrypt $\langle y, t \rangle$ first verify that $\mathsf{Ver}_{k'}(y,t) = 1$, otherwise abort (i.e., output $\perp$). If check passes, compute $\mathsf{D}_k(y)$.

Security: Suppose that A is a T-time algorithm attacking the encryption scheme (E, D). We'll convert A to an algorithm A' that breaks the CPA-secure scheme (E, D).

First, we need to remember what does it mean to have a CPA attack against $(\mathsf{E}', \mathsf{D}')$. The algorithm A' gets black-box access to $\mathsf{E}'_k(\cdot)$ but *not* to $\mathsf{D}'_k(\cdot)$. The algorithm A' will do the following:

- Choose $k' \leftarrow_{\mathrm{R}} \{0,1\}^n$.
- Run A in "its belly"
- Whenever A asks for an encryption of x, pass the request to the encryption box E'_k to obtain $y = \mathsf{E}'_k(x)$, compute $t = \mathsf{Sign}_{k'}(y)$ and give $\langle y,t\rangle$ to A'. Also record this query in a table.
- If A asks for a decryption of $\langle y,t\rangle$ which was previously returned to it as an encryption of x then return x to A.
- If A asks for a decryption of $\langle y,t\rangle$ which was *not* previously returned to from the encryption oracle, then check if $\mathsf{Ver}_{k'}(y,t) = 1$. If check fails then return $\perp$ to A. If check succeeds then abort the computation. In this case we say that A' failed to simulate A.
- When A sends the challenge x_1, x_2 pass it on to the sender to obtain $y = \mathsf{E}'_k(x_i)$ and give $\langle y,t\rangle$ to A, where $t = \mathsf{Sign}_{k'}(y)$.
- When A outputs a guess j, output the same guess j.

We see that the only case that A' fails to simulate the CCA attack of A is when A manages to produce a pair $\langle y,t\rangle$ such that

1. $\langle y,t\rangle$ was *not* obtained as a previous response to a query x of the encryption oracle.
2. $\langle y,t\rangle$ is *not* the encryption of the challenge.
3. $Ver_{k'}(y,t) = 1$

However, if A does that then he breaks the MAC. Indeed, because of the unique signatures property of the MAC, Properties 1 and 2 imply that y was not previously signed by the MAC, and hence it should not be possible for A to find a t such that $\mathsf{Ver}_{k'}(y,t) = 1$. $\square$

Note that the unique signatures property is indeed crucial. Suppose that the MAC had the property that it had an "extra unused bit". That is, the tag is of the form $t \circ b$ where b is a single bit, but verification only looks at the tag t. Thus, $\mathsf{Ver}_{k'}(y,t0) = 1$ if and only if $\mathsf{Ver}_{k'}(y,t1) = 1$.

In this case the encryption scheme will be clearly *not* CCA secure. (If the adversary gets the challenge $\langle y,t0\rangle$ it will give $\langle y,t1\rangle$ to the decryption oracle.) This sensitivity of CCA security to extra unused bits is one reason why some people feel CCA security is a bit too strong, but the research community has yet to find a clean definition that is still sufficient for all the applications of CCA security.