CS 120/ E-177: Introduction to Cryptography

Salil Vadhan and Alon Rosen Nov. 22, 2006

Lecture Notes 16:

Message Authentication

Recommended Reading.

- Katz-Lindell 4.1–4.5

1 The Problem

- *Data authenticity:* How do you know whether a message you receive actually came from who you think it did? And was not tampered with?
- Telephone: their voice, Written letters: handwriting, signature, Electronic communications? E-mail headers?
- Not provided by encryption! The authenticity problem is different from the privacy problem. Here we want data integrity, but many encryption schemes are "malleable" (i.e. the plaintext can easily be modified by modifying the ciphertext).
- Idea: attach a "tag" or "signature" to every message that "authenticates" it as coming from a particular party.
  - Message Authentication Codes: private-key version. The two parties, sender and receiver, share a private key to verify that the message comes from the person whom the key is shared with.
  - Digital Signatures: public-key version. anyone can verify.

2 The Definition

Definition 1 *A* message authentication code *consists of three algorithms* (G, M, V) *such that:*

- *The* key generation *algorithm* G *is a randomized algorithm that returns a* key k*; we write* $k \stackrel{R}{\leftarrow} G(1^n)$.
- *The* tagging *algorithm* M *is a (possibly) randomized algorithm that takes a key* k *and a* message m *and outputs a tag* t*; we write* $t \stackrel{R}{\leftarrow} M_k(m)$.
- *The* verification *algorithm* V *is a deterministic algorithm that takes a key* k*, a message* m*, and a tag* t*, and outputs* $V_k(m,t) \in \{\texttt{accept}, \texttt{reject}\}$.

Associated with the scheme is a message space $\mathcal{P}$ *from which* m *is allowed to be drawn. We require* $V_k(m, M_k(m)) = \texttt{accept}$ *for all* $m \in \mathcal{P}$, $k \stackrel{R}{\leftarrow} G(1^n)$.

May allow randomized or stateful tagging algorithms, but (unlike) encryption, deterministic stateless schemes are possible.

Defining security:

- The adversary's goal is to produce a forgery, i.e. produce any pair (m, t) such that $V_k(m, t) =$ `accept`. We will not make any assumptions on the formatting of messages, so even if m is nonsensical, it still counts as a forgery.
- Attack model: chosen message attack. The adversary selects messages m_i and gets to see their tags t_i before trying to produce a forgery. We allow an adaptive attack, i.e. the adversary can select m_{i+1} based on $(m_1, t_1), \ldots, (m_i, t_i)$.
- Unavoidable attacks: we will not protect against replay attacks in our definition (though there are various ways of accomplishing this, through a stateful verification algorithm). We will require that the forgery (m, t) is not one of the adversary's queries.

Definition 2 (unforgeability under adaptive chosen message attack) *A message authentication scheme (G, M, V) is* secure *if for every PPT A, there is a negligible function ε such that*

$$\Pr\left[A^{M_k(\cdot)}(1^n) \text{ forges}\right] \leq \varepsilon(n) \qquad \forall n,$$

"A forges" $\equiv$ A produces a pair (m, t) for which (a) $V_k(m, t) =$ `accept`, and (b) m is different from all of A's queries to the M_k-oracle.

- Preventing Replay Attacks: time stamps, counters, unique identifiers.
- As usual, definition is conservative, errs on safe side.

3 MACs for Fixed Length

Simple construction: $M_k(m) \stackrel{\text{def}}{=} f_k(m)$ where $\mathcal{F}_n = \{f_k : \{0,1\}^n \to \{0,1\}^n\}$ is a pseudorandom function family.

$$V_k(m, t) = \begin{cases} \texttt{accept} & \text{iff } f_k(m) = t \\ \texttt{reject} & \text{otherwise} \end{cases}$$

Note that the construction is deterministic and stateless.

Theorem 3 *If $\mathcal{F} = \bigcup_n \mathcal{F}_n$ is a pseudorandom function family, then the MAC defined above is secure.*

Proof Sketch: Let A be any PPT.

Claim 4 *The probability that A forges when a truly random function is used (i.e., in the "Ideal MAC") is $\leq 2^{-n}$.*

We use the dynamic view of the truly random function f: the values of f are generated "on the fly". $f(m)$ is chosen at random so $\Pr[t = f(m)] = \frac{1}{2^n}$.

Claim 5 *The probability that A forges when a pseudorandom function is used is at most $2^{-n} +$ $\text{neg}(n)$.*

Assume we have $|\Pr[\text{A forges in the real MAC}] - \Pr[\text{A forges in the ideal MAC}]| > \varepsilon$ where ε is nonnegligible. We will build a PPT D that is a distinguisher for the PRF.

D runs A, answers A's queries using its oracle, and when A outputs the pair (m, t), D makes another query to the oracle to obtain $f(m)$. D outputs 1 if $f(m) = t$ and m wasn't previously queries by A, 0 otherwise. The probability that D outputs 1 is exactly the probability that A produces a successful forgery (m, t), hence D can distinguish between a truly random function and a pseudorandom function with nonnegligible advantage, which is impossible. □

If we believe block ciphers (like DES) are pseudorandom permutations, then they directly give MACs for fixed message length.

4 MACs for Arbitrary Length

The above MAC is for message space $\{0,1\}^n$. What about long messages? Suppose that we have a MAC secure for messages of length n. For a message $m = m_1, m_2 \cdots, m_d$ where $|m_i| = n$.

- Attempt 1: Define $M'_k(m) = M_k(m_1 \oplus m_2 \oplus \ldots \oplus m_d)$.
- Attempt 2: Define $M'_k(m) = M_k(m_1), \cdots, M_k(m_d)$.
- Attempt 3: For, say $|m_i| = n/2$, define $M'_k(m) = M_k(m_1, 1), \cdots, M_k(m_d, d)$
  (here we assume that an index i is encoded by $n/2$ bits).

Let (G, M, V) be a MAC for message space $\{0,1\}^n$. Create a MAC for all messages of length $\leq 2^{n/6}$ by defining $M'_k(m)$ as follows:

- Write $m = m_1, m_2, \cdots, m_d$, where each $m_i \in \{0,1\}^{n/3}$.
- Choose $r \xleftarrow{R} \{0,1\}^{n/3}$ (a random "identifier")
- For $i = 1, \ldots, d$, compute $t_i \xleftarrow{R} M_k(m_i, i, d, r)$.
- Output $(r, t_1 t_2 \cdots t_d)$.

$V'_k(m_1, \cdots, m_d, (r, t_1, t_2, \cdots, t_d))$ accepts if for all $i \in \{1, \cdots, d\}$, $V_k((m_i, i, d, r), t_i) = \texttt{accept}$.

We are implicitly assuming that (i, d) are encoded using $n/3$ bits. This is more than enough, since $||i|| + ||d|| \leq n/6 + n/6 = n/3$.

Theorem 6 *If (G, M, V) is secure, then (G', M', V') is secure.*

Proof Sketch: Suppose there were a forger A' for (G', M', V') with nonnegligible success probability ε. After obtaining MACs of polynomially many messages, A produces a forgery (m, t). If this forgery is successful, then $m = m_1, \cdots, m_d$, $t = (r, t_1, \cdots, t_d)$, and for all i, $V_k((m_i, r, i, d), t_i) = \texttt{accept}$. At a high level, the analysis can be divided into cases:

1. The forgery $t = (r, t_1, \cdots, t_d)$ contains a "new" r (i.e. one that has not appeared in the previous MACs A' has seen). This means that the messages on which M_k is applied in the forgery (i.e. the (m_i, r, i, d)) have not appeared in the MACs A' has previously seen. One can then transform the forger A' for (G', M', V') into a forger A for (G, M, V).

2. The forgery $t = (r, t_1, \cdots, t_d)$ does *not* contain a "new" r. Here we have two sub-cases:

   (a) r has appeared in only one MAC $m' = m'_1, \cdots, m'_d$, $t' = (r, t'_1 \cdots t'_d)$ that A' has seen. Since $m' \neq m$ (by definition of a forgery), it must be the case that $m'_i \neq m_i$ for some i. Again, one can use this fact to transform A' into a forger A for (G, M, V).

   (b) r has appeared more than once. This even happens with probability smaller than $q^2/2^{n/3}$, where q is the number of MAC queries made by A' (which is negligible in n).

□

CBC MAC : The previous construction involves d applications of (G, M, V) and the size of the resulting tag is dn. For practical purposes it would be desirable to have a shorter tag. One example for a more efficient MAC is the CBC MAC, which we describe next.

Let $\mathcal{F} = \{f_k : \{0,1\}^n \to \{0,1\}^n\}$ be a family of pseudorandom functions (or pseudorandom permutations). Define a MAC over message space $\{0,1\}^{d \cdot n} = (\{0,1\}^n)^d$ by defining $M_k(m_1, \cdots, m_d) = y_d$, where $y_i = f_k(m_i \oplus y_{i-1})$ and $y_0 = 0^n$.

Theorem 7 *CBC MAC is secure for message space* $\{0,1\}^{d \cdot k}$.

- DES CBC MAC used extensively in practice.
- There are other ways to convert fixed-length MACs into arbitrary-length MACs, e.g. "hash-then-MAC" which we'll see next time.