MESSAGE AUTHENTICATION CODE CONCEPTS

Overview:

The main requirement of message authentication code is to prevent an adversary from modifying any message that is sent by one party to another, it also prevents injecting a new the message, without the receiver detecting that the message did not originate from the intended party.

DEFINITION 4.1 *A* message authentication code *(or* MAC*) consists of three probabilistic polynomial-time algorithms* (Gen, Mac, Vrfy) *such that:*

1. *The* key-generation algorithm Gen *takes as input the security parameter* 1^n *and outputs a key* k *with* $|k| \geq n$.
2. *The* tag-generation algorithm Mac *takes as input a key* k *and a message* $m \in \{0,1\}^*$, *and outputs a tag* t. *Since this algorithm may be randomized, we write this as* $t \leftarrow \mathsf{Mac}_k(m)$.
3. *The deterministic* verification algorithm Vrfy *takes as input a key* k, *a message* m, *and a tag* t. *It outputs a bit* b, *with* $b = 1$ *meaning* valid *and* $b = 0$ *meaning* invalid. *We write this as* $b := \mathsf{Vrfy}_k(m, t)$.

It is required that for every n, *every key* k *output by* $\mathsf{Gen}(1^n)$, *and every* $m \in \{0,1\}^*$, *it holds that* $\mathsf{Vrfy}_k(m, \mathsf{Mac}_k(m)) = 1$.

If there is a function ℓ *such that for every* k *output by* $\mathsf{Gen}(1^n)$, *algorithm* Mac_k *is only defined for messages* $m \in \{0,1\}^{\ell(n)}$, *then we call the scheme a* fixed-length MAC for messages of length $\ell(n)$.

As with private-key encryption, $\mathsf{Gen}(1^n)$ almost always simply chooses a uniform key $k \in \{0,1\}^n$, and we omit Gen in that case.

DEFINITION 4.2 *A message authentication code* $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$ *is* existentially unforgeable under an adaptive chosen-message attack, *or just* se-

cure, *if for all probabilistic polynomial-time adversaries* $\mathcal{A}$, *there is a negligible function* negl *such that:*

$$\Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

Construction of Code:

There are two kinds of MAC fixed-length MACs and variable-length MACs

For fixed-length MAC construction is as follows:

CONSTRUCTION 4.5

Let F be a (length preserving) pseudorandom function. Define a fixed-length MAC for messages of length n as follows:

- Mac: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^n$, output the tag $t := F_k(m)$.
- Vrfy: on input a key $k \in \{0,1\}^n$, a message $m \in \{0,1\}^n$, and a tag $t \in \{0,1\}^n$, output 1 if and only if $t \stackrel{?}{=} F_k(m)$.

Proof:

Let $\mathcal{A}$ be a probabilistic polynomial-time adversary. Consider the message authentication code $\widetilde{\Pi} = (\widetilde{\mathsf{Gen}}, \widetilde{\mathsf{Mac}}, \widetilde{\mathsf{Vrfy}})$ which is the same as $\Pi = (\mathsf{Mac}, \mathsf{Vrfy})$ in Construction 4.5 except that a truly random function f is used instead of the pseudorandom function F_k. That is, $\widetilde{\mathsf{Gen}}(1^n)$ works by choosing a uniform function $f \in \mathsf{Func}_n$, and $\widetilde{\mathsf{Mac}}$ computes a tag just as Mac does except that f is used instead of F_k.

We show that there is a negligible function negl such that

$$\left| \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1] - \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\widetilde{\Pi}}(n) = 1] \right| \leq \mathsf{negl}(n). \tag{4.1}$$

To prove this, we construct a polynomial-time distinguisher D that is given oracle access to some function $\mathcal{O}$, and whose goal is to determine whether $\mathcal{O}$ is pseudorandom (i.e., equal to F_k for uniform $k \in \{0,1\}^n$) or random (i.e., equal to f for uniform $f \in \mathsf{Func}_n$). To do this, D simulates the message authentication experiment for $\mathcal{A}$ and observes whether $\mathcal{A}$ succeeds in outputting a valid tag on a "new" message. If so, D guesses that its oracle is a pseudorandom function; otherwise, D guesses that its oracle is a random function.

Variable-length MAC can be constructed as follows:

CONSTRUCTION 4.7

Let $\Pi' = (\mathsf{Mac}', \mathsf{Vrfy}')$ be a fixed-length MAC for messages of length n. Define a MAC as follows:

- Mac: on input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^*$ of (nonzero) length $\ell < 2^{n/4}$, parse m as d blocks $m_1, \ldots, m_d$, each of length $n/4$. (The final block is padded with 0s if necessary.) Choose a uniform message identifier $r \in \{0,1\}^{n/4}$.

  For $i = 1, \ldots, d$, compute $t_i \leftarrow \mathsf{Mac}'_k(r\|\ell\|i\|m_i)$, where i, ℓ are encoded as strings of length $n/4$.[†] Output the tag $t := \langle r, t_1, \ldots, t_d \rangle$.
- Vrfy: on input a key $k \in \{0,1\}^n$, a message $m \in \{0,1\}^*$ of nonzero length $\ell < 2^{n/4}$, and a tag $t = \langle r, t_1, \ldots, t_{d'} \rangle$, parse m as d blocks $m_1, \ldots, m_d$, each of length $n/4$. (The final block is padded with 0s if necessary.) Output 1 if and only if $d' = d$ and $\mathsf{Vrfy}'_k(r\|\ell\|i\|m_i,\, t_i) = 1$ for $1 \leq i \leq d$.

[†] Note that i and ℓ can be encoded using $n/4$ bits because $i, \ell < 2^{n/4}$.

PROOF The intuition is that since Π' is secure, an adversary cannot introduce a new *block* with a valid tag (with respect to Π'). Furthermore, the extra information included in each block prevents the various attacks (dropping blocks, re-ordering blocks, etc.) sketched earlier. We prove security by showing that those attacks are the only ones possible.

Let Π be the MAC given by Construction 4.7, and let $\mathcal{A}$ be a probabilistic polynomial-time adversary. We show that $\Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1]$ is negligible. We first introduce some notation that will be used in the proof. Let repeat denote the event that the same random identifier is used in two of the tags returned by the MAC oracle in experiment $\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n)$. Denoting the final output of $\mathcal{A}$ by $(m, t = \langle r, t_1, \ldots \rangle)$, where m has length ℓ and is parsed as $m = m_1, \ldots$, we let $\mathsf{NewBlock}$ be the event that at least one of the blocks $r \| \ell \| i \| m_i$ was never previously authenticated by Mac' in the course of answering $\mathcal{A}$'s Mac queries. (Note that, by construction of Π, it is easy to tell exactly which blocks are authenticated by Mac'_k when computing $\mathsf{Mac}_k(m)$.) Informally, $\mathsf{NewBlock}$ is the event that $\mathcal{A}$ tries to forge a valid tag on a block that was never authenticated by the underlying fixed-length MAC Π'.

We have:

$$\begin{aligned}
\Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1] &= \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \mathsf{repeat}] \\
&\quad + \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\mathsf{repeat}} \wedge \mathsf{NewBlock}] \\
&\quad + \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\mathsf{repeat}} \wedge \overline{\mathsf{NewBlock}}] \\
&\leq \Pr[\mathsf{repeat}] \qquad\qquad (4.3) \\
&\quad + \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \mathsf{NewBlock}] \\
&\quad + \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\mathsf{repeat}} \wedge \overline{\mathsf{NewBlock}}].
\end{aligned}$$

We show that the first two terms of Equation (4.3) are negligible, and the final term is 0. This implies $\Pr[\text{Mac-forge}_{\mathcal{A},\Pi}(n) = 1]$ is negligible, as desired.

To see that $\Pr[\text{repeat}]$ is negligible, let $q = q(n)$ be the number of MAC oracle queries made by $\mathcal{A}$. To answer the ith oracle query of $\mathcal{A}$, the oracle chooses r_i uniformly from a set of size $2^{n/4}$. The probability of event repeat is exactly the probability that $r_i = r_j$ for some $i \neq j$. Applying Lemma A.15, we have $\Pr[\text{repeat}] \leq q^2/2^{n/4}$. Since q is polynomial (because $\mathcal{A}$ is a PPT adversary), this value is negligible.

We next consider the final term in Equation (4.3). We argue that if $\text{Mac-forge}_{\mathcal{A},\Pi}(n) = 1$, but repeat did not occur, then it must be the case that NewBlock occurred. In other words,

$$\Pr[\text{Mac-forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \overline{\text{repeat}} \wedge \overline{\text{NewBlock}}] = 0.$$

This is, in some sense, the heart of the proof.

Again let $q = q(n)$ denote the number of MAC oracle queries made by $\mathcal{A}$, and let r_i denote the random identifier used to answer the ith oracle query of $\mathcal{A}$. If repeat does not occur then the values $r_1, \ldots, r_q$ are distinct. Recall that $(m, t = \langle r, t_1, \ldots \rangle)$ is the output of $\mathcal{A}$. If $r \notin \{r_1, \ldots, r_q\}$, then NewBlock clearly occurs. If not, then $r = r_j$ for some unique j (because repeat did not occur), and the blocks $r \| \ell \| 1 \| m_1, \ldots$ could then not possibly have been authenticated during the course of answering any Mac queries other than the jth such query. Let $m^{(j)}$ be the message that was used by $\mathcal{A}$ for its jth oracle query, and let ℓ_j be its length. There are two cases to consider:

Case 1: $\ell \neq \ell_j$**.** The blocks authenticated when answering the jth Mac query all have $\ell_j \neq \ell$ in the second position. So $r\|\ell\|1\|m_1$, in particular, was never authenticated in the course of answering the jth Mac query, and NewBlock occurs.

Case 2: $\ell = \ell_j$**.** If $\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1$, then we must have $m \neq m^{(j)}$. Let $m^{(j)} = m_1^{(j)}, \ldots$. Since m and $m^{(j)}$ have equal length, there must be at least one index i for which $m_i \neq m_i^{(j)}$. The block $r\|\ell\|i\|m_i$ was then never authenticated in the course of answering the jth Mac query. (Because i is included in the third position of the block, the block $r\|\ell\|i\|m_i$ could only possibly have been authenticated if $r\|\ell\|i\|m_i = r_j\|\ell_j\|i\|m_i^{(j)}$, but this is not true since $m_i \neq m_i^{(j)}$.)

To complete the proof of the theorem, we bound the second term on the right-hand side of Equation (4.3). Here we rely on the security of Π'. We construct a PPT adversary $\mathcal{A}'$ who attacks the fixed-length MAC Π' and succeeds in outputting a valid tag on a previously unauthenticated message with probability

$$\Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A}',\Pi'}(n) = 1] \geq \Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \mathsf{NewBlock}]. \quad (4.4)$$

Security of Π' means that the left-hand side is negligible, implying that $\Pr[\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1 \wedge \mathsf{NewBlock}]$ is negligible as well.

The construction of $\mathcal{A}'$ is the obvious one and so we describe it briefly. $\mathcal{A}'$ runs $\mathcal{A}$ as a subroutine, and answers the request by $\mathcal{A}$ for a tag on m by choosing $r \leftarrow \{0,1\}^{n/4}$ itself, parsing m appropriately, and making the necessary queries to its own MAC oracle $\mathsf{Mac}'_k(\cdot)$. When $\mathcal{A}$ outputs $(m, t = \langle r, t_1, \ldots\rangle)$, then $\mathcal{A}'$ checks whether $\mathsf{NewBlock}$ occurs. (This is easy to do since $\mathcal{A}'$ can keep track of all the queries it makes to its own oracle.) If so, then $\mathcal{A}'$ finds the first block $r\|\ell\|i\|m_i$ that was never previously authenticated by Mac' and outputs $(r\|\ell\|i\|m_i,\, t_i)$. (If not, $\mathcal{A}'$ outputs nothing.)

The view of $\mathcal{A}$ when run as a subroutine by $\mathcal{A}'$ is distributed identically to the view of $\mathcal{A}$ in experiment $\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n)$, and so the probabilities of events $\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1$ and $\mathsf{NewBlock}$ do not change. If $\mathsf{NewBlock}$ occurs then $\mathcal{A}'$ outputs a block $r\|\ell\|i\|m_i$ that was never previously authenticated by its own MAC oracle; if $\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1$ then the tag on every block is valid (with respect to Π'), and so in particular this is true for the block output by $\mathcal{A}'$. This means that whenever $\mathsf{Mac\text{-}forge}_{\mathcal{A},\Pi}(n) = 1$ and $\mathsf{NewBlock}$ occur we have $\mathsf{Mac\text{-}forge}_{\mathcal{A}',\Pi'}(n) = 1$, proving Equation (4.4) and completing the proof of the theorem. ∎