CS 7880 Graduate Cryptography September 11, 2015

Lecture 2: Statistical Authentication and Secret Sharing

Lecturer: Daniel Wichs *Scribe: Ariel Hamlin*

1 Topic Covered

- Authentication
- MACs
- Secret Sharing
- Threshold Secret Sharing

2 Authentication

Consider two individuals **Alice** and **Bob** who wish to communicate a message **m** over an open channel. However on this channel there is an *active* adversary **Eve** who is capable of changing messages that travel across the wire, but not block them entirely. This presents a problem, even if the messages are encrypted. Consider the One Time Pad (OTP) example from last class.

$$\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0,1\}^*$$
$$\textsc{Enc}_k(m) = k \oplus m$$
$$\textsc{Dec}_k(m) = k \oplus c$$

Consider the case where the message is only a single bit. Eve intercepts Alice's message and XORs the message with a 1.

$$c = m \oplus k$$
$$c' = c \oplus 1$$

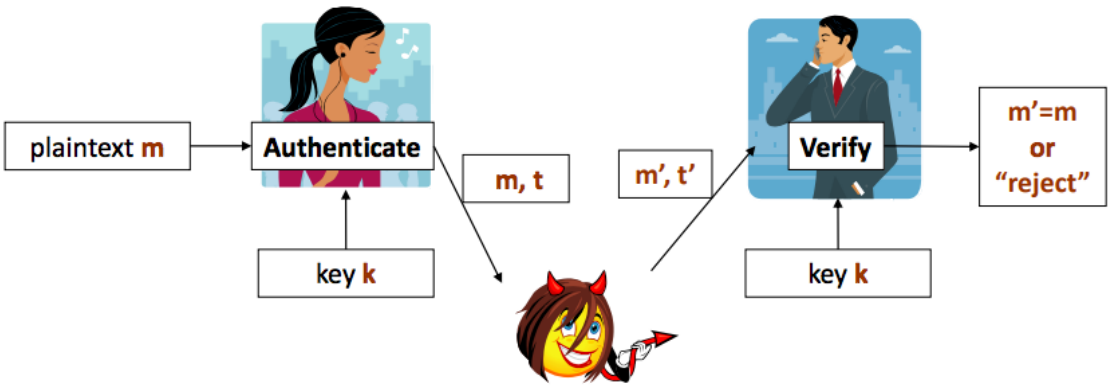
Decryption proceeds without Bob knowing the ciphertext has been altered.

$$m' = c' \oplus k$$
$$m' = m \oplus k \oplus 1 \oplus k$$
$$m' = m \oplus 1$$

Recalling the truth table for the XOR operation, Eve has succeeded in flipping whatever bit that Alice originally intended to send. If, on the other hand, Eve XORed c with zero, Bob would end up receiving a random value. Clearly this is a problem, and we wish Bob to be able to tell that Eve has modified any of Alice's cipher texts! Formally we wish to prevent the cipher texts from being malleable by providing *authentication* - or the ability to detect if Eve has altered the message.

Note when we consider the property of authentication, we are considering it independently of confidentiality provided by encryption. In fact, we will show later how to combine the two primitives.

3 Message Authentication Code (MAC)



MACs provide authentication against statistical adversaries - which is to say that there is no limit placed on the adversary computational ability. Alice has a message to send to Bob and would like to prevent Eve from altering it en-route. She thus computes a 'tag' that she sends along with the message that allows Bob to check if the message has been altered. More formally, a MAC scheme can be viewed as the following tuple:

$\mathcal{M}$	Message Space
$\mathcal{K}$	Key Space
$\mathcal{T}$	Tag Space
MAC: $\mathcal{K} \times \mathcal{M} \to \mathcal{T}$	MAC Function

Alice and Bob use the MAC in the following way. Alice computes $t = \text{MAC}(k, m)$ and sends (m, t) to Bob. Bob receives (m', t') and thus checks if $t' = \text{MAC}(k, m')$. If it does not, he knows that Eve has altered the message. We have now defined the usage of MACs, but what does it mean for a MAC to be secure? As with encryption security we use a game-based definition to define MAC security.

Definition 1 [1-Time Statistically Secure MAC]

Define MAC: $\mathcal{K} \times \mathcal{M} \to \mathcal{T}$ be a message authentication code and let Eve be an adversary. We define the following security game:

Game (Eve):

- k is chosen uniformly at random from $\mathcal{K}$, $k \xleftarrow{\$} \mathcal{K}$
- Eve chooses message m and receives $t \leftarrow \text{MAC}(k, m)$
- Eve chooses (m', t') such that $m \neq m'$, Eve wins if she able to generate a MAC for m' without previously seeing a tag for it, i.e. $t' = \text{MAC}(k, m')$

We say a scheme is ε-secure if for all adversaries $\Pr[\text{Eve wins}] \leq \varepsilon$. $\diamondsuit$

Note that this definition only defines security for a MAC in which the adversary only sees one message for any particular key. The MAC function itself is reused and the adversary is assumed to know the protocol, but for each new message that needs to be authenticated, a new key must be used.

This definition also begs the question, what is the lowest value ε can have – can it ever be zero? Consider an Eve who simply guesses a tag for m', the tag space is fine, so Eve's probability for guessing the right tag is simply $1/|\mathcal{T}|$. Eve can do no worse than simply guessing in this case. Thus in order for ε to be zero, the tag space would have to be infinite, which it is not.

3.1 MAC Construction

Now that we have defined a syntax and security definition for MACs, let us define a construction.

$\mathcal{M}$	$\mathbb{Z}_p$
$\mathcal{K}$	$\mathbb{Z}_p \times \mathbb{Z}_p$
$\mathcal{T}$	$\mathbb{Z}_p$
$\text{MAC}(k, m) = xm + y$	$k = (x, y)$

Note that all operations are occurring in the field $\mathbb{Z}_p$ which means the addition and multiplication are done in modulo p. (For more notes on fields, see the appendix at the end of the notes.) We now wish to show that this MAC construction meets our definition of 1-Time Statistical Security. We can do this by examining the probability that the MAC produces a fixed value. Formally:

Theorem 1 $\text{MAC}(k, m) = xm + y$ *is 1-Time secure for* $\varepsilon = 1/p$.

Proof: Let X and Y be uniform random variables representing the choice of x and y in the key. Thus $K = (X, Y)$ is also a uniform random variable. Thus for all messages m, and all tags t, we have:

$$\Pr[\text{MAC}(K, m) = t] = \Pr[Xm + Y = t] = 1/p$$

This follows from the fact that the equation $t - Xm = Y$ has a unique solution for X for every choice of Y. Now consider the probability for any $m \neq m'$ and for any t, t':

$$\begin{aligned} &\Pr[\text{MAC}(K, m) = t', \text{MAC}(Kim) = t] \\ &= \Pr[Xm' + Y = t', Xm + Y = t] \\ &= \Pr[X = x, Y = y] = 1/p^2 \end{aligned}$$

Where $x = \frac{t-t'}{m-m'}$ and $y = t - xm$. Thus from the probability we just calculated and the properties of conditional probabilities that we went over last class, we can show:

$$\Pr[\text{MAC}(K, m') = t' \mid \text{MAC}(K, m) = t] = 1/p$$

Thus this MAC construction satisfies 1-Time security for $\varepsilon = 1/p$. □

While this MAC construction is shown to be secure, it is not practical. In order to sign any values, the key must be twice the size of the message, and one key can only be used to sign one message. If the same key is used to sign two different messages, Eve can simply solve the system of equations and recover the key used. This begs the question, can we create a MAC with a smaller key size? Yes we can!

3.2 A Better MAC

We wish to improve the ratio between the size of the key and the size of the message it can sign. In order to do so, we introduce a new MAC construction.

$\mathcal{M}$	$\mathbb{Z}_p^d$ for $d \geq 1$
$\mathcal{K}$	$\mathbb{Z}_p \times \mathbb{Z}_p$
$\mathcal{T}$	$\mathbb{Z}_p$

For $k = (x, y)$, $m = (m_1, \ldots, m_d)$, $\text{MAC}(k, m) = \sum_{i=1}^{d}(m_i x^i + y)$. Note that the message space is assumed to be a vector of size d integers from $\mathbb{Z}_p$, but the key is the same size as our previous construction. We now have a scheme where the key size is independent of our message size. Now we have to prove this construction is 1-Time secure.

Theorem 2 $\text{MAC}(k, m) = \sum_{i=1}^{d}(m_i x^i + y)$ *is 1-Time secure for* $\varepsilon = d/p$.

Proof: Let X and Y be uniform random variables representing the choice of x and y in the key. Thus $K = (X, Y)$ is also a uniform random variable. Thus for all messages m, and all tags t, we have:

$$\Pr[\text{MAC}(K, m) = t] = \Pr[\sum_{i=1}^{d}(m_i X^i + Y)] = 1/p$$

This follows from the fact that the equation has a unique solution for X for every choice of Y. Now consider the probability for any $m \neq m'$ and for any t, t':

$$\begin{aligned}
&\Pr[\mathrm{MAC}(K,m) = t', \mathrm{MAC}(Kim) = t] \\
&= \Pr[\sum_{i=1}^{d}(m_i X^i + Y) = t, \sum_{i=1}^{d}(m'_i X^i + Y) = t'] \\
&= \Pr[\sum_{i=1}^{d}((m_i - m'_i)x^i) = t - t', Y = t' - \sum_{i=1}^{d}(m_i x'^i)] \\
&\Pr[E] * \Pr[F|E] \leq d/p * 1/p \leq d/p^2
\end{aligned}$$

Where E and F are the random variables defined as the probability $E := \sum_{i=1}^{d}(m_i x^i + y) = t - t'$ and $F := Y = t' - \sum_{i=1}^{d}(m_i x'^i)$. Thus from the probability we just calculated and the properties of conditional probabilities that we went over last class, we can show:

$$\Pr[\mathrm{MAC}(K, m') = t' \mid \mathrm{MAC}(K, m) = t] \leq d/p$$

Thus this MAC construction satisfies 1-Time security for $\varepsilon = d/p$. □

In practical matters, if we have a message size of 2^{33} (about 8 GB) we can get security of $\varepsilon \leq 2^{-102}$ with a key of only 258 bits! This is much improved from our last construction, however we can still only use that key once for a single message. Can we do better an authenticate more than one message with a single key? As it turns out, we can't do much better if we aren't willing to put some restrictions on our adversary and move out of statistical security. In fact, there is a lower bound on the number of messages and key size required.

Theorem 3 *To authenticate q messages with security $\varepsilon = 2^{-r}$ one needs a key of size $(q+1)r$.*

3.3 Combining Encryption and Authentication

Remember that authentication does nothing to provide confidentiality for the message, Eve still learns its contents. Fortunately, we can combine encryption and authentication to provide both integrity and confidentiality. However, the order in which they are combined is important. Consider the case where we have a message m, an encryption key k_e and an authentication key k_a and Alice computes in the following order.

$$\begin{aligned}
t &= \mathrm{MAC}(k_a, m) \\
c &= \textsc{Enc}(k_e, m)
\end{aligned}$$

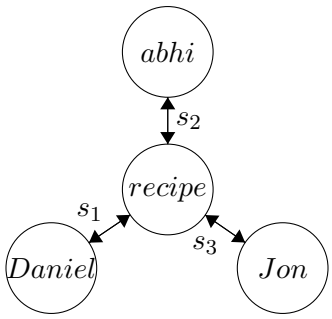
She then sends (t, c) to Bob. Bob simply decrypts c and checks to see if the message generates the same tag. Remember, authentication provides no guarantees as to the confidentiality of the message. As a result Eve may be able to recover something about m from t. This is bad!

As a result, the proper way to combine authentication and encryption is to "encrypt first, then MAC" as in the following construction.

$$c = \text{Enc}(k_e, m)$$
$$t = \text{MAC}(k_a, c)$$

4 Secret Sharing

Daniel, Jon, and abhi wish to found a baking empire, with the recipe for their famous chocolate chip cookies as the corner stone of their success. Now, while they all agreed that they were upstanding individuals, they still wished to control the recipe in such a way that none of them could take entire recipe and found their own baking empire. In fact, they wished to share the recipe amongst themselves in such a way that it required all three of them to recover the recipe, and that with only two or one shares they could not know anything about the recipe. Using secret sharing it is possible to get both properties.



Let n be the number of parties, $\mathcal{M}$ the message space, and $\mathcal{S}$ the share space. We define a secret sharing scheme as tuple of two algorithms (Share: $\mathcal{M} \xrightarrow{\$} \mathcal{S}^n$, Recover: $\mathcal{S}^n \to \mathcal{M}$).

Definition 2 [Correctness] A scheme is correct if the following holds:

$$\Pr_{r \in \text{Share}}[\text{Recover}(\text{Share}(m)) = m] = 1$$

♢

That is to say that reassembling the shares proceeded by Share using the Recover algorithm recovers the same message. Finally we define the security of the scheme.

Definition 3 [Security]

For all $\mathcal{M}$, set of colluding parties $A \subseteq \{1, \cdots, n\}$ where $|A| = n-1$. Let $(s_1, \cdots, s_n) = \text{Share}(M)$ and $S_A = \{s_i \ : \ i \in A\}$. Then the distributions of S_A and $\mathcal{M}$ are independent

♢

This definition is equivalent to the fact that observing a subset of shares doesn't give the adversary advantage, and also the definition that looking at the a subset of shares of the message doesn't allow the adversary to determine between two messages.

4.1 Secret Sharing Construction

Now that we have defined the syntax, security, and correctness of secret sharing, let us provide a construction. Define $\mathcal{M} = \mathbb{Z}_q$ and $\mathcal{S} = \mathbb{Z}_q$. Note, while we use $\mathbb{Z}_p$, any finite group can be used for the message and share space. We define the following share and recover algorithms

Share On input m:

1. Chose $s_1, \cdots, s_{n-1}$ uniformly at random
2. Set $s_n := m - (s_1 + \cdots + s_{n-1})$

Recover On input $s_1, \cdots, s_n$:

1. Compute $s_1 + \cdots + s_n$

We now will show the scheme that is defined above meets our definition of security for secret sharing.

Theorem 4 *The construction outline above is perfectly secure.*

Proof: For all M, set of colluding parties $A \subseteq \{1, \cdots, n\}/i$ any value s_A (the sum of the colluding parties), and for any m, we have:

$$\Pr[S_A = s_A \mid m = M] = 1/q^{n-1}$$

This holds as for a fixed m, each choice of s_A corresponds to a unique value of the remaining shares $s_1, \cdots, s_{n-1}$. As the probability is the same for all M this means that S_A and M are independent.

□

4.2 Threshold Secret Sharing

Coming back to our previous example of the baking empire, Daniel and Jon are interested in a secret sharing scheme that allows two of the three individuals to reconstruct the recipe. More generally, they would like some threshold $t+1$ of the n users can recover the message with just their shares, but only t parties learn nothing. We have a construction for n of n shares, can we generalize to allow for any t? We can, but first we must introduce a few techniques.

Lagrange Interpolation Lagrange interpolation is a method for recovering a t-degree polynomial based on $t+1$ points. Let $(x_0, y_0), \cdots, (x_t, y_t)$ be the points on the polynomial. Define the following function p_i:

$$p_i(x) := \Pi_{i \neq j} \frac{x - x_j}{x_i - x_j}$$

We can then define the polynomial passing through all points:

$$p(x) := \sum_{i=0}^{t} y_i * p_i(x)$$

Shamir Secret Sharing The construction defines a number of parties n, a threshold t such that $t < n$. The message space is $\mathcal{M} = \mathbb{Z}_q$, share space $\mathcal{S} = \mathbb{Z}_q$ where q is a prime and $q > n$. As with the previous construction, $\mathbb{Z}_q$ can be replaced with any finite field. We define the following share and recover algorithms.

Share On input m:

1. Chose t random coefficients, $c_1, \cdots, c_n$ and $c_0 = m$
2. Define polynomial $p(x) = \sum_{j=0}^{t} c_j x^j$
3. Output $s_i = p(i)$ for each party i

Recover On input $(1, s_1), \cdots, (n, s_n)$:

1. Lagrange Interpolation

Theorem 5 *Shamir Secret Sharing satisfies perfect secrecy.*

Proof: For all messages M, any t distinct points $z_1, \cdots, z_t \subseteq \mathbb{Z}_q/\{0\}$ and values $s_1, \cdots, s_t$ we have:

$$\Pr[p(z_1) = s_1, \cdots, p(z_t) = s_t \mid M = m] = 1/q^t$$

This comes from the fact that once we fix $p(0) = c_0 = m$, each choice of $s_1, \cdots, s_t$ corresponds to a unique choice of $c_1, \cdots, c_t$. □

It is an open problem in this space if it is possible to doing sharing with an arbitrary access structure with less than 2^n shares - as the naive approach is to simply assign shares for each possible minimal subset and has exponentially many shares.

Appendix: A Note on Fields

A field defined as a tuple of a set F, an addition operator, and a multiplication operator; $(F, +, *)$. It has the following properties:

1. Both operations $(+, *)$ are associative $((a + b) + c = a + (b + c))$ and commutative $(a * b = b * a)$.
2. Multiplication is distributive over addition: $a * (b + c) = a * b + b * c$
3. $(F, +)$ is a group with the identity 0.
    - For all x: $x + 0 = x$

- For all x: there exists $-x$ such that $x + (-x) = 0$

4. $(F^*, *)$ is a group with the identity 1 where $F^* = F/\{0\}$.
   - For all $x \in F^*$: $x * 1 = x$
   - For all $x \in F^*$: there exists x^{-1} such that $x * x^{-1} = 1$