Lecture Notes 17

87 Authentication While Preventing Impersonation

A fundamental problem with all of the password authentication schemes discussed so far is that Alice reveals her secret to Bob every time she authenticates herself. This is fine in an environment where she trusts Bob but not otherwise, for after authenticating herself once to Bob, then Bob can in turn masquerade as Alice to others.

When neither Alice nor Bob trust each other, there are two requirements that must be met:

1. Bob wants to make sure that an impostor cannot successfully masquerade as Alice.
2. Alice wants to make sure that her secret remains secure.

At first sight these seem contradictory, but there actually are ways for Alice to prove her identity to Bob without compromising her secret.

88 Challenge-response authentication protocols

In a challenge-response protocol, Bob presents Alice with a challenge that only the true Alice (someone knowing Alice's secret) can answer. Alice answers the challenge and sends her answer to Bob, who verifies that it is correct.

A challenge-response protocol can be built from a digital signature scheme (S_A, V_A) as shown in Figure 1. (The same protocol can also be implemented using a symmetric cryptosystem with shared key k.)

	Alice		Bob
1.		$\xleftarrow{r}$	Choose random string r.
2.	Compute $s = S_A(r)$	$\xrightarrow{s}$	Check $V_A(r, s)$.

Figure 1: Simple challenge-response protocol.

The problem with this protocol is that it exposes Alice's signature system to a chosen plaintext attack. With this protocol, a malicious Bob can get Alice to sign any message of his choosing. Among other things, this means that Alice had better have a different signing key for use with this protocol than she uses to sign contracts.

While we hope our cryptosystems are resistant to chosen plaintext attacks, such attacks are very powerful and are not easy to defend against. Anything we can do to limit exposure to such attacks can only improve the security of the system.

We now look at some ways that Alice might limit Bob's ability to carry out a chosen plaintext attack. In the protocol of Figure 2, instead of signing a string r of Bob's choice, Alice signs a string r that is constructed from a part r_1 chosen by Alice and a part r_2 chosen by Bob. The idea is that neither party be able to control r. Unfortunately, that idea does not work here because Bob gets r_1 before choosing r_2. Instead of choosing r_2 randomly, a cheating Bob can choose $r_2 = r \oplus r_1$, where

	Alice		Bob
1.	Choose random string r_1	$\xrightarrow{r_1}$	
2.		$\xleftarrow{r_2}$	Choose random string r_2.
3.	Compute $r = r_1 \oplus r_2$		Compute $r = r_1 \oplus r_2$
4.	Compute $s = S_A(r)$	$\xrightarrow{s}$	Check $V_A(r, s)$.

Figure 2: Attempt to resist chosen plaintext attack: Alice goes first.

r is the string that he wants Alice to sign as part of his chosen plaintext attack on her cryptosystem. Thus, the protocol of Figure 2 is no more secure against chosen plaintext attack than the simpler protocol of Figure 1.

Another possibility is to choose the random strings in the other order—Bob chooses first and then Alice—giving the protocol of Figure 3. Now Alice is the one who has complete control over r.

	Alice		Bob
1.		$\xleftarrow{r_2}$	Choose random string r_2.
2.	Choose random string r_1	$\xrightarrow{r_1}$	
3.	Compute $r = r_1 \oplus r_2$		Compute $r = r_1 \oplus r_2$
4.	Compute $s = S_A(r)$	$\xrightarrow{s}$	Check $V_A(r, s)$.

Figure 3: Attempt to resist chosen plaintext attack: Bob goes first.

This indeed thwarts Bob's chosen plaintext attack since r is completely random (i.e., all strings r are equally likely). No matter how Bob chooses r_2, Alice choice of a random string r_1 ensures that r is also random. Thus, Alice only signs random messages.

Unfortunately, the protocol of Figure 3 is totally insecure against active eavesdroppers. Suppose Mallory listens to a legitimate execution of the protocol between Alice and Bob. From this, he easily acquires a valid signed message (r_0, s_0). Now Mallory can impersonate Alice by choosing $r_1 = r_0 \oplus r_2$ in step 2 and $s = s_0$ in step 4. Bob computes $r = r_1 \oplus r_2 = r_0$ in step 3, so his verification in step 4 succeeds.

Both of these protocols can be improved by letting r be $r_1 \cdot r_2$ (concatenation) instead of $r_1 \oplus r_2$. That way, neither party has full control over r. This weakens Bob's ability to launch a chosen plaintext attack in the protocol of Figure 2, and it weakens Mallory's ability to impersonate Alice in the protocol of Figure 3. A still better idea might be to let $r = h(r_1 \cdot r_2)$, where h is a cryptographic hash function, since this further weakens the control that either party has on the choice of r.

89 Feige-Fiat-Shamir Authentication Protocol

In all of the challenge-response protocols above, Alice releases some partial information about her secret by producing signatures that Bob could not compute by himself. As we will see, the Feige-Fiat-Shamir protocol allows Alice to prove knowledge of her secret without revealing any information about the secret itself. Such protocols are called *zero knowledge*, which we will discuss in subsequent lectures.

The Feige-Fiat-Shamir protocol is based on the difficulty of computing square roots modulo composite numbers. Alice chooses $n = pq$, where p and q are distinct large primes. Next she picks

a quadratic residue $v \in \mathrm{QR}_n$ (which she can easily do by choosing a random element u and letting $v = u^2 \bmod n$). Finally, she chooses s to be the smallest square root of $v^{-1} \pmod{n}$.[1] She can do this since she knows the factorization of n. She makes n and v public and keeps s private.

Alice authenticates herself by successfully completing a protocol that requires knowledge of s. We present a simplified version of the protocol in Figure 4. In a single round of the protocol, Bob has at least a 50% chance of catching an impostor Mallory. By repeating the protocol t times, the error probability (that is, the probability that Bob fails to catch Mallory) drops to $1/2^t$. This can be made acceptably low by choosing t to be large enough. For example, if $t = 20$, then Mallory has only one chance in a million of successfully impersonating Alice.

	Alice		Bob
1.	Choose random $r \in \mathbf{Z}_n$.		
	Compute $x = r^2 \bmod n$.	$\xrightarrow{x}$	
2.		$\xleftarrow{b}$	Choose random $b \in \{0, 1\}$.
3.	Compute $y = rs^b \bmod n$.	$\xrightarrow{y}$	Check $x = y^2 v^b \bmod n$.

Figure 4: One round of the simplified Feige-Fiat-Shamir authentication protocol.

To see that this works when both parties are honest, we just have to verify that

$$x = y^2 v^b \bmod n. \tag{1}$$

But this follows since

$$y^2 v^b \equiv (rs^b)^2 v^b \equiv r^2 (s^2 v)^b \equiv x(v^{-1}v)^b \equiv x \pmod{n}.$$

89.1 Cheating Alice

We now turn to the security properties of the protocol when Alice is dishonest, that is, when a party Mallory is attempting to impersonate Alice.

Theorem 1 *Consider one round of the Feige-Fiat-Shamir protocol of Figure 4. Suppose Mallory, who is attempting to impersonate Alice, doesn't know a square root of v^{-1}. Then Bob's verification will fail with probability at least 1/2.*

Proof: In order for Mallory to successfully fool Bob, he must come up with x in step 1 and y in step 3 satisfying (1). He does not know which value b Bob will choose when he is sends x in step 1. Let y_b be the string that Mallory sends to Bob in response to query b. We consider two cases.

Case 1: There is at least one $b \in \{0, 1\}$ for which y_b fails to satisfy (1). Since $b = 0$ and $b = 1$ each occur with probability 1/2, this means that Bob's verification will fail with probability at least 1/2, as desired.

Case 2: y_0 and y_1 both satisfy (1) (for their respective values of b), so we have

$$x = y_0^2 \bmod n$$

and

$$x = y_1^2 v \bmod n.$$

[1] Note that if v is a quadratic residue, then so is $v^{-1} \pmod{n}$.

We can solve these equations for v^{-1} to get

$$v^{-1} \equiv y_1^2 y_0^{-2} \pmod{n}$$

But then $y_1 y_0^{-1} \bmod n$ is a square root of v^{-1}. Since Mallory was able to compute both y_1 and y_0, then he was also able to compute a square root of v^{-1}, contradicting the assumption that he doesn't "know" a square root of v^{-1}. ■

We remark that it *is* possible for Mallory to cheat with success probability 1/2. Here's what he does. He guesses the bit b that Bob will send him in step 2. He then generates a pair (x, y). If he guesses $b = 0$, then he chooses $x = r^2 \bmod n$ and $y = r \bmod n$, just as Alice would have. If he guesses $b = 1$, then he chooses y arbitrarily and $x = y^2 v \bmod n$. He then sends x in step 1 and y in step 3. The pair (x, y) passes Bob's check if Mallory's guess of b turns out to be correct, which will happen probability 1/2.

89.2 Cheating Bob

We now consider the case of a dishonest Mallory impersonating Bob. Alice would like assurance that if she follows the protocol, her secret is protected, regardless of what Bob does.

Consider what Mallory knows at the end of the protocol. If he sent $b = 0$ in step 2, then he ends up with a pair (x, y), where x is a random number and y is its square modulo n. Neither of these numbers depend in any way on Alice secret s, so it's intuitively obvious that this gives Mallory no direct information about s. It's also of no conceivable use to Mallory in trying to find s by other means, for he can compute such pairs by himself without involving Alice. If having such pairs allows him find a square root of v^{-1}, then he was already able to compute square roots, contrary to the assumption that finding square roots modulo n is difficult.

Instead, suppose Mallory sent $b = 1$ in step 2. Now he ends up with the pair (x, y), where $x = r^2 \bmod n$ and $y = rs \bmod n$. While y might seem to give information about s, observe that y itself is just a random element of $\mathbf{Z}_n$. This is because r is random, and the mapping $r \to rs \bmod n$ is one-to-one for all $s \in \mathbf{Z}_n^*$. Hence, as r ranges through all possible values, so does $rs \bmod n$. What does Mallory learn from x? Nothing that he could not have computed himself knowing y, for $x = y^2 v \bmod n$. So again, all he ends up with is a random number (y in this case) and a quadratic residue that he can compute knowing y.

In both cases, Mallory ends up with information that he could have computed without interacting with Alice. Hence, if he could have discovered Alice's secret by talking to Alice, then he could have also done so on his own, contradicting the hardness assumption for computing square roots. This is the sense in which Alice's protocol releases zero knowledge about her secret.