CPSC 467b: Cryptography and Computer Security

Lecture 18

Michael J. Fischer

Department of Computer Science
Yale University

March 29, 2010

1. Authentication While Preventing Impersonation
   - Challenge-response authentication protocols
   - Feige-Fiat-Shamir Authentication Protocol

Authentication While Preventing Impersonation

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 when Alice trusts Bob but not otherwise.

After authenticating herself once to Bob, then Bob can masquerade as Alice and impersonate her to others.

Authentication requirement

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 are ways for Alice to prove her identity to Bob without compromising her secret.

Challenge-Response Authentication Protocols

Challenge-response authentication protocols

In a challenge-response protocol, Bob presents Alice with a challenge that only the true Alice (or someone knowing Alice's secret) can answer.

Alice answers the challenge and sends her answer to Bob, who verifies that it is correct.

Bob learns the response to his challenge but Alice never reveals her secret.

If the protocol is properly designed, it will be hard for Bob to determine Alice's secret, even if he chooses the challenges with that end in mind.

Challenge-response protocol from a signature scheme

A challenge-response protocol can be built from a digital signature scheme (S_A, V_A).

(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 $V_A(r, s)$.

Requirements on underlying signature scheme

This protocol exposes Alice's signature scheme to a chosen plaintext attack.

A malicious Bob can get Alice to sign any message of his choosing.

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.

Limiting exposure to chosen plaintext attack: try 1

We explore some ways that Alice might limit Bob's ability to carry out a chosen plaintext attack.

Instead of letting Bob choose the string r for Alice to sign, r is constructed from two parts, r_1 and r_2.

r_1 is chosen by Alice; r_2 is chosen by Bob. Alice chooses first.

	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 $V_A(r, s)$.

Problem with try 1

The idea is that neither party should 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 r is the string that he wants Alice to sign.

Thus, try 1 is no more secure against chosen plaintext attack than the original protocol.

Limiting exposure to chosen plaintext attack: try 2

Another possibility is to choose the random strings in the other order—Bob chooses first.

	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 $V_A(r, s)$.

Try 2 stops chosen plaintext attack

Now Alice has complete control over r.

No matter how Bob chooses r_2, Alice choice of a random string r_1 ensures that r is also random.

This thwarts Bob's chosen plaintext attack since r is completely random.

Thus, Alice only signs random messages.

Problem with try 2

Unfortunately, try 2 is totally insecure against active eavesdroppers.
Why?

Problem with try 2

Unfortunately, try 2 is totally insecure against active eavesdroppers.
Why?

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).
How does this help Mallory?

Problem with try 2

Unfortunately, try 2 is totally insecure against active eavesdroppers.
Why?

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).
How does this help Mallory?

Mallory sends $r_1 = r_0 \oplus r_2$ 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.

Thus, Mallory can successfully impersonate Alice to Bob.

Further improvements

Possible improvements to both protocols.

1. Let $r = r_1 \cdot r_2$ (concatenation).
2. Let $r = h(r_1 \cdot r_2)$, where h is a cryptographic hash function.

In both cases, neither party now has full control over r.

This weakens Bob's ability to launch a chosen plaintext attack if Alice chooses first.

This weakens Mallory's ability to impersonate Alice if Bob chooses first.

Feige-Fiat-Shamir Authentication Protocol

Concept of zero knowledge

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.

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.

Feige-Fiat-Shamir protocol: overview

Alice authenticates herself by successfully completing several rounds of a protocol that requires knowledge of a secret s.

In a single round, 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.

Feige-Fiat-Shamir protocol: preparation

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 $u \in \mathbf{Z}_n^*$ 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.

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

A simplified one-round FFS protocol

Here's a simplified one-round version.

	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 $x = y^2 v^b \bmod n$.

When both parties are honest, Bob accepts Alice because

$$x = y^2 v^b \bmod n.$$

This holds because

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

A dishonest 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 the real Alice.

Theorem

Suppose Mallory doesn't know a square root of v^{-1}. Then Bob's verification will fail with probability at least 1/2.

Proof that Mallory can't successfully cheat

Proof.

In order for Mallory to successfully fool Bob, he must come up with x in step 1 and y in step 3 satisfying

$$x = y^2 v^b \bmod n.$$

Mallory sends x in step 1 before Bob chooses b, so he does not know which value of b to expect.

When Mallory receives b, he responds by sending a value y_b to Bob.

We consider two cases.

(continued...)

Proof: case 1

Proof (continued).

Case 1: There is at least one $b \in \{0, 1\}$ for which y_b fails to satisfy

$$x = y^2 v^b \bmod n.$$

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.

(continued...)

Proof: case 2

Proof (continued).

Case 2: y_0 and y_1 both satisfy the verification equation, so $x = y_0^2 \bmod n$ and $x = y_1^2 v \bmod n$.

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

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

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}. $\square$

Successful cheating with probability 1/2

We remark that it *is* possible for Mallory to cheat with success probability $1/2$.

- He guesses the bit b that Bob will send him in step 2 and 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 done.
- If he guesses $b = 1$, then he chooses y arbitrarily and $x = y^2 v \bmod n$.

He proceeds to send x in step 1 and y in step 3.

The pair (x, y) is accepted by Bob Mallory's guess of b turns out to be correct, which will happen with probability $1/2$.

A dishonest Bob

We now consider the case of a dishonest Mallory impersonating Bob, or simply a dishonest Bob who wants to capture Alice's secret.

Alice would like assurance that her secret is protected if she follows the protocol,regardless of what Mallory (Bob) does.

Consider what Mallory knows at the end of the protocol.

Mallory sends $b = 0$

Suppose Mallory sends $b = 0$ in step 2.

Then he ends up with a pair (x, y), where y is a random number and x is its square modulo n.

Neither of these numbers depend in any way on Alice secret s, so Mallory gets 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 would allow 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.

Mallory sends $b = 1$

Suppose Mallory sends $b = 1$ 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 $r \longrightarrow rs \bmod n$ 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 $x = y^2 v \bmod n$.

Again, all he ends up with is a random number (y in this case) and a quadratic residue x that he can compute knowing y.

Mallory learns nothing from (x, 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.