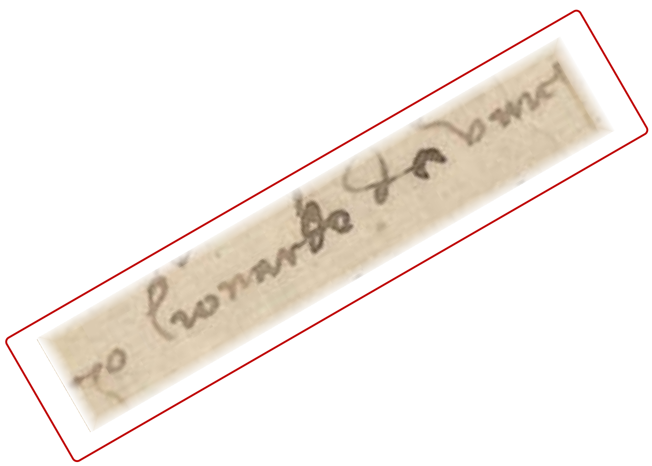
Crypto 10

Note: this material is not intended to replace the live lecture for students.

Contents



10.1 Digital Signatures (DS)

The digital signature provides **message authentication** (the receiver can verify the origin of the message), **integrity** (the receiver can verify that the message has not been modified since it was signed) and **non-repudiation** (the sender cannot falsely claim that they have not signed the message).

10.1.1 Digital Signature scheme

Consists of three probabilistic algorithms ($\mathsf{Gen}(n)$, Sign , Vrfy)

- $\mathsf{Gen}(n)$: the input n is a security parameter and the output is the asymmetric key pair $(\mathsf{pk}, \mathsf{sk})$ both of at least n bit.
- $\mathsf{Sign}_{\mathsf{sk}}(m)$: input a message m and the secrete key sk. The output is the digital signature σ.
- $\mathsf{Vrfy}_{\mathsf{pk}}(m, \sigma)$: input a message m, the public key pk and the digital signature σ; output 1 if "valid signature" or 0 for "invalid signature".

The digital signature scheme is **secure** if somebody who knows pk (but not sk) and lots of valid signatures $(m_1, \sigma_1) \ldots (m_\ell, \sigma_\ell)$ can not produce a new message m and a valid signature σ for it.

Exercise 10.1.2

What is the difference between a MAC and a digital signature?

10.1.3 Digital Signature Protocol

Alice is going to sign a message m by using his secret key sk_A. Bob, actually everybody, knows Alice's public key pk_A generated by using $\mathsf{Gen}(n)$.

Alice		**Bob**
$\sigma = \mathsf{Sign}_{\mathsf{sk}_A}(m)$		
	$\xrightarrow{(m,\sigma)}$	
		check $\mathsf{Vrfy}_{\mathsf{pk}_A}(m,\sigma)$ value

This protocol also satisfies the characteristics we're looking for:

1. The signature is authentic; when Bob verifies the message with Alice's public key, he knows that she signed it.
2. The signature is unforgeable; only Alice knows her private key.
3. The signature is not reusable; the signature is a function of the document and cannot be transferred to any other document.
4. The signed document is unalterable; if there is any alteration to the document, the signature can no longer be verified with Alice's public key.
5. The signature cannot be repudiated. Bob doesn't need Alice's help to verify her signature.

Signing Documents and Timestamps

Actually, Bob can cheat Alice in certain circumstances. He can reuse the document and signature together. This is no problem if Alice signed a contract (what's another copy of the same contract, more or less?), but it can be very exciting if Alice signed a digital check.

Let's say Alice sends Bob a signed digital check for $100. Bob takes the check to the bank, which verifies the signature and moves the money from one account to the other. Bob, who is an unscrupulous character, saves a copy of the digital check. The following week, he again takes it to the bank (or maybe to a different bank). The bank verifies the signature and moves the money from one account to the other. If Alice never balances her checkbook, Bob can keep this up for years.

Consequently, digital signatures often include timestamps. The date and time of the signature are attached to the message and signed along with the rest of the message. The bank stores this timestamp in a database. Now, when Bob tries to cash Alice's check a second time, the bank checks the timestamp against its database. Since the bank already cashed a check from Alice with the same timestamp, the bank calls the police. Bob then spends 15 years in Leavenworth prison reading up on cryptographic protocols.

[Schneier15, page 38]

10.2 RSA digital signature

10.2.1 Naive RSA-signature

Alice's public key $\mathsf{pk}_A = (n, e)$ and secret key is $\mathsf{sk}_A = d$.

Alice		**Bob**
$\sigma = \mathsf{Sign}_{\mathsf{sk}_A}(m) = m^d \pmod n$		
	$\xrightarrow{(m,\sigma)}$	
		$\mathsf{Vrfy}_{\mathsf{pk}_A}(m, \sigma) = 1$ iff $\sigma^e = m \pmod n$

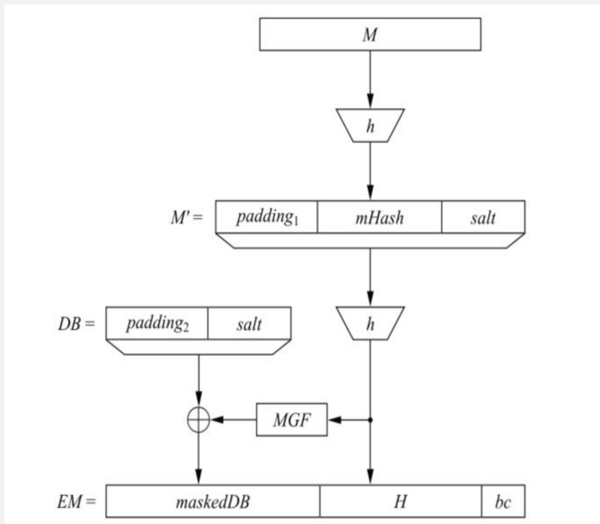
Exercise 10.2.2

Show that the above signature is not secure. Hint: choose any $\sigma \in \mathbb{Z}_n$ and consider $m = \sigma^e \pmod n$.

10.2.1 EMSA-PSS - RSA digital signature

Encoding Method for Signature with Appendix (EMSA) Probabilistic Signature Scheme (PSS). Public-Key Cryptography Standards (PKCS) #1

Figure 10.2.3: EMSA-PSS



After this RSA is used as explained before:

$$\sigma = \mathsf{Sign}_{\mathsf{sk}}(EM) = EM^d \pmod n$$

By adding a random *salt* the encoding is probabilistic i.e. signing twice the same message produce different signatures.
The Vrfy algorithm first recover the *salt* value and then check that the EMSA-PSS encoding is correct.
The receiver knows the values $padding_1, padding_2$ from the standard (see [PKCS, page 41])

$$padding_1 = \text{(0x)00 00 00 00 00 00 00 00}$$

Exercise 10.2.4

Find $padding_2$ in the standard [PKCS]

10.3 Digital Signature Algorithm (DSA)

It is federal US government standard for digital signatures (DSS) and was proposed by the NIST in 1991. DSA was patented (now expired) by David W. Kravitz

> Before it gets too confusing, let me review the nomenclature: DSA is the algorithm; the DSS is the standard. The standard employs the algorithm. The algorithm is part of the standard.

[Schneier15, page 484]

10.3.1 DSA Key Gen and parameters

1. Generate a prime p with $2^{1023} < p < 2^{1024}$.
2. Find a prime divisor q of $p-1$ with $2^{159} < q < 2^{160}$.
3. Find an element α with $\mathrm{ord}(\alpha) = q$, i.e., α generates the subgroup with q elements.
4. Choose a random integer d with $0 < d < q$.
5. Compute $\beta \equiv \alpha^d \bmod p$.

The keys are now:
$k_{pub} = (p, q, \alpha, \beta)$
$k_{pr} = (d)$

NOTE 10.3.2

The primes p, q and the generator α are parameters domains and can be common across a network of users.

10.3.3 DSA $\mathsf{Sign}(x) = (r, s)$

1. Choose an integer as random ephemeral key k_E with $0 < k_E < q$.
2. Compute $r \equiv (\alpha^{k_E} \bmod p) \bmod q$.
3. Compute $s \equiv (SHA(x) + d \cdot r) {k_E}^{-1} \bmod q$.

10.3.4 DSA $\mathsf{Vrfy}(x, (r, s))$

1. Compute auxiliary value $w \equiv s^{-1} \bmod q$.
2. Compute auxiliary value $u_1 \equiv w \cdot SHA(x) \bmod q$.
3. Compute auxiliary value $u_2 \equiv w \cdot r \bmod q$.
4. Compute $v \equiv (\alpha^{u_1} \cdot \beta^{u_2} \bmod p) \bmod q$.
5. The verification $ver_{k_{pub}}(x, (r, s))$ follows from:

$$v \begin{cases} \equiv r \bmod q \Longrightarrow \text{valid signature} \\ \not\equiv r \bmod q \Longrightarrow \text{invalid signature} \end{cases}$$

NOTE 10.3.5

So DSA is based in two cyclic groups and on the DLP. The total signature is of 320 bit. According to the literature it has 80 bit of security level. To increase the security level read NIST suggestions at [Paar10, page 282].

Exercise 10.3.6

Set $p = 59$, $q = 29$, $\alpha = 3$, $d = 7$, $\beta = \alpha^d \pmod{59}$. Assuming that $SHA(x) = 26$ compute the DSA signature (r, s).

Exercise 10.3.7

If a cryptanalyst can predict the random ephemeral k_E then DSA is broken.

In 2010, a non secure implementation of the PRNG used to generate k_E allows an attack to the PlayStation Sony PS3!

Exercise 10.3.8

Consider a variant of DSA in which just messages in $\mathbb{Z}_q$ are signed and the Hash function is omitted i.e. to get the sign (r, s) of $m \in \mathbb{Z}_q$ we compute $r = \alpha^{k_E}$ (as usual) but $s = (m + dr){k_E}^{-1}(\mathrm{mod} q)$. Is this secure?

NIST recommend a specific method for generating p and q by using a SHA hash function as PRNG [Schneier15, page 489]. Below a slightly different generator taken from [Paar10, page 280].

10.3.9

Prime Generation for DSA
Output: two primes (p, q), where $2^{1023} < p < 2^{1024}$ and $2^{159} < q < 2^{160}$, such that $p - 1$ is a multiple of q.
Initialization: $i = 1$
Algorithm:

```
1     find prime q with 2^159 < q < 2^160 using the Miller–Rabin algorithm
2     FOR i = 1 TO 4096
2.1       generate random integer M with 2^1023 < M < 2^1024
2.2       M_r ≡ M  mod 2q
2.3       p − 1 ≡ M − M_r          (note that p − 1 is a multiple of 2q.)
          IF p is prime            (use Miller–Rabin primality test)
2.4           RETURN (p, q)
2.5       i = i + 1
3     GOTO Step 1
```

The point of this exercise is that there is a public means of generating p and q. For all practical purposes, this method prevents cooked values of p and q. If someone hands you a p and a q, you might wonder where that person got them. However, if someone hands you a value for S and C that generated the random p and q, you can go through this routine yourself. Using a one-way hash function, SHA in the standard, prevents someone from working backwards from a p and q to generate an S and C.

This security is better than what you get with RSA. In RSA, the prime numbers are kept secret. Someone could generate a fake prime or one of a special form that makes factoring easier. Unless you know the private key, you won't know that. Here, even if you don't know a person's private key, you can confirm that p and q have been generated randomly.

[Schneier15, page 490]

10.4 Schnorr Signatures

10.4.1 Schnorr DS

Algorithm

Choosing parameters

- All users of the signature scheme agree on a group, G, of prime order, q, with generator, g, in which the discrete log problem is assumed to be hard. Typically a Schnorr group is used.
- All users agree on a cryptographic hash function $H : \{0,1\}^* \to \mathbb{Z}_q$.

Notation

In the following,

- Exponentiation stands for repeated application of the group operation
- Juxtaposition stands for multiplication on the set of congruence classes or application of the group operation (as applicable)
- Subtraction stands for subtraction on set of equivalence groups
- $M \in \{0,1\}^*$, the set of finite bit strings
- $s, e, e_v \in \mathbb{Z}_q$, the set of congruence classes modulo q
- $x, k \in \mathbb{Z}_q^\times$, the multiplicative group of integers modulo q (for prime q, $\mathbb{Z}_q^\times = \mathbb{Z}_q \setminus \bar{0}_q$)
- $y, r, r_v \in G$.

Key generation

- Choose a private signing key, x, from the allowed set.
- The public verification key is $y = g^x$.

Signing

To sign a message, M:

- Choose a random k from the allowed set.
- Let $r = g^k$.
- Let $e = H(r \parallel M)$, where $\parallel$ denotes concatenation and r is represented as a bit string.
- Let $s = k - xe$.

The signature is the pair, (s, e).

Note that $s, e \in \mathbb{Z}_q$; if $q < 2^{160}$, then the signature representation can fit into 40 bytes.

Verifying

- Let $r_v = g^s y^e$
- Let $e_v = H(r_v \parallel M)$

If $e_v = e$ then the signature is verified.

https://en.wikipedia.org/wiki/Schnorr_signature
https://en.bitcoin.it/wiki/Schnorr

10.5 DS Protocols

10.5.1 Subliminal Channel

Alice and Bob have been arrested and are going to prison. He's going to the men's prison and she's going to the women's prison. Walter, the warden, is willing to let Alice and Bob exchange messages, but he won't allow them to be encrypted. Walter expects them to coordinate an escape plan, so he wants to be able to read everything they say.

Walter also hopes to deccive either Alice or Bob. He wants one of them to accept a fraudulent message as a genuine message from the other. Alice and Bob go along with this risk of deception, otherwise they cannot communicate at all, and they have to coordinate thcir plans. To do this they have to deceive the warden and find a way of communicating secretly. They have to set up a **subliminal channel**, a covert communications channel between them in full view of Walter, even though the messages themselvcs contain no secret information. Through the exchange of perfectly innocuous signed messages they will pass secret information back and forth and fool Walter, even though Walter is watching all the communications.

[Schneier15, page 79]

10.5.1 by using DSA, after Gustavus J. Simmons

Alice and **Bob** share a DSA private key d associated to the public one (p, q, α, β) **Alice** wants to communicate an integer $\mathbf{k} \in \mathrm{GF}^*(q)$ to **Bob**.

She signs any innocuous message x using DSA with ephemeral key $K_E = \mathbf{k}$. Namely

Alice		**Walter**		**Bob**
$r = (\alpha^{\mathbf{k}} \pmod p)$ $s = (SHA(x) + d \cdot r)\mathbf{k}^{-1}$				
	$\xrightarrow{(x,(r,s))}$			
		$(x, (r, s))$	$\xrightarrow{(x,(r,s))}$	
				$(x, (r, s))$ $\mathbf{k} = (SHA(x) + d \cdot r)s^{-1}$

10.5.2 Blind signature

These concept was introduced by David Chaum in 1983.

Automation of the way we pay for goods and services is already underway, as can be seen by the variety and growth of electronic banking services available to consumers. The ultimate structure of the new electronic payments system may have a substantial impact on personal privacy as well as on the nature and extent of criminal use of payments. Ideally a new payments system should address both of these seemingly conflicting sets of concerns.

On the one hand, knowledge by a third party of the payee, amount, and time of payment for every transaction made by an individual can reveal a great deal about the individual's whereabouts, associations and lifestyle. For example, consider payments for such things as transportation, hotels, restaurants, movies, theater, lectures, food, pharmaceuticals, alcohol, books, periodicals, dues, religious and political contributions.

On the other hand, an anonymous payments systems like bank notes and coins suffers from lack of controls and security. For example, consider problems such as lack of proof of payment, theft of payments media, and black payments for bribes, tax evasion, and black markets.

A fundamentally new kind of cryptography is proposed here, which allows an automated payments system with the following properties:

(1) Inability of third parties to determine payee, time or amount of payments made by an individual.

(2) Ability of individuals to provide proof of payment, or to determine the identity of the payee under exceptional circumstances.

(3) Ability to stop use of payments media reported stolen.

[Chaum83, Introduction]

Exercise 10.5.2

Read page 200 of Chaum's paper [Chaum83] and make a diagram of the trustee/elector blind signature scheme.

10.5.3

Completely Blind Signatures

Bob is a notary public. Alicc wants him to sign a document, but does not want him to have any idea what he is signing. Bob doesn't care what the document says; he is just certifying that he notarized it at a certain time. He is willing to go along with this.

(1) Alice takes the document and multiplies it by a random value. This random value is called a **blinding factor**.

(2) Alice sends the blinded document to Bob.

(3) Bob signs the blinded document.

(4) Alice divides out the blinding factor, leaving the original document signed by Bob.

Blind Signatures

With the completely blind signature protocol, Alice can have Bob sign anything: "Bob owcs Alice a million dollars," "Bob owcs Alice his first-born child," "Bob owcs Alice a bag of chocolates." The possibilities are endless. This protocol isn't useful in many applications.

10.5.4 RSA blind signature

Bob has a public key e, a secret key d and a public modulus n.
Alice wants **Bob** to sign message m blindly.

(1) Alice chooses a random value, k, between 1 and n. Then she blinds m by computing

$$t = mk^e \bmod n$$

(2) Bob signs t

$$t^d = (mk^e)^d \bmod n$$

(3) Alice unblinds t^d by computing

$$s = t^d/k \bmod n$$

(4) And the result is

$$s = m^d \bmod n$$

10.6 Bibliography

Books I used to prepare this note:

[KatLin15] Jonathan Katz; Yehuda Lindell, *Introduction to Modern Cryptography* Second Edition, Chapman & Hall/CRC, Taylor & Francis Group, 2015.

[Paar10] Paar, Christof, Pelzl, Jan, *Understanding Cryptography, A Textbook for Students and Practitioners*, Springer-Verlag, 2010.

[Schneier15] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C* ,Wiley; 20th Anniversary edition,2015.

Here a list of papers:

[PKCS] *Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography* https://tools.ietf.org/html/rfc3447

[Chaum83] David Chaum; *Blind signatures for untraceable payments*, Advances in Cryptology Proceedings of Crypto. 82 (3): 199–203. http://www.hit.bme.hu/~buttyan/courses/BMEVIHIM219/2009/Chaum.BlindSigForPayment.1982.PDF

[Schno89] C.P. Schnorr; *Efficient Identification and Signatures for Smart Cards*, Proceedings of CRYPTO '89. https://link.springer.com/content/pdf/10.1007%2F0-387-34805-0_22.pdf

and some interesting links:

https://tools.ietf.org/html/rfc3447#ref-31

https://en.wikipedia.org/wiki/Blind_signature#Dangers_of_blind_signing