COM/BLM 426
Information Security

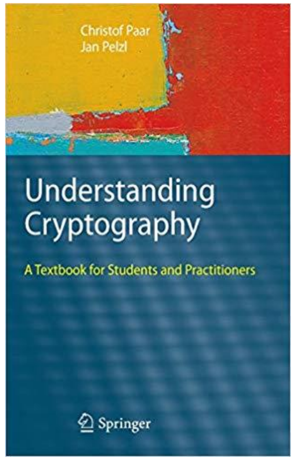
Asst. Prof. Dr. Bulent TUGRUL
btugrul@eng.ankara.edu.tr

Slides are mainly based on:

Understanding Cryptography: A Textbook for Students and Practitioners

by Christof Paar and Jan Pelzl

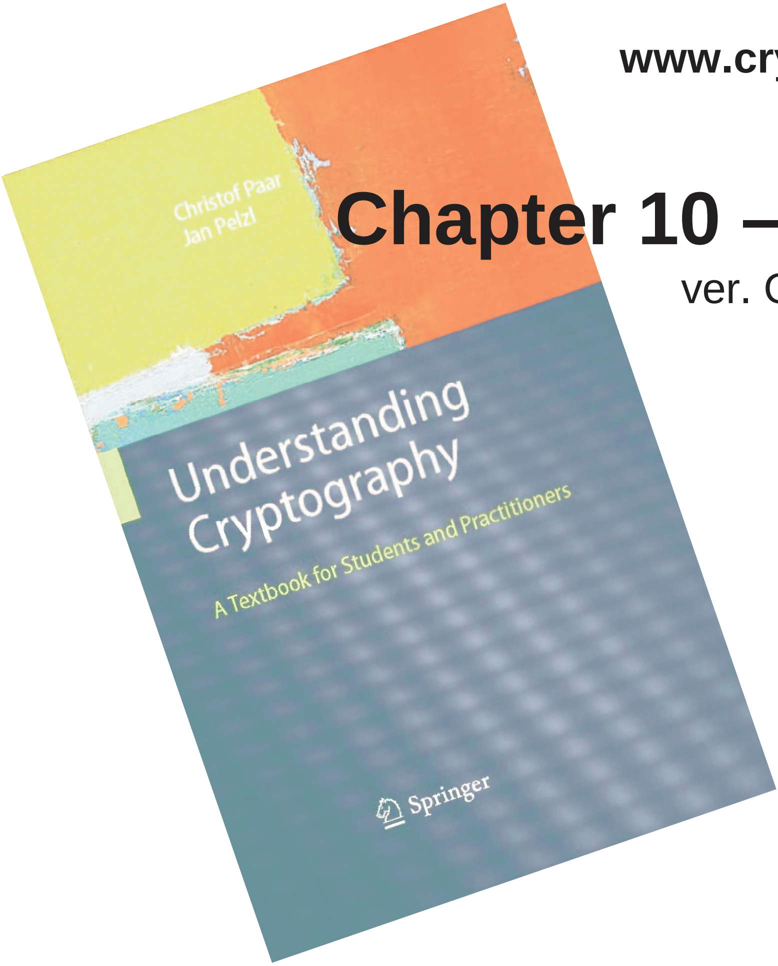
Springer, 1st Edition, 2010



Understanding Cryptography – A Textbook for Students and Practitioners

by Christof Paar and Jan Pelzl

www.crypto-textbook.com



Chapter 10 – Digital Signatures

ver. October 29, 2009

These slides were prepared by Georg Becker, Christof Paar and Jan Pelzl

Some legal stuff (sorry): Terms of Use

- The slides can used free of charge. All copyrights for the slides remain with Christof Paar and Jan Pelzl.
- The title of the accompanying book “Understanding Cryptography” by Springer and the author’s names must remain on each slide.
- If the slides are modified, appropriate credits to the book authors and the book title must remain within the slides.
- It is not permitted to reproduce parts or all of the slides in printed form whatsoever without written consent by the authors.

Content of this Chapter

- The principle of digital signatures
- Security services
- The RSA digital signature scheme
- The Digital Signature Algorithm (DSA)

Content of this Chapter

- **The principle of digital signatures**
- Security services
- The RSA digital signature scheme
- The Digital Signature Algorithm (DSA)

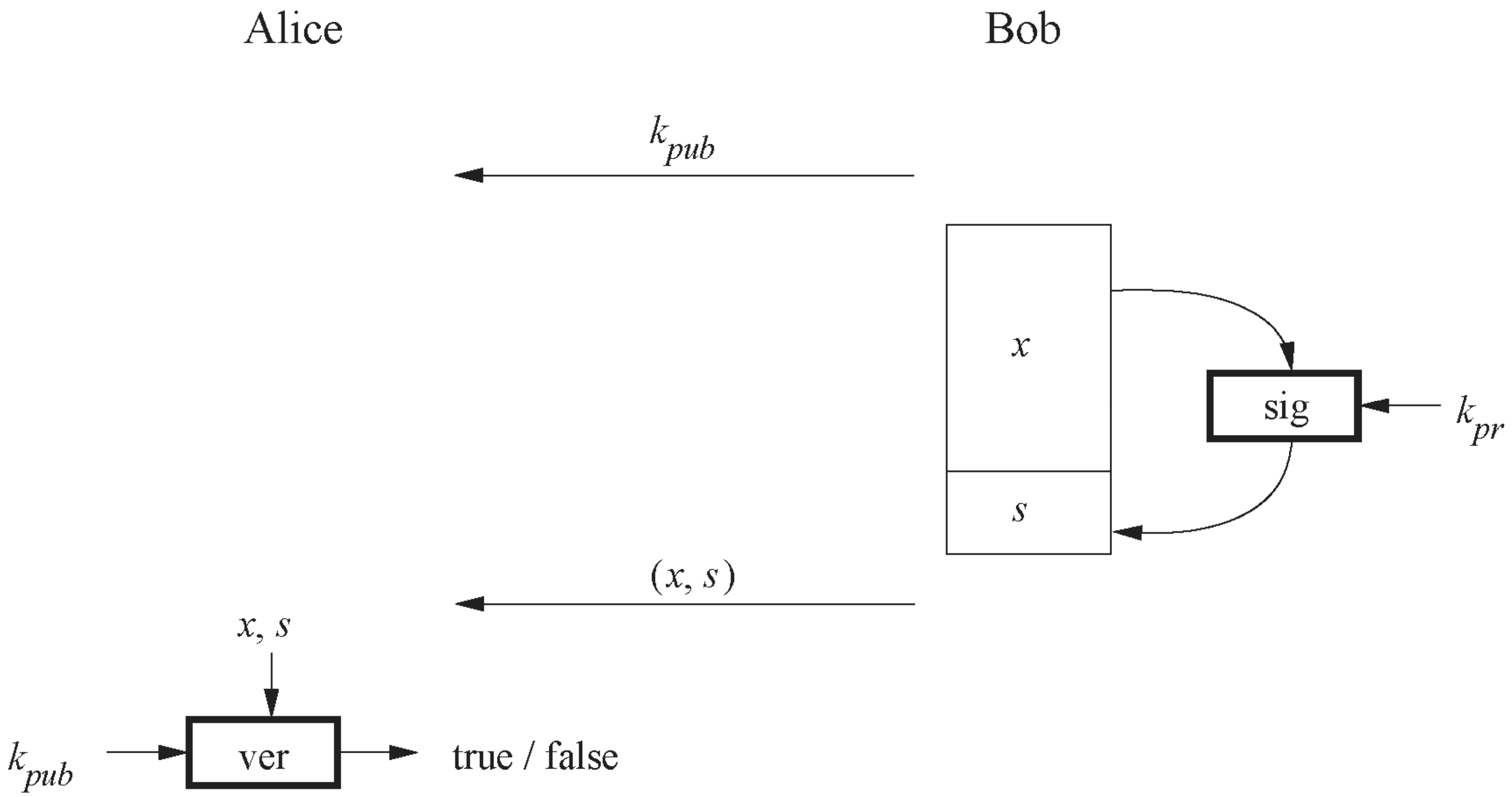
Motivation

- Alice orders a pink car from the car salesmen Bob
- After seeing the pink car, Alice states that she has never ordered it:
- How can Bob prove towards a judge that Alice has ordered a pink car? (And that he did not fabricate the order himself)

$\Longrightarrow$ Symmetric cryptography fails because both Alice and Bob can be malicious

$\Longrightarrow$ Can be achieved with public-key cryptography

Basic Principle of Digital Signatures



Main idea

- For a given message x, a digital signature is appended to the message (just like a conventional signature).
- Only the person with the private key should be able to generate the signature.
- The signature must change for every document.

$\Longrightarrow$ The signature is realized as a function with the message x and the private key as input.

$\Longrightarrow$ The public key and the message x are the inputs to the verification function.

Content of this Chapter

- The principle of digital signatures
- **Security services**
- The RSA digital signature scheme
- The Digital Signature Algorithm (DSA)

Core Security Services

The objectives of a security systems are called *security services*.

1. **Confidentiality**: Information is kept secret from all but authorized parties.
2. **Integrity:** Ensures that a message has not been modified in transit.
3. **Message Authentication:** Ensures that the sender of a message is authentic. An alternative term is data origin authentication.
4. **Non-repudiation:** Ensures that the sender of a message can not deny the creation of the message. (c.f. order of a pink car)

Additional Security Services

5. **Identification/entity authentication:** Establishing and verification of the identity of an entity, e.g. a person, a computer, or a credit card.
6. **Access control:** Restricting access to the resources to privileged entities.
7. **Availability:** The electronic system is reliably available.
8. **Auditing:** Provides evidences about security relevant activities, e.g., by keeping logs about certain events.
9. **Physical security:** Providing protection against physical tampering and/or responses to physical tampering attempts
10. **Anonymity:** Providing protection against discovery and misuse of identity.

Content of this Chapter

- The principle of digital signatures
- Security services
- **The RSA digital signature scheme**
- The Digital Signature Algorithm (DSA)

Main idea of the RSA signature scheme

To generate the private and public key:

- Use the same key generation as RSA encryption.

To generate the signature:

- "encrypt" the message *x* with the private key

$$\boxed{s = sig_{K_{priv}}(x) = x^d \bmod n}$$

- Append *s* to message *x*

To verify the signature:

- "decrypt" the signature with the public key

$$\boxed{x' = ver_{K_{pub}}(s) = s^e \bmod n}$$

- If $x=x'$, the signature is valid

The RSA Signature Protocol

Alice		Bob
	$\xleftarrow{\quad K_{pub} \quad}$	$K_{pr} = d$ $K_{pub} = (n, e)$
		Compute signature: $s = sig_{k_{pr}}(x) \equiv x^d \bmod n$
	$\xleftarrow{\quad (x,s) \quad}$	
Verify signature: $x' \equiv s^e \bmod n$ If $x' \equiv x \bmod n \rightarrow$ valid signature If $x' \not\equiv x \bmod n \rightarrow$ invalid signature		

Security and Performance of the RSA Signature Scheme

Security:

The same constrains as RSA encryption: n needs to be at least 1024 bits to provide a security level of 80 bit.

$\Rightarrow$ The signature, consisting of s, needs to be at least 1024 bits long

Performance:

The signing process is an exponentiation with the private key and the verification process an exponentiation with the public key e.

$\Rightarrow$ Signature verification is very efficient as a small number can be chosen for the public key.

Existential Forgery Attack against RSA Digital Signature

Alice	Oscar	Bob
		$K_{pr} = d$ $K_{pub} = (n, e)$
$\xleftarrow{(n,e)}$	$\xleftarrow{(n,e)}$	
	1. Choose signature: $s \in Z_n$	
	2. Compute message: $x \equiv s^e \bmod n$	
$\xleftarrow{(x,s)}$		

Verification:
$s^e \equiv x' \bmod n$

since $s^e = (x^d)^e \equiv x \bmod n$
$\rightarrow$ Signature is valid

Existential Forgery and Padding

- An attacker can generate valid message-signature pairs (*x*,*s*)
- But an attack can only choose the signature *s* and NOT the message *x*

⇒ Attacker cannot generate messages like „Transfer $1000 into Oscar's account"

Formatting the message *x* according to a *padding scheme* can be used to make sure that an attacker cannot generate valid (*x*,*s*) pairs.

(A messages *x* generated by an attacker during an Existential Forgery Attack will not coincide with the padding scheme. For more details see Chapter 10 in *Understanding Cryptography.)*

Content of this Chapter

- The principle of digital signatures
- Security services
- The RSA digital signature scheme
- **The Digital Signature Algorithm (DSA)**

Facts about the Digital Signature Algorithm (DSA)

- Federal US Government standard for digital signatures (DSS)
- Proposed by the National Institute of Standards and Technology (NIST)
- DSA is based on the Elgamal signature scheme
- Signature is only 320 bits long
- Signature verification is slower compared to RSA

The Digital Signature Algorithm (DSA)

Key generation of DSA:

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 integer α with $\mathrm{ord}(\alpha)=q$
4. Choose a random integer d with $0<d<q$
5. Compute $\beta \equiv \alpha^d \bmod p$

The keys are:

$k_{pub} = (p,q,\alpha,\beta)$

$k_{pr} = (d)$

The Digital Signature Algorithm (DSA)

DSA signature generation :

Given: message x, signature s, private key d and public key (p,q,α,β)

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. Computes $s \equiv (SHA(x)+d \cdot r)\, k_E^{-1} \bmod q$

The signature consists of (r,s)

SHA denotes the hashfunction SHA-1 which computes a 160-bit fingerprint of message x. (See Chapter 11 of *Understanding Cryptography* for more details)

The Digital Signature Algorithm (DSA)

DSA signature verification

Given: message x, signature s and public key (p,q,α,β)

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 $v \equiv (\alpha^{u_1} \cdot \beta^{u_2} \bmod p) \bmod q$

If $v \equiv r \bmod q \rightarrow$ signature is valid

If $v \not\equiv r \bmod q \rightarrow$ signature is invalid

Proof of DSA:

We show need to show that the signature (*r,s*) in fact satisfied the condition *r* ≡ *v* mod *q*:

$s \equiv (\mathrm{SHA}(x))+d \cdot r) \cdot k_E^{-1} \bmod q$

⇔ $k_E \equiv s^{-1}\,\mathrm{SHA}(x) + d \cdot s^{-1} r \bmod q$

⇔ $k_E \equiv u_1 + d \cdot u_2 \bmod q$

We can raise α to either side of the equation if we reduce modulo *p*:

⇔ $\alpha^{k_E} \bmod p \equiv \alpha^{u_1 + d \cdot u_2} \bmod p$

Since $\beta \equiv \alpha^d \bmod p$ we can write:

⇔ $\alpha^{k_E} \bmod p \equiv \alpha^{u_1} \beta^{u_2} \bmod p$

We now reduce both sides of the equation modulo q:

⇔ $(\alpha^{k_E} \bmod p) \bmod q \equiv (\alpha^{u_1} \beta^{u_2} \bmod p) \bmod q$

Since $r \equiv \alpha^{k_E} \bmod p \bmod q$ and $v \equiv (\alpha^{u_1} \beta^{u_2} \bmod p) \bmod q$, this expression is identical to:

⇔ $r \equiv v$

Example

Alice

Bob

Key generation:

1. choose p = 59 and q = 29
2. choose α = 3
3. choose private key d = 7
4. $\beta = \alpha^{\beta} = 3^7 \equiv 4 \bmod 59$

$(p, q, \alpha, \beta)=(59, 29, 3, 4)$ ←

Sign:

Compute has of message H(x)=26

1. Choose ephermal key k_E=10
2. $r = (3^{10} \bmod 59) \equiv 20 \bmod 29$
3. $s = (26 + 7 \cdot 20) \cdot 3) \equiv 5 \bmod 29$

$(x,(r, s))=(x,20, 5)$ ←

Verify:

$w \equiv 5^{-1} \equiv 6 \bmod 29$

$u_1 \equiv 6 \cdot 26 \equiv 11 \bmod 29$

$u_2 \equiv 6 \cdot 20 \equiv 4 \bmod 29$

$v = (3^{11} \cdot 4^4 \bmod 59) \bmod 29 = 20$

$v \equiv$ r mod 29 → valid signature

Security of DSA

To solve the discrete logarithm problem in *p* the powerful index calculus method can be applied. But this method cannot be applied to the discrete logarithm problem of the subgroup *q*. Therefore *q* can be smaller than *p*. For details see Chapter 10 and Chapter 8 of *Understanding Cryptography* .

p	q	hash output (min)	security levels
1024	160	160	80
2048	224	224	112
3072	256	256	128

Standardized parameter bit lengths and security levels for the DSA

Elliptic Curve Digital Signature Algorithm (ECDSA)

- Based on Elliptic Curve Cryptography (ECC)
- Bit lengths in the range of 160-256 bits can be chosen to provide security equivalent to 1024-3072 bit RSA (80-128 bit symmetric security level)
- One signature consists of two points, hence the signature is twice the used bit length (i.e., 320-512 bits for 80-128 bit security level).
- The shorter bit length of ECDSA often result in shorter processing time

For more details see Section 10.5 in *Understanding Cryptography*

Lessons Learned

- Digital signatures provide message integrity, message authentication and non-repudiation.
- RSA is currently the most widely used digital signature algorithm.
- Competitors are the Digital Signature Standard (DSA) and the Elliptic Curve Digital Signature Standard (ECDSA).
- RSA verification can be done with short public keys *e.* Hence, in practice, RSA verification is usually faster than signing.
- DSA and ECDSA have shorter signatures than RSA
- In order to prevent certain attacks, RSA should be used with padding.
- The modulus of DSA and the RSA signature schemes should be at least 1024-bits long. For true long-term security, a modulus of length 3072 bits should be chosen. In contrast, ECDSA achieves the same security levels with bit lengths in the range 160–256 bits.