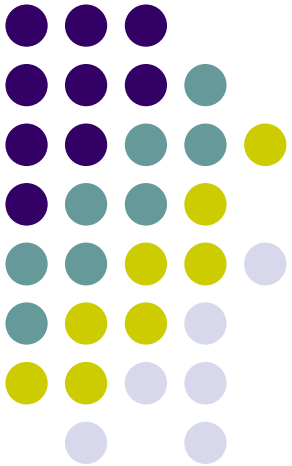
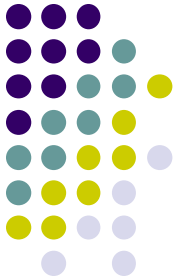
Standards digital signatures & certificates

Zdeněk Říha

Marek Sýs



Digital Signature Standard



- Selection of Parameter Sizes and Hash Functions

- Domain Parameter Generation
  - only for DSA, ECDSA
- Signature Generation

- Signature Verification and Validation

Digital Signature Standard (DSS)

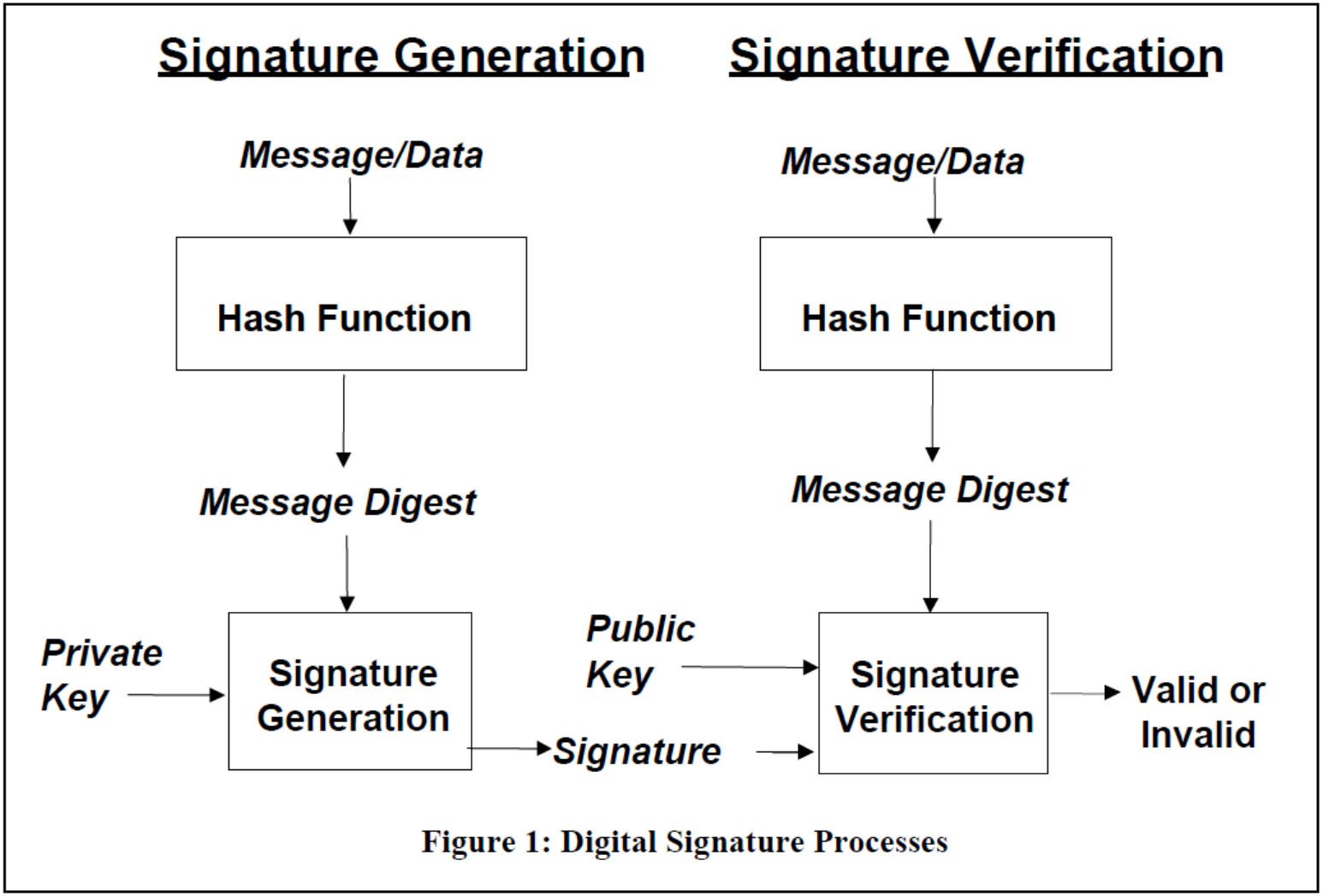


Figure 1: Digital Signature Processes

DSA
Domain parameters

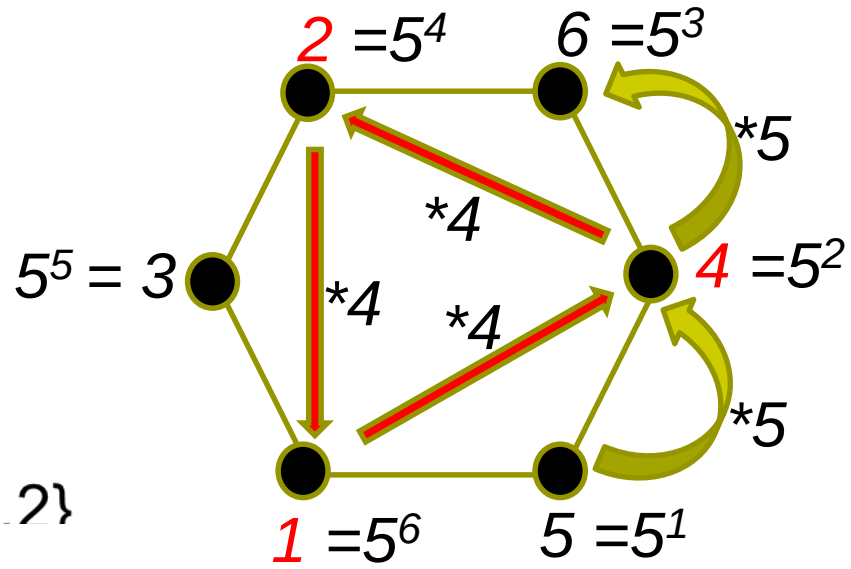
- Generation described in Appendix A.1 and A.2
  - *p* a prime modulus *L* is the bit length of *p*. (p = 7)
  - *q* a prime divisor of (*p* – 1), (q = 3)
  - *g* a generator of a subgroup of order q (g = 4)

G – generator of group Z_p^*

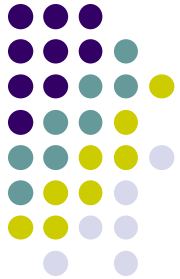
g = $G^{(p-1)/q}$

5 generates Z_7^* i.e.

5^2 =4 generates subgroup {1,4,2}



DSA
Signature generation



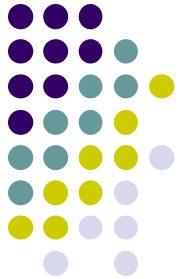
- k – secret number unique to each message
- $r = (g^k \bmod p) \bmod q$.
- z = the leftmost **min**(N, *outlen*) bits of **Hash**(M).

string z obtained from **Hash**(M) **shall** be converted to an integer. The conversion rule is provided in Appendix C.2.

- $s = (k^{-1}(z + xr)) \bmod q$.

Signature = [r, s]

DSA
Signature verification



- $w = (s')^{-1} \bmod q$.
- z = the leftmost **min**(N, *outlen*) bits of **Hash**(M').
- $u1 = (zw) \bmod q$.
- $u2 = ((r')w) \bmod q$.
- $v = (((g)^{u1} (y)^{u2}) \bmod p) \bmod q$.

If v = r signature is verified (see Appendix E)

else signature **shall** be considered invalid.

ECDSA
Domain parameters

- Recommended curves (points [x,y]) defined by

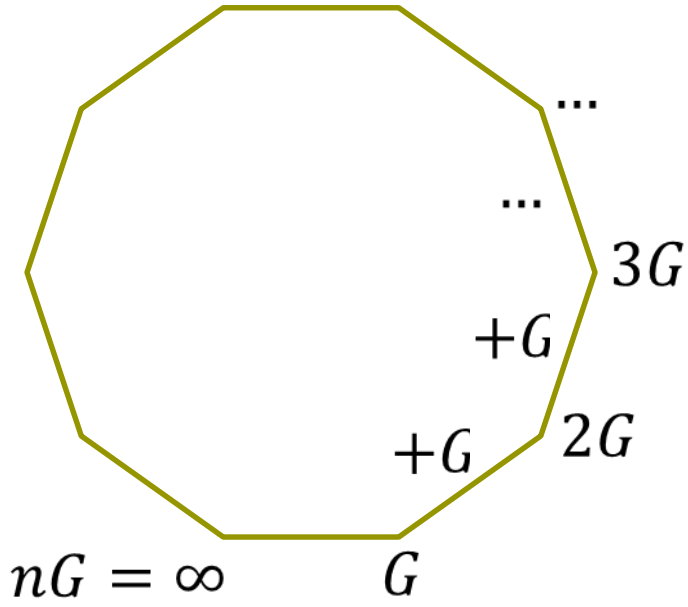
$$y^2 \equiv x^3 - 3x + \boldsymbol{b} \pmod{\boldsymbol{p}}$$

or in finite fields.

n – order of curve

Base point G=[G_x,G_y]

(generator)

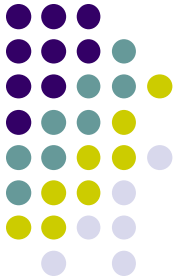


Security of RSA

- We choose randomly 2 primes and compute n and φ(n) :
  - **p, q**
  - **n = p·q**
  - **φ(n) = (p-1)(q-1).**
- **e** is chosen such that **gcd(e, φ(n)) = 1.**
- We compute **$d = e^{-1} \pmod{\varphi(n)}$.**
- Public key: **n, e.**
  Private parameters: **p, q, d.**
  Private key: **d.**

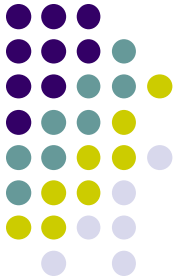
- Security of RSA cryptosystem is based on the problem of factoring large numbers
- If public **n** can be factored into **p** and **q**, we can calculate **φ(n)** and derive d from e.
- Integer factorization is taught at primary schools
- But when integers are very big it takes very long time even for fast computers to factor the number

Computational Security



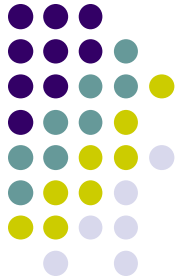
- Unconditional vs. computational security
- Security based on a hard problem
- The problem is solvable, but it takes impractically long time to solve
- The attacker cannot wait thousands/millions of years to break the encryption
- Our expectations can change:
  - Progress in the speed of HW
  - Progress in the efficiency of algorithms

History of RSA Security



- RSA is considered secure
  - But the key size does matter
- 1977: published in “Scientific American”
  - RSA-129 (129 decimal digits of modulus n)
  - Challenge of 100 dollars
  - 40 quadrillion years estimated to factor …
  - Factored in 1994
    - “The magic words are squeamish ossifrage.”

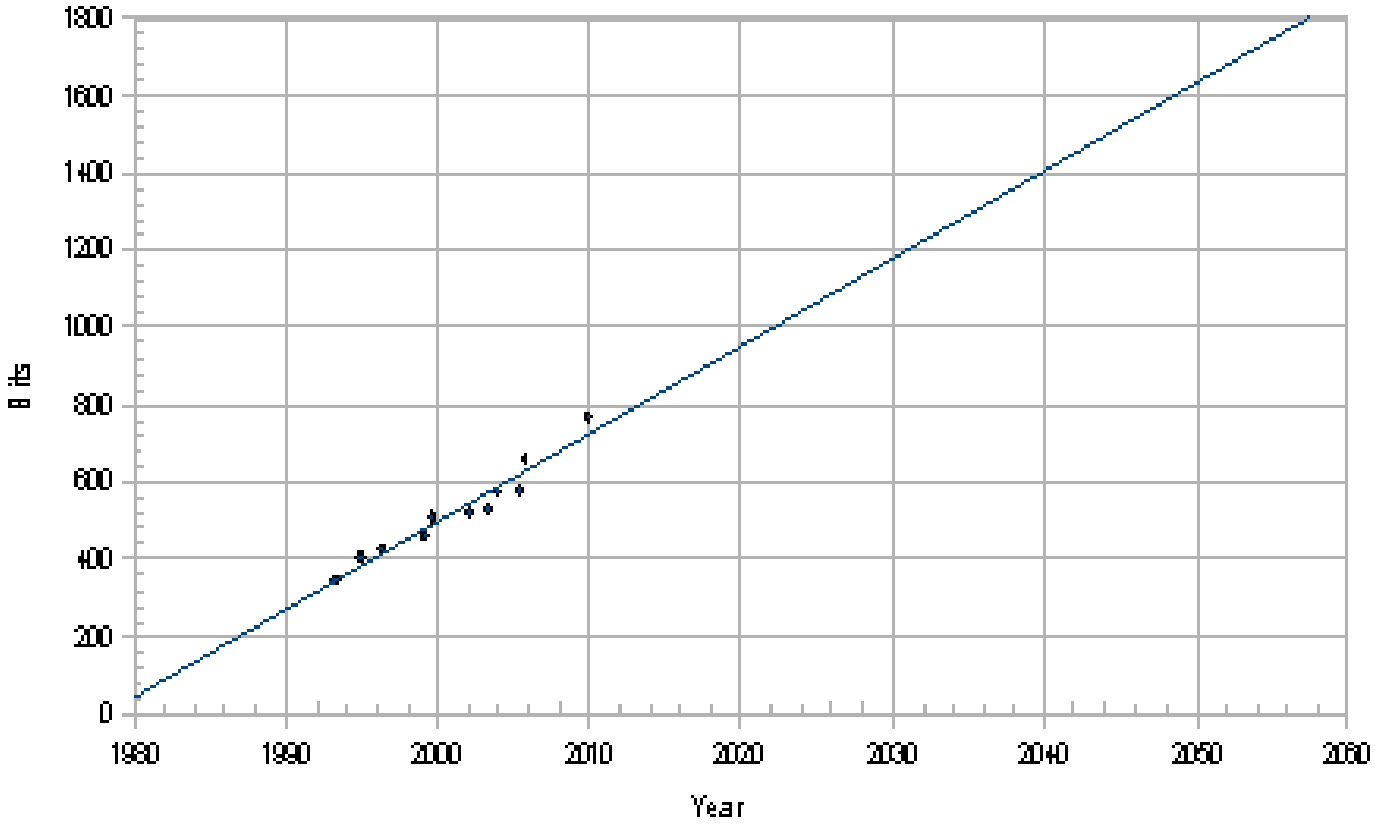
History of RSA Security II



- 1999
  - 512 bit integer was factorized
- 2005
  - 663 bit integer was factorized
- January 2010
  - 768 bit integer was factorized

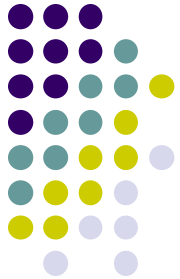
- 1024 bit integers are (probably) not factorable at the moment

Security of RSA



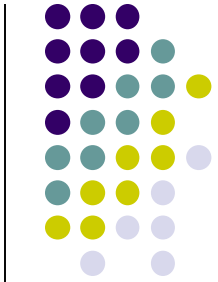
Source: P. Layland, RSA Security and Integer Factorization: The Thirty Years War from 1990 to 2020, IS2 2010, Praha

Key size



- Algorithms are public & keys must be secret
- Key must be large enough that a brute force attack is infeasible
- Depending on the algorithm used it is common to have different key sizes for the same level of security
  - Representing the level of security – number of combinations needed for the brute force attack
  - E.g. 1024 bit RSA key equivalent to 80 bit symmetric encryption key

Comparable strengths of cryptosystems



Bits of security	Symmetric key algorithms	FFC (e.g., DSA, D-H)	IFC (e.g., RSA)	ECC (e.g., ECDSA)
80	2TDEA[19]	$L = 1024$ $N = 160$	$k = 1024$	$f = 160\text{-}223$
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

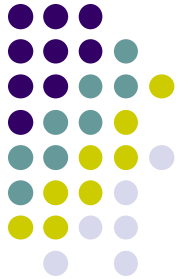
Source:
NIST SP800-57

Security strengths of hash functions

Security Strength	Digital Signatures and hash-only applications	HMAC	Key Derivation Functions[19]	Random Number Generation[20]
80	SHA-1[21], SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-512/224, SHA-224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512-/256, SHA-384, SHA-512
112	SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512
128	SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-1, SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512

Source:
NIST SP800-57

Security strengths of hash functions



Security Strength	Digital Signatures and hash-only applications	HMAC	Key Derivation Functions[19]	Random Number Generation[20]
192	SHA-384, SHA-512	SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-224, SHA-512/224, SHA-256, SHA-512/256, SHA-384, SHA-512
256	SHA-512	SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-256, SHA-512/256, SHA-384, SHA-512	SHA-256, SHA-512/256, SHA-384, SHA-512

Source:
NIST SP800-57

Recommended key sizes

Security Strength		2011 through 2013	2014 through 2030	2031 and Beyond
80	Applying	Deprecated	Disallowed	
	Processing	Legacy use		
112	Applying	Acceptable	Acceptable	Disallowed
	Processing			Legacy use
128	Applying/Processing	Acceptable	Acceptable	Acceptable
192		Acceptable	Acceptable	Acceptable
256		Acceptable	Acceptable	Acceptable

“Acceptable” indicates that the algorithm or key length is not known to be insecure.

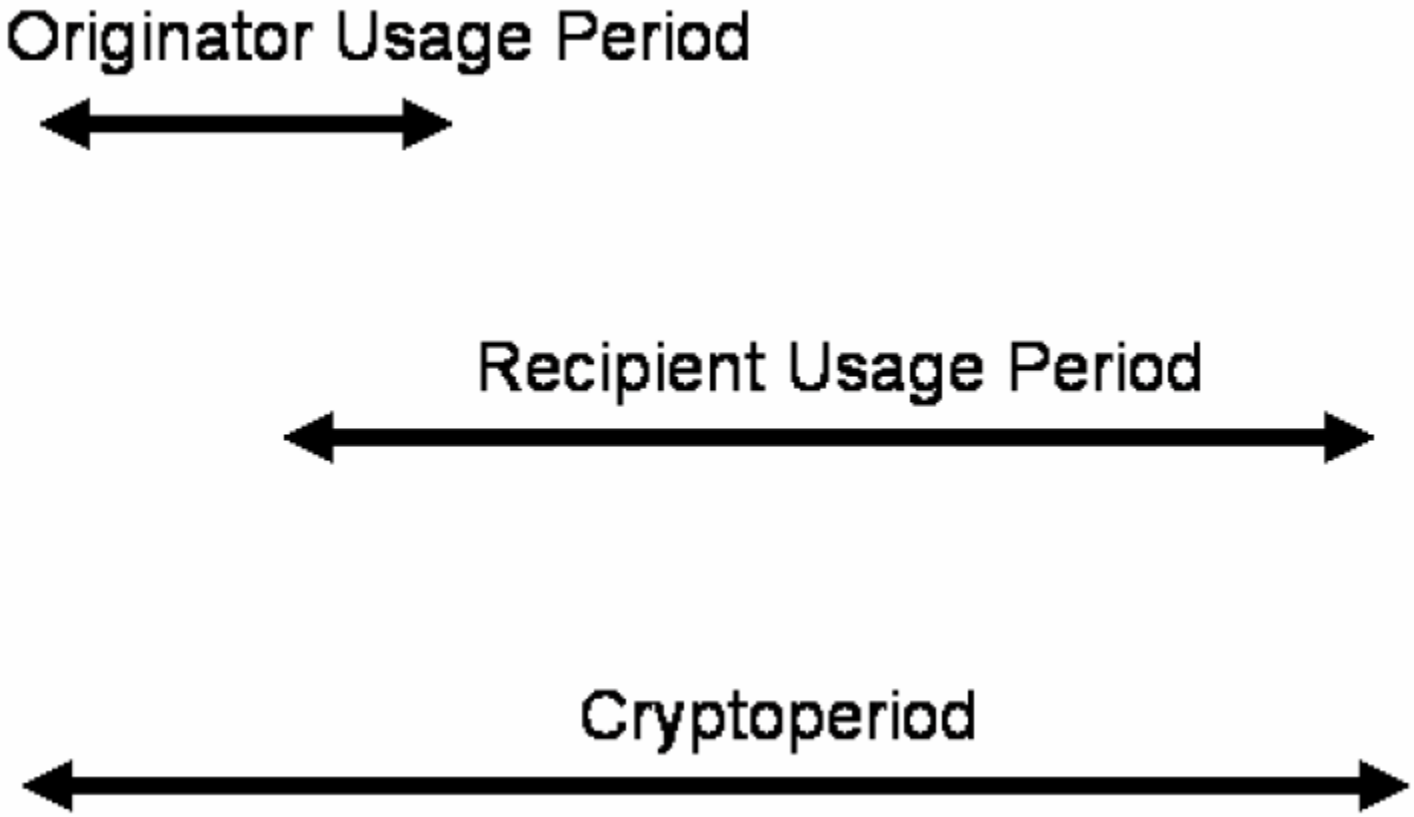
“Deprecated” means that the use of an algorithm or key length that provides the indicated security strength may be used if risk is accepted

“Legacy use” means that an algorithm or key length may be used because of its use in legacy applications

“Disallowed” means that an algorithm or key length **shall not be used for** applying cryptographic protection.

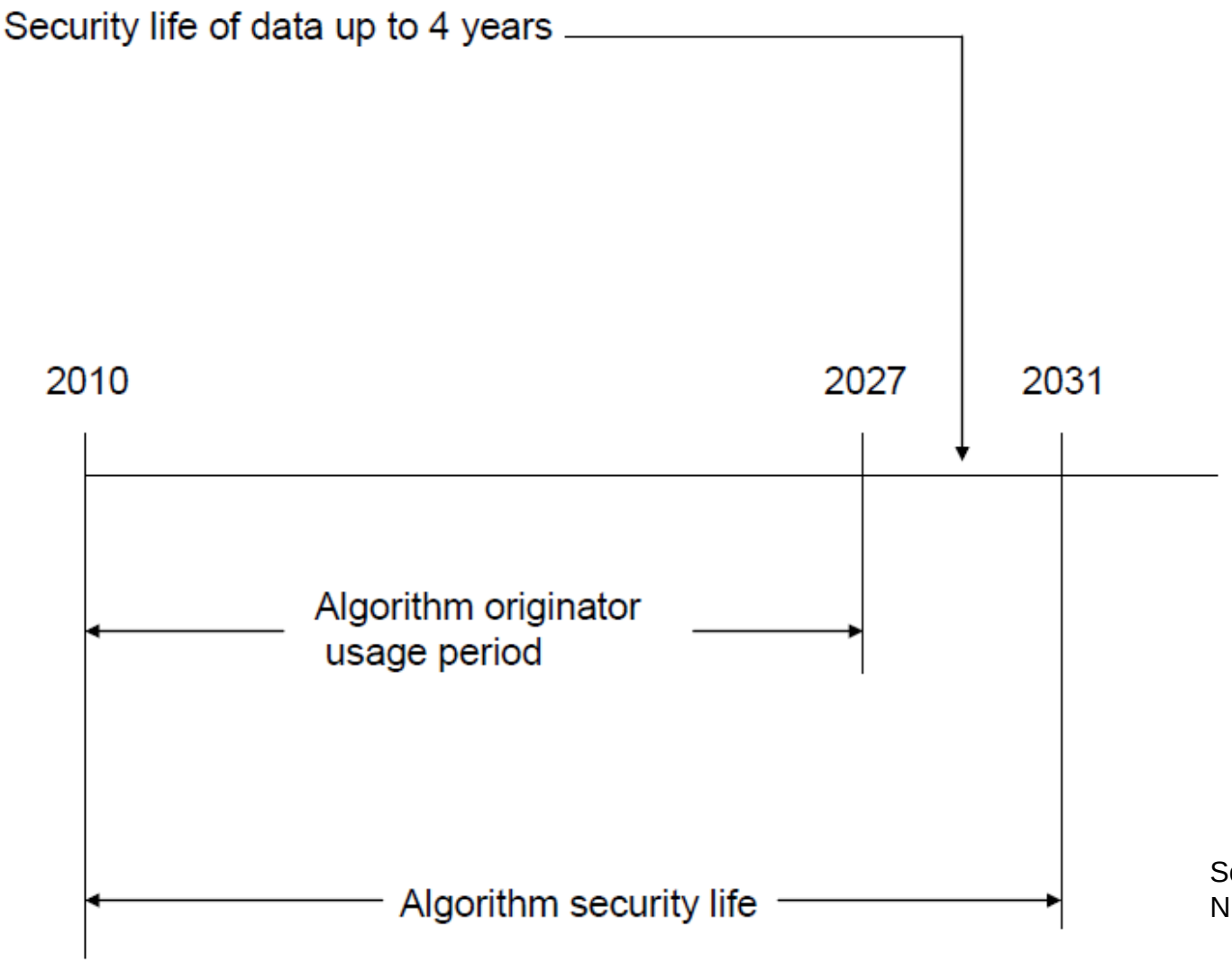
Source:
NIST SP800-57

Crypto period



Source:
NIST SP800-57

Crypto period example



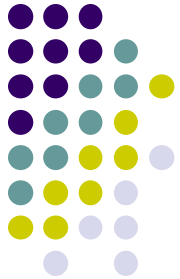
Source:
NIST SP800-57

Recommended crypto periods

Key Type	Cryptoperiod	
	Originator Usage Period (OUP)	Recipient Usage Period
1. Private Signature Key	1-3 years	
2. Public Signature Key	Several years (depends on key size)	
3. Symmetric Authentication Key	≤ 2 years	≤ OUP + 3 years
4. Private Authentication Key	1-2 years	
5. Public Authentication Key	1-2 years	
6. Symmetric Data Encryption Keys	≤ 2 years	≤ OUP + 3 years
7. Symmetric Key Wrapping Key	≤ 2 years	≤ OUP + 3 years
8. Symmetric and asymmetric RNG Keys	Upon reseeding	
9. Symmetric Master Key	About 1 year	
10. Private Key Transport Key	≤ 2 years[13]	
11. Public Key Transport Key	1-2 years	
12. Symmetric Key Agreement Key	1-2 years	
13. Private Static Key Agreement Key	1-2 years[14]	

Source:NIST SP800-57

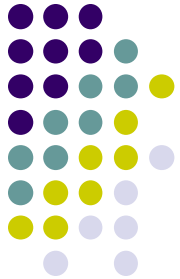
ETSI recommendation (RSA)



Parameter	1 year	3 years	6 years	10 years (speculative)
MinModLen	1 536	2 048	2 048	?
ErrProb	2^{-80}	2^{-100}	2^{-100}	2^{-100}
SeedEntropy/EntropyBits	80	100	100	?

- Source: ETSI TS 102 176-1 V2.1.1 (2011-07)
- Recommended key sizes for RSA and rsagen1 for a resistance during X years
- Starting date: 2011

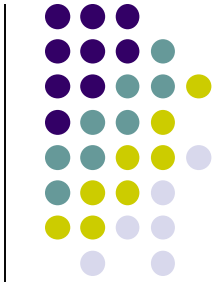
ETSI recommendation (RSA)



entry name of the padding scheme	1 year	3 years	6 years	10 years (speculative)
PKCS#1-v1.5	usable/n.a	usable/n.a.	usable/n.a.	unusable/n.a.
PKCS#1-v2.1	usable/n.a	usable/n.a	usable/n.a.	unusable/n.a.
PKCS#1-PSS	usable/64	usable/64	usable/64	usable/64
ISO-DS 2	usable/64	usable/64	usable/64	usable/64
ISO-DS 3	usable	usable	usable	usable
ISO-DIN-RN	usable/64	usable/64	usable/64	usable/64

- Source: ETSI TS 102 176-1 V2.1.1 (2011-07)
- Recommended padding schemes for RSA and rsagen1
- Starting date: 2011

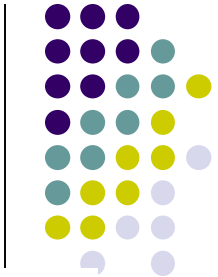
ETSI recommendation (DSA)



Parameter	1 year	3 years	6 years	10 years (speculative)
pMinLen	1 024	2 048	3 072	3 072
qMinLen	160	224	256	256
ErrProb	2^{-80}	2^{-80}	2^{-100}	2^{-100}
SeedEntropy/EntropyBits	80	80	100	100

- Source: ETSI TS 102 176-1 V2.1.1 (2011-07)
- Recommended key sizes for DSA
- Starting date: 2011

ETSI recommendation (ECDSA)



Recommended parameters for ecdsa-Fp and ecgen1 for a resistance during X years

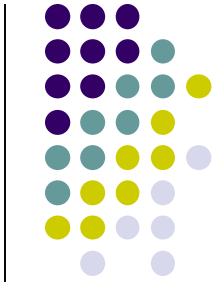
Parameter	1 year	3 years	6 years	10 years (speculative)
pMinLen	-	-	-	
qMinLen	160	224	256	256
r0Min	104	104	104	?
MinClass	200	200	200	?
ErrProb	2^{-80}	2^{-100}	2^{-100}	2^{-100}
SeedEntropy/EntropyBits	80	100	100	256

Recommended parameters for ecdsa -F2m and ecgen2 for a resistance during X years

Parameter	1 year	3 years	6 years	10 years (speculative)
pMinLen	-	-	-	
qMinLen	160	224	256	256
r0Min	104	104	104	?
MinClass	200	200	200	?
ErrProb	2^{-80}	2^{-100}	2^{-100}	2^{-100}
SeedEntropy/EntropyBits	80	100	100	256

- Source: ETSI TS 102 176-1 V2.1.1 (2011-07)
- Recommended key sizes for ECDSA
- Starting date: 2011

ETSI recommendation (hash functions)



entry name of the hash function	1 year	3 years	6 years	10 years (speculative)
sha1	unusable	unusable	unusable	unusable
ripemd160	unusable	unusable	unusable	unusable
sha224	usable	usable	usable	unknown
sha256	usable	usable	usable	unknown
sha384	usable	usable	usable	usable
sha512	usable	usable	usable	usable
Whirlpool	usable	usable	usable	usable
NOTE: The listed hash functions are expected to be 2nd pre-image resistant and pre-image resistant for a longer period of time.				

- Source: ETSI TS 102 176-1 V2.1.1 (2011-07)
- Recommended hash functions
- Starting date: 2011

ETSI recommendation

Entry name of the signature suite	1 years	3 years	6 years	10 years
sha1-with-rsa	not recommended			
sha256-with-rsa	1 536	2 048	2 048	not recommended
RSASSA-PSS with mgf1SHA-1Identifier	1 536	not recommended		
RSASSA-PSS with mgf1SHA-224Identifier	1 536	2 048	2 048	not recommended
RSASSA-PSS with mgf1SHA-256Identifier	1 536	2 048	2 048	3 072
sha1-with-dsa	not recommended			
sha1-with-ecdsa	not recommended			
sha224-with-ecdsa	224	224	224	not recommended
sha256-with-ecdsa	256	256	256	256

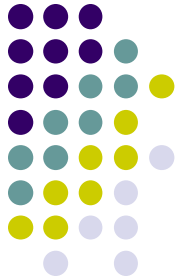
- Source: ETSI TS 102 176-1 V2.1.1 (2011-07)
- Recommended signature schemes
- Starting date: 2011

ICAO recommendations

- RSA (UK, CZ, France, …)
  - Padding: PKCS#1 v1.5, PSS (recommended)
  - For CA: min 3072 bits
  - For DS: min 2048 bits
- DSA
  - For CA: min 3072/256 bits
  - For DS: min 2048/224 bits
- ECDSA (Germany, Switzerland, …)
  - For CA: min 256 bits
  - For DS: min 224 bits
- Hash functions
  - SHA-1, SHA-2

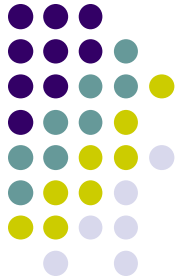
Source: ICAO Doc. 9303

Digital certificate



- proves the ownership of a public key – usually signed by certification authority(CA)
- certificate revocation lists – certificates no longer be trusted (compromised key, CA,…)
- certification path validation algorithm

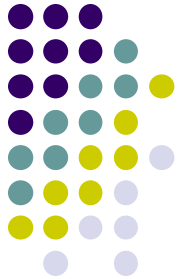
X.509 Public Key Infrastructure Certificate



RFC5280 - defines format and semantics of certificates and certificate revocation lists

- Format – PEM (ASCII) or DER(binary) defined by ASN.1
- X.509 version 3 certificate
- X.509 version 2 CRL
- certification path validation procedures

X.509 Public Key Infrastructure Certificate



Fields

Serial Number: Used to uniquely identify the certificate.

Subject: The person, or entity identified.

Signature Algorithm: The algorithm used to create the signature.

Signature: The actual signature to verify that it came from the issuer.

Issuer: The entity that verified the information and issued the certificate.

Valid-From: The date the certificate is first valid from.

Valid-To: The expiration date.

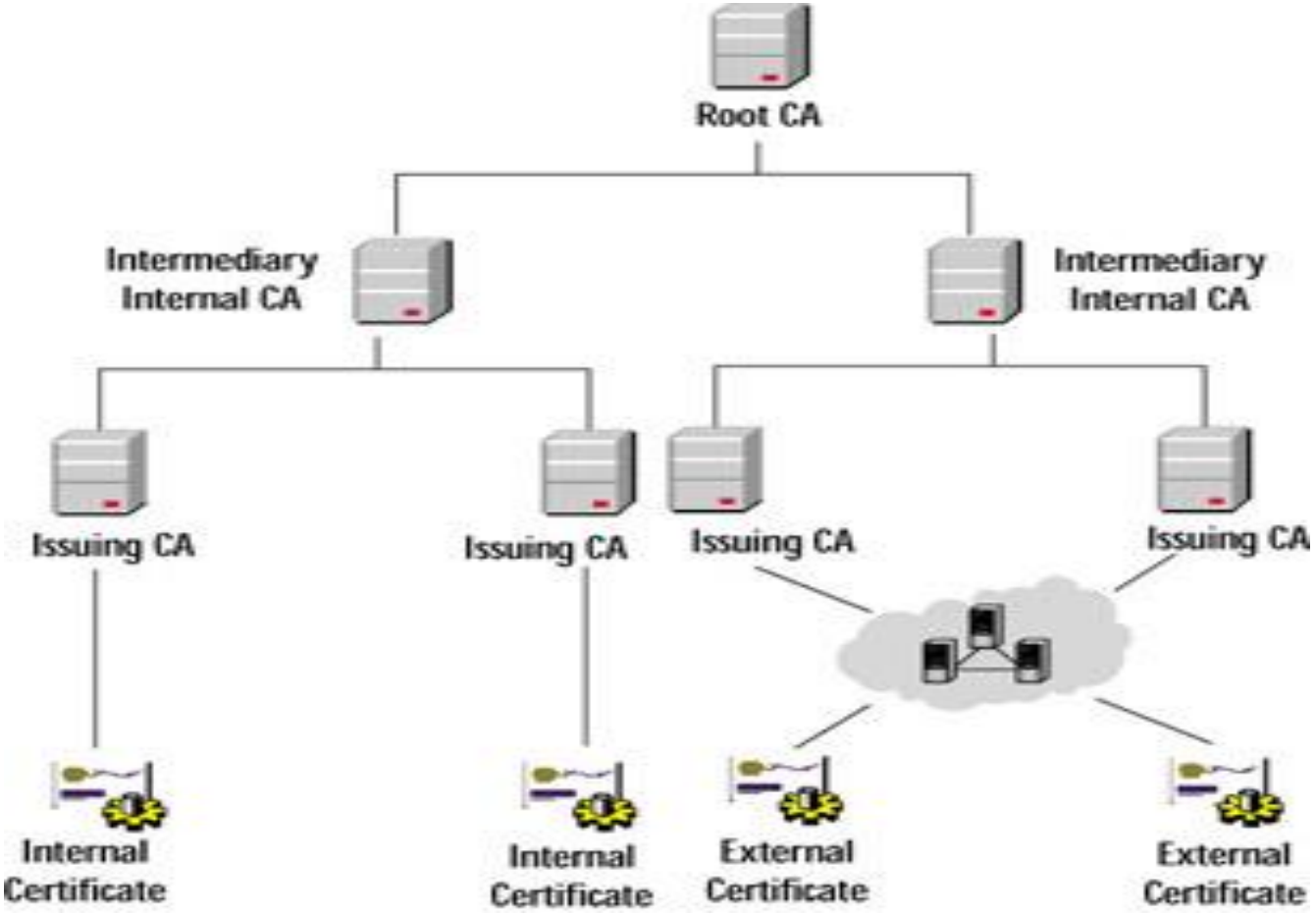
Key-Usage: Purpose of the public key (e.g. encipherment, signature, certificate signing...).

Public Key: The public key.

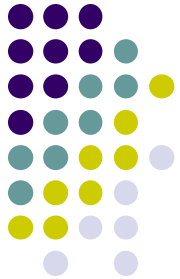
Thumbprint Algorithm: The algorithm used to hash the public key certificate.

Thumbprint (also known as fingerprint): The hash itself, used as an abbreviated form of the public key certificate.

Certification path



Assignment 3



1. Sign and verify signature of the text file **file.txt** (with "**PV181**" string) using ECDSA. (3 points)
2. Identify the signature alg. of www.fi.muni.cz certificate (2 points)
3. For RSA public key **pubRSA.key** find private **d**, **p**, **q**.

Hint : Use YAFU utility for integer factorization. (5 points)

Describe commands you used, signature algorithm of muni and parameters d,p,q in Desc_xxxx.txt (xxxx stands for your UCO). Put the Desc_xxxx.txt into IS (Assignement No. 3)