Cryptography 101: From Theory to Practice

Chapter 2 – Cryptographic Systems

Rolf Oppliger

February 4, 2022

Terms of Use

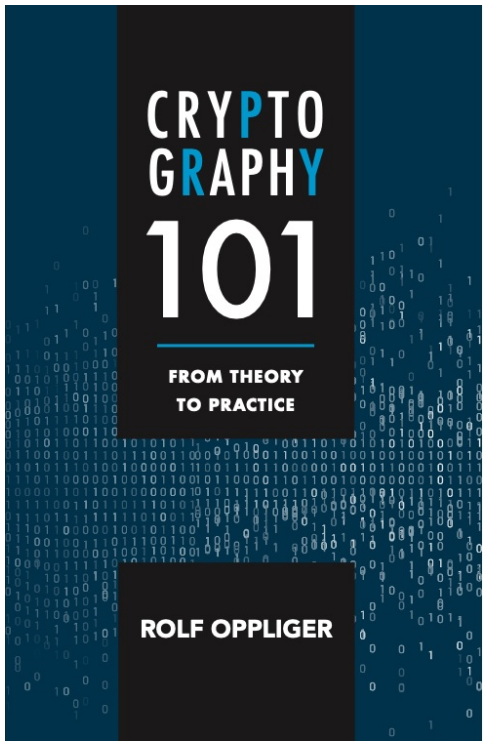
- This work is published with a CC BY-ND 4.0 license ()
  - CC = Creative Commons ()
  - BY = Attribution ()
  - ND = No Derivatives ()

whoami

rolf-oppliger.ch
rolf-oppliger.com

- Swiss National Cyber Security Centre NCSC (scientific employee)
- eSECURITY Technologies Rolf Oppliger (founder and owner)
- University of Zurich (adjunct professor)
- Artech House (author and series editor for information security and privacy)

Reference Book



ISBN 978-1-63081-846-3

`https://books.esecurity.ch/crypto101.html`

Challenge Me

Outline

2. Cryptographic Systems

2. Cryptographic Systems

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems

- Random generators
- Random functions
- One-way functions
- Cryptographic hash functions

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random generators

- Randomness is by far the most important ingredient for cryptography
- Almost all cryptographic systems in use today depend on some form of randomness

Definition 2.1 (Random generator)

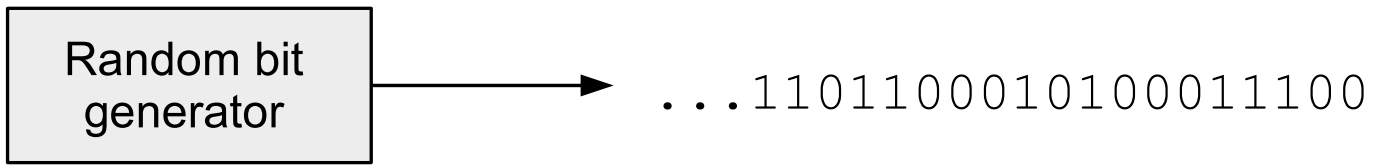
A device that outputs a sequence of statistically independent and unbiased values

If the output values are bits, then it is a random bit generator

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random generators

- A random generator has no input and only generates an output (e.g., a sequence of statistically independent and unbiased bits)
  - All bits occur with the same probability, i.e., $\Pr[0] = \Pr[1] = 1/2$
  - All 2^k k-tuples of bits occur with the same probability $1/2^k$
- Statistical tests can be used to verify these properties (or detect statistical defects, respectively)



2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random generators

- A random generator cannot be implemented in a purely deterministic way
- Instead, it is inherently nondeterministic, meaning that an implementation must use some physical events or phenomena
- Alternatively speaking, every (true) random generator requires a naturally occurring source of randomness
- The proper use of this source is a challenging engineering task
- Random generators and their security properties are further addressed in Chapter 3

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random functions

- A random generator is to output random-looking values
- In contrast, a **random function** (or **random oracle**) is not characterized by its output, but rather by the way it is chosen from a set of functions

Definition 2.2 (Random function)

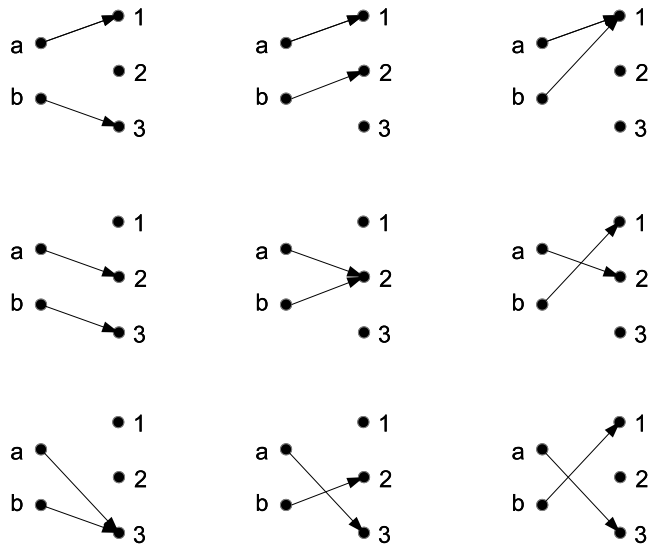
A function $f : X \to Y$ that is chosen randomly from $\mathrm{Funcs}[X, Y]$, i.e., the set of all functions that map elements of domain X to elements of codomain Y

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random functions

- For input value $x \in X$, a random function can output any value $y = f(x) \in f(X) \subseteq Y$
- The only requirement is that the same input value x must always map to the same output value y
- Note that there are $|Y|^{|X|}$ functions in $\mathrm{Funcs}[X, Y]$, and that this number is incredibly large (even for moderately sized X and Y)

Elements of $\mathrm{Funcs}[\{a, b\}, \{1, 2, 3\}]$



2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random functions

- If $X = \{a, b\}$ and $Y = \{1, 2, 3\}$, then $\mathrm{Funcs}[X, Y]$ comprises $3^2 = 9$ functions (see above)
- If X refers to all 2-bit strings and Y refers to all 3-bit strings, then $|X| = 2^2$ and $|Y| = 2^3$, i.e., $\mathrm{Funcs}[X, Y]$ comprises $(2^3)^{2^2} = (2^3)^4 = 2^{12} = 4,096$ elements
- If X and Y both refer to all 128-bit strings, then $\mathrm{Funcs}[X, Y]$ comprises $(2^{128})^{2^{128}} = 2^{128 \cdot 2^{128}} = 2^{2^7 \cdot 2^{128}} = 2^{2^{135}}$ elements
- If one wanted to number the functions and use an index to refer to a particular function, then this index would be 2^{135} bits long

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random functions

- Random functions are conceptual constructs that are mainly used in security proofs (they are not meant to be implemented)
- Referring to the ideal/real simulation paradigm, the random function yields the ideal system and it is shown that no adversary can tell a real system apart from it
- This means that the real system behaves like a random function, and hence an adversary must try out all possibilities
- This is computationally infeasible, and hence the real system is believed to be secure

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Random functions

- If $X = Y$ and $\mathrm{Funcs}[X, X]$ is restricted to the set of all permutations of X, i.e., $\mathrm{Perms}[X]$, then a **random permutation** is a randomly chosen permutation from $\mathrm{Perms}[X]$
- $|\mathrm{Perms}[X]| = |X|!$ also grows incredibly fast
- Most statements that apply to random functions also apply to random permutations
- Random functions and random permutations are further addressed in Chapter 4

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – One-way functions

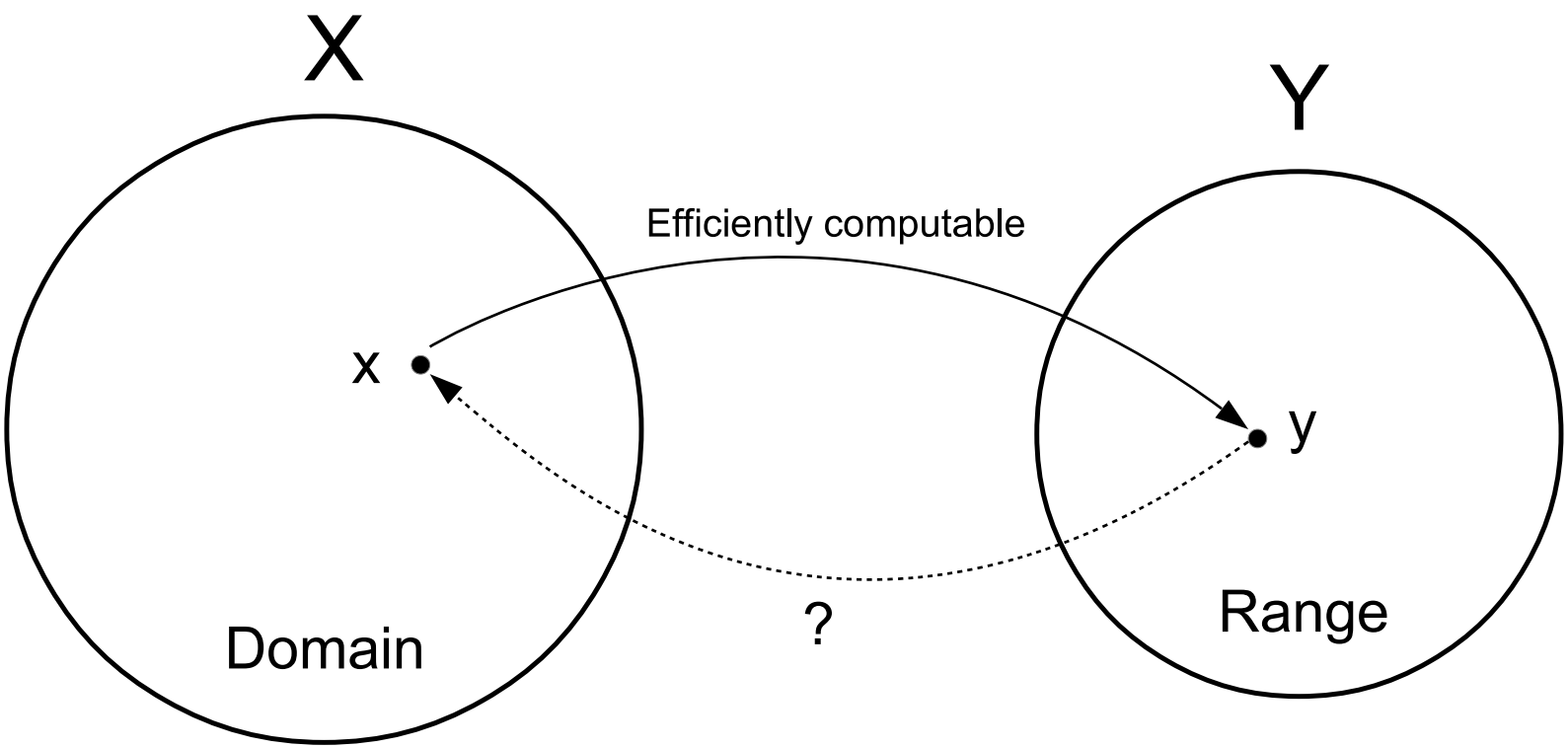
- Informally speaking, a **function** $f : X \to Y$ is **one way** if it is *easy* to compute but *hard* to invert
- Referring to complexity theory, *easy* means that the computation can be done efficiently (i.e., in polynomial time), whereas *hard* means that it is not known how to do the computation efficiently, i.e., no efficient algorithm is known

Definition 2.3 (One-way function)

A function $f : X \to Y$ of which $f(x)$ can be computed efficiently for all $x \in X$, but $f^{-1}(f(x))$ cannot be computed efficiently, i.e., $f^{-1}(y)$ cannot be computed efficiently for $y \in_R Y$

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – One-way functions



2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – One-way functions

- There are many real-world examples of one-way functions
- In a telephone book, the function that assigns a telephone number to each name is easy to compute (because the names are sorted alphabetically) but hard to invert (because the telephone numbers are not sorted numerically)
- Many physical processes are inherently one way
  - Smashing a bottle into pieces
  - Dropping a bottle from a bridge
  - Any time-related process (e.g., aging)
  - ...

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – One-way functions

- In contrast to the real world, there are only a few mathematical functions conjectured to be one way
- Many such functions refer to **modular exponentiation**, i.e.,
  - $f(x) = g^x \bmod m$
  - $f(x) = x^e \bmod m$
  - $f(x) = x^2 \bmod m$

  for some properly chosen modulus m
- These functions are heavily used in public key cryptography
- Note that no function has been shown to be one way in a mathematically strong sense, and that it is theoretically not even known whether one-way functions exist at all

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – One-way functions

- In general, a one-way function cannot be inverted efficiently
- But there may be one-way functions that can be inverted efficiently, if some extra information is known

Definition 2.4 (Trapdoor (one-way) function)

A one-way function $f : X \to Y$ that has some extra information (i.e., trapdoor) with which f can be inverted efficiently, i.e., $f^{-1}(f(x))$ can be computed efficiently for all $x \in X$ or $f^{-1}(y)$ can be computed efficiently for $y \in_R Y$

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – One-way functions

- The mechanical analog of a trapdoor function is a padlock
- The functions $f(x) = x^e \bmod m$ and $f(x) = x^2 \bmod m$ have a trapdoor (i.e., the prime factorization of m), whereas the function $f(x) = g^x \bmod m$ is not known to have a trapdoor — if m is prime
- If X and Y are the same, then a one-way function $f : X \to X$ that is a permutation, i.e., $f \in Perms[X]$, yields a **one-way permutation**
- One-way functions, trapdoor functions, one-way permutations, and trapdoor permutations are further addressed in Chapter 5

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Cryptographic hash functions

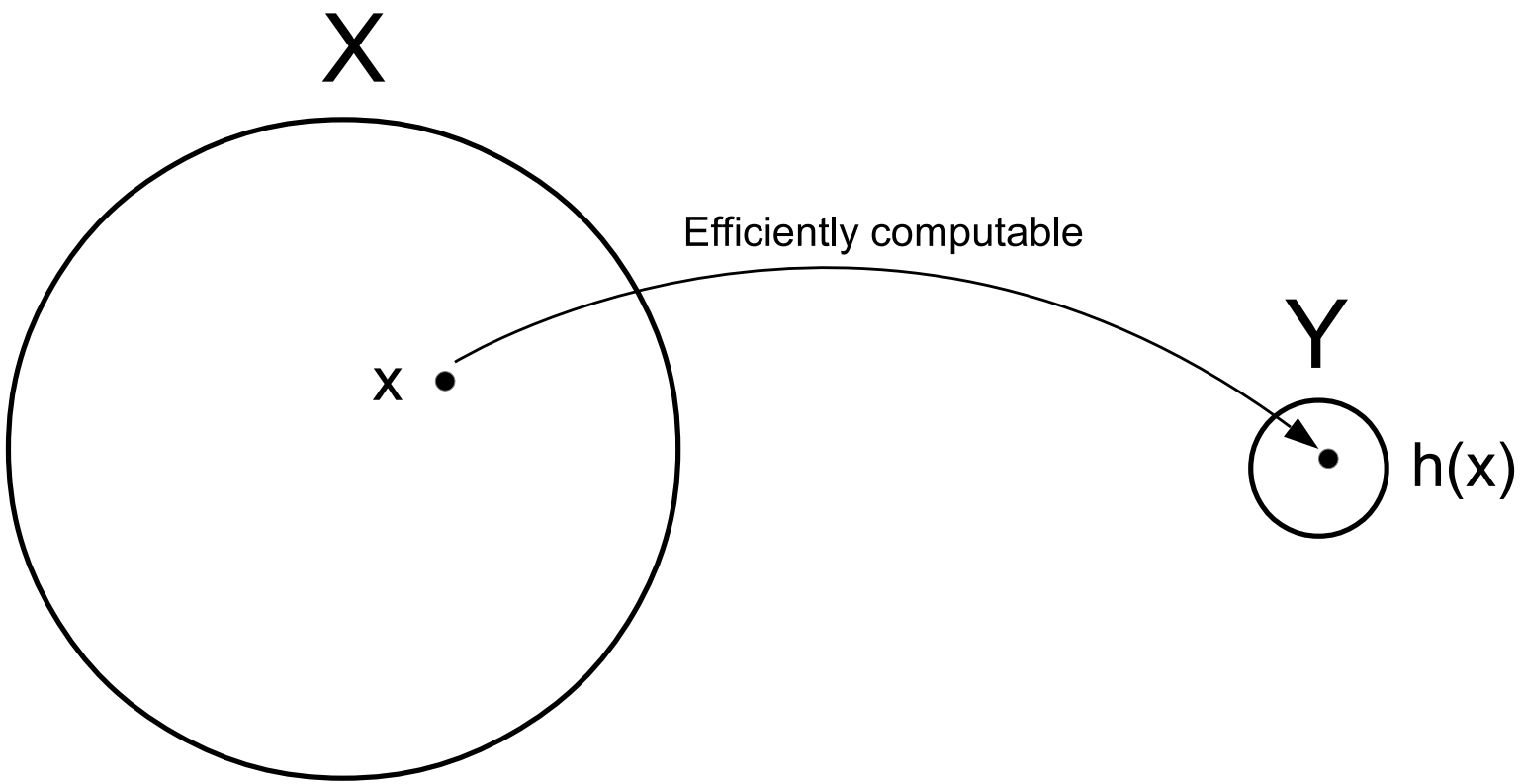
- Hash functions are widely used in computer science
- Informally speaking, a **hash function** is an efficiently computable function that takes an arbitrarily large input and generates an output of a usually much smaller size

Definition 2.5 (Hash function)

A function $h : X \to Y$ that can be computed efficiently for all $x \in X$ and $|X| \gg |Y|$

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Cryptographic hash functions



2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Cryptographic hash functions

- The elements of X and Y are typically strings of characters from a given alphabet
- If Σ_{in} is the input alphabet and Σ_{out} is the output alphabet, then a hash function h can be written as $h : \Sigma_{in}^* \rightarrow \Sigma_{out}^n$
- In many settings, Σ_{in} and Σ_{out} are the same, typically the binary alphabet $\Sigma = \{0, 1\}$
- In this setting, the hash function h takes as input an arbitrarily long bitstring and generates as output a bitstring of fixed size n

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Cryptographic hash functions

- In cryptography, one is talking about bitstrings that are relatively long (typically 256 bits)
- A respective hash function must fulfill two requirements
  - It must be hard to invert, i.e., the function is **one-way** or **preimage resistant**
  - It must be hard to find a collision, i.e., the function is **second-preimage resistant** or **collision resistant**

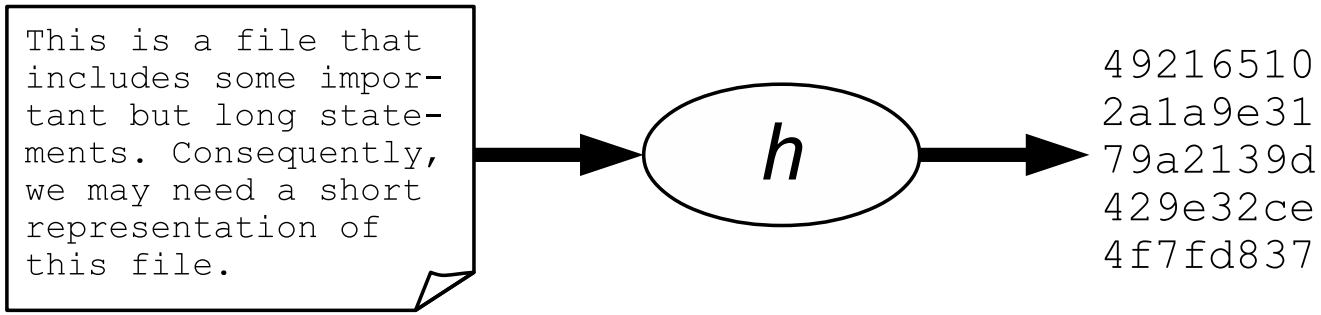
Definition 2.6 (Cryptographic hash function)

A hash function h that is either one-way and second-preimage resistant or one-way and collision resistant

2. Cryptographic Systems

2.1 Unkeyed Cryptosystems – Cryptographic hash functions

- Cryptographic hash functions have many applications
- Most importantly, such a function h can be used to hash arbitrarily sized messages to bitstrings of fixed size



- Cryptographic hash functions are addressed in Chapter 6

2. Cryptographic Systems

2.2 Secret Key Cryptosystems

- Pseudorandom generators
- Pseudorandom functions
- Symmetric encryption
- Message authentication
- Authenticated encryption

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom generators

- If a large number of random values is needed, then it may be appropriate to use a **pseudorandom generator (PRG)** instead of — or in combination with — a true random generator

Definition 2.7 (PRG)

An efficiently computable function that takes as input a relatively short value (seed) of length n and generates as output a value of length $l(n)$ with $l(n) \gg n$ that appears to be random

If the input and output values are bit sequences, then the PRG is a *pseudorandom bit generator (PRBG)*

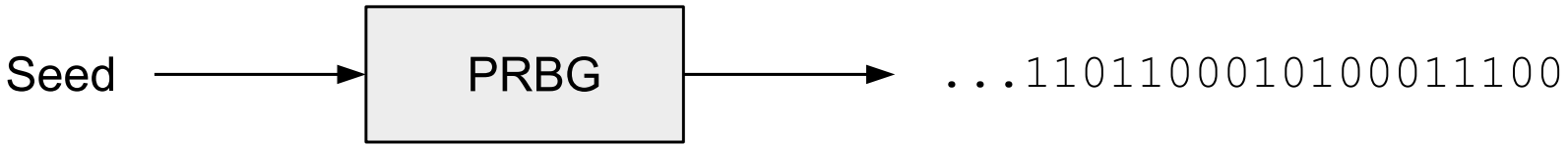
2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom generators

- Note that Definition 2.7 is not precise in a mathematically strong sense (because the statement "appears to be random" is not properly defined)
- Unlike a true random generator, a PRG operates deterministically, and this means that a PRG always outputs the same values if seeded with the same input value
- A PRG thus represents a **finite state machine (FSM)**, and the sequence of the generated values needs to be cyclic (with a potentially very large cycle)
- This is why one cannot require that the output of a PRG is truly random, but only that it appears to be so

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom generators



Formally speaking, a PRBG G is a mapping from $\mathcal{K} = \{0,1\}^n$ to $\{0,1\}^{l(n)}$, where $l(n)$ represents a stretch function, i.e., a function that stretches an n-bit input value into a longer $l(n)$-bit output value with $n < l(n) \leq \infty$:

$$G : \mathcal{K} \longrightarrow \{0,1\}^{l(n)}$$

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom generators

- A PRG is secure, if its output is indistinguishable from the output of a true random generator (according to the security game of the ideal/real simulation paradigm)
- Pseudorandomness and PRGs are key ingredients and have many applications in cryptography
  - Key generation
  - Additive stream ciphers
  - ...
- PR(B)Gs are further addressed in Chapter 7

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom functions

- A PRG "simulates" a random generator
- Similarly, a **pseudorandom function (PRF)** "simulates" a random function
- Remember that a random function $f : X \to Y$ is randomly chosen from $\mathrm{Funcs}[X, Y]$, and that the cardinality of this set is incredibly large, i.e., $|\mathrm{Funcs}[X, Y]| = |Y|^{|X|}$
- The idea of a PRF is to use a subset of $\mathrm{Funcs}[X, Y]$ that is sufficiently small so that one can number its elements and use a moderately sized index (e.g., 135 instead of 2^{135} bits)
- If one uses a secret key as index, then one has something like a random function without its disadvantages

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom functions

Definition 2.8 (PRF)

A family $F : \mathcal{K} \times X \to Y$ of (efficiently computable) functions, where each $k \in \mathcal{K}$ determines a function $f_k : X \to Y$ that is indistinguishable from a random function (i.e., a function randomly chosen from $\mathrm{Funcs}[X, Y]$)

Similar to a PRF, a **pseudorandom permutation (PRP)** is a family $P : \mathcal{K} \times X \to X$ of permutations, where each $k \in \mathcal{K}$ determines a permutation $p_k : X \to X$ that is indistinguishable from a random permutation (i.e., a permutation randomly chosen from $\mathrm{Perms}[X]$)

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Pseudorandom functions

- PRFs and PRPs are important in modern cryptography
- Many cryptographic constructions can be seen in this light
  - A cryptographic hash function is a PRF (with no key)
  - A key derivation function (KDF) is a PRF with a seed acting as a key
  - A block cipher is a PRP
  - A PRG can be built from a PRF and vice versa
  - ...
- PRFs and PRPs are further addressed in Chapter 8

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption

Let $\mathcal{M}$ be a plaintext message space, $\mathcal{C}$ a ciphertext space, and $\mathcal{K}$ a key space

Definition 2.9 (Symmetric encryption system or cipher)

A pair (E, D) of families of efficiently computable functions:

- $E : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ denotes a family $\{E_k : k \in \mathcal{K}\}$ of encryption functions $E_k : \mathcal{M} \to \mathcal{C}$
- $D : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$ denotes a family $\{D_k : k \in \mathcal{K}\}$ of respective decryption functions $D_k : \mathcal{C} \to \mathcal{M}$

For every message $m \in \mathcal{M}$ and key $k \in \mathcal{K}$, D_k and E_k must be inverse to each other, i.e., $D_k(E_k(m)) = m$

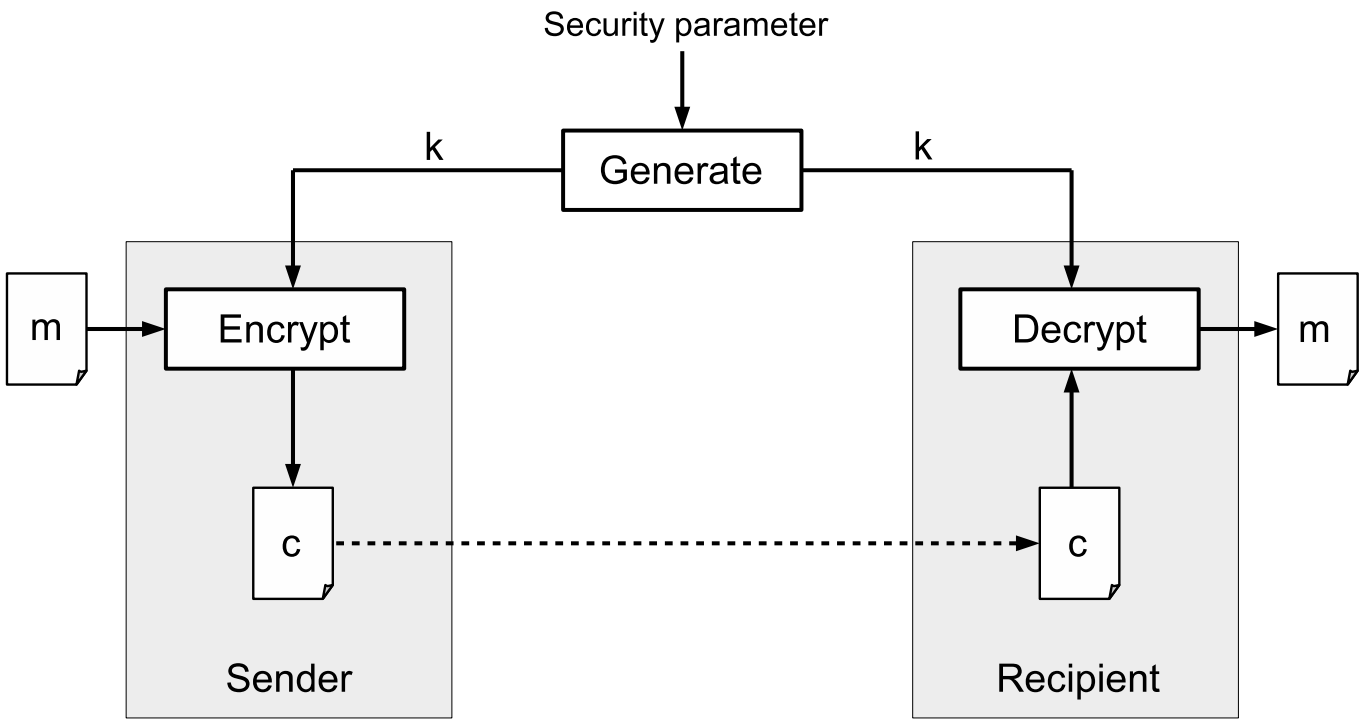
2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption

- While the decryption functions need to be deterministic, the encryption functions can be deterministic or probabilistic
- Probabilistic encryption usually has a security advantage
- In a typical setting, $\mathcal{M} = \mathcal{C} = \{0,1\}^*$ refers to the set of all arbitrarily long binary strings, whereas $\mathcal{K} = \{0,1\}^l$ refers to the set of all l bits long keys
- In this notation, l stands for the key length of the symmetric encryption system (typically $l \geq 128$)

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption



2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption

- The characteristic feature of a symmetric encryption system is that k is the same on either side of the communication channel
- Another characteristic feature is that the system can operate on individual bits and bytes ($\rightarrow$ stream ciphers) or on larger blocks ($\rightarrow$ block ciphers)
- While there are modes of operation that turn a block cipher into a stream cipher, the opposite is not known to be true, i.e., there is no mode of operation that turns a stream cipher into a block cipher

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption

- To make meaningful security statements, one must define the adversary (1) and the task he or she needs to solve (2)
  - With regard to (1), one must specify his or her computing power and the types of attacks he or she is able to mount
    - Ciphertext-only attack
    - Chosen-plaintext attack (CPA)
    - Chosen-ciphertext attack (CCA)
    - ...
  - With regard to (2), one must specify whether he or she must decrypt a ciphertext, determine a key, determine a few bits from either the plaintext or the key, or do something else
- Consequently, there are several notions of security

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption

- If the adversary has infinite computing power but is still not able to solve the task within a finite amount of time, then the cipher is **unconditionally** or **information-theoretically secure** (e.g., one-time pad)
- If the adversary is theoretically able to solve the task within a finite amount of time, but the computing power required to do so is beyond his or her capabilities, then the cipher is "only" **conditionally** or **computationally secure**
- This means that the system can be broken in theory (e.g., by an exhaustive key search), but the respective attack is believed to be computationally infeasible

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Symmetric encryption

- If a cipher is semantically secure, then it is computationally infeasible to retrieve any meaningful information about a plaintext message from a given ciphertext, even if the adversary can mount a CPA
- All symmetric encryption systems in use today are (at least) semantically secure
- Symmetric encryption and the various notions of security (e.g., semantic security) are further addressed in Chapter 9

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Message authentication

- While encryption systems are to protect the confidentiality of data (e.g., messages), there are applications that require the authenticity and integrity of data to be protected
- The typical way to achieve this is to have the sender add an authentication tag to the message and have the recipient verify the tag
- This is conceptually similar to an error correction code
- But in addition to protecting a message against transmission errors, such an authentication tag must also be protected against tampering and deliberate fraud

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Message authentication

- From a bird's eye perspective, there are two possibilities to construct an authentication tag
  - Public key cryptography and **digital signatures**
  - Secret key cryptography and **message authentication codes (MACs)**

Definition 2.10 (MAC)

An authentication tag that can be computed and verified with a secret parameter (key)

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Message authentication

Definition 2.11 (Message authentication system)

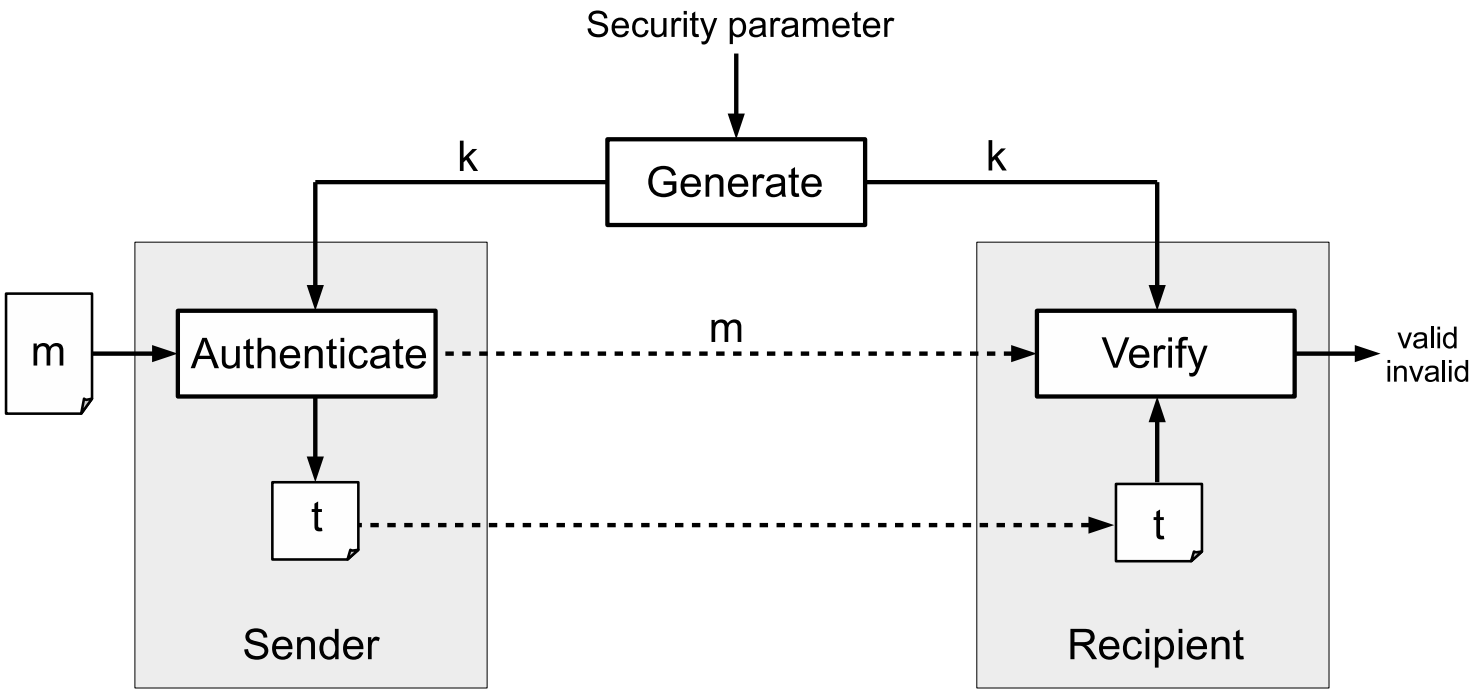
A pair (A, V) of families of efficiently computable functions:

- $A : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ denotes a family $\{A_k : k \in \mathcal{K}\}$ of authentication functions $A_k : \mathcal{M} \to \mathcal{T}$
- $V : \mathcal{K} \times \mathcal{M} \times \mathcal{T} \to \{\mathit{valid}, \mathit{invalid}\}$ denotes a family $\{V_k : k \in \mathcal{K}\}$ of verification functions $V_k : \mathcal{M} \times \mathcal{T} \to \{\mathit{valid}, \mathit{invalid}\}$

For every message $m \in \mathcal{M}$ and key $k \in \mathcal{K}$, $V_k(m, t)$ must yield *valid* iff t is a valid authentication tag for m and k, i.e., $t = A_k(m)$ and hence $V_k(m, A_k(m))$ must yield *valid*

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Message authentication



2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Message authentication

- To argue about the security of a message authentication system, one must define the adversary and the task he or she must solve
- Similar to symmetric encryption systems, one may consider adversaries with infinite computing power to come up with systems that are **unconditionally** or **information-theoretically secure**, or — more realistically — adversaries with finite computing power to come up with systems that are "only" **conditionally** or **computationally secure**
- Message authentication, MACs, and respective message authentication systems are further addressed in Chapter 10

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Authenticated encryption

- In the past, people used symmetric encryption systems to encrypt messages and message authentication systems to generate MACs that were then appended to the messages
- But it was not clear how (i.e., in what order), the two cryptographic primitives had to be applied and combined to achieve the best level of security
- Generic composition methods
  - Encrypt-then-MAC (EtM): $E_{k_e}(m) \parallel A_{k_a}(E_{k_e}(m))$ → IPsec
  - Encrypt-and-MAC (E&M): $E_{k_e}(m) \parallel A_{k_a}(m)$ → SSH
  - MAC-then-Encrypt (MtE): $E_{k_e}(m \parallel A_{k_a}(m))$ → SSL/TLS

2. Cryptographic Systems

2.2 Secret Key Cryptosystems – Authenticated encryption

- Since the early 2000s, it is known that the composition EtM method provides the best level of security
- Most security protocols follow this approach
- More specifically, they combine message encryption and authentication in **authenticated encryption (AE)** or even **authenticated encryption with associated data (AEAD)**
- AE(AD) and respective modes of operation for block ciphers are further addressed in Chapter 11

2. Cryptographic Systems

2.3 Public Key Cryptosystems

- Key establishment
- Asymmetric encryption
- Digital signatures

In a **hybrid cryptosystem**, public key cryptography is used for authentication and key establishment, whereas secret key cryptography is used for everything else (e.g., bulk data encryption)

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Key establishment

- If two or more entities want to employ secret key cryptography, then they must share a secret parameter that represents a key
- Consequently, in a large system many keys must be generated, stored, managed, used, and destroyed in a secure way
- If n entities want to securely communicate with each other, then there are

$$\binom{n}{2} = \frac{n(n-1)}{1 \cdot 2} = \frac{n^2 - n}{2}$$

such keys ($\approx n^2$-problem)

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Key establishment

- In a dynamic system, entities may join and leave at will
- The predistribution of all keys is impossible, because it is not even known in advance who may want to join
- This means that one has to be able to establish keys on the fly (whenever needed)
  - Key distribution center (KDC), e.g., Kerberos
  - Key establishment ($\rightarrow$ Chapter 12)
    - Key distribution
    - Key agreement or exchange, e.g., Diffie-Hellman key exchange

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Asymmetric encryption

- Similar to a symmetric encryption system, an asymmetric encryption system can be used to encrypt and decrypt plaintext messages
- An asymmetric encryption system can be built from a trapdoor function (or a family of trapdoor functions)
- Each public key pair comprises a public key *pk* that yields a one-way function and a private key *sk* that yields a respective trapdoor

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Asymmetric encryption

Definition 2.12 (Asymmetric encryption system)

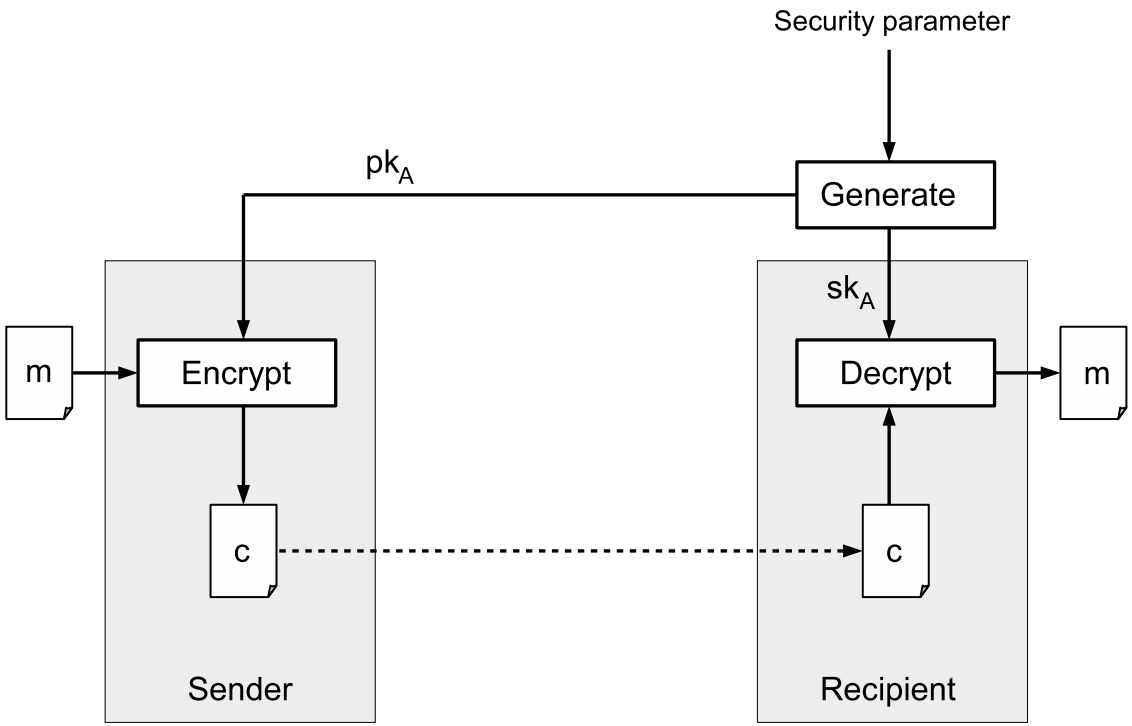
A triple of three efficient algorithms:

- Generate(1^k) generates a public key pair (pk, sk)
- Encrypt(pk, m) generates a ciphertext $c = \text{Encrypt}(pk, m)$
- Decrypt(sk, c) generates a plaintext message $m = \text{Decrypt}(sk, c)$

For every plaintext message m and public key pair (pk, sk), the Encrypt and Decrypt algorithms must be inverse to each other, i.e., $\text{Decrypt}(sk, \text{Encrypt}(pk, m)) = m$

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Asymmetric encryption



2. Cryptographic Systems

2.3 Public Key Cryptosystems – Asymmetric encryption

- There are many asymmetric encryption systems, such as Elgamal, RSA, and Rabin
- These systems are based on the three exemplary one-way functions mentioned above
- Because it is computationally infeasible to invert these functions, the systems provide a reasonable level of security — even in their basic forms (aka textbook versions)
- Asymmetric encryption and respective notions of security are further addressed in Chapter 13

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures

- According to RFC 4949, a **digital signature** refers to "a value computed with a cryptographic algorithm and appended to a data object in such a way that any recipient of the data can use the signature to verify the data's origin and integrity"
- Similarly, the term is defined as "data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery, e.g. by the recipient" in ISO/IEC 7498-2

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures

- According to ISO/IEC 7498-2, there are two classes of digital signatures and respective digital signature systems (DSS)
  - Digital signatures **with appendix**
  - Digital signatures **giving message recovery**
- The entity that digitally signs a data unit or message is called the **signer** or **signatory**, whereas the entity that verifies the digital signature is called the **verifier**
- In a typical setting, both the signer and the verifier are computing devices that are operated on a user's behalf

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures

Definition 2.13 (DSS with appendix)

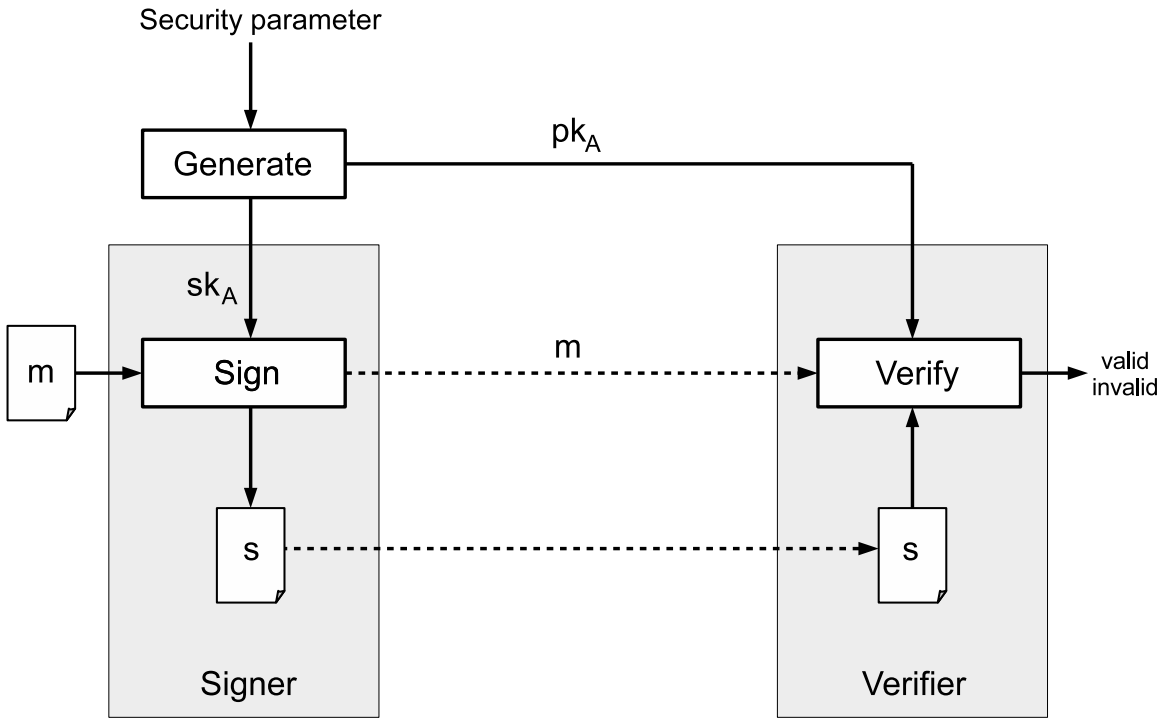
A triple of three efficiently computable algorithms:

- Generate(1^k) generates a public key pair (pk, sk)
- Sign(sk, m) generates a digital signature s for m
- Verify(pk, m, s) generates a binary decision whether the signature is valid

Verify(pk, m, s) must yield *valid* iff s is a valid digital signature for m and pk, i.e., for every message m and every public key pair (pk, sk), Verify(pk, m, Sign(sk, m)) must yield *valid*

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures



2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures

Definition 2.14 (DSS giving message recovery)

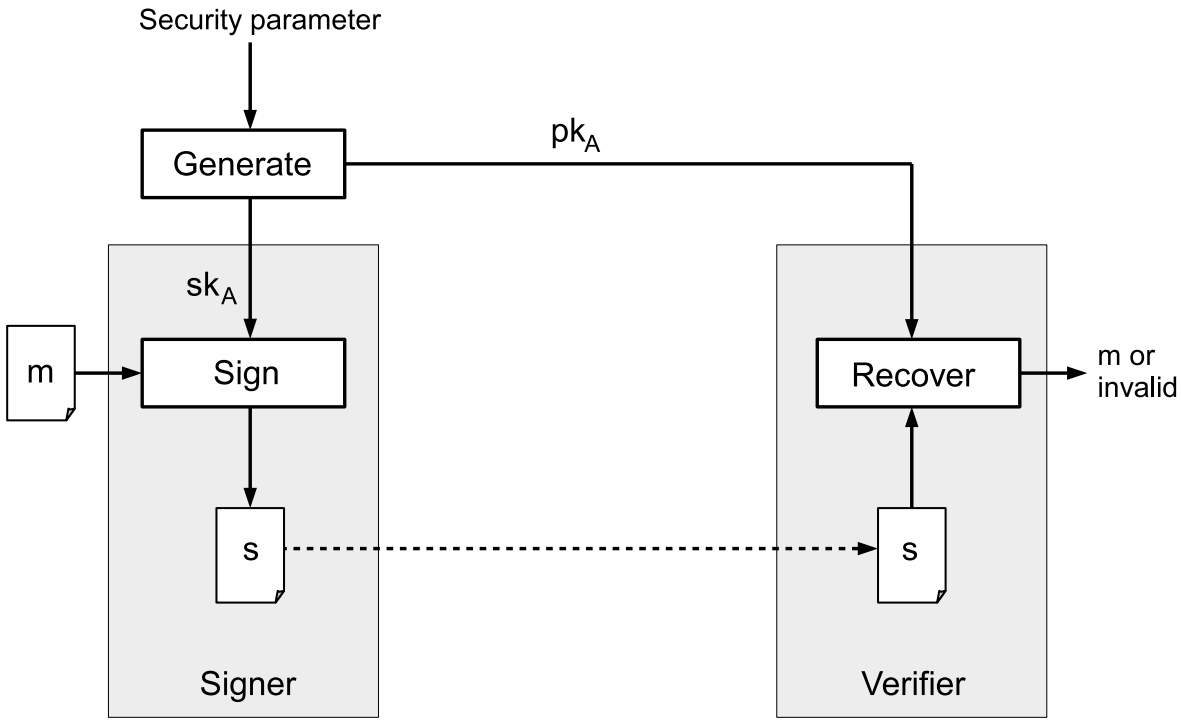
A triples of three efficiently computable algorithms:

- Generate(1^k) generates a public key pair (pk, sk)
- Sign(sk, m) generates as output a digital signature s
- Recover(pk, s) generates either the message m or a notification indicating that the signature is invalid

Recover(pk, s) must yield m if and only if s is a valid digital signature for m and pk, i.e., for every message m and every public key pair (pk, sk), Recover(pk, Sign(sk, m)) must yield m

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures



2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures

- With the proliferation of the Internet in general and Internet-based e-commerce in particular, digital signatures and their legislation have become important and very timely topics
- Many DSS with specific properties have been developed, proposed, and published in the past
- Again, the most important examples are RSA, Rabin, Elgamal, and some variants of Elgamal, such as the Digital Signature Algorithm (DSA) and the elliptic curve DSA (ECDSA)

2. Cryptographic Systems

2.3 Public Key Cryptosystems – Digital signatures

- Similar to asymmetric encryption systems, the security discussion for digital signatures is nontrivial and subtle
- There are several notions of security
- In the strongest case, a DSS can withstand existential forgery, even if the adversary can mount adaptive chosen-message attacks
- Digital signatures and respective notions of security are further addressed in Chapter 14

2. Cryptographic Systems

2.4 Final Remarks

- There are unkeyed, secret key, and public key cryptosystems
- This classification scheme is somewhat arbitrary, and other classification schemes may be used instead
- A major theme in cryptography is to better understand and formally define the notions of security, and to prove that particular cryptosystems are in line with these definitions

2. Cryptographic Systems

2.4 Final Remarks

- Another major theme is how to compose or combine secure cryptographic building blocks in a modular fashion
  - Universal composability
  - Constructive cryptography
  - ...
- Last but not least, there are several models and recommendations regarding keylengths that are appropriate ($\rightarrow$ https://www.keylength.com)

Questions and Answers

Thank you for your attention