Summary of “Einführung in die Informationssicherheit”

Martin Winter

Abstract— **This document should serve as a summary of the necessary information given in the lecture “Einführung in die Informationssicherheit”**

I. ABBREVIATIONS AND STANDARD NOTATION

Abbreviation	Notation
AES	Advanced Encryption Standard
CBC	Cipher Block Chaining mode
CFB	Cipher FeedBack mode
CRL	Certificate Revocation List
DES	Data Encryption Standard
DHP	Diffie-Hellman problem
DLP	Discrete logarithm problem
DN	Distinguished Name
DPA	Differential Power Analysis
DSA	Digital Signature Algorithm
ECB	Electronic Codebook mode
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
GDLP	Generlized discrete Logarithm Problem
HW	Hamming Weight
IDS	Intrusion Detection System
IFP	Integer Factorization Problem
LSB	Least Significant Bit
MAC	Message Authentication Code
MDC	Modification Detection Code
MSB	Most Significant Bit
OFB	Output Feedback mode
OID	Object Identifier
PGP	Pretty Good Privacy
PKI	Public Key Infrastructure
RSA	Rivest-Shamir-Adleman encryption scheme
SCA	Side-Channel Attack
SPA	Simple Power Analysis
SSCA	Simple Side-Channel Attack
DSCA	Differential Side-Channel Attack

II. BASIC CRYPTOLOGIC PRINCIPLES

Because of the rise of digital communication, the value of information keeps growing, while it is subject to a large number of threats. This brings up the need to protect this information, hence Cryptography is the science of protecting information. A **cryptographic algorithm** is a **mathematical function**, that uses a **key** to encipher information, without this key, deciphering wouldn’t be possible.

A. *CIA and Non-repudiation*

Typically two entities want to **exchange messages securely** over an **insecure channel**, the **adversary** can now try to do several things, including eavesdropping the communication or altering it. **Eavesdropping** (Abhören) is a threat to the **confidentiality** of the message, a third member could possible read the content of the message. On the other hand, **altering** (Verändern) is a threat to the **integrity** of the message, therefore it is required to **authenticate** the communicating entities. The last of the security service deals is called **non-repudiation** (Nicht-Anerkennung), which means, that a sender cannot deny a message was sent, this property cannot be achieved by secret key cryptography alone unfortunately.

B. *Secret Key Cryptography*

The most important requirement for such algorithms, besides their resistance against attacks, is that the reduction of performance is minimal. There are three types, which are discussed here

- **Block Ciphers** : Used to Encrypt Data
- **Stream Ciphers** : Used as alternative
- **MDCs**: Ensure integrity of data
- **MACs**: For authentication

*1) **Block Ciphers**:* are defined as a set of boolean permutations on a *n-bit* vector, this set contains a boolean permutation for each value of a key *k*. It takes an element (from plaintext) and transforms it into an element from ciphertext. Such a block cipher usually consists of several transformations, which form the encryption algorithm, those transformations happen several times, the transformation itself is called *round function*, for each round a *round key* K_i is generated from the cipher key. There are mainly two different types of design, the *Feistel ciphers* and the *substitution permutation network (SPN).*
There are also several modes of operation, whenever a message is longer than the block size, four modes are standardized. The *electronic code book* (ECB) corresponds

to the usual use case, the message is split into blocks and each block is encrypted separately with the same key.
In *cipher block chaining* (CBC) each ciphertext block is x-ored with the next plaintext block before encryption.
In the *output feedback* (OFB) mode and the *cipher feedback* (CFB) mode, a keystream is generated and x-ored with the plaintext.

*2) **Stream Ciphers**:* encrypt individual characters, usually bits, of a plaintext one at a time and use an encryption transformation, which varies with time. In contrast to block ciphers, this encryption not only depends on the key and plaintext, but also on the current state.
Synchronous stream ciphers generate a keystream independently of the plaintext, sender and receiver must therefore be synchronized (same key and same state within that key).
Asynchronous stream cipher is a stream cipher, in which the keystream is generated as a function of the key and a fixed number of previous ciphtertext bits. Because the keystream is dependent on only a few previous ciphertext bits, self-synchronization is possible even if some of the transmitted ciphertext bits are corrupted.

*3) **MDCs**:* take an input of arbitrary length and compress it to an output of fixed length, which is called the hash value. It satisfies the following properties

- *preimage resistance*: computationally infeasible to find preimage to given hash value
- *2nd preimage resistance*: computationally infeasible to find 2nd preimage
- *collision resistance*: computationally infeasible to find two different inputs with same hash value

Hash functions are used to ensure the integrity of data, data is used as input to the hash function and the output is stored. It is possible to check the input for alteration by simply redoing the computation and comparing it with the original hash value.

*4) **MACs**:* Hash functions, which involve a secret key, are called MACs (message authentication codes). The output of such a *keyed hash function* is also called MAC, in contrast to MDCs they can also guarantee **data origin authentication** and **data integrity**. In order to ensure authenticity of data, an entity computes a MAC on the data using a private key, to verify the authenticity of the data later on, the MAC can be recomputed using the private key.

C. Public Key Cryptography

In public key cryptography, secret keys are **replaced by keypairs**, consisting of a **private key**, which must be kept confidential, and a **public key**, which is openly accessible.
A message, encrypted by the public key, can be decrypted by the private key, this works, because the keys are linked in a mathematical way, such that knowing the public key does not allow to recover the private key, but enable independent encryption/decryption.

*1) **RSA**:* The Rivest-Shamir-Adleman(RSA) algorithm was the first public key encryption algorithm invented. It is based on IFP (Integer factorization problem) and keys are generated as follows:
One selects two large, **secret prime numbers** p and q and computes the public RSA modules

$$n = p \cdot q$$

Then one chooses a public encryption exponent e wich satisfies

$$gcd\,(e, (p-1)(q-1)) = 1$$

gcd stands for *greatest common divisor*. The private key d can be calculated by solving

$$e \cdot d = 1 \; mod \; (p-1)(q-1)$$

Hence the **public key** is the **pair (e,n)** and the **private key** is the **triple (d,p,q)**.
For encryption, the message needs to represented as a number $m < n$, the ciphtertext itself is computed by raising m to the power of e

$$c = m^e \; mod \; n$$

The message can be decrypted by exponentiation like

$$m = c^d \; mod \; n \tag{1}$$

Example Key Generation:
In real applications, p and q should at least be 512 bits!

$$\begin{aligned} p = 5 \quad q = 11 \\ n = p \cdot q = 55 \\ (p-1)(q-1) = 40 \\ \text{Random Number } e = 7 \\ gcd(e, (p-1)(q-1)) = 1 \\ d = e^{-1} \equiv 23 \; mod \; 40 \end{aligned}$$

The public key is now $(e, n) = (7, 55)$ and the private key is $(d, n) = (23, 55)$.
Example Encryption:
Encrypt the message M, assuming $M = 25$ with public key $(e, n) = (7, 55)$.

$$c = 25^7 \equiv 20 \; mod \; n$$

Example Decryption:
Received message is $c = 20 \; mod \; 55$ and private key is $(d, n) = (23, 55)$. This leads to

$$m = 20^{23} \equiv 25 \; mod \; 55$$

*2) **El Gamal**:* The El Gamal encryption algorithm is based on the DLP, and some public parameters can be shared by a group of users, called *domain parameters*. This parameter is a **large prime p** (such that $p-1$ is divisible by another prime q) and an element $q \in \mathbb{Z}_p^*$ of order p (or that the order is divisible by q. The **private key** e can then be computed by solving

$$e = g^d \; mod \; p$$

Example Key Generation:
In real applications, p should at least be 1024 bits!

$$p = 41 \quad g = 6 \quad d = 17$$
$$e := g^d = 6^{17} \equiv 26 \ mod\ 41$$

The private key d is a random number between 2 and $p-1$.
Example Encryption:
Encrypt the message M, assuming $M = 55$ with public key $(e, p, g) = (26, 41, 6)$. Then, use random number $k = 11$ and compute

$$c_1 = g^k ? 6^{11} \equiv 28 \ mod\ 41$$
$$c_2 = m \cdot e^k = 526^{11} \equiv 19 \ mod\ 41$$

The encrypted message $(c_1, c_2) = (28, 41)$ is now sent.
Example Decryption:
Received message is $(c_1, c_2) = (28, 41)$ and private key is $(d, p, g) = (17, 41, 6)$. This leads to

$$\frac{c_2}{c_1^d} = \frac{41}{28^{17}} \equiv 5 \ mod\ 41$$

*3) **ECC:*** ECC (Elliptic Curve)is a smooth curve in the so called *Weierstrassform.* Similar to the El Gamal cryptosystem, several public parameters can be shared among a group of users. These parameters are the elliptic curve group itself and the base point, the **private key** is a chosen integer d and the public key is given by $e = dB$.

*4) **Key Agreement:*** A major disadvantage of symmetric key cryptography is, that secret keys need to be distributed securely to the users. This problem can be solved using public key cryptography.

*5) **Digital Signatures:*** A digital signature is the equivalent of a handwritten signature, it binds someone's identity to a piece of information. A digital signature scheme consists of a **signature creation algorithm** and an associated **signature verification algorithm**.
Digital signatures *with appendix* require the original message as input to the verification algorithm, digital signatures *with message recovery* **do not** require the original message as input, it is recovered from the signature itself.

The **DSS** is a standardized signatures scheme with appendix, which describes an algorithm called **DSA** (Digital Signature Algorithm) based on DLP, it has recently been extended and is now based on elliptic curves.
In order to sign a message, the owner first computes the hash value of the message, this value and the private key are then fed to the signing algorithm. The output is a signature of the input message, anyone who wishes to verify the signature can simply take the message, the public key and the signatures and compute the verification algorithm.

D. Security and Attacks

There exist many different types of attacks on cryptographic primitives, usually the goal is to deduce the secret key.
A **passive attack** is one, where the adversary only monitors the communication channel or the side-channels of the communication devices, they target the implementation, not the algorithm itself.
An **active attack** is one, where the adversary attempts to alter the transmission on the channel or the computation inside the device (*fault attacks*). Side-channel attacks and fault attacks are so called **implementation attacks**, which pose a serious threat to cryptographic systems. To classify attacks, an assumption is made, that the adversary knows all details, except for the secret key.

- **ciphertext-only**: deduce secret key by only observing ciphertext
- **known-plaintext**: adversary has number of plaintext and corresponding ciphertexts
- **chosen-plaintext**: adversary chooses plaintext and is given corresponding chiphertext
- **adaptive chosen-plaintext**: choice of plaintext may depend on previous chipertexts
- **chosen-ciphertext**: adversary chooses the ciphertext and is given the corresponding plaintext
- **adaptive chosen-ciphertext**: choice of ciphertext may depend on previous plaintexts

Encryption schemes, which are vulnerable to ciphertext-only attacks, are considered completely insecure.

*1) **Security Models:*** There are several different models, under which the security of a cryptograhpic primitive can be evaluated.

- **Unconditional security**: In this model, the adversary is assumed to have unlimited computational power, so this model is also called *perfect secrecy*. A necessary condition is, that the secret key is at least as long as the message, because of that, systems like that are impractical.
- **Computational security**: Adversaries are assumed to have polynomial computational power, an algorithm is considered to be secure, if the best algorithm for breaking requires superpolynomial or exponential number of steps.
- **Provable security**: A cipher has provable security, if the difficulty of breaking can be shown to be equivalent to solving a well known, hard mathematical problem like DLP.
- **Practical security**: It is also related tot he computational power of the adversary, a cipher is considered practically secure, if the best *known* attack requires at least N operations, where N is a sufficiently large number.

E. *Review Exercises*

*1) **Name and explain the four security services (properties) cryptography can provide!:***

- confidentiality: "no unintended third party can understand the content"
- integrity: "no one can change the message while it is in transit"
- authentication: "get assurance of identity"
- non-repudiation: "sender cannot deny a message was sent"

*2) **What is Kerckhoffs' principle?:*** The adversary knows all details about the used algorithms and protocols, only the secret key is unknown.

*3) **What are the basic components of a block cipher?:*** The block cipher is a mathematical function, which takes plaintext as input, separates it into blocks, encrypts those with the secret key and has as output the ciphertext. Examples are DES (Data Encryption Standard) and AES (Advanced Encrpytion Standard).

*4) **What is the difference between a block cipher and a stream cipher?:*** A block cipher encrypts blocks of data using a function dependent on the key, while a stream cipher generates a bitstream from the key and XORs it with the data.

*5) **What is the difference between a MAC and an MDC?:*** MAC (Message Authentication Code)

- keyed hash-function (hash-value is signed)
- ensures data integrity and data authentication

MDC (Modification Detection Code)

- unkeyed hash-function
- ensure data integrity

*6) **What is the meaning of "pre-image resistance"? For what cryptographic primitive is it relevant?:*** It should be computationally unfeasible, to find a pre-image to a given hash-value, this is relevant for MDCs (Modification Detection Codes). So it should be efficiently possible, to find the pre-image of a hash-value, meaning, finding a value that is mapped to a certain hash value.

*7) **Explain how RSA encryption and decryption work!:*** Shown in (II-C.1).

*8) **What is a digital signature with appendix?:*** A digital signature binds someones identity to a piece of information, those with appendix require the original message as input to the verification algorithm.

*9) **Describe and explain the Diffie-Hellman Problem? What is it used for?:*** It is used for RSA, generally used for computing a common secret key. Two entities A and B agree to use a prime number p and a base g, now A generates a secret random number a and B b. Then A sends $g^a \ mod \ p$ and B sends $g^b \ mod \ p$ and both can compute $g^{ab} \ mod \ p$, which is their common secret key.

*10) **Name and explain at least three different models of security:*** Shown in (II-D.1).

*11) **What does the term "exhaustive key search" mean?:*** All keys in the key space are being tried out, until the correct one is found.

III. Electronic Signatures and Public Key Infrastructures

A. *Digital Signatures*

A **digital signature** is a data string, which associates a message in digital form with some originating entity.
DS with appendix require the original message as input
DS with message recovery do not require the original message as input, the message is recovered from the signature itself.
Randomized DS has several valid signatures for a given message.
Most of the signatures schemes which are in use today, belong to the class of (randomized) digital signatures with appendix, this is due to the fact, that in most schemes, the hash of a message is signed instead of the message. This has advantages, the signatures are much shorter, when the hash is signed and several attacks, which apply for signatures schemes with message recovery can be thwarted (vereitelt) by signing the hash. Breaks can be classified as

- **Total break**: An adversary is either able to compute the private key or find an efficient signing algorithm, which is functionally equivalent to the valid signing algorithm
- **Selective forgery**: An adversary is able to create a valid signature for a particular message or a class of messages a priori, creating the signature does not involve directly the legitimate signer.
- **Existential forgery**: An adversary is able to forge a signature for at least one message, there is little or no control over the message and the legitimate user may be involved in the deception.

Attacks can be classified to the capabilities of the attacker

- **Key-only attack**: Adversary knows signer's public key
- **Known-message attacks**: Adversary has signatures for a set of messages, which are known to him.
- **Chosen-message attacks**: Adversary obtains valid signatures from a set of messages, chosen by him.
- **Adaptive chosen-message attack**: Adversary may request signatures which depend on the signer's public key and he may request signatures, which depend on previously obtained signatures.

*1) **RSA Signature Scheme:***

*2) **DSA Signature Standard (DSS):***

*3) **ECDSA Signature Scheme:***

*B. **Review Exercises***

IV. **ELECTRONIC COMMERCE**

*A. **Review Exercises***

V. **NETWORK SECURITY**

*A. **Review Exercises***

VI. **IMPLEMENTATION SECURITY**

*A. **Review Exercises***

VII. **OPERATING SYSTEM SECURITY**

*A. **Review Exercises***

VIII. **PRIVACY**

*A. **Review Exercises***

APPENDIX