

# Curso de Data Security and Protection

Created by: Camila T.

[satie.tomikawa@gmail.com](mailto:satie.tomikawa@gmail.com)

## Topics for the exam:

- Virtualization
- Hashes
- Hybrid encryption
- E2E
- Diffie Hellman
- SQL/ Injection
- Forensics / BSI
- Risk analysis and risk assessment

## Questions:

1 - Describe the difference between the transport layer encryption and end-to-end encryption . Alice wants to send an email to Bob. Use the picture to point out for both encryption methods which parts of the communication can be forced by Alice to be encrypted.

Describe the term Hybrid encryption. Give an example where Hybrid encryption is used. Point out why we use Hybrid encryption and describe how HE is related to Symmetric and asymmetric ciphers.

2 - Decide whether 3 is a primitive root (generator) of group  $Z$  modulo 11. Write down the steps of your calculation in detail . Describe the results of your calculation and point out why 3 is a primitive root (generator) of group  $Z$  module 11 or not.

3 - You want to exchange symmetric encryption key between a communication partner and you. Therefore you both decide to use the Diffie-Hellman exchange.

You choose:

$p = 11$

$g = 6$

$a = 2$

4 - Your communication partner sends you the following additional result value:  $hB = 3$

Calculate  $hA$  - the value to send to your communication partner - calculate the shared key  $K$  and apply  $K$  to the following sentence: this is magic.

5 - You and your communication partner choose 11 as variable  $p$  for usage in a DHE. There are the following primitive roots modulo 11: 2, 6, 7 and 8

You communication partner suggested you use 9 as as variable  $g$ . Point out whether this is a good idea or not. No calculation needed.

6 - Point out two arguments that show that Strategic Preparation is an important part of computer forensics process.

7 - Describe one of the following IT security terms in deep:

- Denial service
- Identity theft
- Software vulnerability

8 - Describe the cryptographic term HASH. Point out how the application of a hash function on data differs from the application of an encryption function on data. Hashes are used to store passwords. Mention an additional use case for the application of hashes.

9 - Describe why hashes are used to store users passwords. Point out how attackers can break hashes and describe how Salt helps you to make it harder for attackers to break huge numbers of stored passwords.

10 - Alice and Bob want to exchange messages in a secure way. Describe in depth the differences between symmetric and asymmetric encryption. Point out the advantages and disadvantages of each technology.

#### **\*Important concepts relating to IT security**

There are three fundamental values of IT security: confidentiality, availability and integrity.

**Confidentiality:** information that is confidential must be protected against unauthorized disclosure.

**Availability:** services, IT system functions, data and information must be available to users as required.

**Integrity:** data must be complete and unaltered. In information technology, the term "information" is used to refer to "data" to which, depending on the context, certain attributes, such as the author or time of creation, can be assigned. The loss of integrity of information can therefore mean that this data has been altered without authorisation, that information relating to the author has been falsified or the date of creation has been tampered with.

## Virtualization

In computing, virtualization refers to the act of creating a virtual (rather than actual) version of something, including virtual computer hardware platforms, storage devices, and computer network resources.

Hardware virtualization or platform virtualization refers to the creation of a virtual machine that acts like a real computer with an operating system. Software executed on these virtual machines is separated from the underlying hardware resources. For example, a computer that is running Microsoft Windows may host a virtual machine that looks like a computer with the Ubuntu Linux operating system; Ubuntu-based software can be run on the virtual machine.

In hardware virtualization, the host machine is the actual machine on which the virtualization takes place, and the guest machine is the virtual machine. The words host and guest are used to distinguish the software that runs on the physical machine from the software that runs on the virtual machine. The software or firmware that creates a virtual machine on the host hardware is called a *hypervisor* or *Virtual Machine Manager*.

**Desktop virtualization** is the concept of separating the logical desktop from the physical machine.

One form of desktop virtualization, virtual desktop infrastructure (VDI), can be thought of as a more advanced form of hardware virtualization. Rather than interacting with a host computer directly via a keyboard, mouse, and monitor, the user interacts with the host computer using another desktop computer or a mobile device by means of a network connection, such as a LAN, Wireless LAN or even the Internet. In addition, the host computer in this scenario becomes a server computer capable of hosting multiple virtual machines at the same time for multiple users.

*-- This does not only virtualizes the desktop but also the hardware that is used to generate the graphic interface. Used in companies: the PC is not that good (ie cheaper) but the computational workload is done in the virtualized server.*

**Advantages** of (server) virtualization:

- Efficiency/ Performance
- Security (containment/ security layer/ snapshot)
- Layer of abstraction
- Better resource allocation

**Server virtualization** is a virtualization technique that involves partitioning a physical server into a number of small, virtual servers with the help of virtualization software. In server virtualization, each virtual server runs multiple operating system instances at the same time.

Server virtualization attempts to increase resource utilization by partitioning physical servers into several multiple virtual servers, each running its own operating system and applications. Server virtualization makes each virtual server look and act like a physical server, multiplying the capacity of every single physical machine.

*-- virtualization also gives you the possibility of a snapshot (image of the whole computer system) - not a backup!!*

*-- the server has a list of snapshots if something goes wrong during installation or update. Then you can use the snapshot to "go back" and restore as it was before.*

*-- Pro: snapshots are fast to restore*

*-- Also the MAC address has to be virtualized -*

*-- Note on Emulation vs virtualization: with emulation you can "change"/ emulate the type of underlying hardware (ex CPU) or other software that "does not exist". - Useful for testing code on different devices/OS (development) , video games, legacy systems (bank systems).*

**Sandbox** is a security mechanism for separating running programs. It is often used to execute untested or untrusted programs or code, possibly from unverified or untrusted third parties, suppliers, users, or websites, without risking harm to the host machine or operating system. A sandbox typically provides a tightly controlled set of resources for guest programs to run in, such as scratch space on disk and memory. Network access, the ability to inspect the host system or read from input devices are usually disallowed or heavily restricted. In the sense of providing a highly controlled environment, sandboxes may be seen as a specific example of virtualization. Sandboxing is frequently used to test unverified programs that may contain a virus or other malicious code, without allowing the software to harm the host device.

A sandbox implements an abstraction layer to the computer system.

It abstracts and restricts the access to the resources of the computer system.

Therefore it provides virtual resources like access to virtual devices like mass storage, system memory, USB ports, etc.

A program that runs in the sandbox must NOT know anything about the computer system outside the sandbox.

Bugs in sandboxes can lead to exploits.

- Such exploits can lead to sandbox break outs.

- This means that a program within the sandbox gains access to resources of the surrounding computer system, maybe with privileged access.

**Containment** is a methodology whereby access to information, files, systems or networks is controlled via access points. Much as a bank vault has only a single well-controlled entry and exit with various security procedures and protections, the security container also has controlled entries and exits known as connectivity points, though when using security containment, there may be more than a single connectivity point. Each of these may handle a specific type of service, such as electronic mail or file transfers. They may also control connections to other systems or networks, such as from the internal network to the global Internet or from an application to the files on the local system. The container has well defined security policies that it enforces and has security protection mechanisms to guard against attack.

-- Notes for Containment:

-- Software running in one VM cannot access other VM unless explicitly allowed to do so.

-- Problem:

**Buffer overflow**

A **buffer overflow**, or **buffer overrun**, is an anomaly where a program, while writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory locations.

One containment invades another one. Solution: dynamically allocation of memory.

Once the memory overflow problem is over, the memory may remain and some other VM may have access to it. Solution: make sure you clear the memory after allocating it.

## Backup

**Backup** is the process of making a secondary copy of data that can be restored to use if the primary copy becomes lost or unusable. Backups usually comprise a point-in-time copy of primary data taken on a repeated cycle – daily, monthly or weekly.

Remote data replication is sometimes assumed to be equivalent to backup, but this is not the case.

Backup involves making a copy or copies of data and storing them offsite in case the original is lost or damaged.

**Replication** is the act of copying data and then moving data between a company's sites, whether those be data centers, colocation facilities, public, or private clouds.

A **snapshot** is a point-in-time copy of data created from a set of markers pointing to stored data and is effectively a backup. Snapshots provide a variety of approaches that can supplement backup and provide rapidly accessible copies to which is it possible to roll back.

## Best practice – combine backup and other methods

A good data protection strategy combines a number of aspects that can include all of the above features. Short-term **snapshots** are great for dealing with user errors and some data corruption scenarios. More importantly, they are very **fast** (data can be reverted or restored in seconds), usually very space-efficient and in many cases, restores can be performed by the user, taking the workload off the backup administrator.

CDP (**Continuous data protection** (CDP), also called **continuous backup** or **real-time backup**, refers to backup of computer data by automatically saving a copy of every change made to that data, essentially

capturing every version of the data that the user saves) takes things a step further with more flexible recovery scenarios that trade off backup capacity and performance against restore granularity. This means CDP is great in environments where granular rollback is required and where that feature isn't provided through the application or database.

Finally, traditional **backup** offers a solid backstop position should a major hardware or site disaster occur. Although traditional backups don't necessarily provide the flexibility or efficiency of other methods, they offer a better long-term solution for data retention especially where backup policies dictate multiple backup copies in geographically dispersed locations.

- Notes for snapshots:
- RAID is data storage!!!
- Snapshot: fast, easy. Does not work for hardware problems.

## Check the download security

- 1- Check if connection is secure (SSL) - before downloading - *Trust the browser and the certification source*
- 2- On the source website check for hash sums (md5 -outdated or sha 256) or signatures (sig file for open PGP)
- 3- Verify the hash sum/signature(certUtil, SHASum, md5)
- 4- Compare the hexadecimal number from step 3 to the Hexnumber from step 2

## Checksum

A **checksum** is a small-sized datum derived from a block of digital data for the purpose of detecting errors which may have been introduced during its transmission or storage. It is usually applied to an installation file after it is received from the download server. By themselves, checksums are often used to verify data integrity but are not relied upon to verify data authenticity.

The actual procedure which yields the checksum from a data input is called a **checksum function** or **checksum algorithm**. Depending on its design goals, a good checksum algorithm will usually output a significantly different value, even for small changes made to the input. This is especially true of cryptographic hash functions, which may be used to detect many data corruption errors and verify overall data integrity; if the computed checksum for the current data input matches the stored value of a previously computed checksum, there is a very high probability the data has not been accidentally altered or corrupted.

Checksum functions are related to hash functions, fingerprints, randomization functions, and cryptographic hash functions. However, each of those concepts has different applications and therefore different design goals.

## Hashing

Hash functions compress a n (arbitrarily) large number of bits into a small number of bits (e.g. 512).

Properties

- One way; can not be reversed
- Output does not reveal information on the input
- Hard to find collisions (different inputs with same hash output)

A **hash function** is any function that can be used to map data of arbitrary size to data of a fixed size. The values returned by a hash function are called **hash values**, **hash codes**, **digests**, or simply **hashes**. Hash functions are often used in combination with a hash table, a common data structure used in computer software for rapid data lookup. Hash functions accelerate table or database lookup by detecting duplicated records in a large file. They are also useful in cryptography. A cryptographic hash function allows one to easily verify that some input data maps to a given hash value, but if the input data is unknown, it is deliberately difficult to reconstruct it (or any equivalent alternatives) by knowing the stored hash value. This is used for assuring integrity of transmitted data.

-- Hash sum in windows :  
-- CertUtil - hashfile <path of the file> <type of hash>

## Signatures

A **digital signature** is a mathematical scheme for presenting the authenticity of digital messages or documents. A valid digital signature gives a recipient reason to believe that the message was created by a known sender (authentication), that the sender cannot deny having sent the message (non-repudiation), and that the message was not altered in transit (integrity).

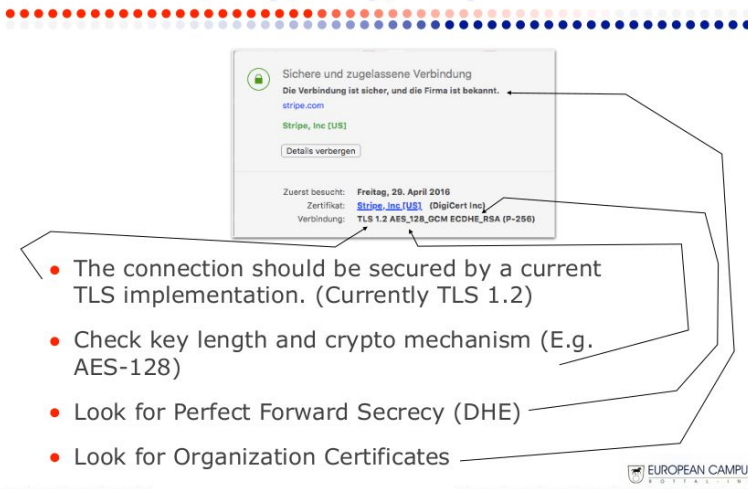
Digital signatures are a standard element of most cryptographic protocol suites, and are commonly used for software distribution, financial transactions, contract management software, and in other cases where it is important to detect forgery or tampering.

## Secure connection - https (with certification)

**HTTP Secure (HTTPS)** is an extension of the Hypertext Transfer Protocol (HTTP) for secure communication over a computer network, and is widely used on the Internet. In HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS, see under), or formerly, its predecessor, Secure Sockets Layer (SSL). The protocol is therefore also often referred to as **HTTP over TLS**, or **HTTP over SSL**.

The principal motivation for HTTPS is authentication of the accessed website and protection of the privacy and integrity of the exchanged data while in transit.

## Check the security of your internet connections, http, imap, smtp ...



Sichere und zugelassene Verbindung  
Die Verbindung ist sicher, und die Firma ist bekannt.  
stripe.com  
Stripe, Inc [US]  
Details verbergen

Zuletzt besucht: Freitag, 28. April 2018  
Zertifikat: stripe.com [US] (DigiCert Inc)  
Verbindung: TLS 1.2 AES\_128\_GCM ECDHE\_RSA (P-256)

- The connection should be secured by a current TLS implementation. (Currently TLS 1.2)
- Check key length and crypto mechanism (E.g. AES-128)
- Look for Perfect Forward Secrecy (DHE)
- Look for Organization Certificates

EUROPEAN CAMPUS  
FORTALINX

## Encryption

In cryptography, **encryption** is the process of encoding a message or information in such a way that only authorized parties can access it and those who are not authorized cannot.

Goals:

- Other persons cannot read the information;
- Control the access rights;

## Symmetric encryption

**Symmetric-key algorithms** are algorithms for cryptography that use the **same** cryptographic keys for both encryption of plaintext and decryption of ciphertext. The keys may be identical or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption (also known as asymmetric key encryption).

Example: **Advanced Encryption Standard (AES)**. The algorithm described by AES is a symmetric-key algorithm, **meaning the same key is used for both encrypting and decrypting the data**.

-- The key is the only secret between two persons, not the encryption technology or mechanism.

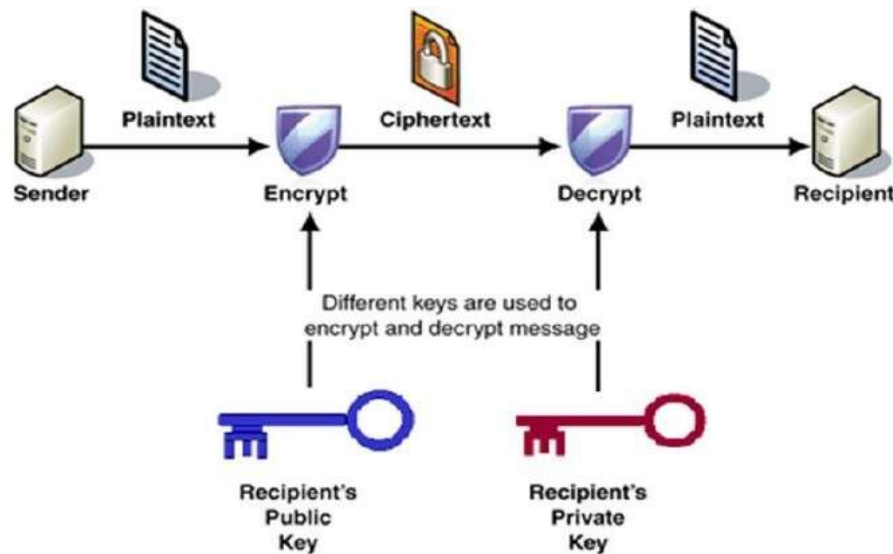
## Asymmetric encryption

**Public-key cryptography**, or **asymmetric cryptography**, is any cryptographic system that uses pairs of keys: *public keys*, and *private keys* which are known only to the owner. This accomplishes two functions: authentication, where the public key verifies that a holder of the paired private key sent the message, and encryption, where only the paired private key holder can decrypt the message encrypted with the public key.

In an asymmetric encryption system, any person can encrypt a message using the receiver's public key. That encrypted message can only be decrypted with the receiver's private key. To be practical, the generation of a public and private key -pair must be computationally economical. The strength of a public key cryptography system relies on the computational effort (*work factor* in cryptography) required to find the private key from its paired public key. Effective security only requires keeping the private key private; the public key can be openly distributed without compromising security.

Asymmetric cryptography systems often rely on cryptographic algorithms based on mathematical problems that currently admit no efficient solution, particularly those inherent in certain integer factorization, discrete logarithm, and elliptic curve relationships. Public key algorithms, unlike symmetric key algorithms, do *not* require a secure channel for the initial exchange of one or more secret keys between the parties.

Because of the computational complexity of asymmetric encryption, it is usually used only for small blocks of data, typically the transfer of a symmetric encryption key (e.g. a session key). This symmetric key is then used to encrypt the rest of the potentially long message sequence. The symmetric encryption/decryption is based on simpler algorithms and is much faster.



-- Notes for asymmetric encryption:

-- Steps:

1. Create key pair (public and private keys);
2. Exchange public keys;
3. Send message;
4. Decrypt message with own private key.

-- Asymmetric encryption: increase convenience/ performance, but the key is weaker (as opposed to symmetric encryption)

-- Tradeoff: the larger the key, the higher the protection, but less is the performance

-- Solution: hybrid encryption

Examples of asymmetric encryption: RSA and ElGamal



In **RSA**, the asymmetry is based on the practical difficulty of the factorization of the product of two large prime numbers, the "factoring problem".

**ElGamal encryption system** is an asymmetric key encryption algorithm based on the Diffie–Hellman key exchange.

Actually, for most applications where we want to use asymmetric encryption, we really want something a bit weaker: key agreement (also known as "key exchange"). When RSA or ElGamal is used for that, one party selects a random string, encrypts it with the public key of the other party, and the random string is used as a key for classical symmetric encryption. Therefore, we must add Diffie-Hellman to the list. You can imagine Diffie-Hellman as a kind of asymmetric encryption in which you do not get to choose the random value you are encrypting: less versatile than ElGamal, yet sufficiently powerful for most protocols where asymmetric encryption is used.

DH and ElGamal accept elliptic curve variants. They rely on hardness of discrete logarithm on elliptic curves, which is distinct from discrete logarithms modulo a big prime. Elliptic curve variants can use smaller fields, so while the mathematics are a bit more complex, performance is better.

## Hybrid encryption

In cryptography, a **hybrid cryptosystem** is one which combines the **convenience** of a asymmetric-key cryptosystem with the efficiency of a symmetric-key cryptosystem. Asymmetric cryptosystems are convenient in that they do not require the sender and receiver to share a common secret in order to communicate securely (among other useful properties). However, they often rely on complicated mathematical computations and are thus generally much more inefficient than comparable symmetric-key cryptosystems. In many applications, the high cost of encrypting long messages in a Asymmetric cryptosystem can be prohibitive. **This is addressed by hybrid systems by using a combination of both.**

A hybrid cryptosystem can be constructed using any two separate cryptosystems:

- a **key encapsulation** scheme, which is an asymmetric cryptosystem, and
- a **data encapsulation** scheme, which is a symmetric-key cryptosystem.

Note that for very long messages the bulk of the work in encryption/decryption is done by the more efficient asymmetric scheme, while the inefficient public-key scheme is used only to encrypt/decrypt a short key value.

Examples: the **TLS** protocol which uses an asymmetric mechanism for key exchange (such as Diffie-Hellman) and a symmetric-key mechanism for data encapsulation (such as AES). The **OpenPGP** (RFC 4880) file format and the **PKCS #7** (RFC 2315) file format are other examples.

### Example

---

To encrypt a message addressed to Alice in a hybrid cryptosystem, Bob does the following:

1. Obtains Alice's public key.
2. Generates a fresh symmetric key for the data encapsulation scheme.
3. Encrypts the message under the data encapsulation scheme, using the symmetric key just generated.
4. Encrypt the symmetric key under the key encapsulation scheme, using Alice's public key.
5. Send both of these encryptions to Alice.

To decrypt this hybrid ciphertext, Alice does the following:

1. Uses her private key to decrypt the symmetric key contained in the key encapsulation segment.
2. Uses this symmetric key to decrypt the message contained in the data encapsulation segment.

-- Notes for encryption in general:

-- It is important that plain messages have very different encrypted output messages - avoid computation of keys!

-- Plaintext attack - pairs of plain text and ciphertext (you need one pair only to decipher all existing messages).

-- Side channel attacks - you get the info by attacking ath that is on the periphery of the encryption algorithm.

-- If you have a weakness in the encryption in one of the layers, the whole thing will be weaker (ex: hard disk encryption weakness, you have to change the whole thing).

-- Hard disk encryption means that if you lose the computer no one will be able to get the info.

-- Do we need system data encryption if we have secure access? Yes:

- Access Control is NOT Encryption.
- Encryption is NOT Access Control.
- Data Encryption is NOT Data Encryption (There are different levels of system data Encryption. E.g. Disk (Startup?) Level Encryption. Encryption needs performance, thus we have to decide how important is performance, how important is security? What can happen when information leaks? We need risk analysis)

- Secure Access protects data accessed over official ways.
- System data encryption protects data accessed over unofficial ways.
- Thus, we need both.

### **Security and privacy: it is a tradeoff!**

If you want to prevent smart mass attacks, you have to let it go of some privacy (collect data for monitoring/security).

## Transport layer encryption

**Transport Layer Security (TLS)** – and its predecessor, **Secure Sockets Layer (SSL)**, which is now deprecated are cryptographic protocols that provide communications security over a computer network. Several versions of the protocols find widespread use in applications such as web browsing, email, instant messaging, and voice over IP (VoIP).

The TLS protocol aims primarily to provide privacy and data integrity between two or more communicating computer applications. When secured by TLS, connections between a client (e.g., a web browser) and a server (e.g., wikipedia.org) have one or more of the following properties:

- The connection is *private* (or *secure*) because symmetric cryptography is used to encrypt the data transmitted. The keys for this symmetric encryption are generated uniquely for each connection and are based on a shared secret negotiated at the start of the session (TLS handshake).
- The identity of the communicating parties can be *authenticated* using asymmetric cryptography. This authentication can be made optional, but is generally required for at least one of the parties (typically the server).

- The connection is *reliable* because each message transmitted includes a message integrity check using a message authentication code to prevent undetected loss or alteration of the data during transmission.

-- Note for TLS:

Stream cipher - A **stream cipher** is a symmetric key cipher where plaintext digits are combined with a pseudorandom cipher digit stream (keystream). In a stream cipher, each plaintext digit is encrypted one at a time with the corresponding digit of the keystream, to give a digit of the ciphertext stream. Since encryption of each digit is dependent on the current state of the cipher, it is also known as **state cipher**. In practice, a digit is typically a bit and the combining operation an exclusive-or (XOR).

Key stream is only valid for one session

--Transport layer encryption is very easy and convenient for the end user (you have only to configure it, not requiring technical skills)

Transport encryption means that the message is encrypted between two locations.

- E.g. an email is encrypted between your notebook and your mail server. But it is neither encrypted on your notebook nor on the mail server. It is maybe transferred in plain text from your mail

## End to end encryption

End to end encryption means that the message is encrypted between sender and receiver.

End-to-end encryption (E2EE) is a system of communication where only the communicating users can read the messages. In principle, it prevents potential eavesdroppers – including telecom providers, Internet providers, and even the provider of the communication service – from being able to access the cryptographic keys needed to decrypt the conversation. The systems are designed to defeat any attempts at surveillance or tampering because no third parties can decipher the data being communicated or stored. For example, companies that use end-to-end encryption are unable to hand over texts of their customers' messages to the authorities.

In an E2EE system, encryption keys must only be known to the communicating parties. To achieve this goal, E2EE systems can encrypt data using a pre-arranged string of symbols, called a pre-shared secret (PGP), or a one-time secret derived from such a pre-shared secret (DUKPT). They can also negotiate a secret key on the spot using Diffie-Hellman key exchange.

-- E2EE encryption works between the sender and the receiver (not only between 2 nodes, as opposed to the TLS)

End to end encryption means that the message is encrypted between sender and receiver.

- The sender encrypts the message on his notebook.
- The receiver decrypts the message on his notebook.
- The message is encrypted on all mail servers or gateways...

## Virtual private network

A **virtual private network (VPN)** extends a private network across a public network, and enables users to send and receive data across shared or public networks as if their computing devices were directly connected to the private network. **Applications running across a VPN may therefore benefit from the functionality, security, and management of the private network.**

To ensure security, data would travel through secure tunnels and VPN users would use authentication methods – including passwords, tokens and other unique identification methods – to gain access to the VPN. In addition, Internet users may secure their transactions with a VPN, to circumvent geo-restrictions and censorship, or to connect to proxy servers to protect personal identity and location to stay anonymous on the Internet.

A VPN is created by establishing a virtual point-to-point connection through the use of dedicated connections, virtual tunneling protocols, or traffic encryption. A VPN available from the public Internet can provide some of the benefits of a wide area network (WAN). From a user perspective, the resources available within the private network can be accessed remotely.

-- Notes on VPN -- remote control maintenance

Applies a secure layer over the internet and creates a new safe network.

Encryption works so that the rest of the internet cannot see what is going on inside the VPN, only that there is a communication channel.

You need keys: you need the secret (both ends have access to it) which also is also used to identify you. Problem: if you have to change the secret, it is a lot of work to change it .

Alternative: you can use a certificate, which is much easier to cancel/change if needed. If someone loses a laptop, you can simply cancel a certificate and get a new notebook with a new certificate.

Dedicated network - problem: no security by default

SSH - you can only use if the nodes are connected to the internet and works only node to node (point to point encryption). Secure Shell (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network. The best known example application is for remote login to computer systems by users.

## Key derivation function

In cryptography, a **key derivation function (KDF)** derives one or more secret keys from a secret value such as a master key, a password, or a passphrase using a pseudorandom function. KDFs can be used to stretch keys into longer keys or to obtain keys of a required format, such as converting a group element that is the result of a Diffie–Hellman key exchange into a symmetric key for use with AES. Keyed cryptographic hash functions are popular examples of pseudorandom functions used for key derivation.

## Diffie–Hellman key exchange

**Diffie–Hellman key exchange** is a method of securely exchanging cryptographic keys over a public channel.

### How it works:

Diffie–Hellman key exchange establishes a shared secret between two parties that can be used for secret communication for exchanging data over a public network.

The simplest and the original implementation of the protocol uses the multiplicative group of integers modulo  $p$ , where  $p$  is prime, and  $g$  is a primitive root modulo  $p$ . These two values are chosen in this way to ensure that the resulting shared secret can take on any value from 1 to  $p-1$ . Here is an example of the protocol, with non-secret values in blue, and secret values in red.

1. Alice and Bob agree to use a modulus  $p = 23$  and base  $g = 5$  (which is a primitive root modulo 23).
2. Alice chooses a secret integer  $a = 4$ , then sends Bob  $A = g^a \bmod p$ 
  - $A = 5^4 \bmod 23 = 4$
3. Bob chooses a secret integer  $b = 3$ , then sends Alice  $B = g^b \bmod p$ 
  - $B = 5^3 \bmod 23 = 10$
4. Alice computes  $s = B^a \bmod p$ 
  - $s = 10^4 \bmod 23 = 18$
5. Bob computes  $s = A^b \bmod p$ 
  - $s = 4^3 \bmod 23 = 18$
6. Alice and Bob now share a secret (the number 18).

Both Alice and Bob have arrived at the same value  $s$ , because, under mod  $p$ ,

$$A^b \bmod p = g^{ab} \bmod p = g^{ba} \bmod p = B^a \bmod p$$

More specifically,

$$(g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p$$

Note that only  $a$ ,  $b$ , and  $(g^{ab} \bmod p = g^{ba} \bmod p)$  are kept secret. All the other values –  $p$ ,  $g$ ,  $g^a \bmod p$ , and  $g^b \bmod p$  – are sent in the clear. Once Alice and Bob compute the shared secret they can use it as an encryption key, known only to them, for sending messages across the same open communications channel.

Of course, much larger values of  $a$ ,  $b$ , and  $p$  would be needed to make this example secure, since there are only 23 possible results of  $n \bmod 23$ . However, if  $p$  is a prime of at least 600 digits, then even the fastest modern computers cannot find  $a$  given only  $g$ ,  $p$  and  $g^a \bmod p$ . Such a problem is called the discrete logarithm problem. The computation of  $g^a \bmod p$  is known as modular exponentiation and can be done efficiently even for large numbers. Note that  $g$  need not be large at all, and in practice is usually a small integer (like 2, 3, ...).

-- Off-the-Record Messaging (OTR) is a cryptographic protocol that provides encryption for instant messaging conversations. OTR uses a combination of AES symmetric-key algorithm with 128 bits key length, the Diffie–Hellman key exchange with 1536 bits group size, and the SHA-1 hash function. In addition to authentication and encryption, OTR provides forward secrecy and malleable encryption.

Encryption of messages

- Authentication during the communication.
- Perfect Forward Secrecy, temp. Per message keys, DHE Key Exchange...
- Deniability of messages. Messages itself have no signature. After the communication no one can find out whether a message A was written by you or written by another person.

## Attacks

**Denial of service attack** - In computing, a **denial-of-service attack (DoS attack)** is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. In a

distributed denial-of-service attack (DDoS attack), the incoming traffic flooding the victim originates from many different sources. This effectively makes it impossible to stop the attack simply by blocking a single source.

**Identity theft** - Identity theft, also known as identity fraud, is a crime in which an imposter obtains key pieces of personally identifiable information, such as Social Security or driver's license numbers, in order to impersonate someone else.

Identity theft is categorized two ways: true name and account takeover. True-name identity theft means the thief uses personal information to open new accounts. Account-takeover identity theft means the imposter uses personal information to gain access to the person's existing accounts.

**Software vulnerability** - in general terms a vulnerability is a weakness which can be exploited by a Threat Actor, such as an attacker, to perform unauthorized actions within a computer system.

Vulnerabilities exist in all types of software.

Software development is not a perfect process. Programmers often work on timelines set by management teams that attempt to set reasonable goals, though it can be a challenge to meet those deadlines. As a result, developers do their best to design secure products as they progress but may not be able to identify all flaws before an anticipated release date. Delays may be costly; many companies will release an initial version of a product and then, when they find problems (or get reports from users or researchers), fix them by releasing security updates, sometimes called patches because they cover the holes.

**VM breakout** - In computer security, virtual machine escape is the process of breaking out of a virtual machine and interacting with the host operating system. A virtual machine is a "completely isolated guest operating system installation within a normal host operating system".

**Cache overflow** <https://meltdownattack.com/> - Meltdown and Spectre exploit critical vulnerabilities in modern processors. These hardware vulnerabilities allow programs to steal data which is currently processed on the computer. Meltdown is a hardware vulnerability affecting Intel x86 microprocessors, IBM POWER processors, and some ARM-based microprocessors. It allows a rogue process to read all memory, even when it is not authorized to do so.

**Side channel attack** - In computer security, a side-channel attack is any attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm itself (e.g. cryptanalysis and software bugs). Timing information, power consumption, electromagnetic leaks or even sound can provide an extra source of information, which can be exploited.

**Rainbow map** - A rainbow table is a precomputed table for reversing cryptographic hash functions, usually for cracking password hashes. Tables are usually used in recovering a password (or credit card numbers, etc.) up to a certain length consisting of a limited set of characters. It is a practical example of a space-time tradeoff, using less computer processing time and more storage than a brute-force attack which calculates a hash on every attempt, but more processing time and less storage than a simple lookup table with one entry per hash. Use of a key derivation function that employs a salt makes this attack infeasible.

You can secure your hashes by using salts.

- You add a unique salt to each password hash and hash it again.
- Use long salts.
- Do NOT use „normal“ hash algorithms. Use specific password hashes to slow down the generation process of a rainbow map. (E.g. bcrypt)

- This makes it even more complex to break the hashes.

**Note: Salt** (it is NOT a threat, it is actually a security element) In cryptography, a salt is random data that is used as an additional input to a one-way function that "hashes" data, a password or passphrase.. The primary function of salts is to defend against dictionary attacks or against its hashed equivalent, a pre-computed rainbow table attack.

A new salt is randomly generated for each password. In a typical setting, the salt and the password (or its version after Key stretching) are concatenated and processed with a cryptographic hash function, and the resulting output (but not the original password) is stored with the salt in a database. Hashing allows for later authentication without keeping and therefore risking the plaintext password in the event that the authentication data store is compromised.

Since salts do not have to be memorized by humans they can make the size of the rainbow table required for a successful attack prohibitively large without placing a burden on the users. Since salts are different in each case, they also protect commonly used passwords, or those who use the same password on several sites, by making all salted hash instances for the same password different from each other.

**SQL injection** - SQL injection is a code injection technique, used to attack data-driven applications, in which nefarious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker). SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.

SQL injection attacks allow attackers to spoof identity, tamper with existing data, cause repudiation issues such as voiding transactions or changing balances, allow the complete disclosure of all data on the system, destroy the data or make it otherwise unavailable, and become administrators of the database server.

-- Notes on SQL injection

- SQL injection is a code injection technique that might destroy your database.
- SQL injection is one of the most common web hacking techniques.
- SQL injection is the placement of malicious code in SQL statements, via web page input.

## Database and security

To access/hack a database:

1. Direct access to the DB: the hacker needs an IP address or DNS or Port.
2. How to get the username: you get REAL emails (work as usernames);
3. Create a huge dictionary with many combinations of first-name, family name, dates, numbers (ex: Bob123);
4. Then you try it but not too much (count the number of tries);
5. And you iterate over the usernames not over the passwords (it does not count as one million tries per account);
6. Then you can use IP masking from not so trustworthy servers (they usually use botnets)

Defense:

1. Authentication layer: username (public), password (secret, avoid impersonation)
2. Why do we need a username? 2 people can have the same password;

3. Username as ID: you can create user groups and assign user roles;
4. The combination of both username and password makes it stronger;
5. Password = secret (only the user knows it, the DB does not know it and it is not easy to guess)
6. Force the user to have strong passwords and add passwords constraints (stupid passwords are not allowed);
7. Limit the number of false tries or block it if the user is trying "too fast" (machine);
8. Block: same IP trying to login 1000 users (careful, because it may be root IP);

-- Notes for DB encryption:

Database encryption - con: lose performance

Usually it happens inside the DB, so if you eavesdrop the queries, you can get some data - that's why you need transport layer encryption also (TLS).

For the databases RDBMs, you may want to encrypt every single entry separately, instead of everything as only one bulk thing, but this will take a lot of processing time. But then again, it is easier to add encryption to the transport layer or to encrypt on column at a time.

## Databases/ SQL

Quick recap of last semester theory:

### Relational database management system (RDBMS)

RDBMS have the following characteristics: ACID = Atomicity (all changes are made or none), **Consistency** (modifications won't violate declared system integrity constraints), Isolated results independent of concurrent transactions), Durability (committed changes survive various classes of hardware failure). RDBMS uses structured data.

**NoSQL** was born out of the necessity to work with "big data" with **performance and scalability** (various servers). It can be used with unstructured data.

### One to many relationship:

In a relational database, a one-to-many relationship exists when one row in table A may be linked with many rows in table B, but one row in table B is linked to only one row in table A. It is important to note that a one-to-many relationship is not a property of the data, but rather of the relationship itself. A list of authors and their books may happen to describe books with only one author, in which case one row of the books table will refer to only one row of the authors table, but the relationship itself is not one-to-many, because books may have more than one author, forming a many-to-many relationship.

The opposite of one-to-many is many-to-one. For example, think of A as Authors, and B as Books. An Author can write several Books, and a Book can be written by several Authors.

In a relational database management system, such relationships are usually implemented by means of an associative table (also known as junction table or cross-reference table), say, AB with two one-to-many relationships A -> AB and B -> AB. In this case the logical primary key for AB is formed from the two foreign keys (i.e. copies of the primary keys of A and B).

SQL Data hierarchy:

Database

Table

Column/Row

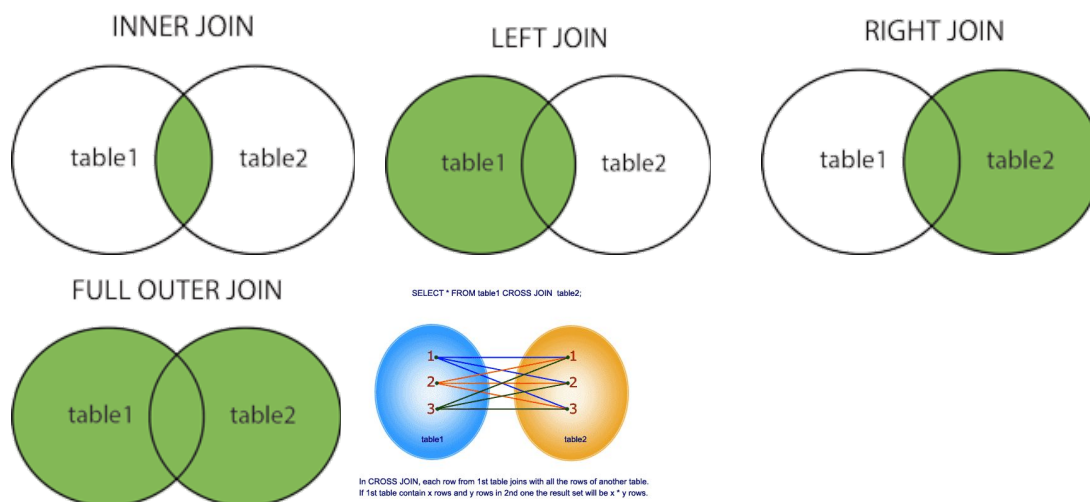
Data type



## Different Types of SQL JOINS

Here are the different types of the JOINS in SQL:

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Return all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Return all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Return all records when there is a match in either left or right table
- Self join: there is only one table



Example:

```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;
```

```
SELECT name, department FROM table INNER JOIN ON Product
```

## SQL commands

```
SELECT column1, column2, ... FROM table_name WHERE condition1 OR condition2
AND condition3 ...;
```

```
SELECT firstname, lastname FROM mytable WHERE id=2;
```

```
SELECT column1, column2, ...FROM table_name WHERE NOT condition;
```

```
SELECT column1, column2, ...FROM table_name ORDER BY column1, column2, ...  
ASC|DESC;
```

```
SELECT column_name(s) FROM table_name WHERE condition GROUP BY column_name(s)  
ORDER BY column_name(s);
```

```
SELECT COUNT(CustomerID), Country FROM Customers GROUP BY Country ORDER BY  
COUNT(CustomerID) DESC;
```

```
SELECT MIN(column_name) FROM table_name WHERE condition;
```

```
UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;
```

```
DELETE FROM table_name WHERE condition;
```

```
CREATE DATABASE myDB;
```

```
DROP myDB;
```

```
EMPTY myDB;
```

```
ALTER TABLE table_name ADD fkey_product integer NULL;
```

```
ALTER TABLE table_name DROP fkey_employee;
```

Other commands:

UPDATE

HAVING (after GROUP BY)

## Risk Analysis

**Risk management** is the identification, evaluation, and prioritization of risks (defined in ISO 31000 as *the effect of uncertainty on objectives*) followed by coordinated and economical application of resources to minimize, monitor, and control the probability or impact of unfortunate events or to maximize the realization of opportunities. Risk management's objective is to assure uncertainty does not deflect the endeavor from the business goals.

**Risk assessment** One way of creating a security concept is the traditional risk assessment. This entails devising individual security safeguards for an existing IT environment. The assets to be protected (IT systems, data, know-how etc.) are ascertained and examined to see which threats they are exposed to. The next stage is to analyse the probability of a security incident, the likely extent of damage, what security safeguards can be taken and what residual risks will remain after the security concept has been implemented. Risk assessments provide valuable information, but are associated with a lot of work because of the need to carry them out on an individual basis: experts with appropriate know-how are needed. The relevant input variables, such as the probability or extent of damage, are very difficult to ascertain and at best can only be calculated roughly. A risk assessment is therefore associated with high costs.

**The risk assessment is comprehensive but expensive.**

The BSI's IT-Grundschutz methodology constitutes an alternative means towards creating a security concept. The IT-Grundschutz methodology is based on the BSI standard 100-2 describing the ITGrundschutz approach and the IT-Grundschutz Catalogues containing the module, threat and safeguard catalogues. IT-Grundschutz builds on the fact that the majority of the IT systems and applications used in practice are run in a similar fashion and in comparable operational environments. Servers under UNIX, client PCs under Windows and database applications are examples here. Through the use of these typical components, the same kinds of threats to IT operations are found on a recurring basis. If there are no special security requirements, these threats are largely independent of the specific application scenario. This leads to two ideas for proceeding:

- An all-embracing risk assessment is not always necessary: the threats to IT operations and the probability of damage resulting from these threats can be roughly calculated if certain assumptions are made.
- It is not always necessary to develop new security safeguards for every application: groups of standard security safeguards can be derived which offer an appropriate and adequate degree of protection against these dangers under normal security requirements.

#### Risk Analysis

- Circumstances
- Situations
- Probability
- Impact
- Measures
- Tests

#### Examples of Threats

Fire, Unfavourable Climatic Conditions, Water, Pollution, Dust, Corrosion, Natural Disaster, Environmental Disaster, Major Events in the Environment, Failure or Disruption of the Power Supply , Failure or Disruption of Communication Networks, Failure or Disruption of Mains Supply, Failure or Disruption of Service Providers, Interfering Radiation, Intercepting Compromising Emissions, Interception of Information / Espionage, Eavesdropping, Theft of Devices, Storage Media and Documents, Loss of Devices, Storage Media and Documents, Bad Planning or Lack of Adaption, Disclosure of Sensitive Information , Information or Products from an Unreliable Source, Manipulation of Hardware or Software, Manipulation of Information, Unauthorised Access to IT Systems, Destruction of Devices or Storage Media, Failure of Devices or Systems, Malfunction of Devices or Systems, Lack of Resources, Software Vulnerabilities or Errors, Violation of Laws or Regulations, Unauthorised Use or Administration of Devices and Systems, Incorrect Use or Administration of Devices and Systems, Abuse of Authorisations , Absence of Personnel, Attack, Coercion, Extortion or Corruption, Identity Theft, Reputation of Actions, Abuse of Personal Data, Malicious Software, Denial of Service, Sabotage, Social Engineering, Replaying Messages, Unauthorised Entry to Premises , Data Loss, Loss of Integrity of Sensitive Information

## Forensics

Computer forensic is an important part of cyber security and should be established in IT management processes.

- Investigation of IT incidents and support of criminal or disaster investigations
- Collecting information and creating knowledge through analysis of computer hardware, software, data and network traffic
- Improvement of cyber security processes

- Security of running IT systems

It is very important to start forensic investigations as fast as possible.

- We need Risk Analysis to detect risks and weaknesses related to a system.
- We need methods to detect incidents as fast as possible.
  - Intrusion detection systems
- We need methods to track activities on computers and within the network.
  - E.g. Identification of users and devices in a network.

**The BSI terms following steps as part of a computer forensic process.**

- **Strategic preparations (documentation)**
- **Operative preparations**
- **Collecting data, data recovery**
- **Core investigation**
- **Data analysis**
- **Documentation**

Strategic preparation helps to gain as much information about an possible incident. **BEFORE IT HAPPENS**. Risk Analysis is an inherent tool of the strategic preparation.

- Preparing the computer system to collect data that is suitable to support the future investigation of incidents.
  - Good documentation of the system
  - Time synchronization
  - Unique user and client authentication
  - Data logging
  - Intrusion detection systems

Operative preparation helps to speed up the forensic investigation team as fast as possible, after an incident was detected.

- Speed is very important.
- Identification of data sources.
- Data preparation.

Collection and recovery of data, every bit can be important.

- One big issue is to evaluate whether a specific data is important or not.
  - Cause every information can be important after further analysis, we collect as much data as possible without losing time, especially if it is volatile data.
  - Although important is the recovery of data.
- Use cryptographic technology to ensure that nobody can compromise the raw data after extraction, e.g. digital signatures and hashes.

The core investigation evaluates the collected data and transfers the raw data into «something readable».

- Extraction of pictures and documents from disk images.
- Cracking of encryption keys.
- Decryption of devices.
- Recreation of session information and network communication

The forensic data analysis creates knowledge out of the collected data.

- Add semantics to files.
- Create time lines.
- Find relations between data on different computers or between files.
- Detect weaknesses of the IT system.
- Detect intrusion strategy of an attacker.
- It is important to safe the integrity of the data and to be sure that the data is authentic.

Documentation is an important part of forensic investigations.

- Documentation is important to show what happened, why it happened and to present the results.
- Especially dependencies between different parts of an incident are important.
- It is also important to document how we investigated the incident and which persons and tools were part of the investigation.
- Another information is whether it is possible to repeat the investigation with the same results, whether you understand the tool chain and whether the tool chain is widely accepted by experts.

Documentation matters: detect something extraordinary

Fast response to the incident - we know the relations to other systems/ knowledge about the systems running on servers.

The understanding of dependencies between parts of an incident are very important (events that led to the incident)

The BSI terms six different types of forensic methods.

- Operating System
- Filesystem
- Explicit methods of intrusion detection
- IT Applications
- Scalability
- Data preparation and data analysis

The Operating System delivers important persistent and volatile forensic data.

- Management of resources
  - Network management
    - Network connections
    - Network configurations
  - CPU management
  - Device management
  - Persistent and Volatile Data
- System management and user management
  - System logs

The Filesystem provides access to persistent data.

- Files
- File Metadata
  - File attributes
  - Access rights
  - Journaling and version control

- Partition-Gap
- Swap (System memory, extension of RAM), Slack (Sector slack, file slack, old data)
- Suspend to disk data

If the user removes files from the file system, the files are not deleted from the storage. There is often a lot of information about old files left. Forensic tools can read or restore such information.

The usage of intrusion detection systems must be planned. This is part of the strategic preparation.

- Specific Intrusion Detection Systems
- On Access Anti Virus programs
- Deep Package Inspection Firewalls
- Detection of attacks and intrusions
- Collection of data about attacks and intrusions

Logfiles of IT applications can contain important forensic information.

- Logfiles
- Databases
- Network Servers (DNS, DHCP)
- Web Server / Mail Server
- File Server
- Local Applications
- Web browsers
- Calculation programs

Scalability increases interesting information during an incident.

- Scalability
  - Collection of more detailed messages in log files.
  - Activation of additional network scanners or file scanners
  - Those measures can affect the normal network or application performance.
- Data Preparation and Data Analysis
- Tools for data consolidation, data synchronization.
- Tools for extraction and recovery of data.

COBIT and ITIL contain requirements and recommendations concerning IT Security Management.