Computer Network I

Reti di Calcolatori I

Università di Napoli Federico II – Scuola Politecnica e delle Scienze di Base
Corso di Laurea in Informatica

Riccardo Caccavale

(riccardo.caccavale@unina.it)

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

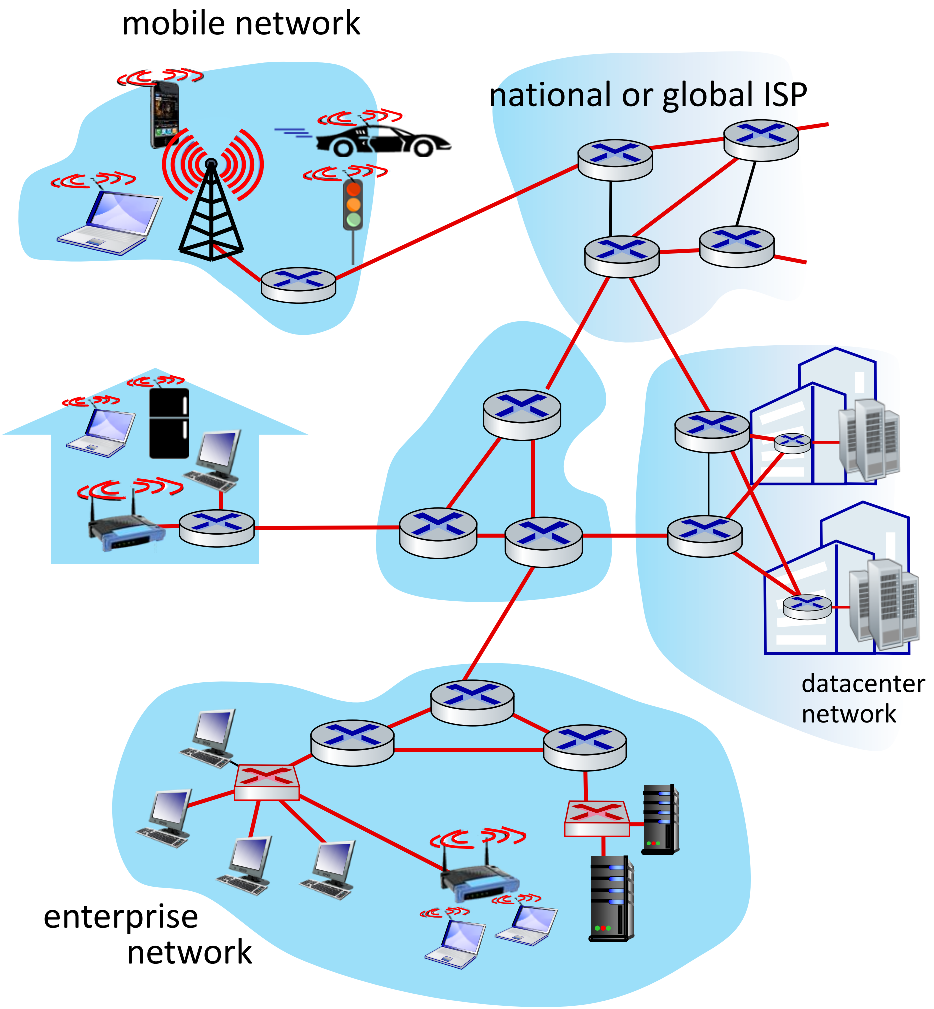
Network Security

Introduction

- The **network security** is the field that studies **possible attacks** to networks and **possible ways to prevent them**.

- Networking is part of our lives (Internet connection is considered almost as water or power supply) and **a large quantity of our sensible data travels on networks**.

- There are **several types of attacks** having different purposes and different mechanics.
  - Attacks evolve along with the **technology** and the networks' **audience**.



Network Security

Attacks: Malwares

- A **malware** is **malicious software** that can be transferred to a computer through the network (e.g., downloaded file, e-mail attachment, etc.).

- Once malware infects our device it can harm in different ways:
  - Forcing a system to show commercial advertisements (**adware**).
  - Showing false alarm messages to induce users to download malwares (**scareware**).
  - Deleting (**wiper**) or encrypting (**ransomware**) our files.
  - Collecting private information such as passwords, security numbers, etc. (**spyware**).
    - For example, **keyloggers** are software recording (logging) the keys struck on a keyboard.
  - Getting root privileges of our system (**rootkits**).
  - Turn a device into a slave or foothold to attack other devices (**zombie** or **botnet**).

Network Security

Attacks: Malwares

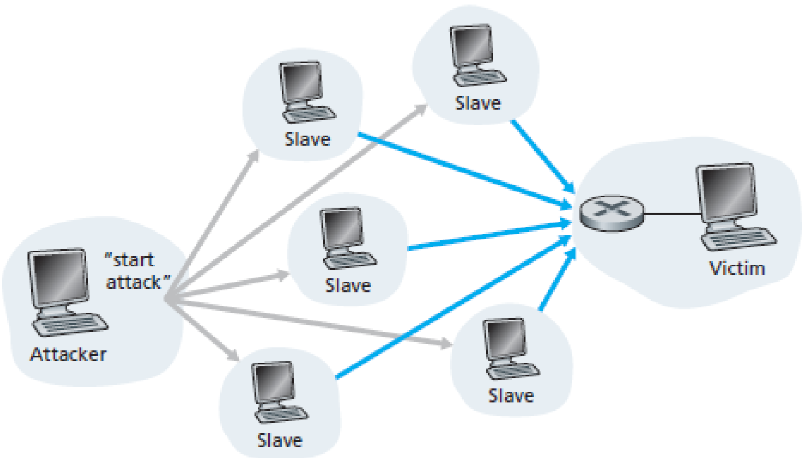
- Todays malware are often **self-replicating**: once it infects one host, from that host **it seeks entry into other hosts over the Internet**, and from the newly infected hosts, it seeks entry into yet more hosts.

- Malware can spread in the form of a virus or a worm:
  - **Viruses** are malware that **require some form of user interaction** to infect the user's device.
    - For example, an **e-mail attachment** containing malicious executable code that self-replicates by sending similar mails to users' contacts.
    - These are typically **disguised as legitimate software** components (**trojan horses**).
  - **Worms** are malware that **can enter a device without any explicit user interaction**.
    - For example, a user may be running **a vulnerable network application to which accepts the worm without intervention**. The worm than scans the network for hosts running a similar application.

Network Security

Attacks: DoS

- **Denial-of-Service** (DoS) attacks are quite common and are designed to render a network, a host, or other piece of infrastructure (e.g., Web servers, DNS, etc.) unusable by legitimate users.
- There are 3 types of DoS attacks:
  1. **Vulnerability attack**: sending suitable messages to vulnerable applications or OSs in order to let them stop or crash.
  2. **Bandwidth flooding**: sending a large quantity of packets to the targeted host, preventing legitimate packets from reaching the server.
  3. **Connection flooding**: establishing a large number of half-open or fully open TCP connections at the target host, so it stops accepting legitimate connections.



Example of a **DDoS** (Distributed DoS) attack using a **botnet** of zombies (or slaves).

Network Security

Attacks: Packets Sniffing

- **Packets sniffing** involves a passive receiver (**sniffer**) that records a copy of relevant packets from a target host trying to steal sensitive information (aka **eavesdropping**).
  - Sniffers can be deployed in all kind of **broadcast** networks (wired or wireless) simply by copying packets that are meant for different destinations instead of discarding them.
  - As for **non-broadcast** networks, a sniffer can be put into a malware (spyware) and used to infect network devices (e.g., routers) so that all forwarded traffic is also copied.

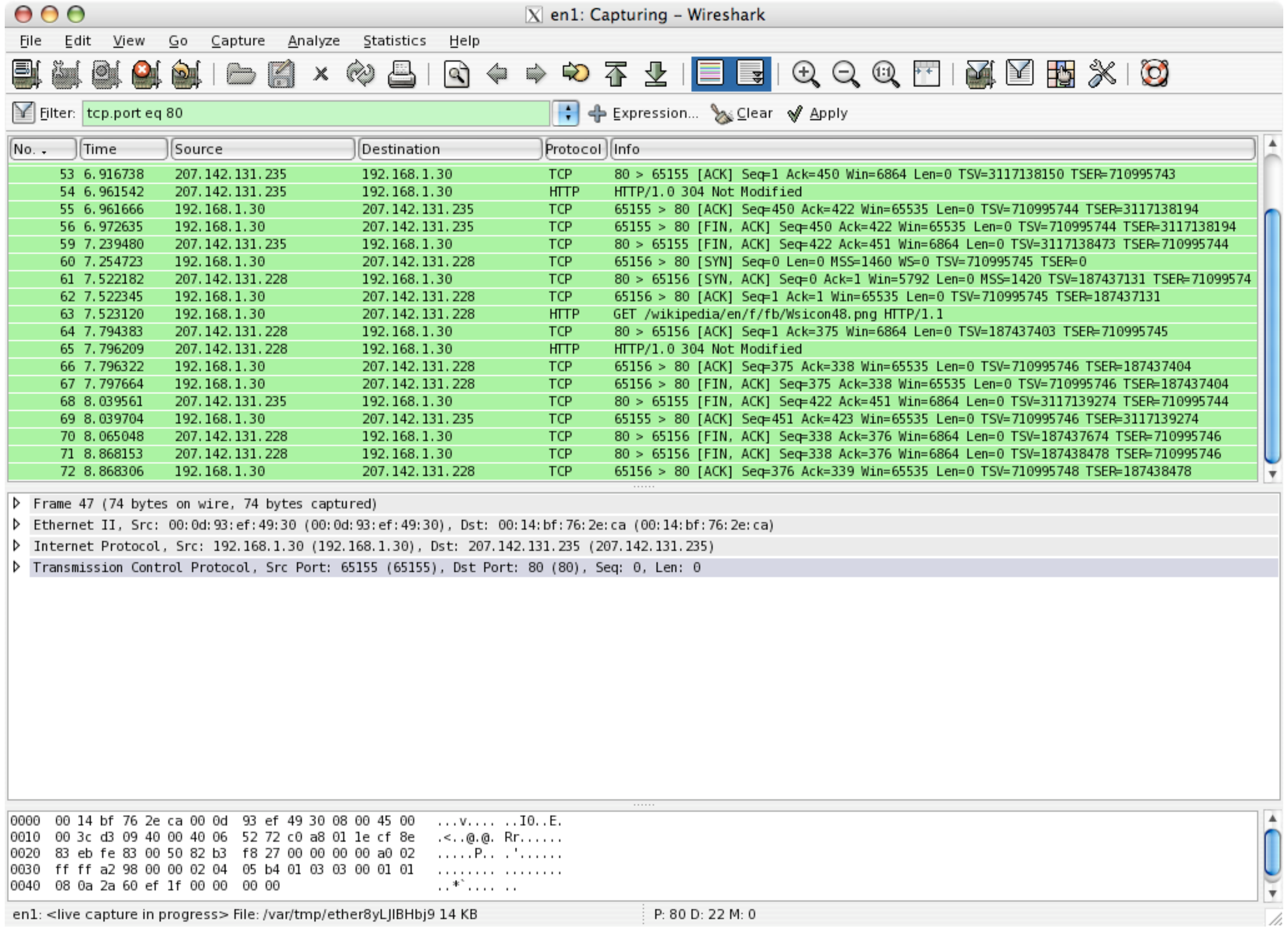
- Since sniffers are **passive** (no additional traffic is injected into the network) **these are very difficult to detect**.

- To prevent sniffing **cryptography** approaches can be used.

Network Security

Attacks: Packets Sniffing

- There are several sniffers freely available on Internet. A notable example of packet sniffer is **Wireshark** (ex **Ethereal**).
- Wireshark is a **packet/protocol analyzer**, it is mainly used for legitimate purposes (troubleshooting, network monitoring, creation of new protocols, etc.).
- Wireshark is available in different OSs (Linux included).



GUI of Wireshark from Wikipedia

Network Security

Attacks: IP Spoofing

- **IP spoofing** is a technique that allow malevolent hosts to inject into a network packets with false source addresses.
  - It can be used **in combination with applications vulnerability** to attack specific hosts being masqueraded as another user.
  - It can also be used **for DoS attacks** (alternative to botnets) as messages from different source IPs are **more difficult to filter**.

- Spoofing can also be used for **man-in-the-middle** (MitM or MiM) attacks, in which the attacker is placed in between 2 communicating hosts disguised as both.
  - The 2 hosts think they are communicating each other, while **they are actually communicating with the attacker**.

- To prevent spoofing, we can use **message integrity checks** and **end-point authentication**, allowing us to determine if the message has not been modified or if the message originates from the right source.

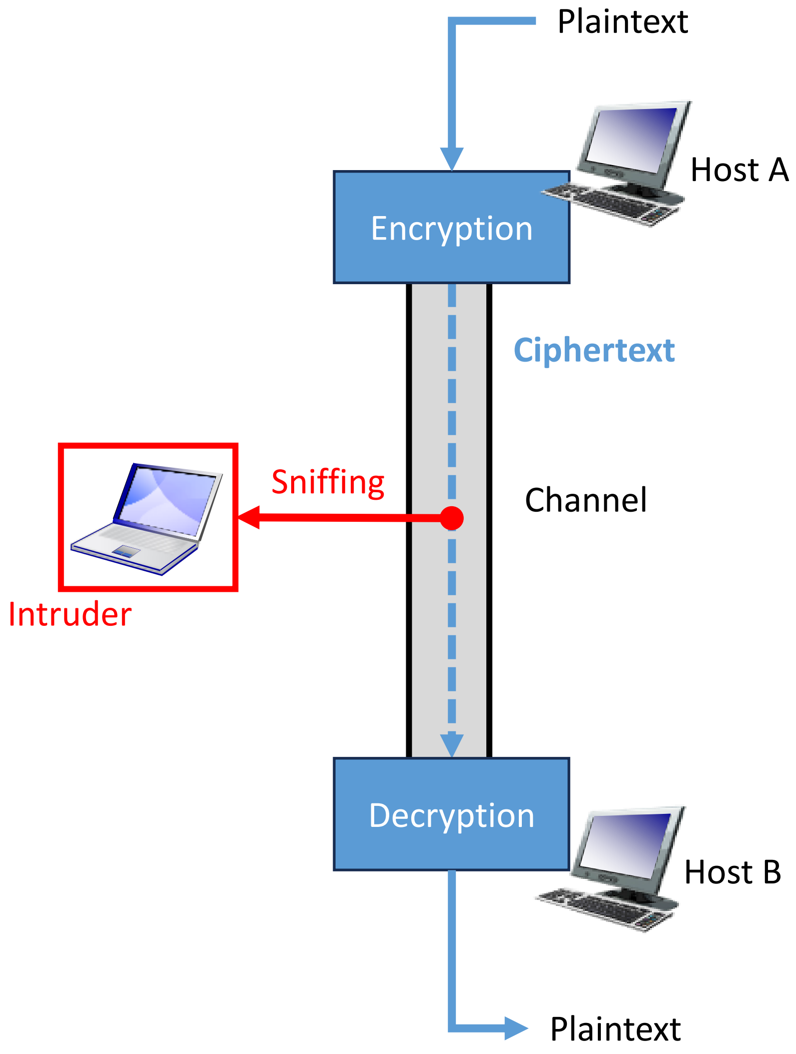
- Given the previous attack types we can now define the set of proprieties that a **secure communication** should guarantee:
  - **Confidentiality**: only the sender and intended receiver should be able to understand the contents of the transmitted message (sniffing avoidance).
  - **Message integrity**: the content of the communication must not be altered, either maliciously or by accident.
  - **End-point authentication**: both the sender and receiver should be able to confirm the identity of the other party involved in the communication (spoofing avoidance).
  - **Operational security**: to rely on a network infrastructure that prevents malicious hosts to sneak into the communication.

- The **first 3 proprieties are software-based** the last one (operational security) typically relies on **specific hardware** (firewalls, intrusion detection systems).

Network Security

Cryptography

- A **cryptographic technique** allows a sender to **disguise data** so that it becomes incomprehensible for an intruder.
  - **Intruders can gain no information** from the intercepted data, but the **receiver must be able to recover the original data** from the disguised data.
- In its initial form the message is called **plaintext** (or cleartext) and is readable for everyone.
- Before to inject the message into the channel a host uses an encryption algorithm to transform the message into a non-readable form called **ciphertext** which must be decrypted when received.



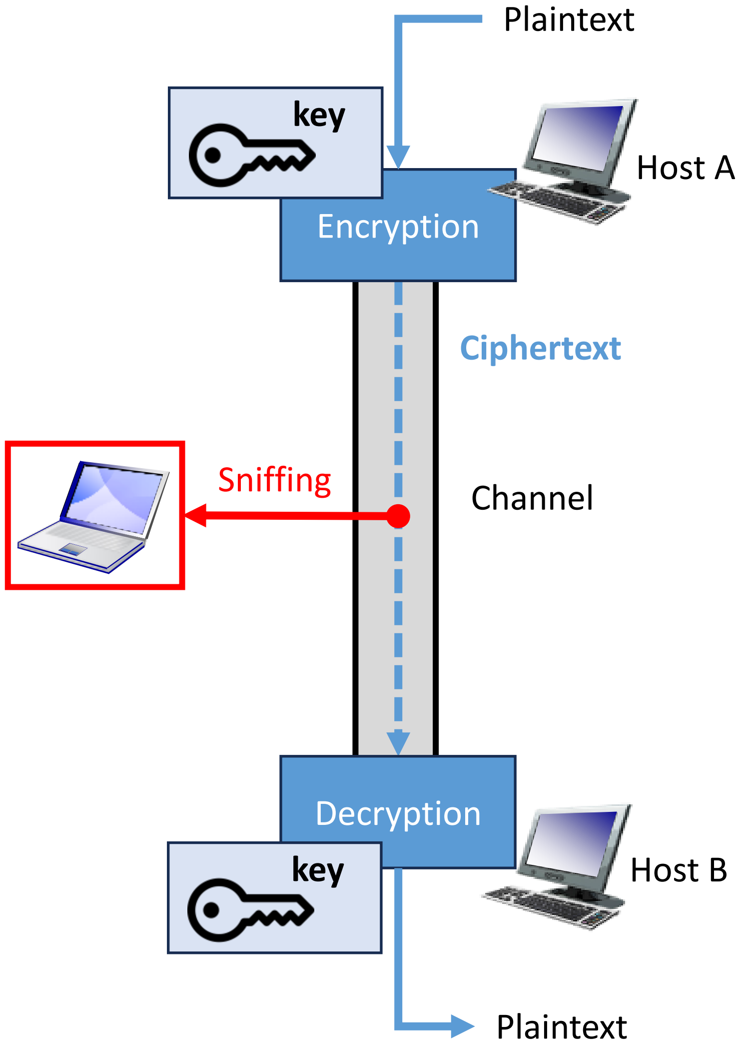
Network Security

Cryptography: Keys

- In many modern cryptographic systems, including those used on the Internet, the **encryption technique is known and standard for everyone** (including the intruder). The **unknown part** of the algorithm are the **encryption/decryption keys**.

- A **key** is an alphanumerical string that must be provided to the encryption/decryption algorithm in order to encrypt/decrypt the messages.

- Encryption and decryption keys can be **identical** (symmetric) or **different** (asymmetric).



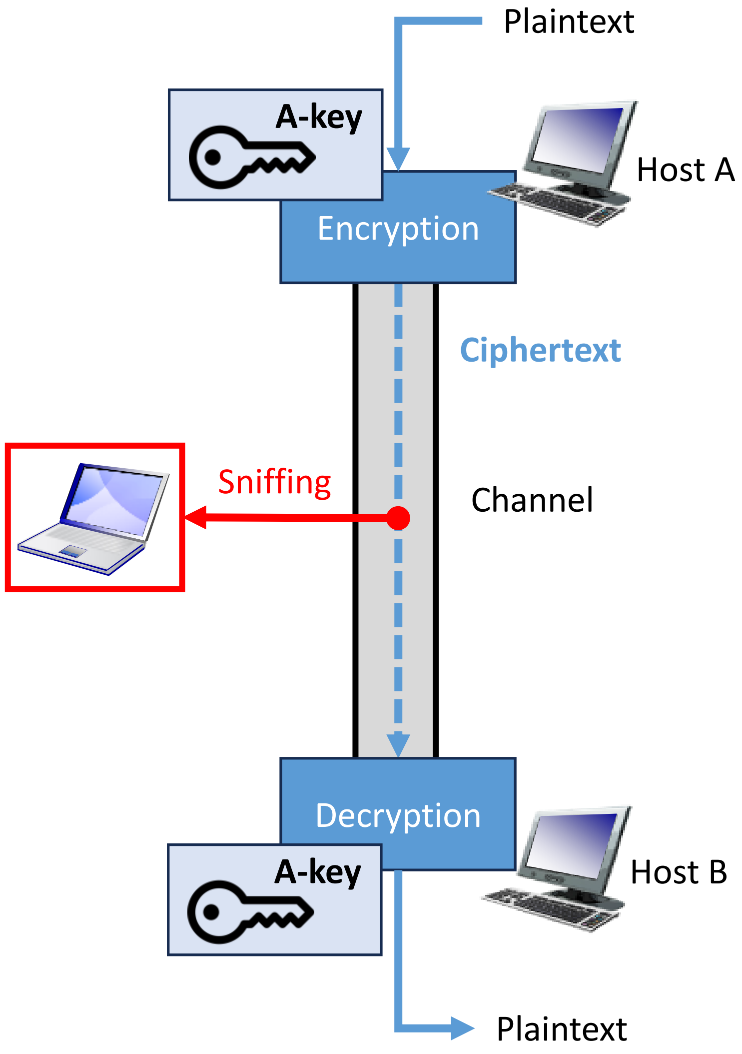
Network Security

Cryptography: Symmetric

- In **symmetric cryptography** there is **only one key** that is used for both encryption and decryption.

- **Encryption**: the plaintext message along with the key is passed to the encryption algorithm to generate a ciphertext that can be safely sent through the network.

- **Decryption**: the cyphertext message along with the key is passed to the decryption algorithm to recreate the initial plaintext message (readable).

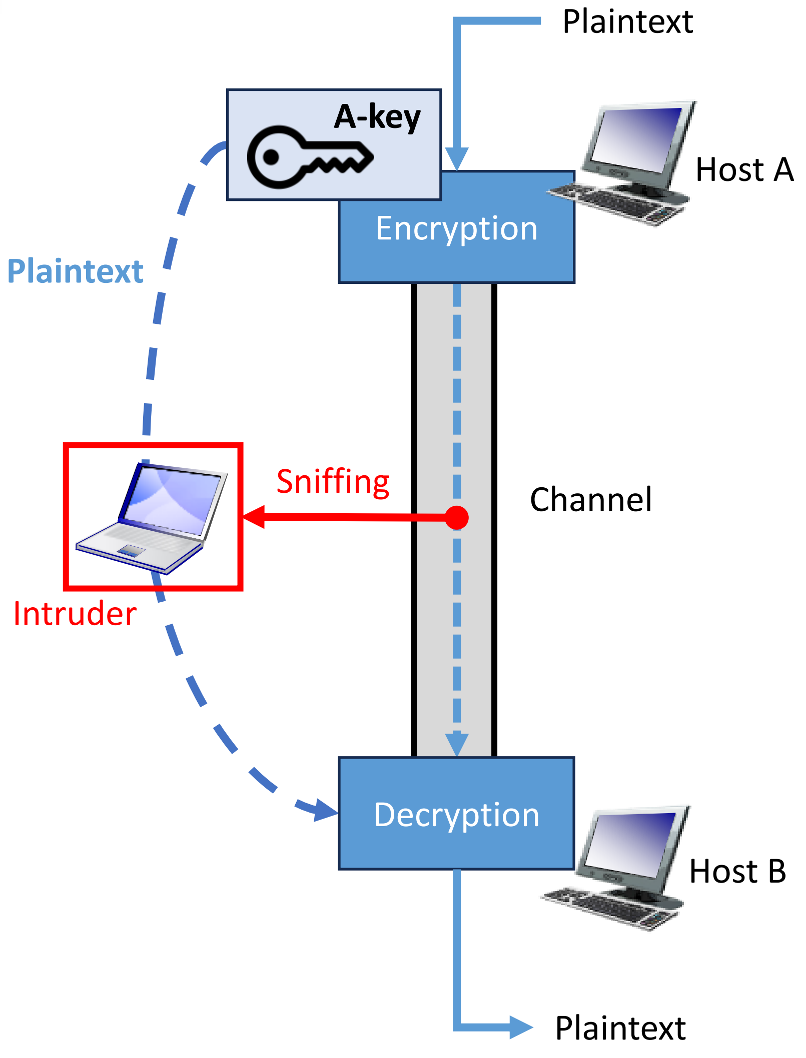


Network Security

Cryptography: Key Exchange

- The **problem** with symmetrical cryptography, or single-key cryptography, is that it requires the secret key to be communicated (**key exchange problem**).
  - Hosts can use a **secure channel** to exchange the key.
  - Hosts can use some **protocol** that allows them to "converge" on a shared key.

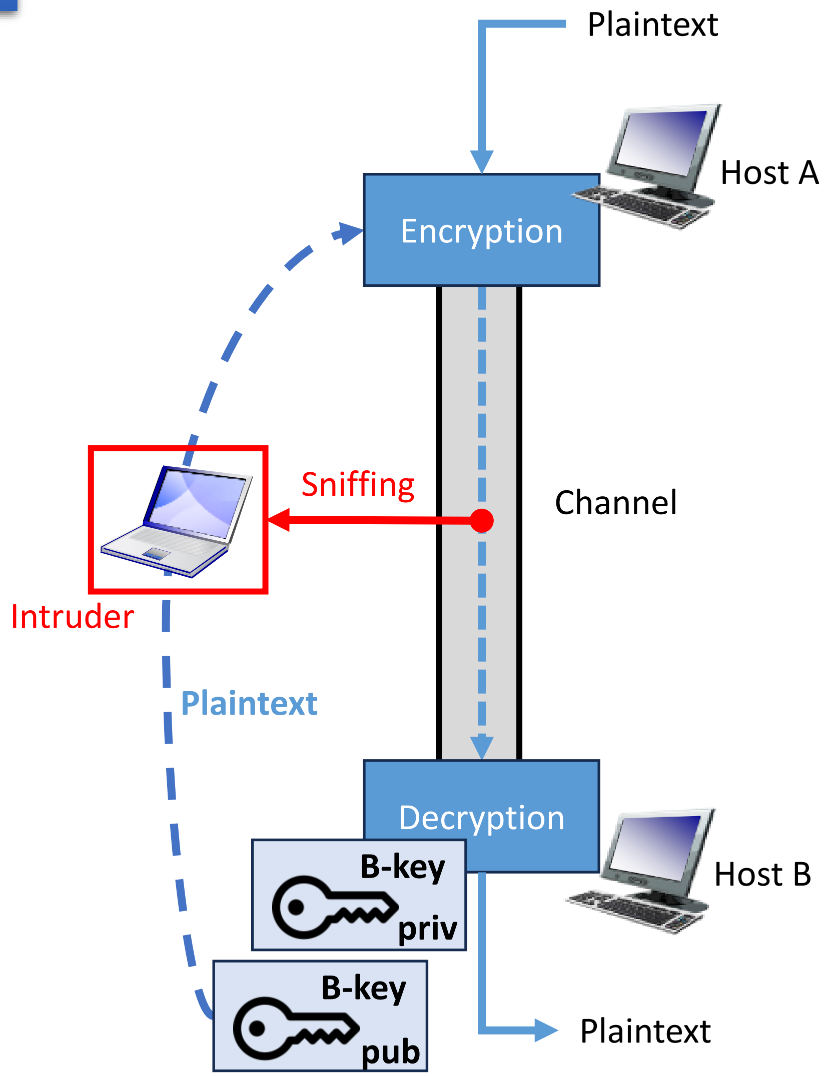
- If two parties **cannot establish a secure initial** key exchange, **they won't be able to communicate securely** without the risk of messages being intercepted and decrypted by a third party who acquired the key during the initial key exchange.



Network Security

Cryptography: Asymmetric

- In **asymmetric cryptography** there is a **two-key system** (public and private keys).
  - A message that is **encrypted with one key must be decrypted with the other** and vice versa.
- The idea is that the **public key can be sent over non-secure channels** or shared in public, while the private key is only available to its owner.
- A typical approach is to use **public key for encryption and private key for decryption**:
  - If the intruder sniffs the public key, it is still impossible for him/her to decrypt messages.
  - Host A will use B-key-public to encrypt messages that can only be seen through B-key-private, which is only into B's hands.



Network Security
Certification Authority

- In public key cryptography it could be useful to **verify if a public key really belong to the entity** with whom you want to communicate.
  - Otherwise, **we could have someone's else key** (attacker) and we could encrypt messages that are readable by illegitimate entities.

- Binding a public key to a particular entity is typically done by a **Certification Authority** (CA), whose job is to validate identities and issue certificates. A CA has the following roles:
  1. A **CA verifies that an entity is who it says it is**. There is no protocol for that, **one must trust the CA** to have performed a suitably rigorous identity verification.
     - It works like a natural selection process: **if a CA is unreliable no on will trust it**.
     - There are **several federal or statal CA** that provide a reasonable reliability, but we still have to trust them.
  2. Once the CA verifies the identity of the entity, **the CA creates a certificate that binds the public key of the entity to the identity**. The certificate contains the public key and a globally unique identifier of the owner (for example**, a name or an IP address**).

- **Message integrity** (also known as **message authentication**) is the problem of checking if:
  1. The message has **not been tempered with**.
  2. The message has been indeed **originated by the expected host**.

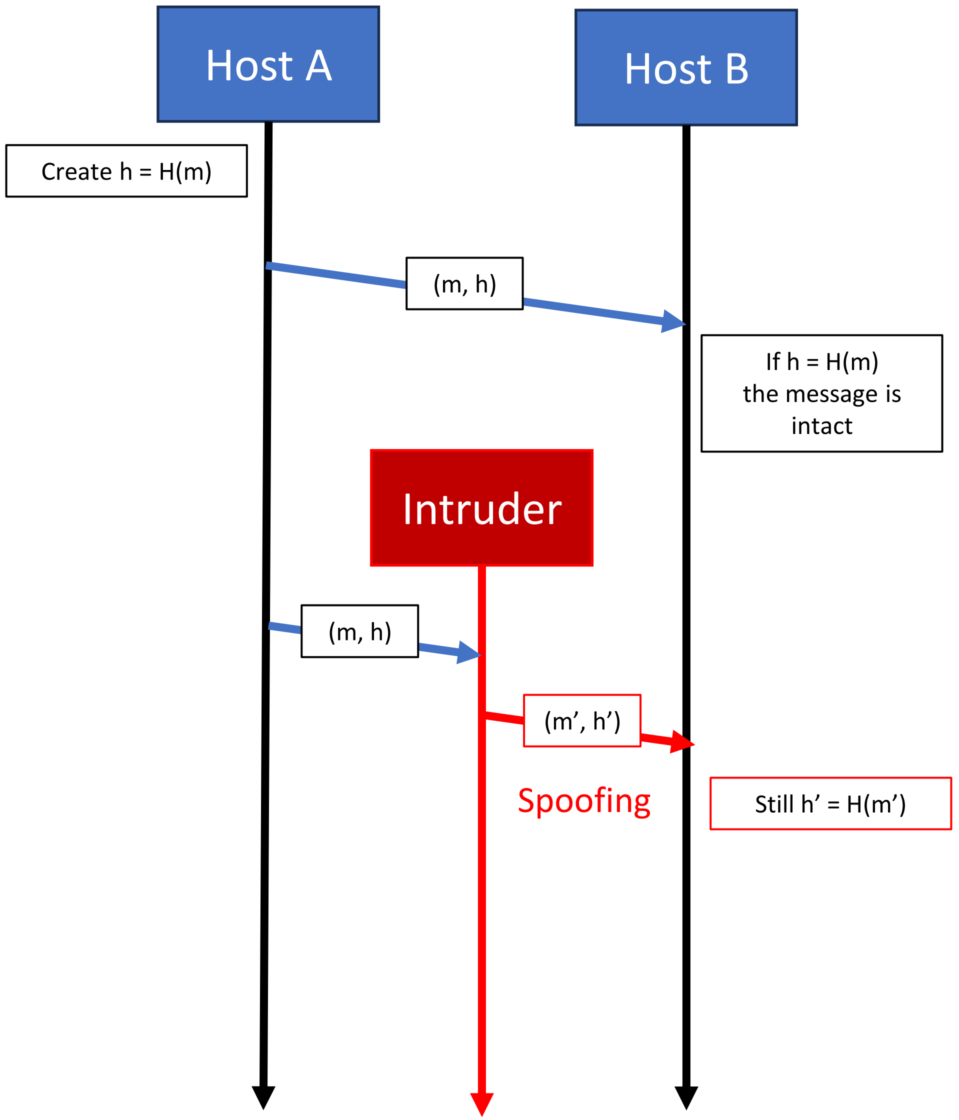
- We can create a **check-item** similarly as the checksum or CRC. Typically, a hash function is used to create such item.
  - Remind: a **hash function** is any function that can be used to **map data of arbitrary size into fixed-size values**.

- A **cryptographic hash function** is a function H that converts a message x into a fixed-size string $H(x)$ so that it is very hard (computationally infeasible) to find another message y such that $H(y) = H(x)$.

Network Security
Message Integrity

- As in checksum or CRC, **we can attach this hash into the message**:
  1. Host **A creates message m and calculates the hash h = H(m)**.
  2. Host **A appends h to the message m**, creating an extended message (m, h), and **sends the extended message to B**.
  3. Host **B receives the message (m, h) and calculates H(m)**. If H(m) = h, the message is intact.

- **This approach is obviously flawed**. An intruder may **spoof the whole message** (m,h), creating **a new “ad hoc” one** (m’,h’) that is still consistent with the hashing function H.



Network Security

Message Integrity

- To avoid this, A and B need a **shared secret** *s* (a **shared key or a password**) which is a string known only to them.
  - This basically **works as a symmetric encryption** where *s* is the unique private key.

- Assuming such *s* exists then:
  1. Host **A creates a message m + s** (as a concatenation of message and secret) and **calculates the hash h = H(m + s)** aka a **message authentication code (MAC)**.
  2. Host **A appends the MAC to the message m**, creating an extended message (m, H(m + s)), and sends the extended message to B.
  3. Host **B receives the extended message** (m, h) and knowing s, **calculates the MAC H(m + s)**. If H(m + s) = h, the message is intact.

- As in all symmetric approaches, also here **we need to exchange such secret**.
  - This **secret can be exchanged combining asymmetric cryptography and certificates**.