8.2

SYLLABUS

Introduction to computer systems, networks and security

Security Protocols and Firewalls

Carsten Rudolph
Updated 7 October 2018

Security Protocols

Cryptography provides the basic mechanisms to protect information. it can provide confidentiality, authentication, non-repudiation, and integrity of data that is transferred over insecure communication links that can potentially be accessed by a malicious entity. However, just applying a cryptographic algorithm is not enough. We need **protocols** that specify how to use the algorithms to satisfy particular security goals.

Some tasks for security protocols are as follows:

- Negotiate the particular cryptographic algorithms to be used and parameters such as key length, padding scheme, block chaining
- Include necessary information, such as random numbers to identify current messages, addresses of communication partners, identifiers for particular messages.
- Interoperability, i.e. specify how to talk to each other
- Key exchange and establishment of short-term session keys
- Combining different security mechanisms e.g. multi-factor authentication, hybrid approaches combining public key cryptography with symmetric cryptography

This module looks at a few common examples of security protocols.

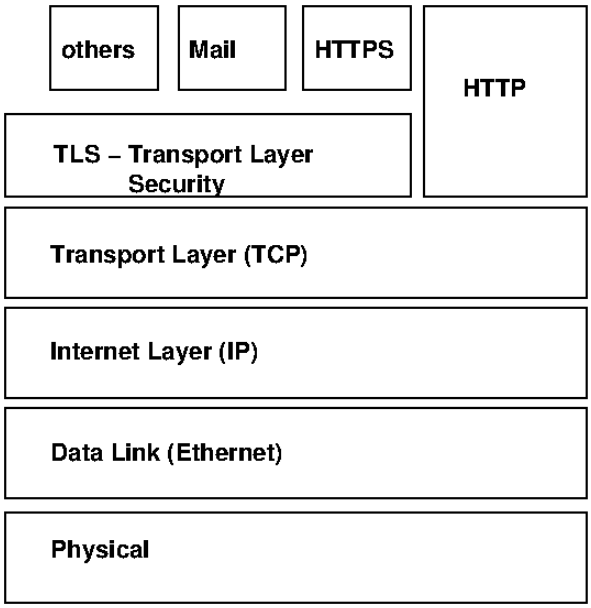
TLS – Transport Layer Security

Every time we use a web-browser to access information from a server and the browser shows that this communication is secure, Transport Layer Security TLS is the protocol that is used. It was originally developed by Netscape (one of the first browsers with graphical user interface) as

Secure Socket Layer SSL protocol. The idea was to establish a protocol, that is independent from the application layer protocol and acts like a What is a socket? for the application layer.

SSL Version 2.0 in 1995 was the first practically used version with some weaknesses and was quickly replaced by SSL 3.0 in 1996. Based on SSL, the IETF (Internet Engineering Taskforce) published the successor Transport Layer Security 1.0 as What is an RFC? 5246 in 1999. The current version of Transport Layer Security is TSL 1.2 published as IETF RFC 5246. All previous versions should be disabled due to security problems.

The following figure shows the location of TLS in the TCP/IP protocol stack. It sits between the Application Layer and the Transport Layer. Thus, in the view of the application it is basically a secure transport layer. When using HTTP over TLS, the combination is called HTTPS (Hyper Text Transfer Protocol Secure).



TLS in the TCP/IP protocol stack

The main goal of the TLS protocol is to establish and use a shared key to protect messages (confidentiality and integrity/authenticity). It is structured in three sub-protocols:

1. **TLS Handshake**

   This is the first phase of the protocol. It negotiates the parameters for authentication and encryption. In particular, these are the used algorithms (e.g. RSA and AES), their modes (which padding, what kind of integrity protection) and the key lengths. Then, both sides can be authenticated. This means, they somehow prove that they are who they claim they are. Note, that this authentication step is optional and when used for HTTPS usually only the server is authenticated using a certificate for the public key, while the client only

checks the server's authenticity, but does not provide any identity to the server. Finally, the TLS Handshake protocol uses a mechanisms like the Diffie Hellman Key Exchange to establish a shared key plus session information that is used to distinguish this session from other TLS sessions.

2. **TLS Record**
   Using a TLS *ChangeCipherSpec* message, the new symmetric key is now used to encrypt the data that is exchanged. In the TLS Record phase, all content is encrypted.
3. **TLS Alert**
   This protocol part immediately closes the current session.

Diffie Hellman Key Exchange

TLS uses a mechanisms called *Diffie Hellman Key Exchange* to establish a new symmetric key. The main advantage of this method is that both sides contribute random numbers to the new key (thus both can control that the key is actually a new key) and only public information is exchanged that is then used by both sides to compute the key. The scheme was published in 1976 by Whitfield Diffie and Martin Hellman.

1. Alice and Bob agree on values g and n (these values are public)

2a. Alice generates random A and $a=g^A \bmod n$

2b. Bob generates a random B and $b=g^B \bmod n$

3. They exchange a and b

4. Shared key is $K= b^A = g^{BA} = g^{AB} = a^B \bmod n$

Anybody without one of the secret values A or B cannot generate the new key K:

- A and B are secret values.
- $a=g^A$ and $b=g^B$ are public
- To get A or B, the attacker would need to compute A from g^A
- This discrete logarithm is difficult to compute!

Certificates

TLS uses *certificates* verify that a particular public key belongs to a particular server. E.g. if the server for monash.edu gets an HTTPS request, it will start the TLS Handshake and provide it's public key plus a certificate to the client. The certificate is digitally signed by a trusted certification authority and provides information on the public key, i.e. the date when it will expire, that it belongs to monash.edu, which algorithms it can be used with and what it can be

used for. As long as monash.edu keeps the matching secret key as a secret, it is guaranteed, that the TLS protocol will run with the correct server owned by Monash University.

Common Web browsers (e.g. Chrome, Firefox, Safari, Internet Explorer, Opera) come with a predefined list of certification authorities. Thus, the certificate check is automatically executed within the browser during TLS Handshake. The user is only informed, if the certificate check fails. If it is successful, the connection will be established, and the browser will somehow indicate that the connection is secure (e.g. by showing a lock next to the address bar).

Certificates are not a fool proof solution. There are a few issues, that can make working with certificates difficult. First, if private keys are not properly secured, a certificate cannot provide correct information on who owns and can use the key. If a certification authority is compromised, a large number of certificates can become invalid at once. The solution for compromised keys or compromised certificates is *certificate revocation*. In addition to the trusted certification authorities, browsers also keep an updated list of revoked certificates. These revocations need to be maintained until the affected certificates are expired. A second problem is the relation between domain names, certificates and actual entities. If for example somebody registers a domain name that is very similar to a bank's name or another legitimate entity, it is obviously possible to get a valid certificate for this different domain name. Now, all automated checks are valid, but the page is not the page the user expects. Therefore, the user needs to have a close look at what is actually shown in the address bar or even better always type critical addresses by hand.

Virtual Private Networks VPNs

A virtual private network VPN virtually places a computer into another network. The following examples show two examples of a use of VPNs:

- An employee of a company is working from home or travelling and wants to access the internal network of the company. For security reasons, this network is behind a firewall and a computer outside the network cannot get access. A VPN now creates a secure tunnel between the teleworker's computer and a VPN gateway. The firewall will let traffic pass that goes to the VPN gateway (or it is even an integrated device with firewall functionality that acts as VPN gateway). The computer gets an IP address from the internal network and all traffic between this network and the computer (and all Internet access if configured correctly) goes through the secure tunnel to the internal network.
- A service that is not available from a particular network (e.g. it is blocked because of a location / country policy). In this case, a computer can be connected to a VPN gateway in another location (e.g. in a different country). Then, the computer gets an IP address that

belongs to the address range of that location and the computer is virtually ¨moved¨ and can get access to previously blocked services.

A VPN can be established using different security protocols, e.g. using TLS or IPSec. It is important to know that security of a VPN connection is not ¨end-to-end¨ but only between the computer and the VPN gateway. Packets will be unencrypted as soon as they leave the gateway.

Firewalls

A firewall in buildings would protect flames from one part of a building to reach another part. Thus, it would block a fire to spread. Obviously, a firewall is some kind of barrier blocking something malicious.

In computer networks, a firewall is a barrier between some (more secure) internal network and a (less secure) outside network (i.e. the Internet). Obviously, a firewall cannot totally block traffic if a computer shall get access to the Internet. Thus, a firewall in a computer network **filters traffic.**
Security rules define what can get through and what is blocked (in both directions in and out)

Packet filter firewall

A packet filter firewall operates on the Network layer (and above) and filters networks packets. Thus, it basically either blocks a packet or lets it through. It usually uses a static set of filter rules that decide based on source and destination IP Addresses, protocols, ports, and the current stage of a connection.

Packet filter firewalls are a standard security mechanism that is cost-effective and easy to deploy. Nevertheless, it is essential and can prevent a large number of attacks and considerably reduces the attack surface for devices in a network behind the firewall. These devices are no longer directly visible from the Internet.

A packet based firewall needs to look at information in the packet:

- Firewall software inspects the first few bytes of TCP or UDP headers in an IP packet
- Finds application protocol and port (e.g. HTTP with port 80 or SMTP with port 25)
- Often, traffic from inside out is allowed (except when explicitly blocked, One would for example block network management traffic from inside out, SNMP on UDP ports 161, 162)
- Traffic from outside in should be blocked if not explicitly permitted
- Different rules need to exist for existing connections and new connections. If a computer has established a TLS session with a server, answers coming from the server should not be blocked by the firewall.

- Rules also depend on applications/services running behind the firewall. If for example a web server or a mail server runs behind the firewall, ports for these services should be open.

Thus, for firewall rules, one needs to define:

Source IP address (or range), Destination IP address (or range) and Destination port (or range)

Source IP address rules can consider for example that any address should be able to connect to a web server. Alternatively. access can be restricted to specific IP addresses. if some computers should not be able to access the Internet.

For destination IP addresses, it is important to consider special addresses, such as the IP address of the server running a service that should be accessed.
Obviously, it is not a good security practice to allow *any* IP address as a destination for incoming traffic. Requests for a web server, for example, should only pass the firewall if the destination address is actually the correct address of the server. A basic rule for firewall configuration is to never allow "any" address.

The destination port (one network layer above the IP address) specifies the service accessed via a particular port.
For example, a web-server needs incoming connections on port 80 (HTTP) and port 443 (HTTPS). Again, we should never allow *any* port.

Where to place a firewall?

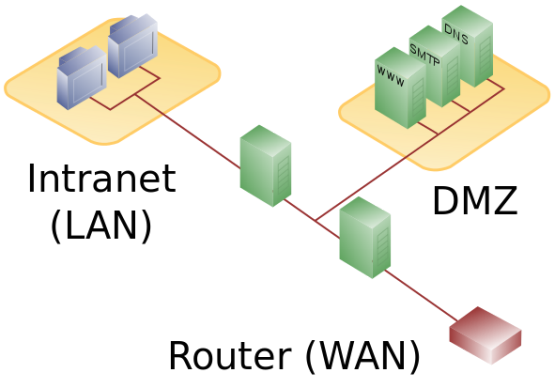
Firewall software on PCs and individual devices is essential. No device should just allow all traffic to reach services potentially listening on a port if it is not explicitly required. For example, a computer that needs remote login, needs to have the port for the remote login protocol accessible. In addition, there needs to be another firewall that prevents arbitrary traffic to reach the computer. If, for example, a computer's network card driver has a vulnerability, a software firewall on the computer cannot help. The firewall will only be able to filter traffic after it has gone through the network card. Thus, the vulnerability could be exploited before the firewall gets to apply filter rules. In a home network, the router usually also acts as a firewall that separates devices from the external network.

Proper placing in a company network is a bit more complicated, as various services need to be considered. Even a very simple company network has an internal network with PCs, servers, printers, etc. and mail server, web-server, VPN gateway, and other services. While the internal network should not be directly accessible, web-server, mail server or VPN gateway need to be accessible.

A good solution for an enterprise network uses two firewalls to create a so-called *demilitarized zone* DMZ. A DMZ creates a zone between two firewalls, that is considered to be less secure than

the internal network, but still protected from direct access.

The following picture shows a DMZ with two firewalls. The two green boxes between the Intranet, the WAN (Internet) and the DMZ are two firewalls, while the boxes in the DMZ are server that shall be available from the internal network as well as from outside, via the router connecting the Intranet to the Internet.



Firewall placement for DMZ

License: 0, Wikimedia Commons

While a main focus of firewalls lies on blocking malicious traffic from the outside, filtering outgoing traffic (from inside out) is also important. Some examples for filtering outgoing traffic are the following:

- Prevent malicious software to send out data
- Block outbound traffic from critical network areas or computers
- Only allow outbound HTTP traffic through a proxy (IP addresses of devices are not visible from the outside)
- Logging of denied outbound traffic can help to detect infections

In addition to the pure security functionality, firewalls often implement additional functionality, such as proxies and NAT network translation:

Network and port-address translation (NAT). Internal network maps the externally visible IP address(es) to internal IP addresses not visible to the outside. Proxies (e.g. for HTTP) can hide individual devices in the internal network. These functions are not directly security functionalities, but hide some information from outside attackers.

Why firewalls are not enough

Filtering becomes very complex, as many applications require Internet access. Examples include social networks, remote access (TeamViewer, RDP, etc.), unified messaging and video conferencing (Skype, WeChat, Zoometc.), collaboration tools (Google Docs, OneNote, OneDrive, iCloud, Facebooketc.). Filtering is additionally made more difficult by port hopping (applications change their ports during a session), hiding traffic in TLS encryption (TLS can mask application traffic, e.g. via TCP port 443), using non-standard ports, or tunneling application data within other services: Example is peer-to-peer file-sharing or messengers running over HTTP / HTTPS

Another issue is that firewalls only work on traffic crossing the network perimeter. this type of security has obvious constraints:

- Firewalls don't help against internal attackers
- Once an attack was successful, firewalls cannot help
- Internet of things, mobile networks, etc.

Security controls beyond firewalls – Intrusion detection and Intrusion prevention

IDS and IPS

IDS – Intrusion Detection Systems are active components that do not only look at single packets, but analyse complete packet streams and other system activities to identify malicious behaviour. An IDS monitors network and/or system activities, generates alerts when potentially malicious activity is found and logs information about activities. Note that IDSs *detect*, but do not prevent or automatically react to attacks. They mainly raise alerts and provide information on the incident.

IPS – Intrusion Prevention System have additional functionality. In addition to detecting adversarial or unknown behaviour, they also attempt to block or stop malicious activities.

The following list provides some examples of monitoring actions by IPSs

- Detect port scans – an attacker might scan a large range of ports in order to find an open port with a vulnerable service. A prot scan produces a range of packets with many different port numbers and are easy to identify.
- Detect OS fingerprinting attempts – an attacker would like to know as much as possible about a computer. Particular versions of an operating system might have vulnerabilities that can be exploited. OS fingerprinting can also produce a particular pattern that might be detected.
- Look for specific attacks (e.g. buffer overflow) – known attack patterns can be detected by IPSs

- Find and block known malware – similar to virus scanners, IPSs can try to detect strings in packets that belong to known malware
- Detect server massage block (SMB) probes – SMB is a protocol for file exchange that sometimes is used for attacks. Some version had vulnerabilities.

Possible reactions can for example be to

- Drop malicious packets and send alarm
- Block traffic from some IP addresses
- Correct fragmentation in packet streams
- Raise alerts and trigger human intervention by incident response teams.

IDS/IPS should always combine *anomaly-based* detection with *signature-based* detection. Signature-based is fast, generates less false positives and does not need a learning phase. In contrast, for anomaly-based detection, the IPS needs to learn how normal traffic looks like in order to be able to recognise deviation. A strong advantage is that this type of detection can potentially detect unknown attacks

Next-generation firewalls (NGF)

Encrypted traffic is a problem for IPSs, as everything that is inside the encrypted part cannot be scanned. NGFs promise an integrated security approach. In order to scan encrypted traffic, they act as a proxy for **all** traffic, even encrypted traffic. This means, they need to decrypt traffic such as TLS and encrypt it again. In principle, an NGF could be a powerful security tools. it should be able to look at applications, logical segments, roles, services, users, etc.

However, there are various potential problems with this approach.

- Policy rules get too complex and are difficult to understand and to maintain.
- A proxy for TLS etc. breaks end-to-end security and introduces a new weakness and attack point
- Encapsulated/nested encryption still possible
- Privacy issues. NGFs can support surveillance
- Single point of attack with full access to decrypted data
- NGFs might also provide a false sense of security, as they are unable to detect new (disguised) malware.