



**SLEZSKÁ
UNIVERZITA**

FILOZOFICKO-
PŘÍRODOVĚDECKÁ
FAKULTA V OPAVĚ

Šárka Vavrečková

Study materials for

Computer Networks and Decentralized Systems

Institute of Computer Science
Faculty of Philosophy and Science in Opava
Silesian University in Opava

Opava

Last updated: December 5, 2022

Annotation: This study material is intended for the students of the subject *Computer Networks and Decentralized Systems* for master study programs at the Silesian University in Opava.

In this subject we deal with the tools and procedures used in computer networks. We discuss topics of the local and wide networks, including wireless, management and optimization, security of networks.

Computer Networks and Decentralized Systems


RNDr. Šárka Vavrečková, Ph.D.

Institute of Computer Science
Faculty of Philosophy and Science in Opava
Silesian University in Opava
Bezručovo nám. 13, Opava

Typesetted in L^AT_EX

Preface






What we can find in this material

 *Preview:* This study material is intended for the students of computer science study programs. Here we can find advanced topics related to computer networks – network protocols, technologies of local and wide networks, network services including various decentralized and distributed systems, network security and management.

Some (additional) areas in text are marked with purple icons, they will not appear on the exam – their purpose is to motivate further study or experiments or to help in the future to obtain more information. If the purple icon is in front of the chapter (section) name, it applies to everything in the chapter or section.

Marking

We use these color icons:

-  New *terms*, marking etc. are marked with the blue symbol we can see here on the left.
-  The particular *procedures* and tools, methods of troubleshooting or solving more or less common situations, etc. are marked with the blue color as well, but different icon.
-  Some parts of the text are marked with a purple icon, which means that these sections are *optional* (students can request them or study them themselves). Their purpose is to voluntarily expand students' knowledge of advanced topics, which usually do not have much time left for teaching.
-  The yellow icon is intended for links to *additional information*.
-  The red icon means *notes*.

If the amount of text belonging to a certain icon is larger, the whole block is bounded by an environment with icons at the beginning and end, for example to define a new term:



Definition

The such an environment defines a new term or explains a well-known but complex term, or a term with multiple meanings or properties.



The environment for a longer procedure or a longer note or multiple links to additional information may look similar. Other environments can also be used:



Example

This is what an example environment looks like, usually a procedure. Examples are usually commented on to make the procedure clear.



Task

Questions and tasks, suggestions for trying out, which is recommended to be done while practicing the curriculum, are closed in this environment with a test-tube. If there are multiple tasks in the environment, they are numbered.



Contents

Preface	iii
1 Basic Concepts and Standards in Computer Networks	1
1.1 Overview	1
1.2 Active Network Equipment	3
1.3 What and How We Transfer	4
1.3.1 Protocols	4
1.3.2 Properties of Protocols	4
1.4 Standards and Standardization Institutions	5
1.4.1 Standardization Institutions for Computer Networks	6
1.4.2 Freely Available Standards	9
1.5 The ISO/OSI Reference Model	11
1.5.1 Overview	11
1.5.2 Cooperation of Protocols	13
1.5.3 Protocol Stacks	16
1.6 The TCP/IP Network Model	16
2 Local Networks – Ethernet	19
2.1 What Ethernet Is	19
2.2 Data-link Layer	21
2.2.1 EtherType	21
2.2.2 L2 Addressing	22
2.2.3 L2 Sublayers	24
2.2.4 Ethernet Frame Format	24
2.2.5 MAC Address Table on Switch	29
2.3 Physical Layer	31
2.3.1 Parameters	31
2.3.2 Implementation	33
2.3.3 Power over Ethernet	35
2.4 The Course of Transmission	36
2.4.1 Half Duplex	36
2.4.2 Full Duplex	38


2.5	Switches and Loops	39
2.5.1	The Spanning Tree Protocol	39
2.5.2	Convergence	42
2.5.3	STP Protocol Variants	44
2.5.4	Configuration	44
2.6	EtherChannel and Port Aggregation	48
2.6.1	Properties	48
2.6.2	Flow Control	49
2.6.3	Creating EtherChannel	49
3	Wide Area Networks and Access Networks	53
3.1	WAN Networks	53
3.1.1	Typical Structure of WAN networks	53
3.1.2	L2 Protocols for WANs	55
3.2	Selection of WAN Networks	57
3.2.1	X.25 – Packet Switching	57
3.2.2	Frame Relay – Frame Switching	57
3.2.3	ATM – Cell Switching	59
3.2.4	MPLS – Label Switching	60
3.3	Optical Networks Infrastructure	65
3.3.1	SONET/SDH	65
3.3.2	WDM	66
3.4	Telecommunications Network	68
3.5	WAN Access Networks	69
3.5.1	PPP Protocol	69
3.5.2	Access Telecommunication Networks	71
3.5.3	DOCSIS	75
3.5.4	FFTx	76
4	Routing	79
4.1	Bridging, Switching, Routing	79
4.2	How Routing Works	81
4.3	Cooperation of Routing Protocols	83
4.3.1	Autonomous System	83
4.3.2	Administrative Distance	84
4.4	Routing Algorithms	85
4.4.1	Distance-Vector Algorithm	85
4.4.2	Link-State Algorithm	86
4.5	Interior Routing Algorithms	88
4.5.1	RIP	88
4.5.2	IGRP and EIGRP	89
4.5.3	OSPF	91
4.6	Exterior Routing	93
4.7	Software-Defined Router	94

5	Decentralized and Distributed Systems	96
5.1	Types of Systems	96
5.2	Domain Name Service	98
5.2.1	Domains and Names	98
5.2.2	Zones and DNS Servers	100
5.2.3	Evaluation of DNS Requests	101
5.2.4	Host Table	103
5.2.5	DNS Protocol and DNS Packet	107
5.2.6	Security in DNS	108
5.2.7	DNS Helps Secure other Services	109
5.2.8	WHOIS Database	110
5.3	QoS	111
5.4	VoIP and Videotelephony	113
5.4.1	Principle	113
5.4.2	Protocols	113
5.4.3	Private Branch Exchange	116
5.4.4	Videotelephony and Videoconferencies	117
6	Network Management and Monitoring	120
6.1	Network Management	120
6.1.1	NMS	120
6.1.2	ISO/OSI Network Management Model	121
6.1.3	ISO/OSI Network Management	122
6.2	TCP/IP Network Management	122
6.2.1	SNMP	122
6.2.2	MIB-II	123
6.2.3	SNMP Usage	127
6.2.4	RMON	128
6.3	Syslog	129
6.4	Network Monitoring	132
6.4.1	Snort	132
6.4.2	NetFlow	134
6.5	VPN	135
6.5.1	Principles	135
6.5.2	IPSec VPN	137
6.5.3	GRE Tunnels	139
6.5.4	L2TP Tunnels	140
6.5.5	OpenVPN	140
6.5.6	SSH	141
6.5.7	MPLS VPN	142
6.5.8	Other VPN Solutions	143
7	Network Security	144
7.1	Terms in Network Security	144


7.2	Networks Attacks	145
7.2.1	Reconnaissance Attacks	145
7.2.2	Access Attacks	145
7.2.3	DoS and DDoS Attacks	147
7.2.4	Human Factor	148
7.3	Security Appliances	148
7.3.1	Network Analyzer	148
7.3.2	Traffic Monitoring	149
7.3.3	Firewall	151
7.3.4	Filtering	153
7.3.5	IDS/IPS and Deep Packet Inspection	155
7.4	Complex Management and Monitoring Systems	156
7.4.1	Network Management Software	156
7.4.2	SIEM	159
	Recommended Resources	162


Chapter 1

Basic Concepts and Standards in Computer Networks


 *Preview:* In this chapter, we will clarify or recall the concepts that we just should know from the bachelor's degree. The discussed topics are the both theoretically and practically focused. We will look how communication works in the ISO/OSI model and the TCP/IP network model.

We will use the term “computer network”, although it is not exact; actually there are a lot of types of devices used in the present networks, not only computers.


 *Keywords:* Network device, computer network, terminal equipment, active and passive network equipment, repeater, hub, bridge, switch, router, gateway, protocol, standard, RFC document, protocol data unit (PDU), ISO/OSI layers, SDU, SAP, entity, communication primitive, socket, protocol suite, protocol stack.

 *Goal:* The aim of this chapter is to remember the basic concepts and procedures in computer network, including the most important protocols, network devices, reference and network models. The goal is to understand principles of the network standardization and to be familiar with publicly available documents describing standards.

1.1 Overview


 In a computer network, we have some devices located at the start and end of transmission paths – the *terminal equipment*: computers, laptops, servers, tablets, smartphones, network printers, IoT devices (Internet of things), etc. The other devices the communication passes through also belong to computer networks:

- *active network equipment* – actively influence communication, they provide routing, signal amplification, etc.,
- *passive equipment* – only passively transfer data, e.g. cables.


 **Definition (Computer network)**

A computer network is a sum of devices – nodes – interconnected by *transmission paths*. A node is either a terminal equipment or an active network equipment.




 You should know these terms from previous studies:

- types of networks: PAN, LAN, MAN, WAN,
- transmission signal, wavelength, frequency, amplitude, bandwidth,
- baseband, broadband, modulation, QAM, multiplex, FDM, TDM, TDMA, OFDM, OFDMA, CDM, statistical multiplex, WDM,
- link, channel, circuit, leased circuit, physical circuit, virtual circuit, PVC (permanent virtual circuit), SVC (switched virtual circuit),
- physical and logical topology: bus, ring, star, tree, mesh, full mesh, point-to-point, backbone,
- network interface, VLAN, STP,
- types of cables: twisted pair, fiber-optic, coaxial, twin-ax; their properties, usage,
- for the twisted pair: unshielded (UTP), various types of shielded cables,
- types of transmission: stream, block (packet),
- connection: connectionless, connection-oriented transmission,
- switching methods: circuit switching, packet switching, datagram service,
- reliability: reliable or unreliable transmission, best effort service,
- types of communication: unicast, multicast, anycast, broadcast,
- methods of communication: simplex, full duplex, half duplex,
- the reference model ISO/OSI, the network model TCP/IP,
- ISP – Internet Service Provider, or NSP – Network Service Provider,

 We also assume that the reader can handle the following activities:

- how to work with numbers in the binary and hexadecimal system,
- how to find out the own MAC and IP address in the graphical and text command interface,
- how to set the IP address, mask, gateway, addresses of DNS servers,
- subnetting for IPv4,
- how to use Wireshark (<https://www.wireshark.org/download.html>),
- how to crimp a twisted pair cable (the RJ-45, 8p8c) including testing,
- how to use either Packet Tracer or some similar program for network simulation,
- how to check availability of a server or any other network device (ping, etc.),
- how to detect processes listening or communicating on some port, how to list various network statistics,
- how to find out wireless information, including availability of channels.

 **Task**

If you are not familiar with any of these terms and procedures, contact your teacher (immediately, not until the middle of the semester), we will solve this problem.



1.2 Active Network Equipment

The following is an overview of common active network equipment. Each is also accompanied by information about which ISO/OSI layer it belongs to.



Repeater is simple active network device with two ports. The incoming signal is amplified (or regenerate) and sent on. Repeaters work at the physical layer of ISO/OSI (L1).

Nowadays, we encounter repeaters in wireless local area networks, but also, for example, HDMI, where they serve to increase the signal range.



Hub is simple active network device usually with more than two ports. Anything comes to a port, is only resent to all other ports (the signal is amplified or regenerated, it is similar to repeaters). Hubs work with signal only, at the physical layer of ISO/OSI (L1).

The advantage of hubs is high speed of signal processing, the disadvantage is that is unnecessarily floods the network (data is sent to all neighbors except for the sender). We usually no longer do not meet hubs in common networks, they are used e.g. in some PAN networks or in USB communication.



Switch is a bit smarter active network equipment that keeps a table of device addresses (neighbors and other members of a network). It detects the recipient of an incoming packet (in the given network, usually not the final recipient), determines the port leading to the recipient according to the address table, and send the packet only to the found port. If the recipient's address is not present in the table or if it is a broadcast packet, it sends to all ports excluding the sender's port (flooding). The switch divides the network into segments – if nodes from the same segment communicate with each other, the such communication does not penetrate into another segment.

Switches work at the data-link layer (L2), but they can also include functionality of higher layers (= multilayer switch).

The advantage is that this device does not flood the network as much as hubs, and it is possible to implement basic management and security functions. The disadvantage is more computationally intensive operation (this is taken into account, a large part of the functions are implemented in hardware). We commonly meet switches in LAN, MAN and WAN networks.



Bridge is a device that separates two network segments from each other, like a switch. Unlike a switch, it usually has fewer ports, its functionality is implemented in software and it is standardized (IEEE 802.1). In practice, we do not encounter bridges. Bridges work at the data-link layer (L2).




Router maintains a routing table – different from a switching table of switches; a routing table contains L3 addresses (e.g. IPv4 or IPv6 addresses) of networks and subnets, not the terminal equipment. The incoming packet is “dissected” a little more thoroughly than by a switch, finding out which network the recipient belongs to based on the destination address, verifying the outgoing port leading to the recipient's network. Broadcasts are dropped.

Whilst switches know only the own network and no other networks, routers don't know the interior of the own networks, they know only paths to networks as a whole. Switches separate various segments of one network, routers separate various networks and work at the network layer (L3).

The advantage is possibility of implementing advanced management and security functions and the ability to separate communication in different networks. The disadvantage is even more computationally intensive operation.

tionally intensive operation than switches, so typically routers have fewer ports (therefore less traffic), a more powerful processor, and more memory to handle higher computational load.


 *Gateway* is intended for interconnection of two different types of networks, it serves as a connection point and “translator”. For example: if we have a VDSL router at home, then this device has a built-in gateway for communication between our local network (Wi-fi and Ethernet) and the ISP’s VDSL network.

Gateways usually work at some higher layer than protocols to interconnect.


1.3 What and How We Transfer

1.3.1 Protocols

A *protocol* determines how the communication should start and end, or how to agree on certain parameters of the communication (negotiation), how to communicate a certain type of data, how the other node should acknowledge signal reception or indicate that the transmission needs to be repeated, etc.

 **Definition (Protocol and its implementation)**


A protocol is a convention according to which a certain type of (mostly electronic) communication takes place. Defines rules determining the syntax (how which signals are composed), semantics (meaning) and synchronization of communication.

The protocol is only a prescription that needs to be implemented (programmed so that it can be used in practice). The implementation can be software, hardware (in circuits) or a combination of software and hardware. 

For example the HTTP protocol is implemented (programmed) in the kernel of the operating system of a user, and in the kernel of the operating system of a WWW server for the given communication.

 **Remark:**

There is one important rule for designing a protocol (in fact, it is maintained elsewhere, for example on a UNIX system): the protocol should be simple, short, and clearly implementable. It should not be too complicated, because the greater the complexity, the greater the probability of errors. Therefore, no protocol is universal – it can do one particular thing, and it can do it well. In English it is expressed by the abbreviation KISS (Keep it Simple, Stupid).

In order for this principle to be observed, there are certain conventions for protocol cooperation: what a protocol cannot do is passed on to another protocol for processing. 


1.3.2 Properties of Protocols

Common protocols have usually the following important properties:

- they are well known and nearly everybody can implement them (except of proprietary ones),
- they are rather simple, not very complex,
- they can cooperate with other protocols.

Why is the first property important? Imagine a situation where a network hardware manufacturer comes up with a new device and decides that the device will communicate according to a new protocol, the specification of which it will not provide to anyone else. However, this device only understands devices that support the same protocol, so it cannot communicate with devices from other manufacturers. It might be advantageous for the manufacturer in question if it could persuade its potential customers to buy only from him, but the reality is different – customers would rather buy equipment that they could easily integrate into their network, in which they already probably have some devices from other manufacturers.

Therefore, a lot of protocols are either freely available (the specification is completely free to view, anyone can implement) or at least available in another way (for a fee, under a license).

 An *open protocol* is a protocol with freely available specification, on the contrary, a *proprietary protocol* is a protocol whose specification is not published anywhere and its creator either keeps it to himself or licenses it for a fee.



Remark:

The prevalence of free specifications is evidenced, for example, by the fact that currently the most common routing protocol on routers is the open OSPF protocol. In contrast, IGRP (Cisco's proprietary routing protocol) was hardly used on devices from Cisco itself, and the specification of its successor EIGRP was released by Cisco for other manufacturers (but the other manufacturers remain with OSPF, and Cisco implemented OSPF as well).

If a manufacturer supports some own proprietary protocol for a given function, most likely some open protocol for the same function is supported too, so the device could “have a talk” with devices from other manufacturers.



1.4 Standards and Standardization Institutions



Definition (Standard)

Standard (in the field of information technology) is a requirement to meet certain specific characteristics for a certain type of hardware, software, etc. It is a document describing requirements, specifications, procedures, and descriptions for a given product, process, or service. We distinguish standards

- *normative (de iure)* – their observance is required, usually determined by the relevant state institution or an international organization with bindings to state institutions,
- *recommendation (de facto)* – their observance is not required, but it is highly recommended and it is in the interest of manufactures to engage in wider infrastructure.

In the field of information technology we often simply say a “standard”.


Standardization is the proces of unifying the features and function of a given product or service by determining a standard.




1.4.1 Standardization Institutions for Computer Networks

The following is an overview of the most important standardization organizations and institutions that issue standards related to computer networks, and technologies in general. Most of them are de facto standards (they are not obligatory, but they are usually observed).


The situation is slightly complicated by the fact that some standards issued by one organization are often adapted by one or even more other organizations (for example, the standard adaptation of standards by the Czech standardization office is common).

 **Czech office for standards, metrology and testing** is the Czech national standardization authority, it deals with the creation of Czech standards. The standards issued by this institute can be recognized by the letters ČSN, followed by the number of the standard. At present, it is largely involved in the adaptation of standards issued by the European Union. We can recognize these standards by ČSN followed by the abbreviation of the original organization or institution, for example ČSN EN ... are standards adopted from the European Union.

 **Additional information:**


The standards issued by the Czech office for standards, metrology and testing can be previewed at <http://www.unmz.cz/urad/csn-online> (only previews, the full texts are available for a fee). The English pages are <https://www.unmz.cz/en/home/>.



 **ETSI** (European Telecommunications Standard Institute) is although originally a European organization, actually its members are from all inhabited continents (states, major manufacturers of communication equipment, network service providers, research organizations, etc.). Proposals for new standards or changes to existing ones come from three sources – either agreed by at least four ETSI members, or an initiative is received from the European Commission (EC) or the European Free Trade Association (EFTA).

There are several types of ETSI standards: for example, EN (European Standard), ES (ETSI Standard), TS (ETSI Technical Specification) and others. The best known standards are related to mobile networks (especially GSM, 3G, 4G networks), smart networks, machine-to-machine communication, ICT in healthcare, etc. Most of ETSI standards are freely available.

 <https://portal.etsi.org>

 **ITU** (International Telecommunications Union) is a global organization for standardization of information and communication technologies. It is a part of the UN (United Nations) and is based in Switzerland. It has three parts:

- ITU-R: for radiocommunications (originally CCIR),
- ITU-T: standardization of telecommunications (originally CCITT),
- ITU-D: development of telecommunications.

ITU-T is part of the ITU for the standardization of telecommunications, its activities are also related to computer networks. Voting members are the countries concerned, and any organization can be a non-voting member. Membership is paid.

ITU-T is further divided into *Study Groups* (SG), each with its own focus. For example, SG13 has recently released several standards on cloud computing, SG16 deals with multimedia (video transmis-

sion, images, sound, etc.), SG17 security, SG20 smart networks (Internet of Things, . . .). The standards created by ITU-T are open, freely available.


We meet the ITU-T standards quite commonly: for example, the H.323 protocol used in multimedia transmissions and Internet telephony is one of them, or G.992.1 and G.992.2 protocols for ADSL access networks, X.509 standard used in encryption, and others.

 <http://www.itu.int/en/ITU-T/Pages/default.aspx>

ITU-R is part of the ITU for radiocommunications, and therefore deals with standards for wireless transmissions. For example, standards for LTE Advanced (formally IMT Advanced), such as M.2012 – *Detailed specifications of the terrestrial radio interfaces of International Mobile Telecommunications Advanced (IMT-Advanced)*, are currently much discussed.

The standards beginning with the letter “M” are related to mobile radiocommunication networks, “RS” for wireless networks of sensors and their remote handling, “SM” for frequency spectrum management, etc.

 <http://www.itu.int/pub/R-REC>


 **ISO** (International Organization for Standardization) is an independent organization issuing standards for communication technologies. Its members are standardization institutions from various countries (each country has a representation there), while ISO is a member of the ITU. ISO is based in Switzerland.

ISO has a hierarchical structure. It is divided into 200 TCs (Technical Committee), each technical committee has its own scope. For example, ISO/TC 47 deals with chemistry, ISO/TC 20 with aircraft and space vehicles, ISO/TC 272 forensic sciences, ISO/TC 299 robotics, ISO/IEC JTC 1 with information technology.

Each TC is divided into subcommittees (SC) and each subcommittee is divided into working groups (WG). Proposals for standards or their change always go from below, from working groups, gradually reworked and “fight through” up until the approval of the standard. The working group creates an *Working Paper*, which is transported into the *concept* of the Committee Draft, followed by the *draft International Standard*, the last step is the *international standard*.


The most “popular” standard by ISO is the well-known reference model ISO/OSI (the standard ISO 7498 and multiple related standards). ISO also deals with sensor networks, remote access, UPnP, web services and other topics.

 http://www.iso.org/iso/home/standards_development/list_of_iso_technical_committees.htm


 **IEEE** (Institute of Electrical and Electronics Engineers, the English pronunciation is similar to “Eye-triple-E”, so it is [aj tripl i:]) is the world’s largest organization of experts in the fields of electronics, electrical engineering, informatics and related fields, so it is a professional association. It does not only deal with standards, but organizes various meetings, conferences, educational events, publishes professional journals and develops other activities. The IEEE is based in the USA. One of the active sections of the IEEE is the Czech one.

The IEEE Association is primarily behind a group of IEEE 802 standards for local and, in part, large-scale networks, such as IEEE 802.11 for wireless local area networks (Wi-Fi) or IEEE 802.3 for Ethernet. Access to the full text of the standards is paid.

 <http://www.ieee.org>


 **IEC** (International Electrotechnical Commission) In the field of computer networks, iec mainly deals with standards for electrical and electronic equipment, cooperates mainly with the ISO organization and many standards bear the designation “ISO IEC”.

 <http://www.iec.ch/>

 **IETF** (The Internet Engineering Task Force) deals mainly with standards related to the Internet, including, for example, TCP/IP protocols. It works very closely with other organizations, such as The Internet Assigned Numbers Authority (IANA), the Internet Society (ISOC), the Internet Architecture Board (IAB), the World Wide Web Consortium (W3C), and others.

The standards issued by IETF usually begin with the abbreviation RFC (Request for Comments) followed by a number. If it is necessary to update some standard, the original text remains as the original RFC and a new document with a new RFC is created.


For example, for the above-mentioned open routing protocol OSPF, several documents (versions) were gradually created, of which they are currently up to date. RFC 2328 (OSPF version 2) and RFC 5340 (OSPF version 3).

 **Additional information:**

The official website of IETF is <https://www.ietf.org/>. All RFC documents are freely available, there are even a number of websites to access these documents:


- <https://tools.ietf.org/html/>
- <http://www.rfc-base.org/>
- <https://datatracker.ietf.org/doc/>
- <https://www.rfc-editor.org/>



 **ANSI** (American National Standards Institute) is the US standards agency. It brings together state institutions, commercial companies, members from academia field, etc. In addition to developing its own standards, it also represents the United States in ISO, IEC and other multinational organizations. The ANSI standards are available for a fee.

ANSI has created the ASCII character set standard, the standard for the C programming language (ANSI C) is also known, in the area of networks e.g. FDDI, CDMA, Frame Relay.

 <https://www.ansi.org/>

 **TIA a EIA** (Telecommunication Industries Association, Electronic Industries Alliance) are the US organizations that focus primarily on the physical level of communication among devices, and they also collaborate a lot with ANSI.

TIA deals with standards for various types of cables and connectors, antenna connections, mobile networks, ICT in healthcare, smart networks. EIA is, for example, behind the old well-known SCART connector, and in networks we also encounter the RS-232 connector. There are standards created by the cooperation of these companies, such as TIA/EIA 568 used for copper cables for Ethernet.

**Additional information:**

- <http://www.tiaonline.org/>
- https://www.eia.gov/about/eia_standards.cfm



1.4.2 Freely Available Standards

**RFC documents.**

RFC documents are also important for the functioning of (not only) the Internet, in which the most important protocols and the way of their cooperation are described. The basic features of these documents are:

- Each RFC document has its unique number and title.
- There are no updates of RFC documents, the updates of the content are issued with a new number (but usually the same title).
- One topic (e.g. protocol) can be described in more than one RFC document.

The advantage of the second feature is that we do not have to get lost in different versions of the same document, the disadvantage is that finding the current text is a bit more difficult. The advantage of the third feature is that RFC documents are usually not long enough to lose clarity, the disadvantage is that sometimes we have to look for related information.

**Example**

Let us look at some RFC documents related to the TCP protocol. This protocol works on the transport layer and its task is to establish and manage a connection between two communicating machines.

At <https://tools.ietf.org/html/> we enter the number of the RFC document 7414 into the search field. The given document does not describe operation of the TCP protocol, but it is a kind of a guidepost to other RFC documents that describe operation of the TCP protocol.

To avoid problems with incompatible formats, the RFC documents are in the plain text. Note the document structure:

- The document starts with the header and the title (“A Roadmap for Transmission...”), the abstract with a brief description follows, then the state of the document and the license agreement.
- The next is the document contents and chapters. The last parts are links to other resources (mostly other RFC documents) and contacts to authors.
- The document is divided into pages of equal length, every page has its own header a footer.

Let us focus on the information header of the document. On the left we find out that it is an IETF product and its RFC number is 7414, the next line is “**Obsoletes:** 4614”. This means that the previous version of this document (obsolete) has the number 4614. Note that the newer the higher the number. In the right column we can find out when this RFC document was created: in February 2015.

The introductory chapter of a document usually contains a discussion of the content of the document, and a summary. In the next chapter – Core Functionality – we find a link to the main document containing the description of the TCP protocol: RFC 793. Note the low number – this document has been valid for a long time (since 1981), and other RFCs add new functionality.

Although it is actually just an annotated list of RFC documents related to the TCP protocol, the document is really very long. However, we will not study the whole document.



**Example**

In the previous example, we learned that RFC 7414 is a newer variant that replaces the obsolete document RFC 4614. Let us look at this older document (in the same way – at <https://tools.ietf.org/html/> enter the number 4614 in the search field, or simply use the link from the newer document).

We can see that the document name (“A Roadmap for Transmission . . .”) is the same as the newer one, only the number is different. In the left column of the document header there is the line “**Obsoleted by:** 7414”. This line is important because if we come across an RFC document whose validity we know nothing about, this line tells us that it is outdated and which document is its successor.

The next line is “**Updated by:** 6247”. This is not a new version, only the “relationship to the environment” has changed, in this case some unused TCP extensions are moved to the history.

**Example**

Now let us look at the “Core” RFC document related to TCP. From the first example we know that its number is 793. The beginning of the document looks a little different; as can be seen, the structure of RFC documents has gradually been enriched.

Note the drawings – they are all purely textual, but it is sufficient for this purpose. In Chapter 1.1 of the given RFC document there is a rough drawing in relation to the surrounding layers (as we can see, under TCP there should be an Internet protocol, i.e. IP), and further in Chapter 2.5 on page 9 there is a more detailed drawing including specific cooperating protocols (of course not all). Chapter 3.1 contains a diagram of the TCP segment header, after which all parts are explained. On page 23 we find a state diagram describing communication using the TCP protocol.

**Task**

Explore RFC 2460 and answer the following questions:

- What does the document describe? When was it published?
- Does it replace any obsolete document?
- Find the chapter 2 with the terminology. Which terms do you know?
- Take a look at the drawings in the document.



ITU-T. Another organization that publishes its standards for free (not all, or for some of them not all versions) is ITU-T. Information is available at <http://www.itu.int/en/ITU-T/Pages/default.aspx>.

**Example**

One of the standards that emerged in ITU-T is X.500. It is dedicated to directory services, which are network services designed to simplify network management – an individual objects in the network (computers, servers, resources available on them, active elements, etc.) as well as users and their access rights to these objects are recorded in a distributed database. A lightweight variant of the X.500 protocol is the LDAP protocol, which is implemented both for Windows (such as Active Directory) and for Linux and other operating systems.

At <http://www.itu.int/itu-t/recommendations/rec.aspx?rec=X.500> we can find basic information about the protocol. The full version of the specification is available for free, but not the newest version.

At the bottom of the page is the version table. If we click on the link in the *Download* column, we will find out whether the document is freely available or only for members (TIES users).



Task


Take a look at the contents of the PDF file with the X.500 specification (it may be an older version), which was the case in the previous example, in particular:

- Compare the structure with RFC documents.
- At the beginning of Chapter 6 (Overview of the Directory), read what a Directory is. Note that this term is understood a little differently here than in common operating systems (similar to a folder).
- Notice the drawings. These are usually drawings of communication with the database or relational diagrams (trees). Figure 3 (printed page 7, in order 13) shows a sample X.500 directory tree structure, try to understand the relationships between the nodes of this tree.



1.5 The ISO/OSI Reference Model

1.5.1 Overview

 The *Protocol Data Unit* (PDU) is a sequence of data provided with metadata (information about data) related to a particular protocol. Protocols usually receive data, process it as needed (make structure, split, encrypt, compress, translate, specify the recipient's address, etc.) and add a *header* with the corresponding information (data length), encryption algorithm used, sender and recipient address, etc.). Some protocols also add a *trailer* containing data such as a checksum. Data transferred inside a protocol data unit is also called the *payload*.

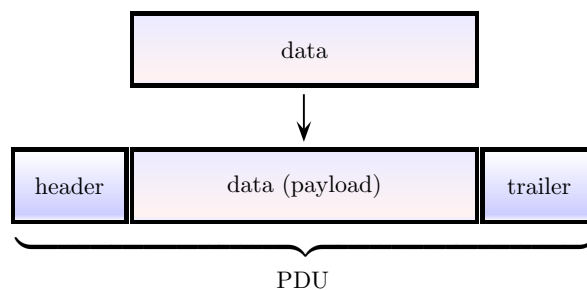



Figure 1.1: Protocol Data Unit (PDU)

 One of the most important standards in the field of computer networks is a group of standards describing the *OSI reference model* (Open Systems Interconnection) published by ISO, hence the name RM ISO/OSI. The original designation of the standard is ISO/IEC 7498-1, currently it is available as ITU-T X.200 on the web <http://www.itu.int/rec/T-REC-X.200-199407-I>.

OSI defines seven layers. Each layer performs a specific function in communication over a computer network and it is determined exactly what can happen on a given layer, so it is a *conceptual model* (ie it describes the design logic, the relationships between components).

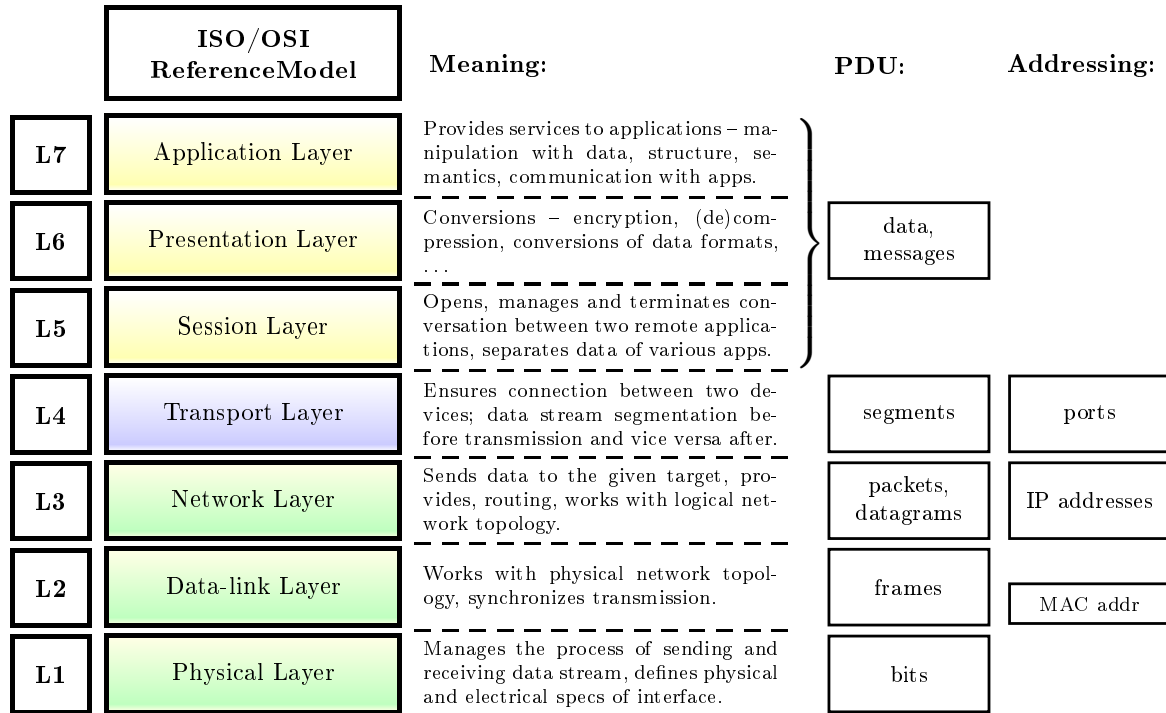


Figure 1.2: The ISO/OSI reference model

The order of the layers and a brief description can be found in the figure 1.2. The layers shown in green (L1 to L3) depend on transmission media, the layer L4 (blue) is transitional, the layers shown in yellow (L5 to L7) are independent of transmission media.


The data units of the network layer are called packets or datagrams. But these terms are more general:

- a *datagram* is a data unit sent with using connectionless service. This term can be used for some of PDUs in other layers as well, e.g. UDP datagrams, because UDP is one of connectionless protocols.
- *Packet* is a protocol data unit sent (not only) with using connection-oriented service, but it is often used in general for PDUs of various protocols. The most common usage is for the IP protocol.


The term *port* is used as “address” at the transport layer. But this term has multiple meanings, now we need two of them:


- physical port as a component of a network interface (e.g. an ethernet port),
- port as a number in the header of a PDU for the transport layer of ISO/OSI, this number determines the protocol or application of the upper layer the PDU is intended for. For example – the port 80 means communication with the http protocol and the related server application (web server).

1.5.2 Cooperation of Protocols

 The upper layer creates its PDU (for example, a packet) and sends it to the lower layer. For the lower layer, the received sequence of bits is referred to as *SDU* (Service Data Unit). Thus, the whole process is such that each layer receives an SDU from the upper layer and, by adding a header (and possibly also a trailer), creates a PDU from it and passes it to the lower layer. This also applies to the application layer – there is no layer above it, but the sequence of data that it receives from an application requesting its services is also called SDU.

We know that protocols should not be very complex, so they need to cooperate with other protocols. The cooperation can take place either within one layer or between adjacent layers, the lower layer provides services to the upper layer.

 *Entity* is an active element at a certain layer in the ISO/OSI model, which has an defined interface – a set of services that can be used by an entity from the upper layer.

 The interface between communicating entities (and therefore between layers) is called *SAP* (Service Access Point). Thus, SAP connects two entities in adjacent layers – the service user and the service provider.

As an entity we can imagine a (running) instance of a particular protocol or their set. One SAP can be used by only one user and one provider at a time, but one entity can use more than one SAP at a time.

The method of communication within one system (chain of entities and SAPs) is called *vertical communication* in ISO/OSI. *Horizontal communication* in ISO/OSI is communication between two

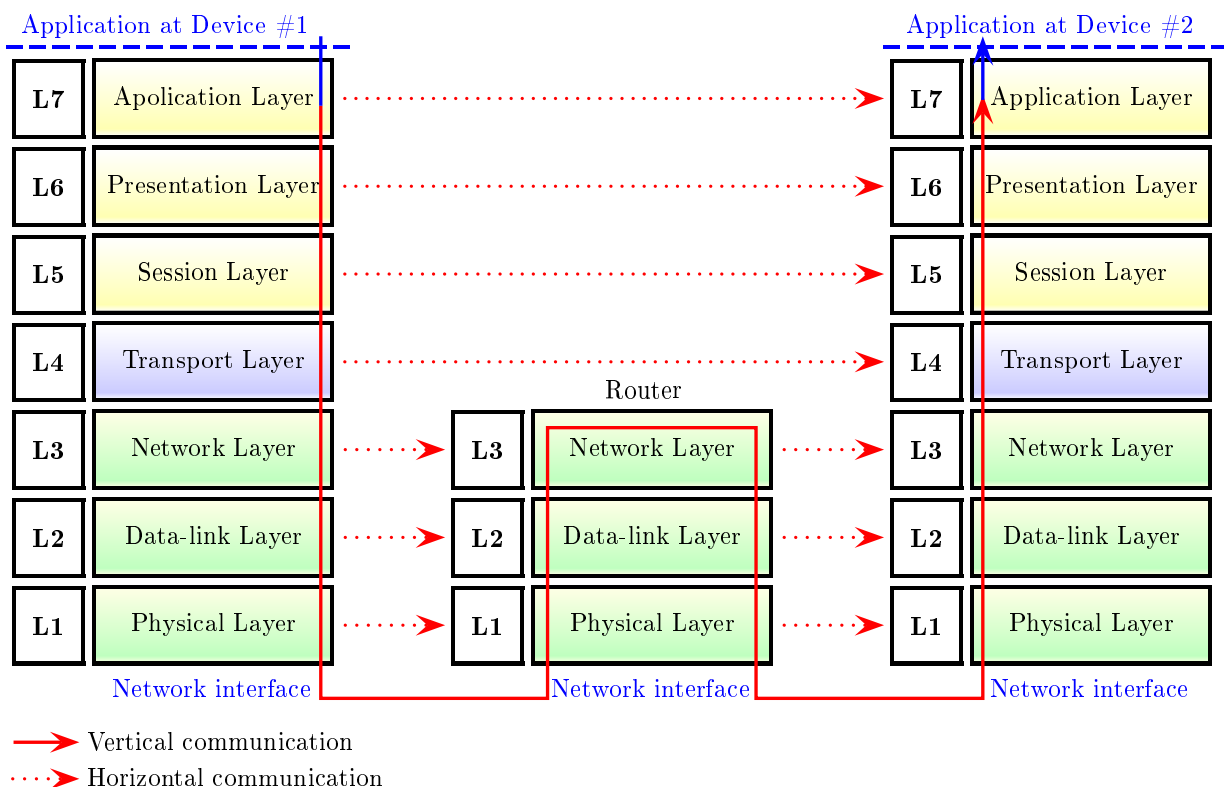


Figure 1.3: Horizontal and vertical communication in ISO/OSI

identical layers located on different machines, at the logical level. Both methods of communication are indicated in the figure 1.3. IDU consists of two parts:

- *Service Data Unit* (SDU) = PDU of some upper-layer protocol,
- *Interface Control Information* (ICI) = information necessary for the receiving entity to correctly receive and process the SDU.

ICI are metadata similarly as for a saved file we need its name, location, access permissions, . . .

 A protocol sending data (according to Figure 1.3 on the left)

- gets data through an SAP from the upper layer,
- if necessary, processes them in a specified way or divides them into smaller blocks,
- sets out the relevant meta-information such as addresses, data size, information on how the data is to be handled along the transport, etc.,
- adds header with meta-information and if need be also trailer, makes a PDU of the given layer,
- passes the PDU over a SAP to the lower layer.

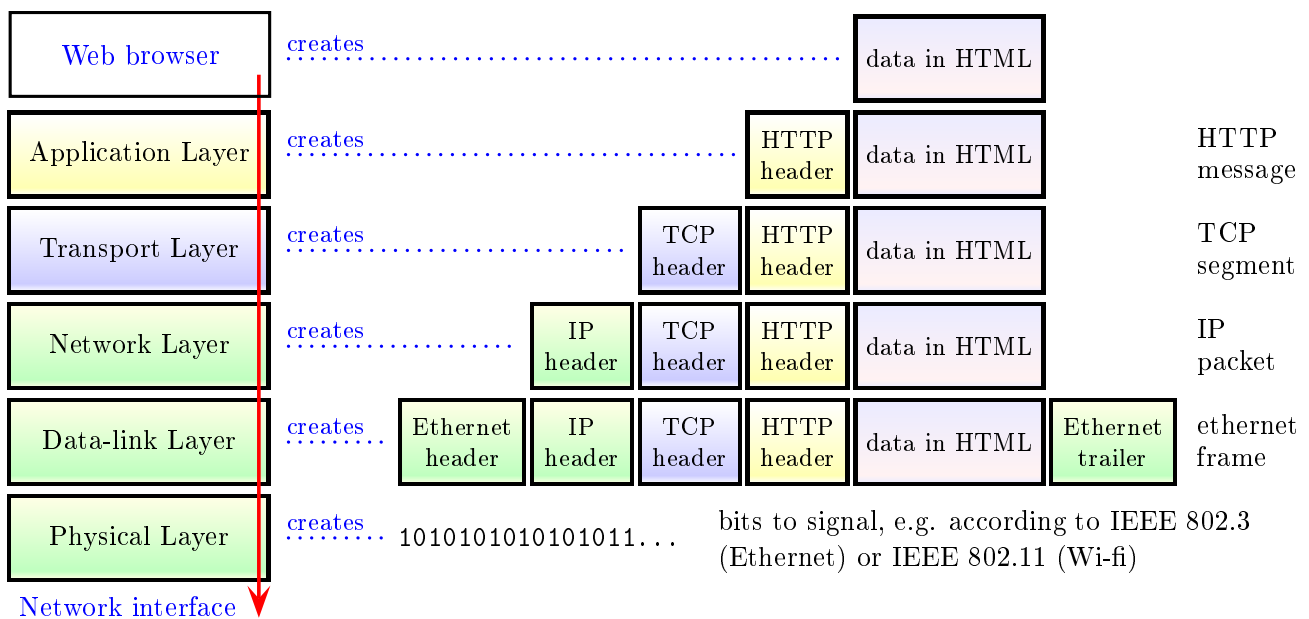




Figure 1.4: Encapsulation of PDU of the HTTP protocol

 Therefore, *encapsulation* is performed when sending, and *decapsulation* when receiving. Figure 1.4 shows the process of encapsulation. On the recipient's side, the opposite process takes place – decapsulation.

In fact, other protocols would be involved in this transmission, such as DNS address resolution or SSL/TLS for secure communication.

Note that some layers of the ISO/OSI model do not participate in this process, so exceptions to the rule are communicated only by immediately adjacent layers – application layer protocols can use SAP access points from layers L6, L5 and L4.

 How does it actually communicate via such an SAP, what is happening on this access point? For each SAP, communication *primitives* are defined. These primitives are simple functions, for example

Request (request for service of the lower layer, at the sending device), *Indication* (notification of the need for communication from the lower layer to the higher layer, at the receiving device), *Response* (response to Indication), *Confirm* (response to Request).

These primitives can have, like common functions, parameters – either SDUs passed over SAP, or they can be additional operational information that should not be part of the transmitted SDU/PDU.



Remark:

The inquisitive readers have thought that there must be a way, for example, for the network layer (L3) to “find out” to which address the packet is actually to be sent. This information cannot be transmitted in the PDU header, because the IP protocol does not get inside the higher layer headers (it is simply part of the data that needs to be sent for it), moreover, in some headers this information is not even present. Yes, it will learn from the parameters of primitives. See

<http://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/service-prim.html>. for more information.



Socket is a combination of a network address (used at the L3 layer) and a port number (used at the L4 layer). If we write this pair “manually” (for example in the address bar of a web browser), we place a colon between the two items. A domain address can be used instead of a network (ie numeric) address.



Example

The socket leading to the server `www.something.org` on port 8080 is represented by `www.something.org:8080`, i.e. we place a colon between the address and the port. For an IPv4 address, this will be similar, for example `193.84.214.5:8080`.

With an IPv6 address, this is more complicated because it contains colons itself, but the entry must be unique. Therefore, for example, when typing in the address field of a web browser, we place the IPv6 address in square brackets: `[2001:718:2201:208:5]:8080`.



Sockets can be found on layers L5–L7, they work as an interface between application protocols and the transport layer. For applications, they are available as Socket APIs (i.e. application programming interfaces) in the form of libraries containing functions for working with sockets (creating a socket, accepting communication on the other side of the connection, listening, reading, writing to the socket). A special type of socket is *stream socket*, which guarantees the delivery of multiple blocks of sent data in the correct order, and thus only cooperates with reliable connection-oriented protocols (e.g. TCP) at the transport layer.

Sockets can be found on both UNIX(-like) systems (including Linux and MacOS X) and Windows (WinSock API).




Task


Look at <https://www.binarytides.com/winsock-socket-programming-tutorial/>. This is a tutorial on programming sockets in Windows in the C/C++ language using WinSock. Find out which library is used to work with sockets, how the socket is created, how it connects to the server, how data is sent and received, and how the socket is closed.




1.5.3 Protocol Stacks


 The *Protocol Suite* is a definition of a group of protocols that work together. The *Protocol Stack* is an implementation of a protocol suite. These two terms are often used interchangeably.


It does not necessarily have to be a protocol specification for all layers of the ISO/OSI model, only some layers can be specified, and cooperation with another (supplementary) protocol stack is assumed.

 The *TCP/IP Protocol Stack* (also called the *Internet Protocol Suite*) is a set of interoperable protocols that we need on devices connected to a large network (typically the Internet). In addition to the protocols contained directly in the name (TCP, IP), it includes others, most of them at the application layer. Protocols on layers L3–L7 are determined directly, for lower layers only the interface is specified.

This protocol stack is formalized as the *TCP/IP network model*, which will be discussed in more detail in the following text (including protocols).


 *IPX/SPX* is a competing TCP/IP protocol stack from Novell designed for the Novell Netware operating system – IPX runs on L3 instead of IP, SPX on the L4 layer instead of TCP. It was optimized for smaller networks, while TCP/IP is also designed for large networks. At present, we do not meet it.

 *LAN Protocol Sets* are Ethernet (IEEE 802.3), Wi-fi (IEEE 802.11) Token Ring (IEEE 802.5, no longer used), and others. Usually they implement only the L1 and L2 layer, above which the TCP/IP protocol stack is usually pushed.

 *WAN Protocol Sets* are ATM, Frame Relay, MPLS, and others. They also usually connect to TCP/IP in various ways.

For example, ATM slides under the network layer (L3), but inserts a special adaptation layer between it and its L2 implementation. Frame Relay implements the L2 layer, it assumes some suitable physical interface on L1, mostly according to EIA/TIA standards.

In contrast, MPLS is inserted between L2 and L3, i.e. the MPLS packet encapsulates a packet from the L3 layer (usually an IP packet) and is encapsulated in an L2 layer frame (for example an Ethernet frame). It can also run over ATM or Frame Relay, so technology from older devices can be used. It can also run on PPP and other protocols for first-mile networks.

 *Protocol sets for mobile networks* are, for example, sets for LTE, GPRS, CDMA, UMTS and others. They usually implement L1 and L2 layers and assume some specific protocols at higher layers, but it depends on the type of device (a terminal device will need a different set of protocols than a base station or other specialized devices in such mobile network) .

1.6 The TCP/IP Network Model

The ISO/OSI reference model is very complex and too theoretical. Gradually, several simplified variants were created. The most well-known of them is the TCP/IP network model, which is also called the DoD model (USA Department of Defense Model). Its components are also standardized by the IETF and available in RFC documents.

The TCP/IP network model is actually a formal description of the TCP/IP network stack, its connection to cooperating stacks, and the general possibility of involving other protocols. It consists of four layers, the relationship to the RM OSI layers is indicated in Figure 1.5.

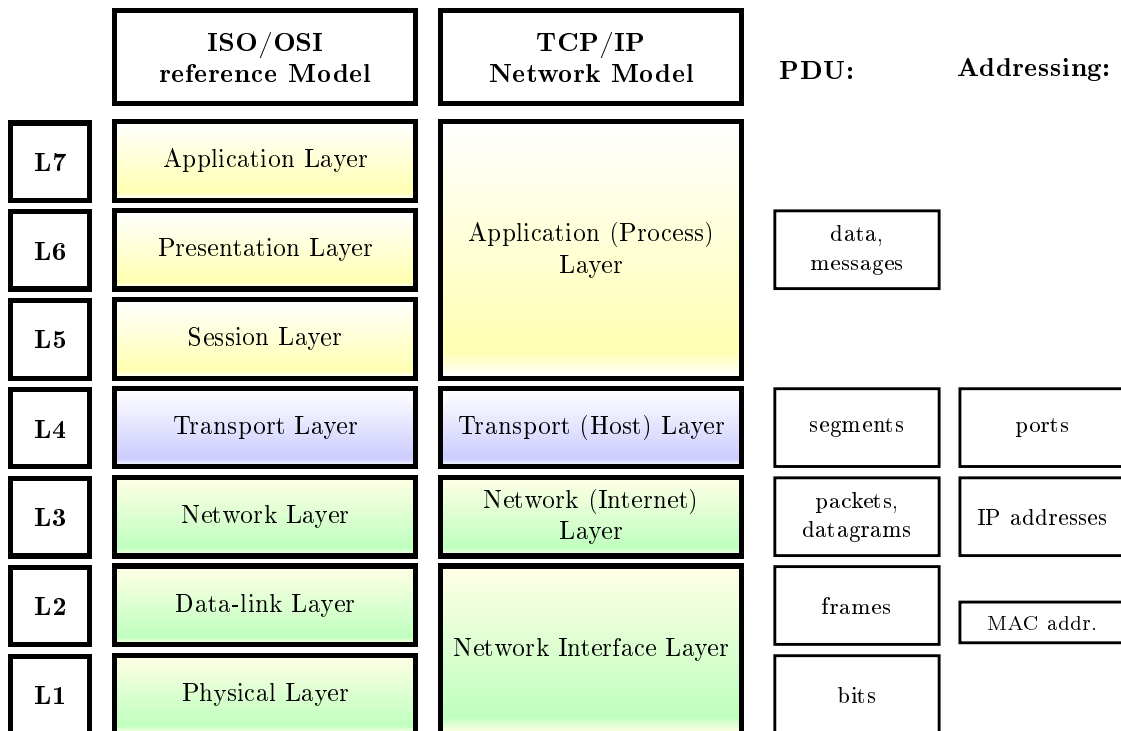



Figure 1.5: Comparison of RM ISO/OSI and TCP/IP


The TCP/IP model has been designed with the following features:

- as decentralized as possible (no central management),
- as resistant as possible to various (even critical) operating conditions and as resistant as possible to transmission errors,
- terminal devices perform maximum of work (the core of the network has to be fast and flexible), network is managed in distributed way (every part manages itself),
- can connect networks with very different network architecture and technologies (to be as universal as possible).


 **Network Interface Layer.** This layer combines the functionality of the L1 and L2 layers of the reference model. No protocols are specified for it directly in TCP/IP, it is only determined how they should communicate with the network layer. It communicates with the hardware (network interface), or part of it can be hardware implemented.

Usually, protocol sets of local and wide area networks are connected to this layer (in other words – no protocols are listed directly in TCP/IP for this layer), for example


- IEEE 802.3 (Ethernet),
- IEEE 802.11 (Wi-fi),
- protocol sets of WAN networks or their parts, xDSL, mobile networks.

 **Network Layer.** Also called “internet layer”, *internetworking* takes place at this layer, including *routing* between networks. The term “internet” (with a lowercase initial letter) generally refers to a network of networks, i.e. a network connecting not only individual nodes but smaller networks.

Network layer protocols are e.g. IP (Internet Protocol – the both versions IPv4 and IPv6), ICMP, IGMP and some routing protocols or a part of their functionality (OSPF, EIGRP, RIP, BGP and others).

 **Transport Layer.** We call it “host layer” as well because it is implemented only on hosts on the network. A *host* is the term for the terminal device (computer, server, tablet, smart TV, etc.) that “hosts” data, applications, services; each host has its own name (hostname).

The most known transport protocols are TCP and UDP, but there are also other transport protocols for “special” purposes.

 **Application Layer.** Also “process layer”, because these protocols communicate with processes. It combines everything that is in RM ISO/OSI on the top three layers (L5–L7). At this layer we find a large number of protocols, most of which are communicated by applications that need to access the network. Examples of application protocols: HTTP, SMTP, IMAP, POP3, DNS, DHCP, etc.



Remark:

Terminology from ISO/OSI is usually used (for example, designation of layers, entities, etc.), but the overall layout of network-related activities and the implementation of procedures are usually according to TCP/IP.




Definition (Internetworking)


Internetworking is a mechanism for interconnecting networks i.e. providing communication among (inter) networks. This mechanism takes place at the L3 layer, on devices such as a router or switch with L3 layer functionality.




Chapter 2

Local Networks – Ethernet


 *Preview:* In this chapter, we will expand our knowledge of Ethernet. The end of the chapter is devoted to the aggregation of Ethernet lines using EtherChannel technology. The reader is expected to have a basic knowledge of what Ethernet is and how communication takes place in it.


 *Keywords:* Ethernet, IEEE 802.3, CSMA/CD, Backoff Algorithm, frame, LAN, EtherType, LLC, MAC, LLC frame, SNAP frame, Ethernet II frame, MAC address table, OUI, L/G bit, I/G bit, MAU, PoE, Collision Window, IFG, Burst mode, EtherChannel, Port Aggregation, LACP, PAgP.

 *Goal:* The aim of this chapter is to understand in-depth communication in Ethernet networks, including related standards.

2.1 What Ethernet Is

The companies Xerox, Digital, and Intel collaborated on the original Ethernet specification, which was published in 1976 and is referred to as DIX Ethernet (according to the initials of the companies).


 Later, Ethernet was standardized as IEEE 802.3, but in this standard the name “Ethernet” does not appear at all and this standard is incompatible with the original DIX Ethernet. Gradually, different variants appeared – for different transmission media (metallic cables of different categories, fiber-optic cables) and the speed also increased.


 **Definition (IEEE 802.3)**

IEEE 802.3 is a standard describing a set of technologies for a local area network using cables (metallic or fiber-optic) where host devices share the same communication bandwidth and compete with each other.

The IEEE 802.3 standard describes the implementation for the physical and data-link layers (i.e. the entire TCP/IP network interface layer), including the method of communication and collision resolution.




 **Collision Method** determines how to decide whether or not a device can start transmitting. The CSMA/CD collision method is specified for Ethernet.

 **Definition (CSMA/CD)**

Ethernet uses the technology of multiple access to the transmission medium with carrier listening and collision detection – CSMA/CD:


- CS (Carrier Sense) – all nodes in the network constantly listen on the carrier to find out if another node transmits,
- MA (Multiple Access) – any node can transmit on the shared media if it finds out by listening that no other node transmits,
- CD (Collision Detection) – if a node does not catch to detect the transmission of another node before its own transmission (e.g. when the both nodes start transmitting at approximately the same time), it has to be able to detect the resulting signal collision and respond to it accordingly.

The use of the collision method is important when communicating in half-duplex (or when we use hubs, but that is quite history).

 What happens when a collision occurs:


- The transmitting node either wants to transmit and has detected that another node is transmitting, or it has detected a collision (it has detected that someone other is transmitting).
- If it has already transmitted, it stops transmitting (not immediately, it must allow the other transmitting node to detect the collision). It sends a *Jam signal*, which is a signal indicating a collision on the media.
- According to a special algorithm (Backoff algorithm), it determines the waiting time and after this time, it tries to transmit again.
- If the next attempt fails (either it detects that the link is not free before the transmission, or it detects a collision again), it returns to the previous point (waiting time and a new attempt).

The *Backoff algorithm* determines how long a node should wait to transmit when it detects that it is transmitting another node. The purpose is to ensure that in the case of multiple senders, each of them waits for a different time (determined by a randomly generated number), thus reducing the likelihood of another collision.

 **Procedure (Backoff Algorithm)**

There are several different variants of this algorithm, the basic (Exponential Backoff) is as follows:

- In the event of a collision, the node is sent the “Jam” signal, informing it of the collision and cancellation of the currently transmitted frame.
- It waits for $0 \dots 51,2 \mu\text{s}$ (randomly generated number from this interval), then it tries to transmit its frame again.
- If the transmission fails again, it waits for $2 \times 0 \dots 51,2 \mu\text{s}$ (randomly generated number) and then it retransmits its frame.
- In case of repeated fails, the node waits for $K \times 0 \dots 51,2 \mu\text{s}$ where K is a randomly generated number from the interval $0 \dots 2^c - 1$ where c is the current number of collisions (failed send attempts).


After 10 attempts, c no longer increases, then K is always from the interval $0 \dots 2^{10} - 1$. After 16 attempts, the process ends, the frame is considered undeliverable. 

As we can see, after the first two transmission failures, the waiting time is determined by generating one random number, but from the third attempt onwards we actually count with two random numbers, the upper limit for the first of them increasing exponentially with each step. This reduces the likelihood that two nodes will retry the transmission at the same time.

2.2 Data-link Layer

2.2.1 EtherType


Before we focus on the structure of the frame, let us talk about one of the identification codes that determine which protocol we are actually communicating with, that is, what exactly is to be encapsulated in the frame.

 *EtherType* is an identifier specifying the type of data encapsulated in the frame. It usually determines the protocol that created the payload to be sent by L3. It is represented by four hexadecimal digits, so the maximum value is $(FFFF)_{16}$ (the full form is $0 \times FFFF$), in the decimal system 65 535. In fact, there is the lower limit as well – for the Ether type we use the numbers with the minimal value 0×0600 , in decimal 1536. Thus, the EtherType is from the interval $0 \times 0600 \dots 0 \times FFFF$.

 **Remark:**

In the header of a frame created in L2 there are two bytes usually used for the EtherType number. But this field can be used for other type of data as well – length of the frame. However, in ICT, everything has to be completely unique, so it has to be possible to distinguish whether the field contains EtherType or frame length.


The length of the SDU encapsulated in the Ethernet frame is (almost) always less than $0 \times 05DC$ (1500 in decimal), we round it up and get 0×0600 .

If it is necessary to transfer a very large frame (above the specified limit), then it is *jumbo frame* and there is a special value for this purpose in the given field. 

There are a lot of EtherType values, some of them follow:

- 0×8870 : jumbo frames
- 0×8100 : VLAN frames according to 802.1Q
- 0×0800 : IPv4
- $0 \times 86DD$: IPv6
- 0×0806 : ARP
- 0×0835 : RARP
- 0×08137 , 0×08138 : IPX
- 0×8847 : MPLS unicast
- 0×8914 : FCoE Initialization Protocol
- $0 \times 814C$: SNMP

 **Remark:**

From the term “EtherType”, it might appear that this identifier is used only in Ethernet. Although it was the first to be used in Ethernet, we actually encounter it in other types of networks as well. 

**Additional information:**

<https://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xhtml>



2.2.2 L2 Addressing



As we know, MAC addresses (physical, hardware addresses, EUI-48) are used at the data-link layer. These addresses can be local (manually set) or BIA (Burned-in Address) specified by the manufacturer. Uniqueness of the BIA addresses is ensured as follows:

- The first half of the address is assigned by the IEEE (large manufacturers are assigned several addresses, smaller ones are enough for one), so no two manufacturers have the same. This number is called the *OUI* (Organizationally Unique Identifier).
- The second half is determined by the manufacturer himself, who makes sure that these numbers are unique within the assigned first half.

Table 2.1 shows the above described structure.

24 bits	+	24 bits	=	48 bits
OUI = identification of manufacturer IEEE assigns	+	identification of particular product manufacturer assigns	=	entire address globally unique

Table 2.1: MAC address of a network interface

**Example**

Here are examples of one MAC address with various types of notation:

50:E5:49:A2:80:61

50-E5-49-A2-80-61

50E5.49A2.8061

50E549A28061

**Task**

At <http://standards-oui.ieee.org/oui/oui.txt> find a list of OUIs assigned to various network device manufacturers. On this web, find the manufacturer of your network card(s).

An alternative is the search tool at <http://www.miniwebtool.com/mac-address-lookup/>.



Changing the MAC address is not so common, but sometimes it is useful – for example, when we need a MAC address in a specific form to use a certain technology or application and there is no time or possibility to change the configuration, or virtualized environment (although unique addresses may be used there as well). Hackers use altered MAC addresses when trying to break into a network protected by a blacklist or whitelist (list of denied/allowed addresses).

There are even two ways to change the MAC address in Linux:

- `ifconfig eth0 down hw ether 00:00:00:00:00:02 up`
- `ip link set dev eth0 address 02:00:00:00:11:22` and then `ip link set dev eth0 up`



Each locally valid address should have the *L/G bit* set. This bit is the second from the right in the first byte of the address, as shown in Figure 2.1.

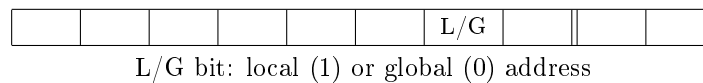



Figure 2.1: L/G (local/global) bit in a MAC address

Example (How to determine local address)

We will find out from the following MAC addresses whether they are BIA or local.

MAC address	first octet	... in binary	Result
00-11-21-30-A2-7C	00-...	0000 00 0 0-...	BIA address
98-E7-9A-26-4B-55	98-...	1001 10 0 0-...	BIA address
02-D3-2A-64-DD-20	02-...	0000 00 1 0-...	local address
86-D3-2A-64-DD-20	86-...	1000 01 1 0-...	local address



 If the address is intended to indicate a group of devices (but not necessarily all of them), it is called a *multicast* address. In Ethernet and some other networks using MAC addresses, we mark out the multicast address by the last bit in the first byte. For multicast address, it is set to 1, which means that the first octet will be an odd number.

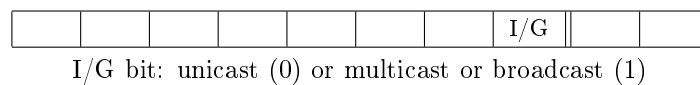


Figure 2.2: I/G (individual/group) bit in MAC address


The general broadcast MAC address FF-FF-FF-FF-FF-FF has these both bits set, it is locally unique and meaning a group of devices.

Example (Individual or group addresses)

The group address has the I/G bit set, so the first octet is an odd number. So:

- 86-D3-2A-64-DD-20 is individual, because the I/G bit is set to 0 (and also local, because the L/G bit is set to 1).
- 87-D3-2A-64-DD-20 is group, because the I/G bit is set to 1.



 An L2 device should have an overview of the *physical network topology*, the addresses of the devices with which it can communicate, and the network interface (which way) the device can reach. Each device keeps this information in its own *table of MAC addresses* (it can be called a MAC table, CAM table, etc., depending on what the manufacturer calls it).

Task

For the following addresses, determine whether they are BIA or local.


- 00-90-CF-23-50-37
- 2C-60-0C-73-22-A5
- 4E-09-B4-A0-BF-33
- 36-09-A4-26-41-0C
- E4-90-69-DD-82-E1
- FF-FF-FF-FF-FF-FF

Are the following addresses individual or group?

- 4E-09-B4-A0-BF-33
- 01-2D-63-BB-00-20
- 4F-00-21-7A-C0-59



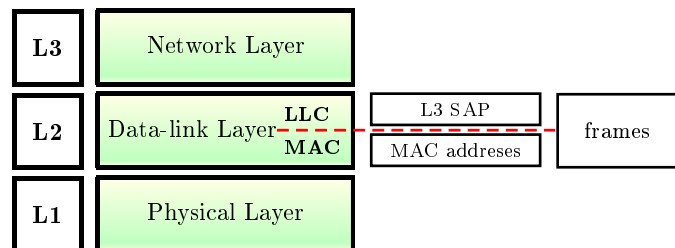
2.2.3 L2 Sublayers

 The data-link layer has two sublayers (and various protocols can implement one of them or the both):

- LLC (Logical Link Control) – upper sublayer cooperating with the L3 layer (network), adds info about SAP leading to L3 and other control information to the frame, provides multiplexing.
- MAC (Media Access Control) – lower sublayer cooperating with the L1 layer (physical), works with MAC addresses, adapts the frame structure for the specified bit rate and

spodní podvrstva, zajišťuje spolupráci s vrstvou L1 (fyzickou), přidává do rámce fyzickou adresu příjemce a odesílatele, inserts a synchronization sequence of bits at the beginning. The CSMA/CD mechanism is implemented here.

The figure on the right shows the relationship of the sublayers of the data-link layer to the surrounding layers. As mentioned above, in a protocol set, either both sublayers can be implemented by one protocol, or two protocols are used – one for the LLC sublayer and the other for the MAC sublayer.




Ethernet can be implemented at the L2 layer as follows (for L2 protocols):

1. The IEEE 802.2 protocol (directly called LLC) is used for the LLC sublayer, and data from the upper layer is encapsulated in the LLC frame. It is then encapsulated in a MAC frame according to the IEEE 802.3 protocol, creating a complete L2 layer frame.
2. Like the first possibility, only the SNAP extension of LLC is used instead of the original LLC protocol.
3. The data from the upper layer (L3) is encapsulated according to the Ethernet II protocol, which “manages” both the LLC and MAC sublayers.

The first two possibilities can be used, but currently we hardly encounter them with Ethernet. The IEEE 802.2 protocol is used to implement the LLC sublayer rather in wireless networks and also in older Token Ring or FDDI networks. SNAP is an improvement on the original LLC protocol, allowing to connect a wider set of protocols. The most common encounter is the third possibility, such frames are called *Ethernet II*.

2.2.4 Ethernet Frame Format

We will gradually go through all three possibilities. The format of the frame (and later the format of various other PDUs) will be represented in the form of a table, where the length of individual fields is indicated in the header.

 **LLC frame.** The structure of the LLC frame (according to the IEEE 802.2 protocol) is simple – a header with three fields is added to the SDU received from the parent layer.

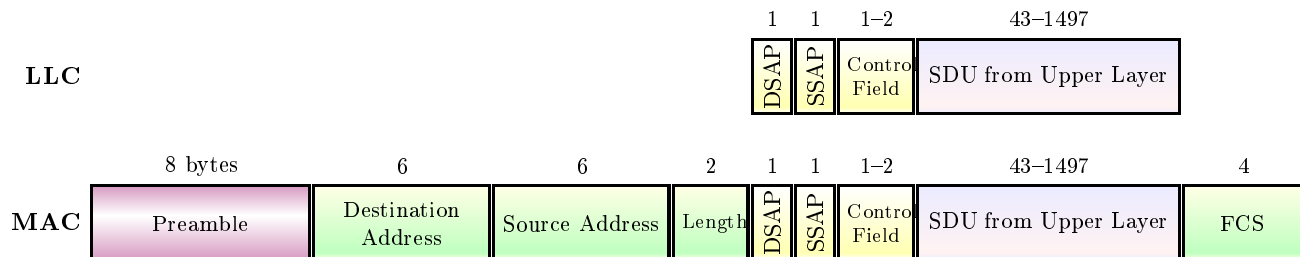


Figure 2.3: IEEE 802.3/802.2: LLC frame encapsulated in MAC frame according to IEEE 802.3

Figure 2.3 shows the format of the given frame type. The LLC header is yellow, the header and footer of the MAC frame is green and purple). The individual fields in the LLC frame (yellow) have the following meaning:

- *SSAP*, *DSAP* (1 octet each) – access points (SAP) on the destination (DSAP) and source (SSAP) devices, which occupy 7 bits from each octet; the remaining (least significant, right) bit in each octet indicates:
 - I/G bit (individual/group) in the destination octet determining if the given SAP is group or individual,
 - C/R bit (command/response) in the source octet for the packet type (command or response),
- *control field* (mostly 1 octet) – determines the type of frame in terms of service, ordinal number, etc.,
- SDU from the upper layer encapsulated in the LLC frame.

The values for the DSAP and SSAP fields are standardized, some of them follow:

- 0×06: DoD IP
- 0×42: IEEE 802.1 Bridge STP
- 0×98: Arpanet ARP
- 0×E0: Novell Netware (IPX)

Some other values are used, for example, for LLC management (service frames, they have nothing to do with the upper layer), others are reserved for some protocols that are no longer used.

Example

If an LLC frame header contains these numbers and other data:

0×42	0×42	0×03	data
------	------	------	------

This means that on both the source and destination devices, the L2 layer (its LLC sublayer) communicates with the Spanning Tree Protocol (STP). The number 0×42 is binary 1000010, so the I/G bit and C/R bit have the value 0 (individual SAP, command). Inside (like “data”) is the STP protocol PDU.

Additional information:

- <https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/tutorials/llc.pdf>
- <https://tools.ietf.org/html/rfc1700> (SAP numbers)




The MAC sublayer adds the following fields according to IEEE 802.3 (green and violet):

- *Preamble* (8 octets) identifies start of frame, it is a synchronization information. The first 7 octets of the preamble contain alternating ones and zeros, except for the last bit of the eighth octet – this is set to one instead of zero. The whole preamble therefore looks like this:

```
10101010 10101010 10101010 10101010 10101010 10101010 10101010 10101011
```

In the literature we also find another characteristic – a preamble on 7 octets with alternating ones and zeros, and then one octet, which violates this rule in its least significant bit. This octet is called SOF or SD (Start-of-frame Delimiter).

- *Destination Address* (DA) and *Source Address* (SA) are the MAC addresses of the communicating nodes in the network. The source address must always be unicast, the destination can be unicast, multicast or broadcast.
- *Length* of the encapsulated LLC frame (so the length of the data from the upper layer plus the LLC header).
- *FCS* (Frame Check Sequence) is a checksum that is calculated from address fields (DA, SA), length, LLC headers and higher layer data (i.e. the header but the preamble, which is always the same, data, and except FCS).

 **Frame Checksum.** For FCS, the CRC-32 algorithm is used in Ethernet, which can be implemented quite easily in hardware, in circuits. The calculation is based on operations in the Galois field GF (2^n), where the division of polynomials over binary numbers modulo 2 (i.e. the remainder after division by two) is used as an operation.

The algebraic field modulo two works in such a way that when the number 2 or higher is the result of the operation, it is replaced by the remainder after division by two. For example, $1 + 1$ is not two, it is zero. In addition, instead of numbers, we work with polynomials.

For example, a polynomial of degree 2 over binary numbers has the form $a_2 * x^2 + a_1 * x^1 + a_0 * x^0$, where the coefficients a_2, a_1, a_0 are binary digits (0 or 1), $x^0 = 1$, i.e. $a_2 * x^2 + a_1 * x + a_0$, for example $1 * x^2 + 0 * x + 1$, respectively $x^2 + 1$ is a polynomial over binary numbers of degree 2. We need a polynomial of degree 31, i.e. with powers of 31 to 0 (that is, 32 members of the polynomial, we need 32 binary digits for the coefficients).

Polynomials can be divided in the same way as numbers (but here in principle so that we still stay with binary coefficients), and if they are disjoint, we get the remainder after division (as with ordinary numbers when dividing the number 15 by two we get the remainder 1). The remainder after division is again a binary polynomial, and we can convert it to a sequence of bits by taking the coefficients (0 or 1) for the members of a polynomial with a certain power and concatenating them into a 32-bit binary number. So we have a loop: we convert the binary number to a polynomial, divide by the polynomial and convert the rest back to a binary number.

Checksum calculation procedure:

1. The bits sent from the destination address field till the data (i.e., everything between the preamble and FCS fields) are divided into a sequence of 32-bit words (numbers), hence CRC-32.
2. These 32-bit words are further understood as polynomials, where the individual bits (0 or 1) determine whether or not a particular member of the polynomial should be used (0 – no, 1 – yes).

For example from the sequence of the 32 binary digits 1000111010000...0001 we make the polynomial $x^{31} + x^{27} + x^{26} + x^{25} + x^{23} + x + 1$.

3. We divide a polynomial from the first 32 bits by a special polynomial of degree greater than 31, which we call “irreducible polynomial”, in a Galois field of binary polynomials that is similar to prime numbers that are greater than the maximum value for divisors (than the number we divide). Unlike arithmetic operations with ordinary numbers, we get here as a remainder after division by a binary polynomial of degree at most 31.
4. We take the polynomial from the next 32-bit word and add the polynomial that is the remainder after dividing the previous 32-bit word. We divide the sum again by an irreducible polynomial. Then we take the polynomial from the next 32-bit word, add the remainder of the previous division, divide . . . , etc. sequentially over all 32-bit words.
5. When we reach the end of the data and get the last remainder after division, we convert this remainder (polynomial) to a 32-bit binary word in exactly the opposite way to how we created polynomials – we take the individual coefficients of 0, 1 for polynomial members (in order from degree 31 to degree 0), concatenate and get a 32-bit number. We store this in the FCS field.

The Galois field $GF(2^n)$ has one peculiarity compared to ordinary numbers: if we take two polynomials and add them, and subtract the same two polynomials from each other, we get the same result in both cases. So addition and subtraction are the same operation here. This is used to check the checksum. So when a frame goes through a switch that actually works with a checksum (that is, it switches frames by store-and-forward method) or already arrives at the terminal device, and you need to determine if the frame has been corrupted along the way, follow these steps:

1. We start in exactly the same way as in the CRC32 calculation, i.e. we take bits from addresses including further through data in 32-bit words, gradually create binary polynomials, divide by an irreducible polynomial, add the rest to the polynomial from another 32 bits, divide. . .
2. Instead of stopping at the end of the data, we add the result of the last division to the contents of the FCS field (no longer divided). Because in the previous algorithm we got a binary number at the end of the data, which we stored in the FCS, now we should actually add two identical values. Because addition and subtraction are identical operations here, it is the same as subtracting two identical numbers from each other, i.e. we should get zero.
3. Evaluation: if the frame is OK, then the result from point 2 should be zero, but if the frame is damaged, something else will come out.

The advantage is that when checking if a frame is OK, we don’t have to load the frame first and then go back to the beginning to run the algorithm, we can count on receiving the frame from the network interface at the same time. We receive, store in a buffer and the special chip counts continuously. After receiving the last 32-bit word (which is FCS), just check the result. When the chip reaches a result of 0, the frame is OK and we accept it. If there is another result, we drop the frame.



Additional information:

<https://www.autohotkey.com/boards/viewtopic.php?f=7&t=35671>



SNAP frame. SubNetwork Access Protocol (SNAP) was created by extending the IEEE 802.2 LLC protocol, modifying and extending the LLC sublayer header. The SNAP frame looks like this:

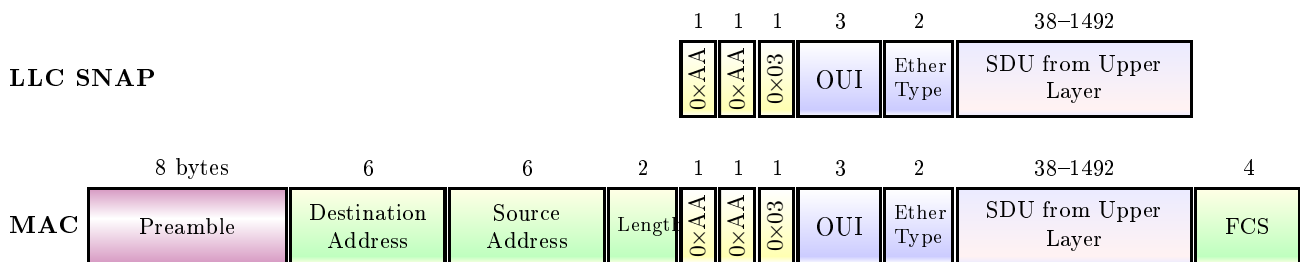


Figure 2.4: IEEE 802.3/SNAP: SNAP frame encapsulated in MAC frame according to IEEE 802.3

In Figure 2.4, the fields added by the SNAP extension are colored blue. As we can see, SNAP sets certain constant values in the LLC header – the code $0\times AA$ is used as DSAP and SSAP (or the code $0\times AB$ is also allowed) and the number 0×03 is saved into the control field.

The SNAP extension adds the following fields (blue in Figure 2.4):

- *OUI* (Organizationally Unique ID) is usually set to 0. If not, then this is the identifier of the manufacturer of the sending device. For example, Cisco has an $OUI = 0\times 00\ 00\ 0C$.
- *EtherType* (or more generally Type) specifies the protocol of the upper layer if $OUI = 0$. If not, then this field is purely directed by the organization whose OUI is in the previous field, for example for packet transmission according to proprietary network protocols.

In the previous pages there is the list of several commonly used values for the SSAP and DSAP fields in the LLC header, and the link to the complete list. The value $0\times AA$ is one of the identifiers for the SNAP extension.



Remark:

How do we know if an incoming frame is of the type IEEE 802.3/LLC or IEEE 802.3/SNAP? Several fields (MAC frame headers according to 802.3) are the same, the difference starts at the beginning of the nested LLC/SNAP frame. If in the “yellow part” (three octets after the MAC header) we find the value $0\times AAAA03$, then it is an SNAP frame. Otherwise, it is an LLC frame.



Ethernet II frame. Currently, this type of frame is used almost exclusively in Ethernet (for all data frames), it is the simplest and the most efficient.

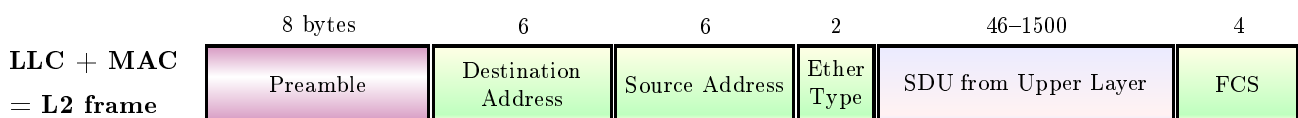


Figure 2.5: Ethernet II frame

In Figure 2.5 we can see that the structure of the frame is actually very similar to what the IEEE 802.3 protocol provided on the MAC sublayer in the previous cases. The preamble fields, both MAC addresses and checksum have the same meaning. The only difference is in the field in which the frame length was in the previous cases – within Ethernet II we find the value EtherType there.



Remark:

How do we know that this is an Ethernet II frame and not any of the previous ones?

One of the previous sections describes the values and meaning of the EtherType field. Among other things, we read in one note that EtherType and frame length are stored in the same field, with a number greater than 0×0600 meaning EtherType, less than meaning frame length. So, while loading the first $8 + 6 + 6 = 20$ octets, the switch still can't distinguish between frame types, but according to the 21st and 22nd octet can already distinguish whether it is Ethernet II.



Additional information:

- <http://www.infocellar.com/networks/ethernet/frame.htm>
- http://www.wildpackets.com/resources/compendium/ethernet/frame_formats



Task

There are many sites on the web that offer files with captured network traffic for download. These files are great for practice, we can see typical traffic on them using certain specific protocols. Some of these sites are better avoided, but others are worth trying, for example:

- <https://wiki.wireshark.org/SampleCaptures>
- <http://packetlife.net/captures/>
- <http://www.netresec.com/?page=PcapFiles> (recommend after gaining more extensive knowledge in the field of computer networks, especially in eavesdropping on malicious software)
- <https://www.nostarch.com/packet2.htm> (the link “Download the capture files for this book”) leads to the files with captured traffic)

Choose at least two of these addresses and go through the content of the websites.



2.2.5 MAC Address Table on Switch


Various manufactures use for the switching table with MAC addresses various names, including CAM (Content Addressable Memory) table. There are (among others) the following entries in this table:

- MAC address of an registered device,
- port, on which the device is available,
- other information (VLAN, timestamp, etc.).

Incoming frames are processed in this way:

- If the recipient of the frame is the device registered in the MAC address table, the switch finds a row of the table with the MAC address of the recipient, finds the port to which the recipient is connected on this row, and sends the frame to that port.
- If the recipient's address is unknown (not in the table), the frame is sent to all ports except the one from which it came. In this case, the switch behaves like a hub = flooding.
- If the destination address is broadcast, the frame is sent to all ports except the one from which it came.

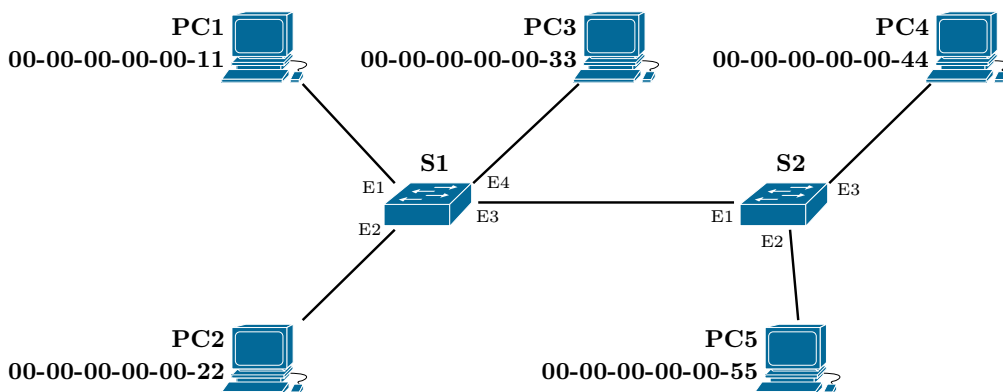
But how does the addressee actually get into that table?

 Suppose we have a new (or restarted) switch, the switching table is empty. In this case, there are two ways to populate the table. Either the relevant data is added into it by means of manual configuration (or by means of a script), or we simply leave it on the switch in question, so that it can look around in the network itself and “get to know the neighbors”.

Ethernet switches use a bit more sophisticated method. Whenever a frame is delivered, the switch is interested not only in the destination address (which is important to know where to send the frame), but also in the source address. For the source address, it is more or less clear on which port the (source) device is available – on the port from which the frame just came. Thus, if the switch does not have a source address in the table, it will add it with information about the port from which the frame came.

Example

Let’s look at the following figure. There are two switches on it, the first of which has four active ports, the second three. Assume that switch S1 has its table empty so far.



This traffic is on the network (everything in S1):

- A frame with the following addresses is received on the port E2:
 - DA = 00-00-00-00-00-33
 - SA = 00-00-00-00-00-22
 ⇒ the switch determines that the device with the MAC address 00-00-00-00-00-22 is on the port E2, the table:

MAC address	Port	...
00-00-00-00-00-22	E2	...

the destination (00-00-00-00-00-33) is unknown for S1, so the frame is sent to the ports E1, E3 and E4.

- A frame with the following addresses is received on the port E3:
 - DA = 00-00-00-00-00-11
 - SA = 00-00-00-00-00-55
 ⇒ the device 00-00-00-00-00-55 is on the port E3, table:

MAC address	Port	...
00-00-00-00-00-22	E2	...
00-00-00-00-00-55	E3	...

the destination (00-00-00-00-00-11) is unknown, so the frame is sent to the ports E1, E2 and E4.

- A frame with the following addresses is received on the port E1:

- DA = 00-00-00-00-00-22
- SA = 00-00-00-00-00-11

⇒ the device 00-00-00-00-00-11 is on the port E1, table:

MAC address	Port	...
00-00-00-00-00-22	E2	...
00-00-00-00-00-55	E3	...
00-00-00-00-00-11	E1	...

the destination (00-00-00-00-00-22) is present in the table (the first row), the frame is sent to the port E2.

- A frame with the following addresses is received on the port E1:

- DA = 00-00-00-00-00-44
- SA = 00-00-00-00-00-11

⇒ the device 00-00-00-00-00-11 is present in the table, no entry is added,

the destination (00-00-00-00-00-44) is unknown, so the frame is sent to the ports E2, E3 and E4.

So we would continue, the MAC table would be complete only if all the devices in the network were gradually “revealed”.




2.3 Physical Layer

The following operations are provided on the physical layer in Ethernet:

- cooperation with the L2 layer, taking the SDU from this layer (i.e. the frame), in the opposite direction passing the data to the L2 layer,
- coding, multiplexing,
- transmitting and receiving signal, synchronization,
- auto-negotiation – the network interface needs to negotiate with the network interface of the neighbor device all transmission parameters (speed, half or full duplex, etc.) in order to understand each other.

2.3.1 Parameters


 The vast majority of Ethernet types use baseband transmission. Other important parameters are the speed and type of transmission medium (cable), both of which affect the coding used and the multiplexing process. On the physical layer, the following appears in the marking:

- speed: 10, 100, 1000 etc. – in Mbps, or Gbps with the letter “G” for faster standards, e.g. 10G,
- “base” for the baseband or “broad” for the broadband transmission,
- cable type – mostly the letter “T” means the twisted pair, “F” fiber-optic.

So let's go by history, but let's skip the oldest standards. We will not deal with all of them, for each speed we will select only some typical representatives.

 **10Mb Ethernet.** The representatives allow the 10 Mbps speed:

- 10Base-T: baseband, unshielded twisted pair (UTP),
- 10Base-F: baseband, fiber-optic cable,
- 10Broad-36: broadband, coaxial cable with the impedance 75Ω ; it failed, but became the basis for later Ethernets operated in cable television networks (today, the DOCSIS standard is used for data transmission in cable television).

 **Fast Ethernet.** The speed 100 Mbps with baseband, possibility of auto-negotiation added. Representatives:


- 100Base-TX: UTP cable at least Cat.5, uses only two pairs (one for each direction); successor of 10Base-T,
- 100Base-FX: fiber-optic cable with two fibers, one for each direction; not compatible with 10Base-F, uses a signal of a different wavelength.

100Base-TX and 100base-FX are standardized in IEEE 802.3u.

 **Gigabit Ethernet** allows the speed 1 Gbps, baseband. Representatives:

- 1000Base-T: UTP cable of the Cat.5 category (at least 5e recommended), all four pairs are used, the direction for individual pairs is determined adaptively – none is reserved for a specific direction,
- 1000Base-TX: an unsuccessful attempt to modify the 1000Base-T standard so that two pairs are reserved for each direction and a Cat.6 cable is required; while 1000Base-T is a standard by IEEE (IEEE 802.3ab), 1000Base-TX comes from the TIA (TIA/EIA-854); commercially unsuccessful – cannot be used on older Cat.5 or 5e wiring,
- 1000Base-X: standards for (not only) fiber-optic cables
 - 1000Base-SX (“S” as short) multimode fiber-optic cables, typical range (length of the path) is in hundreds of meters,
 - 1000Base-LX (“L” as long) singlemode (range in kilometers) or multimode (hundreds of meters) fiber-optic cable,
 - 1000Base-CX does not use optics, but a shielded twisted pair cable, the connectors are similar to UTP, but a different pin assignment (crimped differently); typical range is up to 25 meters, used in data centers to connect servers to switches (e.g. some IBM products).


1000Base-T is standardized in IEEE 802.3ab and the standards for the Gigabit Ethernet for fiber-optic are in IEEE 802.3z.


 **10Gigabit Ethernet** (10GE, 10GigE) allows the speed 10 Gbps, baseband. The CSMA/CD collision method is completely removed (unused), full-duplex communication is compulsory.

- 10GBase-R is a group of standards for fiber-optic media:
 - 10GBase-SR (short range) for multimode fiber-optic cables, range 62 m,
 - 10GBase-LR (long range) for singlemode fiber-optic cables, range cca 10 km,
 - 10GBase-LRM (long reach multimode) for multimode fiber-optic cables, range 220 m,
 - 10GBase-ER (extended range) for singlemode fiber-optic cables, range 40 km,

they differ not only in cables, but also in the wavelengths at which the signal is transmitted,


- 10GBase-CX4 uses a twin-ax cable (similar to coaxial, but instead of one central wire and one metal braiding as the second “wire” it uses two wires), range 15 meters, but very good latency, it is used in data centers to connect servers to switches,
- 10GBase-T is a successor of 1000Base-T, uses a twisted pair at least Cat.6 (with the range 55 m) or higher (better range),
- 10GBase-W goes outside the LAN segment – adds a new WIS sublayer to the L1 layer, which allows communication with WAN networks (SONET/SDH and others); works similarly to 10GBase-R, except that it encapsulates the Ethernet frame into a WIS frame in which data passes through the connected WAN.

 **40Gigabit Ethernet and 100Gigabit Ethernet** (40GigE, 100GigE) prescribes fiber-optic cables with a laser as a light source (for a distance of about 100 m for multimode fibers, or kilometers and tens of kilometers for singlemode fibers with various parameters), for a distance of several meters twinax cable (in data centers for connecting powerful servers to the network), and for a distance of up to 30 m, a twisted pair Cat.8 can be used for a speed of 40 GigE.

 **2.5 and 5Gigabit Ethernet** were published as standards in 2016, i.e. after “faster” siblings. This is twisted pair communication, formally 2.5GBase-T and 5GBase-T, IEEE 802.3bz standard. The physical layer was taken from 10GBase-T and adapted to uv worse cabling. 2.5G-Base-T can use unshielded twisted pair cable category Cat.5e, 5GBase-T uses category Cat.5e (less than 100 m distance) or Cat.6. While 10GBase-T does not support PoE, 2.5GBase-T and 5GBase-T should handle power over Ethernet.

2.3.2 Implementation

The physical layer is typically implemented in circuits, either in the form of a separate chip (on the motherboard or on a Network Interface Card (NIC)) or integrated in the chip with other components. It can consist of two parts.

 *MAU* (Medium Attachment Unit) or *transceiver* (short for transmitter/receiver) is an element on the NIC that detects signal presence, collision and signal transmission/reception. It can be

- external – for the old Ethernet using coaxial and on the contrary for the new (optical or twisted-pair) modules (XENPAK, XFP, SFP, SFP+, etc.),
- integrated in a chip with other parts of the network interface.

On network devices (switches, routers) we can find positions for SFP+ modules, which can be optical or metallic, with different specifications.



Figure 2.6: SFP+ transceivers for 10GBase-LR/LW¹

 The physical layer is divided to sublayers with communication interfaces.

¹Resources: <https://www.alternetivo.cz>, <https://www.atcmarket.cz>

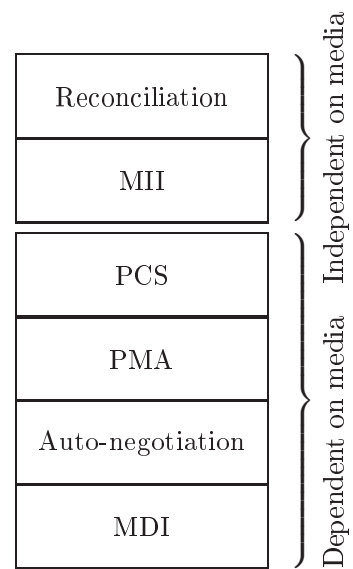
Media-independent sublayers:

- reconciliation sublayer,
- MII (10Mb a 100Mb), GMII (Gb) or XGMII (10Gb) – separately send and receive data paths with a width of 1 bit for 10 Mbps (bit-serial), 4 bits for 100 Mbps (nibble-serial) or 1 B for faster (byte-serial).

They provide a logical connection among the MAC L2 sublayer and various sets of media-dependent sublayers.

Media-dependent sublayers:

- PCS (Physical Coding Sublayer) – coding, multiplexing, synchronization of outgoing and incoming data,
- PMA (Physical Medium Attachment) – signal transmission and reception, timer recovery mechanism (ensuring timer synchronization at the beginning of the data stream),
- Auto-negotiation – these sublayers negotiate the specific connection characteristics to suit both NICs at the beginning of the transmission,
- MDI (Medium-Dependent Interface) – leading to the cable connector.



An example of a layer scheme (here for 1Gb Ethernet) can be seen in Figure 2.7 on page 34. The physical layer is completely redefined in standards for 10G Ethernet.

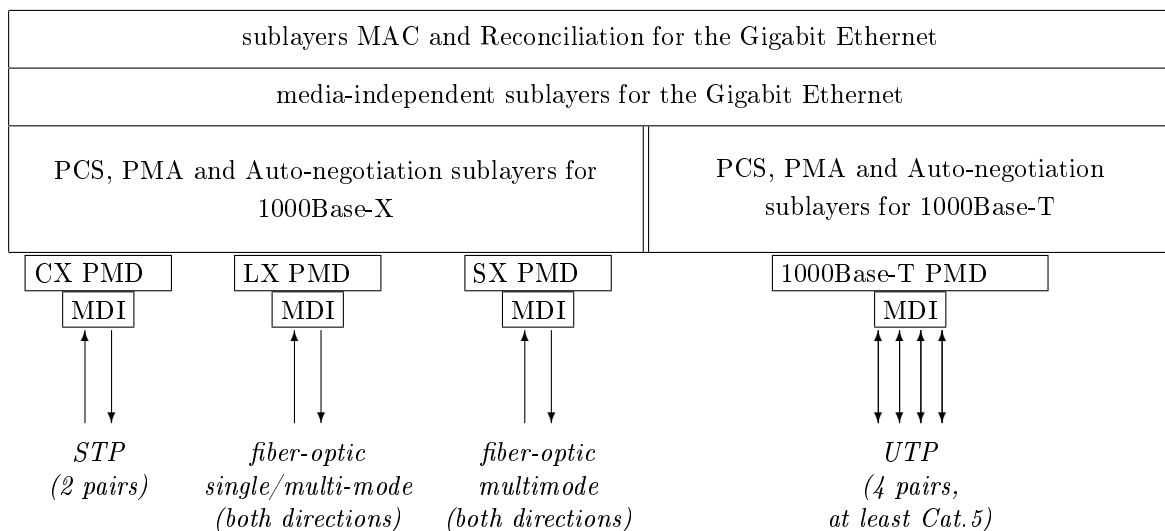



Figure 2.7: Physical layer schema for the Gigabit Ethernet


 **Task**

The web page <http://www.cables-solutions.com/sfp-vs-sfp-plus.html> explains the differences between SFP and SFP+ modules. Read this page and clarify the differences between these two types.

Choose an online store and browse the current offer of optical SFP+ modules. Are there any SFP+ modules that are not optical?



2.3.3 Power over Ethernet

 *PoE* (Power over Ethernet) is a standard describing the possibility of powering smaller devices via a network (metallic) cable. The advantage is that we do not have to supply the power cord to the given device. Of course, such a powered device must suffice with lower consumption, typically for example IP phones, IP cameras, smart lights and their controllers, and some Wi-fi access points connected by twisted pair to a device that can function as a PoE source (which is usually a switch).

The original PoE standard is IEEE 802.3af-2003 (i.e. from 2003), the newer version PoE+ is IEEE 802.3at-2009. Then the IEEE 802.3bu-2016 amendment (PoDL, Power over Data Lines) came and the newest one is IEEE 802.3bt, called 4PPoE or PoE++. They differ in the amount of current supplied, the wires used, the purpose for certain devices (for example, PoDL is mainly intended for automotive and industrial devices) and other characteristics. All of them suffice with Cat.5 metallic cables.


In practice we encounter several different solutions – active PoE is standardized as the IEEE 802.3af standard, and there are also several non-standardized older implementations.



Remark:

A switch serving as a PoE source needs a sufficiently dimensioned power supply (the component which, as with conventional computers, provides the AC/DC conversion and adaptation of voltage), and yet it usually cannot supply power to all connected devices, the remaining connected devices need their own power supply.



 We distinguish *power classes* 0–4 (for PoE+) or 0–8 (PoE++) according to what current and maximum power the given device needs, while class 0 is for non-PoE devices. The higher the power consumption, the higher the class.

For power purposes, some of the wires are used in the cable. If we only need two pairs for the data (Fast Ethernet), then the remaining two pairs are used for power supply (i.e. the pairs 4+5, 7+8). If we need all four pairs for data (Gigabit Ethernet and faster), then all four pairs are used for power at the same time.

Similar to the auto-negotiation of connection parameters as the speed and duplex, for PoE the class is determined too. First, a detection is performed in which a low-voltage current is applied to the connected device, which, if the device does not know PoE, should not harm (corresponds to class 0). If the device responds, detection continues by determining the appropriate power class.



Additional information:

- <https://community.fs.com/blog/poe-switch-types.html>
- <https://www.veracityglobal.com/resources/articles-and-white-papers/poe-explained-part-2.aspx>
- <https://gcti.com/power-over-ethernet-poe-vs-power-over-ethernet-poe/>



2.4 The Course of Transmission

We have learned that bidirectional transmission between two devices can take place in one of the following ways:

- in half duplex, where both devices can transmit, but alternately (only one device can transmit at a time), i.e. they share the same communication channel,
- in full duplex, where both devices can transmit at the same time, because (at least) one communication channel is reserved for each direction.

2.4.1 Half Duplex

Half duplex mode can be used up to a maximum speed of 1 Gbps. Because the devices on the segment share a communication channel, the CSMA/CD collision method described above is used, including the Backoff algorithm.

Definition (Collision Window)

A Collision Window (Slot Time) is the amount of time a device has to listen on a carrier to intercept another device's transmission and detect a collision hazard.



As far as the collision window is concerned, the following relations apply:


- the largest possible distance between devices in the same collision domain (the greater the distance, the longer the device has to listen, because the signal takes “longer”) – the larger the maximum allowed distance, the larger the collision window has to be,
- minimum frame length (the device must find out during the frame transmission that a collision has occurred in order to cancel the transmission, wait according to the Backoff algorithm and retransmit) – the smaller the minimum frame length, the smaller the collision window and the smaller the collision domain,
- if there is a small collision window, the collision domain has to be small too.

It follows that there is a very close relationship between the collision window, the size of the collision domain (maximal Distance between devices in the network) and the minimum frame length.

So here we balance between two requirements:

- We want to cover as much space as possible, so long cables and a large collision domain are desirable.
- We do not want to overwhelm the transmission path unnecessarily in case we need to send even small frames (if it is necessary to send a smaller amount of data than the minimum limit, a “pad” – data without meaning – have to be added), so it is more efficient to have a rather low minimum length limit.


The third factor can be added – the transmission speed. If we only increase the transfer rate, we have to increase the minimum frame length limit or decrease the domain, because increasing the rate will affect the transfer time.

 For 10 Mbps, the minimum frame length without preamble is set to 512 bits = 64 octets (see Ethernet II frame structure on page 28), so transmitting 512 bits takes 51,2 μ s. The maximum time

that the transmission between the two outermost devices in the collision domain can take is half that value. The distance in meters then depends on which transmission medium is used (metal and fiber-optic cables propagate the signal at different speeds per meter).

If the minimum length of a MAC frame without a preamble is set to 64 octets, then what is encapsulated inside should be at least $64 - 18 = 46$ octets long (subtract the length of the fields in the header and the back of the MAC frame).

Theoretically, there may be a situation where less than 46 octets of data should be transmitted within a frame. The IEEE 802.3 standard, in conjunction with LLC and SNAP, solves this with a “pad” added after the data, and the pad length can be derived from the value in the *Length* field. We will show the solution by example.

 **Example**

If the *Length* of the LLC/SNAP frame is < 46 (here specifically 31), then the following applies:

$$\begin{aligned} \text{DSAP} + \text{SSAP} + \text{Control field} + \text{Data} + \text{Pad} &= 46 \\ \text{Length} + \text{Pad} &= 46 \\ \text{Pad} &= 46 - \text{Length} \end{aligned}$$

For the case indicated in Figure 2.8, $\text{Pad} = 46 - (28 + 1 + 1 + 1) = 15$. This information is important to make it clear at which octet of the frame the checksum field begins. Note that the *Length* field also includes the length of the LLC header (yellow in the figure).

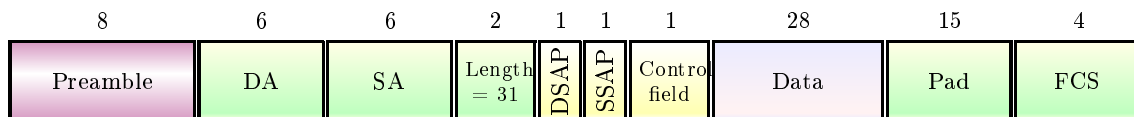




Figure 2.8: Short LLC frame encapsulated in MAC frame according to IEEE 802.3

Ethernet II frames do not solve this, it is assumed that the encapsulated data maintains a certain minimum length. Because “inside” is usually an IPv4 packet with a header at least 20 octets long, there is at least 26 octets left for what is encapsulated inside the IP packet, which is not so much.

 When going to a higher speed 100 Mbps, it was decided to keep the minimum frame length (i.e. the same applies to the frame format as written in the previous paragraph) and the size of the collision window, so it was necessary to reduce the size of the collision domain between switches (approximately $10\times$). On a metal cable, this means reducing the collision domain from 2000 m to 200 m, which means that there is a distance of at most 100 m between a terminal device and a switch (it remains) and between two switches a maximum of 200 m.

 When switching to Gigabit Ethernet (1 Gbps), this solution was not possible (20 m as the highest possible distance would be really small), so for a change, the minimum frame length without preamble was increased to 512 octets (4096 bits). Because smaller amounts of data are often transmitted, it is necessary to add an additional “pad” (Carrier Extension, non-data suffix) for frames below the limit, after the checksum. The location is indicated in Figure 2.9.

The carrier extension is added to outgoing frame after the checksum calculation on the MAC

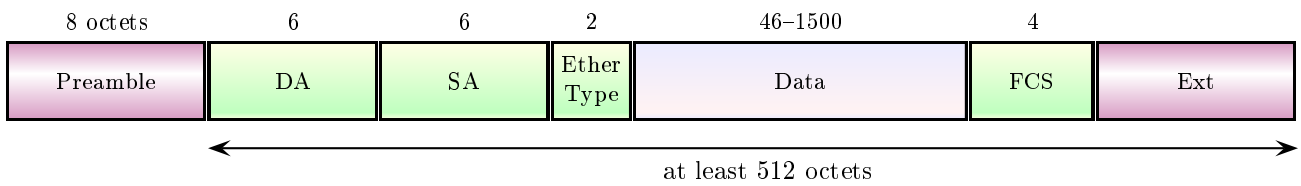


Figure 2.9: Carrier extension for the Gigabit Ethernet frame


sublayer. When the frame is received, it is removed on the MAC sublayer of the target system before the checksum is checked. The composition of the symbols in the carrier extension is patented – it has to be distinguishable from the field with the checksum.

**Additional information:**

The text of the patent is available at <http://www.google.com/patents/US5940401>.



If a larger number of frames with a small amount of encapsulated data were sent very often, the line would be unnecessarily used. Such a case is handled differently:

 *Burst mode* is just for sending sequences of short blocks of data. It is used only for transmissions at speeds from 1 Gbps upwards, ie from Gigabit Ethernet. “Short” frames are sent in a burst that has the following structure:

- the first frame in the cluster has the structure described above, including a carrier extension if it is less than 512 octets,
- *IFG* (InterFrame Gap) follows – a special block of bits filling the space between frames, used to detectably separate frames in a burst,
- other frames in sequence separated by *IFG* blocks; if they are shorter than 512 octets, it is not necessary to add a carrier extension.

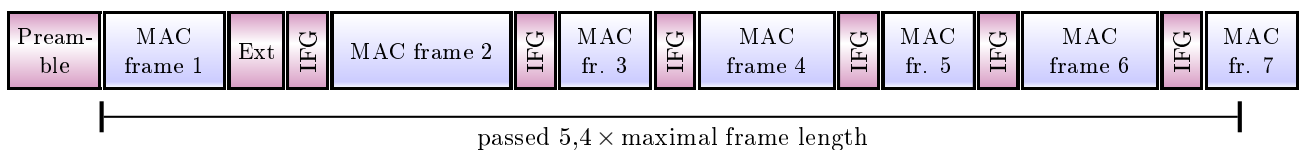



Figure 2.10: Using the burst mode

The total length of the burst should not exceed approximately $5.4 \times$ the maximum possible frame length (more precisely, 8 192 octets), but if this limit is not exceeded until the last frame in the burst is sent, it may still be sent and the burst is completed after the extra frame.

2.4.2 Full Duplex

In full duplex, the situation is much simpler. At the logical level, we basically only encounter point-to-point links and all links are actually reserved, the device can transmit at any time. The CSMA/CD collision method is not used and the connection parameters are not limited by the properties of the collision domain and the collision window, only by the physical properties of the transmission path (for example by attenuation). Transmission still takes place in burst mode, as described for half duplex.

 The only problem (which can occur with half-duplex, but is typical of full-duplex) occurs when the transmitting device is constantly active but the receiving device is unable to receive the frames. To receive a frame, it is not trivial; in addition to processing at the physical layer, the frame has also to be decapsulated and processed accordingly, and the processor (which is also involved) may have a different work than processing frames from the given sending device. This problem is solved by a combination of the following methods:

- large *buffer*, it stores what could not be processed yet,
- the sender needs to be slowed down, informed that it has to wait a while.

The second option means using the so-called *pause frame*, ie a special frame that is sent to a “diligent” sender; this framework contains, in particular, information on how long the transmission needs to be interrupted.

2.5 Switches and Loops

We will focus on the procedure that is standardized for bridge devices, however, we will talk about switches. There is some difference between bridges and switches (switches are not standardized, they have more ports and the switching process is implemented in hardware), but the principle is the same (they work at the L2 layer, maintain a table of MAC addresses and ports, switch traffic, separate segments).

2.5.1 The Spanning Tree Protocol

Let’s imagine a larger network where we have several switches. Because we want the network to work even if a transmission path is interrupted or a switch goes down, the switches are physically connected more than necessary – some paths are *redundant* (multiplied).

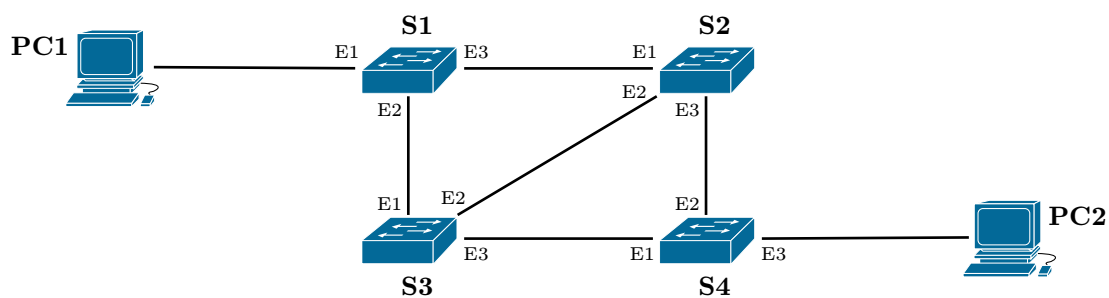


Figure 2.11: Network with loops

Redundant paths are practical for when part of the network goes down, but can cause some problems. Let’s imagine a few possible scenarios:

1. The computer PC1 sends the frame to PC2. The first part of the path (switch S1) is unambiguous, but what about the next steps? Should the frame be sent from the switch S1 via the port E2 or via the port E3?
2. The switch S4 receives the frame on the port E3 and sends it to the ports E1 and E2. What will the other switches do?


- The switch S2 receives the frame on the port E3 and sends it to the ports E1 and E2.
 - The switch S3 receives the frame on the port E3 and sends it to the ports E1 and E2. The same frame (which it does not know) is received on the port E2 (from the switch S2) and sent to the ports E1 and E3.
 - The switch S2 again receives the frame on the port E2 (from switch S3) and sends it to the ports E1 and E3.
 - The switch S1 receives the same frame twice from the port E2 and twice from the port E3, sends it sequentially a total of twice to the port E2, twice to the port E3 and four times to the port E1, thus the frame reaches PC1.
 - Meanwhile, the broadcast frame has made several trips back to S4, PC2, and then back to other switches...
3. Suppose PC1 is just joining the network, so there is no switch in the MAC table yet. The computer PC2 sends it a frame (PC2's MAC address is the destination). The switch S4 does not know this address, so it sends the frame to the ports E1 and E2, etc. – the frame travels through the network in the same way as a broadcast frame. It will reach PC1, but copies of it will still roam the network for a long time.

**Remark:**

The scenario described in point 2 is called the *broadcast storm*. It is characterized by identical copies of the same broadcast frame constantly wandering the network, making the network defacto congested and unusable after a certain period of time.



That implies that something is wrong. Redundancy of paths is good, but we should also prevent uncontrolled repeated roaming network frames. We can prevent this if we *remove loops* from the network (as far as communication is concerned; loops may remain in the physical topology due to redundancy).

 The *Spanning Tree Protocol* (STP) is standardized as IEEE 802.1D as a mechanism to remove (logical) loops in the graph of a bridge network (i.e. switches in our case). This protocol only knows the bridges (or switches), it is not interested in any other devices (so it will ignore PC1 and PC2 in our network).

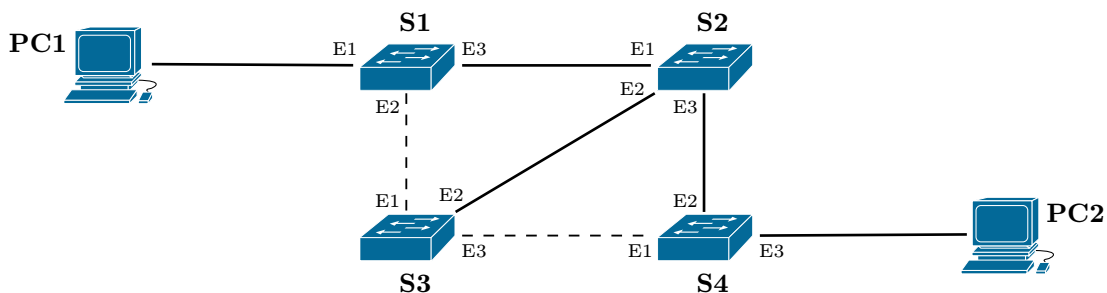



Figure 2.12: Corrected network, loops are removed

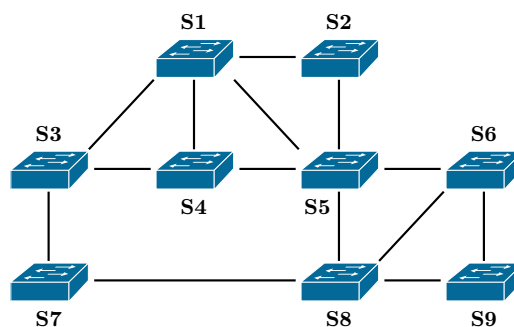
The purpose is to reconfigure the switches in the network in such a way that some ports are not used for communication (they are either disabled/blocked or used only for service purposes), thus removing

loops from the network. The STP protocol basically solves the “graph skeleton search problem”, the purpose of which is to leave only such links in the graph that there is exactly one path (no more, no less) between any two nodes in the network (switches), ideally the fastest one.

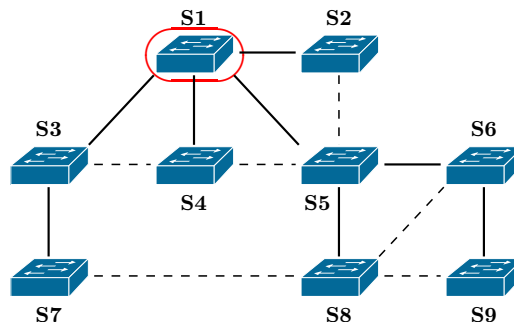
 Actually, we create *tree* from the switch network (“spanning tree” means covering tree). One of the switches is declared to be the root of the tree (*root switch*) and the fastest path to the root switch is sought from each other switch. All other paths are disabled. In our example, switch S2 could be the root switch and only links S1–S2, S3–S2 and S4–S2 would remain active in the network.

Example

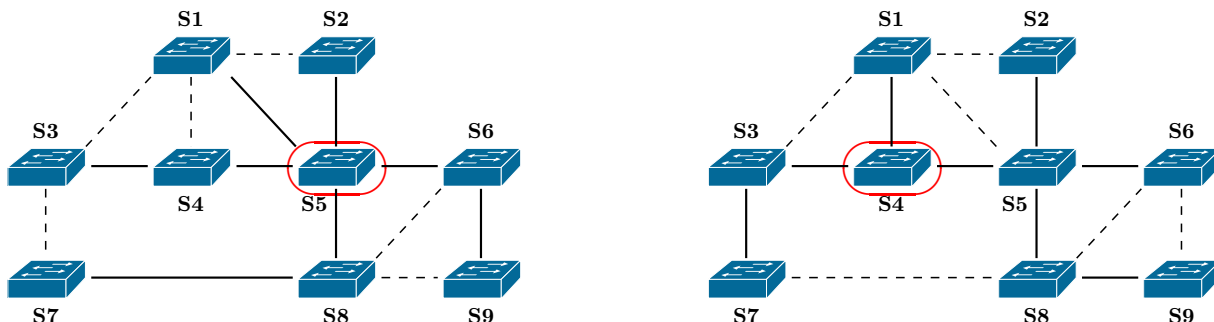
In the network in the following figure we see a total of nine switches that are connected in a physical mesh topology. Here we have loops that need to be resolved using STP.



The first task is to select the root switch. We have a choice, but in a real situation we would probably have some criteria for this choice. Here we simply choose the switch S1. We block some ports, making a total of six paths unavailable for regular communication.



There are other options – we can choose for example the switch S5 or S4 as the root switch:



Also in these cases, some ports are disabled for normal operation, and there is just one (preferably optimal) path from each switch to the root switch.



2.5.2 Convergence

What determines the root switch? Each switch that “understands” the STP protocol (the protocol is implemented on it) has a special identifier assigned to it. This identifier is 8 octets long and consists of two parts:


- Priority (2 Bytes)
- MAC address (6 Bytes).

1	2	3	4	5	6	7	8
Priority		MAC address					

By default, there is a number 0×8000 in the priority field, which is decimal 32768. It follows that no two switches have the same identifier (they differ at least in MAC address, each switch has a different one).


STP selects as the root tree the switch whose identifier is the *smallest number*. If we relied only on this procedure, the root tree might be selected to be the switch we don’t like (it would select by switch MAC address) – the solution is just to change the priority of the switch we selected to *smallest value*. Because the identifier contains the priority first and then the MAC address, this change always has the desired effect.


It remains to determine which connections (or ports) should be deactivated and which should remain active. If there is a path to the root bridge through more than one port, then the optimality of the path is calculated for each port (depending on the bandwidth/speed, utilization, we are talking about the cost of the path, or cost per port); the priority parts of all switches on the path to the root bridge are summed and the path with the lowest sum is selected. Only the port for the most optimal path will determine the active path to the root switch.

 So the port is either active or not active in terms of forwarding frames with regular traffic. We refer to this property as *state*, distinguishing the following port states:

- *forwarding* – works, forwards all traffic,
- *blocking* – does not forward regular traffic but receives STP administrative frames (called BPDU frames) in order to be able to activate itself when needed,
- *listening* and *learning* – intermediate states in case the port has to transition from the blocking state to the forwarding state; in the listening state the switch processes BPDU frames, in the learning state it receives (but does not forward, discards) frames with regular traffic and starts to create a MAC address table,
- *disabled* – completely disabled port, it forwards nothing.


Ports move between these states, although they spend most of their time in the forwarding or blocking states.

 From time to time (especially when the network topology changes) a reconfiguration – *convergence process* – is performed. The purpose is to ensure that the network is in a converged state (i.e., in our case, that there are no loops in the network and that only the ports for the most optimal path are active). A new root switch may need to be determined, and port states may also change (for example, if a port is to go from the blocking state to the forwarding state, it will spend some time in the listening and learning states to be able to set all parameters correctly and complete the MAC address table).


 Each port has a specific role, which is determined by the position of the switch in the hierarchy, and also distinguishes whether the port is pointing up to the root switch or down to the child switches.

Suppose we have the switches drawn so that the root is at the top. The role determines what state this port will be in at any given time. The following port roles can be used:

- *root port* – port that directs to the root switch, i.e. every switch except the root switch has one root port pointing “up”, the root port is always active, i.e. in the forwarding state,
- *designated port* – a port that is active and is directed down the hierarchy, i.e. to the child switches (so downstream of the designated port we have a subtree of switches); the root switch has all its ports except the blocked ones in the designated role, while the switches at the bottom of the hierarchy (the ones with the longest path to the root switch) have no designated ports,
- *blocked port* – this port has not been selected, it is not used for regular traffic (i.e. it is usually in blocking state); the RSTP variant (see later) distinguishes two states instead of a blocked port:
 - *backup port* – this port is blocked, but it is a backup for the root port; the real root port has more optimal path to the root switch, but if the optimal path were no longer usable or lost its optimality, the backup port will become the root port,
 - *alternate port* – this is the alternative for some of designated ports.


 To make it all work, the switches talk to each other using special frames called BPDUs (Bridge PDUs). They are used to maintain and possibly update network information needed for STP, and are sent to a multicast address that identifies all devices implementing STP (i.e., the 01:80:c2:00:00:00 address). BPDUs are sent periodically every 2 seconds, and then whenever the switch network changes.


The BPDU includes, among other things, information that it is the STP frame and which version, which switch is the root switch (its bridge ID/switch ID), the ID of the sending switch, information about the frame validity period, the cost of the path from the sender to the root switch, etc. The BPDU is encapsulated in an IEEE 802.2 LLC frame and then in an IEEE 802.3 MAC frame.

 **Additional information:**

<https://wiki.wireshark.org/STP>



 If a new switch is added to the switch network, it must also be added to the STP tree. The new switch considers itself a root switch at first, but leaves its ports in a listening state for now (it receives BPDUs but does not send anything). If it finds a root switch ID in a received BPDU that is lower than its own ID, it stops considering itself as a root switch and gradually integrates itself into the structure (for each port, it calculates the path cost to the neighbor switches and sums the cost with the path costs to the root switch declared by neighbors in the received BPDUs, and chooses root port and designated ports).

 Each port has its cost assigned. This is determined by the speed of the port.

If the switch receives a BPDU from a neighbor in which the neighbor claims that the path from itself to the root switch is worth, for example, 8, the switch adds the cost of the port leading to the neighbor to the neighbor’s entry to get the cost of the path from itself to the root switch. In our example, if the neighbor advertises a cost of 8 and the port cost is 19, then the cost of the path to the root switch is $8+19=27$.


Link speed	Port cost
10 Gbps	2
1 Gbps	4
100 Mbps	19
10 Mbps	100

Table 2.2: Port costs


If two ports calculate the same path cost to the root switch, then a different decision must be made. In this case, the neighbor ID of the neighbor located behind the port is used to decide. The path through the neighbor with the lower ID wins.

2.5.3 STP Protocol Variants

The original STP protocol has been standardized as IEEE 802.1D.

 **RSTP.** Because the original specification was no longer sufficient for faster standards, a new standard, Rapid STP (RSTP), was created as IEEE 802.1w, but is now included in the IEEE 802.1D-2004 revision (so that the original STP and the newer RSTP have been merged into one standard).

While the original STP took 50 seconds to converge (during which time the entire network was down, which is an awfully long time), the newer RSTP takes about 1 second, or less. There are other differences between the original STP and RSTP, but this difference is the most significant.

 **MSTP.** The MSTP (Multiple STP) variant has been standardized as IEEE 802.1s, also referred to as MST (Multiple Spanning Tree). The purpose is to streamline the operation of STP for the case where virtual networks (VLANs) are defined in the network. It has also been added to the standard, but IEEE 802.1Q, as the IEEE 802.1Q-2005 revision: the extension of RSTP for VLANs.

Basically, the idea is to have a separate STP instance for each VLAN or group of VLANs, so there could be a slightly different active link structure for each VLAN (or several). The purpose is both to make communication within each VLAN more efficient (so that, for example, no unnecessary communication is carried out through switches that have nothing to do with the VLAN affected), but also to be able to balance the load in the network (a certain port will be blocked for one VLAN and active for another, for another port it may be different, each using different paths).

Task

Find the last entry at <https://packetlife.net/captures/protocol/stp/> (802.1D_spanning_tree.cap) and display it (either in Wireshark or Cloudshark). Check the parameters sent in the BPDU. What is the role of the sender in the network?



2.5.4 Configuration

If the switch implements STP, then this protocol is usually already active. Therefore, we can just connect the switch to the network and the STP algorithm itself will take care of assigning port roles.

Example

Suppose the topology presented in Figure 2.13.

We can check the STP parameters – first for S1:

```
S1#show spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    32769
            Address    0001.C713.324A
```

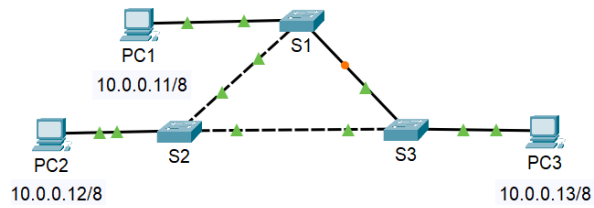


Figure 2.13: Topology for the STP configuration

```

Cost          19
Port          1(FastEthernet0/1)
Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec

```

```

Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
Address    00E0.B007.E7BB
Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
Aging Time  20

```

Interface	Role	Sts	Cost	Prio.Nbr	Type
Fa0/1	Root	FWD	19	128.1	P2p
Fa0/2	Altn	BLK	19	128.2	P2p
Fa0/24	Desg	FWD	19	128.24	P2p

The header contains information about the root switch, followed by information about this switch. The Root ID is different from the Bridge ID of this switch (also MAC addresses), so S1 is not the root switch.

As we can see, f0/1 is the root port, f0/2 is blocked and it is the alternative for the root port. The port cost is 19 for all ports (the value for the fast ethernet ports).

The STP parameters on S2:

```

S2#sh spanning-tree
VLAN0001
Spanning tree enabled protocol ieee
Root ID    Priority    32769
Address    0001.C713.324A
This bridge is the root
Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
Address    0001.C713.324A
Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec
Aging Time  20

```

Interface	Role	Sts	Cost	Prio.Nbr	Type
Fa0/1	Desg	FWD	19	128.1	P2p
Fa0/2	Desg	FWD	19	128.2	P2p
Fa0/24	Desg	FWD	19	128.24	P2p

As we can see, S2 is the root switch. Its priority is the lowest among all switches. All connected ports are active and their state is “designated” (they all are downlinks in the hierarchy).

The same for S3:

```
S3#show spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    32769
            Address    0001.C713.324A
            Cost      19
            Port      2(FastEthernet0/2)
            Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769 (priority 32768 sys-id-ext 1)
            Address    000B.BE73.B349
            Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time 20

Interface          Role Sts Cost      Prio.Nbr Type
-----
Fa0/2              Root FWD 19        128.2   P2p
Fa0/1              Desg FWD 19        128.1   P2p
Fa0/24             Desg FWD 19        128.24  P2p
```

We have decided that the root switch will be S1. We can do this by changing its priority – we will choose a number smaller than the default priority (32769). The priority value must be a multiple of 4096 (we will be warned if we want to choose a different value).

```
S1#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
S1(config)#spanning-tree vlan 1 priority 20480
```

Within seconds, convergence takes place, changing the configuration of all switches. The port role assignments will also change.

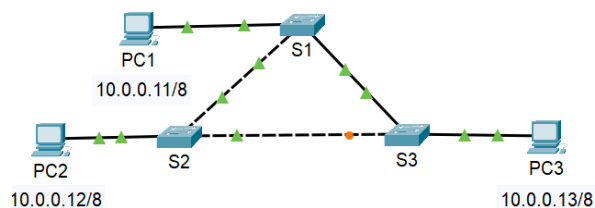


Figure 2.14: Topology for the STP configuration after root change


If we don't want to remember directly the priority numbers that are used during configuration, there is another possibility. We can simply label the future root switch as primary:

```
S1(config)#spanning-tree vlan 1 root primary
```

In addition, we can select a backup root switch and mark it as secondary. If a failure occurs on the root switch, the secondary switch will take over its role.

```
S2(config)#spanning-tree vlan 1 root secondary
```



 A special case is the situation when we have redundant links stretched between two adjacent switches (simply two cables between two switches), the situation is indicated in Figure 2.15. This situation is called a *tie*. Which switch will be the parent (i.e. root) is determined as suggested above (i.e., switch ID). All ports of the root are set to the “designated” role. But what about the two ports of the second switch, which one will be active?

Logically, the same path cost applies for both of these ports and we can’t even decide by the switch ID of the neighbor, because we actually have the same neighbor behind both ports. In this case, the third criterion applies – because even on the ports we have a pair of data: port priority and port number. Port priority is usually 128, so the port number is the deciding factor. Of course, we can change the priority to affect the determination of the active port.

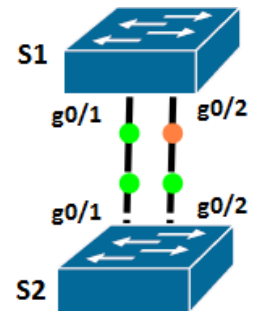


Figure 2.15: STP tie

Example

In the picture 2.15 we have switches S1 and S2. According to the LED lights we can conclude that S2 has become the root switch and the connection between ports g0/2 on both switches will be blocked.

```
S1#show spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    32769
            Address    0007.EC99.A154
            Cost      4
            Port      25(GigabitEthernet0/1)
            Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769 (priority 32768 sys-id-ext 1)
            Address    000D.BDC5.2941
            Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time 20

Interface          Role Sts Cost          Prio.Nbr Type
-----
Gi0/1              Root FWD 4             128.25  P2p
Gi0/2              Altn BLK 4             128.26  P2p
```

Port g0/1 is the root port, i.e. it leads to the root switch, and is in the forwarding state (so it forwards all traffic). Port g0/2 has the role of Altn, which is “alternated”, or blocked (it is the one with the orange light). The numbers of these ports are not 1 and 2, but 25 and 26 – they are determined globally uniquely for the whole switch, the lower numbers are for fast-ethernet ports f0/1–f0/24.

Same listing on switch S2:

```
S2#show spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    32769
            Address    0007.EC99.A154
            This bridge is the root
            Hello Time 2 sec  Max Age 20 sec  Forward Delay 15 sec
```

```

Bridge ID Priority    32769 (priority 32768 sys-id-ext 1)
Address    0007.EC99.A154
Hello Time 2 sec Max Age 20 sec Forward Delay 15 sec
Aging Time 20

```

Interface	Role	Sts	Cost	Prio.	Nbr	Type
Gi0/1	Desg	FWD	4	128.25		P2p
Gi0/2	Desg	FWD	4	128.26		P2p

Both switches have the same priority (32769), they differ in address. The MAC address of switch S2 is the smaller number, so S2 becomes the root.


Ports g0/1 and g0/2 are both in the designated role and both forward, which may seem strange when the counterpart of the second port is actually blocked. However, this is a common case – only one side of the link is blocked, so that it can be activated as quickly as possible if necessary. So on the parent switch side, the port in the designated role is in the forwarding state, and on the other side, the port in the alternate role is in the blocked state. Regular traffic does not pass through (it is dropped on the blocked port), so a loop is not created.

```
S4(config-if)#spanning-tr vlan 1 port-priority 32
```



2.6 EtherChannel and Port Aggregation

2.6.1 Properties

 *EtherChannel* is a technology that allows to combine (aggregate) up to 8 physical Ethernet lines into one. Two nodes (typically a server, switch, or router that support this technology) are connected by multiple active physical Ethernet links, with PDUs passing through one of these links.

In addition, up to 8 failover physical lines are supported in case active lines are inoperative. But this feature is supported only by some manufacturers, not all.

Remark:


This technology uses dual terminology – while Cisco speaks of EtherChannel, Solaris uses the term *trunk* for the same, but Cisco means by trunks something completely different. Therefore, when we hear the term “trunk”, we should be clear whose terminology is actually being used. The term *NIC Teaming* is also used for servers, but this is not exactly the same (grouping NICs into one logical)



The links that are part of EtherChannel are of the Fast Ethernet or faster type, and they must all be of the same type (i.e., for example, Fast Ethernet and 10Gb Ethernet links cannot be combined – same speed, same duplex, etc.). If we use VLANs, the connections should be within one VLAN or in trunk mode with the same parameters.

However, the total throughput is not equal to the sum of the throughputs of the individual physical paths. If multiple PDUs are sent within one communication to one node (by default), they are all sent along the same physical path, so the communication cannot be split.

2.6.2 Flow Control

 The default setting is that links within the EtherChannel are assigned according to the destination MAC address, i.e. PDUs that have a specific destination MAC address are all sent to the same link. However, this is not always convenient, so it is possible to set another distribution:

- [source MAC, destination MAC],
- only according to the source MAC,
- similar for IP addresses and ports.

Load distribution (i.e. individual connections) among lines within EtherChannel is performed according to a hashing balancing algorithm, which sorts PDUs on the basis of a specified criterion (for example, according to the target MAC address) and divides the entire communication depending on the rule indicated in table 2.3.

Each “data stream” determined according to the selected criteria (for example on the basis of the destination address) gets assigned a number from the interval 0–7 (i.e. one of 8 numbers) by hashing, regardless of the actual number of links in EtherChannel. This number then identifies the specific physical link.

		Number of EC ports						
		8	7	6	5	4	3	2
Hashing of communication streams among EC ports	1	1	2	2	2	2	3	4
	1	1	1	2	2	2	3	4
	1	1	1	1	2	2	2	
	1	1	1	1	1	2		
	1	1	1	1	1			
	1	1	1	1				
	1	1	1					
	1	1						

Table 2.3: Communication Hashing in EtherChannel


Example

If we have EtherChannel made over three physical links (EC ports), which corresponds to the last but one column of the table on the right, PDUs with three specific assigned numbers (0, 1, 2) will be sent to the first link, the other three numbers (3, 4, 5) will be sent to the second link and the remaining two (6, 7) to the third link.




It is obvious that the packets will be optimally exported if we combine 2, 4 or 8 links into EtherChannel.

2.6.3 Creating EtherChannel

 For EtherChannel, there are two *protocols negotiating the properties* of the EC connection:

- *LACP* (Link Aggregation Control Protocol) defined as IEEE 802.3ax,
- *PAgP* (Port Aggregation Protocol), proprietary protocol by Cisco.

The first of these protocols is generally applicable to all EtherChannel-enabled devices (including Cisco devices), and the second one is used only when connecting two Cisco devices. Only one of these protocols must be used in the EC link at a time (and the both devices must “understand” it).

 Both protocols allow to set the device to one of two states:

- *active* for LACP, *desirable* for PAgP, in which the device actively negotiates EtherChannel assembly,
- *passive* for LACP, *auto* for PAgP), in which the device passively waits for the start of EtherChannel negotiations from the neighbor.

In a pair, each device should be in a different of these states.

EtherChannel can also be created manually, the state is *on*. Then EtherChannel is created by us and no negotiation is needed and therefore neither of the previous two protocols is actually used.

Protocols cannot be mixed.

- If we choose the *LACP protocol*, the ports of the first device have to be in *active* mode (all ports in the bundle), and the ports in the bundle of the second device have to be in either in the *passive* or also *active* mode, otherwise EtherChannel is not activated.
- if we choose the *PAgP protocol*, the ports in the bundle of the first device have to be set to *desirable* mode, and the ports of the second device to either *auto* or *desirable* (otherwise the connection will not work).
- If the ports in the bundle on one device are set to *on* mode, the same mode has to be on the other side of the connection (this situation means that we did not use any protocol to negotiate EtherChannel, but manually determined that it should be EtherChannel).

Ports of Device #1	Ports of Device #2	⇒ This protocol i used:
active	passive or active	LACP
desirable	auto or desirable	PAGP
on	on	none

Table 2.4: Modes for EtherChannel negotiation

The slightest mistake or inconsistency will cause EtherChannel to not work. We need to look at the speed, duplex and the correct mode, as well as, for example, the VLAN settings. For access ports (which is less common for a connection between two switches), the same VLAN number has to be set on all ports in the bundle and on both sides. As for trunk ports (forwarding frames according to IEEE 802.1Q, this is the most common for the connection between switches), the trunk must be on both sides and also a list of allowed VLANs that can be sent to the trunk must fit.



Remark:

We have two opposing mechanisms here: while STP ensures that if there is more than one physical link between two devices, only one of them will be active; EtherChannel needs multiple active physical links to run between the two devices (which at the logical level will be understood as the only one). So if we connect two devices with more than two cables, STP will start up immediately (turned on by default), and only after the EtherChannel is properly configured all the physical links will be really active.



Procedure

Figure 2.16 shows two switches S1 and S2 connected with two cables. The both cables are active, so STP doesn't work here, and EtherChannel is configured.

Let's look at an abbreviated list of EtherChannel settings for the switch S1:

```
S1#show interface etherchannel
GigabitEthernet0/1:
Port state      = 1
Channel group = 1      Mode = Active      Gcchange = -
```

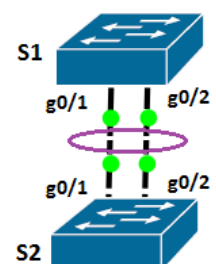


Figure 2.16: EC between switches

```

Port-channel = Po1    GC = -          Pseudo port-channel = Po1
Port index    = 0     Load = 0x00     Protocol = LACP
... (shorten)

```

```

GigabitEthernet0/2:
Port state    = 1
Channel group = 1     Mode = Active   Gcchange = -
Port-channel  = Po1   GC = -          Pseudo port-channel = Po1
Port index    = 0     Load = 0x00     Protocol = LACP
... (shorten)

```

So on switch S1 we set the active mode on ports g0/1 and g0/2 (i.e. we chose the LACP protocol to negotiate parameters), so it has to be passive or active on the ports of the second switch (it is better to passive, but the both is possible). The same listing on switch S2:

```

S2#show interface etherchannel
GigabitEthernet0/1:
Port state    = 1
Channel group = 1     Mode = Passive  Gcchange = -
Port-channel  = Po1   GC = -          Pseudo port-channel = Po1
Port index    = 0     Load = 0x00     Protocol = LACP
... (shorten)

```

```

GigabitEthernet0/2:
Port state    = 1
Channel group = 1     Mode = Passive  Gcchange = -
Port-channel  = Po1   GC = -          Pseudo port-channel = Po1
Port index    = 0     Load = 0x00     Protocol = LACP
... (shorten)

```

The procedure for the switch S1:

```

S1>enable
S1#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
S1(config)#vlan 10
... (we create all the VLANs that we want to transmit)

S1(config)#interface range g0/1-2
S1(config-if-range)#shutdown
S1(config-if-range)#channel-group 1 mode active
S1(config-if-range)#interface port-channel 1
S1(config-if)#switchport mode trunk
S1(config-if)#switchport trunk native vlan 98
S1(config-if)#switchport trunk allowed vlan 10,20,30,98,99
S1(config)#interface range g0/1-2
S1(config-if-range)#no shutdown

```

If possible, it's a good idea to configure ports when disabled. To make sure that all ports in the channel are set exactly the same, we first create EtherChannel and then set up everything (for example a VLAN trunk) on this virtual link.



**Task**

Review the Etherchannel configuration in the procedure above, and if you have two switches or some simulator (such as a Packet Tracer), try it out.



Although Etherchannel seems to refer to the L2 and Ethernet layers by name, it is actually possible to create Etherchannel channels at the L3 layer as well.


**Additional information:**


- http://www.cisco.com/en/US/tech/tk389/tk213/technologies_tech_note09186a0080094714.shtml
- <http://www.ciscozine.com/2008/11/04/configuring-link-aggregation-with-etherchannel/>




Chapter 3

Wide Area Networks and Access Networks

 **Preview:** WAN (Wide Area Network) is a large network that usually covers the area of a state, continent or the entire world. It connects not terminal devices, but various LAN, MAN or smaller WAN networks. In this chapter, we deal with WANs: the related protocols, architecture, communication, and the special types of networks to access WANs.

 **Keywords:** WAN, DCE, DTE, ABM mode, i-frame, s-frame, u-frame, X.25, Frame Relay, ATM, MPLS, label, FEC, LDP protocol, SONET/SDH, WDM networks, PPP, first-mile/last-mile networks or access networks, xDSL, DOCSIS, FTTx.


 **Goal:** The aim of this chapter is to understand principles of WAN networks – their architecture, connections, transmissions, related protocols, types of access. The purpose is to understand the principle of operation of WAN networks as a whole, including access to these networks.

3.1 WAN Networks

3.1.1 Typical Structure of WAN networks

 What topology is used in WAN networks? It depends on the point of view.

- External view: WAN networks are arranged in a (mostly) hierarchical structure very similar to the IP hierarchy. This is due to the fact that local and regional connectivity providers have to also obtain this connectivity from somewhere, from large multinational providers. The overall structure is like *backbone*.
- When we look at the internal structure of a WAN network, we usually find that it uses the *mesh* architecture (but not full mesh) on the physical layer: redundant connections that ensure high availability, fault tolerance.

 While for local networks it is more or less the fact that the data sent over these networks is related to the network owner, this is not the case for MAN and especially WAN networks. WAN operators simply own the network infrastructure they rent to their customers, the backbone to which customer networks are connected.

Network connectivity (possibly in a certain defined quality) is the main service provided by WAN, the functionality is also subject to the possibility to charge for the provided services. There may even be a problem with routing to non-unicast addresses, because connection-oriented communication over virtual circuits is typical of WAN.

 In WAN networks, we usually distinguish the following types of devices:

- *DCE (Data Circuit-terminating Equipment)* are devices within WAN (core). They usually do not communicate directly with anything that does not belong to the WAN, they can simply cooperate to establish a communication circuit and then transmit data through it. They are “invisible” for the customer.

Their typical feature is the speed of data switching (usually with hardware support), it is similar to switches in LAN networks. DCEs do not delay viewing transmitted PDUs, they are only interested in the “localization part” of the header.

- *DTE (Data Terminal Equipment)* are devices in the border between the WAN network and another network (LAN, MAN, . . .) connected to it, so they are border devices.

They need to “understand” both WAN and connected network protocols (they usually need L3 layer functionality). They work as translators between two types of networks.

Some WANs distinguish other types of their devices, but these terms will suffice for our purposes.

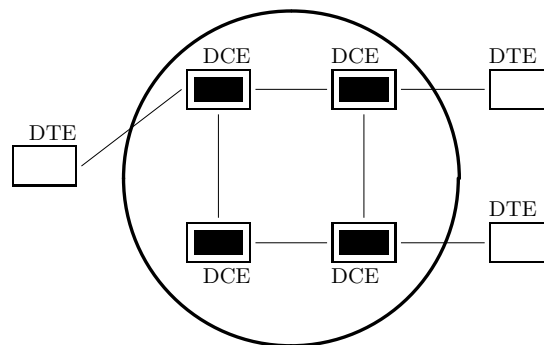


Figure 3.1: Devices in the Frame Relay network


 **Remark:**

Certain WANs use different terminology:

- P devices (Provider) are the same as DCE, they are inside WAN,
- PE devices (Provider Edge) border devices of WAN,
- CE devices (Customer Edge) border on the customer’s side, they provide access to WAN for the customer.


Thus, PE and CE devices can together for DTE.




 *SLA (Service Level Agreement)* is an agreement between a WAN network operator and its customer (a company that wants to interconnect its remote branches, for example). The SLA also includes guaranteed parameters for the transmission of customer data over the WAN, including the maximum guaranteed throughput that the customer pays.

3.1.2 L2 Protocols for WANs


L2 protocols, or data-link protocols, are simply the protocols working on L2 ISO/OSI layer. We know one of them – LLC (IEEE 802.2), but we can meet other L2 protocols in WAN networks. Usually, each WAN network has “its own” data-link protocol, however, we can trace certain common features to these protocols (because there are actually ancestor-descendant relationships between them). The ancestor of the common data-link protocols is HDLC (High Data Link Control). Its descendant is LAPB for X.25 networks, the next generation was LAPD for ISDN, then LAPF for Frame Relay, and others.

 **Network nodes and communication modes.** Data-link protocols usually distinguish between a *primary node* (starts communication, determines connection parameters) and *secondary nodes* (only reply). If it is a point-to-point communication, then we have one primary and one secondary node; while if it is a point-to-multipoint communication, one of the nodes is the “most important” – primary.

 The nodes communicate with each other in a specific *mode*:

- *normal response mode* (NRM) – the secondary node has to wait for a command from the primary node to communicate,
- *asynchronous response mode* (ARM) – the secondary node can start communication with the primary node without call,
- *asynchronous balanced mode* (ABM) – every node can start communication, the primary node is dynamically determined – the node that needs to communicate is set to the role of the primary node and others become secondary.

Newer data-link protocols typically operate in ABM mode.

 **Data-link frames.** We distinguish several types of L2 frames, and most of data-link protocols maintain these types:

- *I-frame* (Information Frame) – carries information from the upper layer and possibly control information, these frames support ordering, flow control, error detection, recovery. Purpose: data transfer.
- *S-frame* (Supervisory Frame) – contains control information, it is used for start or termination of connection, status message, acknowledge of I-frame.
- *U-frame* (Unnumbered Frame) – for control information (e.g. agreement of communication mode), unlike the previous types, it does not support frame numbering in sequence (ordering), and may also contain data from the upper layer. In general: for what fits into a single frame.

I-frames use Sequence Numbers for both directions (similar mechanism as for TCP), S-frames have numbers only for one direction, U-frame does not use any such numbers (it is unnumbered, it does not divide the communication into blocks).

The less space Sequence Numbers take up, the more space is left for special meaning bits (frame type, determining a specific request or response to a request, etc.).

 The general (simplified) structure of the WAN frame is as follows (see Figure 3.2, page 56):

- flag at the start of the frame – synchronization information of the form 01111110,
- address – this field has different formats for various WANs, depending on the way of addressing virtual circuits,
- control field – Sequence Numbers and special meaning bits according to the frame type,
- data (if there is some payload inside the frame),

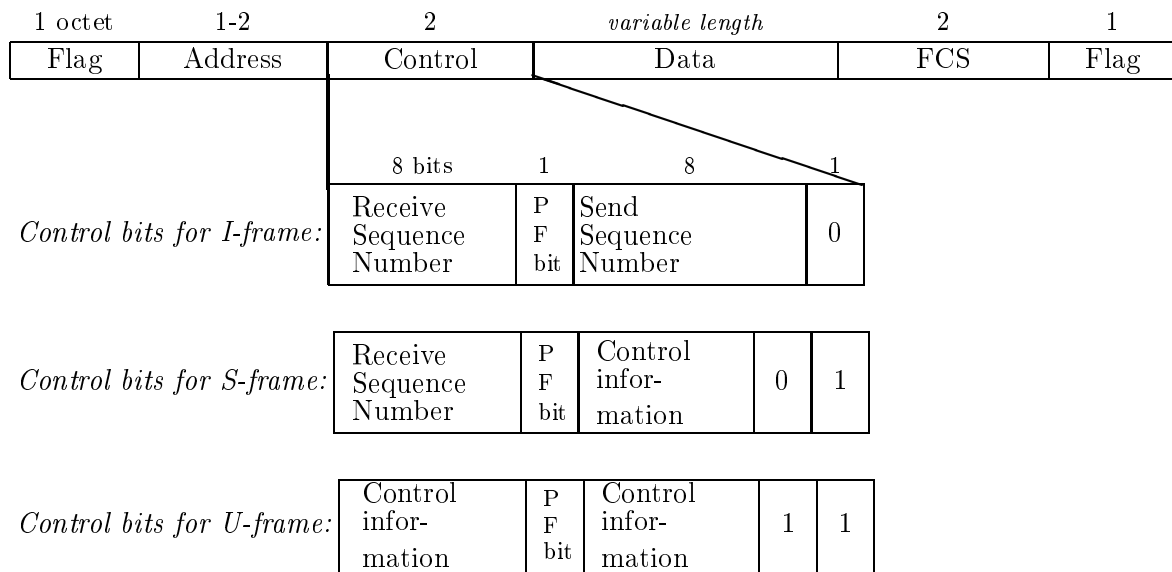




Figure 3.2: Typical format of a WAN L2 frame, here for the HDLC protocol

- FCS – checksum,
- flag – the same as at the start of frame.

 As indicated above, data-link protocols for WANs were actually evolving from one another. The ancestor of these protocols is SDLC by IBM, followed by HDLC, followed by LAPB used in X.25 networks, followed by LAPD for ISDN, then LAPF used in Frame Relay networks, and others. The PPP protocol and its variants are somewhat better known.

 **IEEE 802.2 (LLC)** we already know from the chapter on Ethernet, it is also a data-link protocol. It is designed for point-to-point connections, it works in asynchronous balanced mode (ABM). There are three types of frames – I-frame, S-frame and U-frame. The frame format is generally simpler because the LLC frame is always encapsulated in an IEEE 802.3 frame. Another address type (SAP) is used.

LLC offers three types of services:

- *unreliable connectionless service* (type 1) – most used, works as the datagram service (the frame is equipped with all available information including source and destination addresses), no error handling, in this respect it relies on upper layer protocols,
- *reliable connection-oriented service* (type 2) – for virtual circuits, includes establishing a connection, acknowledging after delivery of a limited group of frames, etc., only the first “service” frame contains address information,
- *reliable connectionless service* (type 3) – frames are acknowledged, this service is rarely used.

Terminal devices always support at least type 1, and then usually one of the remaining services.

Task

HDLC frame format has several variations. One of them is Cisco HDLC, which Cisco still uses for serial links. Look at <https://www.freecnastudyguide.com/study-guides/ccna/ch11/11-3-point-point-wans-layer-2/> and examine the Cisco HDLC format. What is the difference compared to the original HDLC format?



3.2 Selection of WAN Networks

The predecessors of WANs are the protocols SDLC and HDLC used for remote connection to mainframes and LANs, e.g. over telephone lines.

3.2.1 X.25 – Packet Switching


The oldest type of truly wide network was X.25 by ITU (later the ISO standard was created). This network was characterized by extreme reliability and low speed. It implemented approximately layers L1–L3, worked on the principle of *packet switching*.


X.25 used LAPB (Link Access Procedure Ballanced) as its L2 protocol, working on point-to-point links in the ABM mode.

The X.25 network is characterized by an excess of control mechanisms because it has been designed for unreliable transmission paths. In the past, it was used for public data networks, it lasted longer, especially in developing countries and on securely critical connections (for example, connecting payment terminals and ATMs to a protected network).


3.2.2 Frame Relay – Frame Switching

The newer WAN technology is represented by the Frame Relay network. The specification was originally created within ISDN. Because the lifetime of this solution proved to be good, it became independent in 1989. The service is connection-oriented, unreliable (with error detection, but no error corrections).

 **Addressing.** Frame Relay uses LAPF (Link Access Procedure – Frame), its address field is 2 bytes long. The numbers used as addresses are called *DLCI* (Data Link Connection Identifier), and they identify circuits, not customers.

 A customer needs to interconnect its branches. This connection corresponds to the virtual circuit, which is created between the branches if necessary. The WAN provider must be able to separate the circuits of different customers, but also the circuits to different branches of the same customer.

So at the beginning of the path, a packet sent on the edge DTE is encapsulated to a frame with a certain DLCI value, which tells the neighbor (probably a DCE) which circuit it is. The neighbor has information in its switching table that when a frame with this DLCI is received on a given port, it should be sent to a certain other port (because the given circuit continues there), and the DCE should adjust header in advance (it can change the DLCI value to the number expected by the neighbor to determine continuation of the circuit). Thus, the DLCI number gradually changes along the circuit.

 On individual devices, frames are switched according to the *switching tables*. Inside the FR network we find FR switches (= DCE), their table is simple, as we can see in Figure 3.3. Switching is implemented in hardware.

At the network boundary there are FR routers (in the role of DTE), which on one side communicate with the devices of the customer's local network, on second side communicate with FR switches. These devices contain two tables:

- the *routing table* specifying where the frame coming to the network with the given IP address should be sent,
- the *DLCI mapping table*.



Figure 3.3: Simplified Frame Relay switching table

The contents and meaning of both tables can be seen in Figure 3.4. On the third layer, the packet is routed to the given DCE (on the other border of the FR network) and the corresponding interface, the DLCI is set by the L2 protocol.

Routing Table:

Network	Next Router	Interface
10.5.0.0/16	172.16.1.2	S ₀
10.27.0.0/16	172.16.1.8	S ₁
⋮		

FR Map:

Next router	DLCI
172.16.1.2	100
172.16.1.8	200
⋮	

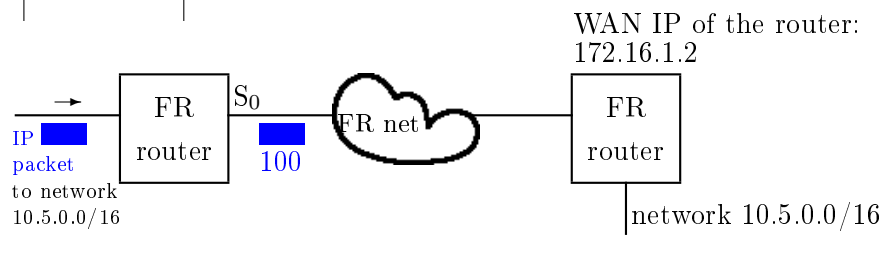




Figure 3.4: Routing on a Frame Relay router (DTE), incoming packet

 **Task**

Focus on the above Figures and check what happens when a packet with a destination IP address 10.5.24.7 arrives to the router whose tables are shown in Figure 3.4. What DLCI will it get for the first part of the path over FR? How will this frame with a given DLCI be handled on the switch shown in Figure 3.3?





 **Data flow control.** Frame Relay networks use several options to control the data flow.

Committed Information Rate (CIR) determines the throughput to be guaranteed on the connection. The frames sent above the CIR value can be dropped at higher network loads. For users, the CIR value has the following meaning:

- the higher the CIR, the more expensive the service,
- the smaller the CIR, the greater the probability of packet dropping and the potentially slower transmission.


Excess Information Rate (EIR) is the maximum data flow that a transmission path can handle (more precisely, what is above the CIR).

 Frames sent above the CIR value that do not exceed the EIR boundary are marked as *Discard Eligible (DE)*. They are passed repeatedly to the Frame Relay switches until sufficient bandwidth is available (until they fit in the CIR). Marking a frame as suitable for discarding usually does not mean that the frame will not be delivered, but that it can be delayed.

 The DE bit is one of the control bits in Frame Relay frame headers. There are other control bits in these headers, for example:

- FECN bit (Forward-explicit Congestion Notification) – with this bit, a DCE notifies all devices on the path (circuit) that this frame has been sent on congested links, and thus the DTE that is the target of the communication frames should lower the frequency of receiving frames (if the upper layer can respond),
- BECN bit (Backward-explicit Congestion Notification) – with this bit set in the response (e.g. acknowledgement), a DCE notifies the original source of communication about congestion, thus the source device should reduce the transmission speed.

Frame Relay networks were quite popular and were recently used, but MPLS networks gradually replaced them.

 **Additional information:**

- <https://www.cb nuggets.com/blog/certifications/cisco/networking-basics-how-does-frame-relay-work>
- https://www.cse.wustl.edu/~jain/cis788-95/ftp/frame_relay/index.html
- <https://learningnetwork.cisco.com/s/article/fundamentals-of-frame-relay-part-1-how-it-works-x>



3.2.3 ATM – Cell Switching


ATM (Asynchronous Transfer Mode) are WAN networks working on the principle of connection-oriented communication, unreliable service (without error reporting), on full duplex, in statistical multiplex. An important feature is the guarantee of quality of service for different types of data – in fact, one of the most important characteristics of ATM is the wide range of quality of service management options.

“Asynchronous” in the name does not mean asynchronous transmission, but asynchronous sending of “full” cells along the way, cells are occupied by data as needed, not by algorithm.

 **Remark:**

Be careful not to confuse the meaning of abbreviations. ATM is not only a WAN network, but it is also referred to as an cashpoints (British English] – in US English it is Automated Teller Machine (ATM) as well. However, this does not mean that ATMs as machines should be connected via the ATM network...



 Unlike FR, circuit addressing is two-valued. *Virtual Path* and *Virtual Channel* are similar to DLCI. Multiple virtual paths can lead in one physical transport path, multiple virtual channels lead in one virtual path. The virtual path number is called *VPI* (Virtual Path Identifier), the virtual channel number is called *VCI* (Virtual Channel Identifier). The *VPI/VCI value* fully identifies the transmission path between two neighboring nodes, changing on each ATM switch.

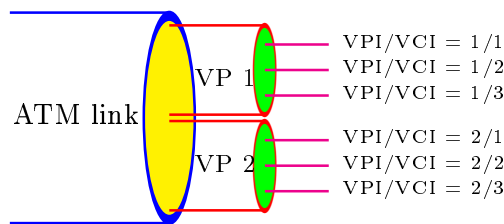


Figure 3.5: Virtual paths and virtual channels in ATM (simplified)

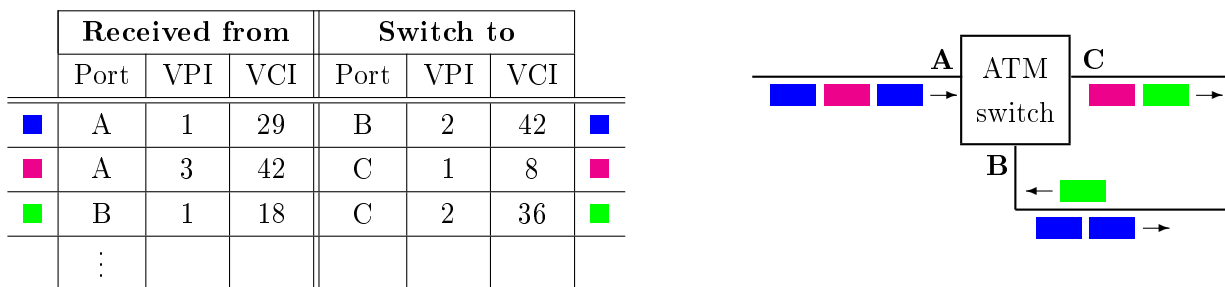



Figure 3.6: Example of a simple ATM switching table

An ATM switch maintains a *switching table*, in which incoming and outgoing VPI/VCI numbers are associated, the record is created during the connection establishment (circuit creation).

Figure 3.6 shows a simplified and abbreviated switching table of an ATM switch. There are three virtual paths there. Each row contains switching information about one customer's circuit, for example in the first line we find that the cell coming from the port A, the path 1 and the channel 29 is switched to the port B, the path 2 and the channel 42.

 ATM PDUs are called *cells*, they are very small with a constant length of 53 octets (for easy and fast switching). In order for any data to fit in the cells at all, the header is very small, only 5 B, with 48 octets left for transmitted data. In the header, in addition to the VPI/VCI addresses, we also have, for example, a one-bit CLT (Cell Loss Priority) flag, which fulfils the same role as the DE bit in Frame Relay.



Remark:

We could go on ATM for a long time (especially the architecture of this network is very complex – in addition to layers, there are also planes and several different types of adaptation layers to connect to other protocols), but given that today we do not encounter ATM, it would not make sense. ATM networks were originally considered a very promising technology, but in practice they were gradually replaced by simpler and more flexible MPLS networks.



3.2.4 MPLS – Label Switching

MPLS (MultiProtocol Label Switching) is a network based on *label switching*. The purpose is to move as much routing overhead as possible (including administration, QoS, etc.) to the edge of the network so that the inner area of the network is as fast as possible. MPLS can very quickly transmit not

only ordinary data, but also voice and video, with QoS provisioning. Another advantage is easier implementation of virtual networks.



Remark:


The technology was introduced by Ipsilon Networks under the name *IP Switching* dependent on ATM. Later, Cisco created the proprietary *Tag Switching* standard, which freed the network from ATM dependency. The similar but open standard was later issued by the IETF.



A huge advantage of MPLS is multiformity. It does not directly define addressing or routing, it can cooperate with several different protocols. Originally, MPLS worked as a superstructure over ATM, but now it also works over Ethernet, Frame Relay, SONET/SDH, DWDM and others. So the benefits of MPLS can be summarized as follows:

- very fast switching,
- traffic control assurance (load balancing) and QoS (different treatment of different types of PDUs),
- VPN support,
- ability to cooperate with many different protocols, various technologies can work in the lower layers (under MPLS).

In the ISO/OSI model, we could classify MPLS somewhere between the second (data-link) and third (network) layer, also referred to as layer 2.5 or 2+.

 On an edge router, each packet entering the MPLS network is provided with one or more *MPLS headers* arranged in a *stack*, between the L2 and L3 headers (e.g. between the Ethernet and IP headers). The MPLS header has the following structure:

- label (20 bits),
- QoS information (3 bits), in the case of MPLS we encounter the term CoS (Category or Class of Service) or also Experimental (Exp.),
- end of stack (1 bit, bottom of label stack); if it is not followed by another header, this bit is set to 1 (that is, it is followed directly by the IP packet header),
- TTL (Time to Live, 8 bits) valid for the path over MPLS, may be mapped from the encapsulated IP packet.

The label stored in the MPLS header uniquely identifies the path of the packet on the way to the next MPLS device, in a similar meaning to DLCI for FR or VPI/VCI for ATM.



Example

Figure 3.7 (page 62) shows the ICMP Echo request message encapsulated in an IPv4 packet, which is inside the MPLS frame, which is encapsulated in an Ethernet frame (the screenshot at the top).

This MPLS frame has only one header in the stack (it can have more headers), the label is set to 18 in it, the QoS (experimental) bits are not used (they are set to 0), the bit indicating the last header in the stack is set (bottom of label stack) and the TTL is 254.



¹Figure according to the frame in the CAP file from packetlife.net

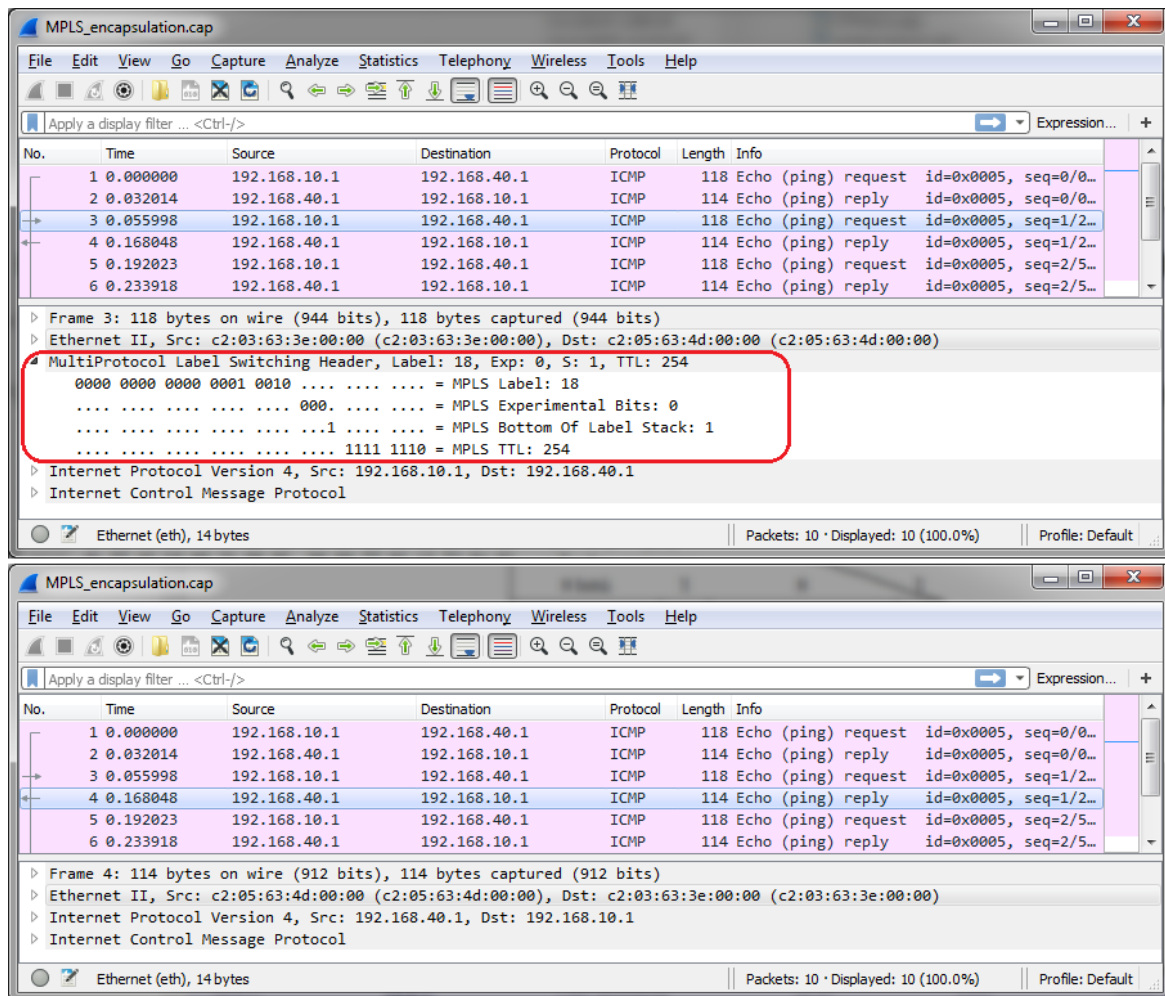




Figure 3.7: At the top a ping request sent over an MPLS link, at the bottom a response received over a regular Ethernet link¹

 **Task**

At <https://packetlife.net/>, find the CAP file shown in Figure 3.7 and examine all the MPLS header fields of some PDU.



 Therefore, a MPLS frame can have more than one header in the stack. The *outer header* on the stack is for specifying paths. It doesn't offer much more than IP headers (with one important difference – it's faster to process). Other headers deeper in the stack have a slightly different, specific purpose, such as VPN headers to specify a virtual network, QoS headers, or traffic control headers.

L2 frame header	MPLS header 1, ES bit = 0	MPLS header 2, ES bit = 0	MPLS header 3, ES bit = 1	L3 packet header (e.g. IP)	Packet payload
-----------------	---------------------------	---------------------------	---------------------------	----------------------------	----------------

Table 3.1: Stack of MPLS headers

✂ Labels are assigned (in all headers of the stack) by sorting PDUs into *classes* (FEC – Forward Equivalence Class), PDUs belonging to the same class are sent with the same label. The class is determined on the basis of several criteria – mainly according to the IP address prefix and, for example, according to some characteristics of the VPN.

📎 **MPLS switching.** Routers (rather switches by their operation) in an MPLS network are called *LSR* (*Label Switching Router*). LSRs at the edge of the network (i.e. also communicating with nodes in local networks that are interconnected by the MPLS network) are referred to as *ELSR* (*Edge LSR*), and are either inbound (*Ingress ELSR*) or outbound (*Egress ELSR*) depending on the direction of operation.

Cisco uses different terminology – LSRs are called a provider/core router, ELSRs are called a provider edge router (but many documents use the “original” terminology).

The third used terminology is the following:

- P (Provider) – MPLS core devices, always owned by the provider,
- PE (Provider Edge) – the same as ELSR, owned by provider,
- CE (Customer Edge) – the same as ELSR, owned by customer or leased from provider (communicates with PE).

Figure 3.8 shows a usual structure of an MPLS network with nodes P, PE and CE. The CE is part of the customer’s local network, multiple CEs (i.e. multiple LANs) can be connected to one PE.

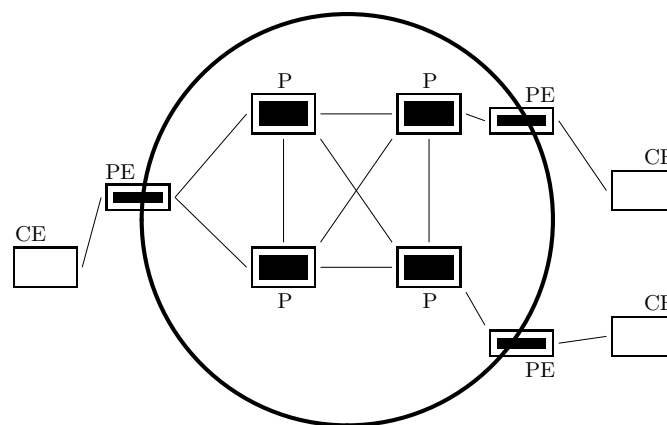


Figure 3.8: Structure of MPLS network

📎 Routing (label switching) is similar to ATM, but even easier. Routers maintain a simple *table* (also called the *Label Forwarding Database* – LFD), which is similar to the ATM table, but instead of a VPI/VCI pair, there is only the label value (a packet from the PPP port marked by the label XXX is provided with the label YYY and sent to the QQQ port, etc.). It is just a matter of determining the relation between the incoming and outgoing mark, the paths are one-way.

Figure 3.9 shows a sample table at LSR within the network. If it were an input Ingress ELSR, the values from the column with the input tag and the port would not be used, similarly for the Egress ELSR the values from the column with the output tag and the port would not be used.

📎 The label stored in the MPLS header changes when passing through the router, depending on how it was set when the connection (circuit) was established, according to the contents of the tables on the passing LSRs. This process is called *Label Swapping*.

	In port	In label	Prefix addresses	Out label	Out port	
■	A	29	128.95.0.0/16	2	B	■
■	A	42	128.101.0.0/16	8	C	■
■	B	18	141.62.0.0/16	36	C	■
	⋮					

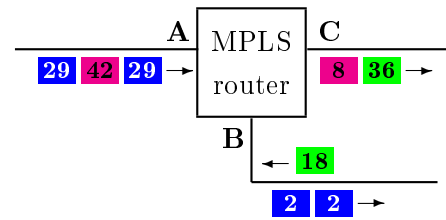


Figure 3.9: Example of a simple MPLS switching table

The specific content of the table on the LSR also depends on the protocol with which MPLS cooperates (i.e. on the network over which it works). For example, if MPLS works above ATM, the VPI/VCI numbers from ATM can be used as MPLS labels.

Making connections. The *Label Distribution Protocol (LDP)* is used to exchange label information among routers; the exchange of address prefix information is carried out according to some routing protocol generally referred to by the abbreviation IGP (Interior Gateway Protocol), very often OSPF. However, in MPLS we also encounter other protocols, for example BGP is used in the implementation of virtual networks (VPN). We will learn about routing protocols later.

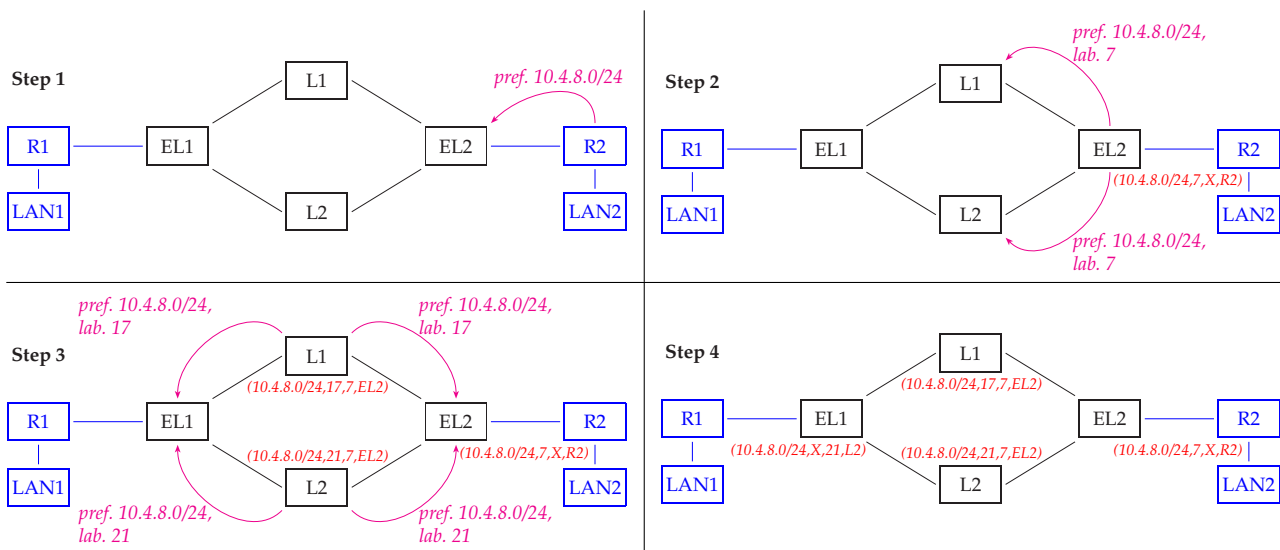


Figure 3.10: Making connection over MPLS for a new LAN

The procedure for adding records to tables in the basic variant is shown in Figure 3.10. The notification takes place in the “opposite direction” of the future path, each node for the incoming request determines the number (label) that it has currently free, and sends the request to all neighboring nodes (even to the one from which the request came). For example, according to Figure 3.10, the *egress* ELSR router is EL2:

- EL2 connects the LAN router R2, the address prefix is 10.4.8.0/24, assigns the label 7, adds a record to the table that whatever comes with this label and destination prefix, has to be sent to the router R2,
- EL2 sends the information “the path to the network with the prefix 10.4.8.0/24 can be found using the label 7” to the neighboring MPLS routers (L1 and L2),

- L1 receives this information, finds a free label 17, adds a record to the table that whatever comes with the label 17 and the given prefix, changes the label to 7 and sends it to router EL2, then L1 sends information to the routers EL1 and EL2,
- similarly L2, the free label 21 found,
- EL1 receives information from two different directions, but the information by L2 came earlier, so it stores the output label 21.



Task

Look at Figure 3.10. What would happen if a local network LAN3 of the same customer was connected to the R1 router, with the network address 10.4.12.0/24?



GMPLS (Generalized MPLS, also G-MPLS) is an MPLS extension that generalizes MPLS to other ISO/OSI layers and other interfaces. The purpose is to eliminate as many “communication intermediate steps” as possible and to enable the deployment of MPLS technology even over such solutions, which were not possible before (not only over solutions using packets and frames). The label does not have to be just a number, but it can be, for example, the wavelength in an optical network, the physical port, the time slot in the TDM, etc., anything that can be used to extend various communications, so MPLS can be deployed over DWDM.

ASON (Automatically Switched Optical Network) networks have similar features to GMPLS, they also run over optical paths or SDH, they also include support for QoS, VPN and fast transport.



Additional information:

- https://www.juniper.net/documentation/en_US/junos/topics/topic-map/mpls-overview.html
- <http://www.ietf.org/dyn/wg/charter/mpls-charter.html>
- http://www.cisco.com/en/US/products/ps6557/prod_white_papers_list.html
- <http://www.faqs.org/rfcs/rfc6002.html>
- FARREL, A., BRYSKIN, I. *GMPLS: Architecture and Applications*. San Francisco, Elsevier, 2006. Most of pages available at <http://books.google.com/books?id=cCWS75MIUpC&printsec=frontcover>



3.3 Optical Networks Infrastructure

3.3.1 SONET/SDH

SONET (Synchronous Optical Network, ANSI T.105) is a digital optical transmission system using time division multiplexing, centrally synchronized by an atomic clock. It is standardized and used in the USA, Canada and Japan.

SDH (Synchronous Digital Hierarchy, ITU-T G.707) is similar to the SONET network in the rest of the world, the differences are small (including in control protocols). We often meet the designation SONET/SDH, referring to the technology that both solutions have in common.

SONET/SDH was created for voice communication and is also optimized for this purpose, it is not effective solution for data transmission. Therefore, another type of WAN/MAN network usually works over SDH (for example, ATM, POS – Packet over SONET, MPLS, EoS – Gigabit Ethernet over SONET/SDH).

A customer is forced to choose from a few types of service according to throughput (2 Mbps, 34 Mbps, 130 Mbps), he also pays for this throughput, even if he is not currently using it (the band cannot be rented to another customer).

Like some previous WANs, a constant-length PDU is used here – a *block* is 810 octets long. It works in time division multiplexing, i.e. the transmitted blocks of data are placed in free *slots* transmitted at regular intervals of 125 μ s.




Additional information:


<http://electrosofts.com/sonet/>




3.3.2 WDM

 **Wavelength Division Multiplexing (WDM)** so called “normal” WDM, is a technology for multiplexing light in a fiber-optic to wavelengths, the first solutions are from the 70s of the 20th century. Without the use of WDM, it is possible to carry only one signal over one fiber-optic, with the use of WDM multiple signals simultaneously at different wavelengths, at each wavelength according to a different protocol and at a different speed as well.

WDM works at the physical layer (L1) and uses two wavelengths 1310 and 1550 on one fiber.

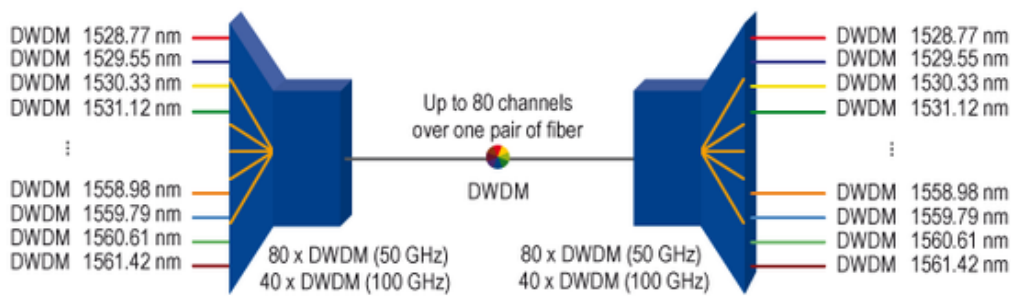
 **Coarse WDM (CWDM)** standardized as the ITU-T standard G.694.2 and also newer standards G.652.C and G.652.D, is an optical fiber multiplexing solution using relatively wide channels (20 nm wide). The typical range is in the tens of kilometers and combines up to 16 different wavelengths into one signal (depending on the specific standard), which is sufficient for metropolitan networks and especially quite cheap.

Today, CWDM is used, for example, in the networks of cable television providers (both television and data signals are transmitted), in Ethernet we encounter CWDM on the physical layer in 10GBase-LX4, also in FTTx networks (Fiber to the Node/Building/Home/...) or in the industry when networking large sites with backbones.

 **Dense Wavelength Division Multiplexing (DWDM)** allows to transmit channels at wavelengths with a very small spacing, even below 1 nm. The smaller the wavelength spacing, the more channels can be transmitted in parallel on one side, but on the other hand the transmission quality may deteriorate, so it is usually not fully utilized. Up to 128 channels can be stacked into one optical fiber, but in busy data backbone networks we tend to encounter 32 channels in one fiber. The number of channels also depends on the optical multiplexers used (which perform the actual mapping of the signal to the channel and back).

The common speed per channel (one wavelength) is approximately 10 Gbps (or 40 Gbps), and the total throughput of one fiber is (almost) the sum of the transmissions of the individual channels.

²Resource: <https://www.pandacomdirekt.com/en/technologies/wdm/what-is-dwdm.html>

Figure 3.11: Signal composing in DWDM²

Additional technologies are implemented over DWDM (between DWDM and TCP/IP) to speed up and generally streamline transmission, currently mainly IP/MPLS and at the edges of a large network, for example, EoMPLS (Ethernet over MPLS), which works very well with connected Ethernet LANs.

OTN (Optical Transport Network) combines the advantages of SONET/SDH and DWDM. SDH takes advantage of transparency, advanced management (including monitoring, error detection) and the ability to create point-to-multipoint links, DWDM has the effective multiplexing of a large number of original signals into an optical fiber.

OTN can work under Ethernet, MPLS, IP, etc.

There are several different physical layer solutions for implementation, differing in throughput and which upper layer protocols need to work with. For example, when working with Ethernet, there are various solutions for 1000Base-X, 10GBase-W, 40Gb Ethernet, 100Gb Ethernet and more. The principle is similar to assembling links in SONET/SDH, for example ODU4 (for 100Gb Ethernet) can be composed of up to two ODU3 or ten ODU2 or forty ODU1 or eighty ODU0 modules.

Task

Devices that multiplex a signal into a color spectrum for transmission over optics (and vice versa) are called optical multiplexers.

Find a CWDM multiplexer on the web (for example CWDM 8-channel multiplexer by Fiber24 or FS, see Figure 3.12) and check its parameters. Note that if you have chosen a really pure optical device, then it is a passive device without the need for power.

Figure 3.12: Examples of CWDM 8-channel multiplexer³

Devices sending and receiving data to/from these multiplexers need optical converters, mainly with replaceable SFP/SFP+/XFP/... modules. Look for a suitable converter on the same site where you found the multiplexer.



³Resources: <https://shop.fiber24.net>, <https://www.fs.com/>


**Additional information:**

- https://www.cisco.com/c/dam/global/de_at/assets/docs/dwdm.pdf
- <http://www.cables-solutions.com/capacity-expansion-and-flexibility-dwdm-network.html>
- <https://www.optcore.net/introduction-to-dwdm/>
- <https://www.itu.int/itu-t/recommendations/rec.aspx?rec=13076>
- <https://pon.fibrain.com/artykuly-techniczne/optical-transmission-lexicon-catv-cwdm-dwdm-ftth,44.html>
- <http://www.fiber-optic-cable-sale.com/dwdm-vs-otn-whats-difference.html>




3.4 Telecommunications Network

A telecommunications network is a network consisting of interconnected devices transmitting (any) types of messages. We include both networks of telephone operators (including mobile) and computer networks, especially MAN and WAN networks.

 The typical architecture of a telecommunications network is based on planes (which intersect various ISO/OSI layers):


- the *data plane* (user plane) – this plane is visible for users, it carries users’ traffic as a payload,
- the *control plane* (signalling) – to transmit control information,
- the *management plane* – for the traffic for the network management, administration.

To better understand the following text, we will explain a few terms related to the telecommunications network.

 The *Public Switched Telephone Network (PSTN)* – a common telephone network designed for voice transmission, originally purely analog with frequency multiplexing, after digitization, time multiplexing with communication band reservation is used. Digitization mainly concerns telecommunication exchanges, part of the connection at the customer is analog.


After digitization, it was necessary to digitize the (in principle analog) voice in telephone exchanges, some CODEC was used (COder-DECoder). A similar principle is now used in VoIP.


The structure of a PSTN is based on telephone exchanges organized in a hierarchical structure. The exchanges at the bottom of the hierarchy are in the “street cabinets”.


 The *Plain Old Telephone Service (POTS)* is a technology that uses PSTN to transmit analog voice and digital data. Because the PSTN is initially analog and only digital from exchange, the digital data must be modulated into an analog signal, MODEM (MODulator-DEModulator) is used.

In the telephone network, the following applies: for the transmission of voice, frequencies in the range of 300–3400 Hz are fully sufficient; trimming the surrounding frequencies practically does not affect the quality of call. It follows that higher frequencies can be used for data transmission in a telecommunication cable.


According to Shannon’s theorem, we can increase the transmission speed (or range) either by extending the frequency band (upwards) or by increasing the signal quality – we will probably not exchange cables, but there are various ways to influence the noise. In any case, the first option is more feasible for data transmission.

 The *modem* is a device that modulates a digital signal to analog and demodulates it back to digital form at the destination. Modems used to be analog, nowadays we use digital modems (for ADSL, VDSL, etc.) – the telecommunication line on the customer’s side is still analog.

 The *Integrated Services Digital Network (ISDN)* was the first attempt to digitize access networks, but a little halfway. It was a dial-up connection (tariffing according to time), because the network management built on the original POTS, in fact it was just a digital POTS superstructure.


 The *Digital Subscriber Line (xDSL)* is a group of technologies that have got rid of most of the POTS components, only the connection between the subscriber line and the nearest telecom exchange is used from the telecommunications network. Then the communication path turns into the provider’s fully digital computer WAN network.

3.5 WAN Access Networks


 ISPs operate their own WAN network, through which they transmit their customers’ data, but this data needs to somehow get into the WAN. Few customers have their network close to a WAN edge router, so it is necessary to use an intermediate network to bridge the distance to the customer’s network. These networks are commonly referred to as *first-mile networks* because they form the “first mile” in the path of data from the customer to the Internet.

3.5.1 PPP Protocol

The PPP protocol is one of the data-link protocols used, inter alia, in the first-mile networks, and unlike others it is more universal (in terms of cooperation with other protocols) and simpler (in terms of header fields). Nowadays, we encounter PPP and its derivatives mainly in first-mile networks.


 The Point-to-Point Protocol (PPP) is a point-to-point protocol, as the name suggests, for duplex links. It does not distinguish between primary and secondary nodes, and it supports the both synchronous and asynchronous transmission. PPP performs automatic configuration when establishing connection and testing the quality of connection. The important property is a customer authentication.

There is a field in the frame header to determine the upper layer protocol, so it can not only encapsulate IP packets, but also other types of PDUs.

 The PPP specification consists of two parts (more precisely – it defines two sublayers):

- *Network Control Protocols (NCP)* – set of protocols providing communication with the upper layer; there are several NCP protocols for different network architectures (TCP/IPv4, TCP/IPv6, NetWare, AppleTalk, etc.), this layer partially extends into the network layer in ISO/OSI,
- *Link Control Protocol (LCP)* – establishing and maintaining a connection, agreeing on configuration at the beginning of the connection (negotiation), connection testing and authentication.


The purpose is to limit the number of functions of the own data-link protocol (i.e. LCP) as much as possible and thus make its operation more efficient.

 *Authentication* is an optional feature, it can be performed by any of the following protocols:

- *PAP* (Password Authentication Protocol) – the text password is sent over the link, so it is not a secure authentication method,

- *CHAP* (Challenge Handshake Authentication Protocol, RFC 1994) – an asymmetric cipher (key) is used, the initiating station sends a request, receives a randomly generated sequence of characters, which it processes using a secret key and sends back, after verification by the recipient, communication can begin,
- *EAP* (Extensible Authentication Protocol, RFC 2284) – completely separates the phase of establishing the connection from the authentication phase.

PPP also supports encryption and compression.

 The PPP frame is largely similar to the frames of other data-link protocols. The data frame has this format:

- Flag (01111110 in binary),
- Address (1 B) – this field is in the PPP header, but its value is constant (eight ones in binary),
- Control field (1 B) – the value is always 0000011 (unnumbered frame, connectionless service),
- Protocol (2 B) – identification of an upper layer protocol (e.g. 0×0021 for IPv4, 0×0057 for IPv6, 0×002B for IPX), can be found in the RFC documents,
- Data – at most 1500 B,
- FCS,
- Flag.


Task

Look at <https://aticleworld.com/difference-between-hdlc-and-ppp/>. This article compares HDLC and PPP. Find the table at the end of the article and review the differences between these protocols.



 Service LCP packets are also used, which need to be encapsulated in a PPP frame as follows:

- the Protocol field holds the value 0×C021,
- the Data field contains the encapsulated LCP packet with the following information:
 - Code – identifies the function of the LCP packet (requests to change parameters, acknowledgment receiving of change requests, rejection of requests, termination of connection, etc.),
 - Identifier – for the command packet it is a randomly generated number, for the response it is the same value as in the packete with the command to response,
 - Length of the LCP packet in octets,
 - data – usually a sequence of ordered pairs (option, value), e.g. type of authentication protocol, maximal data length, etc., or for example in case of frame rejection type there is a part of the rejected LCP frame.

 **PPP Extensions.** Essentially the same applies to PPP extensions as for PPP (L2 layer, approximately the purpose, encryption, usually the frame format). In addition, these extensions are adapted to the specific usage.

Multilink PPP (MP, MPPP, MLP) (PPP over multiple connections, RFC 1990) is a variant of PPP for multipoint connections. This does not mean multiple nodes, but rather multiple serial connections for a single communication. It is possible to combine several serial connections into a single logical

channel with higher throughput (aggregation), the protocol can divide the data, properly rearrange and then restore the original state.

Tunneling PPP (PPTP) (Point-to-point Tunneling Protocol, RFC 1171) is intended for virtual private networks and for connecting a remote client to a corporate network (generally a LAN). However, nowadays it is supported only in Windows and its support will be limited here as well, it is recommended to switch to its successor *L2TP* (Layer 2 Tunneling Protocol, RFC 3931).

L2TP encapsulates PPP frames. By itself, this protocol does not have security features, so it is often combined with IPsec or other methods to secure the connection.

Internet connection is solved by connecting PPP (connection initiation phase) to WAN protocols. It is also used in xDSL technologies for connecting to provider's WAN, in the following variants:

- *PPPoA* (PPP over ATM, RFC 2364) – PPP packet should be sent to the ATM WAN,
- *PPPoE* (PPP over Ethernet, RFC 2516) – PPP packet should be sent over Ethernet.

These variants allow to establish an authenticated connection with the ISP and dynamically obtain an IP address.



Additional information:

- http://www.tcpipguide.com/free/t_PointtoPointProtocolPPP.htm
- <https://www.ciscopress.com/articles/article.asp?p=2202412&seqNum=5>
- <https://www.geeksforgeeks.org/point-to-point-protocol-ppp-frame-format/>




3.5.2 Access Telecommunication Networks

There are various technologies for access networks (*first-mile* networks or *last-mile* networks, depending on the direction of communication) – these can be wireless networks (such as WiMAX) or telecommunication networks (usually ADSL/VDSL or similar technologies), coax links by cable TV operators, optical access networks (FTTx), in some cases an access network even appears as a special part of the WAN network (mobile networks, e.g. LTE).

ISPs use access networks not only to connect customers, but also to authenticate them and control the flow of data.

The specificity of the access network is that it interconnects very different technologies due to the method of communication (in LAN networks we typically have packet data transmission, while in WAN networks we communicate over circuits), and also due to the physical layer and signal handling.

 **Asynchronous Digital Subscriber Line (ADSL).** *xDSL* (Digital Subscriber Line) is a group of broadband technologies that unites several different technologies (ADSL, SDSL, HDSL, HDSL-2, G.SHDL, IDSL, VDSL). This is a dedicated service, point-to-point. It is a connection-oriented service, but tariffication is not time-dependent.

ADSL (Asymmetric Digital Subscriber Line, ANSI T1.413) is an asymmetric service: the downstream (downloading data) is faster than upstream (sending data to network).


Theoretically, speeds of up to 24 Mbps can be achieved (even up to 48 Mbps in the newer standard). The speed offered by us is 8 Mbps or 16 Mbps *for downstream*, but the real speed may be lower. Upstream is usually at a speed of 1 Mbps. Speed is affected by several different criteria. It is not

only the equipment used and the transmission protocols, but also the length of the link the greater the distance to the local exchange, the slower the connection.

Frequencies	Bandwidth	Purpose
0 kHz –4 kHz	4 kHz	voice transfer (analog phones)
4 kHz –26 kHz	22 kHz	pause
26 kHz –138 kHz	112 kHz	upstream
138 kHz –1100 kHz	962 kHz	downstream

Table 3.2: Use of transmission band in ADSL


The modulation method for ADSL is DMT. *DMT (Discrete MultiTone)* divides the whole band into channels of approximately 4 kHz (a total of around 255 channels), the channels 7–32 for the upstream, from 33 till maximum for the downstream. The quality of transmission in individual channels is continuously monitored, the transmission is adaptively decomposed into those channels that are considered to be of the highest quality. Multiple carrier frequencies are used (these are the tones in the name, also called Multicarrier Modulation).


 In xDSL technologies, the increase in speed is achieved, among other things, by diverting the data flow from telephone lines to provider’s data network (usually a WAN network on optical cables). The principle is indicated in Figure 3.13.

Aggregation means ADSL connection sharing. Each ISP distributes the line capacity at its disposal in the form of a time division multiplex. The formula for the aggregation ratio is as follows:

$$\text{aggregation ratio} = \frac{\text{speed (ISP)}}{\sum_i \text{speed (i)}}$$

Typical aggregation values can be e.g. 1 : 50, 1 : 20.

 There are two annexes of the ADSL standard representing three “branches” of ADSL, and several versions. The branches (variants) are Annex A (over POTS), Annex B (over ISDN, digital lines), and ADSL Lite. In the Czech Republic, only Annex B can be used. The newest version is ADSL2++ (ITU G.992.5) with the theoretical throughput up to 48 Mbps.

 *Devices in the ADSL network:* If we do not count the terminal devices in the customer’s LAN (which in fact does not belong to the ADSL network at all), the following devices are used in ADSL:

- *ADSL modem* (MODulator/DEModulator) – modulates the outgoing digital signal to analog (width approx. 1.1 MHz) and demodulates the incoming signal,
- *splitter* – combines analog voice transmission and modulated data into one stream and vice versa, modem and voice phones (analog) are connected to it, it is hardly used today (if we do not have an analog phone, then it is unnecessary; in addition, the splitter can a bit “corrupt” the signal),
- *DSLAM* (DSL Access Multiplexer) – on the ISP’s side, combines connections from ISP splitters/modems for individual connections.

In SOHO (Small Office, Home Office), the ADSL modem is often just a module in a more complex device, for example there are common ADSL Wi-fi routers (i.e. ADSL modem with Wi-fi router plus

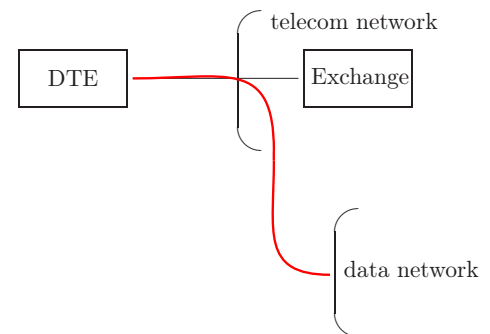


Figure 3.13: ADSL design in respect to user

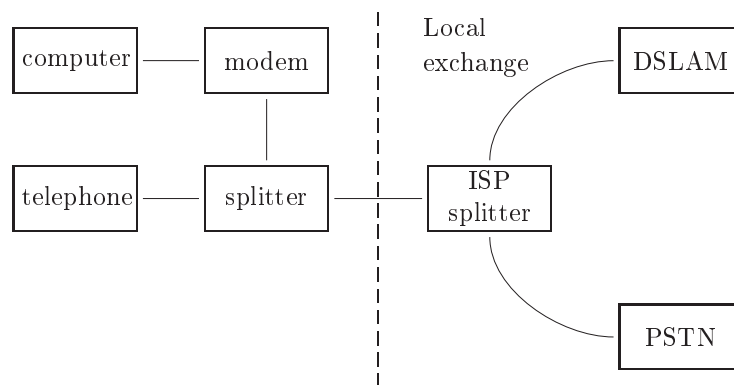


Figure 3.14: Devices in the ADSL network

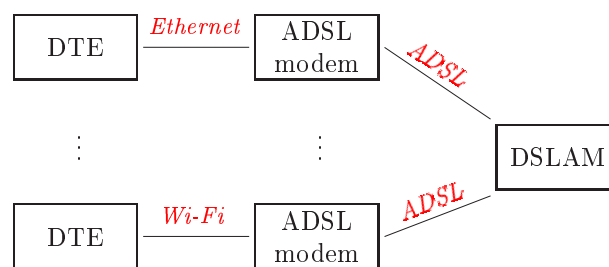



Figure 3.15: ADSL is one of the first-mile networks


other functionality: hardware firewall, DHCP server, etc.). It connects to the subscriber line via RJ-11 (telephone cable). Its functions are both establishing a connection (usually when the device is switched on or reset) and providing the communication itself. It encapsulates packets into PPPoE, PPPoA or IPoA PDUs (nowadays, the first one is the most common).


 *DSLAM* is the gateway to the provider's data network. There are smaller DSLAMs for connecting 24 or 48 xDSL links (and usually the same number for POTS – analog telephone), and then larger for thousands of links. It usually supports both PPPoE and PPPoA protocols.

The DSLAM has an RJ-11 interface towards the customer and then another interface – usually two RJ-45s for Gigabit Ethernet or faster, or rather optics. The administration is performed via web interface or via console, depending on the specific manufacturer.

Figure 3.16 shows a simplified diagram of the network structure on the ISP side. The terms for this figure:

- *PTA* is a broadband server, for setting and storing IP addresses configuration, user authentication, authorization, accounting (e.g. RADIUS or other auth server),
- *AP (Access Point)* is an access point for the particular ISP; it serves to access an Internet router from the ISP virtual network (ISP WAN).

 We can also find the designation *MSAN* (Multi-Service Access Node). MSAN is a device installed in a street cabinet, which provides several different services (various DSL, VoIP, FTTx, Ethernet, etc.), so it is more universal than DSLAM.

 **Very High Bit-Rate Digital Subscriber Line (VDSL).** VDSL is considered a successor to ADSL. It is also asymmetric (can be set to symmetric transmission), uses a wider band than ADSL

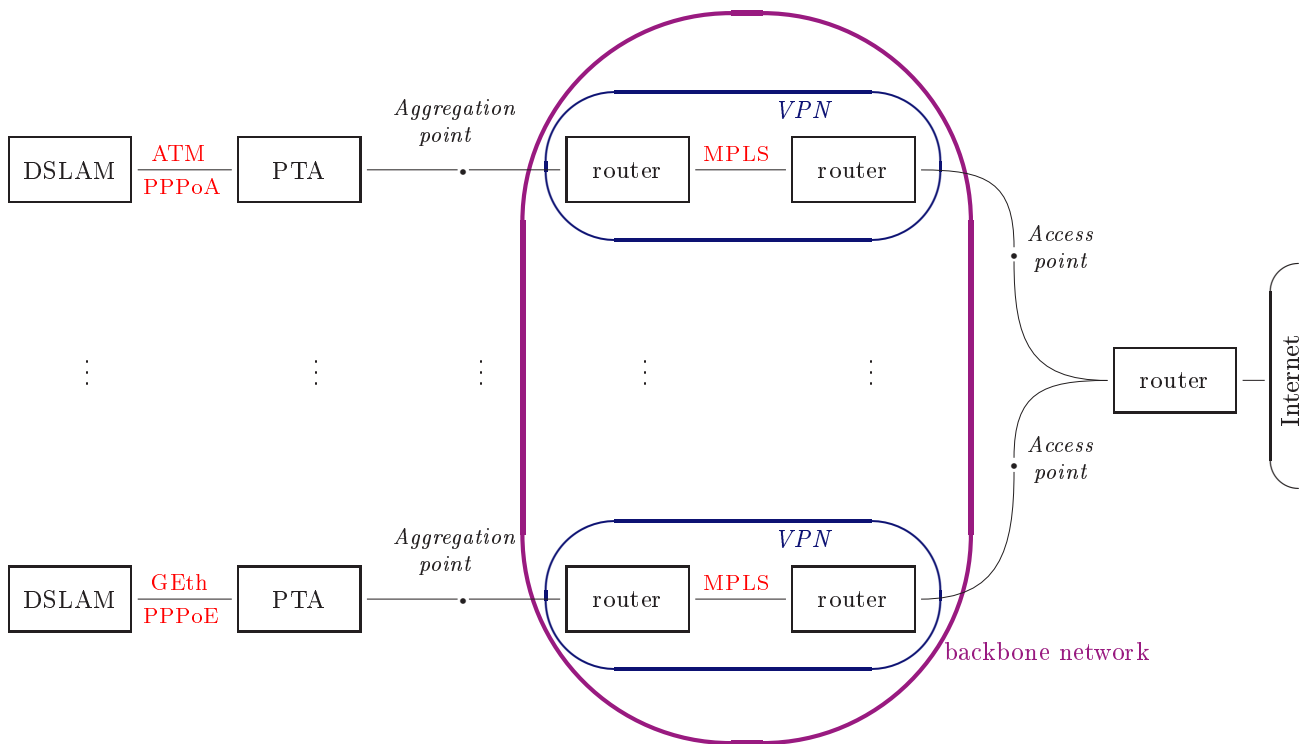


Figure 3.16: ADSL – ISP's side

(stretching to higher frequencies), and is therefore faster. The speed reaches up to 52 Mbps (asymmetric, downstream) or 36 Mbps (symmetric variant). Theoretically up to 200 Mbps, in reality only cca 50 Mbps.

Figure 3.17 shows that, compared to ADSL technology, VDSL uses a wider frequency band, so there are more communication channels. While ADSL is upstream at lower frequencies and downstream at higher frequencies, in VDSL the channels for both directions are distributed quite evenly across the spectrum.

While ADSL transmits over a distance of up to 5 km (less at higher speeds), VDSL only cca 1 km, which is caused by transmission at higher frequencies. However, if we really want to take advantage of the high speeds offered by VDSL, then we should be a maximum of 500 m from a DSLAM. Upstream and downstream are separated by frequency multiplexing similar to ADSL.

VDSL version 1 (ITU G.993.1) allows QAM or DMT multiplexing, VDSL version 2 (ITU G.993.2) uses only DMT. VDSL version 2 offers speeds of up to 100 Mbps in both directions at the same time, but only with maximal distance 300 m. Typical speeds are around 40 Mbps, but can be improved with other assistive technologies, such as vectorization.



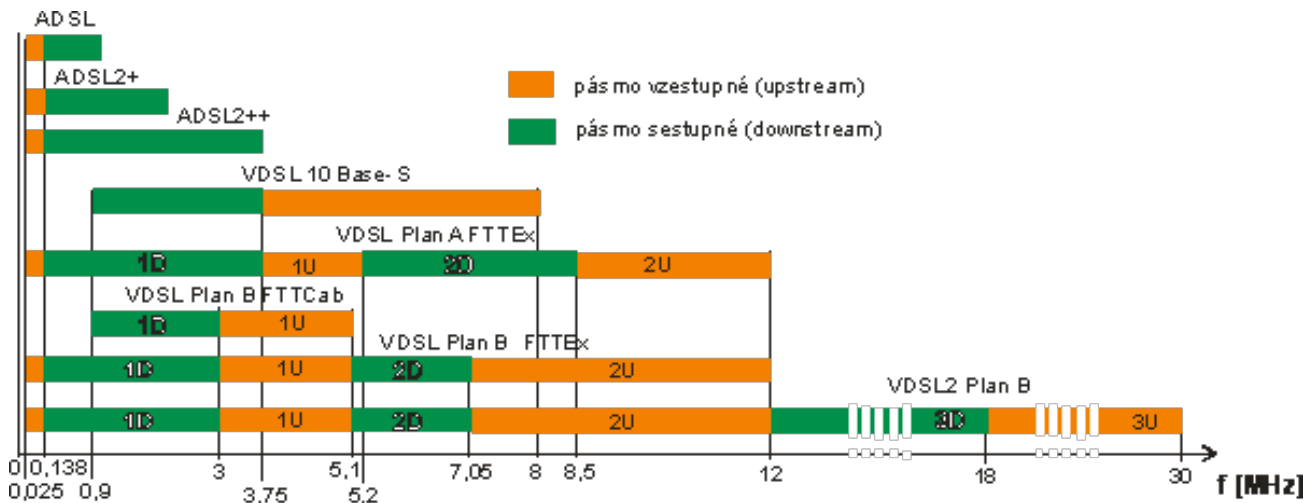
Additional information:

https://www.juniper.net/documentation/en_US/junos/topics/topic-map/security-interface-configuring-vdsl2.html



Task

⁴Resource: <http://access.feld.cvut.cz/view.php?cisloclanku=2004120302>

Figure 3.17: Comparison of ADSL and VDSL frequencies⁴

Find any medium-sized DSLAM or MSAN on the Internet. If you can't select, choose for example the MSAN "Zyxel 2U 5-slot Temperature-Hardened MSAN IES4105 Series". Find out the parameters of this product.



Other xDSL technologies. In addition to ADSL and VDSL, there exist other xDSL technologies. They differ in the number of RJ-11 interfaces they require, (a)symmetry and other parameters.

High bit-rate Digital Subscriber Line (HDSL) is not intended for subscriber connections, it is used by ISPs to connect PBXs (branch exchanges in companies), or it can be used by large companies for their internal needs. It requires two or three pairs of telephone wires, which is considered one of the disadvantages.

It is a symmetric technology (same range of bands for downstream and upstream), but it can also work asymmetrically. It only supports data transfer, it does not support phone calls.

HDSL-2 (also SHDSL, SinglePair HDSL) uses only one pair of telephone wires, so when implementing HDSL-2 technology on the telephone line itself, there is no need to change anything.

Symmetric Digital Subscriber Line (SDSL) is a range of functions similar to ADSL technology, but it is symmetrical (same range of bands – number of channels - for upstream and downstream), does not support telephone calls (only for data, analog phoned cannot be connected). The transmission principle is similar to HDSL.

It is used for subscriber connections where transmission symmetry is preferred. Usual speed is up to 2.3 Mbps, range up to 5 km.

3.5.3 DOCSIS


Data Over Cable Service Interface Specification (DOCSIS) is the standard for the "cable internet", it permits a broadband connection over existing coaxial cable TV networks (CATV – Community Antenna TV). The European variant is EuroDOCSIS. DOCSIS version 3 is standardized as ITU-T J.222.


At present, the infrastructure is hybrid – a combination of optical links (provider's network) and coaxial (which leads to customers, due to backward compatibility and price). This principle is called

hybrid fiber-coaxial (HFC) infrastructure.

DOCSIS is asymmetric, similar to ADSL/VDSL. Typical speeds are higher than xDSL, DOCSIS version 3.1 from 2013 (D3.1) and version 4 from 2017 (D4) can transmit data at speeds up to 10 Gbps (downstream), upstream is slower.


At the physical layer, QAM modulation is mostly used (depending on the quality of the connection, it can be 4096-QAM, but it is usually 256-QAM), or others depending on the type of traffic, including DVB-C. The channels are combined by OFDM or OFDMA.

 Just as xDSL technologies use DSLAM, DOCSIS uses *Cable Modem Termination System (CMTS)*. Its role is the same – it is a device in a street cabinet, into which connections from customers lead.

 **Additional information:**

- https://www.rohde-schwarz.com/es/tecnologias/tv-por-cable/docsis/tecnologia-docsis/tecnologia-docsis_55513.html
- <https://community.cisco.com/t5/networking-documents/docsis/ta-p/3115673>
- https://www.ieee802.org/3/efm/public/jul01/presentations/gummalla_1_0701.pdf



 **Task**

Find any DOCSIS CMTS on the Internet. If you can't select, choose for example the CMTS “Casa CMTS/CCAP C40G”. Find out the parameters (technical specification) of this product (for the above mentioned device it is <https://www.casa-systems.com/assets/Casa-Datasheet-C40G.pdf>). What versions of DOCSIS are supported? What type of port is used for management? Is it possible to communicate with it via SSH? Is there any DHCP support? What about downstream interfaces – how many ports per module are supported? What type of connector?



3.5.4 FTTx


Fiber-to-the-x (FTTx) is an optical broadband access technology. The x “variable” means

- FTTN (fiber-to-the-node or neighborhood): fiber-optic cable is terminated in a street cabinet,
- FTTC (fiber-to-the-cabinet or curb): similar to FTTN, but the street cabinet is much closer (at most hundreds of meters),
- FTTB (fiber-to-the-building) – the end of the optics is at the entrance of the building (e.g. in to the ground floor, shared electrical room), from where it is already distributed to individual flats by another technology,
- FTTH (fiber-to-the-home) – the most comfortable, fast and expensive variant.

The first two types are often referred to as FTTP (fiber-to-the-premises).

From the FTTx termination point, either twisted pair or coax or other similar technology leads. The closer this point is to the user, the more advantageous it is for the user (in terms of speed, while the price is the opposite).

The FTTx network is hierarchical, similar to telecommunications (DSL), the paths are divided in *street cabinets*.

 The connections can be implemented as either an *active optical network* (AON) or a *passive optical network* (PON). AON uses the analogy of switches or routers (located in street cabinets) and each customer gets its traffic, not a traffic of another customer. Fully dedicated fibers are possible too, but the price corresponds to that.

PON does not perform switching (at most on some nodes, but not at every division of traffic), communication of individual customers is separated differently. Time division multiplexing (TDMA) is used. This variant is more common, it is cheaper.



Remark:

For the previously mentioned technologies, we mentioned that the service provider has street cabinets connected by an optical network, and from them leads, for example, a VDSL line or a coaxial to customers' flats. The optical network, that's exactly it – FTTN or FTTC.

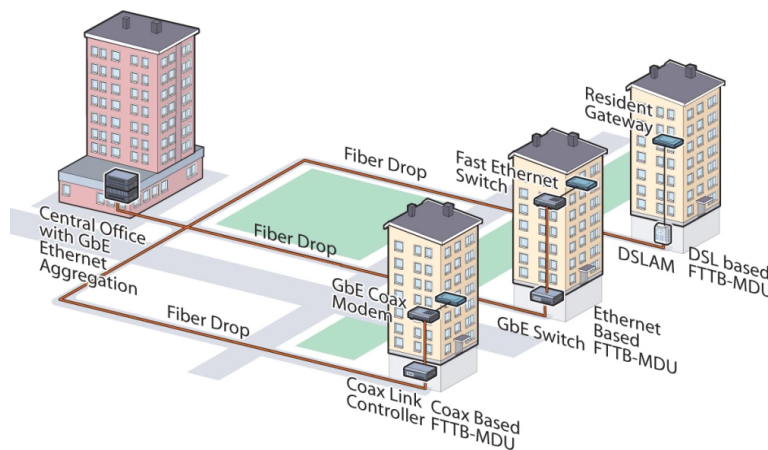



Figure 3.18: Three solutions using FTTx⁵

 The optical network ends in a street cabinet, from where another type of network continues. Here it is usually necessary to convert the optical signal into an electrical signal (for example into a coaxial or twisted pair) and, if necessary, to recode or otherwise modify the signal. The device that handles the signal in this way is usually referred to as *Optical Network Terminal* (ONT, ITU terminology for FTTH and FTTB) or *Optical Network Unit* (ONU, IEEE terminology for other types of FTTx). On the other side of the line (at the service provider's) is the *Optical Line Terminal* (OLT).

FTTx standards are built using other protocols, for example for PON networks:

- GPON (Gigabit PON, ITU-T G.984) can slide under Ethernet, ATM and more,
- EPON, resp. GEAPON (IEEE 802.3ah) works over Ethernet,
- 10G-EPON (IEEE 802.3av) is backward compatible with the previous standard, but faster, uses WDM with TDMA multiplex,
- 10G-PON (ITU-T G.987) is 10Gb evolution of GPON by ITU, and it is backward-compatible with 10G-EPON at the physical layer (it is possible to use the same ONT and OLT devices for

⁵Resource: <https://broadbandworldforum.wordpress.com/2013/09/23/alternative-access-options-in-building-coax-networks-vs-vdsl-for-ftth/>

the both standards).

At present, therefore, Ethernet frames are mostly transmitted in FTTx.

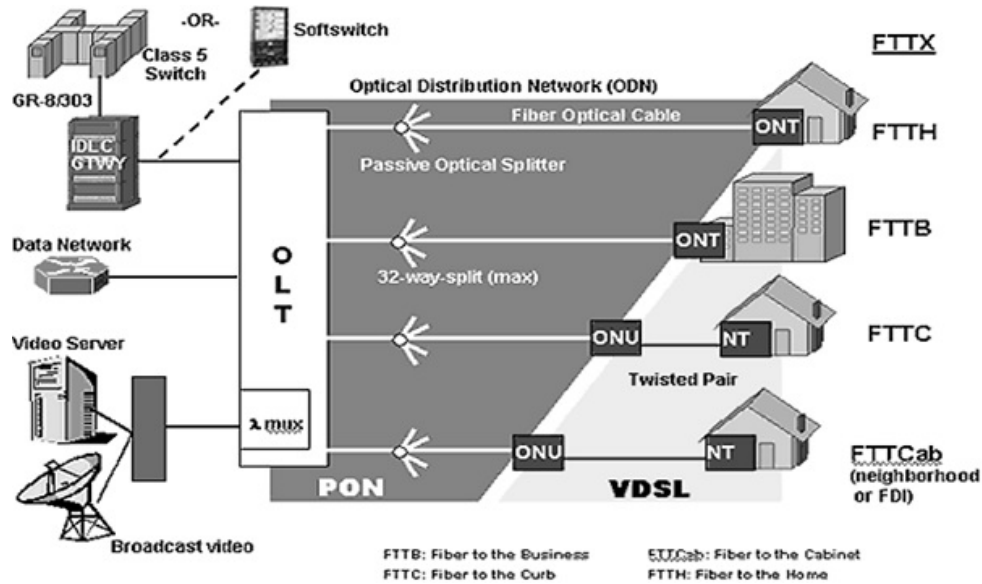


Figure 3.19: FT Tx OLT, ONT and ONU terminations⁶



Additional information:

- https://www.tutorialspoint.com/ftth/ftth_quick_guide.htm
- <https://get.ospinsight.com/fiber-to-the-home-the-ultimate-guide>
- <https://www.linkedin.com/pulse/abc-pon-understanding-olt-onu-ont-odn-gharbi-skander?trk=seokp-post-cta>
- <https://broadbandworldforum.wordpress.com/2013/09/23/alternative-access-options-in-building-coax-networks-vs-vdsl-for-ftth/>



Task


Find some GPON OLT (e.g. from Ubiquiti) with at least 8 GPON ports, examine its parameters. Find some FTTH ONT, examine its parameters.





⁶Resource: https://www.tutorialspoint.com/ftth/ftth_quick_guide.htm

Chapter 4

Routing


 *Preview:* This chapter is devoted to routing. We deal with the principle of routing, the difference between switching, bridging and routing, routing algorithms and the most used routing protocols.

 *Keywords:* Bridging, switching, routing, routing table, autonomous system, administrative distance, routing protocols, distance-vector algorithm, link-state algorithm, path-vector algorithm, RIP, IGRP, EIGRP, OSPF, BGP.

 *Goal:* The aim of this chapter is to understand principles of routing, properties of routing algorithms and common routing protocols operation.

4.1 Bridging, Switching, Routing

First, we will briefly look at a few concepts related to distributed data processing (transmission in this case) in computer networks, while we have dealt with these technologies in previous chapters.


 **Bridging** is the term related to bridges as the network devices standardized as IEEE 802.1, including the STP protocol which is intended to avoid logical loops in L2 networks.

Bridges (generally any device on L2) can work in one of the following modes – in terms of working with frame path:

- *Source-route bridging* – the sending node fixes a list of bridges (generally intermediate L2 devices) through which the frame is to pass (either used to create a circuit equivalent or stored in the frame header).
- *Transparent bridging* – the sender only determines the physical address of the destination, the bridges maintain and dynamically update the switching table, by means of which they determine the direction to the given destination.


It is obvious that transparent bridging is used in Ethernet. Source-route bridging was used, for example, in the Token-Ring network. In source bridging, it is therefore necessary to provide information for the correct determination of addresses and ports, whereas in transparent bridging, there must be a mechanism in the network for transporting physical addresses to bridges.

As we know, this problem is solved very simply in Ethernet – if a frame with an unknown source address is detected on the port, we add this source address to the switching table and the port from which the frame came because somewhere behind this port this device will most likely be connected.

 **Switching** does not relate to a standard, but rather to specific technologies that we find on switches, in fact it is a kind of superstructure of bridging. We distinguish the following types of switching:

- *Store-and-Forward* – an incoming frame is first received and stored in the switch buffer; after the entire frame is received, the header is read and the output port for sending the frame is determined. It can catch damaged frames, but it is slow, suitable for networks with higher error rates or switches with more powerful hardware.
- *Cut-Through* (on-the-fly) – an incoming frame starts to be sent continuously during its reception, immediately after reading the information from the header needed to determine the output port. Very fast switching, less need for cache, but can't catch damaged frames.
- *Fragment Free* – something between the first two methods. Processing of a frame begins after reading its first 64 octets (i.e., the header and at least a portion of data). The purpose is to detect at least one type of damaged frames: too short.

What is better? It depends on network utilization and switch performance. It is usually possible to set the mode on devices intended for the corporate field. Some network functions require a specific mode, for example if we need quality of service (QoS), store-and-forward should be set on the switches.

 **Example**

For Cisco switches, the cut-through mode is usually set as the default for most switching operations, but the “CPU port” (=for internal switching processes, such as STP or VTP) is switched by store-and-forward. Some models allow to change the mode, others do not, it also depends on the license used.

If we want to set the store-and-forward mode, we enter:

```
switch(config)# switching-mode store-forward
```


If we want to set the cut-through back, we precede this command with no:

```
switch(config)# no switching-mode store-forward
```

Certain Cisco switches have a fixed store-and-forward mode and it cannot be changed. If we enter a question mark in the global configuration mode, we get a list of supported commands. If there is the command `switching-mode`, we write it and put a question mark instead of a parameter. So we find out which mode can be set.



In general, bridges and switches send unicast traffic (with a known destination address) to a single port, while broadcast and unicast with an unknown destination address are sent to all ports except the one from which the frame came (we call this situation by *flooding*).

 **Task**

If you have a switch made by Cisco, find out if it allows to change the switching mode (you can find out by using a question mark in the global configuration mode). Or you can try it in the Packet Tracer.



**Additional information:**

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus3000/sw/layer2/503_U3_1/b_Cisco_n3k_Layer_2_Switching_Config_503_U31/b_Cisco_n3k_Layer_2_Switching_Config_503_U31_chapter_01000.pdf



Routing refers to the operation of routers and other devices operating at the L3 layer, including switches with higher layer functionality. We already know what routing tables look like; as part of routing, it is necessary to ensure the distribution of the current content of these tables (or some equivalent information suitable for making the routing content) and use these tables when routing packets.

The purpose is to keep the routing tables up-to-date, and the process of passing the routing tables to a consistent state is called *network convergence*. If dynamic routing is used in the network (i.e. the exchange of routing information using protocols), the process of network convergence is one of the most important parameters of a given protocol – convergence should be fast and should not generate too much traffic.

4.2 How Routing Works

Routing takes place at the L3 layer, and actually means the process of determining the path to specific IP networks. This information is further used to determine the path for a particular IP packet.

**Definition (Routing table)**

At least one routing table is kept on each router and all other devices of the L3 layer, in which we have the following information for each record:

- the (sub)network address with the mask or prefix length,
- the outgoing network interface or the neighbor (its IP address) leading to the given network,
- the type of the record (static, dynamic gained by some protocol, identification of such protocol),
- metrics (quality of the chosen path) and other information.

The routing table also includes information about the gateway (also called “gateway of the last resort”), i.e. determining the direction for everything to send for which we do not have a specified path elsewhere in the routing table.




In a device supporting the both IPv4 and IPv6 protocols there are at least two routing tables, one for each of these protocols.



This table needs to be filled in and further updated according to the current situation (some paths will cease to be valid, or, conversely, another path is made available, metrics are changed or a new subnet is being added). This is either done manually (*static routing*) or left to *routing protocols* (*dynamic routing*).

In practice, both are combined, i.e. we insert some paths manually and let the rest be added and updated dynamically. Static routing data is considered more trusted, so it is usually preferred over data obtained dynamically using protocols.

 *Metric* is a number indicating the quality of a path, it has been calculated according to certain criteria. Each routing protocol uses slightly different criteria and calculates metrics from them in a slightly different way. For example, the following can be used as criteria:

- number of routers at the path (it means the path length),
- path throughput (for example, whether it is Fast Ethernet or Gigabit Ethernet),
- latency (delay in ms),
- path reliability (low error rate, few dropped packets, etc.),
- the path load,
- the maximum MTU value over the path (i.e. the maximum packet size that can be sent),
- the credibility of the source from which the path information was obtained, etc.


Each routing protocol has its own metric. It is not only that it uses different criteria, but also that the metrics of the different protocols are not comparable. The same path is evaluated completely differently by two different protocols.




Remark:

Usually: the smaller the metric number, the better the path. Directly connected addresses have the lowest metric.



 When and how to work with the routing table: if the router receives a packet, it needs to find out to which port this packet should be forwarded. It goes through the routing table and finds out which record in the table corresponds to the packet's destination address, i.e. it determines whether the address belongs to the subnet specified in the given record.

 If there are two corresponding paths in the table (there is a subnet address in two records to which the address from the packet belongs), the one with the longer prefix is usually preferred (it is more accurate, specific). For example, if we have the following subnet addresses:

10.160.0.0/12

10.160.0.0/17

and the compared address corresponds to both, the second target will be preferred. These are two different entries because they differ in the length of the prefix, although the address looks the same in both cases. The second address will probably be the subnet of the network specified by the first address.

If no path is found in the routing table, either the packet is passed to the gateway because the packet appears to belong to another network, or dropped (when no gateway is available). This procedure corresponds to the procedure from the previous paragraph, because the gateway of the last resort is specified by the address 0.0.0.0/0, which is the least specific possible address, and therefore will be used only if no other record from the routing table is selected.



Task

All devices working on the L3 layer have a routing table, including computers. The routing table can be shown by these commands:

- `route print` (Windows device),
- `route` or `ip route show` (Linux, MacOS or other UNIX),

- `show ip route` (Cisco network device),
- `display ip routing-table` (HP network device),...

Try everything you can. The commands on network devices allow to enter other parameters, you can also try them.



Additional information:


- <https://www.geeksforgeeks.org/routing-tables-in-computer-network/>
- <https://www.ciscopress.com/articles/article.asp?p=2180210&seqNum=12>
- https://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/5_x/nx-os/unicast/configuration/guide/l3_cli_nxos/l3_route.html
- https://techhub.hp.com/eginfolib/networking/docs/switches/RA/15-18/5998-8165_ra_2620_mrg/content/ch03s04.html



4.3 Cooperation of Routing Protocols

Various organizations use various routing protocols. Some of them are intended for interior usage inside a network of routers of a single organizations, others are able to provide routing among these relatively homogenous areas.

4.3.1 Autonomous System

 An autonomous system is a group of routers that are managed by the same organization. Each autonomous system is identified by a 16-bit or, more recently, a 32-bit *ASN* number.

- the *interior routing protocols* can only route within their own autonomous system (they send everything “foreign” simply to a single gateway), they do not distinguish between other autonomous systems (they distinguish only “own” and “foreign” network).
- the *exterior routing protocols* provide routing even between different autonomous systems (they can work with the ASN number, the routes are characterized by ASNs of the autonomous systems through which the destination can be reached. They can distinguish between different “foreign” paths.





Remark:

Does each organization have to manage its own autonomous system and its own ASN? No. Smaller companies only need to be part of their ISP’s autonomous system (or of the autonomous system of the ISP of their ISP), and they usually don’t even know its ASN. If the company uses just interior routing protocols, there is no reason to have its own ASN (mainly to pay for it).



4.3.2 Administrative Distance

 A *single-protocol* router can only work with a single routing protocol, a *multiprotocol* router can combine information from multiple routing protocols.

 In addition, a multiprotocol router has to decide between paths from different sources (protocols), which are created according to different criteria and thus they are not comparable with each other. For this reason, an evaluation constant called *administrative distance* (AD) was set for each protocol to compare multiple paths to the same destination from different protocols. This number really means the trustworthiness of the source (how much we trust that this resource provides “good” paths).



Remark:

The smaller the number, the more trustworthy the protocol and path.



Administrative distances of various protocols are in Table 4.1. Note the differences between protocols, which can work as both internal and external (for example, BGP). In fact, AD can be configured on routers.


Origin of path information	Administrative distance
Connected interface	0
Static routing	1
EIGRP summary route	5
External BGP	20
Internal EIGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
EGP	140
ODR (On Demand Routing)	160
External EIGRP	170
NHRP	250
Internal BGP	200
other (unknown)	255

Table 4.1: Default administrative distances¹


If multiple protocols can route to a given destination, the path of the routing protocol that has a smaller administrative distance number will be selected. As we can see, the connected interface is the most trusted, because the router puts them into the routing table itself (it trusts itself the most) and the path is the shortest. Statically added records follow. Internal EIGRP has AD = 90, OSPF has AD = 110, so EIGRP is expected to have a slightly better path determination capability (but on the

¹Resource of information: <https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/15986-admin-distance.html>

other hand, EIGRP is usually supported only in Cisco devices, so it can only be used in a homogeneous network).

 The path leading within the custom managed area (from the point of view of a routing protocol) is *internal*, the path found outside the routing process (for example from another autonomous system or from another protocol) is *external*. Most protocols prefer internal paths, except, for example, BGP, and others do not distinguish their reliability (for example, OSPF).

4.4 Routing Algorithms

 Each routing protocol works according to some *routing algorithm*. This is usually one of the following algorithms (or some modification): Distance-Vector Algorithm and Link-State Algorithm.

4.4.1 Distance-Vector Algorithm


The distance-vector algorithm determines the path metric primarily by distance (in the simpler case, by the number of routers on the path). For each network registered in the routing table, we have the following data: distance and vector. The vector determines the direction to the given network, i.e. either through which network interface or through which neighbor the path leads. The position of the target is determined only by the vector, the devices do not have an overview of the topology of the router network.


A router operating according to this algorithm first has paths to connected networks in the routing table, i.e. paths to direct neighbors, and gradually receives information from them about other routers. If it receives an update packet from its neighbor with information about the (sub)network with a specific metric (according to that neighbor), it will add this (sub)network to the routing table:


- the destination (sub)network address reported by the neighbor,
- the vector – the neighbor's address or the port through which the destination is available,
- as a metric a slightly higher number than the one sent to him by the neighbor is used, in a simpler case the number reported by neighbor is increased by 1 (i.e. the distance between the neighbor and the given device is added to the reported number representing the path between the neighbor and the given network).


It is typical (but not fully) for this algorithm that updates (information about the current state of routing) are sent to neighbors at regular intervals.


As we can see, a problem can arise here as in a switched network, where we had to use the STP protocol to remove the loops. Loops can also occur here (we have a network of routers using a given algorithm/protocol), and the consequences could be similar. For example, imagine a situation where the same network is accessible via two different paths. The router receives network information alternately from two directions, which does not indicate the stability of the routing tables. The stability of the operation is improved in various ways, as followed.

 Using this algorithm, the *maximum number of hops* can be defined, which specifies the maximum network size (in number of routers on the path) – usually 15 or 255. If a router receives network information with a metric higher than this limit (for example, RIP with a value of 16), then considers such a network unavailable.

 The *hold-down timer* is specified as well, usually set to three times the interval for sending updates to neighbors. If a router receives information about the unavailability of a network, then for the time specified by the hold-down timer it ignores the messages about the path to this network, considering them potentially erroneous.

 *Poison reverse* is a notification to the router of its unavailability. The router sends its neighbors an update of the path to itself with the value of the metric $\max(TTL)$ and the neighbors add 1 to it, i.e. for example for RIP it will be the value 16 or for other protocols 256. A metric greater than the maximum specified indicates a “unreachable” path, so no packets will be sent to the port.

 Another way to reduce network load and loop risk is the *split horizon* procedure. A router does not send its entire table to another router, but only those records that do not lead to that router (there is no need to inform another router of what it originally informed). This also reduces the risk of sending incorrect information.

 The protocols using the distance-vector algorithm are the following:

- RIP (Routing Information Protocol),
- IGRP (Interior Gateway Protocol),
- EIGRP, however, this algorithm enriches with some elements typical of the link state algorithm, its metric is considered very good.



Remark:

A typical feature of protocols implementing a distance vector algorithm is that they *know* only their neighbors and have no information about the network topology. They only know that a specific (sub)network can be reached through a specific neighbor and the path takes a specific number of hops.




4.4.2 Link-State Algorithm


The link-state algorithm requires the construction of a *network topology database* (a network map). A router with a protocol according to this algorithm maintains three tables:

- a *neighbors table*, which it creates immediately after switching on and continuously updates,
- a *topology table* representing the topology database of the network, which it creates gradually according to the update information of other routers,
- a *routing table* made according to the topology table.

Each router constantly monitors the quality of connections leading to neighbors (this applies to the first mentioned table). If it detects a change (a connection/neighbor becomes unavailable or changes a parameter), it informs other neighbors. The routers that receive this information record the change in the topology database (the second table) and calculate a new path to the destinations according to it (update the routing table).

It is typical for protocols implementing the link state algorithm that they have an overview of the network topology and the convergence time is significantly shorter (updates are only forwarded, each router calculates the change itself, they do not have to wait for each other).

 The calculation of routes and the determination of the shortest (most optimal) path to a given (sub)network is performed by the *Dijkstra's algorithm* (shortest path first algorithm, SPF) or its modifications. It is one of the most well-known graph algorithms by Dutch computer scientist Edsger Dijkstra.

 **Example**

Let us see how Dijkstra's algorithm works. Each link between routers is evaluated by a number, while a smaller number means a better connection. Each router creates a topology database containing all routers in a given routing domain and for each pair of routers information on whether there is a connection between them and, if so, what is its evaluation. We can imagine a topology database as a table, where rows and columns are marked by individual routers, in the cell for a certain pair of routers there is information about the link between them (or it is blank).

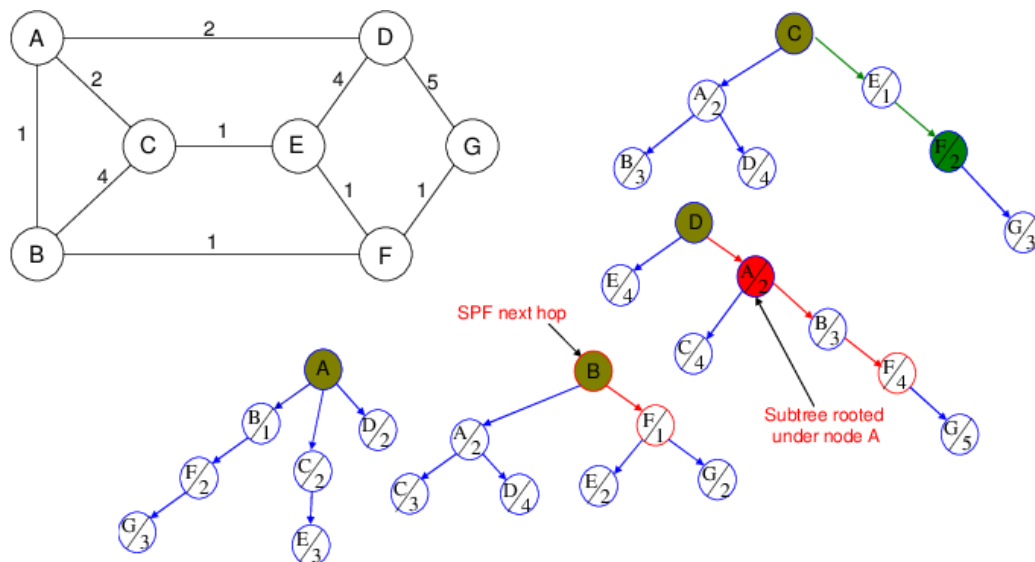


Figure 4.1: Example of use of Dijkstra's algorithm²


Figure 4.1 shows a graph with seven routers (labeled A to G) with an evaluation of the links between them. The topology table could look something like this:

	A	B	C	D	E	F	G
A	0	1	2	2			
B	1	0	4			1	
C	2	4	0		1		
D	2			0	4		5
E			1	4	0	1	
F		1			1	0	1
G				5		1	0


This table should be the same on all routers in the routing domain, so each router has an idea of the network topology. According to the topology table, the path cost is calculated and thus the most suitable path to all network elements, which is the basis of the routing table.

²Resource: https://www.researchgate.net/figure/Example-topology-and-routing-trees-of-node-A-and-its-neighbours_fig1_225143830

The procedure is to transform the original graph (in which we have redundant paths) into a tree – at the root of the tree is the router on which this calculation takes place (i.e. each router has a different tree, placing itself on the top), some links are removed. Only those links that are part of the optimal paths remain.

In Figure 4.1 we can see the trees for the routers A, B, C, D. For example, the router A only left the path through the neighbor B to the router F, while it removed the paths through other neighbors. It left the path $A \rightarrow B \rightarrow F \rightarrow G$ to the router G, the cost of which is $1 + 1 + 1 = 3$, it did not use the path $A \rightarrow D \rightarrow G$, which leads through fewer intermediate nodes, but its price is worse: $2 + 5 = 7$. 

The SPF algorithm is one of the most used graph algorithms.

 The most known link-state protocol is OSPF, and the algorithm is used in the IS-IS protocol too.


 **Remark:**


A typical feature of protocols implementing a link state algorithm is that they have an overview of their network or its essential parts (an autonomous system can be divided into relatively separate *areas*, which are interconnected by border routers, and then it is sufficient for the protocol to have overview of the area in which it is located). Another typical feature is fast convergence and less susceptibility to loops.




4.5 Interior Routing Algorithms

4.5.1 RIP

 RIP (Routing Information Protocol) is one of the oldest routing protocols, developed by Xerox in 1981. It is an internal routing protocol communicating on the UDP port 520. It implements a distance-vector algorithm, the number of routers on the path to the destination is used as a metric.

 *RIPv1* (version 1, RFC 1058) uses address classes (does not support classless routing), meaning that neither the subnet mask nor the prefix length is included in the routing information. This is disadvantage that RIPv1 eliminates from larger networks.

The highest accepted metric is 15, the number 16 already indicates an unavailable network, so in the Poison Reverse mechanism, the number 16 is used to report unavailability. At 30s intervals, routing information (entire routing table) is sent as broadcast, which can greatly decrease network throughput.

 **Example**


The subnetwork addresses

10.10.22.0/24

10.20.10.0/24

would be considered the same network, because the RIP router would treat both addresses as a class A address, where the network is specified by the first octet. If these networks are routed through different routers, routing will not work properly (one entry will be overwritten by the other, one of the networks will alternately not be available).



 *RIPv2* (STD-56, RFC 2453) is an update to the RIP protocol from the mid-1990s. The main task was to take into account the use of prefixes and classless routing (VLSM and CIDR). Differences from RIPv1:

- it allows to use classless routing,
- routing updates are sent as multicast to the multicast address 224.0.0.9 instead of broadcasts, in case of the known neighbors as unicast packets,
- support for authentication between two routers, but at a poor level (either unencrypted or encrypted with MD5).

The comparison of the sent information is shown in Figure 4.2 (on the left RIPv1 sending only the address and metrics, on the right RIPv2 sending also the mask and other information).

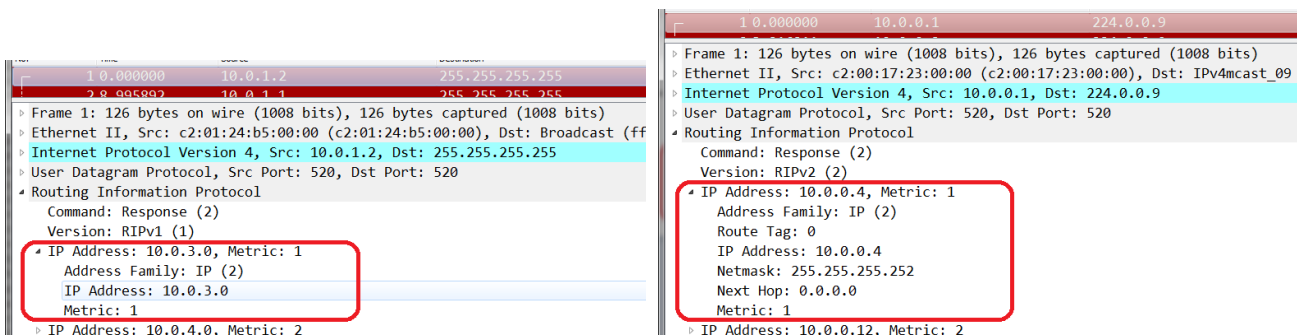




Figure 4.2: Comparison of a RIPv1 and RIPv2 update

For the both versions of RIP, the metric is considered to be of poor quality, so usability is only in small networks, where it does not matter much. The limit of 15 jumps remains.


 RIPv2 may or may not use *automatic path summarization*. The summary works so that if there are two (sub)networks in the routing table with the same “class” part of the address, they are merged into one row of the routing table. Merging is performed at the boundary according to the address class.

Example

The subnets 10.1.0.0/16 and 10.2.0.0/16 have the class A addresses, i.e. we are interested in the first byte of the address – in both it is 10. These two subnets would be merged into one row of the routing table labeled by 10.0.0.0 when autosummarization is enabled. If the path to both leads through the same network interface, that’s ok, otherwise we have a problem. So the decision to turn autosummarisation on/off is very important.

 **emph** RIPv2 (in substance the third version, next gen, RFC 2080) adds IPv6 support, so it sends IPv6 addresses in updates. It has other features similar to RIPv2.

4.5.2 IGRP and EIGRP


 **IGRP** (Interior Gateway Routing Protocol) is Cisco’s proprietary protocol. It implements a distance vector algorithm, but generally has better properties than RIP, at least in terms of metrics. The protocol configuration includes the autonomous system number (therefore it can be used besides other

routing protocols on a single device). Unfortunately, it does not support classless routing (no mask or prefix length is used).

Routing information is sent every 90 seconds (by default) by broadcast. The metric is multi-criteria, a combination of the following criteria is used:


- *bandwidth* in b/s,
- network *delay* in ms,
- network *load*, the value of 255 means load,
- *reliability*, the value of 255 means 100% reliable link,
- *maximum transmission unit* (MTU).

By default, the first two criteria are used (they can be determined from the topology), others are optional, can be configured to use them.

 **EIGRP** (Enhanced IGRP) is an enhancement to the IGRP protocol. The routing algorithm according to this protocol has some elements of the link state algorithm appear, but nevertheless this protocol is included in the group using the distance vector algorithm. As with IGRP, the metric is calculated from network bandwidth and delay, although network load, reliability and MTU criteria can be added, but this is not usually done.

While IGRP is classful, EIGRP is classless. The network information also includes a network mask. The newer version of EIGRP also supports IPv6.


EIGRP maintains three tables: a neighbor table, a topology table, and a routing table. Interestingly, these tables are managed at the application layer, so EIGRP is actually an application protocol, although it affects the operation of the routing layer. The tables are updated when the topology changes, by multicast messages. The address is 224.0.0.10 for IPv4 and FF02::A for IPv6.


 RIP uses a simple algorithm for the number of routers in the path, OSPF creates a routing table using Dijkstra's algorithm, EIGRP uses the *DUAL* algorithm (Diffusing Update Algorithm).

DUAL is a relatively complex algorithm, the aim of which is not only to calculate the most optimal paths to targets without loops, but also to maintain data on some other less optimal paths. Unlike other algorithms, it can balance the load on multiple paths to the same destination (so it can add multiple paths to the routing table for a single destination), even with different metrics (then multiple packets are sent in a “better” path and fewer packets in a path with worse metrics).

In addition, pre-calculated backup routes are stored so that when changing the topology, it is not necessary to restart DUAL (it is quite CPU-intensive).

There are several types of messages that are sent when using EIGRP (Hello, Update, Query, Reply, Ack), some acknowledged, others not.

 At the transport layer, Cisco uses its own special *Cisco RTP* protocol (Reliable Transport Protocol – not to be confused with the “standard” RTP in means of Real-time Transport Protocol, despite the same abbreviation, these are two different protocols). Cisco RTP has features somewhere between TCP and UDP in the sense that it can function both acknowledged and unacknowledged, depending on what is transmitted. For example, Hello packets are transmitted as unacknowledged, while Update packets are transmitted as acknowledged.

 EIGRP mainly uses two timers: the *hello timer* determines how often neighbors should send each other hello packets, the *hold timer* determines how long the router should wait for a hello packet from

a neighbor (after a hold without response from a neighbor, the neighbor is declared unavailable).

Multiple instances of (E)IGRP and other routing protocols can run on a single router because the configuration includes an ASN.

The EIGRP protocol was originally proprietary (Cisco used it only on its devices), However, it was released in 2013 (RFC 7868) and today it is a freely applicable standard that can be implemented on devices from other manufacturers (however, no one gets involved too much, the implementation of EIGRP is relatively complicated despite the undeniable advantages).



Remark:

While IGRP is practically not used today, the EIGRP protocol is still encountered in practice. It is considered a better algorithm than OSPF, but on the other hand it is not supported by most manufacturers and is computationally intensive.




Task


The RFC standard for EIGRP is available at <https://tools.ietf.org/html/rfc7868>. Look at the web page and find the Terminology section and read the definitions Diffusing Computation and Diffusing Update Algorithm (DUAL) with subsequent definitions. The third chapter describes DUAL in detail.



4.5.3 OSPF


 OSPF (Open Shortest Path First) uses a link-state algorithm, which means fast network convergence with each topology change. It is an open protocol, we meet it on devices from many different manufacturers. Although it is typically used as an interior routing protocol, it can also be used as an exterior one (between autonomous systems).


OSPF uses classless routing, so we also enter the prefix length for the address information. OSPF version 2 (RFC 1247 and RFC 2328) supports IPv4 addresses, while OSPF version 3 (RFC 2740) works with IPv6 addresses.

 Routing information is sent immediately after a topology change or at least every 30 minutes (this is still significantly less often than with previous protocols). However, the routing table is not sent, only topology information is transmitted to neighbors.


Like EIGRP, OSPF sends several types of packets (Hello, Database Description with a brief description of the topology, Link-state Request with a request for more detailed link information, Link-state Update with more detailed link information, Link-state Ack with acknowledgement), but unlike EIGRP, it encapsulates these packets into IP packets (it operates at the L3 layer).

Every router maintains a topology database of the network or its area on the network. The topology database is in the form of an oriented graph, with paths being assigned a different metric in each direction. The Dijkstra algorithm (SPF) explained above is used to calculate paths according to the topology database.

 The metric is denoted by the term *cost*. The path cost is based on the costs of all links of the path (their sum). The cost of a link is derived from the link bandwidth – the faster the link the better (lower) the cost.

 Divide the number 100 000 000 by the bandwidth in b/s, i.e. for example 100 Mbps has the cost of 1, an interface with a throughput of 1.2 Mbps has the cost of 84, etc.

This is a rather annoying limitation, because we usually connect routers in faster links, it is no exception to use 10Gb or faster links (but all links 100 Mbps or faster would have cost=1). Fortunately, the conversion factor can be changed.

 **Example**

If the interface is faster than 100 Mbps, it costs 1 by default (rounded up). This *inconsistency* is prevented by changing the formula in the configuration (changing the reference bandwidth). For example, a sequence of commands (for Cisco devices):


```
R1(config)# router ospf 1
R1(config-router)# auto-cost reference-bandwidth 1000
```

The number in the first command is the process number (several instances of OSPF can run on the router, in reality it is not recommended). Thus in OSPF with process number 1 we change the reference bandwidth to 1000 kbps, so in the formula we would use the number 1 000 000 000 (this has to be done in all routers in the domain so that we don't make unstable network with unpredictable behavior). This will distinguish a fast Ethernet link from a gigabit link (but a 10Gb link will have the same cost as a gigabit link). If we want to differentiate even 10Gb links, we add one more zero:

```
R1(config-router)# auto-cost reference-bandwidth 10000
```





The path cost is the sum of the costs for the individual links of the path. So when SPF finds different paths to the same destination, sums link costs for each path, compares the results and chooses the path with the lowest cost.

 OSPF supports *hierarchical routing*. This means that routers (and their networks) can be divided into *areas* within the routing domain. Each router maintains only the topology of its area, the other areas remain hidden from it, and updates to topology changes are limited to the given area. We distinguish *internal routers* (within an area), *area border routers* (belonging to several areas – ABR, Area Border Router) and *autonomous system border routers* (between different autonomous systems).

The backbone area is referred to as *area 0*, the routers in the backbone network are *backbone routers*. The backbone network can also be WAN. There should always be a backbone network and all “non-zero” areas should only be adjacent to area 0.

Of course, network address prefixes are also part of the routing table. Like many other protocols, OSPF uses *routing information summary* – CIDR. Summarization is enabled especially on border routers to reduce amount of routing information sent.

 In LANs (generally in broadcast networks), there must be one *Designated Router* that controls all routing information updates on the network. In case the designated router fails, there should be *abackup designated router*. The designated router calculates the paths and thus lowers load from the other routers in the area. Although the selection of a designated router can be left to the OSPF protocol, it is better to do it manually, because automatic selection does not always turn out perfectly.

 OSPF also supports authentication. In version 2 routers can authenticate themselves using MD5 or SHA checksums, in version 3 the authentication mechanism has been reworked – IPsec authentication and encryption (which is built into IPv6) is used.

Other features of OSPF:

- support for *load ballancing* among paths with the same cost (but either all packets with the same final destination IP address go the same path),
- support for routing according to IP ToS (Type of Service).

OSPF is currently the most widely used interior routing protocol.




Additional information:


<https://www.ciscopress.com/articles/article.asp?p=2262897&seqNum=5>




4.6 Exterior Routing

The term EGP (Exterior Gateway Protocol) actually has two meanings – it refers generally to exterior routing protocols, but also to one specific exterior protocol (the oldest one).


 **EGP** (Exterior Gateway Protocol) is a simple exterior routing protocol for exchanging network availability or unavailability information (previously described protocols were internal), the protocol number is 8. Routers constantly check the functionality of their neighbors and also exchange information with them about the availability of connected networks. No metrics are used, so the existence of multiple parallel paths to the same network is excluded; not applicable if there is a loop on the roads (requires a router tree). At present, we no longer encounter EGP.

 **BGP** (Border Gateway Protocol) is also an exterior routing protocol, but it is already more complex, functional and usable. It interconnects different autonomous systems, so it serves to route between autonomous systems. In fact, it can be used for both internal (iBGP, within a single autonomous system) and external (eBGP, between different autonomous systems) routing. It supports VLSM, CIDR (uses classless routing, prefixes), path aggregation and in the newer version of course IPv6.

While other routing protocols use unreliable traffic (UDP) or partially reliable (Cisco RTP) for greater flexibility, if they use any transport protocol at all, BGP uses reliable traffic (encapsulates in TCP). The reason for the use of connection-oriented communication is the principle of WAN networks in which it is used – there is no communication other than connection-oriented.


 Routers using BGP maintain “neighbor relationships” (peering) – they periodically send *KeepAlive* messages to make sure they are working properly. However, they do not detect their neighbors automatically, the neighborhood must be manually configured, and we also enter the ASN of the neighbor in the configuration. When exchanging routing information for the first time, the router informs about its *prefix* and ASN number and receives the entire routing table, and when making changes, only the parts of the routing table affected by the change.

When announcing its prefix, the router adds its ASN to the prefix. During handover between routers, each router also adds its ASN, so the total length of the path identification string is constantly increasing. Thus, a path description is simply a sequence of ASNs. Higher priority is given to a shorter path, i.e. a path with a smaller number of ASNs, leading through fewer BGP routers. The path that has that sequence shorter, is selected as better. However, the quality of the path can actually be affected by configuring several other parameters.

 As we can see, the BGP protocol cannot be precisely classified as a distance-vector algorithm or a link-state algorithm. The *path vector algorithm* is used, which is similar to the distance-vector algorithm (but we have list of ASNs instead of number of routers on the path).

If multiple paths lead to a target autonomous system, it usually chooses the one that leads through fewer “pass-through” autonomous systems.

In the routing table, each path is primarily prefixed with the appropriate network and a sequence of ASN numbers (identifiers of autonomous systems through which the path leads).

 One must be careful with the BGP protocol. Because when we use it, we actually connect to the routing domain of the entire Internet, we must pay attention to what information we send to the Internet via BGP. It’s not just about what we shouldn’t see “for security reasons”, but if we are careless, we can affect the way the whole Internet works. Legendary is, for example, the problem with incorrect configuration of BGP routers in Pakistan, which caused, among other things, a failure of YouTube servers for several hours.

It is recommended to use BGP only when our autonomous system is connected to at least two other autonomous systems (i.e. the path between different ASs passes through us). Conversely, it does not make sense to use BGP if we are only connected to one other autonomous system (and of course if we are only part of a larger AS), and also if we are not completely experts in BGP . . .



Remark:

BGP is the main routing protocol on the Internet and WAN networks (there are also modified variants for specific types of networks), its use is relatively common.



Additional information:

- <https://inl.info.ucl.ac.be/blogs/08-02-25-bgp-misconfigurations-strike-back-pakistan-and-affect-youtube.html>
- <https://nakedsecurity.sophos.com/2018/10/30/china-hijacking-internet-traffic-using-bgp-claim-researchers/>
- <https://www.networkworld.com/article/2272520/six-worst-internet-routing-attacks.html>
- <https://www.noction.com/blog/bgp-hijacking>



4.7 Software-Defined Router

Routers are usually special hardware devices, but for this purpose we can use any computer (routing in a smaller network is not very computationally intensive) with special software. If static routing is sufficient, then we can use any Linux distribution. It certainly requires something extra for dynamic routing.

*Zebra*³ is very popular free software for UNIX-like systems supporting the routing protocols RIPv1, RIPv2, OSPFv2 and BGPv4.

*Quagga*⁴ is the open-source software for UNIX-like systems (including Linux, FreeBSD, NetBSD,

³<http://www.zebra.org/>

⁴<http://www.quagga.net/>

Solaris) supporting OSPFv2, OSPFv3, RIPv1, RIPv2, BGPv4 and others. Quagga is a clone of Zebra, currently the most recommended in Linux.

*XORP*⁵ (Extensible Open Router Platform) is the open-source platform supporting nearly all known routing protocols. It is even possible to download a demo LiveCD on the project website.

*NAT32 Windows Software Router*⁶ is software for Windows (only for small networks, it supports RIP). It is free to download, pay for support.



Additional information:

- <https://teklager.se/en/best-free-linux-router-firewall-software-2019/>
- <https://devconnected.com/how-to-configure-linux-as-a-static-router/>
- <https://monoinfinito.wordpress.com/series/setting-up-a-linux-gatewayrouter-a-guide-for-non-network-admins/>



Task

Choose one of the above solutions and explore its license, method of installation and use, features, including currently supported routing protocols.





⁵<http://www.xorp.org/>


⁶<http://www.nat32.com/>

Chapter 5

Decentralized and Distributed Systems


 *Preview:* In this chapter, after discussing the concepts, we will deal in more detail with some network technologies, which are in principle decentralized or distributed. In addition to the routing already discussed, this includes DNS, QoS, VoIP and video telephony.

 *Keywords:* Centralized, decentralized and distributed system, DNS, domain, TLD, SLD, FQDN, zone file, primary and secondary DNS server, caching-only server, DNS request, recursive and iterative query, host table, resource record, nslookup, dig, DNSSEC, DoT, DoH, SFP, DKIM, DANE, DNS RPZ, WHOIS database, QoS, IntServ, DiffServ, VoIP, SIP, H.323, video telephony, PBX.

 *Goal:* The aim of this chapter is to understand principles of selected decentralized a distributed technologies, such as DNS system, quality of service, VoIP, video telephony and others.


5.1 Types of Systems

We will first clarify a few important concepts and define the types of systems.

 **A centralized system** is a system with a single central control unit. Communication usually takes place in the form of a client-server, where the server is the central control unit.

In the field of computer networks, we can classify the following systems as centralized:

- *the RADIUS protocol* where the central control unit is the RADIUS server, communicating with its clients (mostly, but not only, wireless access points),
- *the CMIP protocol* (Common Management Information Protocol) – one of the ISO/OSI protocols intended for network management (such as SNMP in TCP/IP).

 **A decentralized system** has several control units (server/administrative nodes), more or less equivalent (e.g. with similar functions). The individual nodes of the network control themselves and possibly their surroundings, so there is no single node controlling all the others.

The purpose (and an advantage over centralized systems) is to increase the robustness of the network (the same function is provided by multiple “subcentral” – server – nodes, client nodes can communicate


with any node providing the required service, mostly the nearest one). It is easy to resolve a situation where one of server nodes stops providing service (for example, it is disconnected from the network).

The disadvantage is a more demanding implementation, because the server nodes must be synchronized (especially their databases), or they have to communicate to get necessary information stored elsewhere.

In the field of computer networks, we encounter decentralized architecture, for example here:

- many *routing protocols* are decentralized,
- *the SIP protocol* (Session Initiation Protocol) used in VoIP or generally anywhere where it is necessary to establish a session.

The Internet itself was originally decentralized, and to a large extent it still applies. But full decentralization would mean keeping the system running smoothly (perhaps with some slowdown or not providing some services that can “wait”, are not time-critical) even after the vast majority of server nodes are disconnected, which property the Internet does not do.

 **A distributed system** is such system is a system whose functions are distributed (divided) to different nodes of network while maintaining several important properties.

This is mainly a property of *transparency*. For example, access transparency means that both local and remote resources are accessed in the same way, here via protocols. Migration transparency means the possibility of moving the provided services between different network nodes without a significant effect on network performance.

Another important feature of distributed systems is *flexibility*. This feature means that the system is adaptable to changes in the environment (including failures and outages of system components). This precludes any centralization of decision-making, each node of the network must be as independent as possible in its activities, and requires the services of other nodes while maintaining transparency.

A flexible system can be extended in any way (almost – there are technical boundaries that are very difficult to cross) or, conversely, reduced, the function of a removed node can be performed by another node.

We encounter distribution in some routing protocols and, for example, in the DNS system (generally for domain services).

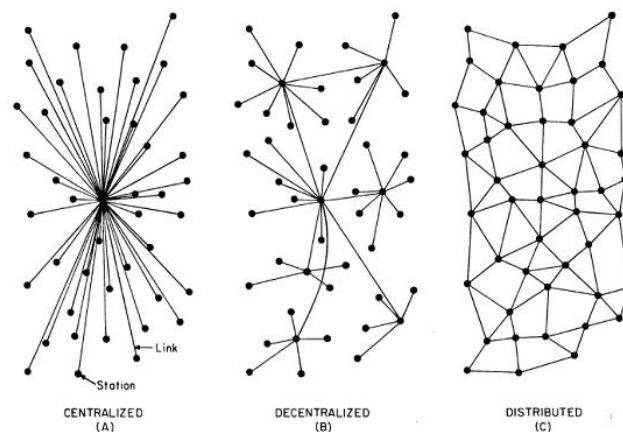



Figure 5.1: Centralized, decentralized and distributed architecture¹

¹Resource: <https://medium.com/distributed-economy/what-is-the-difference-between-decentralized-and-distributed-systems-f4190a5c6462>

 A distributed system can be determined by the *client-server* architecture, where it is clearly determined which nodes are server (provide services, implement functions) and which are client (use server services). Another way to deploy a distributed system is *integrated architecture* (symmetric system), where each node can be a server or a client, depending on the requirements of the ongoing communication.

What can be distributed? In particular, *data* (content management systems, distributed information systems, etc.), *calculations* (complex calculations are distributed among multiple devices in the network) or *resources* (computing performance, storage, etc.) – this is actually a superset of the previous two options.



Remark:

What is the difference between a distributed and a decentralized system? If a node in a distributed system needs a particular service, it does not have to determine where specifically that service should be provided (because the system should be transparent). It sends a request and “someone” processes this request. In a decentralized system, nodes usually turn to their “local center”.





Task

Find more examples of centralized, decentralized, and distributed systems, whether from computer networks or elsewhere.



5.2 Domain Name Service

 *Name Service* is a general principle for translating names (text strings) to numeric addresses. A typical representative of name services is DNS (Domain Name Service, RFC 1035, IANA considerations in RFC 2929), but there are other name services – such as WINS (Windows Internet Name Service, used to translate NetBIOS names to IP addresses).

 The DNS mechanism is a typical representative of distributed systems in the field of computer networks. The database of names that DNS works with is distributed on many network devices in the network, with each device registering mainly addresses from its own network – the *principle of locality*. Distribution works here mainly because the DNS address system is hierarchical.

5.2.1 Domains and Names

We will assume that the reader knows what DNS is and what a domain address looks like. We focus on basic definitions, the functionality of the whole system and the PDU format.



Definition (Domain, domain name)

Domain is a network or group of networks under common management and sharing a common (domain) name. Each *Domain Name* must meet the following conditions:


- may contain letters of the English alphabet, numbers and a hyphen, the hyphen not being at the beginning or end,

- the maximum length of a single name is 63 characters,
- the maximum length of concatenated domain names is 255 characters,
- within a domain, subdomain names are unique.

A domain can have multiple names assigned to it. One of them is *canonical name* (main), the other names are called *aliases*.



The domain address space is also *hierarchical*, but in the right-hand direction – devices in the same domain have the part of the name address the identical, domains are built hierarchically. We can draw their relationship using a tree as well.

 The top levels of the domain tree also have their names:


- At the root of the tree there is the *root domain*.
- *Top-Level Domains* (TLD) are at the first level of the tree, e.g. `.org`.
- There are *Second-Level Domains* (SLD), third level, etc.

A domain or group of domains is managed by a specific entity – *domain administrator*. According to our picture, for example, the domain `.slu`, including subdomains, is managed by Silesian University, the domain `.cz` is managed by the main domain registrar for the Czech Republic, CZ.NIC.


The list of TLD domains is maintained by IANA (Internet Assigned Numbers Authority), similarly as IPv4 and IPv6 address spaces, at the Root zone database (database present at the root servers).

There are several types of the TLD domains:


- generic TLD (gTLD): created in early development of the Internet, named after the focus:
 - `.com` – commercial organizations,
 - `.edu` – educational institutions,
 - `.gov` – USA government,
 - `.mil` – USA military,
 - `.net` – organizations maintaining network standards, nodes for distributed computing, anything related to computer networks,
 - `.org` – various organizations,
- country code TLD (ccTLD): international country codes, national abbreviations (`.cz`, `.uk`, `.sk`, etc.) or for groups of states (`.eu`),
- infrastructure TLD (ARPA): the domain `.arpa` is intended for the network infrastructure management, for example the subdomain `.in-addr.arpa` is used for reverse DNS lookup (translation of an IPv4 address to the corresponding domain address) or `.ipv6.arpa` (the same for IPv6),
- sponsored TLD (sTLD): hundreds of new TLDs, for example `.audio`, `.cafe`, `.download`, `.help`, `.chat`, `.training`, etc.

 *FQDN* (Fully Qualified Domain Name) is the full name of a device, the full domain address. This address is compiled by going up from the leaf (the given device/domain from the request) upwards to the root and separating the domains on the path with a dot, for example `www.slu.cz`. The maximum length of the FQDN is 255 characters.


5.2.2 Zones and DNS Servers

 We associate one or more domains into a common *zone* (typically part of a subtree of domains). A specific *authority*, i.e. a responsible organization, is determined for each zone.


Zoning (as grouping of domains) is related to responsibility and technical administration, zones usually do not overlap, except for their boundaries (there must be some communication between the zones). For example, the entire subtree `slu` forms a zone managed by the Silesian University, while the upper domain `cz` belongs to the zone managed by CZ.NIC.


 Management of a zone is ensured on the *domain (DNS) server* performing the following tasks:

- keep an address table (host table), it is stored in the *zone file*,
- answer DNS requests according to this table, e.g. translate a name address to the corresponding IP address.


 We have a single primary DNS server in the zone and we can also have auxiliary servers there – secondary DNS servers and caching-only servers. In relation to the zone file:

- The *primary DNS server* maintains and guarantees a zone file with the host table, we always make changes to the table on this server and the records on this server are always trusted, *authoritative*.
- The *secondary DNS servers* copy the zone file from the primary DNS server at regular intervals (synchronize, the procedure is called *zone transfer*), their purpose is to distribute the load so that the primary server is not overloaded. Their answer is *authoritative* as well.
- The *caching-only servers* do not keep zone file. When such a server receives a request for an address, “asks” the primary or secondary server, but stores the response in cache for a short time and, if it is requested for the same address, uses the information from cache (so it doesn’t have to query further). The caching-only server response is not authoritative, but usually sufficient.

 The most well-known DNS servers are BIND, MyDNS, TinyDNS, djbdNS (rather as caching-only), in Windows servers we also have the role of DNS Server.

 *DNS resolver* is a component that provides DNS requests and keeps the results of previous requests in its cache for some time. This component can be found in any system that is connected to a network using DNS, including computers and mobile devices. The most important information that a DNS resolver needs is the address of the responsible DNS server.


DNS servers also have their own DNS resolvers, because even these devices need such DNS records that they do not have in their zone file. Their resolver queries another DNS server and stores the result in its cache (so both primary and secondary DNS servers have a cache, not just caching-only servers).

 **Procedure (Possibilities of DNS request evaluation)**

The DNS resolver gradually tries the following options when evaluating a DNS request:

- the file `/etc/hosts` (UNIX-like systems) or `... \system32\drivers\etc\hosts` (Windows) contains a list of pairs
IP address + name address,
- the result is in the DNS cache if we requested the same name address in the near past (each record has the TTL value for validity), but some UNIX-like systems usually do not use DNS caching,
- the last and most time consuming option is to query a DNS server.



 Medium and larger companies have their own DNS server, while smaller companies and households use the DNS server services of their ISP. In addition, there are many *public DNS servers* that we can use when the one from our ISP does not work as it should or is not stable enough, for example:

- Cloudflare: 1.1.1.1 (figures) and 1.0.0.1,
- Google: 8.8.8.8 (snowmen) and 8.8.4.4,
- Cisco OpenDNS: 208.67.222.222 and 208.67.220.220,
- CZ.NIC: 193.17.47.1 and 185.43.135.1,
- Quad9: 9.9.9.9 and 149.112.112.112,
- Verisign: 64.6.64.6 and 64.6.65.6, etc.

We can set it up on the end device, but it is more practical to do it on a router with DHCP server function, so that this information reaches all end devices during obtaining address, and on laptops we avoid problems if we bring the device to work and try join Active Directory.

Tasks


1. Find the `hosts` file on your computer and look at its contents. Most of the content is likely to be commented out (maybe everything). Consider what would happen if you added a pair for a domain address and wrote 127.0.0.1 (or localhost) as the IP address. Note: You need administrator privileges to open this file for writing.
2. The command `ipconfig /flushdns` displays contents of the DNS cache in Windows. If you are currently using the given operating system, try this command and examine your DNS cache.
3. Test and compare latency of the above listed public DNS servers (simply use `ping`).




5.2.3 Evaluation of DNS Requests

DNS servers, as mentioned above, keep information about its domain, the subdomains in the zone file, as well as information about the DNS server of the upper-level domain. Each DNS server knows at least one root DNS server.

DNS querying can be distributed – the DNS server often cannot respond to a request immediately according to its records, so it turns or references some related TLD DNS server or one of the root servers.

 A questioner (actually a resolver) sends a query (what is the IP address of the device named `www.abcd.org`?) To the so-called *local DNS server* (the closest one). If this server already knows the response, it will provide it. If not, the next steps differ depending on the type of query used. We distinguish two types of DNS queries:


- a *recursive query* – the questioner receives a ready-made answer or an error message (domain does not exist),
- an *iterative query* – the questioner gradually gets references to servers where a better answer should be got.

 **Recursive query.** If the requested DNS server knows the response, it will reply back immediately. Conversely, if it does not know the response (the address is not from its domain), it contacts a relevant TLD server or a root server. The next queried DNS server behaves similarly – if it knows the destination, it replies with the address, if it does not know the destination, it continues to query (usually in its subdomains).

In this way, the query is sent recursively down the domain tree. When it finally reaches a DNS server that knows the response, that server sends the response in the recursive path back.

Thus, the procedure is as follows:

- If the queried name is from a subordinate zone (subtree, the right part of the address is the same), the name server finds the first domain in the FQDN that is not its from right-to-left. It has links to all domains on the boundary with subordinate zones (for example, in the zone file for `cz` there is a link to the DNS server of the domain `slu`), so it passes the query to the DNS server of this subordinate domain. In the lower-level domain, the query is processed or resubmitted below. The response is found recursively downwards, which is then distributed back to the first resolver in the same way.
- If the name being queried is neither from the own zone nor subordinate zones, the DNS server forwards the query to the DNS server from the relevant TLD zone (if it does not know the relevant TLD server, the query is forwarded to some of the root servers). It either handles the query or passes it to a low-level node. After processing, the reply is again distributed in the same way to the first resolver.


 **Example**


If a device from the domain `fpf.slu.cz` asks for the IP address of the server `www.fpf.slu.cz`, the DNS server of the domain `fpf.slu.cz` responds immediately, because this web server belongs to its zone.

If a device from the domain `fpf.slu.cz` queries the IP address of the server `www.cuni.cz`, the DNS server of the domain `fpf.slu.cz` deals with the query first. The requested domain is not in the zone file, but the TLD domain is `.cz`, so the name server for the given TLD domain is the next part of the request chain.

This TLD server finds out that the queried address belongs to the lower-level domain `cuni.cz`, in the zone file it finds a contact to the DNS server from that domain, which already finds the response – the IP address of the web server `www.cuni.cz`, because it belongs to its zone. The response goes back to the first querying device in the same way as the query.



 **Iterative query.** Here the activity is mainly on the side of the resolver. If the queried (local) DNS server does not know the answer, it does not formulate further queries, but instead sends back the list of DNS servers that might know the answer (“I don’t know, ask xxxx”). The resolver chooses one of the recommended DNS servers and sends query further. Each queried server again sends either a final response or a recommendation with the names of other DNS servers (so called best possible answer).


 **Example**

If we need to find out the IP address of the computer `www.something.org`, we contact the local DNS server. It will recommend that we send the query to one of the TLD servers of the `org` domain. The


next queried root server either looks up the address in its records or returns only the addresses of the name servers that may know the response.

The query is evaluated distributively in this way (gradually according to the nesting of domains of different levels), the last queried DNS server returns the required IP address.




 In the real world, recursive and iterative querying are combined. The recursive method is used in the first phases of the path, when a simple resolver is queried, in subsequent phases iterative querying takes place. Some DNS servers (as software) can only work recursively, others can also work iteratively.

5.2.4 Host Table

 A key element for DNS resolution is the *host table* stored in the zone file. We can think of it as a table in which each *record* (Resource Record, RR) has its own row, and on that row we have the following data (in columns) – for the most common types of records:


- name (e.g. name address to map),
- record type,
- class – usually “IN” as Internet,
- TTL,
- IP address to map or other data, depending on the record type.


The TTL value is the lifetime of the resource record in seconds in case this record is downloaded to cache of another device.

 The fact that “record type” (the third item of the previous list) is present means that there are different types of DNS records. The most used record types are:

- A – to map a name address to an IPv4 address,
- AAAA – to map a name address to an IPv6 address (four “A”s are here because IPv6 addresses are four times longer than IPv4 addresses),
- CNAME (Canonical Name) – to map an alias to a canonical name,
- SOA (Start of Authority) – administrative information for the given domain,
- NS (Name Server) – record for IP addresses of authoritative DNS server(s) for the given domain,
- MX (Mail eXchanger) – e-mail server for the given domain,
- SRV (Service locator) – similar meaning as NS and MX, but more general, for various types of services,
- TXT – variable type record used, for example, to store public keys that anyone can use to verify that an e-mail sent from our domain has actually passed through our mail server or has been spoofed,
- typ PTR – for reverse mapping (to map an IP address to the relevant name address),
- IPSECKEY, OPENPGPKEY, SMIMEA, SSHFP, TLSA – records to store security related information such as public keys,
- etc.

We usually need records of type A or AAAA, when we know the name address and we need either an IPv4 address or an IPv6 address.

 We distinguish between *forward lookup* (finding an IP address by domain name) and *reverse lookup* (finding a domain name by IP address). The reverse lookup is performed through the domain `in-addr.arpa`. The main problem here is that both name and IP addresses are hierarchical, but each in a “different direction” – the TLD domain in the name address is on the right, while the network address (main part) in the IP address is on the left. The PTR records are used for reverse lookups. They map IP addresses to canonical names.


 **Example**

Suppose that somebody wants to know the domain name of the IPv4 address `77.201.75.176`. The appropriate PTR record holds the IP address with bytes in the reverse order, followed by the infrastructure domain `in-addr.arpa`:

```
176.75.201.77.in-addr.arpa
```

This PTR record leads to the searched domain (its canonical name).



 Every operating system has the tool providing mapping between IP address and name address, e.g. `nslookup`. UNIX-like systems including Linux and MacOS have better command: `dig` (abbreviation of Domain Information Gropher). This tool is more popular than `nslookup`, especially for testing DNS server settings, because its output is more extensive and matches the data in the DNS packet.

```
dig www.root.cz
```

we get a comprehensive response with all the necessary information, in addition to the requested IP address (or multiple addresses) such as the responding DNS server, how long the communication took, the class of the record (where valid), record type, etc.,

```
dig -x 91.213.160.118
```

reverse query,

```
dig www.root.cz +short
```


short reply similar to `nslookup` (only the IP address),

```
dig google.com ANY
```

all types of records related to the given server,

```
dig seznam.cz NS +short
```

the domain names of DNS servers in the specified domain are listed, their IP addresses can be found by a subsequent query according to the listed domain names.

 **Example**

We will try the `dig` command:

```
sarka@notebook:~$ dig www.root.cz
```

```
; <<>> DiG 9.7.1-P2 <<>> www.root.cz
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 55084
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;www.root.cz. IN A

;; ANSWER SECTION:
www.root.cz. 5 IN CNAME root.cz.
root.cz. 5 IN A 91.213.160.118

;; AUTHORITY SECTION:
root.cz. 5 IN NS ns.iinfo.cz.
```



```

root.cz. 5 IN NS ns6.adinit.cz.

;; ADDITIONAL SECTION:
ns6.adinit.cz. 5 IN A 81.91.85.230
ns6.adinit.cz. 5 IN AAAA 2001:1568:b:149::1

;; Query time: 18 msec
;; SERVER: 192.168.184.2#53(192.168.184.2)
;; WHEN: Mon Jul 11 07:09:03 2011
;; MSG SIZE rcvd: 152

```

These items are in the output:

- in the header of the report there is information about how the command was called and which version we have, below part with numerical data (numbers correspond to the numbers of records given in the following sections), options, etc.,
- the “Question section” with query data (queried domain address, class – “INternet”, type – “A”),
- the “Answer section” with response data (with two records),
- the “Authority section” with the list of name addresses of DNS servers that provide an authoritative answer about the requested name (if we do not trust the information obtained, we can ask one of these servers),
- the next section contains IP addresses of the DNS servers listed in the previous section,
- statistical information.

Similarly, we will now find out information about the host `mail.google.com`, but we ask another DNS server. The DNS server name must be indicated by the “at sign” to make it clear that this is not the name to be mapped, we can enter a domain or IP address after the at sign. Here we (coincidentally) query the DNS server provided by Google:

```

sarka@notebook:~$ dig @8.8.4.4 mail.google.com

;<<>> DiG 9.7.1-P2 <<>> @8.8.4.4 mail.google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20135
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;mail.google.com. IN A

;; ANSWER SECTION:
mail.google.com. 86399 IN CNAME googlemail.l.google.com.
googlemail.l.google.com. 299 IN A 74.125.232.246
googlemail.l.google.com. 299 IN A 74.125.232.248
googlemail.l.google.com. 299 IN A 74.125.232.247
googlemail.l.google.com. 299 IN A 74.125.232.245

;; Query time: 56 msec
;; SERVER: 8.8.4.4#53(8.8.4.4)
;; WHEN: Mon Jul 11 07:21:14 2011
;; MSG SIZE rcvd: 124

```

Obviously, we received the answer, but we requested it from a server that is not authoritative for the record (it takes data from its cache or queries elsewhere).

Next, we will try a shortened statement (we will use the + `short` parameter), we want DNS servers in the `seznam.cz` domain.

```
sarka@notebook:~$ dig seznam.cz NS +short
```

```
ns.seznam.cz
ms.seznam.cz
```



Example

We can also use the `dig` command to find the entire path of our query:

```
sarka@notebook:~$ dig seznam.cz +trace
```

```
; <<>> DiG 9.7.1-P2 <<>> seznam.cz +trace
;; global options: +cmd
. 5 IN NS i.root-servers.net.
. 5 IN NS j.root-servers.net.
....(shortened)
;; Received 272 bytes from 192.168.184.2#53(192.168.184.2) in 12 ms


cz. 172800 IN NS f.ns.nic.cz.
cz. 172800 IN NS d.ns.nic.cz.
....(shortened)
;; Received 334 bytes from 192.5.5.241#53(f.root-servers.net) in 10 ms

seznam.cz. 18000 IN NS ms.seznam.cz.
seznam.cz. 18000 IN NS ns.seznam.cz.
;; Received 93 bytes from 194.0.14.1#53(c.ns.nic.cz) in 43 ms

seznam.cz. 300 IN A 77.75.72.3
;; Received 43 bytes from 77.75.77.77#53(ms.seznam.cz) in 9 ms
```



If we do not specify the DNS server to which the query is to be sent, the `dig` command will use the DNS servers specified in `/etc/resolv.conf`.

 There are several *DNS lookup tools* available at Internet, two of them are in the list of additional resources below (MXToolbox, DNS Checker).



Additional information:

- <https://www.liquidweb.com/kb/reverse-dns-lookup/>
- <https://mxtoolbox.com/DnsLookup.aspx>
- <https://dnschecker.org/all-dns-records-of-domain.php>
- <https://simplifiedns.plus/help/dns-record-types>
- <https://www.cloudflare.com/learning/dns/dns-records/>



Tasks

1. Verify that the interactive mode of the `nslookup` command is used.
2. If you use any UNIX-like system (Linux, MacOS, FreeBSD, Solaris, etc.), try the `dig` command.
3. Examine one of the above listed DNS lookup tools, try with some domain address (e.g. `www.slu.cz`).



5.2.5 DNS Protocol and DNS Packet

DNS is not just a service, it is also the application protocol that provides this service. This protocol determines what a DNS packet should look like, how it should be handled, how the entire DNS system works, including how to work with zone files, their transfer between DNS servers (zone transfer), and mutual communication between DNS servers.

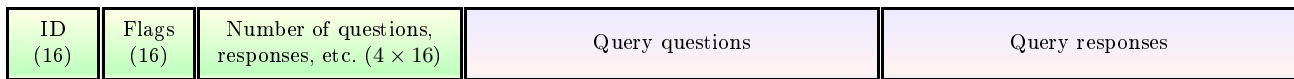




Figure 5.2: DNS packet


 The DNS protocol defines client-server communication, i.e. we distinguish between *query question* and *query response*. The same *DNS packet format* is used for both question and response. The header holds

- *identifier* (similar to IP) used to match the question and the response,
- *flags*, the most important of which is the one that determines whether it is a question or response packet, or, for example, in the case of a response, whether the server being queried is authoritative,
- several two-octet numerals (number of questions, responses, authoritative responses, additional information),
- question(s),
- response(s).

 The DNS response packet is created as follows:

- we use the same identifier as in the query question,
- in the flag field we set the flag for the answer and then a few more as needed,
- we copy the field for the number of questions, enter the number of found records from the zone file in the field for the number of responses,
- we copy the query questions field,
- we fill in the field for the response with the content of the found records from the zone file.

The authoritative response comes from a primary or secondary DNS server. From the DNS packet we also know whether the given response is authoritative or not, which servers are authoritative in the given zone, or it is possible to force an authoritative response by setting a flag in the query question packet.

 DNS packets are encapsulated into UDP or TCP segments, on the DNS server side it communicates *on the port 53* (for both UDP and TCP). UDP segments are preferred because speed is important for this type of communication. TCP is typically used when a DNS packet needs to be split into multiple segments, and when executing a zone transfer.

Because there can be several different clients communicating with the DNS server, each such client (for example, a web browser window, a mail client, Skype, etc.) must have its own dynamic port number in order to distinguish these communications.


 **Task**

If you haven't done so already, browse a few different DNS packets (there are usually a lot of them wandering around the network, or use a web browser). Compare the query question and the corresponding query response. Notice how the query and the response are mapped – how the client recognizes that a particular response belongs to a particular query.



5.2.6 Security in DNS

DNS security has been much discussed lately. We usually trust the mapping of a name address to an IP address, and many users have no idea that they can actually communicate with a counterparty with a different IP address than they expect. DNS attacks can result in, for example, a Man-in-the-Middle attack, redirection to an infected server to spread malicious code, eavesdropping on a password, card number, and so on. Therefore, as part of DNS security, we must ensure that DNS packets with mapping information cannot be spoofed.

 *DNSSEC (DNS Security Extensions)* is a security extension of the DNS system that allows the DNS client to verify the origin of the data (whether it comes from the correct DNS server).


When using DNSSEC, asymmetric encryption is used as follows:

- the DNS domain operator generates a key pair (private and public key),
- the public key is stored to the server of the superordinate authority of its domain (e.g. the upper-level domain, a hierarchy of trust is created), the certificate is generated and digitally signed by the superordinate authority,
- the DNS server encrypts (in fact, digitally signs) packets using the private key,
- the recipient (DNS client, resolver) uses the public key obtained from the superordinate authority for the given domain to verify validity.


These are actually electronically signed DNS packets.

The administrator of the Czech TLD domain also uses DNSSEC on its DNS servers, while its superordinate authority (and the place where the relevant public key is) is the administrator of the worldwide network of DNS servers. The superordinate authority for the second level domains is the appropriate top level domain DNS server.

The DNSSEC mechanism is specified in the following documents: RFC 4033 (introduction and requirements), RFC 4034 (Resource records for DNSSEC), RFC 4035 (protocol modifications). But several additional RFC documents are related to RFC.

 DNS messages can not only be digitally signed, but also encrypted. Currently, the following solutions exist for encrypting DNS communication:

- *DoT (DNS over TLS)* – DNS messages are encrypted using TLS (RFC 7858),
- *DoH (DNS over HTTPS)* – DNS messages are encapsulated into HTTPS messages (RFC 8484).

 **Additional information:**

- List of DNSSEC RFCs: <https://www.internetsociety.org/resources/deploy360/2011/dnssec-rfcs-3/>
- <https://www.cloudflare.com/dns/dnssec/how-dnssec-works/>

- <https://www.icann.org/resources/pages/dnssec-what-is-it-why-important-2019-03-05-en>
- <https://www.internetsociety.org/deploy360/dnssec/registrars/>
- <https://labs.apnic.net/presentations/store/2019-04-15-why-dnssec.pdf>
- <https://developers.google.com/speed/public-dns/docs/dns-over-tls>
- <https://www.cloudflare.com/learning/dns/dns-over-tls/>
- <https://www.cloudflare.com/learning/dns/dns-over-tls/>




Task

The main condition to deploy DNSSEC is that the relevant TLD must be signed. The full list of signed TLD domains is available at http://stats.research.icann.org/dns/tld_report/. Examine this list and verify that your superordinary TLD domain is signed.



5.2.7 DNS Helps Secure other Services


There are several techniques to avoid spam in e-mail, some of them are based on DNS mechanism.

 **SPF (Sender Policy Framework)** allows to specify which addresses are authorized to send e-mails from a given domain. These addresses are stored in the TXT record with the following content:

```
v=spf1 ip4:first.ipv4.address ip4:second.ipv4.address -all
```

This means that some two specified IPv4 addresses may be used to send e-mails (e.g. the e-mail server and the web server), and nothing else (because “minus” all others are added). There exists a sequence to determine that some third-party addresses are allowed for this purpose too.


Any receiver of an e-mail originated in our domain is able to verify if the sender IP address of the packet with the e-mail is listed in the SPF TXT record. There are special tools for this purpose, and it is possible to use `dig` or similar tools for this reason as well (to request the TXT records).


 **DKIM (Domain Keys Identified Mail)** is an e-mail authentication method based on certificates. An e-mail server has its certificate (a pair of a public and private key, signed by some certificate authority). The public key is stored in the TXT record with the following content:

```
v=DKIM1; k=rsa; p=public_key_of_the_given_e-mail_server
```

As we can see, the algorithm RSA is used for digital signatures, and the public key is the part of the record as well. When sending an e-mail, the server counts the digital signature of the e-mail using the private key, and the signature is included into the header with other information (e.g. algorithms).

Any receiver of the e-mail is able to get the public key from the appropriate TXT record and verify the digital signature.

 **DANE (DNS-based Authentication of Named Entities)** is a protocol that binds TLS certificates and DNS using DNSSEC, standardized as RFC 6698. DANE uses the TLSA records in DNS to store certificates. These certificates can be used to verify the trustworthiness of servers (any type – web servers, mail servers, etc.) in a given domain. It is either the alternative to PKI, or it is a PKI supplement. DNSSEC can be used to validate TLSA records.

 **DNS RPZ (Resource Policy Zone, DNS firewall)** is technique used to protect users from choosing “bad” addresses, mostly used to protect users from accessing known sites infected with mali-

cious code. This functionality should be supported in the product deployed as DNS server, and then we need a *provider of reputation data* providing data about the infected domains.



Additional information:

- <https://www.validity.com/how-to-build-your-spf-record-in-5-simple-steps/>
- <https://www.dmarcanalyzer.com/spf/>
- <https://support.google.com/a/answer/33786?hl=en>
- <https://help.returnpath.com/hc/en-us/articles/222481088-DKIM-DNS-record-overview>
- <https://www.dmarcanalyzer.com/dkim/>
- <https://www.dnswatch.info/dkim/create-dns-record>
- <https://www.internetsociety.org/resources/deploy360/dane/>
- <https://dnssrpz.info/>
- <https://kb.isc.org/docs/aa-00525>



5.2.8 WHOIS Database

As mentioned above, each domain has its own responsible administrator. How to find information about a domain including responsibility? The *WHOIS* service is used for this.

The WHOIS database is distributed among the responsible organizations – the individual RIR providers (each maintains its own database for its own domain and subdomains), similarly, the information is distributed below (LIR, etc.).



Example

If we want to search for an SLD domain within the Czech TLD domain, we look in the WHOIS database maintained by the main Czech registrar CZ.NIC. If we want to search for a TLD domain from Europe or most of Asia, we look in the WHOIS database of the RIPE registrar.

However, it is often easier to use “general services” servers that are not regionally limited, such as <http://www.whois.com/whois/> or <http://www.whois-search.com/>.



Task

Find information about the following domains:

- `slu.cz`
- `google.com`
- `cloudflare.com`
- `ietf.org`
- `netacad.com`



5.3 QoS

QoS (Quality of Service) is a mechanism for prioritizing given types of traffic in the network. There are methods that offer a guarantee of certain parameters of the service provided. For example, it is possible to guarantee:


- bandwidth for transmission, free network capacity,
- minimum loss of data units (or almost zero loss),
- loss dynamics (i.e. in order to avoid cluster losses, this could have a negative effect on the transmission of multimedia, which tolerates the occasional loss of a data unit),
- delay, latency,
- jitter (dynamics of latency),
- etc.

Common network protocols (e-mail, file transfer, etc.) usually do not require QoS, but on the contrary, VoIP and video transmissions may mind higher delay values, as well as higher jitter values.

Thanks to the IETF, the following two mechanisms have existed since the 1990s:

- IntServ (Integrated Services),
- DiffServ (Differentiated Services).


QoS technology is available in specifications of some WAN technologies such as ATM, Frame Relay, and MPLS, and LAN technologies too. And mainly, one way to ensure QoS in IP networks is DiffServ.

 **IntServ** is an older simple mechanism that works as follows:


- an application which wants to use QoS reserves bandwidth for its transmissions,
- this bandwidth is fully reserved for the application for the entire duration of the reservation, whether used or not,
- it is simplex, so the reservation is valid only in one direction of communication (we call the simplex path by a “flow”).

One direction is usually not a problem, because IntServ is typically used for multimedia transmissions, which are in principle asymmetric, or it is not a problem to establish two simplex connections for both directions.

The reservation should be valid for the entire path, so the protocol should be supported on all nodes through which the path passes. If the reservation path is interrupted somewhere (perhaps because the given protocol is not supported there), the packets will go in the usual way in the given part of the path.


 The best known implementation of the IntServ mechanism is the *RSVP protocol*, which is often used to reserve the communication bandwidth based on the request parameters, for example, as a supplement for the SIP protocol. The purpose of this protocol is not to transport data, but to make reservations for data through the network.


The first version of RSVP is standardized as RFC 2205, and other related RFCs were added with some extensions, cooperation with other technologies, tunneling, etc.

 **Additional information:**

- <https://networklessons.com/quality-of-service/introduction-to-rsvp>
- https://www.juniper.net/documentation/en_US/junos/topics/topic-map/rsvp-overview.html





 **DiffServ** works differently. It does not statically reserve bandwidth or force the application to notify its needs in advance, but instead adds priority information to the packet header, which information is taken into account on network devices in the path of the packet. PDUs are categorized and each category is assigned a priority or, more generally, a handling of network devices.

 In IPv4 and IPv6 headers, it works with 6 bits in the field *DSCP* (Differentiated Service Code Point or DS field – Differentiated Services field), which means 64 possible access classes (in practice, however, only a few of them are used).

In practice the following (groups of) classes are mostly used:

- *Default Forwarding* (DF, Best Effort) – no guarantee, the value DSCP is 0.
- *Expedited Forwarding* (EF, RFC 3246) – guarantee of small delay, loss and jitter. This class is suitable for various real-time services (e.g. multimedia transitions).
- *Assured Forwarding* (AF, RFC 2597, RFC 3260) – works similar to using CIR for Frame Relay. It is a group of several classes with various parameters and within each class with defined precedence. If the marked content exceeds an established rate for the given class and precedence it can be dropped. The higher precedence the higher possibility of dropping packets. Various combinations of a class and precedence have different values for the DSCP field.

 When entering the network, packets are classified, marked with a class. If not (all bits are 0), the Best Effort category remains for the given packet. Classification takes place only when entering the network (i.e. the network using DiffServ, we speak about a *DiffServ domain*), the following routers only use this information.

 Packets are marked based on the given class, as close their source as possible (begin of the DiffServ domain). While IntServ reserves bandwidth for data flows, DiffServ reserves resources for classes. Router has separate queues for different classes, into which it sorts sent packets according to their marking. Some queues have higher priority than others, some queues allow dropping, some of them not.

This method of QoS control is relatively easily compatible with other QoS protocols. At the DiffServ domain boundary (on the ingress edge router), the packet classification described above takes place, or mapping from other QoS technology (ATM, MPLS, etc.). MPLS header has only 3 bits for this purpose (the Experimental field), Ethernet CoS (Class of Service, inside the VLAN tag) and IEEE 802.11 TID (Traffic ID) have 3 bits as well, so the mapping procedure is not 1:1.




Additional information:


- CCNP Self-Study: Understanding and Implementing Quality of Service in Cisco Multilayer Switched Networks [online]. <https://www.ciscopress.com/articles/article.asp?p=170743> (several sections)
- https://www.cisco.com/en/US/technologies/tk543/tk766/technologies_white_paper09186a00800a3e2f.html (about DiffServ)
- http://www.ipinfusion.com/pdf/IP_InfusionQoS_MPLS2.pdf




5.4 VoIP and Videotelephony

5.4.1 Principle

 *VoIP* (Voice over IP, “Internet telephony”, IP telephony) is a technology for transmitting digitized voice in IP packets. A call can also be made between more than two participants. The end devices can be either hardware VoIP phones or software, a VoIP application.

 We can call

- within the provider’s network,
- outside the provider’s network, but still within the data network,
- to a regular telephone number in the public telecommunications network.


 The third option requires a connection via the *VoIP gateway*, which forms the interface between the data and the public telecommunications network. The VoIP gateway is selected as close as possible to the destination, so that the longest possible part of the connection runs over data lines (or so that part of the connection over telecommunication lines is, if possible, “local call”).

At present, we meet comprehensively designed branch exchanges for companies, which can ensure their own VoIP network in this way. However, their competitors are the offers of mobile operators, which are very advantageous, especially for larger companies.


 VoIP usually requires the following criteria:

- Internet connection speed over 128 kbps
- maximum delay on the connection to the provider’s server 150 ms
- maximum jitter 30 ms
- packet loss below 2 %, better about 1 %

Furthermore, a provider can set requirements for the method of NAT implementation, etc. However, these requirements are relative, depending on the total load on the connection (therefore it is important to use QoS) and the usual number of calls on the connection. Although VoIP requires less bandwidth than data services, as constant as possible.

 VoIP protocols provide various functions. The most important function is establishing the connection, other functions are signalling during session and other session management, QoS, security, sound processing (including compression, if needed), etc. Within a session, multiple protocols usually work together to provide these functions.

5.4.2 Protocols

 **SIP (Session Initiation Protocol)** is a plain-text protocol developed by IETF for use on the Internet (RFC 3261) in 1999. It is an application layer protocol (meant in TCP/IP) for creating, maintaining, and terminating interactive sessions, including VoIP. It does not provide QoS on its own, but can work with protocols designed for this purpose.

The tasks of SIP are:

- locate the recipient of the call,
- verify the characteristics of the equipment used,
- ensures data transfer, QoS and security by other protocols.

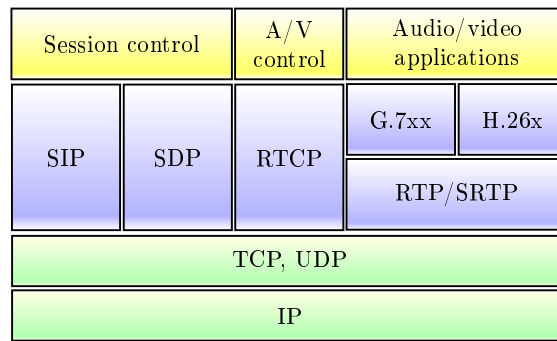


Figure 5.3: Protocol stack for SIP

SIP is *distributed*, it shifts the logics of transmission and functions into end devices as much as possible.

Another important feature of SIP is *flexibility*. It can cooperate with many protocols and, in addition to VoIP, it can also be used for Instant Messaging, and it can work well with mobile technologies. It can use both numeric addresses (phone numbers) and URI (Universal Resource Identifier) – web addressing.

The UDP protocol is mainly used on the transport layer, which means a higher transmission speed (less delay).

H.323 is a set of binary protocols for converting packet protocol signaling to telephone network signaling. It is older than SIP (1996) and is based on signaling common in telecommunication networks. This is a more complex approach, H.323 addresses not only the establishing and maintenance of connections but also QoS and other transmission characteristics. Uses telephone numbers to address devices.

H.323 has a longer delay when establishing a connection and, in addition, its cooperation with other protocols is problematic. This is a largely *centralized* protocol, unlike distributed SIP.

At the transport layer, the TCP protocol is mainly used, which has a higher transmission overhead, i.e. a larger transmission delay.

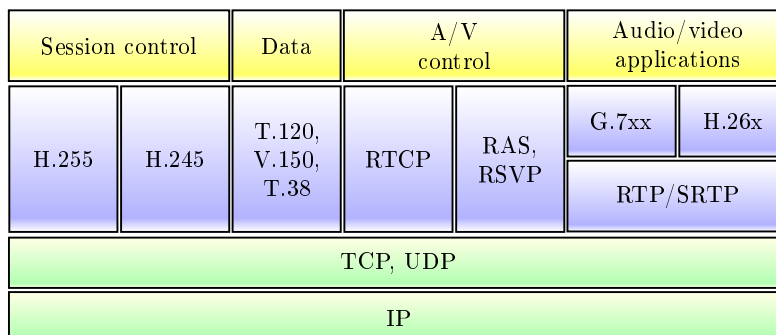


Figure 5.4: Protocol stack for H.323


The both SIP and H.323 are peer-to-peer protocols, however SIP maintains distributed and H.323 is centralized architecture.

Additional protocols. As mentioned above (and as we can see in Figures 5.3 and 5.4), there are several protocols cooperating with SIP and H.323.

- RTP (Real-time Transport Protocol) complements UDP's real-time fast packetization activity, synchronizes traffic, and allows to detect a lost packet or incorrect packet order.

- SRTP (Secure RTP) is secured variant of the RTP protocol.
- RTCP (Real-time Transport Control Protocol) provides feedback for the RTP protocol. Using RTCP, the data flow can be regulated, for example, dynamically changing the transmission rate according to the requirements of the receiving node.
- SDP (Session Description Protocol) is used by the SIP protocol to negotiate multimedia communication parameters.
- RAS (Registration, Admission, Status) – a support protocol for negotiating and maintaining a connection when using H.323. It is part of the H.255.0 protocol used for multimedia communication administration.
- RSVP (Resource Reservation Protocol) is used to reserve resources, for QoS control.
- G.711, G.729, G.723.1, etc. – audio codecs.
- H.261, H.263, etc. – video codecs.
- T.120 – conference data transmissions.


Other protocols include STUN or TURN to support NAT, SCCP on Cisco devices, IAX on Asterisk, and others.

 **Other session initialization protocols.** Besides SIP and H.323 there are other protocols with similar purpose.

MGCP (Media Gateway Control Protocol, H.248) is a plain-text protocol for session management and providing signalization in VoIP a multimedia conferencing, standardized as RFC 3435. Unlike SIP and H.323, this protocol is not peer-to-peer, it implements master-slave communication. The master is called *Call Agent*, slaves are *Media Gateways*. The MGCP communication architecture is centralized with most functions implemented in master, and uses other additional protocols (e.g. SDP or RTP from the above listed protocols).

SCCP (Skinny Client Control Protocol, SKINNY) is proprietary Cisco protocol with the similar communication architecture as MGCP: master-slave communication with a call agent as the central point (CallManager in older implementations, Unified Communications Manager in newer implementations) and media gateways (often in IP phones). Although a proprietary solution, there are implementations outside of Cisco products, an open-source implementation of the SCCP call agent is even in Asterisk.

Skype protocol is a peer-to-peer protocol with hierarchical communication structure: there is one login server, then a layer of supernodes, and a layer of ordinary nodes (client devices or client software). Ordinary nodes keep a host table with IP addresses and ports of reachable supernodes. The communication structure and directory (with communication related data) is decentralized and distributed to supernodes. Unlike others, the Skype protocol is proprietary, closed-source and any other implementations (not MS) are considered illegal by the US law, they were obtained by reverse engineering. This protocol has been deprecated and in 2014 replaced by *MSNP24* (Microsoft Notification Protocol 24) from MSN Messenger (Windows Live Messenger). Nowadays, Skype and MS Teams use SIP.


 **Additional information:**

- <https://www.voip-info.org/sip/>
- https://www.tutorialspoint.com/session_initiation_protocol/session_initiation_protocol_introduction.htm

- <http://www.netlab.tkk.fi/opetus/s38130/k01/Papers/He-H323.pdf>
- https://sc1.checkpoint.com/documents/R80.40/WebAdminGuides/EN/CP_R80.40_VoIP_AdminGuide/Topics-VOIPG/87727.htm?tocpath=H.323-Based%20VoIP%7C_____0
- https://sc1.checkpoint.com/documents/R76/CP_R76_VoIP_WebAdmin/87748.htm
- <https://ribboncommunications.com/company/get-help/glossary/media-gateway-control-protocol>
- https://sc1.checkpoint.com/documents/R76/CP_R76_VoIP_WebAdmin/87758.htm
- https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/cusrst/admin/sccp_sip_srst/configuration/guide/SCCP_and_SIP_SRST_Admin_Guide/srst_setting_up_using_sccp.html
- <https://docs.microsoft.com/en-us/microsoftteams/direct-routing-protocols>





5.4.3 Private Branch Exchange


 PBX (Private Branch Exchange) is a private telephone network used within an organization. PBXs are used to handle internal calls within organizations and also to connect calls between internal and external telecommunications networks. They are actually a connection point to the network of the service provider.


PBXs are analog and/or digital, at present we usually encounter digital ones. Analog PBXs can be connected to regular PSTN telephone lines, digital exchanges also to data networks providing VoIP or wireless networks (DECT). The provided services are not only point-to-point calling and voice gateway, but usually chat/IRC, conference calls, videocalls, mobile apps, integration to CRM systems, etc.

A PBX exchange can be either directly a hardware device or software installed on a server or other device, or a cloud software. Let's look at several software solutions (these are server applications).

 **Asterisk** is a popular open-source software for creating VoIP exchanges. It is configured via the web interface, there are also several GUIs (e.g. FreePBX), and the process can be accessed via APIs from applications written in common programming languages. Asterisk can also be found in many commercial hardware and software PBXs (including the commercial FreePBX project).

 **Cisco Unified Communications Manager** (older: Cisco CallManager) can be installed on devices that support this product (usually Cisco servers). It is a commercial product with generally good features.

 **SIPFoundry** is a SIP-related open-source, but commercial system similar to Asterisk, with quite wide portfolio of services and commercial support. The company propagates the term “communications network” instead of PBX.

 **Elastix** is an open-source project providing a software-based cloud PBX, either hosted or installed in own company private cloud.

Task

At the <https://getvoip.com/blog/2016/09/23/best-open-source-pbx-software/> website, you will find a brief description of several PBXs with a comparison table. First of all, go through the table and get an idea of the parameters that are considered when purchasing a PBX.



**Additional information:**

- <https://www.voip-info.org/voip-pbx-and-servers/>
- <https://getvoip.com/blog/2016/09/23/best-open-source-pbx-software/>
- <https://www.3cx.com/pbx/pbx-phone-system/>
- <http://www.asterisk.org/>, <http://www.asteriskguru.com/tutorials/>
- <http://www.freepbx.org/>
- <https://www.cisco.com/c/en/us/products/unified-communications/unified-communications-manager-call-manager/index.html>
- https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/cucm/admin/9_0_1/ccmsys/CUCM_BK_CD2F83FA_00_cucm-system-guide-90/CUCM_BK_CD2F83FA_00_system-guide_chapter_0100111.html



5.4.4 Videotelephony and Videoconferencies



Video Telephony is a conversation between two or more participants while watching a synchronized video stream (usually a video camera image) – it is transmission of synchronized audio and video signals. Today, it is usually not so much a separate device (videophones), we encounter mainly software implementation. The video telephony function is also usually implemented in PBX software.

The most used standards for video coding and decoding (codecs) used for videotelephony are H.264 (MPEG-4/AVC) and its successor H.265 (HEVC, High Efficiency Video Coding, MPEG-H). The both these codecs are intended for video compression optimized for fast transmission, but the second one, HEVC, offers cca double data compression ratio with the same or better video quality and the same bit rate.



While video telephony requires a point-to-point connection, *video conferencing* is already a more demanding way to communicate. These are multipoint-to-multipoint connections (actually mapped to point-to-multipoint communication), the signal is propagated in the form of multicast. We can consider videotelephony as a special simpler case of video conferencing. There are solutions using a web application, a mobile application, a desktop application, or a combination of all. The transmitted signal is usually encrypted.

Audio signal transmission requires the smallest bandwidth (up to several tens of kbps). Higher bandwidth is required for image/video transmission even in lower quality. In addition to audio and video transmission, there are other ways to use the capacity of multimedia transmissions, such as *shared whiteboard* (participants “draw” on the whiteboard, which is visible and accessible to all participants in the videoconference).





MS Teams is a simple cloud-based video conferencing system. It cannot be configured very broadly, but the basic functions include: creating teams with channels, private channels, scheduling and setting up video conferencing calls, recording them, sharing the desktop, sharing files, whiteboard functions. A user is either in the role of channel owner (teacher) or in the role of ordinary member, or guest.





BigBlueButton (BBB) is an open-source project enabling the operation of a video conferencing system in the own private cloud, in fact it is a modified and enriched Linux distribution (it can


also be obtained in the form of a virtual machine for VirtualBox or VMWare Fusion on MacOS). It was created in a university environment and is primarily intended for online teaching. BBB can do video conferencing, content sharing, desktop sharing, chat, shared whiteboard. There are two roles: moderator (teacher) and viewer (student).


 **Google Meet and Google Zoom** are two video conferencing products by Google, very similar in their features. They both have freely available limited versions, functions of video calls, video conferencing, screen sharing, integration of other Google services (different in Meet vs. Zoom). Zoom seems a bit better for education, but the both have education specific solutions.


 **Apache OpenMeetings** is a solution for the commercial sphere rather than for education, but it is also a video conferencing system. It was created by extending the Red5 media server and offers video conferencing, meetings, desktop sharing, document sharing, etc. It is integrated in some CMS systems.

 **Cisco Webex Meetings** is a commercial solution, the full version can handle video conferencing with up to 200 participants. The free version has certain limits, but it is also usable when we do not have high demands. Webex can be used in the form of a web application, but the desktop application offers a wider range of configuration options. Overall, Webex is perhaps the most widely configurable solution.

 **Jitsi Meet** is a simple open-source solution for smaller teams, which is built on a web interface, so it does not depend on the operating system and there is no need to install anything (but there is also a mobile application). It offers common features that we expect from video conferencing tools, including recording, screen or window sharing, etc.

 **MBone** is a distributed system composed of freely available tools. It requires a suitably multimedia computer connected to the network, it works over the IP protocol. It supports a wide range of operating systems from Windows to various UNIX-like systems. It allows to organize video conferences, join an existing video conference, transfer audio and video, share text and multimedia (shared whiteboard).

 **VRVS** (Virtual Room Videoconferencing System) is a comprehensive free system provided in the form of a service. It is necessary to have either MBone or H.323 support installed. To run VRVS, we need a computer running Solaris or Linux (restrictions do not apply to client workstations, there is an important version of the Java Virtual Machine). VRVS is very well documented. Recently, VRVS has a problem with availability.

 **Ekiga** (GnomeMeeting) is a free VoIP and video conferencing software developed within the Gnome project. It also communicates with other clients supporting the SIP or H.323 protocol. The client side is available for both Linux and Windows.



Additional information:

- <https://www.techradar.com/best/best-video-conferencing-software>
- <https://www.businessinsider.com/google-meet-vs-zoom>
- <https://bigbluebutton.org/>
- <https://www.arxys.com/video-surveillance-h264-vs-h265/> <https://ucvnext.org/2017/09/examining-network-traffic-for-microsoft-teams-in-office365/>

- <https://openmeetings.apache.org/>
- <https://www.webex.com/>
- <https://jitsi.org/jitsi-meet/>
- <http://www.ekiga.org/>




Task


In fact, there are many more video conferencing tools. Read about other such tools on <https://www.techradar.com/best/best-video-conferencing-software>, select one that is not listed in this section, and learn more about it.




Chapter 6

Network Management and Monitoring

 *Preview:* This chapter is focused on computer network management and monitoring techniques and tools. We will deal with network monitoring and management, including common protocols, architectures, principles and tools. We will continue in topic of network monitoring in the next chapter. The last section is intended

 *Keywords:* Network management, NMS, manager, agent, probe, CMIP, SNMP, MIB-II, RMON, Syslog, network monitoring, Snort, NetFlow, VPN, IPSec, GRE, L2TP, OpenVPN, SSH, MPLS VPN, IP-in-IP, VPLS, DMVPN.

 *Goal:* The aim of this chapter is to understand principles of network monitoring and management. The goal is to understand network management protocols, network management systems and their architecture, and the corresponding operations, techniques and software. The important part of network management is network monitoring, thus the goal of this chapter is to get acquainted with the basics of monitoring. In this chapter, we also discuss various VPN solutions, so students will learn the principles of VPN and gain an overview of existing tools.


6.1 Network Management


6.1.1 NMS

Network management includes the following activities:

- supervision, control (monitoring) on the physical layer (media status, etc.), data-link layer (error rate of frames, etc.) and higher, interpretation of collected data,
- diagnosis and troubleshooting, fault prevention,
- management of individual network elements, making topology changes (for example, addition of a new node),
- resource accounting.


Within the ISO/OSI model, most of these activities take place at the application layer, always in some form on all network nodes.

 *Network management* is a service that uses a number of tools, applications and devices to monitor, maintain and secure the network, with the greatest possible degree of automation.

 We distinguish *control* and *controlled entities*. Controlled entities send messages containing reports of events or problems, control entities respond to them adequately, for example

- notify the operator by e-mail or otherwise,
- write the event to a LOG file (event logging),
- shut down or restart the system, log off a “problematic” user, kill a “problematic” process,
- perform automatic adjustment of the system configuration, reset the specified limits.


Software agents (managed entities) collect information from end devices and send to control entities.

 **Network Management System (NMS)** is a network management system including control entities, database and network management protocols. The best known protocols for NMS:

- SNMP (Simple Network Management Protocol) in TCP/IP,
- CMIP (Common Management Information Protocol) in ISO/OSI.


6.1.2 ISO/OSI Network Management Model


Network management can be divided into several areas.


 **Performance management.** The purpose is to ensure that network performance is at an acceptable level. These include variables such as network throughput, user response times, optimal use of cables and other communication paths and elements, etc. It includes:

1. collection of related data,
2. data analysis and level determination for normal operation (baseline),
3. determination of the upper (or for some quantities lower) limit, with the fact that exceeding this limit is indicated as a problem that needs to be solved.


When the set limits are exceeded, the NMS is informed.

 **Configuration management** means monitoring configuration of the network and connected systems (hardware and software) and influencing them. A rule is created for each monitored variable (version of operating system with updates, version of the TCP/IP protocol stack, etc.), and if it is not met, consequences must be drawn.


 **Accounting management** means defining parameters for users and user groups. Appropriate regulation minimizes network problems, especially when using resources (hardware, memory space, etc.), and maximizes the fairness of network resource usage relative to other users.

 **Fault management** represents the detection, logging, notification and, if possible, automatic resolution of network problems. This is one of the most important modules of network management systems, because undetected errors can cause communication slowdowns, data loss or a generally unacceptable worsening in overall network traffic (or network crashes).


The module uses the data obtained and stored within other network management functions, detects possible problems (to be captured as soon as possible), designs and tests solutions.

 **Security management** means controlling access to resources on the network according to established rules, preventing damage to the system (whether intentional or unintentional) and the transmission of sensitive information.


6.1.3 ISO/OSI Network Management

 It is a centralized system based on the *CMIP* protocol (Common Management Information Protocol). It is an object model (works with objects and their properties – attributes, operations, relation to other objects, uses ISA hierarchy):

- *manager* – software on the network management station (central),
- *agent* – software on individual nodes of the network, sends reports on (extraordinary) events to the manager,
- *MIB* (Management Information Base) – object database of controlled objects, binary in this model.


 *MIB* is therefore an object database of controlled objects. For each object is here


- name, class object,
- attributes as ordered as the type–value pairs,
- behaviour – reaction expected and actual to control operations,
- reports – about events related to the object, including information on what is to be communicated by the agent to the manager,
- list of operations that can be performed with the object (for the class).

 *CMIP* works at the application layer of network nodes. When communicating, it uses a connection-oriented service (i.e. if the connection cannot be established completely reliably, *CMIP* cannot communicate with the agents), which can be a disadvantage in a way. The ASN.1 (Abstract Syntax Notation One) language is used to represent the control information, which is common for these purposes (we will also see it in *SNMP*).

6.2 TCP/IP Network Management


6.2.1 SNMP

 In TCP/IP, network management is performed using the *SNMP* (Simple Network Management Protocol) protocol. *SNMP* (or its extensions) is currently the most used protocol for network management and is widely supported in virtually every device that can be connected to the network (including printers, various sensors and analyzers, access points, etc.)).

 *SNMP* exists in several versions. The current version is *SNMPv3*, but it is currently the most used version of *SNMPv2*, more precisely its variant *SNMPv2c*. The main and very important difference between version 3 and the previous ones is the level of communication security. Older versions use only the (text) password – *community string* – for authentication (more precisely two strings: read community string to read, write community string to allow writing).

- *SNMPv1* (RFC 1157) authenticates only using a text-based community string,
- *SNMPv2* (RFC 1901 till 1908) allows digital signatures for authentication (MD5, SHA),
- *SNMPv3* (RFC 2273 till 3415) also can encrypt communication (DEC, AES).

Compared to the previous version, SNMPv2 brought, for example, support for communication between different administrators (in SNMPv1, multiple administrators were not considered) and improved security of communication. Although community strings are still used, new mechanisms for uniquely identifying entities have been added.

 The main feature of SNMP is simplicity. Like CMIP, SNMP uses the *agent-manager* concept, but the functions of these modules are divided differently. Each network node (*managed device*) can contain any number of agents that specialize in a particular activity. There has to be at least one manager in the network, there can be more than one.

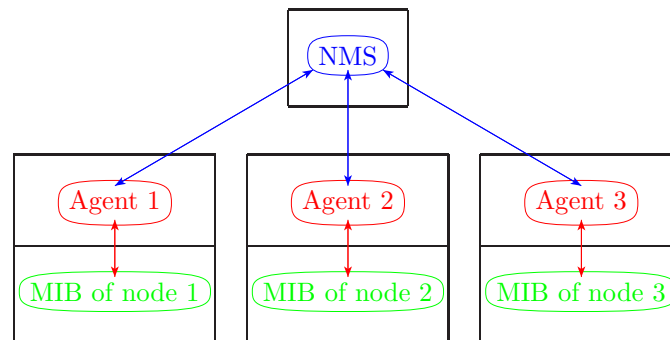




Figure 6.1: Communication in SNMP


 Communication between the agent and the manager takes place mainly using the UDP protocol. However, this protocol does not support delivery acknowledgment (but on the other hand it is fast and sending usually works), so SNMPv2 and higher contain its own delivery control mechanism. SNMP supports two types of communication:

- *Request-Reply* – the activity is on the manager’s side, the manager sends requests to the agents and receives their reactions. According to SNMPv2, a manager sends requests to an agent on port 161, the agent responds from the port 161 to the dynamic port of the manager. According to SNMPv3, the agent listens on another port – number 10 161.
- *Trap* – activity is on the agents side, agents send traps (notifications) to the manager, for example when a defined event occurs, a specified value is exceeded, the network topology changes (for example, a new node is connected) or at regular intervals. The SNMPv2 agent sends traps to the manager from its dynamic port (with different numbers) to port 162. According to SNMPv3, the manager listens on port 10 162.

 We talk about an NMS group within a managed domain as *community*. Each community is uniquely identified by the string *community name*. In older versions, SNMP is used as a unique identifier during authentication.


SNMP is an asynchronous, transaction-oriented, client/server type (communication is usually a question-answer type) protocol.

6.2.2 MIB-II

 The *MIB database* is also used, but generally has a different structure, the data is stored in the text form (the MIB handler is programmed in ASN.1, which is typical of TCP/IP for this purpose).

Although it is object-oriented (in the sense that we work with objects), unlike the OSI MIB, it is controlled by attributes (it is not full object-oriented approach).

In fact, we always meet MIB-II, the second version of MIB. In the following text, we will use the name MIB, but it will always be MIB-II.

 The MIB is formed by a tree with a single root. The root contains an “dot value”, its purpose is only to join the branches coming from it. All nodes except the root are assigned a text name and a numeric value – *OID* (Object ID). The OID clearly distinguishes nodes with the same “parent”, see Figure 6.2. The text name is rather intended for “human orientation”, it has no other meaning in the database.

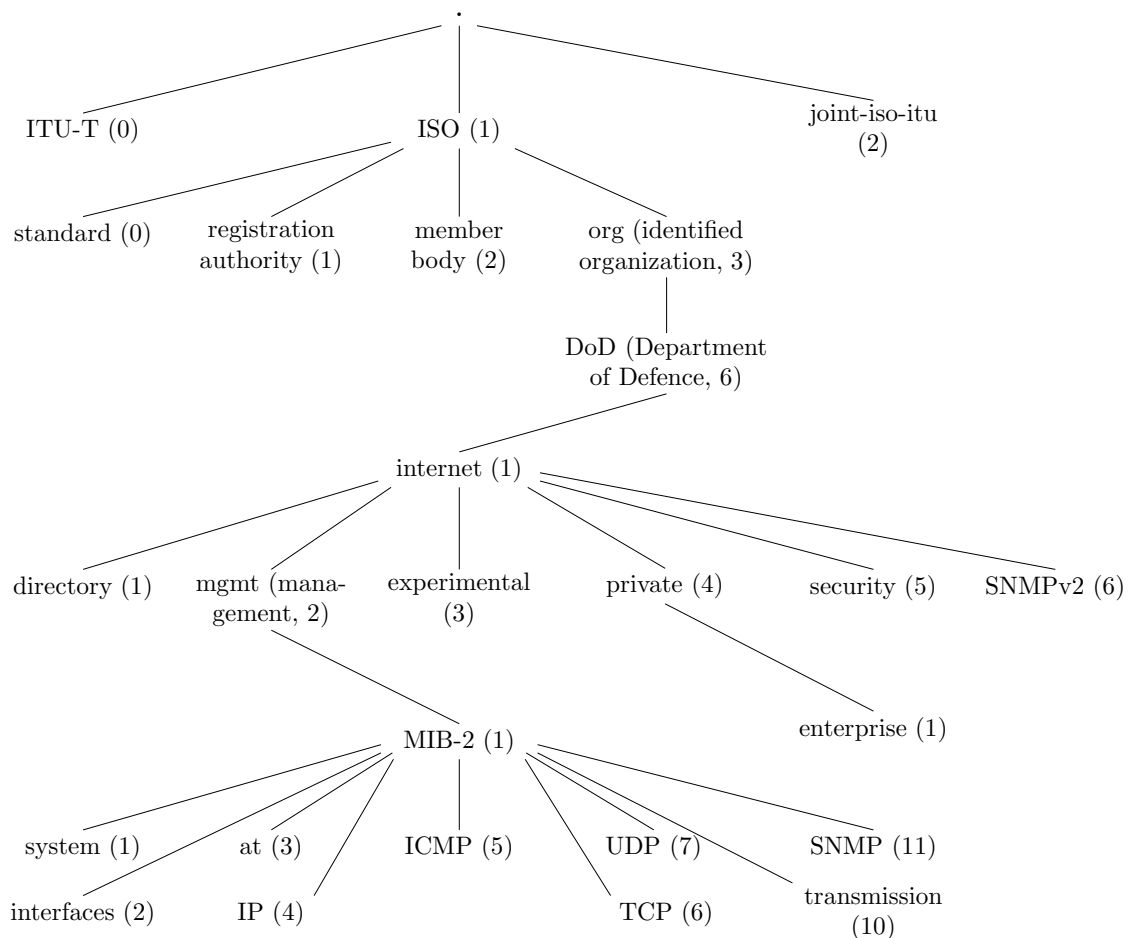



Figure 6.2: SNMP MIB (shortened)

Nodes (objects) in MIB that are direct descendants of the root are named by organizations whose protocols use subtrees of those nodes (for example, ISO or ITU). Some parts (branches) of the MIB are standardized by the IETF organization, others are issued by network equipment manufacturers. Thus, the MIB is usually stored in more than one (text) file, it is distributed in managed devices, so that the processing of the data stored there is sufficiently flexible and the database is scalable.

 An address of an object in MIB is a sequence of either

- *node names* on the way from the root of the tree; for people, objects are separated by a slash or a dot, or

- *node OID numbers* on the way from the root ; for anyone/anything else, objects are separated by a dot (including the empty root name, so the whole string actually starts with a dot).

Example

For example, according to Figure 6.2, the node *enterprise* intended for company nodes (devices of various vendors) has the following name and OID address:

- .iso.org.dod.internet.private.enterprise
- .1.3.6.1.4.1




Task

According to Figure 6.2, find the full path to the *system* node (in the MIB-2 subtree), express it both in names and numerically.



Each network device, protocol or value (or assigned object) that can be managed via the MIB has its own unique OID address, which is the same in any MIB, ie all OID addresses must be globally unique.

 The leaves of the MIB tree are *scalar objects*, *variables* containing a specific value used for network control purposes (for example, the number of packets passing through a router). Variables can also be arranged in a table (*table objects*). SNMP provides limited ability to move through the table, column by column. Types of variables in MIB:

1. numerals
 - ordinary *Integer*,
 - *Counter* – for counters, when the maximum value is reached, they are reset; used primarily to determine the speed of monitoring changes,
 - *Gauge* – if the monitored quantity exceeds the given limit, the Gauge value remains at the maximum value and does not “overflow” (nor reset),
 - *Time Ticks* – monitors the time (in hundredths of a second) since a specified event, such as when the device starts,
2. *IP Address*,
3. *Octet String* – for character strings,
4. *Object Identifier* – OID string of some object,
5. *table* of above listed value types.

Variables are treated in a somewhat special way. As for a simple variable (one that is not a table), the syntax is

`.path.variable_name.0`

As for a table, instead of the number 0 at the end we use the record index in the table.

Example

The following path leads to the *sysUpTime* variable:

`.iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0` or numeral:

`.1.3.6.1.2.1.1.3.0`

We access the values in tables by substituting the index in the table (beginning with 1) instead of the number 0 at the end, for example the values in the table showing the status of the device ports (up/down, in Ethernet it usually means is or is not something connected) we get as follows:

```
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOperStatus.1
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOperStatus.2
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOperStatus.3
etc., the value is either up(1) or down(2).
```

In *interfaces* there is a simple numeric variable *ifNumber*, in which we have stored the number of interfaces in the system. Obviously, it can be `ifNumber.0 = 2` (one physical interface and one loopback) on a workstation with a simple cheap network card or chip, alternatively some additional products of virtualization software. &

📎 The structure of MIB is quite logical. As mentioned above, in the *mib-2* branch we can find variables that do not directly concern a specific vendor, while in the *private.enterprise* branch there are variables related to products from specific vendors (depending on which specifically we find in the relevant network device). Devices that are not standardized for MIB are usually located in the *experimental* branch.

General information about the device whose MIB we work with can be found mainly in the subtree of the *system* node. The contained variables can be seen in Figure 6.3. There is a description of the device (*sysDescr*), the OID address leading to the *enterprise* branch with more detailed information (*sysObjectID*), the time since the device was turned on in hundredths of a second (*sysUpTime*), contact information for a device administrator (*sysContact*), the device name (*sysName*), information about the physical location of the device (*sysLocation*) and the number of the ISO/OSI layer, which the given device operates in (*sysServices*).

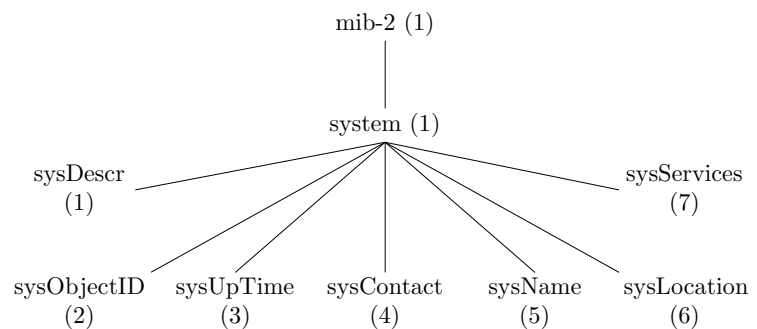


Figure 6.3: Subtree of the node *system*

& Example

Let us focus on the *sysServices* value specifying the ISO/OSI layers on which the device (the one on which the agent's database is stored) operates. The individual bits of this value are set according to the layers support (bits from right to left from the least significant bit).

For example, `sysServices.0 = 10` means that the device is running on L2 and L4 ($2^{2-1} + 2^{4-1}$, So the second and fourth bits from the right are set). It means that it is probably a switch (L2) and contains L4 functionality for management purposes. &

📎 Another useful node in the general part of MIB is the *interfaces* node. Here we can find information about device interfaces, information about individual devices is in the table *ifTable* (which we have already looked at). This table has many columns, for example *ifSpeed* (bandwidth), *ifOperStatus*


(operating status of the interface), number of octets, received/sent/dropped unicast packets, non-unicast packets (*ifOctets*, *ifInUcastPkts*, ...), etc.

6.2.3 SNMP Usage


 SNMP defines the following commands (for agents and managers):

- **get-request** – the SNMP manager requests information from the MIB; gives the name of the variable whose value it requires, gets the value and the name of this variable,
- **get-next-request** – the purpose is the same (request for information from MIB), the administrator also specifies the name of the variable, but obtains the value of the variable and the name of the next variable from the database that is a descendant of the same node (i.e. if the requested the variable ...2.1.1.3, gets information about the existence of the variable ...2.1.1.4, which can be queried subsequently),
- **set-request** – the manager stores the information into MIB,
- **trap** – the agent transmits unsolicited information about the emergency to the manager,
- **get-response** – the agent's response to the previous **get-request** command from the manager, in addition to the values from MIB, it also contains the original query, because SNMP does not allow the manager to match the query/response (stateless behaviour),
- **get-bulk** – similar to **get-request**, but it is possible to request several rows of a table (from SNMPv2),
- **inform** – communication between two managers (from SNMPv2).

The **get-next-request** command used in a suitable loop allows to get the values of the same attributes sequentially over all objects of the same type (column operation).

 Specifically, we can enter these commands (either in text or graphical form) in one of the tools that create an interface to a SNMP manager. For example, the following tools can be used:


- *net-snmp* is an open-source tool working in text mode, under Linux, other UNIX systems and also Windows. It is actually a software package that allows to take full advantage of SNMP in any version, including receiving traps.
- *Kiwi* is a syslog server with SNMP functionality with GUI, very easy to use, with a free edition.
- *MIB Browser* is a freely available application with a graphical interface enabling work with MIB.
- *GNetWatch* is an open-source application with a graphical interface for real-time network monitoring via SNMP and ICMP.

 **Additional information:**

- <https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/snmp/configuration/xs-16/snmp-xe-16-book/nm-snmp-cfg-snmp-support.html>
- <http://www.snmplink.org/>
- <https://www.kiwisyslog.com/free-tools/kiwi-free-syslog-server>
- <http://www.net-snmp.org/>
- <http://www.ks-soft.net/hostmon.eng/mibbrowser/index.htm>
- <http://gnetwatch.sourceforge.net/>



6.2.4 RMON

 RMON (Remote Monitoring, RFC 3577) is a protocol used for network monitoring. It is closely connected with TCP/IP and SNMP, it sends its messages via SNMP. The information is stored in MIB in the *rmon* branch (OID 1.3.6.1.2.1.16). Unlike SNMP agents, in addition to the current state, it can also monitor the history of a given quantity and react based on the consequences.

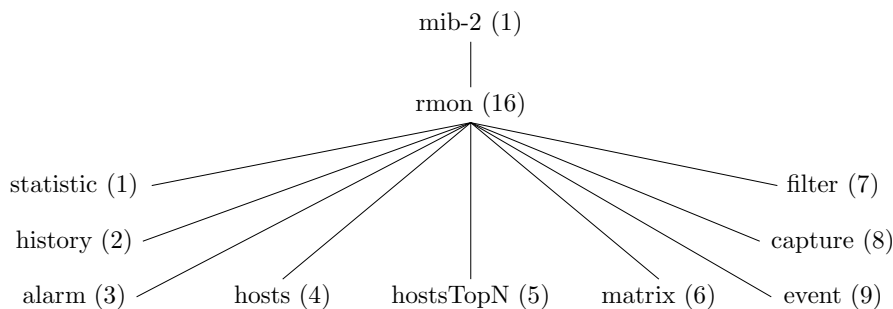




Figure 6.4: The *rmon* branch in MIB-II

 As for the RMON protocol, we distinguish two types of devices in the network:


- *RMON-compliant console manager*,
- *RMON probe* sends information to the manager, a single probe works within single LAN segment.


 RMON works with the so-called *groups*. Each group includes a certain type of information that is monitored. For example, the groups for Ethernet are

- *Statistics* – statistics (current status) of a monitored device (number of sent packets, broadcast and multicast packets, octets, errors in CRC checksums, collisions, etc., as well as counters, for example for the number of packets of size from a given interval),
- *History* – same as statistics, but recorded in history,
- *Alarms* – threshold values are set for some monitored quantities, and when this threshold value is exceeded, a corresponding event is generated,
- *Hosts* – statistics typical for individual nodes in the network (network segment) are kept – address, sent/received packets, errors in CRC checksums and dropped packets or frames,
- *HostsTopN* – similar to the previous one, the nodes are arranged in tables according to a specific monitored quantity,
- *Matrix* – the quantities related to the conversation between two nodes in the network are monitored, for each communicating pair of nodes,
- *Filters* – allow you to define filters for capturing certain packets or generating a certain event when specified packets pass,
- *Packet Capture* – conditions for capturing packets are defined (for example, the size of the buffer for captured packets, the number of captured packets when to trigger an alarm, etc.),
- *Events* – generating and verifying events (the type of event, its description and the time of the last generation of this event by the given device).

Networked devices usually support most of these groups, ideally all. So we follow the variables in MIB that interest us.

The main benefit of RMON is the transfer of a large part of data processing from managers to agents (probes), which also reduces the amount of data transferred in the network.

 Because RMON uses SNMP MIB, we operate with RMON using the same tools as SNMP.


 **Additional information:**


- <https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/rmon/configuration/xe-16/rmon-xe-16-book.pdf>
 - <https://www.solarwindmsp.com/content/rmon-vs-snmp>
-




6.3 Syslog

Syslog (RFC 5423) is a mechanism for logging and related tasks available on every device running (almost) any UNIX-like system, including Linux. The core of the mechanism is the syslogd daemon.

 The *syslogd* daemon reads its input from the `/dev/log` device. It filters this input (selects what interests it) according to the configuration, which is located in the configuration file `/etc/syslog.conf` and then saves the report on events detected according to the settings to one or more LOG files.

 The received data consist of three parts:

1. *category* specifies the type or sender of the event, there are the following categories:
 - auth – users authentication,
 - authpriv – authentication information intended for the administrator,
 - cron – messages coming from the cron daemon (processes scheduling),
 - daemon – messages coming from various daemons,
 - kern – messages coming from kernel,
 - lpr – messages coming from printing subsystem,
 - mail – messages relating to e-mail system,
 - mark – timestamps which are regularly written to the log,
 - news – discussion groups,
 - security – similar to auth,
 - syslog – messages coming from syslog (including different nodes with syslog),
 - user – usually messages from user-mode applications,
 - local0–local7 – the actual meaning can be defined for these categories,
2. *priority* determines the importance of an event:
 - emerg – the system is unusable or seriously compromised (emergency),
 - alert – immediate action is required,
 - crit – critical situation,
 - err – error,
 - warning,
 - notice – normal but significant message,
 - info – informative report,
 - debug – debug message (debugger),
3. the actual *text* of the message.


 Thus, *syslog* gets this type of information. As mentioned above, it has determined in its configuration file what to do with such an entry. The records in this file are in the form

```
category.priority [TAB] destination           this and higher priority
category.=priority [TAB] destination         this priority only
category.!priority [TAB] destination        all priorities except this one and higher values
category.!=priority [TAB] destination       all priorities except this one
```

(priorities can be even more, separated by a semicolon, the same applies to pairs of category.priority). If there is only a dot before the priority value, the setting applies to the specified and all higher priorities. If we want to limit the setting to the specified priority only, we add “=” sign before it. The symbol “!” Means negation, so if it is used, the given priority (and all higher ones) will not be included. Can be combined: “!=” means that only the specified priority will not be included.

We can enter the category with the symbol *. This means that the configuration applies to any category.

There must always be at least one [TAB] key between the priorities and the destination.

 *Destination* specifies where the event should be reported and logged. It can be either a specific log file (default or any other), or a dump to the console or forwarded to another computer on the network. If we want the event to be reported to multiple destinations (for example, displayed to a specific user and in addition saved to a file), we separate the destinations with a comma. Possibilities:

- file – the default is `/var/log/messages` or `/var/log/syslog`, but we can specify file names for various categories or priorities,
- `@server.something.com` or `@IPAddress`,
- `user1, user2, ...` – the event will be reported to the specified user (this can be root, admin, operator, etc., depending on what users we have created), but only when the user is currently logged in,
- * – the event will be reported to all logged in users,
- `@loghost` – we can use it if the destination `loghost` is defined in `/etc/hosts`.

It is also possible to use redirection to a named pipe, from which any specified process can then read, we just specify the file name and write the pipe symbol in front of it:

```
kern.=err      | /var/log/jadro
```

Procedure

The entries in the `/etc/syslog.conf` file might look like:

```
mail.*                /var/log/maillog
    all events from the mail category will be stored to the /var/log/maillog file

security.*;security.!=debug /var/log/secure
    all events from the security category except the debug priority will be stored to the /var/log/secure
    file

cron.*                /var/log/cron
    all events from the cron category will be stored to the /var/log/cron file (that is, events related
    to the scheduled start of processes)
```

```
kern.debug;auth.notice      /dev/console
    the events about the kernel debugging and common but still significant authentication events will
    be sent to the system console

authpriv.*                  /var/log/secure
    all "sensitive" auth events will be stored to the /var/log/secure file


lpr.info                    /var/log/lpd-info
    information messages and and all with higher priority about printing subsystem events will be
    stored to the /var/log/lpd-info file

*.err                       admin,/var/log/errors
    all error and higher-priority messages will be immediately reported to admin (if logged in) and
    stored to the /var/log/errors file.

*.=crit                     /var/log/messages,@loghost,root
    critical events will be stored to the /var/log/messages file, sent to the address defined under the
    alias loghost and at the same time the root will be immediately informed if logged

*.emerg                     *,/var/log/emergency
    all logged-in users are informed about extremely dangerous events and at the same time an entry
    will be added to the /var/log/emergency file
```



 If we want *syslog* to receive messages from other systems, we have to run the *syslogd* daemon with the `-d` option (valid for Linux):

```
/usr/sbin/syslogd -m 0 -r
```

The `-m` option specifies the length of the interval in which *syslogd* responds, i.e. writes to the log that “lives”. Setting to 0 cancels this behaviour. The `-r` option means “remote”, *syslogd* then listens on the port 514 and receives all UDP packets.

Syslog in FreeBSD and relatives uses different syntax:


```
/usr/sbin/syslogd
```

(no parameters, because listening on port 514 is the default behaviour here). On FreeBSD, it is also possible to specify nodes on the network whose UDP packets will be received. OpenBSD, Solaris and others have different syntax (even from this one), so it is always necessary to study the man page:

```
man syslogd
```

It is also possible that we will need to enable listening for the *syslogd* service on the UDP port. We perform it in `/etc/services`, the line `syslog 512/UDP`, but it is likely that such an entry is already there. On devices from which UDP packets are received, the method of sending to this device is set up as described above.


If *syslogd* is currently running and we have changed its configuration (in the `/etc/syslog.conf` file), we need to restart *syslogd* to register the changes in our configuration.

 So far, we have shown what data *syslogd* receives on its input, how it filters it and where it stores the data. We have to look at what format. The output contains a record with the following items:

- the time when the event occurred,
- category (kernel, mail etc.),
- the text of the message.


The priority is not included because it has already been applied during filtering. For example, there may be a record:

```
Feb 21 10:00:28 IOL kernel: device eth0 left promiscuous mode
```

 Manually managing logs can be quite challenging. It is necessary to monitor the contents of files and in addition to monitor their length (delete older records), listening on the UDP port should be sufficiently protected (there is a high risk of DoS attacks, so the firewall is definitely in place). Additional tools can help with some tasks. For example, *logrotate* handles log rotation (including tagging or deleting older records).

There are also more sophisticated variants of *syslog*, such as *syslog-ng* (it can also communicate over TCP and can also handle regular expressions, not just an asterisk), *nsyslogd* (it communicates over TCP/SSL), *Secure Syslog*, *Modular Syslog* and others. The freely available program *NTSyslog* (actually a service that we can install) allows to use *syslog* also in Windows (logs from Windows can be sent to a remote system and evaluated there, for example).

We can use the *logwatch* tool to summarize logs, analyze logs for a specified period, and create a summary report. The program *swatch* (Simple Watch) is useful when, on the contrary, we need to be informed about a certain event as immediately as possible – it detects situations defined by us and informs in a specified way, and because it is written in Perl, it is very flexible a tool that uses regular expressions. And there are tools with GUI for this purpose, such as the above mentioned Kiwi Syslog Server.


 **Additional information:**


- http://linux.about.com/od/commands/l/blcmdl8_syslogd.htm
- <http://books.google.com/books?id=8KUzFB DAT6EC&pg=PA189>
- http://www.softpanorama.org/Logs/Syslog/syslog_configuration_examples.shtml
- <http://ntsyslog.sourceforge.net>
- <http://www.logwatch.org>
- <http://swatch.sourceforge.net>
- <https://www.kiwisyslog.com/kiwi-syslog-server>



6.4 Network Monitoring


6.4.1 Snort


 Snort is simple freely available NIDS (the web site is <http://www.snort.org>) with the open-source license (but for updates we need to register, “immediate” updates are paid).

 Snort works in one of three possible modes:

- *sniffer*: writes data from packet headers to the screen or sends them somewhere,
- *traffic logging*: stores data to disk and/or database,
- *full NIDS* including rules: packet headers analysis.

Snort can either save the output to a LOG file, or send it to *syslog*, send it as an SNMP trap, save it to a database, and even configure some network elements accordingly.

 Snort works on the principle of *signature detection* in combination with *rules*. Many rules are created during installation (there are really a lot of them), and it is also possible to add custom rules according to the specific situation in the network. Using rules generally make the system more powerful (for example, a combination of several “slightly suspicious” actions is *very suspicious*, and unlike signature-only systems, the system generates fewer false-positive alarms.

 The form of the rules is

```
action protocol sourceIP sourcePort direction destIP destPort (options)
```

Action can be, for example, pass, log, alert, drop, etc. This is followed by *packet properties*, Snort recognizes according these properties that the rule should be used (protocol, such as TCP, then the source and destination address and port, as well as the direction of packet transmission represented by the “arrow”). The last part of the rule is *options* that actually describe what happens when Snort captures a packet matching the data in the rule and also what else is to be tested (for example, the TTL field of the packet or ICMP information).

Example


Let us look at the following Snort rule:


```
alert tcp any 3389 -> 81.28.192.0/24 3389 (msg: "TCP paket na Net5");
```

This means that an alert should be generated if a TCP packet is found from any address on port 3389 leading to the network with the given IP address (prefixed CIDR addresses are used) on the same port. A message is to be generated with the alert in the specified form (the message is reported and saved in the log). In fact, the rules tend to be a bit more sophisticated and may also include, for example, a description of the comparison with the signature and other options.

In the example shown, there are specific addresses, but we can also use variables in which Snort has a predefined internal network address and some other addresses:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any
(msg:"ICMP Large ICMP Packet"; dsize: >800;)
```

 Snort distinguishes between administrators and agents. Agents are called *probes* and are located in different areas of the infrastructure, as needed (location is determined by the type of information we want to obtain, it depends on whether the probe is in front of or behind the firewall, in the DMZ, given collision domain, etc.).

 It is a scalable system. If we have multiple probes and there is more traffic on the network (which means a large number of records), it is recommended to use some database system, such as MySQL. There are also frameworks (interfaces) that simplify the access and analysis of Snort data, such as *Prelude*, which is a framework applicable not only to Snort but also to Nessus.

Another system that simplifies the use of Snort is *ACID* (Analysis Console for Intrusion Databases). ACID is interesting because its use is foreseen in the recommendations of the CERT working groups. ACID needs a web server (for example Apache), a functional PHP and a database in which Snort stores records (for example MySQL).

**Additional information:**

- <http://www.snort.org/>
- <http://www.scribd.com/doc/6777057/Snort-Manual>
- <http://www.inguardians.com/research/docs/snortguis.pdf>
- <https://www.prelude-siem.org/projects/prelude/wiki/InstallingAgentThirdpartySnort>
- <http://www.andrew.cmu.edu/user/rdanyliw/snort/snortacid.html>



6.4.2 NetFlow



NetFlow is a protocol developed by Cisco. We meet it mainly on devices from this company, but also on devices from Juniper and some other vendors (more precisely: these devices can export information in the same format as NetFlow). It doesn't have to be just routers and switches, but there are also simple portable devices that act as network monitors in this way.

There are multiple versions of NetFlow with different features. If we need IPv6 support, we have to use at least NetFlow version 9. NetFlow version 10 has been standardized by the IETF as RFC 7011 under the name IPFIX (IP Flow Information Export).



NetFlow works on the principle of *flows*. Flow is what we usually consider to be a complete one-way network conversation. In the case of transmissions realized by connection-oriented protocols (such as TCP), one stream includes not only sending one packet, but communication within the whole connection (but only in one direction, i.e. bidirectional communication is technically in two flows). A single stream also includes, for example, sequences of ICMP packets sent as part of a ping command. Each flow is uniquely described by the following data:

- source IP address,
- destination IP address,
- source port,
- destination port,
- protocol.


Packets that match in all these parameters are sorted into the same flow. Here's why there are different directions of communication in different flows: the opposite directions have swapped addresses and ports in the headers.

Communication can also be split into multiple flows if it takes too long (for example, 30 minutes on Cisco routers), or is inactive for an extended period of time, or the device buffer is full (flows are nowhere to be stored, so the older ones will be deleted).




The information retrieval location (with NetFlow protocol implemented) is called *Observation Point* (also *NetFlow Exporter*). This device detects the necessary information from the passing packets and sends it to the flow collector. *Flow collector* is hardware or software that receives and processes flow data. One flow collector can receive data from multiple observation points (so again it is an agent-manager architecture). The flow collector does not receive information about the flow until this flow has ended.

In addition to the determining parameters, other information is also recorded for each flow: the number of packets in the flow (at the Observation Point this data increases by 1 for each identified packet during flow processing), duration (the start and end timestamps are compared before sending information to the Flow collector), the total amount of data transferred (with each packet this value gradually increases).

 Thanks to the “more concentrated” information that NetFlow allows to obtain, it is possible to detect, for example, a DoS attack (all packets within this attack have common parameters considered, belong to one stream), but not DDoS attack (it comes from lots of sources, usually from computers of a large botnet) or scanning nodes in the network (various destination addresses).

Compared to Snort, NetFlow has the advantage that it does not overwhelm the administrator with information: only one record of the complete (one-way) flow gets into the Flow collector, even if there are hundreds of packets.

 In addition to storing data on the Flow collector, we also need an application that would simplify the analysis of this data. There are multiple such applications, many of them freely available.

- *OSU Flow-Tools* are a freely available text-based software package developed at the University of Ohio. In the package we find tools for data collection on the flow collector and flow data processing.
- *NFDump* is a freely available collector (distributed under the BSD license). There is also an extension with GUI *NFSen*, which is actually a web interface.



Additional information:

- <https://www.flowmon.com/cs/solutions/use-case/netflow-ipfix> (including link to video)
- https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper_0900aecd80406232.html
- <http://www.splintered.net/sw/flow-tools/>
- <http://nfdump.sourceforge.net/>
- <http://nfsen.sourceforge.net/>


**Task**


Follow the link <https://www.flowmon.com/en/solutions/use-case/netflow-ipfix> and find a link to the video at this web page. Watch this video and try to formulate the main features of NetFlow.



6.5 VPN

6.5.1 Principles

 A *Virtual Private Network* (VPN) is a secure connection that travels through an untrusted environment (usually the Internet). It is about creating a tunnel, a communication channel between two points (it can be specific end devices or routers).


 VPN is used in the following cases:

- *Remote Access*: a mobile employee needs secure access to a corporate (local) network in order to be able to access the information system and other secure resources, or it is an employee working at home (Home Office).
- *Site-to-Site* (Network-based): we need to connect several remote local area networks (for example, various branches of a company).

Another situation is communication with a business partner via a secure communication channel, when we do not want to allow access directly into our local network. It can be *site-to-site* or *remote access*, depending on the situation.


In all these cases, it is necessary to build a secure encrypted *tunnel*, through which communication will pass through an untrusted environment – we provide a suitable encapsulation with possible address translation (because packets will go through a foreign network with other address ranges and possibly other protocols), and we also provide encryption (sometimes it is necessary to use an additional solution for this). In the case of a business partner, in addition, we must use a firewall that will allow only traffic allowed for this purpose.

A VPN tunnel usually leads to a corporate network either directly on a router with a firewall at the border of the local network, which is visible on the Internet, or in a demilitarized zone (there is often a VPN concentrator, which is actually a kind of VPN gateway).

 VPN protocols create either *multipoint* tunnels (it is possible to set up a mesh network of tunnels, where each point can communicate directly or indirectly with several other points) or *point-to-point* tunnels (each tunnel must be configured separately, all tunnels have exactly two ends).

 VPN should provide the following:

- authentication – it is necessary to verify the identity of both communicating points (e.g. a user located anywhere and a firewall with VPN support in the company's LAN), or authentication of transmitted PDUs is ensured,
- authorization – determination of specific access privileges,
- ensuring data confidentiality - transmission is always encrypted,
- ensuring data integrity - detection of packet alteration or damage along transmission.

 **Tunneling Problems.** Sometimes there is a problem with the operation of the tunnel – everything seems fine, but if a larger packet is to be transmitted, nothing arrives and an ICMP message with a reason does not come to the source. The reason could be the packet size, which is too close to the MTU values on the path. Remember that too large a packet is either fragmented or dropped along the path (when it cannot be fragmented), and the VPN usually works by encapsulating the original packet/frame into a wrapper and carrier packet - one more headers are added, the frame size increases.

Usually, network devices are expected to pass frames encapsulating a maximum of 1500 B, and this is also calculated by the sending end device: usually the maximum segment size is set at the transport layer so that after adding the IP header this value was not exceeded by adding IP header (only longer PDUs are divided into multiple segments). But the sender does not know that his work will increase by tunneling.


Well, but why isn't it fragmented? IPv6 must not be fragmented along the path, and IPv4 often uses the *Path MTU Discovery* mechanism, which is useful (it detects the MTU for all the path and modifies

the packet to pass on all elements before sending), but automatically disables path fragmentation (sets the DF flag in the IP packet header). We often do not know the problem properly, because the sender is being informed about the discarding of the unfragmentable packet by the message ICMP Destination Unreachable, code 4, and unfortunately these ICMP messages are dropped on some routers (yes, such are the consequences).

So what's the solution? For example, we can specify a smaller value for the maximum segment length on the transport layer. We take the usual minimum MTU (1500) and subtract the length of all headers that are added during encapsulation. For example, a value of 1220 can be used for GRE tunnels.

Next, several common VPN solutions will be introduced. These are protocols working on different ISO/OSI layers, other differences are in the complexity of deployment, (non)need to have an external service provider, and also whether the solution includes encryption and other security mechanisms.

6.5.2 IPsec VPN

 *Internet Protocol Security* (IPsec) allows to create secure point-to-point tunnels at the L3 layer. Provides both tunnel creation and encryption. It works at the network layer, making it transparent for application protocols. It can be used for both site-to-site and remote access solutions.

The tunnel is created by encapsulation as follows:

- inside is the PDU of the transmitted protocol, mostly IP (but can also be IPX, NetBEUI or other), or only its data part where the header is being used according to the third step,
- followed by a wrapping protocol, which is usually IPsec (can be e.g. GRE, PPTP, L2TP or other), the encapsulated PDU is encrypted and a security header is added,
- the network layer carrier protocol (usually IP) encapsulates the PDU created in the previous step so that the packet can be transmitted over the untrusted Internet or another public network.

A bonus is the ability to encapsulate such protocols that are not commonly supported in the external network, or ensure the use of private IP addresses (encapsulated IP packet is addressed by a private destination IP address, but in order to deliver the entire PDU, the external IP header holds a public destination IP address of the corporate border device supporting VPN).

IPsec is also often combined with GRE or L2TP protocols operating at the L2 layer, which is mutually beneficial. GRE and L2TP cannot encrypt, and on the other hand, IPsec has a problem with NAT passing, because modifies the IP header. However, even in this case there is a solution, we can use the mechanism *NAT-T* (NAT Traversal), which encapsulates packets in UDP. It is called IPsec over UDP or IPsec over NAT-T.

 Encryption and encapsulation in IPsec is performed in one of two ways:

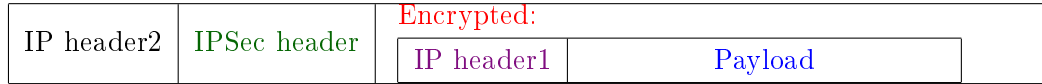
- *Tunnel Mode*: the entire original IP packet is encrypted and the IPsec header is appended (and then of course the carrier protocol header with the new IP address),
- *Transport Mode*: the data portion of the encapsulated packet without a header is encrypted, then an IPsec header is added, preceded by the original header of the encapsulated data.

The tunnel mode is more secure (the whole encapsulated IP header is hidden, encrypted), the transport mode is more flexible (we can use elements from the original IP header, for example for QoS) and packets are shorter (less impact on network throughput). We can see the comparison in Figure 6.5.

Original IP packet:



- *Tunnel Mode:*



- *Transport Mode:*

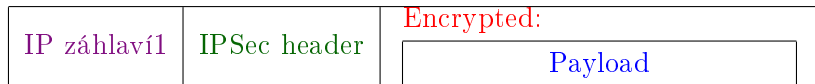




Figure 6.5: PDU before and after encrypting and encapsulating into IPSec tunnel

The tunnel mode is used primarily for site-to-site connections, while the transport mode is more typical of connecting a remote computer to a corporate network (remote access). The transport mode is also used in combination with other tunneling protocols.

 There are two types of *IPSec headers*: AH (Authentication Header) and ESP (Encapsulated Security Payload). While AH is simpler, it only provides authentication and data integrity but not encryption, the ESP header is more secure (provides authentication and encryption). Even both can be used for a single packet: AH/ESP.

 When creating any (including IPSec) tunnel, it is necessary to negotiate the *Security Association* (SA) parameters. IPSec uses the *Internet Key Exchange (IKE)* protocol for this purpose.



Remark:


IPSec is already natively implemented in the IPv6 protocol (this was already the case in the original IPv6 design, IPSec was additionally added to the IPv4 version as a “auxiliary” protocol providing encryption). Therefore, the transition to IPv6 has one bonus: native possibility of IPSec tunnels creation.



Task

The AH and ESP headers have a different structure. Take a look at <http://www.unixwiz.net/techtips/iguide-ipsec.html> and compare both their structure and how they are used (including NAT).



 **Configuration of the IPSec protocol.** IPSec is configured in a similar way as a firewall: we define rules and other parameters (in the so-called policy table). For example, we specify that packets arriving from a specific address (remote extensions) should be processed by the IPSec mechanism (decrypt, etc.), packets outgoing to the same address should be encrypted, IPSec headers added, etc., packets destined for the internal network should be left as they are, etc. The routing table usually needs to be modified as well.

In Linux, it depends on the specific distribution. In Debian-derived distributions, the *Freeswan* package is usually available (or installed), where the configuration is done in the `/etc/ipsec.conf` file

and the `ipsec` command is used. In other distributions we have the *IPSec Tools* package with commands `setkey` and `racoon`, the configuration files are `/etc/racoon/racoon.conf` and `/etc/racoon/setkey.conf`.

In Windows and on network devices configured via the web interface, the configuration is usually done “by mouse”, for example in Windows Server we have a console in *Start – Programs – Administrative Tools – Local Security Settings*. In addition, the `ipsecpol` program is available in Windows Server to define IPSec policies.

The connection is quite simple on the client side. VPN is pre-configured on the server, we basically configure a new network connection on the client: *Create new connection*, choose a connection to the corporate network, VPN, etc. In the wizard we fulfill the IP address of the VPN server with which we want to communicate. Then we configure the IPSec protocol in the connection properties, including security settings.




Additional information:

- <http://www.unixwiz.net/techtips/iguide-ipsec.html>
- <http://www.ipsec-howto.org/ipsec-howto.pdf>
- https://www.cisco.com/c/en/us/td/docs/net_mgmt/vpn_solutions_center/2-0/ip_security/provisioning/guide/IPsecPG1.html
- https://www.juniper.net/documentation/en_US/junos/topics/topic-map/security-ipsec-vpn-overview.html



6.5.3 GRE Tunnels

 The *GRE* (Generic Routing Encapsulation) protocol works at the network layer and creates point-to-point tunnels, typically site-to-site. It is standardized by the IETF as RFC 2784.

The usual procedure is that the original packet is provided with a GRE header (very simple, a few fields) and then a carrier protocol header, usually IP, is added (there is a number 47 in the Protocol/NextHeader field specifying the GRE protocol). Note that there is not a word about encryption in the process, because GRE cannot encrypt.

Its advantage is versatility, it can encapsulate other types of PDUs of the network layer than just IP. It can even encapsulate other layer protocols, including L2 frames. It can also transport multicast and broadcast traffic through the tunnel. Another advantage is that this protocol is supported practically everywhere (in Linux, Windows, on network devices of various vendors).

On the other hand, the disadvantage of GRE is that it does not perform encryption. In practice, GRE is combined with IPSec so that the conversation is encrypted.

Configuration in Linux is simpler than IPSec, we only need built-in tools (command `ip tunnel` and other variants of the command `ip`).




Additional information:

- <https://www.ietf.org/rfc/rfc2784.txt>
- <https://www.cloudflare.com/learning/network-layer/what-is-gre-tunneling/>
- https://www.cisco.com/c/en/us/td/docs/iosxr/ncs5500/interfaces/61x/b-ncs5500-interfaces-configuration-guide-61x/b-ncs5500-interfaces-configuration-guide-61x_chapter_01101.pdf

- <https://networklessons.com/cisco/ccie-routing-switching/how-to-configure-gre-tunnel-on-cisco-ios-router>
- https://www.juniper.net/documentation/en_US/junos/topics/topic-map/switches-interface-gre.html




6.5.4 L2TP Tunnels

 L2TP (Layer 2 Tunneling Protocol) is a descendant of the older PPTP (Microsoft) and L2F (Cisco) protocols. It is standardized by the IETF, the version L2TPv3 from 2005 is in RFC 3931.

As the name suggests, this protocol works on the L2 layer. The data-link layer VPN protocols are interesting in that they usually make a “loop” to a higher layer: they encapsulate traffic from the upper layer, but they encapsulate themselves in a TCP or UDP segment.

The main purpose of the L2TP protocol is to tunnel (encapsulate) PPP packets (which is used for data transmission over the telephone network, including ADSL/VDSL), and is very popular with various ISPs. It is encapsulated in UDP segments, it does not use TCP (the tunnel has no support at the L4 layer).

 Like GRE, L2TP does not perform encryption, so it is usually combined with IPsec in transport mode, which is standardized in RFC 3193. So for the packet to be transmitted:

- PPP header and footer are first added at L2 (PPP ensures connection establishment and control, basic authentication, encapsulated protocol identification in the header, checksum in the trailer), but this step is not mandatory (L2TP can encapsulate IP packages too),
- then the L2TP header is added (the parameters for the tunnel and the session and other data are in the header),
- this L2TP frame is encapsulated in UDP (so loop upwards),
- then an IPsec header and a trailer can be added, followed by a carrier protocol header, usually IP.

L2TP is supported on both Linux and Windows. In Linux it is recommended to use OpenL2TP (using IPsec), in Windows there is an L2TP/IPsec client for the transport mode.




Remark:

In Windows, we can also meet with the Secure Socket Tunneling Protocol (SSTP), which transmits PPP or L2TP encrypted using the SSL protocol.



6.5.5 OpenVPN


 An interesting implementation of VPN is the project *OpenVPN*. It runs on most known UNIX-like systems, including Linux and MacOS, there is also a variant for Windows. It is an open remote access solution that handles both tunnel creation and encryption, only transmitted data is encrypted.

This solution works at higher layers (above L3), which means that, for example, there is no need to solve problems with NAT or other IP address mappings. SSL or TLS protocol is used for encryption, authentication and key management, mostly UDP is used at the transport layer, but it is also possible to use TCP. The TLS protocol is standardized, but the entire OpenVPN solution is not.


While IPsec is a network solution (transparent to various applications), SSL/TLS is an application solution (i.e. it has to be supported by the specific application whose communication is to be encrypted). Web browsers have built-in SSL/TLS support, so there are no problems with the functionality, at least when communicating over HTTP or similar. If the application does not support SSL/TLS, this can be solved, for example, with *STunnel*.

SSL is considered a less secure but more flexible solution than IPsec (for example, IPsec has problems with NAT, while SSL handles it without any problems).

6.5.6 SSH

 *SSH* (Secure Shell) is a more secure variant of the Telnet protocol, but in addition to securing communication itself, it has additional functionality compared to Telnet. It is used for secure access to a remote device in various directions. In addition to enabling secure configuration, it also performs a similar (but secure) role as FTP (file transfer), allows monitoring processes and resources on a remote system, creating encrypted VPN tunnels, etc. SSH version 2, issued in 2006 by the IETF (RFC 4254), is currently in use.

This is a client-server communication via TCP (connection is established), there is a port 22 on the server side. Due to the fact that this is mainly about security, it is better to configure a different port than 22 on the server. It depends on the specific product, it is usually about a change in a configuration file. However, if we change the port number for SSH on the server, the client has to use this number when establishing a connection (and thus be informed about it).

 The best known implementation of the SSH protocol is the open-source project *OpenSSH*. There is a client and server variant for this product, and it is usually already installed on UNIX-like systems including Linux and MacOS. There is a client variant for Windows and it is used to access UNIX servers.


For Windows and Debian Linux systems, we also have the program *PuTTY*, which, however, exists only in the client variant. Again, it is used for remote access to UNIX servers.

When using SSH, we should use secure versions of data transfer software. In UNIX (including Linux) we use the commands `scp` (copy, instead of `cp`) and `sftp` (instead of `ftp`). There is also a *WinSCP* file manager that implements the equivalent of these commands.

 The SSH conversation looks like this:

- First a TCP connection is established, the client starts.
- The SSH client and the server introduce themselves (communicate their version to each other), the server starts.
- The server tells the client which encryption algorithms it manages, and the client then chooses between them in response.
- The security procedure follows – the encryption chain, etc. will be replaced, according to the selected algorithm. The purpose is to ensure that the following communication does not interfere with anyone who could eavesdrop on the traffic.
- After completing the security procedure, further traffic is encrypted, including any authentication if required.

When we establish a connection with a server via SSH for the first time (we have not yet communicated with it from our device), a public key is sent (not yet secured) and appears on the display. If we know the public key of the server (we have it “from elsewhere”), we can compare and check.

 SSH creates remote access, point-to-point, tunnels at higher layers. Tunneling works on the principle of forwarding traffic to the ports to which the SSH tunnel is connected. The SSH server is listening on the TCP port 22.




Additional information:

- <http://blog.trackets.com/2014/05/17/ssh-tunnel-local-and-remote-port-forwarding-explained-with-examples.html>
- <https://www.ietf.org/rfc/rfc4254.txt>
- <http://www.debianhelp.co.uk/ssh.htm>



6.5.7 MPLS VPN

MPLS networks are mostly used for the implementation of interconnection of corporate branches through a tunnel (or a company and a business partner), i.e. a VPN of the site-to-site type, the connections are of the point-to-point type. We have already become acquainted with MPLS networks and we know that a header stack is used, one of which is intended for VPNs. The solution works at the border of layers L2 and L3.

 We distinguish the following devices:


- CE (Customer Edge) – the border node of the customer’s network, through which the data is transmitted to the MPLS tunnel,
- PE (Provider Edge) – the border node of the provider’s network is directly connected to the CE node,
- P (Provider) – internal nodes of the MPLS network provider, basically core MPLS routers.

CE nodes are not part of the MPLS network, only TCP/IP protocols are required on them. PE nodes are border (LER) nodes of the MPLS network and implement the BGP protocol, resp. iBGP (distribution of tables between nodes). Multiple CE nodes can be connected to a single PE node.

The IP packet coming from CE on the PE node is provided with two MPLS headers:

- the internal header ($B = 1$) will be used only at the other end of the tunnel, determined by the CE recipient,
- the outer header ($B = 0$) specifies the path in the MPLS network through P nodes.

The *VRP table* (VPN Routing and Forwarding Table) is a table of routing information for specific VPNs and networks. Located on PE nodes, one PE can contain multiple VRFs.

 There are two basic solutions for MPLS VPN: MPLS Layer-3 VPN, MPLS Layer-2 VPN. Both IP packets and Ethernet frames (Ethernet over MPLS, EoMPLS solution) or PDUs of other protocols on these layers can be encapsulated.

The disadvantage of MPLS VPN is that we need an external provider of this service (usually an ISP, through whose WAN network our tunnel will run), we cannot create these tunnels ourselves. On the contrary, the advantage is that packets transmitted through the tunnel do not pass through the untrusted Internet, the configuration is on the side of the service provider and the service also includes QoS, so that despite relatively high financial costs, this service finds its customers.

6.5.8 Other VPN Solutions

IP-in-IP is simply encapsulating an IP packet into an IP packet. It can be the same version or different versions, for example, this often solves the transport of IPv6 packets over a part of the network that “does not understand” IPv6 (so we encapsulate the IPv6 packet into an IPv4 packet).

An important feature of IP-in-IP is the change of IP addresses in the header, which is actually one of the features of tunneling: in the outer header, the IP address of the device (router) at the beginning of the tunnel is used as the source, the end address of the tunnel is used as the destination. Unlike NAT (which is actually also address translation), the original header is preserved (NAT interferes with the original header, does not create a new one).

In the case of IPv6, the possibility of encryption is built directly into this mechanism (in the optional IPv6 header there would be an optional security header), in IPv4 we would have to provide encryption differently (for example using the IPSec protocol).

VPLS (Virtual Private LAN Services) is a data-link layer multipoint solution, essentially a multipoint equivalent of EoMPLS, designed for applications that require remote multicast and broadcast communication functionality. A provider of this service actually simulates a switch (we are at L2) connecting remote LAN networks from the client’s point of view.

DMVPN (Dynamic Multipoint VPN) is a solution based on the NHRP (Next Hop Resolution Protocol), which is an extension of the GRE protocol for use on multipoint links. It works on the network layer and does not require an external provider (configuration can be done by ourselves, the transfer usually takes place via the Internet, but in a tunnel). It can be said that this is a multipoint alternative to GRE and IPSec.




Additional information:


- https://www.cisco.com/c/en/us/td/docs/net_mgmt/vpn_solutions_center/2-0/installation/guide/IG20_1.pdf
- <http://www.ipsec-howto.org/ipsec-howto.pdf>
- <http://openvpn.net/>
- <http://www.openl2tp.org/documentation>
- <https://www.cisco.com/c/en/us/support/docs/multiprotocol-label-switching-mpls/mpls/29828-mplsvpnte.html>




Network Security


 *Preview:* This chapter is focused on computer network security. We will deal with types of network attacks, network devices and software to ensure network security including firewalls, IDS/IPS and network taps, and with specialized software tools for network management in respect to security.

We assume that the reader already has at least basic knowledge in the field of computer security (about malicious software, hackers, etc.).

 *Keywords:* Asset, threat, vulnerability, attack surface, exploit, perimeter, network analyzer, firewall, IDS/IPS, SPI, DPI, packet filter, proxy, network management software, SIEM.

 *Goal:* The aim of this chapter is to understand principles of network security, including various types of network attacks. The next goal is to understand firewalls and other security related devices operation.

7.1 Terms in Network Security

 The following terms related to network security are used in the next sections of this chapter:

asset is an object or subject to protect: people, data, end devices, network devices, everything related to and relevant to the organization,

threat is potential danger to protected assets; threat intelligence is an evidence-based knowledge about hazard to assets,

vulnerability is a weakness in some system or the design of the system, the weakness could be exploited by a threat,

attack surface is a sum of points (or ways) where an attacker could come into a protected system or network, including cloud resources,

exploit is a mechanism which can be misused by an attacker to utilize a vulnerability to compromise an asset,

perimeter is a secured boundary between a company's LAN (intranet) and the public network (internet).


7.2 Networks Attacks

Different types of attacks need to be faced in a computer network, some of which are interconnected. We are mainly interested in the following attacks:

7.2.1 Reconnaissance Attacks


Reconnaissance attacks (recon attacks, mapping, information gathering) are actually a kind of preparation for the following attacks. The hacker obtains as much information as possible (domain information, open ports, used IP address ranges (logical topology), operating systems and their versions, etc.) so the following attack itself can be as successful as possible.

The tools are `nslookup`, `dig`, `whois`, `wireshark`, `nmap`, other network scanning tools,...


 *Network sniffing* is a procedure where a packet sniffer diverts traffic on the network, intercepts packets and obtains information from them (then sends them on to the destination, undelivered packets would mean its detection). The purpose of sniffing is primarily to obtain passwords transmitted in text form and also other interesting information. It is also possible to capture and modify the contents of packets or even read and modify information on nodes in the network (including configuration files or passwords).

Reliable encryption (including network authentication) as well as physical network security are an effective defense, as this type of attack requires physical access to the network. This can be a problem with wireless connection types.

However, a packet sniffer (such as Wireshark) can be legally used by an administrator to monitor network traffic within the own network.

 *ICMP attacks* serve to discover network topology (subnets, hosts), but this kind of attacks can generate a DoS attack as well. The following ICMP message can be used for recon attacks:

- ICMP Echo request and reply,
- ICMP unreachable,
- ICMP mask reply,
- ICMP router discovery, etc.


 **Additional information:**

- <https://www.sciencedirect.com/topics/computer-science/reconnaissance>
- https://owasp.org/www-pdf-archive/ICMP_Attacks.pdf




7.2.2 Access Attacks


Access attacks gain access to systems, escalate privileges, steal or damage data by exploiting vulnerabilities in end or network devices. This type of attack usually follows recon attacks.


 *IP Spoofing* is spoofing of IP address. The communicating node pretends to own an IP address that actually belongs to another node (or does not belong to any node). This is usually the case when an attacker tries to pretend to be a legitimate member of a private network using a range of IP addresses, or tries to impersonate a specific node on the network.


A common use is to misuse some protocols, including SMTP for sending e-mail. It is also used as a means for more sophisticated attacks, such as DoS (in packets that are sent to an infected device, the source address is spoofed).

 *Other types of spoofing:* identity spoofing can be committed in other ways, often involving the hacking of some address tables, for example:

- *ARP Spoofing* means forgery of records in ARP tables on devices (the mechanism of IP address to MAC address mapping is compromised),
- *DNS Spoofing* is spoofing of records in a zone file on a DNS server (the mechanism for mapping a name address to an IP address is compromised).

 *Eavesdropping* is capturing network traffic and gaining information from the traffic. Data is either sent to the original receiver, or sent with modifications, or dropped.


 *Man-in-the-Middle* (MitM) – a hacker gets between two communicating devices and eavesdrops on or even alters communication between them.


 *Password attacks* include various types of gaining access privileges to “interesting” systems. There are dictionary attacks, rainbow table attacks, brute-force attacks, phishing, pharming, network sniffing, social engineering techniques.

Task

Do you know what a strong password is? Are your passwords strong? Do you use a password manager with a strong password generator? If not, consider using it.



 *Session hijacking* means gaining access to the network and using MitM attack to hijack a session, generally any connection requiring authentication (including private Wi-fi networks).

 *SQL Injection* is a type of attack where a hacker succeeds in injecting an SQL statement somewhere where he has nothing to do, or by a suitable choice of the strings “makeing” and sending a command to an SQL server. This is usually an abuse of forms on the web or a string passed over an HTTP GET, where an attacker fills in some “malicious” string containing symbols with special meaning, instead of the regular string. A typical misused symbol is an apostrophe.


This type of attack can be completely defended if the programmer of the web application (i.e. a web interface to some database) ensures careful analysis of user input, and above all it is necessary to have correctly set permissions on the database (for example requests for destructive operations such as DROP, DELETE, but also changes to data or requests for protected information should not be made by a normal user from an unprotected part of the network, i.e. the Internet).


Additional information:

- <https://www.ciscopress.com/articles/article.asp?p=1728833&seqNum=3>
- http://www.cis.syr.edu/~wedu/seed/Book/book_sample_tcp.pdf
- <https://www.utc.edu/center-academic-excellence-cyber-defense/pdfs/course-paper-5620-attacktcpip.pdf>
- <https://unit42.paloaltonetworks.com/dns-tunneling-how-dns-can-be-abused-by-malicious-actors/>




7.2.3 DoS and DDoS Attacks


 *DoS (Denial of Service)* is enforcing the denial of service which then cannot be used by legitimate users. This type of attack occurs either by exploiting a code error (of a protocol or operating system), or by network congestion, or by overflow of system tables, eventually by spoofed network status packets. The attacked server is overwhelmed with connection requests (TCP handshake) or data requests that it is unable to handle, and therefore subsequent traffic is ignored or delayed.

 *DDoS (Distributed DoS)* is a variant of the previous type of attack, against which it is almost impossible to defend, because there are multiple (large amount) of involuntary “assistant” attackers.

Botnets are a common involuntary “participant” in DDoS attacks. *Bot* is an attacked computer controlled remotely by a hacker (without permission and usually also without the knowledge of the legitimate owner). Botnets, even very large ones (or their services), are “sold” on the black market, among other things, for DDoS attacks.

 A DoS attack can also be performed using otherwise quite useful mechanisms. For example, the *Ping Flood* attack (read [flad]) takes place in such a way that an attacked device is flooded with ICMP Echo Request messages (these are those sent by the `ping` command). This overwhelms both its input capacity (receiving) and its output capacity (sending) when it tries to respond.

To hide the identity, the attacker spoofs the source IP address in IP packets encapsulating these messages. The DDoS variant of the attack means that ICMP messages are sent by devices connected in the botnet.

 *Smurf Attack* has an even higher distribution degree. A large number of devices (typically servers and routers) receive ICMP Echo Request messages whose source address is spoofed – set to the victim’s address. These devices respond with ICMP Echo Reply messages to the victim’s address, overflowing their input capacity.



Additional information:

- <https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/>
- <https://www.imperva.com/blog/security-glossary-top-12-ddos-attack-types-need-know/>



Remark:

There are many more types of attacks, here is just a brief overview of the most common ones.




Task

Look at <https://www.cloudflare.com/learning/ddos/ping-icmp-flood-ddos-attack/>. This web page explains the Ping flood ICMP attack. In the upper menu there is the item *Common DDoS Attacks*. Expand this menu item and see which of the attacks you have heard about and which you haven’t heard of.



7.2.4 Human Factor

 An “enemy” can even be inside an organization (*insider*), even one that has no idea. Especially recently, *social engineering methods* are often chosen for attacks. Social engineering is a method of extracting the necessary information from a user, even without the use of technology, by misleading him (deceiving him). The user is convinced that he is passing the information to a trusted person for absolutely necessary reasons.

An attacker pretends to be, for example, an employee of the security department, a repairman, an employee of a telephone company, a new colleague, etc. and unobtrusively pulls everything he needs (especially passwords) out of his victims, or technology in the workplace. This happens especially in larger companies, where not all employees are expected to know each other, but contact can also take place “impersonally” via telephone or e-mail.

Many users incomprehensibly trust the e-mail: for example, somebody just sends an e-mail informing a victim that the account may have been compromised, and it is necessary to check the login information so that the administrator can verify that everything is OK, and many users blindly listen.

Social engineering can also be of a “material” form: for example, “forgotten” portable devices, which are irresistible to many people. According to DHS (Department of Homeland Security, <http://www.dhs.gov>), cca 60 % of ordinary users are able to connect a found USB dongle to their computer, although they have no idea if it does not contain malicious software. Similarly, other portable devices can be misused. The best protection is honesty, i.e. passing the found object on to lost property.

Social engineering is currently one of the most used (and also the most effective) methods of obtaining information. This should be kept in mind especially by corporate network administrators and ensure appropriate training for all those who join the corporate network.

7.3 Security Appliances

7.3.1 Network Analyzer


 A *network analyzer* is a device that analyzes network traffic on a link between some devices of a network, it can also be part of another device. A stand-alone network analyzer (portable) can be very small, it can resemble a mobile phone.




Figure 7.1: Network analyzers¹

It works on the physical or higher layers, depending on which characteristics it can monitor. At the physical layer, especially the state of the medium, traffic (load), it can generate its own traffic, at


¹Resource: <http://www.fluketestery.cz/>

higher layers it can also support virtual networks, monitoring the status of available network nodes, monitoring frames or packets, etc.

Network analyzers “reach” only where the data comes from. Thus, it is limited to a single collision domain (i.e. to the nearest switch or router). Therefore, it is advisable to place a network analyzer on a network node that occurs in multiple collision domains, such as a router.

 **Hardware by Fluke** is well known in the field of network testing. Most used devices:


- *LinkRunner* – works on the physical layer; identification of Ethernet link properties: negotiation (10/100/1000 Mb, duplex, etc.), detection of cable failures, including distance to failure, sockets, segment operation, etc.
- *NetTool* – network layer, testing at the physical layer (cabling, negotiation, segment usage), data-link (collision and error frame detection, VLAN, searching for MAC addresses in the network, etc.), network layer (searching for IP addresses in the network, subnets, routers and other sources, ping, etc.), PoE measurement, monitoring of VoIP parameters, etc.
- *AirCheck* – analysis of wireless networks, including channel utilization, detection of connected devices and looking for unauthorized devices.


 **Software network analyzers:** Network analyzers can also be software – they monitor and possibly affect network traffic related to the computer on which they are installed (ideally a server, but it can be any computer on the network).

One of the best known software network analyzers is freely available *Wireshark* or the similar tool *TCPdump* for command line. The next known analyzer is *Kismet*, but the purpose of this software is a bit different: Kismet is focused on Wi-fi, not Ethernet, and its additional properties relate to this focus (uncovering Wi-fi APs not broadcasting beacon frames and other possibilities of wireless signal live analysis).

Solarwinds Bandwidth Analyzer is the next freely available tool (with registration), it can be used for performance monitoring. Other tools are Cain and Abel, Network Miner, EtherApe, KisMAC and others.

7.3.2 Traffic Monitoring

 *Port mirroring* gives administrators the ability to eavesdrop on network traffic on a given segment, even though we have monitoring devices on another segment.

 **A SPAN** (Switch Port ANalyzer) is a feature on switches providing port mirroring.

The source SPAN port and the destination SPAN port (monitor port) are specified on the switch, the switch copies the frames coming to the source SPAN port to the destination SPAN port. This means that all traffic coming to the source SPAN port is mirrored to the destination SPAN port (and of course sent to the “correct” port behind which the frame recipient is located). Either a device with a packet sniffer running or an IDS is connected to the destination SPAN port.

Local SPAN means that the source and destination SPAN ports are on the single device (switch). RSPAN (Remote SPAN) means that these two SPAN ports are on different switches, and the path between them for the monitored traffic is provided using VLAN. The source SPAN port is on the switch through which the monitored traffic passes, the destination SPAN port is on the switch to

which the IDS or other monitoring device is connected. SPAN and transport VLAN settings must be performed correctly on all switches along the path.

Network Tap is a special analyzer whose task is to “collect” traffic on a given network segment (usually in both directions) and forward it to one or two selected ports (for newer ones, for example to a USB output). Some types of taps can function as an IDS, browse all packets, or it can monitor only a specific type of packets (for example, multimedia, or for a specific protocol), it is usually integrated into the network protection system.

The operation seems simple, but it should be noted that the requirement “in both directions” actually means twice the normal traffic, because common communication is in full duplex. We must get full-duplex operation into eavesdropping “in one direction” (towards the eavesdropping computer). Devices intended for use in high-traffic networks typically have at least two ports for this direction, one for each original direction of communication, or (as mentioned above) any type of port other than Ethernet that has at least twice the throughput (e.g. USB version 3.0 or higher).

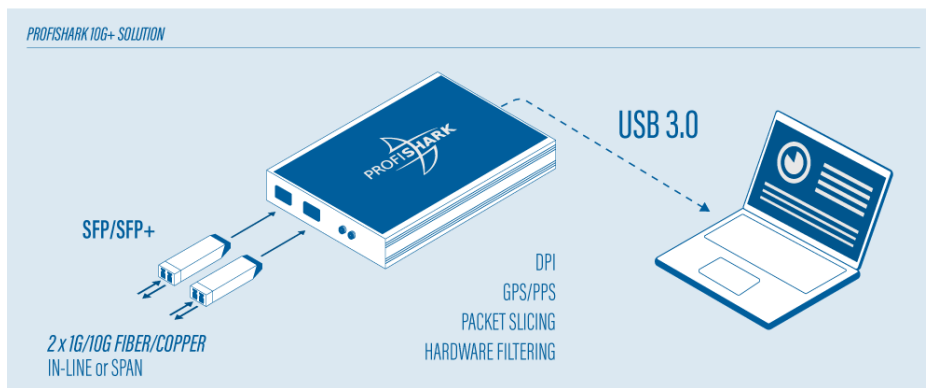


Figure 7.2: Network tap wiring diagram²

Figure 7.2 shows an example of a network tap connection in a network. This device has:

- *inputs* leading from adjacent devices, in the figure these are SFP+ modules (for example for Ethernet optical cables),
- *output(s)*, this can be one or more network interfaces, for newer ones usually USB interfaces with the version 3.0 or higher (new USB versions already have sufficient throughput, but of course it depends on the data flow between the monitored devices),
- *power supply* usually realized via output (in case of twisted pair it is PoE, in case of USB then power wires in USB cable are used, or additional USB cable can be used for power supply); for a fully optical device, no power is required.

PoE can also be used on the link between the monitored devices. In that case, the tap should not “feed”, the power should be transparently forwarded, as we see in Figure 7.3.



Additional information:

- <https://www.profitap.com/profishark-10g-plus/>
- <https://www.garlandtechnology.com/network-tap>

²Resource: <https://www.profitap.com/profishark-10g-plus/>

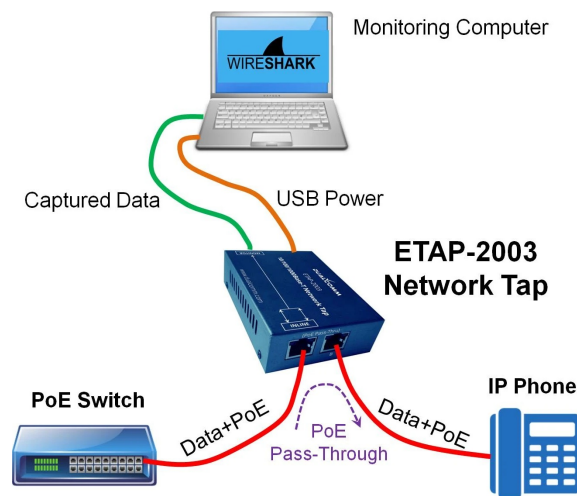



Figure 7.3: Wiring diagram for a network tap with PoE³

- <https://www.dualcomm.com/collections/network-tap>
- <https://www.netscout.com/product/enterprise/netscout-taps>



Figure 7.4: Dualcomm ETAP 2306: Dual-Mode GbE Copper and Fiber Network Tap (4×RJ-45, 2×SFP with optics) and Garland Single-mode Passive Fiber Network TAP⁴

7.3.3 Firewall


 A firewall is a network element (hardware or software) that is used to control (filter) traffic among networks with different levels of trust or security. The firewall defines *rules* for communication between networks, according to which it responds either by enabling communication, disabling it, or requesting the user's opinion. The rules usually cover the following information:

- the source and destination of communication, can be represented by an IP address,
- port number (source and destination),
- protocol, connection status, etc.


The firewall can be either *one-way* (filters only incoming packets) or *two-way* (filters both incoming and outgoing packets). Current firewalls are two-way, they can more effectively capture the possible transmission of sensitive information.

³Resource: <https://www.dualcomm.com/collections/network-tap>

⁴Resources: <https://www.dualcomm.com/>, <https://www.garlandtechnology.com/>

 In addition to software firewalls, there are *hardware firewalls* that act as intermediate network devices. Today, they are mostly part of routers or other common network elements (depending on the type of device, what information the firewall gets to). A hardware firewall has the great advantage of a lower risk of infection (it does not depend on the client computer's operating system). Another advantage is the independence of the computer's performance – if the computer's processor is heavily loaded, it also affects the operation (especially the speed) of the software firewall installed on this computer. The hardware firewall (for example built into another network device) is not affected in this way.

In respect to security, the combination of a simple hardware firewall in a router (for example an VDSL router) and a software router on a computer, each of a different type, is considered ideal in households and small companies.

 **Network Areas.** Firewall divides network into several areas:

- *private network* (trusted, internal network, intranet) – our network with our devices, we usually trust these devices,
- *public network* (external network, internet) – untrusted network (of our ISP, WAN, Internet),
- *demilitarized zone (DMZ)* – somewhere between private and public network; we own everything inside DMZ, this area is usually intended for servers.

Inside DMZ, we have devices that should be accessible from the external network (for example, web server, mail server, DNS server). The servers in the DMZ should be secured as if they were really directly on the Internet, even though they are separated from the Internet by a firewall.

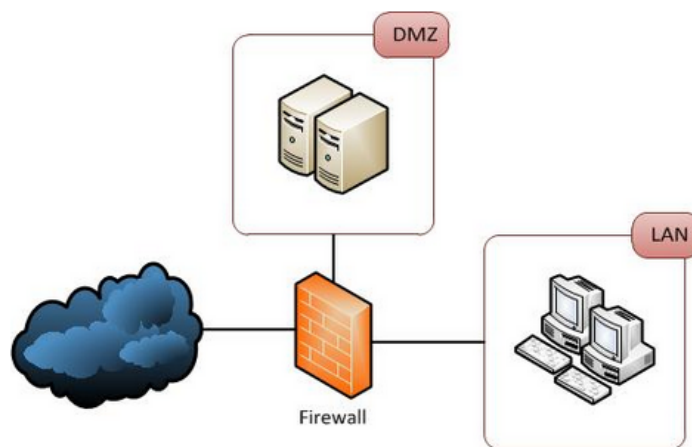


Figure 7.5: Areas divided by firewall⁵


Figure 7.5 shows the above listed areas. The firewall in the middle of the schema filters the communication in the following way:


- an attempt to establish a connection from LAN wherever is allowed (passes),
- an attempt to establish a connection from the Internet to the DMZ is allowed,
- an attempt to establish a connection from the DMZ to the Internet is allowed,
- everything else is denied (will not pass).

⁵Resource: <https://resources.infosecinstitute.com/topic/virtual-dmzs-cloud/>

If a device from the Internet tries to establish a TCP connection to LAN, it is denied, as well as if a device from DMZ tries to establish a connection to LAN.

On a network device that supports DMZ (e.g. firewall or a computer with Linux), we usually have one or more (hardware) ports marked this way, or we can configure some ports ourselves so that they are treated as DMZ.

 A *zone-based policy firewall* (ZPF) allows to flexibly configure a various number of zones with different filtering methods (with different sets of rules). Each firewall interface is assigned to a specific zone and the rules set for that zone are valid for such interface. Usually we have one or more “inner” more or less trusted zones including DMZ and one “outer” untrusted zone. More than one interface can belong to a single zone.


 *OpenWRT* is interesting project: embedded Linux (i.e. Linux adapted for a specific type of device, typically firmware of such a device) for routers by some manufacturers (e.g. Mikrotik producing RouterBoard device).


Its advantage over the “original” operating system (for example, RouterBoard is a device running Linux-based RouterOS) is that it is much closer to common Linux distributions than other embedded Linuxes. While usually the firmware is updated as a whole (this can be quite dangerous, the device can be completely damaged in the event of an update failure), OpenWRT is a flexible system that can be updated, configured and changed in any way during operation, and we can use common packaging system.

As mentioned above, OpenWRT uses a packaging system, so we can install different packages as needed, the choice is wide. Of course, the distribution includes a NetFilter firewall with the iptables front-end program, supports ftp, samba, telnet, ssh, we can install X Window and some suitable window manager (i.e. GUI), we usually configure via ssh. By default, this distribution is typically equipped for a Wi-fi router, but we can change it according to our own needs.

7.3.4 Filtering

We distinguish between different types of filtering according to which ISO/OSI layer a given method can be assigned to (and therefore which information in the headers we reach).

 **Packet filter.** This is the simplest filtering on L3 and partially L4, only IP addresses and port numbers are specified in the rules. It is a simple and fast solution (operation is delayed only minimally), which used to be commonly applied mainly to intermediate network elements, today we can find them in some of the cheapest routers. The disadvantage is the inability to look at the ongoing communication in more complex protocols.


 **ACL at the network layer.** ACL (Access Control List) is a method widely used in both desktop and server operating systems, as well as in network devices. The purpose is to determine access rules for different clusters of users/communications. In essence, it is a functional extension of the packet filter method.

The access control list is actually a list of items marked by categories:

- *deny* – not allowed traffic,
- *permit* – allowed traffic,
- *deny all else* – what was not previously allowed must not pass.

The ACL is usually implemented in the form “what is not allowed is denied”, so we will find only *permit* items (and what is not in these items will not pass). The order of the items is also important.


It is usually filtered according to the destination or source IP address, their combination, and possibly according to other criteria (for example items in the PDU header of the network layer – IP protocol, ICMP, TCP/UDP ports, etc.).

 **Stateful packet inspection (SPI).** At the transport layer (strictly: L3+L4), SPI is performed. While on L3 packets are taken as “individuals” without mutual binding, on L4 it is possible to take into account their mutual relations. For example, we can distinguish packets that establish a connection from packets that belong to an existing connection.

The packet establishing connection is thoroughly scanned (data from the L3 and L4 layers, such as source and destination addresses, protocols, source and destination ports, flags set by each protocol, anything in the PDU headers), and if it passes, a record in the *status table* is created for this connection. If the packet fails the check, the record is not created and packet is denied.


Packets that belong to (or impersonate) an already established connection are filtered according to whether their connection is recorded in the status table.

Currently, SPI is basically a standard for quality firewalls. Compared to a conventional packet filter, it offers greater security with a relatively small slowdown in filtering operation.

 **Proxy firewalls** (also called application gateways) operate on the TCP/IP application layer. They work with application protocols, for example they understand HTTP, FTP, IMAP, they can work with packets containing elements for ActiveX, etc.

This type of firewall actually works as a man-in-the-middle (but legal), i.e. it divides the path between the client and the server into two parts. Server “looks at” proxy instead of client, client “looks at” proxy instead of server. This can cause problems, especially for encrypted connections.

All communication is processed using so-called *proxy* – software gateways either programmed for a specific type of communication (protocol) with a relatively high level of security (for example for FTP or HTTP), or using a generic proxy usable generally for various protocols.

 We distinguish two types of proxy:

1. *standard proxy* – everything that was previously written about proxy applies to it. It filters all packets according to data from PDU protocols of the application layer and lower layers, the analysis is time consuming and can have a great influence on the throughput of the line.
2. *Dynamic proxy* – behaves differently in different packets; to the initiating connection it behaves the same as the first type of proxy, but to packets belonging to the already established connection it behaves as an SPI (i.e. at the lower TCP/IP layer). The result is faster processing of incoming and outgoing traffic.


Task

A firewall can behave towards a rejected packet in several different ways: prohibit, unreachable, black-hole. In all cases, the packet is dropped, but the difference is in whether and how the destination or source device is informed. Find differences between them in some suitable sources (for example for iptables, which is the front-end for the Linux firewall).




7.3.5 IDS/IPS and Deep Packet Inspection

SPI can be taken as the basis for IDS/IPS systems – IDS and IPS.

 *IDS* (Intrusion *Detection* System) is a passive appliance (the traffic flow does not go through this device, but is mirrored to it). IDS does not interfere in the data flow, it only announces in some way that something is wrong.

IPS (Intrusion *Prevention* System) is able to actively respond to detected problems. The response is intended to prevent an attack, so the specific location of the IPS probe has to be planned so that it can also interfere with the configuration of other network devices (for example, change the rules in the firewall) or drop packets (IDS cannot drop packets). The main problem of IPS is that its overload or e.g. technical problems might affect network traffic, and the performance of this device may affect the network (slower, worse latency, etc.) even during normal operation.

A firewall can have an integrated IDS/IPS function (module), but it is generally safer (especially for larger networks) to deploy IDS/IPS separately from the firewall. Thus IDS/IPS is either a separate device, or a hardware module in a router or firewall, or a software module or application.

 IDS and IPS appliances search for malicious traffic patterns and compare them with *signatures* obtained from trusted security sources (clouds of security providers). Additional functionality is using *heuristics* and detection of *unusual network traffic*.

Unusual network traffic is called *anomaly-based detection*, it means real-time comparing of host/network behaviour with a learned *baseline*. A baseline is a model for common host/network behaviour, for example a network baseline is characterized by the network performance, current protocols in the traffic, the ratio of source and destination addresses from the internal and external network, the dynamics of changes in the operation and use of the network, etc.

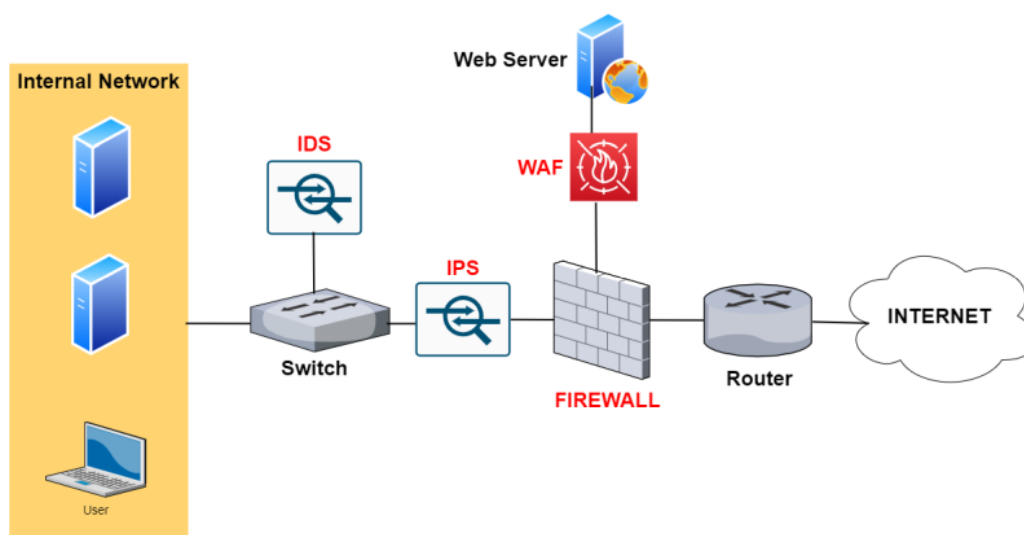




Figure 7.6: Scheme of possible IDS/IPS connection⁶


 Stateful packet filter with *deep packet inspection* (DPI) is just a firewall with an integrated IDS/IPS module. This type of firewall adds additional options for defining rules related to the usual properties of communication with a given (known) application or protocol. For example, if it detects that HTTP

⁶Resource: <https://www.networkstraining.com/firewall-vs-ips-vs-ids-vs-waf/>

is being used for a different type of communication than with the Web server, it blocks this request as suspicious.

 *HIDS, HIPS* (Host-based IDS/IPS) is a module in an end device (server, computer, etc.) intended for protection of the host device, it is an agent-based system.

Examples of HIDS/HIPS: OSSEC (Open Source HIDS Security), Safetica, AlienVault USM, Tripwire and its open-source variant AIDE, Cisco AMP. OSSEC uses a central server for IDS/IPS management and agents installed on individual hosts (computers, servers).

 *NIDS, NIPS* (Network-based IDS/IPS) is implemented as a sensor through which the data flow in the network passes (NIPS) or the data flow is mirrored to it for monitoring purposes (NIDS).

Examples of NIDS/NIPS: Snort, Squil, Suricata, Fortigate, FortiOS, Zeek (Bro), OpenWIPS-ng. Most of them are freely available, except Fortigate and FortiOS.

In Figure 7.6 we see a possible scheme of connecting firewalls and IDS/IPS in a larger network. The whole network is protected by a firewall, behind which is the IPS. All communication goes through the IPS, so it can respond to anything suspicious in any packet. The firewall defines three zones, one of them is DMZ (there is a web server inside, protected by a web application firewall, WAF, possibly with a built-in IDS/IPS module). A switch mirrors all traffic leading to and from the internal network to an IDS.




Additional information:

- <https://www.networkstraining.com/firewall-vs-ips-vs-ids-vs-waf/>
- <http://www.symantec.com/connect/articles/intrusion-detection-terminology-part-one>
- <http://www.symantec.com/connect/articles/intrusion-detection-terminology-part-two>



7.4 Complex Management and Monitoring Systems

7.4.1 Network Management Software

 *Network Management Software* is a system that can monitor the state of network (various variables), i.e. collect information from various sources in network, generate reports and in case of problems (abnormal values of monitored variables, large fluctuations etc.) to respond appropriately (for example, send information to the admin).

When choosing a network management software, we take into account:

- what can be monitored (number and types of monitored services, probes, ...),
- the specificity of reports (some faults are “hierarchical”, so if something fails due to anything else failing, the report should not include everything, but only “root of the problem hierarchy”),
- user interface (today it is usually a web interface), it should be sufficiently clearly arranged and appropriately hierarchically divided, what can be displayed and in what form, the ability to visualize various data flows or anything else that can be changed over time,
- method of sending messages: via SMTP server (or it can be integrated), SMS, instant messaging, etc.,
- possibilities of setting user privileges for access to collected information,

- functional limitations: method of communication with monitored devices (via which protocol, e.g. SNMP), method of logging, etc.

There are both open-source and commercial network management systems. Commercials include, for example, WhatsUp Gold, SonicWALL, and others. We will look at some open-source products in the following text.



Additional information:

- <https://www.whatsupgold.com/>
- <http://www.sonicwall.com/>
- <http://www.networksecuritytoolkit.org/nst/index.html>
- <https://www.networkmanagementsoftware.com/network-monitoring-tools/>
- <https://www.monitortools.com/>



Nagios is a very popular open-source monitoring system. It handles not only the network monitoring, but also the visualization. It can monitor various types of network services (HTTP, POP3, ICMP, SMTP), has no problems with encryption, also monitors resource usage on network nodes (“understands” various operating systems including Windows), performs network state visualization (can also plot network topology, including 3D graph). We can use the *web interface* for configuration (but there are other interfaces, such as Centreon in combination with MySQL).

A new fork of Nagios was created, which is called *Icinga*. According to the developers of this fork, the main purpose is to speed up and make development more flexible.



Additional information:

- <http://www.nagiosbook.org/>
- http://nagios.sourceforge.net/docs/3_0/quickstart.html
- <https://icinga.com/>



Zabbix is another popular open-source network management system. We need Zabbix Server (installed on the monitoring server, it should run a UNIX-like system including Linux) and Zabbix agents (on client devices, various operating systems including Windows). The configuration is performed again via the web interface, i.e. nothing complicated after the basic orientation.


The functionality is similar to Nagios, usually Zabbix is considered simpler in features and configuration (Nagios is considered more stable). It also supports SNMP and agents can be installed on various operating systems.




Additional information:

- <http://www.zabbix.com/>
- <http://www.zabbix.com/documentation.php>




 **OpenNMS** is an open-source network management system, which, unlike the previous ones, is written in Java. It is designed for monitoring large networks with a large number of monitored devices. Monitoring can be distributed (on multiple servers), thanks to which such a large number of devices can be monitored.

It is a scalable system. Monitoring of various services is provided using plugins (support for monitoring another service can be added simply by adding a plugin). The configuration is again performed via web interface. It is possible to try working with the system on the demo application on the project website.


 **Additional information:**

- http://www.opennms.org/wiki/Main_Page
- http://www.howtoforge.com/opennms_network_management
- <http://demo.opennms.org/opennms/acegilogin.jsp>




 **Zenoss** is another open-source network management system (one variant is open-source and free to download, the other – Enterprise – commercial). It is based on the idea of using existing open-source projects.

Its core is the object web server *Zope* written in Python, and Zenoss can also be further extended in Python (it is even possible to use plugins from Nagios, which is also written in Python). Next, the *MySQL* database system is added and the whole system runs over *Twisted*, which is an event-driven interface for programming network applications that understands many different protocols on multiple ISO/OSI layers.


 An important part of the system is *RRDTool* (Round Robin Database Tool). This tool is used to store, process and create a graphical representation (graphs) of any collections of numbers that we pass to it. The basis is a database organized as a round robin queue (hence the name Round Robin Database) with a static length, in which older data is overwritten by newer data.

Zenoss can handle SNMP, ICMP, other TCP/IP protocols, communicates with UNIX-like systems and their common tools (including syslog) and also with Windows and the related tools (it also supports WMI). We can find information on the company's website that cloud devices are also supported (the term Cloud Computing is very popular). The configuration takes place via web interface.

 **Additional information:**

- <https://www.zenoss.com/product>
- <http://oss.oetiker.ch/rrdtool/tut/>
- <https://help.ubuntu.com/community/Zenoss>



 **Cacti** is another open-source extension to RRDTool, similar to Zenoss. Also in Cacti it is about how to obtain data, save it appropriately and then analyze and display it. Data can be obtained from various sources, such as SNMP, scripts in various scripting languages or WMI, and stored either in an SQL database or using RRDTool. The system is designed for small and medium networks (hundreds, maybe thousands, network nodes).

The community around Cacti is growing comfortably, so there are a lot of plugins and templates for various types of values. As with the previous tools, we also have web interface for configuring and manipulating data.



Additional information:

- <https://cacti.net/>
- <https://forums.cacti.net/about30438.html>
- <http://www.ubuntugeek.com/install-and-configure-cacti-monitoring-tool-in-ubuntu-9-10-karmic-server.html>



Figure 7.7: The Cacti Network management System⁷



Task

Choose one of the mentioned network management systems and find as much information about it as possible. If it is possible, try it.



7.4.2 SIEM

SIEM (Security Information and Event Management) may be similar in definition to a network management system, but is more security-oriented. These systems serve as tools for collecting, analyzing and presenting information obtained from various connected devices, whether common network components or security devices (cameras, various sensors, IDS/IPS probes).

⁷Resource: <https://linitut.com/best-free-monitoring-system-for-linux/>

Common tasks in SIEM:

- data collection from various sources, working with logs, data normalization (conversion to a uniform format),
- correlation – searching for and evaluating relationships among data, e.g. among different events in the network,
- immediate warning when a problem occurs,
- presentation of information – creating reports, graphs, statistics.

The purpose is to have all the information needed to resolve security incidents in one place and to be able to respond to them as quickly as possible.

We can meet either a complete SIEM solution, or separately SIM (Security Information Management, i.e. working with logs, information sorting, reporting, visualization, etc.) and SEM (Security Event Management, i.e. mainly working with logs, real-time analysis, searching dependencies among events and quick response to events).

SIEM uses both common methods from network management systems (such as working with logs) and methods similar to those used by anti-virus programs (working with signatures, heuristic analysis, behaviour analysis, etc.).

SIEM's offer is relatively extensive. We can choose an open-source solution or a commercial product, but we should choose mainly according to our own needs. An important criterion is the number of devices we have in the network and to whose logs SIEM will be connected, we should be interested in which types of devices SIEM can work natively and, conversely, which devices will connect the problem, the size of the network is related to EPS), ie how many events per second SIEM will have to handle, etc.

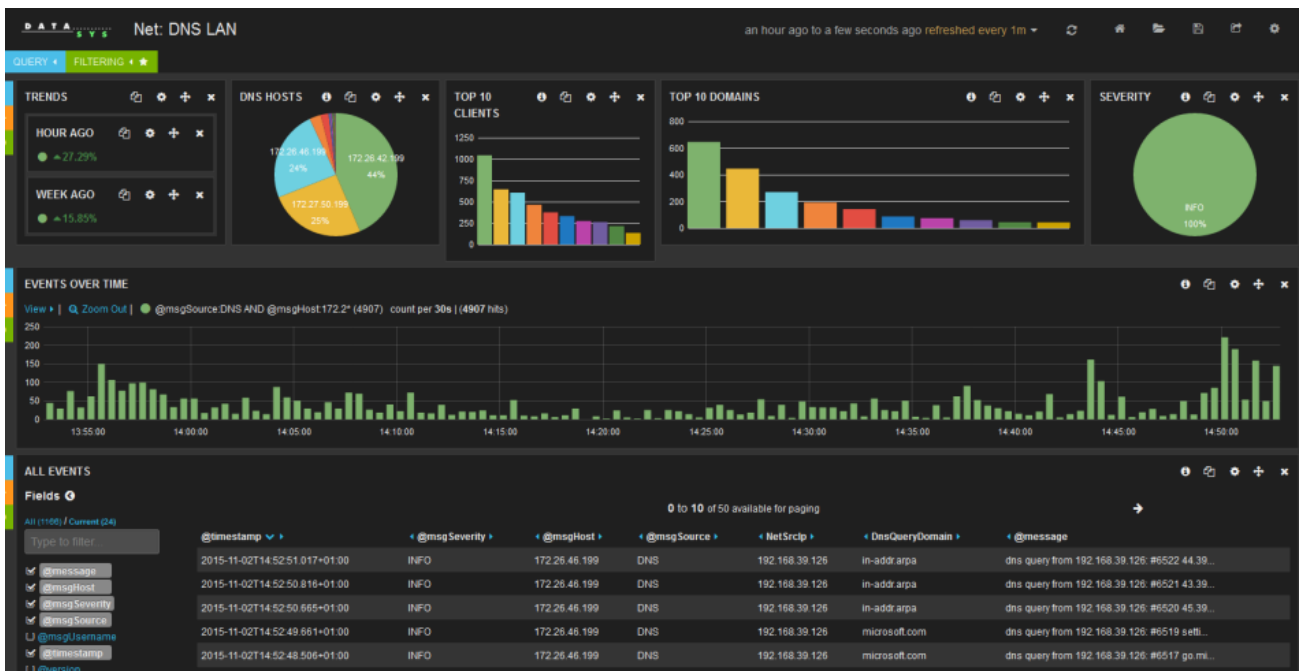





Figure 7.8: Kibana – interface used in Elisa SIEM⁸

⁸Resource: <https://docplayer.cz/36323951-Datasys-elisa-log-management-rizeny-zabbixem-lukas-maly-dis-it-konzultant-bezpecnost-a-monitoring.html>

 The *EPS* value (Events per second) means how many events per second can be handled by the particular SIEM. It should be noted that especially when problems occur, the system may be overloaded with data from various sources, and if it cannot process incoming traffic quickly enough, it becomes unnecessary.

 The selection of SIEM systems is relatively extensive. We can choose an open-source solution or a commercial product, but we should choose mainly according to our own needs. An important criterion is the number of devices we have in the network to manage, and whose logs SIEM will be connected to, we should be interested in which types of devices SIEM can work natively with and, conversely, which devices are problematic to connect them, and certainly, the EPS value.

 Cisco offers its own SIEM, others include Splunk (it is available in several variants), Helix Security Platform (by FireEye), Qradar SIEM (by IBM), AlienVault OSSIM and AlienVault USM (the both by AT&T Cybersecurity), FortiSIEM (by Fortinet), McAfee Enterprise Security Manager, LogRhythm NextGen SIEM Platform, Trustwave SIEM, Rapid7 InsightIDR, Juniper JSA SIEM, or the Czech product ELISA (by DataSys, builds on Zabbix, Xlog and NetFlow).



Additional information:

- <https://searchsecurity.techtarget.com/feature/Comparing-the-best-SIEM-systems-on-the-market>
- <https://www.itcentralstation.com/categories/security-information-and-event-management-siem>
- <https://www.comparitech.com/net-admin/siem-tools/>
- <https://www.selecthub.com/siem-solutions/>
- <https://www.softwaretestinghelp.com/siem-tools/>

**Task**

Choose any SIEM (use the links above to sites comparing various SIEM systems) and find out more information about it.



Recommended Resources

- [1] American institute of Certified Public Accountants, Inc. *Guide: Reporting on an Entity's Cybersecurity Risk Management Program and Controls*. New York: John Wiley, 2017. ISBN 978-1-94354-672-5.
- [2] *Aruba (HP) OS 6.1 Documentation* [online].
URL: https://www.arubanetworks.com/techdocs/ArubaOS_61/ArubaOS_61_UG/ [cit. 2020-10-30]
- [3] CHAUDHARY, Harry. *Hands on Computer Networks 1500+ MCQ Test Series: Handy Book Series for All IT Exams & Interviews*. STCD Company, 2018. ISBN 978-0-359-03847-3. Most pages available at: <https://books.google.cz/books?id=fpJqDwAAQBAJ&printsec=frontcover>
- [4] *CCNA Cybersecurity Operations Companion Guide*. Indianapolis, IN: Pearson Education, Cisco Press, 2018. ISBN 978-1-58713-439-5. Most pages available at:
<https://books.google.cz/books?id=FxRbDwAAQBAJ&printsec=frontcover>
- [5] *Check Point Support Center* [online]. URL: <https://supportcenter.checkpoint.com/supportcenter/portal> [cit. 2020-10-30]
- [6] *Cisco Documentation* [online]. URL: <https://www.cisco.com/c/en/us/tech/index.html> [cit. 2020-10-30]
- [7] *Fortinet Docs Library* [online]. URL: <https://docs.fortinet.com/> [cit. 2020-10-30]
- [8] GORALSKI, Walter. *The Illustrated Network: How TCP/IP Works in a Modern Network*. 2nd edn. Cambridge, MA: Elsevier, 2017. ISBN 978-0-12-811027-0. Most pages available at:
<https://books.google.cz/books?id=6nDtNA6VJ5YC&printsec=frontcover>
- [9] HUBERT, B. et al. *Linux Advanced routing & Traffic Control* [online].
URL: <http://lartc.org/howto/>, others <http://lartc.org/> [cit. 2020-10-30]
- [10] *Introduction to Networks v6: Companion Guide*. Indianapolis, IN, USA: Cisco Press, [2017]. ISBN 978-1-58713-360-2. Also available at:
<http://ptgmedia.pearsoncmg.com/images/9781587133602/samplepages/9781587133602.pdf>
- [11] Infosec Institute. *Ethical Hacking* [online]. Available at: <https://resources.infosecinstitute.com/topics/ethical-hacking> [cit. 2020-10-30]

-
- [12] *Juniper Documentation* [online]. URL: <https://www.juniper.net/documentation/> [cit. 2020-10-30]
- [13] LAMMLE, Todd. *CCNA: Routing and Switching: Study Guide*. Indianapolis, Indiana: SYBEX, [2013]. ISBN 978-1118749616. Dostupné také z: <http://index-of.co.uk/Various/CCNA>
- [14] *Linux Home Networking* [online]. Books on computer network management in Linux. URL: <http://www.linuxhomenetworking.com/wiki/index.php> [cit. 2020-10-30]
- [15] *Routing and Switching Essentials: Companion Guide*. Indianapolis, IN: Cisco Press, c2014. Cisco Networking Academy Program series. ISBN 978-1-58713-318-3. Also available at: <http://ptgmedia.pearsoncmg.com/images/9781587133183/samplepages/1587133180.pdf>
- [16] SANDERS, Chris. *Practical Packet Analysis: Using Wireshark to Solve Real-World Network Problems*. 3rd edition. San Francisco: No Starch Press, [2017]. ISBN 978-1-59327-802-1. Sample capture files available at: https://nostarch.com/download/ppa3ecaptures_updated.zip
- [17] *Scaling Networks v6: Companion Guide*. Indianapolis, Indiana: Cisco Press, [2018]. ISBN 978-1-58713-434-0. Also available at: http://ptgmedia.pearsoncmg.com/images/9781587134340/samplepages/9781587134340_sample.pdf
- [18] TAUB, Mark L. *Connecting Networks v6: Companion Guide*. Indianapolis, Indiana: Cisco Press, [2018]. Cisco Networking Academy Program series. ISBN 978-1-58713-432-6. Also available at: http://ptgmedia.pearsoncmg.com/images/9781587134326/samplepages/9781587134326_sample.pdf
- [19] *The TCP/IP Guide* [online]. URL: <http://www.tcpipguide.com/free/index.htm> [cit. 2020-10-30]
- [20] WALRAND, Jean and Shyam PAREKH. *Communication Networks: A Concise Introduction*. Second Edition. Morgan & Claypool Publishers, 2018. ISBN 978-1681-73615-0. Most pages available at: <https://books.google.cz/books?id=dcBIDwAAQBAJ&printsec=frontcover>