

MINISTRY OF SCIENCE AND EDUCATION OF THE REPUBLIC OF
KAZAKHSTAN

Non-Profit Joint Stock Company
ALMATY UNIVERSITY OF POWER ENGINEERING AND
TELECOMMUNICATIONS

Department Telecommunication systems and networks

«Admitted»

Head of the Department Baykenov A.S.
d.t.s., professor

(Surname and initials, degree, rank)

« » 20 y.
(sign)

DIPLOMA PROJECT

Theme: Development of a complex of laboratory works
for studying the technologies of virtualization
of data transmission networks

Specialty: 5B071900 – Radio engineering electronics and telecommunications

Implemented by: Li-Shan-Shin D.V. ICTe-14-9
(Student's surname and initials) group

Scientific Supervisor: Panchenko S.V., M.S., senior lecturer
(Surname and initials, degree, rank)
«31» 05 2018 y.
(sign)

Advisors:
of Economy section: Tuzelbayev B.I., PhD, associate professor
(Surname and initials, degree, rank)
«25» 05 2018 y.
(sign)

of Life activity safety section: senior lecturer Begimbetova A.S., PhD
(Surname and initials, degree, rank)
«25» 05 2018 y.
(sign)

of Computer Science section: Panchenko S.V., M.S., senior lecturer
(Surname and initials, degree, rank)
«31» 05 2018 y.
(sign)

Standards compliance controller: Panchenko S.V., M.S., senior lecturer
(Surname and initials, degree, rank)
«31» 05 2018 y.
(sign)

Reviewer: Vassin V.V., CTO of KVINT LLP
(Surname and initials, degree, rank)
«05» 06 2018 y.
(sign)

Almaty 2018 y.

MINISTRY OF SCIENCE AND EDUCATION OF THE REPUBLIC OF
KAZAKHSTAN

Non-Profit Joint Stock Company
ALMATY UNIVERSITY OF POWER ENGINEERING AND
TELECOMMUNICATIONS

Institute of Space Engineering and Telecommunications (ISET)
Specialty: 5B071900 – Radio engineering electronics and telecommunications
Department: Telecommunication systems and networks

ASSIGNMENT

For diploma project implementation

Student: Li-Shan-Shin Dmitry Vladimirovich
(name, patronymic and surname)

Theme: Development of a complex of laboratory works
for studying the technologies of virtualization
of data transmission networks

Approved by Rector order № 155 of « 23 » 10 20 17 y.

Deadline of completed project: « 25 » 05 20 18 y.

Initial data for project, required parameters of designing result, object initial data:

The prerequisites for this project are high requirements
to the quality of knowledge of specialists in the network
technologies virtualization field. It is proposed to implement
a complex of laboratory works for the preparation of these
specialists in the diploma project.

List of questions for development in diploma project or brief content:

Introduction
SDN and NFV
VLAN
VXLAN
IPv6
Troubleshooting prerequisites
General principles of Linux diagnostics
Proposed sequences of laboratory works
Mathematical model for troubleshooting
Life Safety part
Economical part
Conclusion

List of illustrations (with exact specifying of mandatory drawing):

Software-Defined Architecture

Network Functions Virtualization

VLAN Configuration

VXLAN Packet format

Linux performance observability tools

USB method algorithm

Output of Linux commands

Proposed sequences of laboratory works

KVM architecture

Recommended main references:

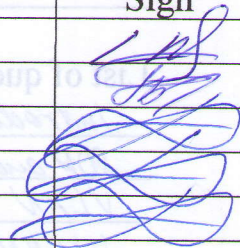
Kumar R. Software Defined Networking - a Definitive Guide, 2017

Open Networking Foundation. Software-Defined Networking: The New Norm for Networks. ONF White paper, April 13, 2012

David H. Jonassen, Woei Hung. Learning to Troubleshoot: A New Theory-Based Design Architecture. Educational Psychology Review Vol 1, No. 1, March 2006

MacPherson R.T. Factors affecting technological troubleshooting skills, 1999

Project adviser with corresponding sections specifying:

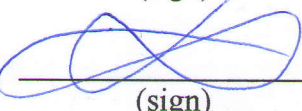
Section	Advisor	Dates	Sign
Life Safety	Beginbitova A.S.	25.05.18	
Economic part	Tuzelbayev B.I.	25.05.18	
Computer science	Panchenko S.V.	31.05.18	
Technical part	Panchenko S.V.	31.05.18	
Standard compliance	Panchenko S.V.	31.05.18	

SCHEDULE of diploma project implementation

№	Sections, list of developing questions	Dates of bringing to Scientific Supervisor	Notes
1	Introduction	30.11.17 - 01.12.17	Done
2	SDN and NFV overview	05.12.17 - 11.12.17	Done
3	VLAN, VXLAN configurations	12.12.17 - 15.12.17	Done
4	Description of IPv6	18.12.17 - 20.12.17	Done
5	Troubleshooting principles	22.01.18 - 26.01.18	Done
6	Principles of Linux diagnostics	05.02.18 - 10.02.18	Done
7	Defining the sequences for laboratory works	19.02.18 - 20.02.18	Done
8	Open vSwitch overview, isolation of VLAN using Open vSwitch	10.03.18 - 20.03.18	Done
9	Description of KVM hypervisor	02.04.18 - 09.04.18	Done
10	Storage virtualization using LVM and iSCSI technologies	30.04.18 - 04.05.18	Done
11	Calculation part	07.05.18 - 20.05.18	Done
12	Analysis of working conditions in a laboratory room	06.03.18 - 09.03.18	Done
13	Calculation of lighting in a room	09.03.18 - 19.03.18	Done
14	Life Safety part conclusion	20.03.18 - 21.03.18	Done
15	Economic and technical justification for laboratory works	09.04.18 - 13.04.18	Done
16	Calculation of economic efficiency and annual operating costs	17.04.18 - 20.04.18	Done
17	Economic part conclusion	23.04.18 - 24.04.18	Done
18	Conclusion	20.05.18 - 21.05.18	Done

Assignment issue date « 12 » 01 20 18 y.

Head of Department: _____ (sign) Baykenov A.S.
(Surname and initials)

Scientific Supervisor:  (sign) Panchenko S.V.
(Surname and initials)

Assignment submitted for implementation:  (sign) Li-Shan-Shun D.V.
(Surname and initials)

Аңдатпа

Берілген дипломдық жобада тегін бағдарламалық қамтамасыз ету негізіндегі виртуалдық ортада зертхана жұмыстар жиынтығы жасалды. Зертханалық жұмыстар барысында ақауларды анықтау және жою жүзеге асырылады. Үш түрлі зертханалық жұмыс жолдары бөлінді: Linux желілік бөлімі, желілік виртуалдау технологиясы, сақтау және есептеу. Өміртіршілік қауіпсіздік бөлімінде еңбек жағдайларына талдау жүргізілді және жарықтандыру шарттары есептелді. Дипломдық жобаның экономикалық бөлімінде күрделі шығындар және зертханалық жұмыстарды енгізудің экономикалық тиімділігі есептелді.

Аннотация

В данном дипломном проекте разрабатывается комплекс лабораторных работ в виртуальной среде, основанных на свободном программном обеспечении. Предлагается подход при котором лабораторные работы будут выполняться путем выявления и устранения неисправностей. Были выделены три трека лабораторных работ: сетевая часть Linux, технологии виртуализации сетей, хранилищ и вычислений. В разделе безопасности жизнедеятельности произведен анализ условий труда, а также произведен расчет условий освещения необходимых для комфортной работы. В экономической части дипломной работы рассчитаны капитальные затраты и экономическая эффективность введения лабораторных работ.

Abstract

In this diploma project complex of laboratory works is developing in a virtual environment based on free software. An approach is proposed in which laboratory work will be performed by identifying troubles and troubleshooting. Three tracks of laboratory works were presented: the Linux networking part, virtualization technologies of networks, storages and computing. In the section of life safety, the analysis of working conditions is made, as well as calculation of lighting conditions necessary for comfortable work. In the economic part of this paper were calculated capital costs and economic efficiency of the integration of laboratory work.

Content

Introduction	7
1 Theoretical part.....	9
1.1 SDN and NFV	9
1.3 VLAN.....	17
1.3 VXLAN.....	21
1.4 IPv6	24
2 Technical part.....	30
2.1 Troubleshooting prerequisites.....	30
2.2 General principles of Linux diagnostics	33
2.3 Proposed sequences of laboratory works	40
2.4 Mathematical model for troubleshooting.....	48
3 Life safety part.....	52
3.1 Analysis of working conditions in a laboratory room with a PC.....	52
3.2 The operator visual load during the work with a PC	54
3.3 Calculation of natural lighting	55
3.4 Calculation of artificial lighting	59
4 Economical part.....	61
4.1 Technical and economic justification for the realization	61
4.2 Calculation of capital costs	62
4.3 Calculation of annual operating costs	65
4.4 Calculation of income	68
4.5 Calculation of economic efficiency	68
Conclusion.....	71
List of abbreviations.....	72
List of references	74
Appendix A Listing of the error service simulation model in the GPSS World	76
Appendix B GPSS World simulation report	78
Appendix C Anti-plagiarism report	
Appendix D Electronic version of DP and demonstration materials (CD-R)	
Appendix E Handout	

Introduction

Troubleshooting is a common form of problem fixing. Technicians and specialists diagnose defective systems and take direct, corrective action to get rid of any faults with the intention to return the systems to their normal states. In the existing laboratory courses there are step-by-step tasks that lacks developing troubleshooting skills in students. Nowadays in telecommunication industry it is more likely to face the existing network system and it is necessary to be able to understand, service, locate and fix errors in those systems.

With the challenges of the revising lab courses in mind, the goals of the courses transformation are clear. First, the course should be an excellent learning experience for students. It should prepare students for research by integrating them into the practices of diagnostics of systems. Second, the course should be an excellent teaching experience for faculty by providing the resources and structure so that faculty can concentrate on having quality interactions with students by sharing their expertise in troubleshooting. Third, with all physical equipment and space the courses should cost as low as possible.

Learning skills in troubleshooting helps future specialists to reduce losses from the network downtime. Sometimes it is impossible to avoid system downtime, but planning and training in most cases will help to resume operations faster and reduce negative consequences. Most of the accidents at the plant are caused by human factors. The human factor includes poor staff training, poor maintenance procedures and poor management of operations. There are material and non-material costs of downtime. They can be expensive, but apart from financial consequences, there is a loss of reputation of a company.

With the ever-increasing volume and variety of network traffic, generated by such high-demand sources as big data, cloud computing, and mobile traffic, it becomes increasingly difficult to meet stringent QoS and QoE requirements. Networks need to be more adaptable and scalable. To provide adaptability and scalability, two key technologies that are rapidly being deployed by a variety of network service and application providers are software-defined networking (SDN) and network functions virtualization (NFV).

SDN has reached a tipping point at which it is replacing the traditional networking model. Software-defined networks provide an enhanced level of flexibility and customizability to meet the needs of newer networking and IT trends such as cloud, mobility, social networking and video.

Virtualization encompasses a variety of technologies for managing computing resources into logical or virtual resources. Virtualization enables users, applications, and management software operation above the abstraction layer to manage and use resources without needing to be aware of the physical details of the underlying resources. Moreover, students can study network operation deeper while exploring protocols and principles behind them. However, with a variety of commercial SDN platforms the University cannot afford using them but there are many open-source platforms that has a sufficient and necessary capabilities for studying virtualization.

The purpose of the work: introduce a new way for laboratory works methodology, taking into account modern technologies and demand in networking. With the following goal, the objectives are: to create studying space for learning SDN and NFV technologies with the use open-source products, to reduce economic costs, to abandon expensive physical equipment and to revise existent methodic for laboratory works.

1 Theoretical part

1.1 SDN and NFV

1.1.1 SDN Functionality. The two components concerned in forwarding packets through routers are a control function, that chooses the path the traffic transports as well as the distant relative concern of traffic, along with a data function, which forwards data dependent on control function policy. Before SDN, these capabilities were carried out in an integrated way in each network device e.g. router, packet switch, bridge, and so forth. Control in these kinds of a traditional network is exercised using a routing and control network protocol that is applied at each network node. This approach is fairly inflexible and requires all of the network nodes to implement exactly the same protocols [1]. With SDN, a central controller performs all complex functionality, including routing, naming, policy declaration and security checks (Figure 1.1).

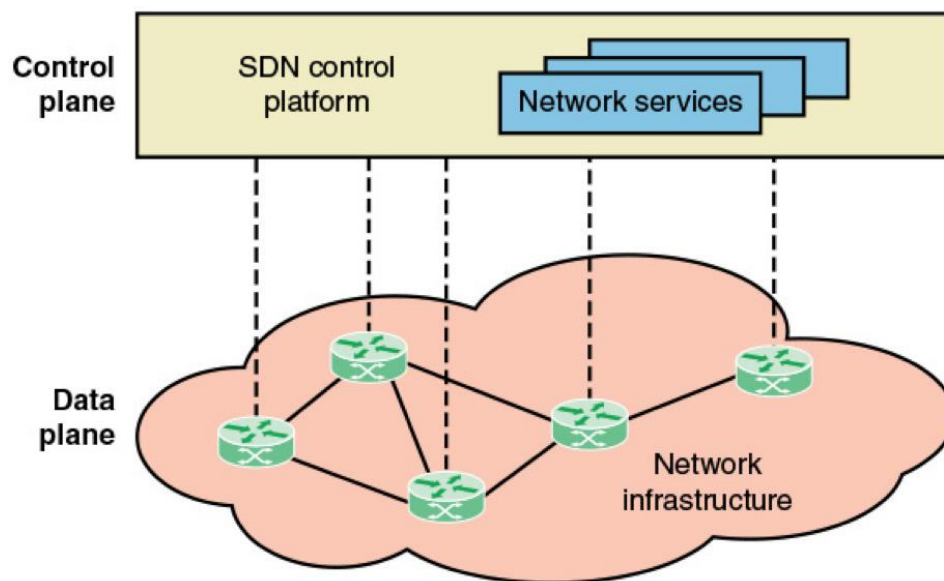


Figure 1.1 – Software-Defined Networking

The SDN controller defines the data flows that arise in the SDN data plane. Each and every flow throughout the network is configured by the controller, that verifies the interaction is allowable by the network policy. In case the controller enables a flow requested by an end process, a path for the flow to consider is computed by it, and also gives an entry for that flow at every one of the changes across the course.

1.1.2 SDN architecture. The SDN strategy splits the switching feature between a data plane along with a control plane which are on individual devices (Figure 1.2). The data plane is actually accountable for forwarding packets, while the control plane offers the "intelligence" in developing routes, setting precedence and routing parameters to meet up with QoE and QoS demands [2].

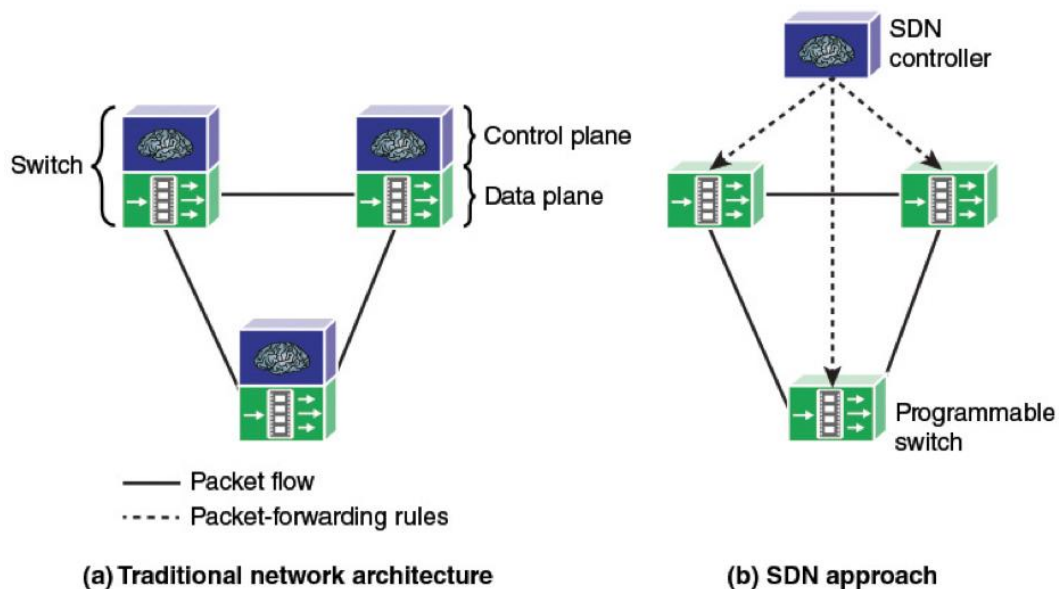


Figure 1.2 – Control and Data Planes

Figure 1.3 shows detalization of the SDN structure. Virtual switches and physical switches are included by the data plane. Within every situations, the switches are accountable for forwarding packets, the inner implementation of buffers, precedence of details, along with various other information. Structures associating with forwarding may be reliant from vendor. Nevertheless, each switch must apply abstraction, or a model, of packet forwarding that's uniform and open to SDN controllers. This particular design is defined in conditions of an open application programming interface (API) in between the control plane and also the data plane (southbound API). Probably the most prominent example of these open API is OpenFlow. The OpenFlow specification defines equally a strict process between the control plane as well as an API and data planes by that the control plane is able to invoke the OpenFlow protocol.

SDN controllers could be applied on a server or perhaps on a virtual server. OpenFlow or various other open API is utilized to regulate the switches in the data plane. Additionally, information regarding capacity as well as demand obtained from the network equipment whereby the traffic flows are used by controllers. SDN controllers additionally expose northbound APIs, which enable designers as well as community supervisors to deploy a broad range of custom-built network applications and off-the-shelf, a lot of which were not doable prior to the arrival of SDN. As but there's simply no standardized northbound API neither a consensus on a wide open northbound API. A selection of vendors give you a REpresentational State Transfer (REST)-based API to make a programmable screen to the SDN controller of theirs. Additionally horizontal APIs (east/westbound), would allow cooperation and communication amongst organizations of controllers to synchronize state for top accessibility. At the application plane are an assortment of uses that communicate with SDN controllers. SDN programs are applications that

could implement an abstract perspective of the system for the decision-making goals [3].

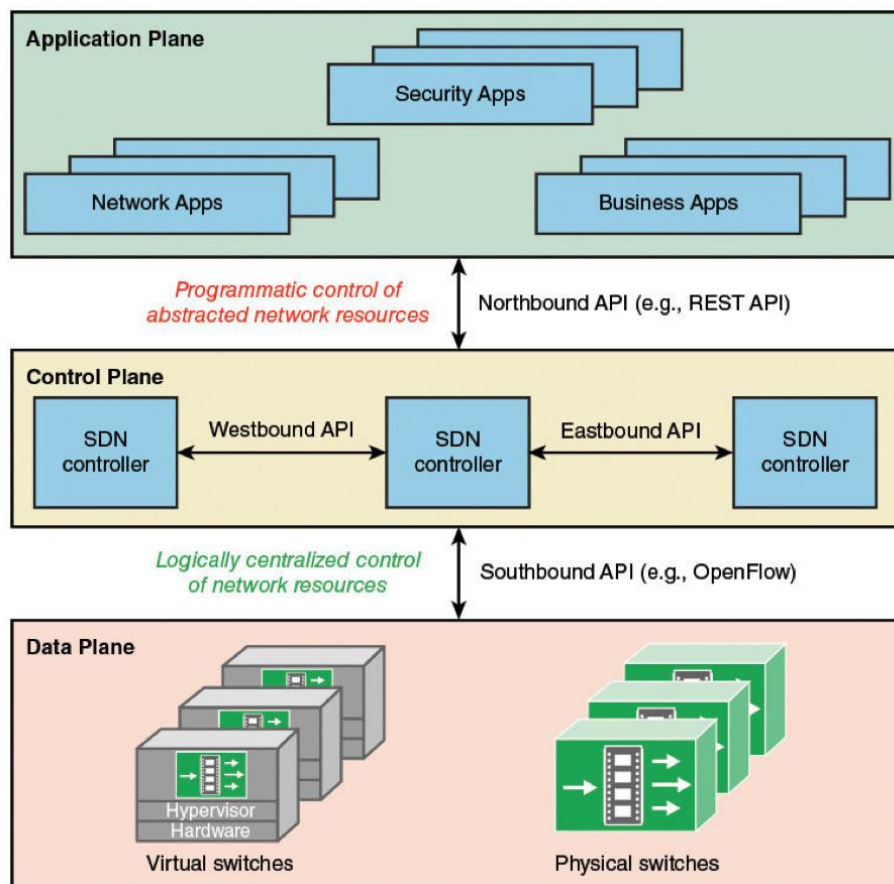


Figure 1.3 – Software-Defined Architecture

1.1.3 OpenFlow protocol and specification. In order to rotate the idea of SDN into functional implementation there has to be a typical rational structure in all the switches, routers, along with additional network products being handled by an SDN controller. This particular logical structure might be implemented in ways that are different on various vendor equipment and in various kinds of network systems, provided that the SDN controller views a consistent logical switch functionality. And standard format, secure protocol is required in between the SDN controller as well as the network unit. These requirements are resolved by OpenFlow, that is both a protocol among SDN controllers as well as a specification and network devices of the rational structure of the system switch functionality. OpenFlow is identified at the OpenFlow Switch Specification, released by the Open Networking Foundation [4].

Message switches which occur between an OpenFlow controller as well as an OpenFlow switch is described by the OpenFlow protocol. This particular protocol offers the SDN controller with three information types to be utilized with controlling the network:

- Event-based messages: Sent by the switch on the controller every time a link or port changes takes place;

- Flow statistics: Generated by the switch primarily based on traffic flow. The controller is enabled by this information to observe traffic, reconfigure the system as necessary, as well as alter flow parameters to meet up with QoS demands;
- Encapsulated packets: Sent by the switch on the controller since there's an explicit steps to transmit this particular package at a flow table entry or perhaps since the switch requires info for starting a brand new flow.

The OpenFlow protocol gives an ability to the controller to change and manage the switch logical structure, without regard to the details of how the switch implements the OpenFlow logical architecture. Figure 1.4 describes the key components of an OpenFlow environment, consisting of SDN controllers including OpenFlow software applications, OpenFlow switches as well as end systems.

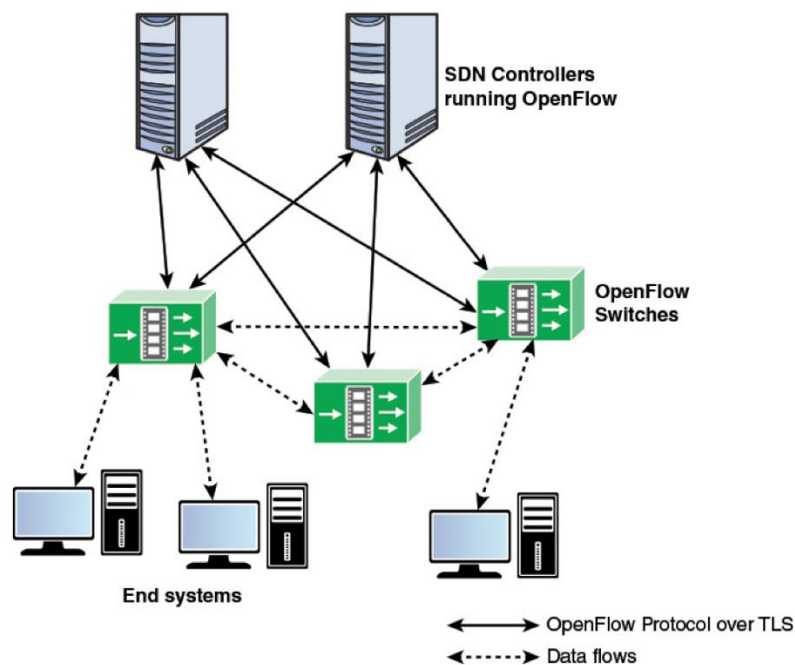


Figure 1.4 – OpenFlow environment

Figure 1.5 displays the key parts of an OpenFlow switch. An SDN controller communicates with OpenFlow suitable switches with the OpenFlow protocol operating around Transport Layer Security. Each switch links to various other switches and also, potentially, to end user equipment which are the sources as well as destinations of packet flows. On the switch facet, the interface is generally known as an OpenFlow channel. These junctions are through OpenFlow ports. An openFlow port additionally links the switch on the SDN controller. Three kinds of ports are defined by OpenFlow.

Physical port corresponds to a hardware user interface of the switch. For instance, on an Ethernet switch, physical ports map one to one on the Ethernet interfaces.

Logical port does not correspond straight to a hardware interface of the switch. Logical ports are higher level abstractions which could be identified in the switch using non OpenFlow techniques as link aggregation groups, loopback

interfaces, tunnels. Logical ports can include packet encapsulation and could map to different physical ports. The processing carried out by the logical port is implementation reliant and should be transparent to OpenFlow processing, and also all those ports should meet up with OpenFlow processing as OpenFlow physical ports.

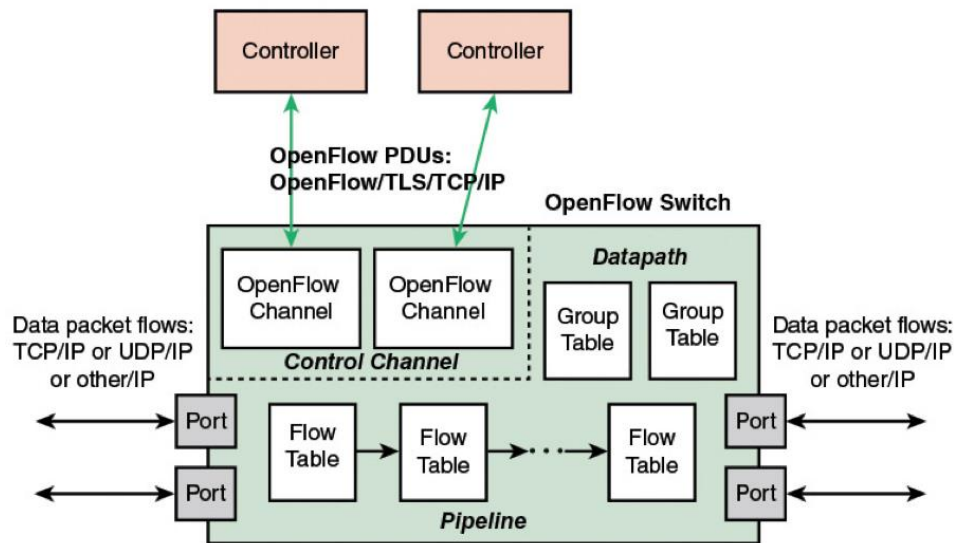


Figure 1.5 – OpenFlow Switch

Reserved port is identified by the OpenFlow specification. It fixes generic forwarding actions including sending to and getting out of forwarding, flooding, or the controller utilizing non OpenFlow solutions , like "normal" switch processing.

Within each switch, a number of tables is utilized to handle the flows of packets with the switch The OpenFlow specification describes three kinds of tables in the logical switch architecture. A flow table matches inbound packets to some articular flow and specifies what functions are being done on the packets. There might be several flow tables which work in a pipeline manner, as explained consequently. A flow table might point a flow to a group table, which might cause a range of measures that affect one or more flows. A meter table is able to cause a range of performance related steps on a flow. Using the OpenFlow switch protocol, the controller is able to lend, redesign, and delete flow entries in tables, both reactively, meaning in reaction to packets and also proactively.

1.1.4 Network Functions Virtualization. A major driving element in the deployment of SDN certainly is the importance to offer flexible networking reaction to the prevalent use of virtualized servers. Virtual machine (VM) technology on the internet or an enterprise network has, until recently, been utilized for application level server capabilities including database servers, e-mail servers, web servers, cloud servers, and so forth. This exact technology, nonetheless, may at the same time be placed on to network equipment, like routers, firewalls, LAN switches, along with IDS/IPS servers (Figure 1.6). Network functions virtualization (NFV) decouples network capabilities, like routing, intrusion detection, firewalling, and

Network Address Translation from proprietary hardware platforms and tools the features in a program. It uses regular virtualization solutions which operate on high performance hardware to virtualize network capabilities.

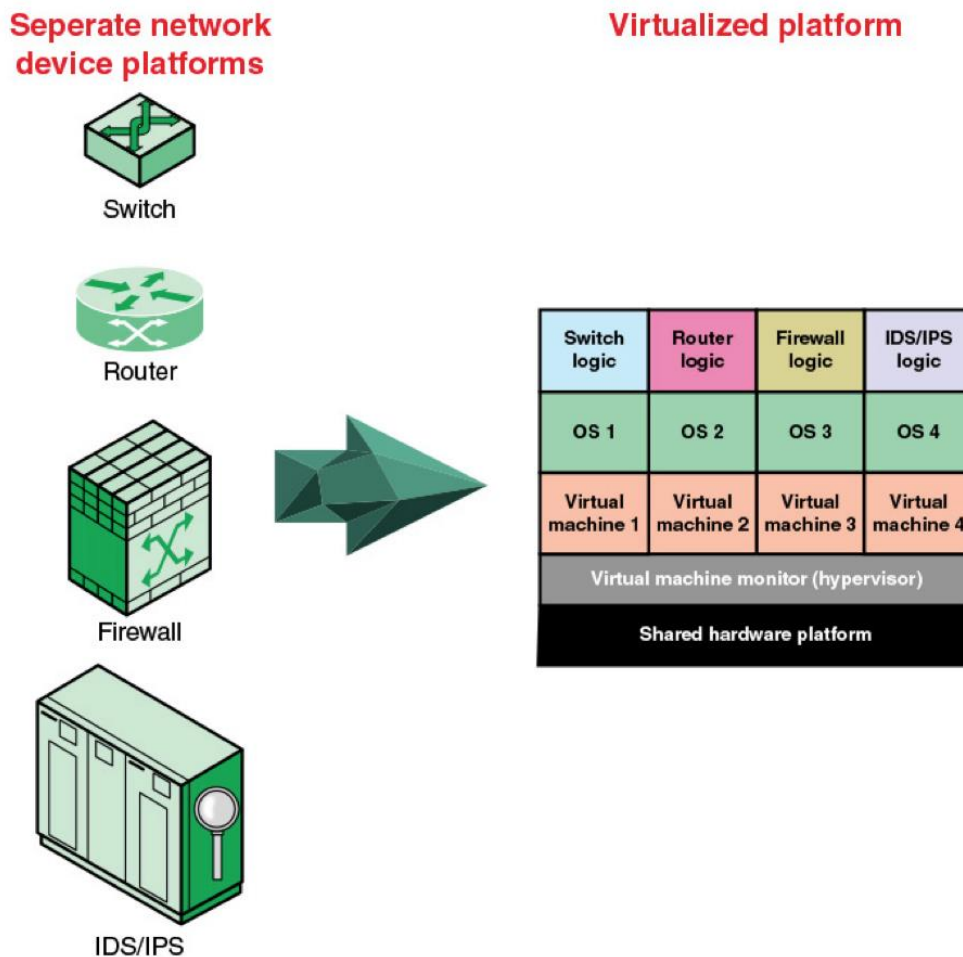


Figure 1.6 – Network Functions Virtualization

It is appropriate to the data plane processing or control plane perform in each wired and wireless network infrastructures.

SDN and NFV are impartial but complementary schemes. SDN decouples the data as well as control planes of network traffic control, making the management and also routing of network traffic much more adaptable and productive. NFV decouples network features from certain hardware platforms via virtualization to come up with the provision of the functions much more effective and versatile. Virtualization can be utilized on the data plane functions of the routers along with other network capabilities, including SDN controller functions.

1.1.5 Traditionally, software programs have run right on an operating system (OS) on a personal computer (PC) or on a server. Each PC or server would run only one OS at a time. Hence, application vendors must rewrite elements of their applications for every platform they would operate on and help, that enhanced some time to market for brand new capabilities or functions, enhanced the chance of defects, improved quality evaluation initiatives, and in most cases resulted in

heightened value. In order to support several operating systems, software vendors must develop, manage, and also help many operating and hardware system infrastructures, a resource-intensive and costly procedure. One powerful strategy for offering with this issue is recognized as hardware virtualization. Virtualization technology allows one server or PC to concurrently run several operating systems or perhaps numerous sessions associated with a single OS. A piece of equipment running virtualization program is able to host a number of software applications, which includes the ones that operate on various operating systems, on an individual hardware platform. Essentially, the host operating system is able to support a selection of virtual machines (VMs), each of that has the qualities of a specific OS and, in certain variations of virtualization, the qualities of a specific hardware platform.

The answer which allows virtualization is a virtual machine monitor (VMM), or typically known today as being a hypervisor. This particular application rests in between the hardware and also the VMs acting as being an useful resource broker (Figure 1.7). To put it simply, multiple VMs is allowed by the hypervisor to easily coexist on a single physical server host as well as share that host 's resources and capabilities.

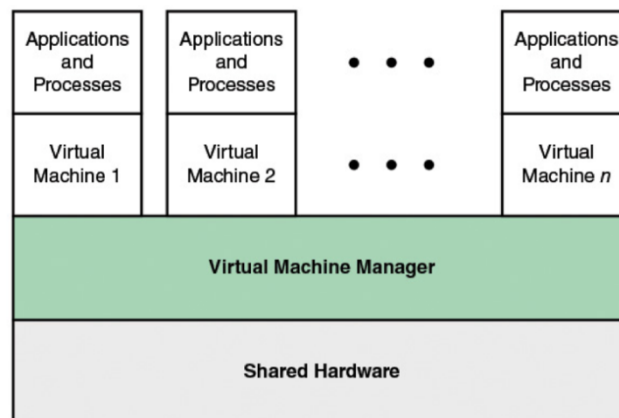


Figure 1.7 – Virtual Machine Concept

The VM strategy is a very common way for individuals and businesses in order to cope with legacy software and also to optimize the hardware usage of theirs by maximizing the different kinds of uses that a single computer system is able to deal with. Business hypervisor offerings by organizations including Microsoft and VMware are commonly used, with countless copies having been offered. A vital element of server virtualization is the fact that, additionally to the ability of managing several VMs on a single machine VMs can be seen as network resources. Server virtualization masks server resources, like the amount as well as identity of specific physical servers, processors, along with operating systems from server users. It is then possible to partition a single host in several impartial servers, saving hardware resources. It likewise causes it to be easy to easily migrate a server from a single machine to the next for load balancing or even for powerful switchover in the situation of machine failure. Server virtualization has turned into a central aspect in

dealing with big data applications and in applying cloud computing infrastructures [5].

Virtual machine technology allows migration of committed database and application servers to x86 servers. Precisely the same technology could be applied to network based devices, along with the following:

- Network function equipment for example switches, routers, network access points, customer premises equipment (CPE), and deep packet inspectors for profound packet inspection;
- Network related compute equipment like as firewalls, intrusion detection systems, and network management systems;
- Network attached storage: database and file servers connected to the network.

1.1.6 In conventional networks, most devices are deployed on proprietary and closed platforms. Most network components are enclosed containers, and hardware cannot be shared. Although this particular hardware is actually nonproductive if the method is working under capability, each unit needs extra hardware for increased capacity. With NFV system components are impartial applications which are flexibly deployed on a single platform comprising typical servers, switches, and storage devices. This way, software as well as hardware are decoupled, and also limit for every application is elevated or even decreased by including or reducing virtual resources (Figure 1.8).

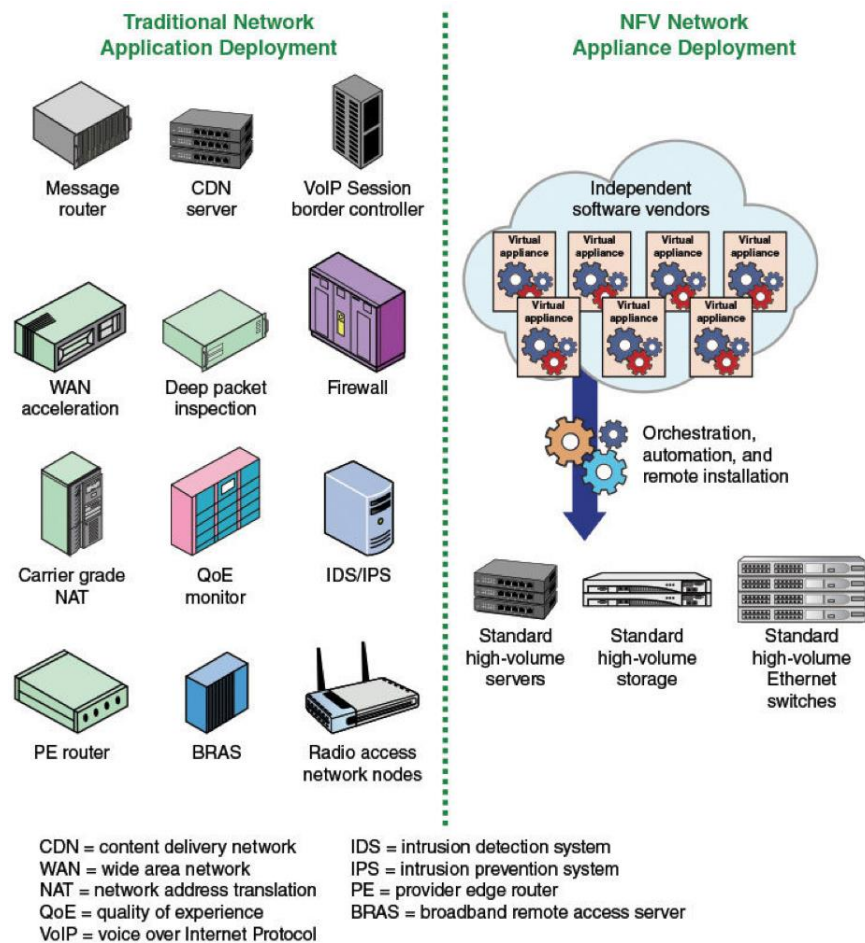


Figure 1.8 – Vision for Network Functions Visualization

Three major NFV concepts are linked to create useful network services:

- Service chaining: Virtual Network Functions (VNF) are modular and every VNF offers restricted performance by itself. For a certain traffic flow in a certain application, the service provider steers the flow through several VNFs to accomplish the preferred network functionality. This is described as service chaining;

- Management and orchestration (MANO): This includes deploying as well as controlling the lifecycle of VNF cases. VNF example development, VNF service chaining, shutdown, relocation, monitoring, and billing is included by examples. MANO in addition manages the NFV infrastructure components;

- Distributed architecture: A VNF might be made up of one or more VNF components (VNFC), each of that implements a subset of the VNF's features. Every VNFC may be deployed in a single or even numerous instances. These instances might be deployed on different, distributed hosts to offer redundancy and scalability.

- If NFV is implemented effectively and efficiently, it can provide a variety of benefits as compared to conventional networking approaches;

First of all, decrease of costs by utilizing commodity servers and switches, consolidating equipment, exploiting economies of scope, and supporting pay-as-you

grow designs to eradicate wasteful overprovisioning. In terms of power usage as well as room consumption, by utilizing commodity servers and switches, consolidating gear, and exploiting economies of scope, and also decreased network management as well as control expenditures.

Then, the capability to innovate as well as move out services fast, decreasing the time to deploy brand new networking services to allow for altering business needs, grab new market opportunities, and also boost return on investment of innovative services. Additionally reduces the risks associated with coming out new solutions, enabling providers to quickly trial and evolve offerings to find out what best meets the requirements of customers.

Use of an individual platform for various purposes users as well as tenants. This enables network operators to tenants and users. This enables network operators to share resources across services and throughout various client bases.

In order to provide these advantages, NFV should be created as well as implemented to meet up with a selection of technical challenges and requirements, like the following: Portability/interoperability: The ability to load as well as execute VNFs offered by diverse vendors on a range of standardized hardware platforms.

1.2 VLAN

1.2.1 A VLAN is number of devices on a single or more LANs which are configured to communicate as in case they had been placed on similar wire, when actually they are placed on a variety of distinct LAN sections. Because VLANs are derived from logical rather than physical connections, they are incredibly flexible.

Broadcast domains in a level two community are explained by VLANs. A broadcast domain will be the set of all devices which will get broadcast frames originating out of any device inside the set. Broadcast domains are generally bounded by routers since routers do not forward broadcast frames. Layer 2 switches cause broadcast domains depending on the setup of the switch. Switches are multiport bridges which enable you to create several broadcast domains. Every broadcast domain is similar to a distinct virtual bridge inside a switch.

The VLAN reasoning is applied in LAN switches as well as features in the MAC layer. Since the goal is isolating traffic in the VLAN, a router is necessary to connect from one VLAN to another. Routers could be applied as individual products, so that visitors from one VLAN to the next is directed to some router, or maybe the router logic could be implemented during the LAN switch, as shown in Figure 1.9.

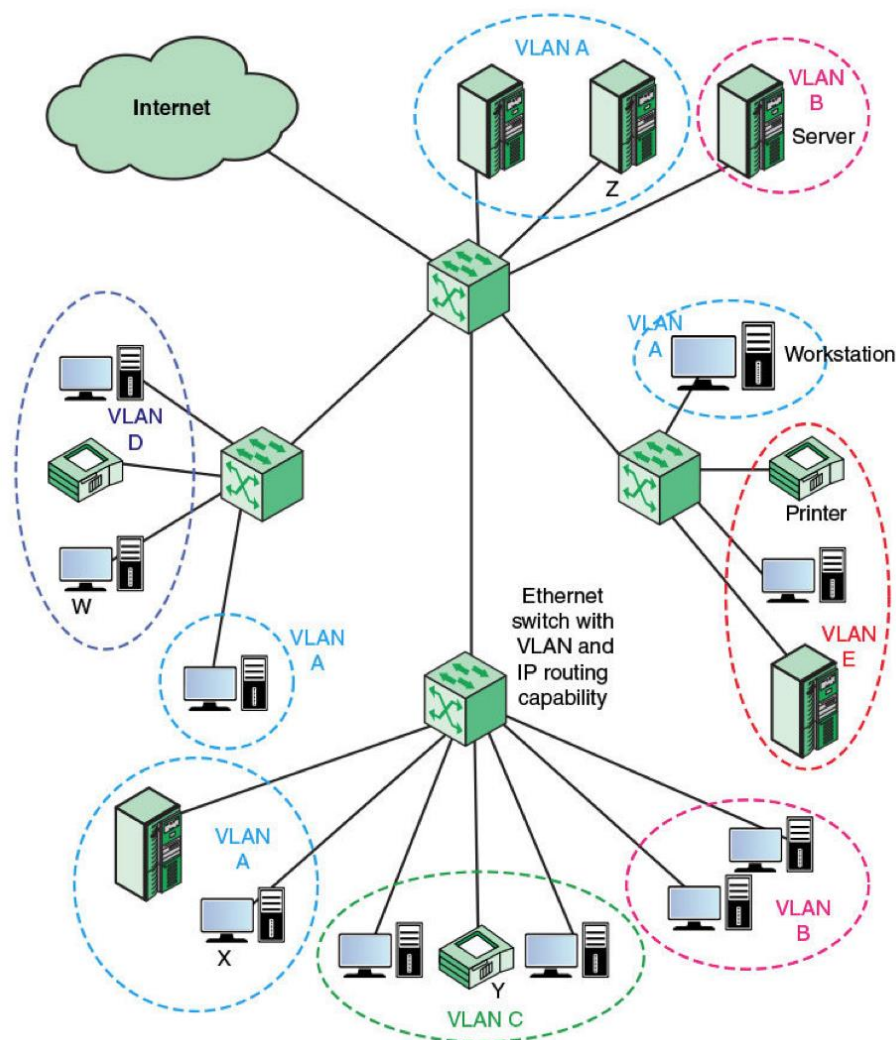


Figure 1.9 – VLAN Configuration

VLANs enable any institution to be physically dispersed all over the company while keeping its group identity. For instance, accounting personnel may be placed on the retail store floor, in the research and development center, in the cash disbursement office, and in the corporate and business offices, while most participants are located on an equivalent virtual system, sharing traffic just with one another.

Figure 1.9 shows five defined VLANs. A transmission from workstation X to server Z is in the same VLAN, therefore it is well switched at the MAC level. A transmitted MAC frame from X is transmitted to other products in most areas of the identical VLAN. But a transmission from X to printer Y moves from one VLAN to another. Appropriately, router logic at the IP level is necessary to move the IP packet coming from X to Y. Figure 1.9 shows that logic incorporated into the switch, to ensure that the switch determines if the new MAC frame is destined for one more device on a single VLAN. In case not, the switch routes the enclosed IP packet at the IP level.

1.2.2 A VLAN is a broadcast domain comprising of number of end terminals, certainly on several physical LAN segments, which re not constrained by their

physical location and will connect as in case they had been on the same LAN. Some means is as a result required for determining VLAN membership.

Membership by port team. Every switch in the LAN setup has two kinds of ports: a trunk port, which links two switches; as well as an end port, that links the switch to an end system. A VLAN could be identified by setting each end port to a certain VLAN. This technique has got the benefit that it is reasonably simple to configure. The process disadvantage would be that the network supervisor should reconfigure VLAN membership when an end system moves through one port to another.

Membership by MAC address: Since MAC layer addresses are hardwired into the workstation's network interface card (NIC), VLANs dependent on MAC addresses allow network administrators to move a workstation to an alternative physical spot on the network and also have that workstation instantly retain its VLAN membership. The primary problem with this method is the fact that VLAN membership should be assigned initially. In networks with a huge number of users, this is no simple task. Additionally, in environments where notebook PCs are utilized, the MAC address is related with the docking station and never with the notebook PC. Consequently, when a notebook PC is transferred to an alternative docking station, its VLAN membership has to be reconfigured.

Membership based on protocol information: VLAN membership is often given according to IP address, transport protocol information, as well as higher layer protocol information. This is a rather adaptable method, though it can involve switches to look at parts of the MAC frame above the MAC layer, which might have a functionality impact.

1.2.3 The IEEE 802.1Q standard format, previous updated in 2014, describes the functioning of VLAN bridges as well as switches which permits the meaning, functioning, and administration of VLAN topologies inside a bridged or switched LAN infrastructure. In this particular section, we focus on the application of the image standard to 802.3 LANs. Recall that the vLAN is an administratively configured broadcast domain, consisting of a subset of end stations connected to a LAN. A VLAN is not restricted to just one switch but could span a couple of interconnected switches. If so, traffic between switches have to indicate VLAN membership. This is accomplished in 802.1 valuation in the range from 1 to 4094. Every VLAN in a LAN setup is given a globally distinctive VID. By setting the identical VID to conclude methods on numerous switches, one or more VLAN broadcast domains will be extended throughout a large network.

Figure 1.10 presents the placement as well as information in the 802.1 tag, known as Tag Control Information (TCI). The TCI includes three subfields:

- User priority (3 bits) which is the priority level for this frame;
- Canonical format indicator (1 bit) which is usually set to zero for Ethernet switches. CFI is utilized for compatibility between Ethernet style networks as well as Token Ring type networks. In case a frame obtained at an Ethernet port features a CFI set up to 1, which frame should not be forwarded as it is to an untagged port;

– VLAN identifier (12 bits) - the identification of the VLAN. Of the 4096 likely VIDs, a VID of 0 is used to recognize the TCI has just a high priority worth, along with 4095 (0xFFFF) is reserved, therefore the maximum possible amount of VLAN configurations is 4094.

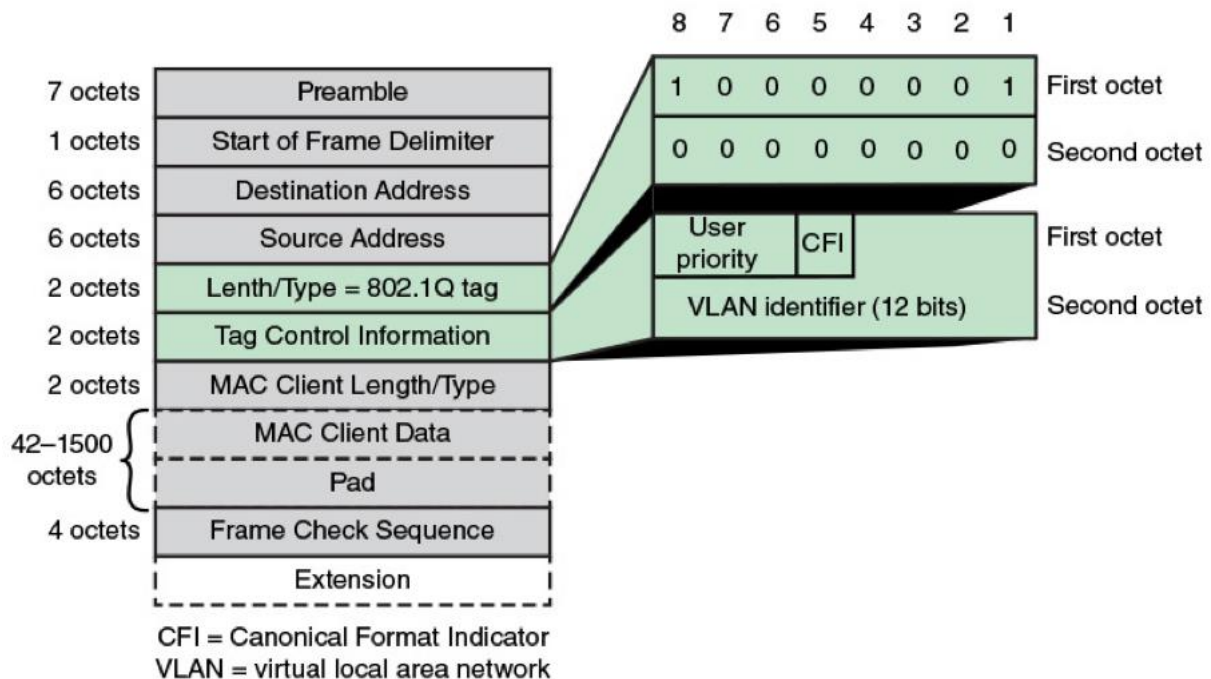


Figure 1.10 – Tagged IEEE 802.3 MAC Frame Format

1.2.4 OpenFlow VLAN Support A regular 802.1Q VLAN calls for that the network switches work with a total understanding of the VLAN mapping. This knowledge might be manually configured or acquired automatically. Another disadvantage is connected to the option of one of three ways of defining group membership (port group, protocol information), MAC address. The network administrator should assess the trade-offs based on the network type they would like to deploy and select one of the possible solutions. It will be tough to deploy a far more flexible characterization of a VLAN or perhaps a customized definition with conventional networking equipment. Reconfiguring VLANs is additionally a tough process for administrators: Multiple switches and routers have to be reconfigured whenever VMs are moved.

SDN, and particularly OpenFlow, enables far more adaptable control and management of VLANs. It must be apparent just how OpenFlow is able to put together flow table entries for forwarding based on a single or both VLAN tags, and also just how tags are usually included, modified, and deleted.

1.3 VXLAN

1.3.1 Virtual Extensible LAN protocol (VXLAN) technology enables networks to support more VLANs. Based on the IEEE 802.1Q standard, conventional VLAN identifiers are 12 bits in length - this specific naming limits

networks to 4094 VLANs. The VXLAN protocol overcomes this limitation by implementing an extended logical network identifier which allows more VLANs and also, consequently, a lot more logical network isolation for substantial networks like clouds which generally include a number of virtual machines.

VXLAN technology enables to segment networks as VLANs do, though it offers advantages which VLANs cannot. In theory it could be created as much as 16 million VXLANs in an administrative domain [6].

Utilizing VXLANs to develop reduced Layer 2 domains that are attached over a Layer 3 network implies that there is no requirement to use Spanning Tree Protocol (STP) to converge the topology but tend to work with much more sturdy routing protocols in the Layer 3 network instead. In the absence of STP, not any of the links are obstructed, meaning that we are able to get complete value from each one of the ports. Using routing protocols to connect Layer 2 domains additionally allows to load-balance the visitors to ensure receiving the most effective use of accessible bandwidth. Because of the quantity of east west traffic which usually flows within or between data centers, maximizing network effectiveness for that traffic is extremely important.

1.3.2 VXLAN Encapsulation and Packet Format. VXLAN is a layer 2 overlay scheme over a layer 3 network. It utilizes MAC Address-in-User Datagram Protocol (MAC-in-UDP) encapsulation to supply a way to lengthen Layer 2 segments throughout the data center network. VXLAN is a strategy to help versatile, large-scale multitenant surroundings with a shared typical physical infrastructure. The transport protocol with the physical data center network is IP plus UDP.

VXLAN describes a MAC-in-UDP encapsulation system where the initial Layer 2 frame features a VXLAN header added and then will be placed in an UDP-IP packet. With this particular MAC-in-UDP encapsulation, VXLAN tunnels Layer 2 network over Layer 3 network. The VXLAN packet format is demonstrated in Figure 1.11.

VXLAN presents an 8-byte VXLAN header which consists of a 24-bit VNID and several reserved bits. The VXLAN header combined with the initial Ethernet frame moves in the UDP payload. The 24-bit VNID is utilized in order to determine Layer 2 segments and also to preserve Layer 2 isolation among the segments.

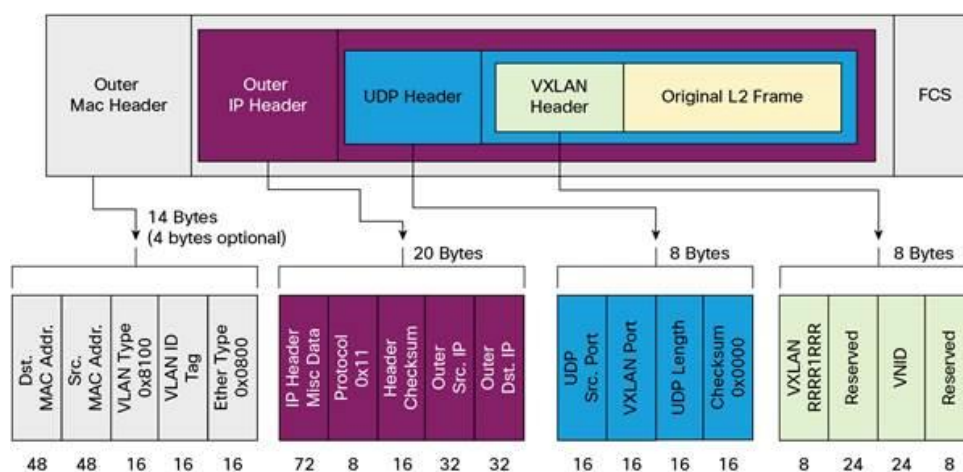


Figure 1.11 – VXLAN Packet Format

External MAC header has the MAC address of the source VXLAN tunnel endpoint (VTEP) and also the MAC address of the next-hop router. Each router across the packet's route rewrites this particular header therefore the source address will be the router's MAC address and also the destination address will be the next-hop router's MAC address.

Outer IP header incorporates the IP addresses of the source as well as destination VTEPs.

Source UDP port - the VXLAN protocol repurposes this standard held in a UDP packet header. Rather than running this particular field for the source UDP port, the protocol employs it as a numeric identifier for the specific flow among VTEPs. The VXLAN standard does not define the way this particular amount is derived, though the source VTEP typically calculates it starting from a hash of some conjunction of elds from the internal Layer 2 packet and also the Layer 3 or Layer 4 headers.

Destination UDP port is the VXLAN UDP port. The internet Assigned Numbers Authority allocates port 4789 to VXLAN.

Outer UDP header contains source as well as destination UDP ports.

1.3.3 VXLAN utilizes VTEP devices in order to chart tenants' end devices to VXLAN segments and also to do VXLAN encapsulation and de-encapsulation. Any VTEP function has two interfaces: One is a switch interface on the local LAN segment to allow for local endpoint interaction via bridging, and also the other is an IP interface on the transport IP network [7].

The IP interface features a distinctive IP address which identifies the VTEP device on the transport IP network referred to as the infrastructure VLAN. The VTEP device employs this particular IP address to encapsulate Ethernet frames and sends the encapsulated packets on the transport network with the IP interface. A VTEP device additionally finds the remote VTEPs for the VXLAN segments of its and also learns remote MAC Address-to-VTEP mappings through its IP interface. The purposeful parts of VTEPs as well as the logical topology that is made for Layer 2 connectivity throughout the transport IP network is shown in Figure 1.12.

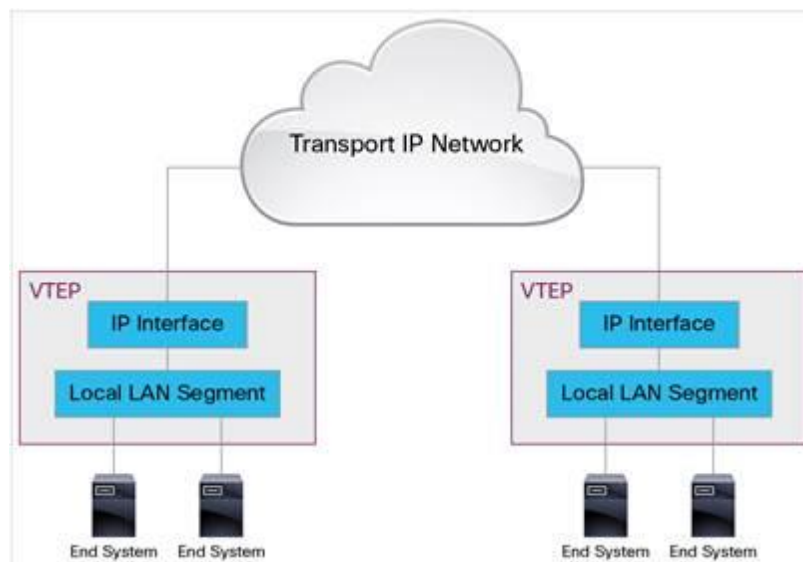


Figure 1.12 – VTEP

The VXLAN segments are liberated from the basic network topology; alternatively, the underlying IP network among VTEPs is free from the VXLAN overlay. It routes the encapsulated packets depending on the external IP address header, that has the initiating VTEP as the source IP address and also the terminating VTEP as the destination IP address.

1.3.4 VXLAN Packet Forwarding Flow. VXLAN works by using stateless tunnels between VTEPs to transmit traffic of the overlay Layer 2 network with the Layer 3 transport network. A good example of a VXLAN packet forwarding flow is shown in Figure 1.13.

In Figure 1.13, Host-A and Host-B in VXLAN segment 10 communicate with each other over the VXLAN tunnel between VTEP-1 and VTEP-2. This particular instance assumes that address learning continues to be accomplished on each side, and also corresponding MAC-to-VTEP mappings are available on both VTEPs.

When Host-A transmits traffic to Host-B, it forms Ethernet frames with MAC-B address of Host-B as the destination MAC address and sends them away to VTEP-1. VTEP-1, with a mapping of MAC-B to VTEP-2 in its mapping table, executes VXLAN encapsulation on the packets by including VXLAN, UDP, and outer IP address header to it. In the external IP address header, the source IP address is the IP address of VTEP-1, so the end point IP address is the IP address of VTEP-2. VTEP-1 subsequently performs an IP address lookup for the IP address of VTEP-2 to deal with the other hop in the transit network and also consequently utilizes the MAC address of the next-hop unit to additionally encapsulate the packets in an Ethernet frame to transmit to the next-hop device.

The packets are routed toward VTEP-2 with the transport network dependent on their external IP address header, that has the IP address of VTEP-2 as the end point address. Following the packets are received by VTEP-2, it strips from the exterior Ethernet, IP, UDP, and VXLAN headers, along with forwards the packets to Host B, dependent on the initial location MAC address in the Ethernet frame.

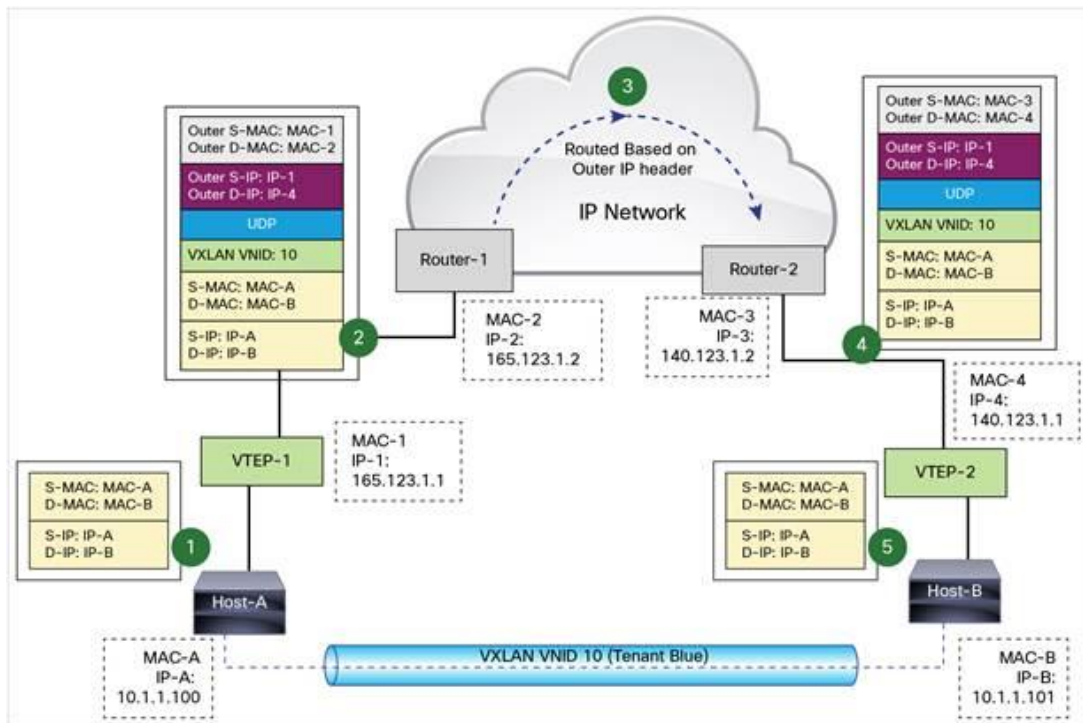


Figure 1.13 – VXLAN Unicast Packet Forwarding Flow

1.4 IPv6

1.4.1 IPv6 is the substitute Internet protocol for IPv4. Several of the inadequacies of IPv4 is corrected by it and also simplifies the way in which that addresses are set up and also the way they're handled by Internet hosting companies.

IPv4 has shown to be sturdy, quickly applied, and interoperable, and has stood the check of scaling an internetwork to a worldwide utility the dimensions of the Internet. Nevertheless, the original design did not anticipate recent exponential growth of the Internet as well as the approaching exhaustion of the IPv4 address space as well as the following conditions:

- The capability of Internet backbone routers to keep large routing tables;
- Simpler autoconfiguration and renumbering necessity;
- Requirement for security within the IP level;
- Demand in better support for real-time delivery of data.

With the 32-bit address format of its, IPv4 is able to withstand an optimum 4.3 billion unique IP addresses. While this particular number might seem huge, it is not sufficient to sustain as well as scale the quickly rising development of the Internet. Though improvements to IPv4, including the usage of NAT, have permitted the prolonged usage of the protocol, address exhaustion is unavoidable.

With its 128-bit address format, IPv6 is able to support 3.4×10^{38} or 340,282,366,920,938,463,463,374,607,431,768,211,456 custom IP addresses. This particular selection of addresses is big enough to configure a distinctive address on each node in the world wide web but still have loads of addresses left over. It is likewise large enough to eliminate the demand for NAT, that has its own inherent problems.

A number of countries around the world, governmental agencies, along with multinational corporations have sometimes currently deployed or even mandated deployment of IPv6 in their networks as well as application solutions. Several emerging nations have no option but in order to deploy IPv6 due to the unavailability of new IPv4 addresses.

Apart from providing a nearly endless amount of distinctive IP addresses for worldwide end-to-end reachability as well as scalability, IPv6 has got the next extra advantages:

- Simplified header format for effective packet handling;
- Larger payload for enhanced throughput as well as transport efficiency;
- Hierarchical network architecture for routing efficiency;
- Hierarchical network structure for routing effectiveness;
- Support for extensively deployed routing protocols (OSPF, BGP, etc.);
- Plug-And-Play and autoconfiguration support;
- Increased number of multicast addresses.

Moreover, IPv6 provides elimination of necessity for network address translation (NAT) and application layered gateway (ALG).

The primary IPv6 header is the same as the fundamental IPv4 one despite a number of field variations which are the outcome of lessons learned by operating IPv4. Figure 1.14 presents the IPv6 and IPv4 main headers.

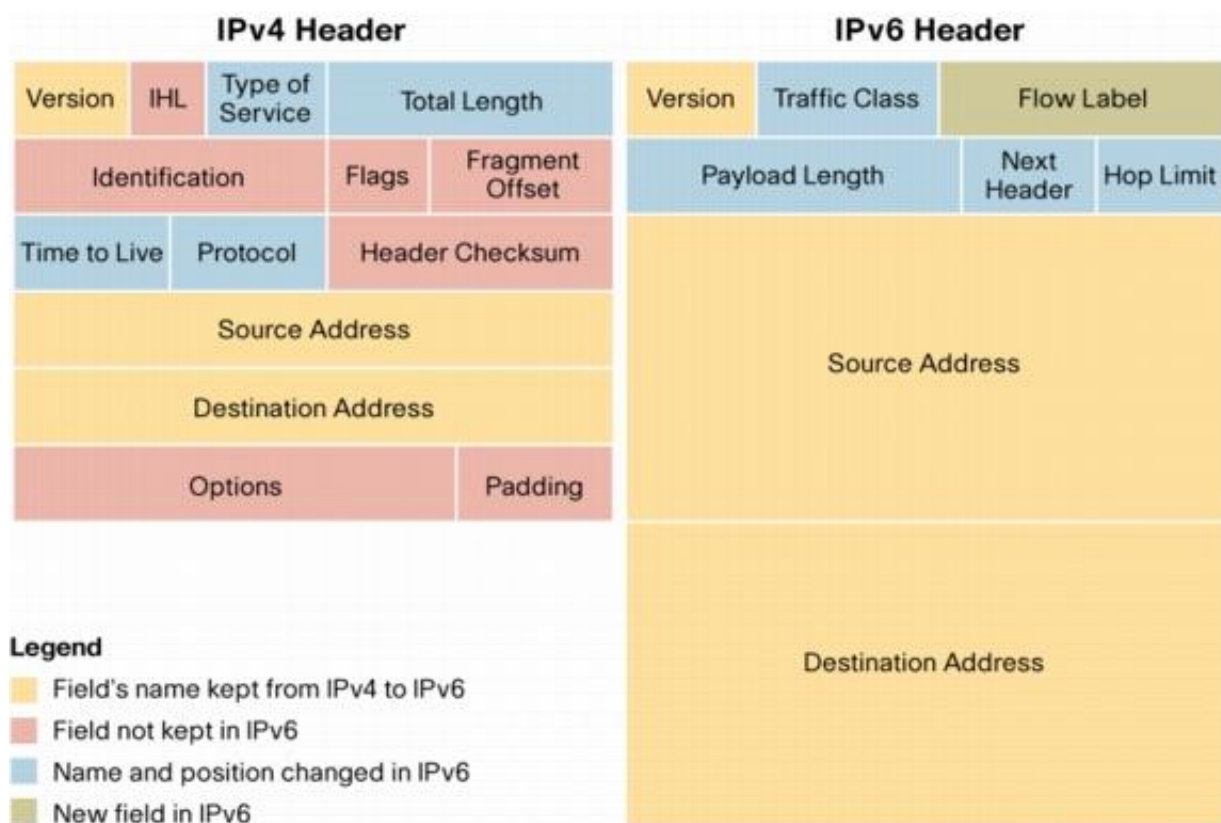


Figure 1.14 – IPv4 and IPv6 Headers

Table 1.1 – Description of IPv6 Header

Field	Description
Version (4-bits)	Represents the Internet Protocol version.
Traffic Class (8-bits)	These 8 bits are split into two parts. The most important 6 bits are utilized for Type of Service to allow the router known what services must be offered to this particular packet. The very least significant 2 bits are utilized for Explicit Congestion Notification (ECN).
Flow Label (20-bits)	This kind of label is utilized to keep the sequential flow of the packets belonging to a communication. The sequence is labeled by the source to assist the router identify that a specific packet belongs to a certain flow of information. This particular field helps avoid re-ordering of data packets. It is designed for streaming/real time media.
Payload Length (16-bits)	This field is used to express to the routers the amount of information a specific packet contains in its payload. Payload is composed of Upper Layer and extension Headers data. With 16 bits, up to 65535 bytes could be indicated; but provided that the Extension Headers include Hop-by-Hop Extension Header, subsequently the payload might surpass 65535 bytes and also this particular field is set to 0.
Next Header (8-bits)	This field is utilized to show both the kind of Extension Header, or in case the Extension Header is not present then it signifies the upper Layer PDU. The values for the kind of Upper Layer PDU are exact same as IPv4's.
Hop Limit (8-bits)	This field is used to stop packet to loop in the network infinitely. This is exact same as TTL found in IPv4. The importance of Hop Limit field is decremented by 1 as a link is passed by it (router/hop). If the field reaches 0 the packet is discarded.
Source Address (128-bits)	The address of originator of the package is indicated by this field.
Destination Address (128-bits)	The address of intended receiver of the package is provided by this field.

1.4.2 IPv6 addresses are 128 bits in length. They are logically divided into a network prefix along with a host identifier. The amount of bits in the network prefix is represented by a prefix length, for instance, /64. The other bits are utilized for the host identifier. In case a prefix length is not specified for an IPv6 address, the default prefix length is /64 (Figure 1.15).

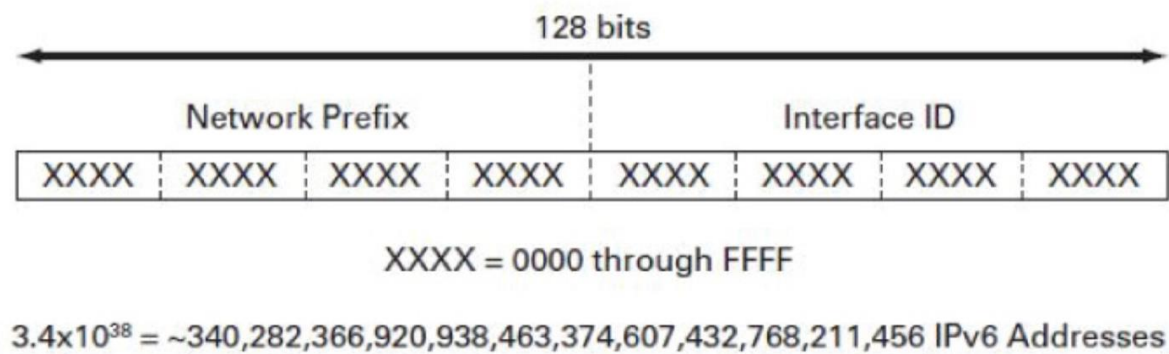


Figure 1.15 – IPv6 Address Format

Each IPv6 address type possesses a scope which identifies the part of the network in which the address is different. Some IPv6 addresses are distinctive just in a subnet or a local network (link local scope), others are unique in individual networks or perhaps between organizations (unique local scope), while yet others are globally unique (global scope), that is, everywhere in the Internet.

There is the compressed IPv6 standard address structure where leading zeros in a 16-bit block are not shown and the longest string of 16-bit address blocks is compressed. Although the command line interface accepts both compressed as well as uncompressed addresses, most IPv6 addresses are going to display as compressed. For instance, the following 2 addresses are equivalent:

2001:0000:ABCD:EF22:0000:1234:5678:0001

2001::ABCD:EF22:0:1234:5678:1 The double colon (: :) in the second address shows that a string of zeros was omitted. You are able to use this compressed format just once in an address. In case there are actually a couple of contiguous strings of zeros in an address, you are able to change all of them with a two-fold colon (: :). Although not trailing zeros, top zeros can additionally be omitted.

The next kinds of addresses are supported by IPv6 addresses:

- Unicast Addresses;
- Multicast Addresses.

1.4.3 IPv6 supports a number of kinds of unicast addresses: anycast, unique-local, link-local, and global addresses.

A global IPv6 address (Figure 1.16) is a unicast address with a predefined prefix of 2000::/3 (001). IPv6 addresses with a prefix of 2000::/3 (001) through E000::/3 (111), excluding the FF00::/8 (1111 1111) multicast addresses, are needed to have 64-bit interface identifiers (VLAN IDs) in the IEEE 64-bit Extended Universal Identifier (EUI-64) structure. The internet Assigned Numbers Authority (IANA) allocates the IPv6 address space in the assortment of 2001::/16 to the registries.

A global unicast address generally contains a 48-bit global routing prefix and a 16-bit subnet ID. In the IPv6 aggregatable global unicast address structure document (RFC 2374), the global routing prefix provided two other hierarchically

organized fields called Top-Level Aggregator and Next-Level Aggregator. Simply because these fields were policy-based, the IETF determined to eliminate the fields from the RFCs. Nevertheless, several current IPv6 networks deployed in the first days may continue to use networks depending on the older architecture.

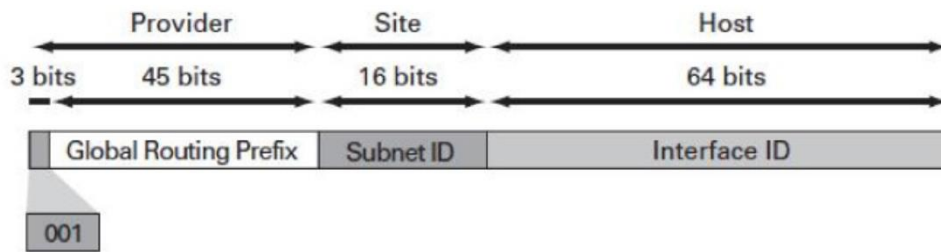


Figure 1.16 – Global IPv6 Address Format

A 16-bit subnet field known as the Subnet ID may be used by specific organizations in order to generate their own personal local addressing hierarchy as well as to identify subnets. An organization is allowed by this field to use as many as 65,535 particular subnets.

A link-local address (Figure 1.17) is a unicast address which features a range of the local link just and one is needed on each interface for IPv6 to function. The Application Control Module (ACE) instantly generates a link-local address for each IPv6-enabled interface working with the EUI-64 format. Conversely, the ACE accepts a user-configured link-local address. Each link-local tackle has a predefined prefix of FE80::/64.

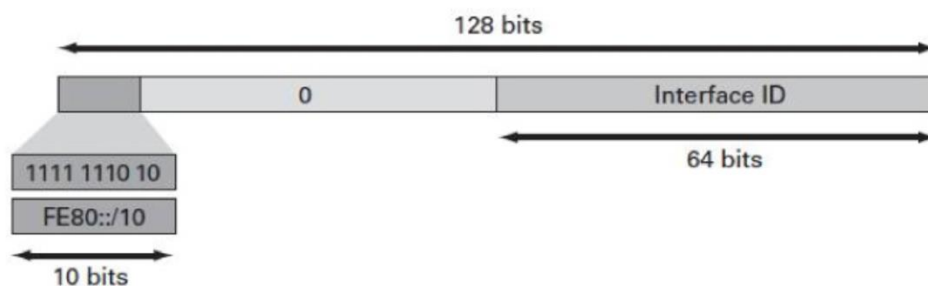


Figure 1.17 – Link-Local Address Format

Link-local addresses hold the following characteristics:

- Automatically assigned whenever you enable IPv6 working with the `ipv6 enable` command;
- Used for next hop calculations in routing protocols;
- The first ten bits of the prefix are always 1111 1110 10 (FE80::/64);
- The last 54 bits of the prefix can be zero or any configured value.

A unique-local address (Figure 1.18) is a unicast address which is valid only inside a site or enterprise and between a restricted number of sites. It is used for localized communications and intersite VPNs. Unique-local addresses are much like IPv4 private addresses and are not routable on the web.

The initial seven bits of the prefix are predefined as 1111 110 (FC00::/7). FC00::/7 is split into two /8 blocks: FC00::/8 (eighth most significant bit set to 0) and FD00::/8 (eighth MSB set to 1). FC00::/8 is not defined yet. FD00::/8 is needed with /48 prefixes by placing the 40 least significant bits (LSBs) to a randomly generated bit string.

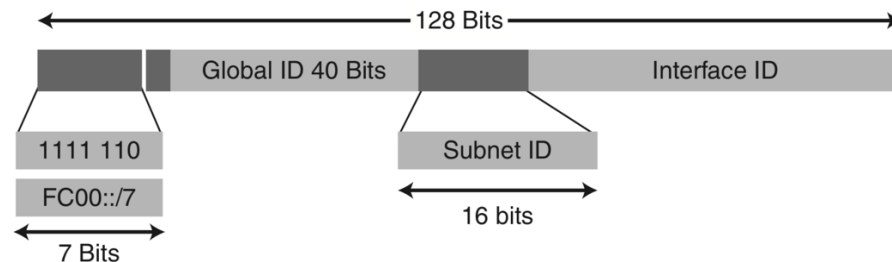


Figure 1.18 – Unique-Local Address Format

An IPv6 anycast address is an address which is given to much more than a single interface. Generally, the address belongs to various nodes. A packet which is delivered to an anycast address is routed to the closest interface which has that address.

Figure 1.19 shows IPv6 multicast address, it has a predefined prefix of FF00::/8 (1111 1111). 1/256 of the entire IPv6 address room is used by the multicast address range. An IPv6 multicast address is an identifier for a set of interfaces which generally belong to various nodes.

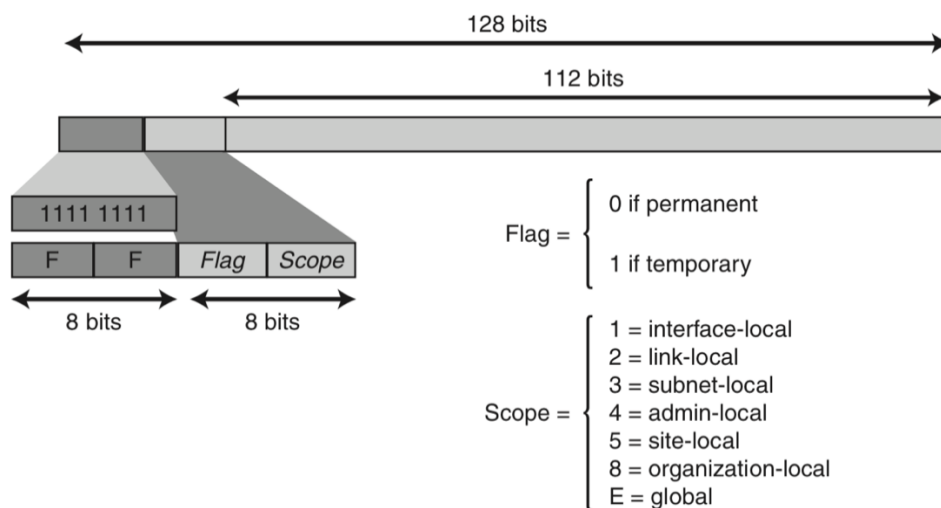


Figure 1.19 – Multicast Address Format

The lifetime as well as range of the multicast address are defined by the next octet adopting the predefined prefix. A long term (well-known) multicast address has a lifetime parameter equal to 0 and it is given by IANA. A short-term multicast address has a lifetime parameter equal to 1 and it is dynamically assigned. A multicast address which provides the scope of an interface, link, subnet, admin, site,

enterprise, or a worldwide scope has a scope parameter of 1, 2, 3, 4, 5, 8, or E, respectively. The IPv6 addressing system is created to help countless multicast group addresses.

2 Technical part

2.1 Troubleshooting prerequisites

Troubleshooting is of all the most commonplace sorts of issue solving. when troubleshooting a defective modem or maybe communication issues in an advertising as well as advertising business, troubleshooting efforts to isolate fault states in a device and regain or even upgrade the defective components in case you would like to reinstate the device to consistent operation. Troubleshooting is generally associated with the repair of physical, mechanical, or electronic systems. Locating factors of failures and errors, tracking down misconfigurations inside the frequently difficult interconnection networks of heterogeneous networking devices is rather a challenge. what is greater, the superiority of progressively complicated software pieces, due to the approaching software defined networks, gives distributed software package debugging as an extra issue to deal with. To cope with this developing complexity, the networking analysis community shows using knowledge based selection support combined with the typical network tracking and diagnostic tools, as well as the transformation of troubleshooting into a relatively automated technique [8].

On the continuum of issues from well-structured (algorithms, story troubles) to unwell-established (systems analysis, design), troubleshooting issues are in the center. Troubleshooting issues:

- appear ill-defined since the troubleshooter should figure out what information is required for trouble diagnosis (which data concerning the power and fuel systems are required in troubleshooting an automobile that will not start);
- require the building of sturdy conceptual model of the system actually being troubleshot (how do electric, energy, and mechanical solutions interact) usually possess a fault state, though several faults might happen simultaneously (e.g., flawed battery, clogged injector);
- have recognized answers with readily interpreted achievement criteria (part replacement results to model restart);
- rely most effectively on experience based rules for diagnosing the majority of the cases, which makes it harder for novices to study (mechanics rely initially on experiences for diagnosis);
- require learners making judgments about the dynamics of the issue, and vary considerably in terminology of method complexity as well as dynamicity (age, engine size, manufacturer, reliance on computer settings in the automobile) [9].

Troubleshooting is predominately a cognitive problem which consists of the attempt to find probable causes of faults by way of a doubtlessly substantial difficulty spot of potential reasons. more to fault detection or fault analysis, troubleshooting typically entails the maintenance or replacement of the defective

unit. The focus in troubleshooting, although, is on fault diagnosis, and that calls for attempt to find the parts of the system that are producing substandard outputs (cause of discrepancy). Troubleshooters then search for actions to be able to effectively eliminate the discrepancy.

Troubleshooting is generally taught as a linear sequence of choices which point the fault isolation. Determination tables and flowcharts are generally used to direct the novice troubleshooter by way of a series of actions which will isolate the fault. Although it's inadequate for instruction competent or maybe proficient troubleshooters, this method often works with easy troubleshooting issues [10].

Learning to troubleshoot starts with the building of a conceptual design for the system which contains device knowledge, system, or domain knowledge, visual spatial information of the device or device, procedural understanding of the way to do examinations along with information gathering activities, and strategic awareness that guides search activities. Learning to troubleshoot requires a gradual change out of conceptual understanding of methods and context independent awareness of techniques to individual, context-dependent memories of problems that are similar [11].

Troubleshooters must understand the unit or system they troubleshoot at various levels of abstraction:

- purpose of the system (represented as manufacturing flow designs, system objectives);
- abstract functional model of the system (represented as causal information or structure flow topology);
- generalized functions of the system (standard processes and functions as well as management loops);
- physical features of the system (electrical, mechanical, chemical operations of the components);
- physical types in the system (physical overall look as well as anatomy, material, and form) [12].

Domain knowledge refers to the basic principles and theories in that the system or device was created. For instance, Ohm's law is a foundation concept utilized for describing the flow of electrical energy from the power supply, through the starter, and also to the spark plugs. Although it's not sufficient for learning how to be a competent troubleshooter, domain understanding is really an essential condition for starting troubleshooters. Domain information is essential when troubleshooters transfer their skills to systems that are different [13], as well as domain knowledge is needed for building deeper understanding of the system.

System or alternatively device knowledge is an understanding of the framework of the system, the performance of the elements inside the unit, so the actions of those parts as they meet up with various other components in the system. It is the understanding of every individual component's purpose in a certain system and the causal relationships in between the parts and their structure. Troubleshooting techniques based on functional knowledge of the performance of the device lead the troubleshooter to the issue more proficiently.

Performing troubleshooting work, like measuring voltage, performing tests, and pulling in observations of the functioning of various parts, involves methods which should be noted and practiced. Expertise of these activities allows troubleshooters to handle the operations for performing regular servicing methods as well as testing the parts throughout the troubleshooting procedure. Procedural knowledge is particular to the tools and the system utilized to troubleshoot it. Therefore, its application is restricted to that specific system or content [14].

Strategic knowledge plays a crucial part in troubleshooting by decreasing the issue area, isolating the prospective faults, and examining and evaluating solutions and hypotheses. Understanding what part of the system systems to test primarily when diagnosing a system which will not function is essential strategic knowledge. Strategic information helps the troubleshooters verify the hypotheses as well as answers they have generated or seek new alternatives if the existing hypotheses or answers are confirmed unfeasible or false. Methods used in the troubleshooting procedure may be classified as local strategies or global strategies. Global strategies are free from a certain domain content or system and could certainly be applied across various domains. Local strategies tend to be the ones that are only relevant to a certain content domain or system. Global strategies assist the troubleshooter lessen the issue space, although local approaches assist the troubleshooter perform the reduction operation.

There are five commonly used global strategies in the troubleshooting process [15].

Trial and Error: Randomly encounter any section of the unit where the potential fault may have occurred. This particular technique is most typical in the operation of novice troubleshooters.

Exhaustive: List all of the possible faults and evaluate them one by one until the particular fault is determined. This kind of method, much like serial elimination, is useful solely in systems that are simple.

Topographic: Isolate the fault by means of determining a number of working as well as malfunctioning checks following the traces with the product. The topographic strategy is generally implemented in 2 ways, forward or backward. The forward topographic plan begins the troubleshooting process in a place in which the device is recognized to be working normally and then works to the fault by using the system. The backward topographic strategy uses the identical process but begins at the purpose of malfunction and after that works backwards towards the input point.

Split half: Split the issue space in half and examine the functioning problem to find out where half the fault is situated. This method minimizes the issue space by confirming the faulty section. The method is repeated until the possible defective spot is reduced to an individual element. This strategy is effective when the faulty device is complicated and also the original problem space seems to contain several potential faults without any good indication of where the particular fault lies.

Functional/discrepancy detection: Isolate the fault by searching for the mismatches between what is predicted in a typical system operation as well as the

real actions exhibited. By detecting the mismatches, the troubleshooter is able to recognize the parts in which the distinction is situated plus, in turn, separate the particular fault. Doing this particular strategy calls for a comprehensive integration of system understanding (especially the interrelationship between functional awareness as well as behavioral knowledge).

2.2 General principles of Linux diagnostics

During performance searching in the command line, utilizing regular Linux tools?we ought to search for errors along with saturation metrics, as they are both simple to interpret, and then resource utilization. Linux general performance observability tools and commands for different sections of operating system are depicted in the Figure 2.1.

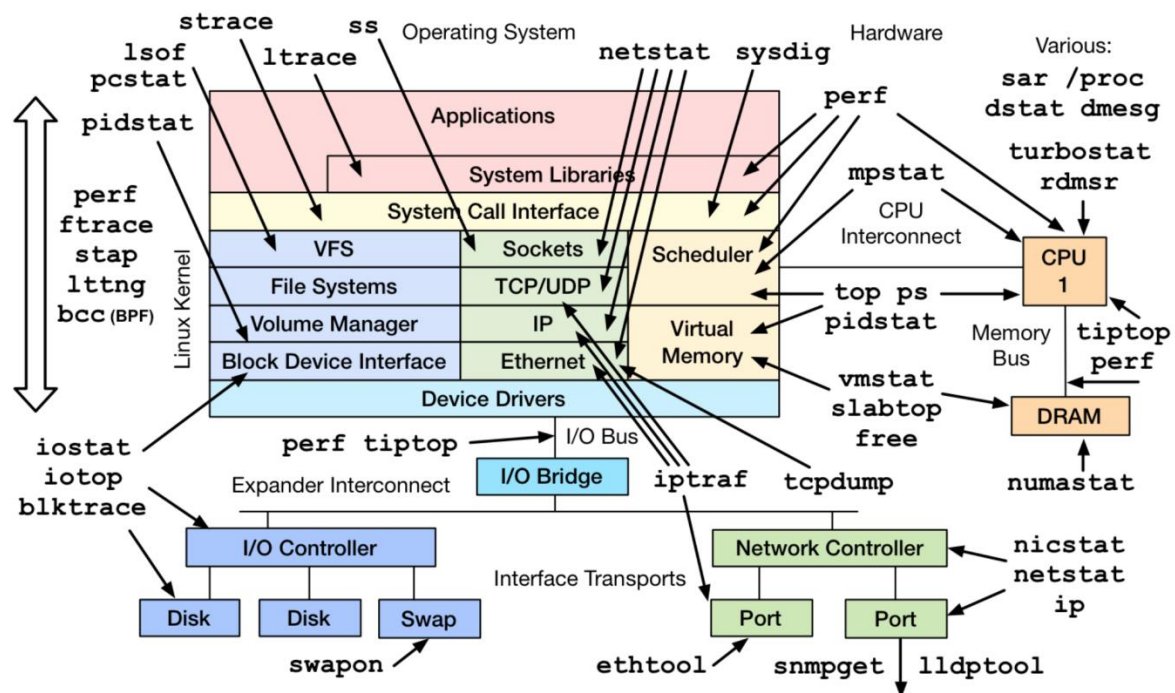


Figure 2.1 – Linux performance observability tools

Saturation is when a resource has much more load than it is able to deal with, and may be open also as the length of a request queue, or perhaps time spent waiting.

There are commands which can help completing several of the Utilization, Saturation and Errors Method: a strategy for finding performance bottlenecks. The USE Method is a technique for examining the functionality of any system. The construction of a checklist, which for server analysis can be utilized for quickly identifying errors or even resource bottlenecks are directed by it. It starts by posing questions, after which seeks answers, rather than beginning with provided metrics (partial responses) and then attempting to work in reverse. The USE Method may be summarized as: for each resource, check utilization, saturation, and also errors.

Resource is the every physical server functional elements (CPUs, busses, disks, etc.).

Utilization is the standard time that the resource was demanding servicing tasks. There is another definition in which utilization identifies the proportion of an useful resource that is used, and therefore 100 % utilization means no additional job could be acknowledged, in contrast to the "busy" description above.

Saturation is the level to that the resource has additional work that it cannot service, frequently queued.

Errors will be the count of error occasions.

The USE methodology is pictured as a flowchart below in Figure 2.2. Errors are often examined before saturation and utilization, as a small optimization since they are usually quicker and easier to interpret.

Issues that are prone to be program bottlenecks are identified by the USE Method. Regrettably, systems could be suffering much more than one performance issue, and so the very first one you discover may be an issue but not the problem. Every finding may be examined working with additional methodologies, prior to continuing the USE Method as needed to iterate over additional resources.

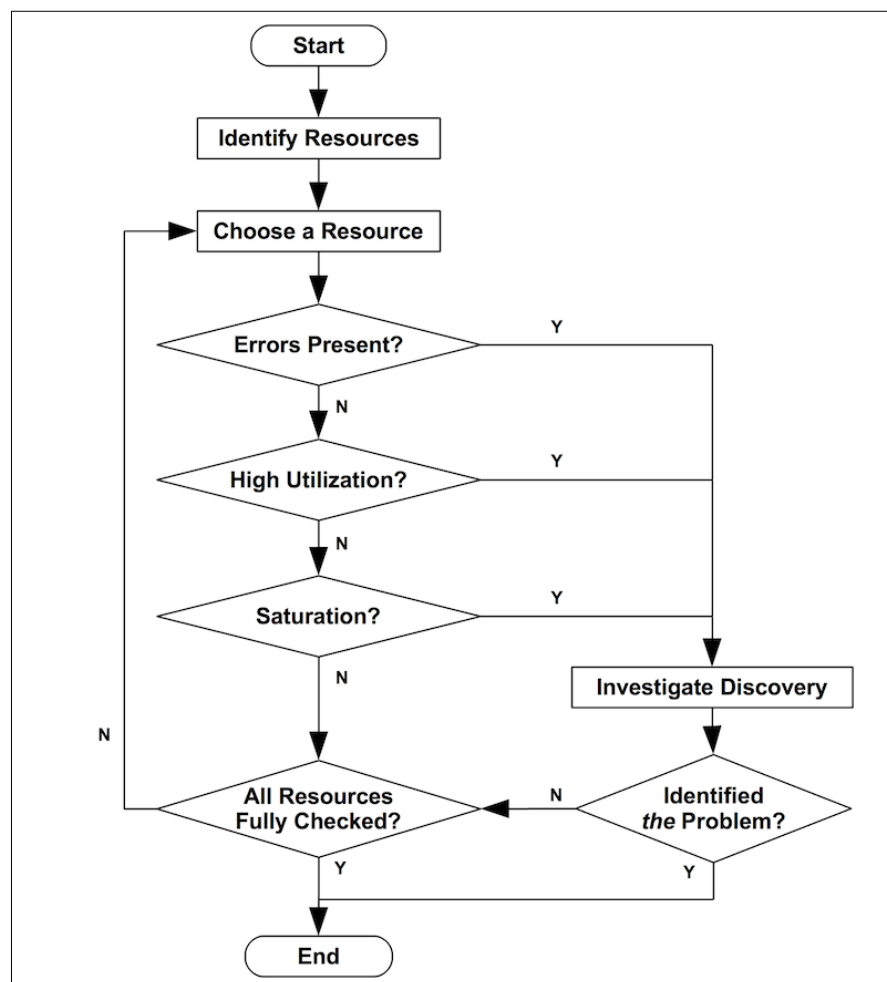


Figure 2.2 – USE Method algorithm

There are several commands in Linux which can get us a top level notion of system resource usage and running processes.

Command *uptime* is a fast way to see the load averages, and they signify the amount of tasks (processes) wanting to run, an instance of outcome is shown in Figure 2.3. On Linux systems, these figures include procedures desiring to operate on CPU, and also procedures obstructed in uninterruptible I/O (usually disk I/O). This will give a high degree thought of resource load (or perhaps demand), but may not be appropriately understood with no different instruments.

```
$ uptime
23:51:26 up 21:31, 1 user, load average: 30.02, 26.43, 19.02
```

Figure 2.3 – Output of command *uptime*

The three numbers are exponentially damped shifting sum averages with a 1 minute, 15 minute constant, and 5 minute. The three numbers provide us some idea of exactly how load is changing as time passes. For instance, in case you have been asked to examine an issue server, and the 1 minute value is significantly less than the 15 minute value, then you can try to have logged in too late and missed the problem.

In the illustration earlier, the load averages clearly show the latest increase, hitting 30 for the 1 minute value, in comparison to 19 for the 15 minute valuation.

Command *dmesg* / *tail* sights the very last 10 system messages, in case there are any. Right here we can search for errors which can cause performance issues. In the Figure 2.4 we are able to see the oom-killer, then TCP dropping a request.

```
$ dmesg | tail
[1880957.563150] perl invoked oom-killer: gfp_mask=0x280da, order=0,
oom_score_adj=0
[...]
[1880957.563400] Out of memory: Kill process 18694 (perl) score 246
or sacrifice child
[1880957.563408] Killed process 18694 (perl) total-vm:1972392kB,
anon-rss:1953348kB, file-rss:0kB
[2320864.954447] TCP: Possible SYN flooding on port 7001. Dropping
request. Check SNMP counters.
```

Figure 2.4 – Output of command *dmesg* | *tail*

Vmstat 1 which stands for virtual memory stat, *vmstat* is a widely available tool. A summary of key server data on each line (Figure 2.5) is printed by it.

Vmstat was operated with an argument of 1, to print one second summaries. The very first line of output has several columns that show the average since boot, rather than the preceding second.

Column *r* is the amount of procedures operating on CPU and awaiting a turn. This offers a much better signal than load averages for determining CPU saturation, as it does not include I/O. To interpret: an *r* worth greater compared to the CPU count is saturation.

```
$ vmstat 1
procs -----memory----- ---swap-- -----io----- -system-- ----
--cpu-----
 r  b swpd   free   buff  cache   si   so    bi    bo    in    cs us
sy id wa st
34  0    0 200889792  73708 591828    0    0    0    5    6   10
96  1  3  0  0
32  0    0 200889920  73708 591860    0    0    0   592 13284 4282
98  1  1  0  0
32  0    0 200890112  73708 591860    0    0    0    0 9501 2154
99  1  0  0  0
32  0    0 200889568  73712 591856    0    0    0   48 11900 2459
99  0  0  0  0
32  0    0 200890208  73712 591860    0    0    0    0 15898 4840
98  1  1  0  0
^C
```

Figure 2.5 – Output of command `vmstat 1`

- *free* is the free memory measured in kilobytes. If there are way too many digits to count, there is plenty of free memory. The `free -m` command, included as command 7, much better describes the state of complimentary memory;

- *si* and *so* stand for swap-ins and swap-outs. If these are non-zero, the machine is out of memory;

- *us*, *sy*, *id*, *wa*, *st* are breakdowns of CPU time, on common throughout all CPUs. They are user time, system time (kernel), idle, wait I/O, and stolen time.

The CPU time breakdowns are going to confirm in case the CPUs are demanding, by adding user along with system time. A continuous level of wait I/O points to a disk bottleneck; this is exactly where the CPUs are nonproductive, as jobs are hindered awaiting pending disk I/O. You are able to handle wait I/O as an additional type of CPU idle, one that provides a clue why they are idle.

System time is needed for I/O processing. A top system time average, more than 20 %, could be exciting to check out further: possibly the kernel is processing the I/O inefficiently.

In the above mentioned example, CPU time is nearly completely in user level, pointing to application level usage instead. The CPUs can also be effectively more than 90 % used on average. This is not always a problem; examination for the level of saturation using the *r* column.

Command `mpstat -P All 1` prints CPU time breakdowns per CPU, which may be utilized to check out for an imbalance. An individual hot CPU might be evidence

of a single threaded application. A good example of output outcome is displayed in Figure 2.6 underneath.

```
$ mpstat -P ALL 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015 _x86_64_
(32 CPU)

07:38:49 PM CPU %usr %nice %sys %iowait %irq %soft %steal
%guest %gnice %idle
07:38:50 PM all 98.47 0.00 0.75 0.00 0.00 0.00 0.00
0.00 0.00 0.78
07:38:50 PM 0 96.04 0.00 2.97 0.00 0.00 0.00 0.00
0.00 0.00 0.99
07:38:50 PM 1 97.00 0.00 1.00 0.00 0.00 0.00 0.00
0.00 0.00 2.00
07:38:50 PM 2 98.00 0.00 1.00 0.00 0.00 0.00 0.00
0.00 0.00 1.00
07:38:50 PM 3 96.97 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 3.03
[...]
```

Figure 2.6 – Output of command mpstat -P ALL 1

Command iostat -xz 1 is a good tool for understanding block units (disks), both workload utilized as well as the resultant performance, report of that is depicted in Figure 2.7. Kernel I/O data on terminal, CPU operations and device is displayed by iostat. The very first data which are printed are averaged over the system uptime. In order to obtain information about the present task, the right wait time must be specified, so the consequent sets of printed stats will likely be averaged over that period.

```
$ iostat -xz 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015 _x86_64_
(32 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           73.96    0.00    3.73    0.03    0.06   22.21

Device:            rrqm/s   wrqm/s     r/s     w/s    kB/s    kB/s avgrq-sz
avgqu-sz   await  r_await  w_await  svctm  %util
xvda          0.00     0.23    0.21    0.18     4.52     2.08    34.37
0.00     9.98   13.80    5.42    2.44     0.09
xvdb          0.01     0.00    1.02    8.94   127.97   598.53   145.79
0.00     0.43    1.78    0.28    0.25     0.25
xvdc          0.01     0.00    1.02    8.86   127.79   595.94   146.50
0.00     0.45    1.82    0.30    0.27     0.26
dm-0          0.00     0.00    0.69    2.32    10.47    31.69    28.01
0.01     3.23    0.71    3.98    0.13     0.04
dm-1          0.00     0.00    0.00    0.94     0.01     3.78     8.00
0.33   345.84    0.04   346.81    0.01     0.00
dm-2          0.00     0.00    0.09    0.07     1.35     0.36   22.50
0.00     2.55    0.23    5.62    1.78     0.03
[...]
```

Figure 2.7 – Output of command `iostat -xz 1`

- *r/s*, *w/s*, *rkB/s*, *wkB/s* are classified as the delivered reads, writes, read Kbytes, and also write Kbytes per second for the device. They are utilized for workload characterization. A performance issue might merely be due to an excessive load used;

- Column *await* shows the typical time for the I/O using milliseconds. This is the time that the application program suffers, as it provides each time queued and time getting serviced. Larger than expected common times can be a signal of device saturation, or perhaps device problems;

- Column *avgqu-sz* reveals the average quantity of requests issued on the device. Values in excess of 1 might be proof of saturation, though equipment may generally run on requests in parallel, particularly virtual devices which forward several back-end disks;

- *%util* signifies unit utilization. This is genuinely a busy percent, displaying the time each second the unit was doing work. Values greater than 60 % usually lead to poor performance that ought to be observed in *await*, though it is determined by the device. Values close to 100 % generally indicate saturation.

Whenever the storage unit is a logical disk device fronting numerous back-end disks, consequently 100 % utilization could suggest that many I/O has been processed 100 % of the time, nonetheless, the back-end disks might be far from saturated, and also might be ready to deal with even more work.

Using command `sar` you are able to monitor performance of different Linux subsystems in real time. Tool `sar -n DEV 1` is used to to check out network interface throughput: *txkB/s* and *rxkB/s*, as a measure of workload, and to check in case any limit have been reached. In the Figure 2.8, `eth0` get is reaching 22 Mbytes/s, and that is 176 Mbits/sec.

```
$ sar -n DEV 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015
_x86_64_ (32 CPU)

12:16:48 AM      IFACE  rxpck/s  txpck/s    rxkB/s    txkB/s
rxcmp/s  txcmp/s  rxmcs/s  %ifutil
12:16:49 AM      eth0  18763.00  5032.00  20686.42    478.30
0.00      0.00      0.00      0.00
12:16:49 AM        lo    14.00    14.00      1.36      1.36
0.00      0.00      0.00      0.00
12:16:49 AM  docker0      0.00      0.00      0.00      0.00
0.00      0.00      0.00      0.00

12:16:49 AM      IFACE  rxpck/s  txpck/s    rxkB/s    txkB/s
rxcmp/s  txcmp/s  rxmcs/s  %ifutil
12:16:50 AM      eth0  19763.00  5101.00  21999.10    482.56
0.00      0.00      0.00      0.00
12:16:50 AM        lo    20.00    20.00      3.25      3.25
0.00      0.00      0.00      0.00
12:16:50 AM  docker0      0.00      0.00      0.00      0.00
0.00      0.00      0.00      0.00
^C
```


Figure 2.8 – Output of command `sar -n DEV 1`

`Sar -n TCP,ETCP 1` stands for a summarized perspective of a few major TCP metrics, example is shown in Figure 2.9. These include:

- *active/s* which happens to be the amount of locally-initiated TCP connections every second (e.g., via `connect()`);
- *passive/s* is the amount of remotely-initiated TCP connections per second (e.g., by `accept()`);
- The amount of TCP retransmits a second is represented by *retrans/s*.

```
$ sar -n TCP,ETCP 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015
_x86_64_ (32 CPU)

12:17:19 AM active/s passive/s iseg/s oseg/s
12:17:20 AM 1.00 0.00 10233.00 18846.00

12:17:19 AM atmptf/s estres/s retrans/s isegerr/s orsts/s
12:17:20 AM 0.00 0.00 0.00 0.00 0.00

12:17:20 AM active/s passive/s iseg/s oseg/s
12:17:21 AM 1.00 0.00 8359.00 6039.00

12:17:20 AM atmptf/s estres/s retrans/s isegerr/s orsts/s
12:17:21 AM 0.00 0.00 0.00 0.00 0.00
^C
```

Figure 2.9 – Output of command `sar -n TCP,ETCP 1`

The active and passive counts are usually beneficial as an approximate way of measuring server load: quantity of new established connections (passive), and quantity of downstream connections (active). It may assist thinking of active as outbound, along with passive as inbound, but this is not purely accurate (e.g., contemplate getting a localhost to localhost connection).

Retransmits are a signal of a network or server issue; it might be an unreliable network (e.g., the public Internet), or perhaps it might be due a server getting overloaded as well as dropping packets. The example above shows only one new TCP connection per second.

The `top` command includes many of the metrics checked previously in this section are included in the command `top`. It might be convenient to run it to find out in case something looks extremely distinct from the earlier commands, which could indicate that load is adjustable. Representation of command `top` is shown in Figure 2.10 beneath.

A drawback to `top` is it is more difficult to discover patterns over time, which might be more clear in tools as `pidstat` and `vmstat`, which offer rolling output.

```

$ top
top - 00:15:40 up 21:56,  1 user,  load average: 31.09, 29.87, 29.92
Tasks: 871 total,   1 running, 868 sleeping,   0 stopped,   2 zombie
%Cpu(s): 96.8 us,   0.4 sy,   0.0 ni,  2.7 id,   0.1 wa,   0.0 hi,   0.0
si,   0.0 st
KiB Mem:  25190241+total, 24921688 used, 22698073+free,    60448
buffers
KiB Swap:          0 total,          0 used,          0 free.  554208
cached Mem

   PID USER      PR  NI   VIRT   RES   SHR S  %CPU  %MEM    TIME+
COMMAND
 20248 root        20   0  0.227t 0.012t 18748 S   3090   5.2   29812:58
java
  4213 root        20   0 2722544 64640 44232 S   23.5   0.0   233:35.37
mesos-slave
 66128 titanc1+   20   0   24344  2332  1172 R    1.0   0.0    0:00.07
top
  5235 root        20   0 38.227g 547004 49996 S    0.7   0.2    2:02.74
java
  4299 root        20   0 20.015g 2.682g 16836 S    0.3   1.1   33:14.42
java
    1 root        20   0   33620  2920  1496 S    0.0   0.0    0:03.82
init
    2 root        20   0        0      0      0 S    0.0   0.0    0:00.02
kthreadd

```

Figure 2.10 – Output of command top

Proof of intermittent problems may additionally be lost if the result is not paused fast enough (Ctrl-S to pause, Ctrl-Q to continue), so the screen clears.

2.3 Proposed sequences of laboratory works

In learning directions, we defined our focus on three aspects: network virtualization, storage virtualization and also compute virtualization (Figure 2.11). All aspects are derived from knowing the fundamentals of Linux and ways to read, operate as well as navigate within command line. Networking part is dependent on Open vSwitch, an open-source virtual switch. Open vSwitch was developed by the group at Nicira, which was eventually acquired by VMware. OVS was intended to meet up with the requirements of the open source community, since there was no a feature rich virtual switch providing created for Linux-based hypervisors, for example Kernel based Virtual Machine (KVM). Hypervisors are utilized for simplifying regulation of virtual machines. These software solutions enable to swiftly run, stop as well as open new virtual devices within one host. KVM is the tool which will be utilized for compute virtualization part. KVM is software that

enables to implement computer-based virtualization in OS Linux and Linux-Like systems. For third aspect we are planning to work with Local Volume Management or LVM, and internet Small Computer system Interface or iSCSI. LVM is a storage device management technology which gives users the capability to pool and abstract the physical format of component storage units for flexible and easier administration.

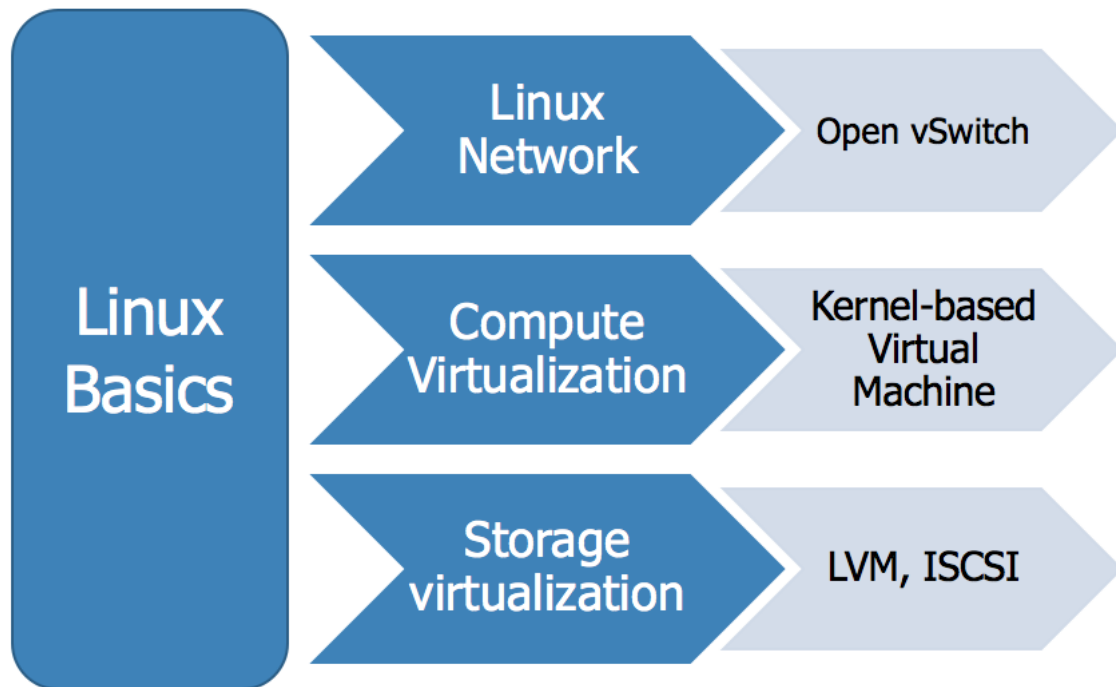


Figure 2.11 – Proposed sequences of laboratory works

2.3.1 Open vSwitch overview. Open vSwitch is a multilayer software switch licensed under the open source Apache 2 license.

Open vSwitch is well suited to run like a virtual switch in VM environments. Along with exposing standard control as well as visibility interfaces on the virtual network layer, it was created to help distribution throughout many actual physical servers. Open vSwitch supports a number of Linux-based virtualization technologies like Xen/XenServer, VirtualBox, and KVM.

The majority of the code is developed in platform independent C and it is conveniently ported to various other environments. The present release of Open vSwitch supports the following features:

- Standard 802.1Q VLAN model with trunk and also access ports;
- NIC bonding with or with no LACP on upstream switch;
- NetFlow, sFlow(R), and also mirroring for improved visibility;
- QoS (Quality of Service) setup, in addition policing;
- Geneve, GRE, VXLAN, STT, as well as LISP tunneling;
- 802.1ag connectivity fault management;
- OpenFlow 1.0 in addition countless extensions;
- Transactional configuration database with Python and C bindings;

- High-performance forwarding utilizing a Linux kernel component.

The capability to bridge traffic between VMs along with the external planet is needed by hypervisors. On Linux-based hypervisors, this utilized to mean utilizing the integrated L2 switch, and that is quick and reliable. But Open vSwitch is focused at multi-server virtualization deployments, a landscape that the earlier stack is not well suited. These locations are usually characterized by extremely powerful end-points, the servicing of logical abstractions, and also (sometimes) integration with or offloading to unique objective switching hardware.

Most network state related to a network entity (say a virtual machine) must be migratable and identifiable easily between several hosts. This might include conventional "soft state" (such as an entry in an L2 learning table), L3 forwarding declare, policy routing phase, QoS policy, ACLs, checking configuration (e.g. NetFlow, sFlow), IPFIX, etc. Open vSwitch has support for each migrating and configuring both slow (configuration) as well as quick network state among instances. For instance, in case a VM migrates between end hosts, it is possible not only to migrate connected setup (ACLs, QoS) but every live network state (including, for instance, present state which might be tough to reconstruct). Additionally, Open vSwitch express is typed as well as backed by a genuine data-model allowing for the improvement of organized automation systems.

Virtual environments are usually characterized by high-rates of modification. VMs coming and going, VMs moving forwards and backwards in time, alterations on the logical network environments, and so forth. Open vSwitch supports a selection of attributes that allow a network management system to respond as well as adapt when the environment changes. This consists of basic accounting as well as visibility assistance like NetFlow, IPFIX, and sFlow. But maybe far more helpful, a network state database (OVSDb) which supports remote triggers is supported by Open vSwitch. Thus, a portion of orchestration program is able to "watch" different facets of the network and react if/when they change. This is used predominantly nowadays, for instance, to react to and track VM migrations. Open vSwitch additionally supports OpenFlow as a technique of exporting remote entry to control traffic. There are selection of uses for this such as worldwide network finding via assessment of finding or link-state traffic.

Distributed virtual switches (such as VMware vDS as well as Cisco's Nexus 1000V) frequently keep logical context within the system via appending or manipulating tags in network packets. This may be utilized to exclusively determine a VM (in a fashion resistant to hardware spoofing), or to support another context which is just appropriate in the logical domain. A lot of the issue of creating a distributed virtual switch is usually to efficiently and correctly organize these tags.

Numerous techniques for indicating as well as maintaining tagging rules, every one of that are available to a remote practice for orchestration are included by Open vSwitch. Additionally, in most instances these tagging regulations are kept in an optimized form so that they do not have to be merged with a heavyweight network device. This allows, for instance, a huge number of tagging or address remapping rules being configured, altered, and migrated.

In an equivalent vein, Open vSwitch supports a GRE implementation which could deal with a huge number of simultaneous GRE tunnels and also supports remote setup for tunnel development, configuration, and tear-down. This, for instance, may be utilized to link private VM networks in various data centers.

Open vSwitch's forwarding route (the in-kernel datapath) is created to be amenable to offload packet inspection to hardware chipsets, whether housed in a traditional hardware switch chassis or in an end-host NIC. This allows for any Open vSwitch management path to have the ability to both regulate a clean software implementation or even a hardware switch.

The benefit of hardware integration is not only performance inside virtualized environments. If physical switches additionally expose the Open vSwitch management abstractions, both virtualized and bare-metal hosting environments may be managed using the identical mechanism for automated network control.

In many ways, Open vSwitch targets an alternative thing in the design space compared to last hypervisor networking stacks, concentrating on the demand for automated and dynamic community influence in large-scale Linux-based virtualization environments [16].

2.3.2 Isolation virtual machine traffic using VLANs with Open vSwitch. An example comprises of two physical network and assumes the environment is set up as outlined below in Figure 2.12.

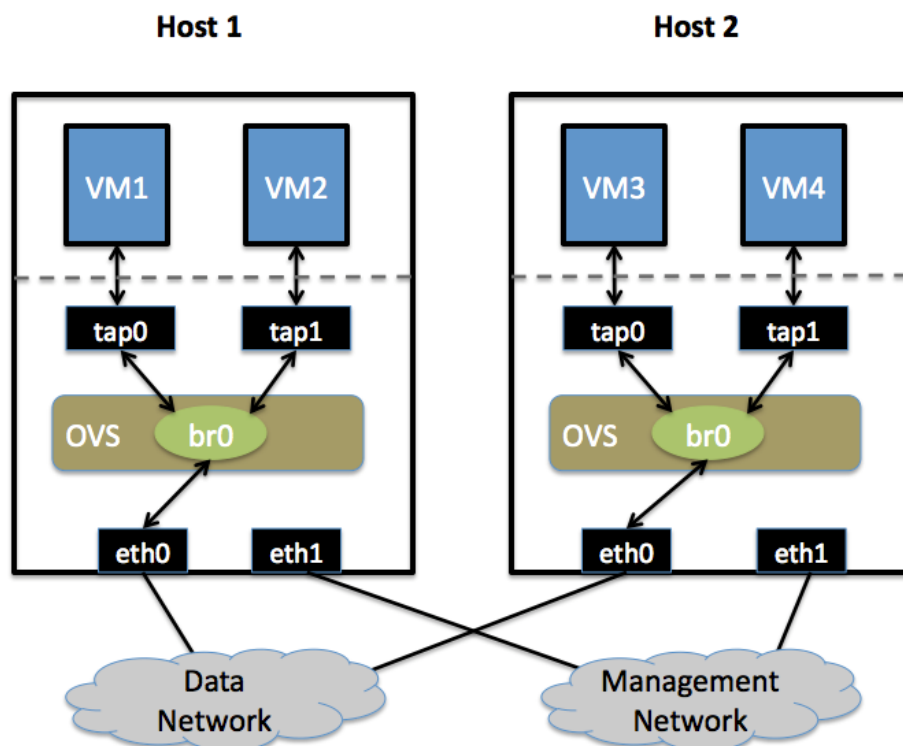


Figure 2.12 – Network description for VLANs

Data network is Ethernet network for VM information traffic, which could bring VLAN-tagged traffic among VMs. Physical switch should be able to forwarding VLAN-tagged traffic plus the physical switch ports must work as VLAN trunks. Management network is not absolutely necessary, though it is a very simple method to make the physical host an IP address for remote accessibility, since an IP address cannot be given straight to eth0.

The usage of 2 hosts: host1 and host2 are assumed by the environment. Open vSwitch is being run by both hosts. Each host has 2 NICs, eth1 and eth0, that are configured as follows:

- eth0 is linked to the Data Network. No IP address is given to eth0;
- eth1 is linked to the Management Network (if necessary). eth1 has an IP address which is utilized to attain the physical host for control.

Every host is going to run two virtual machines (Vm2 and vms). vm1 are working on host1, while vm3 as well as vm4 are working on host2. Each VM features one interface which shows up like a Linux device (e.g., tap0) on the physical host.

Configuration of host1, beginning with formation of Open vSwitch bridge:

```
$ ovs-vsctl add-br br0
```

Inclusion of eth0 to the bridge:

```
$ ovs-vsctl add-port br0 eth0
```

Initially, most OVS ports are VLAN trunks, therefore eth0 is going to pass each VLANs. Whenever we include eth0 to the OVS bridge, any IP addresses that might have been given to eth0 cease working. IP address given to eth0 ought to be migrated to an alternative interface before adding eth0 on to the OVS bridge. This is the reason behind the distinct management connection by eth1..

Following step will be the inclusion vm1 as an "access port" on VLAN 100. What this means is that traffic coming into OVS from VM1 is going to be untagged and also considered part of VLAN 100. Traffic from VM2 will be untagged and also considered a component of VLAN 200, consequently:

```
$ ovs-vsctl add-port br0 tap0 tag=100
```

```
$ ovs-vsctl add-port br0 tap1 tag=200
```

The exact same actions are going to be repeated for host2, establishing a bridge with eth0 as being a VLAN trunk:

```
$ ovs-vsctl add-br br0
```

```
$ ovs-vsctl add-port br0 eth0
```

Addition of VM3 to VLAN100 and VM4 to VLAN200:

```
$ ovs-vsctl add-port br0 tap0 tag=100
```

```
$ ovs-vsctl add-port br0 tap1 tag=200
```

As an outcome, pings from vm1 to vm3 must be successful, as these two VMs are on the identical VLAN.

Pings from vm2 to vm4 will typically be successful, because these VMs will also be on exactly the same VLAN as one another.

Pings from vm1/vm3 to vm2/vm4 should not do well, as these VMs are on separate VLANs. If we have a router set up to forward between the VLANs, now

pings will work, but packets showing up at vm3 must have the source MAC address of the router, not of vm1.

2.3.3 The KVM hypervisor. KVM - the Linux Virtual Machine Monitor - is a kernel extension, after loading, which, spins the Linux kernel into a virtual machine monitor or hypervisor.

Presently, KVM interacts with the kernel through a loadable kernel component. A number of guest operating systems are supported, like Linux, Haiku, Windows, Solaris, BSD, ReactOS as well as AROS Research Operating system [17].

KVM does not accomplish any self-emulation; as an alternative, the application running in operator space employs the /dev/kvm interface to configure the address space of the guest virtual server, takes the simulated I/O resources of its as well as displays its image on the host image, other equipment like NIC (network interface card) controller and also disk controller. KVM architecture is demonstrated in Figure 2.13.

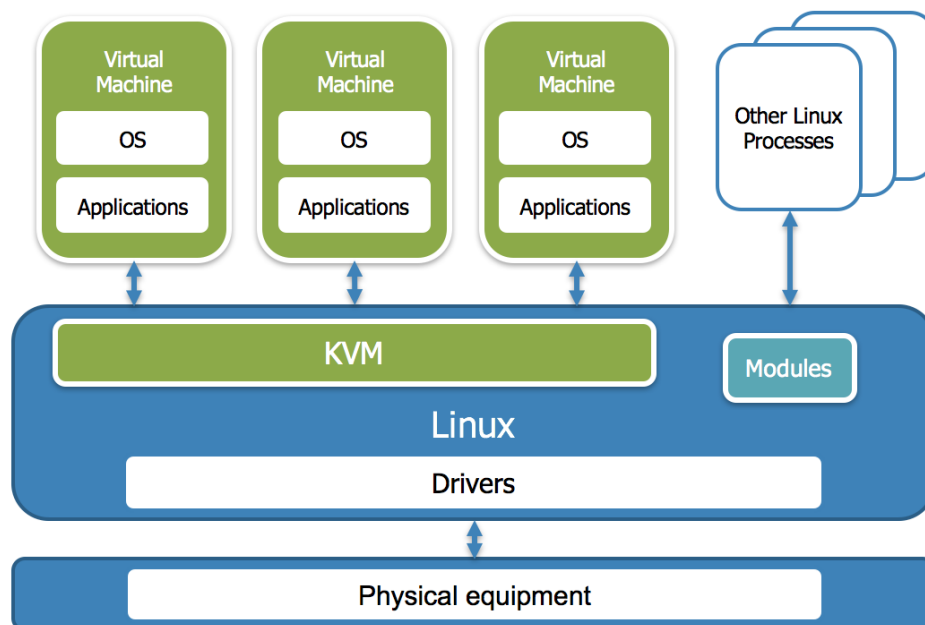


Figure 2.13 – KVM architecture

In the KVM structure, the virtual machine operates as an ordinary Linux procedure designed by the conventional Linux scheduler. In reality, each virtual processor is found as a typical Linux process. This enables KVM to work with all of the functions of the Linux kernel.

The emulated devices are managed by an altered version of qemu that delivers emulation of the BIOS, PCI bus, USB bus, along with a regular range of devices, like SCSI and IDE disk controllers, network cards, and others.

KVM inherits effective memory management capabilities from Linux. The virtual machine 's memory is kept in the exact same manner as the memory of every other Linux procedure, and may be replaced, copied with big pages to enhance

performance, aggregated or even saved in a file on disk. Assistance for NUMA technology (Non-Uniform Memory Access, a memory architecture for multiprocessor systems) allows for virtual machines to effectively access substantial memory.

KVM supports the most recent memory virtualization features from processor making companies, particularly, Intel Extended Page Table (EPT) as well as AMD Rapid Virtualization Indexing (RVI), to reduce CPU use and achieve higher throughput.

A generalization of memory pages is backed by the Kernel Same-page Merging (KSM) kernel functionality. KSM scans the memory of every virtual machine, and in case a number of pages of virtual machine memory are the same, it merges them in one page which becomes shared between these virtual machines and it is saved in an individual copy. In case the guest system attempts to change this shared page, it is offered with its own copy.

KVM supports dynamic migration, supplying the capability to move running virtual machines between actual physical nodes without interrupting upkeep. Dynamic migration is transparent to users: the virtual machine remains on, the network connections are working, so the end user programs keep on working, while the virtual machine moves on the brand new physical server.

Along with dynamic migration, KVM supports preserving a copy of the present state of the virtual machine to disk, enabling you to keep it and restore it later.

Every unit in KVM is a Linux-based procedure, therefore it conforms to all regular security policies and it is isolated from other tasks. Specific add-ons also include additional security elements, like access control, encryption, and so forth.

2.3.4 Introduction to LVM concepts. LVM, or Logical Volume Management, is a storage device management technological approach which gives users the ability to pool and abstract the physical format of component storage equipment for flexible and easier administration. Using the device mapper Linux kernel framework could be utilized to gather existing storage units into groups and allocate logical units from the combined space as needed.

The primary advantages of LVM are elevated abstraction, control, and flexibility. Logical volumes are able to have meaningful names like "root-backup" or "databases". Volumes could be resized dynamically as space needs change and migrated between actual physical devices inside the pool on a running system as well as exported conveniently. LVM also provides advanced features as snapshotting, mirroring, and striping.

LVM functions by layering abstractions on above of physical storage equipment. The standard layers which LVM uses, beginning with the most primitive, are physical volumes, volume groups and logical volumes.

Physical block equipment or any other disk-like devices are utilized by LVM as the raw construction material for higher degrees of abstraction. Physical volumes

are routine storage devices. A header is written by LVM on the device to allot it for control.

Volume groups are the physical volumes fused into storage pools. Volume groups abstract the qualities of the underlying equipment as well as perform as a single logical device with merged storage capacity of the component physical volumes.

A volume group could be sliced up into several logical volumes. Logical volumes are functionally equivalent to partitions on an actual disk, although with a lot more versatility. Logical volumes are the main component that applications and users will meet up with.

Each and every volume inside a volume group is segmented into minor, extents were called by fixed-size chunks. The scale of the extents is resolved by the volume group. All volumes inside the group conform to the identical extent size.

The extents on a physical volume are known as physical extents, even though the extents associated with a logical volume belong in the logical extents. A logical volume is basically a mapping which LVM maintains between physical and logical extents. Due to this particular relationship, the extent size belongs to probably the smallest level of space which could be issued by LVM.

Extents are behind a lot of the versatility as well as power of LVM. The logical extents which are presented as a single unit by LVM do not have to map to constant physical extents. LVM is able to duplicate and then reorganize the physical extents which compose a logical volume with no interruption to users. Logical volumes may as well be easily expanded or shrunk by merely adding extents to or taking away extents from the volume.

To sum up, LVM could be utilized combining physical volumes into volume groups to unify the storage area located on a system. Afterwards, administrators can part the volume group into arbitrary logical volumes, which serve as adaptable partitions

2.3.5 Internet small computer system interface. iSCSI is an acronym which stands for internet Small Computer system Interface. It is a storage area networking protocol used to send obstruct storage space coming from storage arrays or devices to client computers which are not directly linked to those equipment [18].

An iSCSI storage area network consists of iSCSI goals on storage array controllers and iSCSI initiators on storage clients. These goals as well as initiators are put into use by the iSCSI protocol to link storage to clients and are represented by a distinctive label known as the iSCSI Qualified Name or Iqn.

Computer operating systems include or may be installed with iSCSI clients as well as initiators. While a few storage arrays are supposed to just supply storage above iSCSI, lots of storage units come built with iSCSI targets along with many other popular protocols.

Unlike fiber channel, iSCSI does not work over a distinct network protocol. It transmits data with the TCP/IP protocol more than well known Ethernet networks which are supporting network traffic in the data center at the moment.

Little or no investment in new network technological innovation is necessary to begin utilizing an iSCSI Storage Area Network technology. However, a number of deployments of iSCSI are put in place with networks or subnets which are focused for iSCSI traffic simply to take full advantage of the bandwidth obtainable for storage space. It is additionally common for an iSCSI network to be deployed on fast Ethernet like 10 Gigabit or even better because of this.

A Host Bus Adapter or basic Ethernet port will be actually necessary and sufficient to link iSCSI targets and also initiators to a system. Specific ports are generally set up on computer servers that will offload iSCSI transactions coming from the CPU on the adapter for higher effectiveness or that support numerous Ethernet storage protocols for example simultaneously iSCSI and Fiber Channel over Ethernet.

A number of high-speed network switches incorporate a technology known as Data Center Bridging, and that helps make it simpler to help storage traffic and standard network traffic on the identical switch ports. This is accomplished by producing rules for quality of service. These rules define the amount of bandwidth each kind of network traffic is allowed to ingest.

To be able to provision iSCSI storage an iSCSI client will generally have to be set up or activated on the client computer system. The technique to accomplish this differs by operating system but is typically pretty easy. iSCSI client initiators are then added to initiator groups for iSCSI client definitions on the storage unit. The name because of this grouping of initiators might differ from seller to vendor but the purpose of its is the same. These initiated organizations are utilized to find which customers might visit storage targets.

An iSCSI Logical Unit Number is generated on the storage device and given to an initiator group or client characterization only at that point assuming the goal along with initiator are on a single IP network, the client might have the ability to immediately identify the target. After the initiator is attached to the target, the Logical Unit Number at that target IQN can be obtained to be used by the client.

iSCSI LUNs are configured as well as used the just like every other block storage by the prospect operating system. The iSCSI protocol supports numerous features to improve performance and security.

2.4 Mathematical model for troubleshooting

The mathematical part of this paper is presenting a way for automating diagnosis and troubleshooting tasks in networks applying Bayesian Networks (BN). BN is an expertise primarily based technique, generally called a Bayesian cognitive method, which learns the way to create a model from information. The information consists of the collective troubleshooting expertise, specifically the list of observed alarms and symptoms with the corresponding faults which have been placed by the network specialist into a database. The BN utilizes the data to develop a statistics based mathematical model which relates symptoms to reasons of faults. The level of the model improves with the quantity of information in the database. After the model has become produced, a new issue could be addressed by feeding

observations of symptoms and alarms to the BN that in turns will calculate probabilities for those faults. This technique can be viewed as statistical clustering.

Denote by C_i a cause for poor performance of the network; by S_j a symptom that could lead to evaluating the system quality; and by E a set of N symptoms S_j . The diagnosis consists of determining the triggers with the highest probabilities provided a set of symptoms. This process can be viewed as a classification activity in which each category corresponds to a certain cause. Utilizing Bayes' rule, one may estimate the probability for the cause C_i to happen considering the set of noticed symptoms E :

$$P(C_i/E) = \frac{P(C_i)P(E|C_i)}{P(E)}, \quad (2.1)$$

where $P(C_i/E)$ – posterior or conditional density function;

$P(C_i)$ – prior density function;

$P(E/C_i)$ – likelihood function.

For a fixed C_i , $P(E/C_i)$ is considered additionally as distribution. Usually, particular conditions on the diagnosed network are acknowledged a priori, like the sort of services provided or any kind of additional a priori knowledge. We denote the set of conditions by D . Introducing the set D to (2.1) gives

$$P(C_i/E, D) = \frac{P(C_i|D)P(E|C_i)}{P(E)}. \quad (2.2)$$

It is assumed in (2.2) that the signs and symptoms are free from the conditions. When this assumption is not verified, the circumstances must be put into the likelihood distribution as well as towards the $P(E)$ term.

Calculating the joint likelihood distribution $P(E/C_i)$ is impractical and difficult. Thus it is necessary to make use of the Naïve Bayesian Network, and that helps make the assumption that symptoms because of the causes are self-reliant. With this assumption, one simply has to establish the likelihood distribution of every symptom, $P(S_j|C_i)$, individually. This particular assumption is an approximation; however the Naïve Bayesian classifier stays effective despite having substantial dependencies between the symptoms [19]. Hence, the distinction in the investigation process will be conducted utilizing the approximation:

$$P(C_i/E, D) = \frac{P(C_i|D) \prod_{j=1}^N P(S_j|C_i)}{P(E)}. \quad (2.3)$$

The model is going to be evaluated in a virtual network environment. In each learning as well as exploitation phases, the sources for dysfunctions (faults) are launched, thus the previous density of the faults, $P(C_i|D)$, is determined by how frequently the faults are placed into the simulator. The remaining probabilities

found in (2.3) are discovered from the witnessed values estimated by the simulator when faults are introduced.

Firstly, it is required to define the causes of errors that could be found as well as associated symptoms while developing a troubleshooting model. Causes can be hardware problems as well as bad parameter values. Symptoms are the circumstances or quality indicators when some operation incline from ordinary behavior. Packet loss and delay could be classified as symptoms.

Bayesian model requires the following elements:

- List of causes;
- List of symptoms;
- List of statistical relations among the fault and the quality indicators.

For the networking aspect of the labs symptoms or anomalies could be lost connection between nodes due to connectivity problems or hardware configuration mistakes. Channel problems can be caused by babbling node, a situation where two hosts are communicating unnecessarily, like a small packet that is sent by a node in an infinite loop to check on some information, which leads to busy link. Switching loop can cause broadcast storm which is considered as the cases where broadcast packets are greatly utilized to the point of disabling the network. In case network system is overloaded, its performances degrade, consequently, the delay as well as packet loss grow and the channel is saturated. Poor network management and setup could also cause any of problems such as packet loss, delay, lost connection between nodes or network disability.

Causes of symptoms for network aspect are connectivity failure, transmission failure, channel failure, broadcast storm, babbling node, switching loop, poor network management.

For the following causes the symptoms are: network overload, packet loss, delay, busy link, link absence and so on.

With the next aspect involving usage of KVM there are many causes and symptoms. For example, when we do not see any virtual machines we must verify KVM kernel modules and make an input if they are not presented; or it could be turned off virtualization extensions in the BIOS setup. In case there are several VM set up on a device and after reboot they are vanished, it is needed to check relevant XML file and make configurations in it. Having several VMs and not being able to find each other for them is caused by flaws in network management. For sequence of compute virtualization aspect we can define several causes:

- Settings that were set up by default;
- Mistakes in network management;
- Functions not enabled in configuration files.

With these causes, the symptoms are unidentifiable VMs, loss of VM after halt or restart, unable to find specific interface, etc.

In sequence of storage virtualization, we can identify performance degradation of logical volumes and cause of this is the hot spot that experiences most of the disk I/O. If we are not able to create storage with LVM type it is needed to check on the physical storage if it is able to work or also if it exists. Or logical

volume is not working correctly due to a change of physical volume. Causes of failures are disk I/O overload, unmanaged physical storages due to its substitution.

To simulate error flows, it is necessary to have system that fully correspond to realities of life. Errors are happening mostly one at a time, failure of two or more elements of the object at a time is very small, and it can be neglected, that is characterized by ordinary flow. The probability of subsequent error of the object at any time is independent of previous failures - the flow of errors without aftereffects. And flow of errors should be stationary, thus these features are presented in simple flows that could be simulated in GPSS World.

In simulator we considered system having three types of objects to look for. They are virtual switches, servers and client computers. The objects enter the system for maintenance with the intensity λ . The flow of objects entering the service is Poisson flow, so $\lambda = 0.1$ (1/time unit). The objects are served in the service channel with the intensity μ . The distribution of service time for the objects is exponential: $\mu = 0.05$ (1/time unit). At the same time, all objects are affected by Poisson failure flows with the corresponding intensities: $\lambda_1 = 0.01$ (1/time unit), $\lambda_2 = 0.008$ (1/time unit), $\lambda_3 = 0.0125$ (1/time unit). The objects that have been rejected immediately begin to be serviced and repaired. In this case, to restore the failed objects, there are two recovery points (RP). The distribution of the time for the recovery of objects on each RP will be assumed to be exponential. For the first one, the recovery intensity is $\mu_1 = 0.033$ (1/ time unit). For the second one, the recovery intensity is $\mu_2 = 0.025$ (1/time unit). To evaluate the efficiency of such a system there will be determined coefficients of usage of all objects, the average time to restore the objects, utilization factors for recovery points.

The intensity of object failure is defined as the inverse of the time t_f of the next object failure:

$$\lambda = \frac{1}{t_f}. \quad (2.4)$$

The intensity of the recovery object is defined as the inverse of the time t_r of the repair time of the object:

$$\mu = \frac{1}{t_r}. \quad (2.5)$$

The model is created in three segments and described in the Appendix A. In the first segment, a stream of rejects of various objects is generated. In the second segment, serviceable objects are maintained. In the third, repair of the failed objects is carried out.

In the first section for third parameter exponential distribution, which is time t_f of the next object failure, we calculate as described below.

For the first object, this parameter will be:

$$t_{f1} = \frac{1}{\lambda_1} = \frac{1}{0.01} = 100.$$

For the second object, this parameter will be:

$$t_{f2} = \frac{1}{\lambda_2} = \frac{1}{0.008} = 125.$$

And for the third object, this parameter will be:

$$t_{f3} = \frac{1}{\lambda_3} = \frac{1}{0.0125} = 80.$$

The second segment of model provides a service not failed objects. The flow of objects entering the service is Poisson flow, and it can be generated using the built-in exponential distribution function. The third parameter in the exponential function is the average time between the inputs of two objects that go one by one to the system for maintenance. It will be equal to:

$$t_{obj} = \frac{1}{\lambda} = \frac{1}{0.1} = 10.$$

The average service time t_s of the object is determined based on the intensity of service of the objects. The distribution of service time for the objects is exponential. Average service time is defined as follows:

$$t_s = \frac{1}{\mu_3} = \frac{1}{0.05} = 20.$$

The report window of simulation model is represented in Appendix B. It can be seen from the simulation results that during the simulation period, equal to 50482.087 time units, 5005 objects were received into the system. From there, losses due to the use of the object or the failure are 1077 objects. The average employment of OBJECT1, OBJECT2 and OBJECT3 is equal to 0.662, 0.534 and 0.380, respectively, and the utilization of the recovery point Point_1 is 0.002. The maximum length of the repair queue for objects is one object. All this information quite fully illustrates the effectiveness of the system.

3 Life safety part

3.1 Analysis of working conditions in a laboratory room with a PC

The objective of this diploma project is the development of virtual lab courses in the Linux operating system environment, that will subsequently be used by students throughout the learning process. The peculiarity of laboratory works will be the replacement of physical network equipment with a virtual one, the virtual model can be quite close to genuine equipment. When changing this particular kind, there are benefits not just in terms of economics, but additionally in terms of protection. Usually, whenever you manage a little known products in the event of incorrect use, different breakdowns might happen. The computer systems on which the software would be installed do not have a danger, both equally for students and for equipment. In the chapter of life safety, the room for the work of the laboratory operator is going to be considered, that is depicted in Figure 3.1

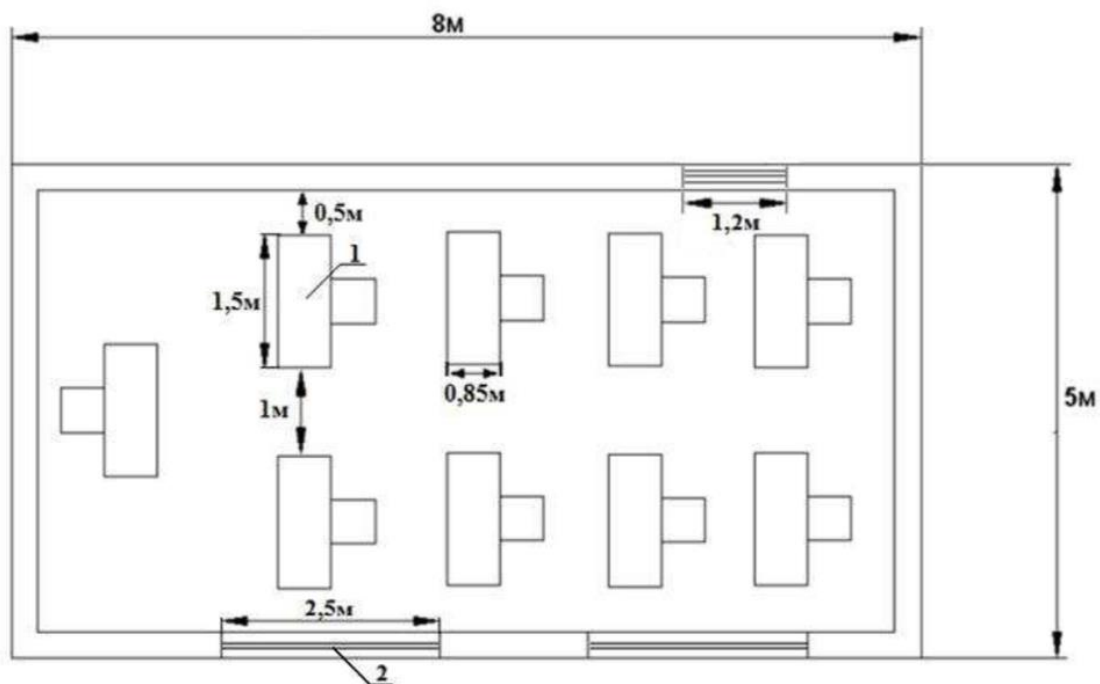


Figure 3.1 – Floor plan

One of the primary conditions under that the regulatory working factors are guaranteed is the appropriate illumination of the operator's location. Places are illuminated to the degree that you are able to work without straining your eyes to be concerned about the whole learning process. It is not welcome to make lighting of sharp areas and shadows of light in an area where laboratory work is going to be performed, as a result of the presence of the impact of blinding.

The audience for the classes is listed at the university, in which classes are mainly conducted in the evening, on this particular basis, natural lighting is considered. As is known, everyday illumination is created by a solar energy, as well as diffusion light of sky radiation.

For regular operation, the computer requires a direct current, and also the feedback of the energy supply device (AC) is provided with an alternating current

of 220 V. In the power cord, the alternating current is transformed into a constant current. The average power of contemporary power items is from 300 to 500 watts.

As in any educational or production establishment, fresh air is provided for proper operation. Not to mention there are actually particular requirements for the amount of cubic meters of air flow for regular working conditions. Because of the air conditioner, a good temperature is properly maintained, that typically corresponds to the requirements.

3.2 The operator visual load during the work with a PC

As many of us use computers at the workplace, computer eye strain has turned out to be the primary gripe related to work. Studies indicate that eye strain and other annoying graphic symptoms happen in 50-90 % of computer personnel.

These issues are able to vary from physical fatigue, decreased efficiency and improved number of operational mistakes, to small irritants like red eyes and eye twitches.

In the procedure of dealing with the Pc, the primary load is a result of vision, because of loss of concentration and power by the eyes throughout the work together with the monitor, unlike various other work types. The blinking of the monitor seems because of the frequency of horizontal deployment in standard monitors, and that does not exceed 60 Hz.

Reflection from walls and other surfaces, and also from the computer screen, can cause computer eye strain. It is essential install a monitor with an anti-reflective covering and also, if at all possible, make the wall surfaces with a matte surface texture.

Students that use glasses must invest in lenses with anti-reflective coating (AR). AR coating decreases glare, reducing the quantity of light reflected out of the front side and back surfaces of the lens cups.

Natural light passes with the light barriers, that are directed to the north and north-east. Those. the sign of natural illumination is generally greater than 1.2 %, in locations with snow cover not less than 1.5 %.

Eliminating external light by closing curtains or blinds is a way out of the circumstances. You are able to additionally decrease the number of fluorescent lamps or bulbs, or you can use tubes and lamps of lower intensity. The place of the monitor or laptop screen needs to be so that the windows are on the side, not the front.

The degree of illumination is supplied taking into account the necessary contrast between the environment and the display around, and the attributes of the works performed as well as the requirements of the user. Illuminance is estimated by the formula (3.1)

$$E = \frac{F}{S} \quad (3.1)$$

where E – surface flux density;

F – uniform distribution of the light flux, perpendicular to illuminated area;

S – illumination.

Among the crucial components used in covering the audience is natural light. For the standardization as well as evaluation of natural lighting, a coefficient or natural lightness coefficient (NLC) is utilized, that could be displayed as the percentage by formula (3.2)

$$NLC = \frac{E_i}{E_n} \quad (3.2)$$

where E_i – Illumination of the object inside the room, lx;

E_n – one-time external illumination of the horizontal area by a diffuse reflection of the sky (without taking into account the direct rays of the sun), lx.

The distribution of NLC inside is uneven and is determined by the way the light apertures shall be located.

In order to keep the operator's normal working state, it is essential to take into consideration the norms and regulations governing the optimality of the workplace content.

3.3 Calculation of natural lighting

Premises with a lasting stay of individuals ought to have natural light. The source of natural lighting will be the sunshine. The lighting conditions within the room are decided primarily by the diffuse light of the sky, and additionally by the reflected light produced by light mirrored out of the earth's surface area.

Establishment of requirements to lighting has various elements, including engineering and physiological. To begin with, the uniform design strategy in addition to maintenance of the apparatus of various functionality (lighting engineering, electrical, ventilating, etc.) is essential. It is identified by economic feasibility, and in certain cases and also that in an interior (it could be carried and not simply to interior lighting) it is not possible to position individual kinds of the equipment without complicated engineering design. Hence it must be saved in mind improving of all the work environment of an inside as an entire.

Light as well a building material, as a stone, a brick and other components of constructions of structures, components as they are perceived just because of light. Therefore light is among the most crucial factors of design. Nevertheless, as practice shows, designers or architects not always are competent to work with properly light that not just depletes architectural idea, but occasionally results in project distortion after the interior due to inappropriate distribution of brightness or even, say, an inappropriate source choice of light looks absolutely different, compared to was slated. It is essential to pay special attention to subjective appraisal which is much more vital, compared to strict fulfillment of numerical values of normed values. Subjective appraisal helps to assess as well as justify new

specifications to quality of burning. Each and every attribute of an observed interior calls for a separate assessment, often merely subjective, often compared to the evaluation of subjective data and objective measurements and it is much more uncommon - merely objective measurements. Natural lighting is split into lateral, upper and combined (top and side). Because natural lighting is unstable, it is able to change significantly even for a couple of minutes, then natural light is normalized not by light, but by the natural lightness coefficient (NLC).

When developing brand new premises, when reconstructing older ones, when developing the natural illumination of various other items, it is essential to identify the spot of the light apertures which provide the standardized worth of NLC in accordance with the demands of BCR RK 2.04-05-2002 "Natural and artificial lighting. Norms of designing» [19].

The cabinet parameters are indicated in Figure 3.2: room length $L = 8$ m, room width $B = 5$ m, room height $H = 4$ m. The height of the working surface above the floor level is $h_r = 0.7$ m. Light sources are gas-discharge lamps. The rated power is 40 W, the nominal luminous flux is 3120 lm.

The office has two windows, window parameters: width 2.5 m and height 1.3 m each. The windows are located on one side of the room, the depth is $l = B - 1 = 4$. Nearby is a nine-story building, located at a distance $P = 15$ m. Windows are plastic with retractable, external adjustable shutters.

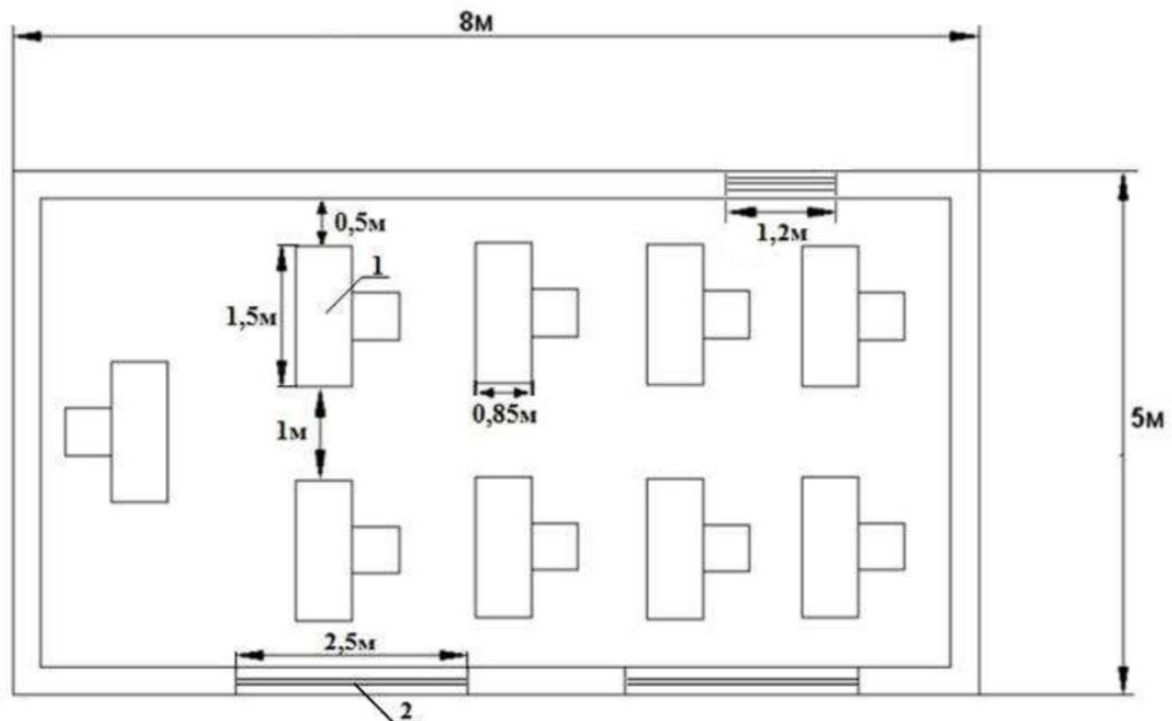


Figure 3.2 – Floor plan (1 – working place, 2 – window)

The total area of the windows is determined by the formula (3.3) for lateral illumination:

$$S_0 = \frac{S_n \cdot e_n \cdot \eta_0 \cdot K_{30} \cdot K_3}{100 \cdot \tau_0 \cdot r_1} \quad (3.3)$$

where S_0 – total area of the lateral light apertures;

S_n – floor area, m^2 ;

e_n – normalized value of NLC;

K_3 – safety factor;

τ_0 – common light transmittance $\tau_0 = \tau_1 \cdot \tau_2 \cdot \tau_3 \cdot \tau_4$;

η_0 – light characteristic of windows;

r_1 – coefficient which takes into consideration the increased NLC with side lighting effects due to light reflected out of the surfaces of the room and also the main level adjacent to the building;

$K_{3\text{д}}$ – coefficient that takes into account the shading of windows by opposing buildings.

Floor area of the room

$$S_n = B \cdot L = 8 \cdot 5 = 40m^2.$$

Safety factor $K_3 = 1.2$ with natural side lighting.

Find the value of the natural light coefficient by the formula (3.4)

$$e_N = e_{NLC} \cdot m \quad (3.4)$$

where e_{NLC} – value of NLC, for class of visual work IV, $e_{NLC} = 1.2$;

m – coefficient of light climate, $m = 0.65$.

The NLC value, taking into account the coefficients m , is equal to

$$e_N = 1.2 \cdot 0.65 = 0.78.$$

For windows with double-glazed windows, the light transmittance $\tau_1 = 0.8$. Since the windows with double-glazed windows, then $\tau_2 = 0.75$. With side illumination, $\tau_3 = 1$. As shading devices used with retractable adjustable louvers, so $\tau_4 = 1$.

$$\tau_0 = 0.8 \cdot 0.75 \cdot 1 \cdot 1 = 0.6.$$

To determine the coefficient η_0 , it is necessary to know the ratio of length to depth (to the most remote point from the window). Since the windows are located only on one side, this ratio is equal to

$$\frac{L}{l} = \frac{8}{4} = 2.$$

It is necessary to know the ratio $\frac{B}{h_1}$, where h_1 – height from the level of the conventional working surface to the top of the window.

$$h_1 = h_w + h_{n.w} - h_r = 3 + 0.8 - 0.7 = 3.1 \text{ m},$$

$$\frac{B}{h_1} = \frac{5}{3.1} = 1.61.$$

For the relations found, we determine that the coefficient η_0 is equal to $\eta_0 = 9.5$.

To define coefficient r_1 it is necessary to find $\frac{l}{B}$, where l – distance of the calculated point from the outer wall at the side lighting. Because of this situation, for probably the most distant point from the window, we are able to take $l = 3$, in which situation this ratio is equal to

$$\frac{l}{B} = \frac{3}{5} = 0.6.$$

Taking factor $\rho_{av} = 0.5$, the ratio r_1 is equal to

$$r_1 = 2.$$

In order to choose the coefficient $K_{3д}$, we figure out the level of the neighboring home. To get this done, conditionally take into consideration the interfloor overlapping is 30 m, and the attic is 2 m. Hence the height of the structure is

$$H = 30 + 2 = 32 \text{ m}.$$

Figure out the height of the place of the curtain of the opposite building above the window sill of the window in question, in case the functioning place is on the fourth floor. Then

$$H_c = 32 - 4 = 28 \text{ m}.$$

Coefficient $K_{3д}$ can be found from ratio $\frac{P}{H_c}$ which is equal to

$$\frac{P}{H_c} = \frac{15}{28} = 0.53.$$

From there, $K_{3д}$ is determined as $K_{3д} = 1.7$.

Substituting the numerical values in the formula (3.4) we get

$$S_0 = \frac{40 \cdot 0.78 \cdot 9.5 \cdot 1 \cdot 1.7}{100 \cdot 0.6 \cdot 2} = 4.20 \text{ m}^2.$$

Based on the calculation, the window area of one room must be no less than 4.2 m², but within the room there are two windows, each with an area of 3.25 m², that provides complete window area of 6.50 m². Consequently, the current window does match to the norms of healthy lighting.

3.4 Calculation of artificial lighting

Of all energy types that individuals are able to use, light is the most crucial. The majority of the information that an individual receives through his senses comes because of the light - approximately 80 %. The state of mind and also the degree of fatigue count on the lighting as well as color of the surrounding items. In terms of work safety, visual comfort and visual ability are incredibly significant. This is clarified by the simple fact that many of accidents happen since of unsatisfactory coverage or since of mistakes made in order to the worker because it was hard for him to identify an item or even to recognize the level of danger regarding servicing devices, vehicles, and so on.

The types of artificial lighting are: common, when the lighting devices are placed in such a way as to provide sufficient illumination in the work area, and combined, when in addition to general lighting, local lighting fixtures are installed to create higher levels of illumination in workplaces where intense visual work is performed. The device in rooms only with local lighting is prohibited.

Visual impairments associated with deficiencies in the lighting system are routine in the workplace.

Calculation of electrical lighting is carried through to establish the primary parameters of the illumination installation: the amount as well as power of number, the type, and light sources of lamps, the placement of theirs, that will provide normalized lighting at workplaces.

Room length $A = 8$ m, room width $B = 5$ m, room height $H = 4$ m. The height of the working surface above the floor level is $h_r = 0.7$ m. The luminaire overhang distance is $h_{ov} = 0$ m.

The worth of the coefficient is discovered out of the tables connecting the geometric details of the rooms (index of rooms i) with their optical characteristics (the coefficients of the reflection of the ceiling ρ_c , walls ρ_w and floor ρ_f).

Index of rooms i is calculated from the formula (3.5)

$$i = \frac{A \cdot B}{h \cdot (A + B)} \quad (3.5)$$

where A – length of the room;

B – width of the room;

$h = H - (h_r + h_{ov})$ – estimated height.

Determine the value of h

$$h = 4 - (0.7 + 0) = 3.3 \text{ m.}$$

When the estimated height is found, determine the index of the room

$$i = \frac{8 \cdot 5}{3.3 \cdot (8 + 5)} = 0.93.$$

For the values of reflection, it is taken

$$\rho_c = 70\%, \rho_w = 50\%, \rho_f = 30\%.$$

Defining coefficient η , which is equal to $\eta = 42\%$.

For lighting of laboratory room we will use luminescent gas-discharge lamps with power of 40 W and a nominal light flux of 3120 lm. As luminaires we will use luminaires of PVLM 1x40 type.

Calculation of artificial lighting is performed using the light flux utilization method, a light flux is determined by the formula (3.6)

$$F = \frac{E \cdot K_r \cdot S \cdot Z}{\eta} \quad (3.6)$$

where E – minimum illumination according to standards, for IV (b) class of work according to table 3.12 [28] – 200 lx;

$K_r = 1.5$ – safety factor;

S – area of a room;

$Z = 1.1 \div 1.2$ – illumination unevenness factor.

$$F = \frac{200 \cdot 1.5 \cdot 40 \cdot 1.1}{0.42} = 31428 \text{ lm.}$$

The number of required luminaires is calculated by the formula (3.7)

$$N = \frac{F}{F_l} \quad (3.7)$$

where $F_l = 3120$ – rated light flux.

$$N = \frac{31428}{3120} = 10 \text{ pc.}$$

The actual illumination is determined by formula (3.8)

$$F = \frac{E \cdot K_r \cdot S \cdot Z}{N \cdot \eta} \quad (3.8)$$

$$F = \frac{200 \cdot 1.5 \cdot 40 \cdot 1.1}{10 \cdot 0.42} = 3142 \text{ lm.}$$

Arrangement of luminaires is described in Figure 3.3.

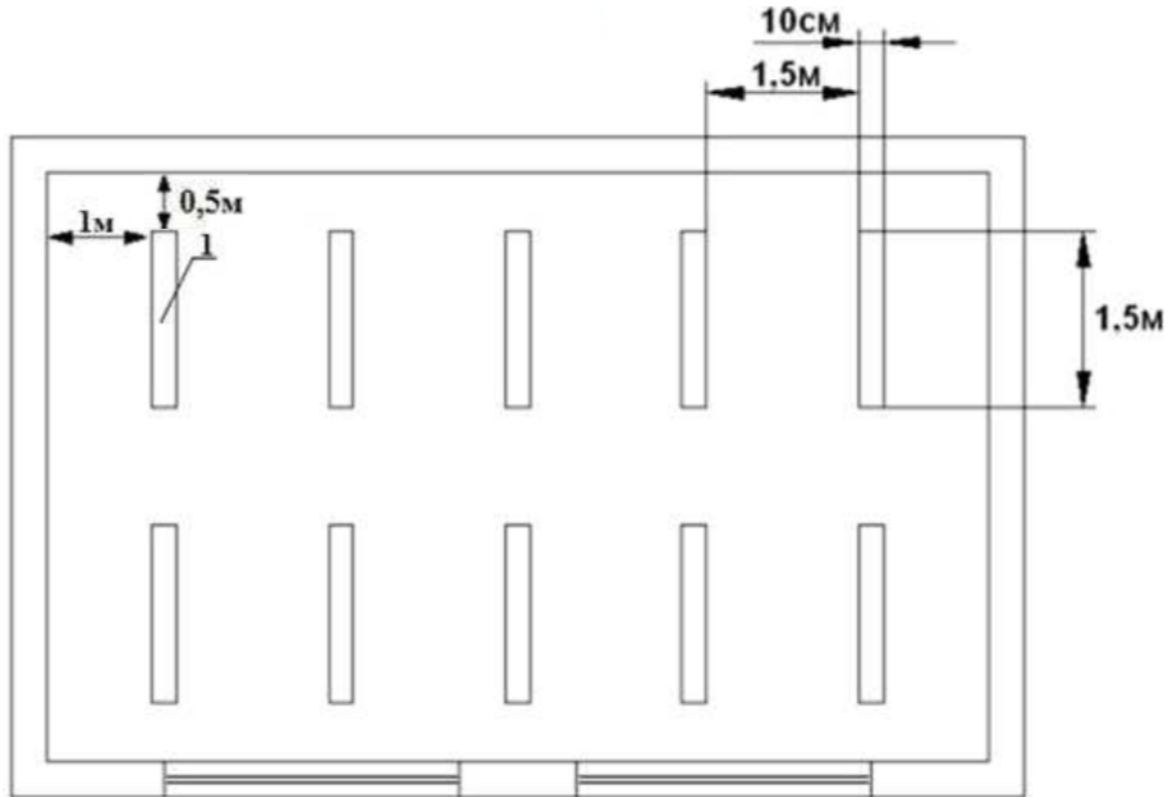


Figure 3.3 – Plan of arrangement of luminaires

Conclusion: As can be seen from the calculations, 10 luminaires of the PVLM 2x40 type are sufficient to provide illumination of 200 lx of a lab room with an area of 40 m². Thus, in the evening, artificial lighting will provide the necessary illumination.

4 Economical part

4.1 Technical and economic justification for the realization

The theme of the paper focuses on the creation of a complex laboratory work, the main aim of which is the study of data networking virtualization technologies.

Modern trends, such as an increase in the number of devices connected to the Internet, exponential growth in information volumes, the development of cloud technologies are changing the telecom. There is an increase in the volumes of network traffic, and business increasingly needs to configure large-scale networks.

Now, in the absence of a miscalculation of the direct benefits from the introduction and operation of information technologies, no reasonable leader who manages a competitive company will enter the information system into the creation without careful analysis and determining its economic efficiency and expediency, since the cost of the error has the potential to amount to hundreds of thousands of dollars. The introduction practice has shown that the methodology and special approaches will be required to assess the economic efficiency.

This section provides an overview of the economic implementation of this work, reflecting the time, labor and financial costs of the project. In addition to technical parameters, one of the main parameters is the economic efficiency of the project. Characteristics of increasing the efficiency of net income and the discounted payback period of investments. When the project is implemented, the project must also take into account the internal rate of return and the cost of the project.

Thus, to assess the technical and economic efficiency of the project, it is necessary: to calculate the capital costs; calculate the number of employees, the cost of services, income, effective capital investments and investments.

4.2 Calculation of capital costs

Capital costs are calculated by the formula (4.1)

$$C = P + C_p + C_s + C_d, \quad (4.1)$$

where P – price for network equipment;

C_p – cost of workplaces for 1 year;

C_s – cost for installation of equipment (5% of the cost of equipment);

C_d – cost of the development period.

In the diploma project, laboratory work is virtual, therefore it follows that physical equipment will not be used, the work is done in a virtual environment. This virtual machine will easily replace the required switches and client machines. Table 4.1 describes physical equipment that could be discarded.

Table 4.1 – Name and value of physical equipment

Name of equipment	Quantity	Price, tenge	Cost, tenge
D-Link Switch DGS-3650	4	366 120	1 464 480
SFP module D-Link DEM-315GT	10	73 600	736 000
Total			2 200 480

The cost of equipment is provided by Ruba Technology, taking into account the exchange rate as of April 13, 2018 - 329 tenge per dollar. Based on the data in Table 4.1, it can be noted that higher costs are expected, which is the reason for the urgency of training on the virtual simulator, which fully pays for these costs [20].

Costs of equipment will be 2 200 480 tenge, but creating virtual laboratory works we reduce costs to zero.

Table 4.2 – Calculation of costs for the organization of the workplace [21, 22, 23]

Name of equipment	Quantity	Price, tenge	Cost, tenge
Laptop Lenovo IdeaPad 320	8	195 990	1 567 920
Computer desk	8	9 825	78 600
Chair	8	5 165	41 320
Total			1 687 840

Total costs for the organization of workplace: 1 687 840 tenge.

Calculating the cost of the development of laboratory work by the formula (4.2)

$$C_d = S_{dev} + D_{PC} + P_r + C_{el} + E_n, \quad (4.2)$$

where S_{dev} – payment to the developer;

D_{PC} – computer depreciation;

P_r – monthly rent price;

C_{el} – the amount for electricity during the project;

E_n – cost of missing equipment and additional materials.

Electricity costs are calculated by the following formula (4.3)

$$C_{el} = W \cdot T \cdot S, \quad (4.3)$$

where W – consumable power $W = 0.3$ kW;

T – working hours;

S – cost of 1 kW per hour of electricity $S = 18.32$ tenge/kW-h.

On average, in one month 22 six-hour working days, $T = 132$ h.

Calculation of electricity costs

$$C_{el} = 0.3 \cdot 132 \cdot 18.32 = 725 \text{ tenge.}$$

The power consumed for necessary needs is calculated as 5% of the power used by the main equipment. Cost of electricity for necessary needs

$$C_{el.n} = C_{el} \cdot 0.05 = 36 \text{ tenge.}$$

Total energy costs

$$C_{el.total} = C_{el} + C_{el.n} = 725 + 36 = 761 \text{ tenge.}$$

The cost of additional materials is 5% of the cost of the system

$$E_n = 195\,990 \cdot 0.05 = 9800 \text{ tenge.}$$

Computer depreciation will be 40% of the price

$$D_{PC} = 195\,990 \cdot 0.4 = 78\,396 \text{ tenge.}$$

Calculate the cost of developing laboratory works according to formula (4.2)

$$C_d = 255\,370 + 78\,396 + 30\,000 + 761 + 9800 = 374\,327 \text{ tenge.}$$

Table 4.3 – Calculation of costs for the development

Name of expenses	Cost, tenge
Monthly salary for the developer	255 370
Computer depreciation at development time	78 396
Monthly rent of a premise in development	30 000
Energy costs during the development	761
Costs for additional materials and missing equipment at the time of development	9800
Total	374 327

Calculate the capital costs by the formula (4.1)

$$C = 0 + 1\,687\,840 + 84\,392 + 374\,327 = 2\,146\,559 \text{ tenge.}$$

Table 4.4 displays all capital costs that are necessary and sufficient for the organization of workplace.

Table 4.4 – Capital costs for the organization of the workplace

Name of expenses	Cost, tenge	Specific weight, %
Cost of equipment	0	0
Cost of workplaces	1 687 840	79
Installation of equipment	84 392	4
Development costs	374 327	17
Total	2 146 559	100

Figure 4.1 builds the structure of capital expenditures. The primary attribute of capital expenditures is actually the duration of the use of theirs. In case the business plans to make use of investments in assets for over one season, then, probably, they will be classified as capital. You can clearly see that most of the costs covers the organization of workplaces. There was no cost for equipment as we abandoned all physical equipment. We made a substitution of physical devices for computers with virtual open source applications and solutions such as virtual switches, virtual machines.

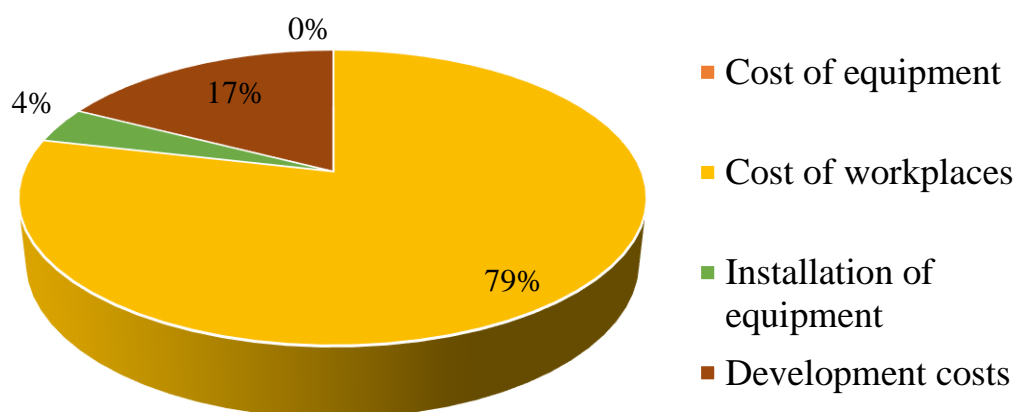


Figure 4.1 – Structure of capital costs

4.3 Calculation of annual operating costs

Operational costs are determined by the formula (4.4)

$$C_{op} = PF + T_s + D + C_{el} + M + C_{adm}, \quad (4.4)$$

where PF – payroll fund (basic and additional wages);

T_s – social tax;

D – depreciation deductions;

C_{el} – electric power from the production side;

M – costs for materials and spare parts;

C_{adm} – other administrative and operational expenses.

Table 4.5 - Average salary of staff per month

List of staff	Quantity	Monthly salary, tenge	Salary per year, tenge
Engineer	1	80 000	720 000
Teacher	1	150 000	1 350 000
Total			2 070 000

Average salary of the working staff per month is given in Table 4.5 to determine the wage,

The wage fund also takes into account additional wages (payment for work on holidays, overtime, etc.) in the amount of 30% of the basic salary.

Laboratory work courses are designed for a period of 1 month. Given that the operator will maintain the equipment for 9 months.

The additional wage is calculated by the formula (4.5)

$$W_a = W_p \cdot 0.3, \quad (4.5)$$

where W_p – annual basic salary fund.

Substituting these values into formula (4.5), we calculate the total sum of additional wages:

$$W_a = 2\,070\,000 \cdot 0.3 = 621\,000 \text{ tenge.}$$

Payroll fund is equal to the amount of basic and additional wages:

$$PF = W_p + W_a, \quad (4.6)$$

We calculate the wage fund according to the formula (4.6)

$$PF = 2\,070\,000 + 621\,000 = 2\,691\,000 \text{ tenge.}$$

Deductions for social tax are from 11%:

$$T_s = 0.11 \cdot (PF - 0.1 \cdot PF) = 266\,409 \text{ tenge.}$$

The amount of depreciation is accounted according to the established norms, which are calculated as a percentage of the value of fixed assets (4.7)

$$D_0 = \frac{F \cdot N_D}{100\%}, \quad (4.7)$$

where F – carrying value of fixed assets, tenge;

N_D – depreciation rate.

Calculate the depreciation of equipment and office furniture using the formula (4.7).

Depreciation of computer equipment is 40% of the price:

$$D_1 = 1\,567\,920 \cdot 0.4 = 627\,168 \text{ tenge.}$$

Depreciation of office furniture is 15% of the price:

$$D_2 = 119\,920 \cdot 0.15 = 17\,988 \text{ tenge.}$$

$$D = D_1 + D_2 = 627\,168 + 17\,988 = 645\,156 \text{ tenge.}$$

Cost of electric power is calculated by the formula (4.8)

$$C_{el} = W \cdot T \cdot S, \quad (4.8)$$

where W – consumable power $W = 1.6 \text{ kW}$;

T – hours of work for 1 year, $T = 2200 \text{ hours per year}$;

S – cost of 1 kW per hour of electricity $S = 18.32 \text{ tenge/kW-h}$.

Define the cost of electricity:

$$C_{el} = 1.6 \cdot 2200 \cdot 18.32 = 64\,486 \text{ tenge.}$$

The power consumed for necessary needs, for example light, or occasionally the inclusion of an air conditioner is calculated as 5% of the power used by the main equipment. The cost of electricity for the necessary needs:

$$C_{el.n} = C_{el} \cdot 0.05 = 3\,224 \text{ tenge.}$$

Total energy costs:

$$C_{el.total} = C_{el} + C_{el.n} = 64\,486 + 3\,224 = 67\,710 \text{ tenge.}$$

The cost of additional materials and missing equipment is calculated as 5% of the cost of the system:

$$M = 1\,687\,840 \cdot 0.05 = 84\,392 \text{ tenge.}$$

Expenses of additional costs is 10% of the operational costs:

$$C_{adm} = 3\,754\,667 \cdot 0.1 = 375\,467 \text{ tenge.}$$

Consequently, the operating costs based on the formula (4.4) will be:

$$C_{op} = 2\,691\,000 + 266\,409 + 645\,156 + 84\,392 + 67\,710 + 375\,467 = 4\,130\,134 \text{ tenge.}$$

Enter the data on operating costs in Table 4.6 and calculate the specific weight of each cost.

Table 4.6 – Operating costs

Operating cost Items	Cost, tenge	Specific weight, %
Payroll fund	2 691 000	65%
Social tax	266 409	6%
Depreciation deductions	645 156	16%
Costs for materials and spare parts	84 392	2%
Electricity costs	67 710	2%
Other expenses	375 467	9%
Total	4 130 134	100

The most part of the operating costs is occupied by the payroll fund. Important in these costs are depreciation charges. The smallest part of the operating costs is allocated to electricity because of the selection of energy-saving equipment.

Figure 4.2 represents the structure of operating costs.

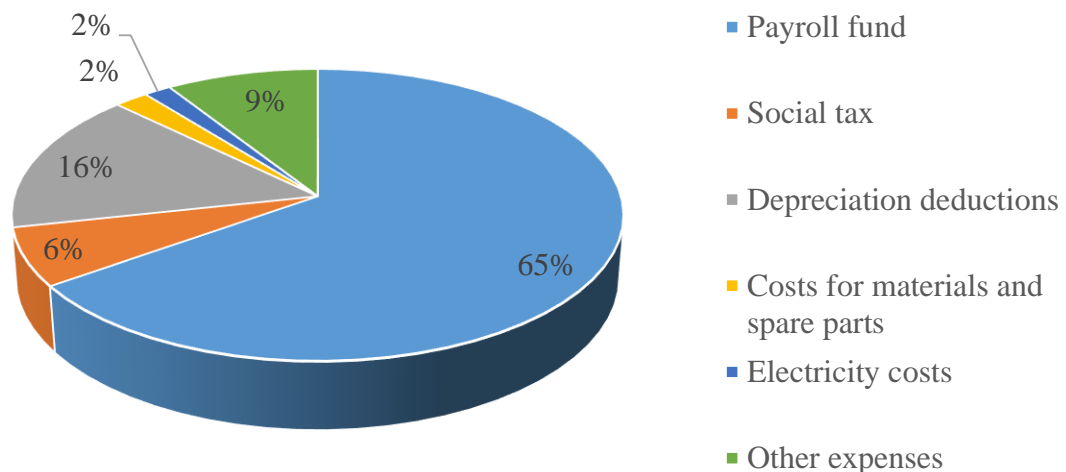


Figure 4.2 – Structure diagram of operating costs

4.4 Calculation of income

It is calculated that the courses will be 4 groups of 10 people, given that in the academic year is 9 months, the cost of tuition is expected to be set at 30 000 tenge per person.

$$I = 30\,000 \cdot 5 \cdot 8 \cdot 9 = 10\,800\,000 \text{ tenge.}$$

As a result, we get that the income will be 10.8 million tenge.

4.5 Calculation of economic efficiency

The profit from the introduction of these rates is the income from the core business, net of operating expenses. Net profit is determined by deducting income tax of 30% (for the Republic of Kazakhstan)

We calculate the profit of the enterprise before taxation.

Profit from the introduction of training courses is calculated by formula (4.9)

$$P = I - C_{op}, \quad (4.9)$$

where I – annual income;

C_{op} – operating costs.

$$P = 10\,800\,000 - 4\,130\,134 = 6\,669\,866 \text{ tenge.}$$

The net income remaining available is profit taking into account income tax.

The calculation of corporate tax will be 20% of the profit under the formula (4.10)

$$CT = P \cdot 20\%, \quad (4.10)$$

$$CT = 6\,669\,866 \cdot 0.2 = 1\,333\,973 \text{ tenge.}$$

The amount of net profit taking into account income tax according to the formula (4.11) will be

$$NP = P - CT, \quad (4.10)$$

$$NP = 6\,669\,866 - 901\,973 = 5\,335\,893 \text{ tenge.}$$

Absolute economic efficiency will be calculated using formula (4.12)

$$E = NP / C_{\Sigma}, \quad (4.12)$$

$$E = 5\,335\,893 / 2\,146\,559 = 2.48.$$

The payback period is the reciprocal of the absolute economic efficiency and is given by formula (4.13)

$$T = 1/E, \quad (4.13)$$

$$T = 1/2.48 = 0.40 \text{ year.}$$

Based on the received data, we get that the costs for this project will pay off in approximately 5 months.

All the economic indicators for the project are shown in Table 4.7.

Table 4.7 – Indicators of economic efficiency of the project

Indicator	Value
Capital costs, tenge	2 146 559
Operating costs, tenge	4 130 134
Profit before taxation, tenge	6 669 866
Profit after taxation, tenge	5 335 893
Economic efficiency	2.48
Payback period, months	5 months

Taking into account all the data obtained, in the development of complex of laboratory works on data networking virtualization technologies at an educational institution with capital expenditures in the amount of 2 146 559 tenge, net annual income will amount to 5 335 893 tenge. This project pays off for 5 months. Consequently, it can be concluded that the introduction of such laboratory works is cost-effective.

Conclusion

This paper has covered technologies and necessary knowledge to introduce different methodology for laboratory works. The theoretical part describes modern network technologies overview. Technical part of this paper consists of troubleshooting principles and necessary prerequisites, Linux diagnostics commands and sequences of laboratory works that are being proposed. Laboratory works in this case are based on learning networking, virtualization and storage management using Linux and tools such as Open vSwitch, Kernel-based Virtual Machine, Logical Volume Manager, etc. Mathematical approach of troubleshooting is introduced as automating diagnosis and troubleshooting tasks using Bayesian Networks.

In life safety part of this paper, the conditions required for the correct operation of the operator are calculated, namely the necessary natural and artificial illumination of the laboratory room.

Economic justification, capital costs, annual operating costs are presented in the economic section. With making laboratory courses paid, the payback period of the project is determined.

List of abbreviations

SDN – software-defined network
NFV – network functions virtualization
QoE – quality of experience
QoS – quality of service
API – application programming interface
REST – Representational State Transfer
LAN – local area network
VM – virtual machine
IDS – intrusion detection system
IPS – intrusion prevention system
OS – operating system
PC – personal computer
VMM – virtual machine monitor
CPE – customer premises equipment
VNF – virtual network functions
MANO – management and orchestration
VNFC – virtual network functions components
VLAN – virtual local area network
MAC – media access control
NIC – network interface card
IEEE – Institute of Electrical and Electronics Engineers
VID – virtual local area network identifier
TCI – tag control information
VXLAN – virtual extensible local area network
STP – Spanning Tree Protocol
UDP – User Datagram Protocol
IP – Internet Protocol
VTEP – virtual extensible local area network tunnel endpoint
NAT – network address translation
IPv4 – Internet Protocol version 4
IPv6 – Internet Protocol version 6
OSPF – open shortest path first
BGP – Border Gateway Protocol
ALG – application layered gateway
ECN – Explicit Congestion Notification
PDU – protocol data unit
TTL – time to live
IANA – internet Assigned Numbers Authority
IETF – Internet Engineering Task Force
RFC – request for comments
ACE – application control module
VPN – virtual private network

LSB – least significant bit
USE – utilization, saturation and errors
CPU – central processing unit
I/O – input/output
TCP – Transmission Control Protocol
OVS – Open vSwitch
KVM – Kernel based Virtual Machine
LVM – local volume management
iSCSI – internet small computer systems interface
LACP – Link Aggregation Control Protocol
GRE – generic routing encapsulation
STT – stateless transport tunneling
LISP – Locator/Identifier Separation Protocol
ACL – access control list
IPFIX – Internet Protocol Flow Information Export
OVSDb – Open vSwitch database
BSD – Berkeley Software Distribution
AROS – Amiga Research Operating System
EPT – extended page table
AMD – Advanced Micro Devices
RVI – rapid virtualization indexing
KSM – Kernel Same-page Merging
LUN – logical unit number
IQN – iSCSI Qualified Name
BN – Bayesian network
AC – alternating current
NLC – natural lightness coefficient
BCR – building control and regulations
RK – Republic of Kazakhstan

List of references

- 1 Jim Doherty, SDN and NFV Simplified. 2016 Pearson Education Inc.
- 2 Benzekki Kamal, Software-defined networking (SDN): a survey., Security and Communication Networks 9, no. 18 (2016): 5803-5833.
- 3 Kreutz D., Software-Defined Networking: A Comprehensive Survey. Proceedings of the IEEE, January 2015.
- 4 William Stallings, Foundations of Modern Networking, 2016 Pearson Education Inc.
- 5 Steve Herrod, "Towards Virtualized Networking for the Cloud". (August 30, 2011). VMware.
- 6 RFC-7348 // ietf.org: Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks: <https://tools.ietf.org/html/rfc7348>
- 7 VXLAN Overview: Cisco Nexus 9000 Series Switches // cisco.com: <https://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/white-paper-c11-729383.html>
- 8 Istvan Pelle, Felician Nemeth and Andras Gulyas. A Little Less Interaction, A Little More Action: A Modular Framework for Network Troubleshooting. Budapest University of Technology and Economics, Hungary, HSNLab, Dept. of Telecommunications and Media Informatics. March 19, 2018.
- 9 David H. Jonassen and Woei Hung. Learning to Troubleshoot: A New Theory-Based Design Architecture. Educational Psychology Review, Vol. 18, No. 1, March 2006.
- 10 Dreyfus, H. L., and Dreyfus, S. E. Mind Over Machine, The Free Press, New York. (1986).
- 11 Aamodt, A., and Plaza, E. Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications 7, (1994). p. 39–59.
- 12 Rasmussen, J. Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering, North-Holland, Amsterdam. (1984b).
- 13 MacPherson, R. T. Factors affecting technological trouble shooting skills. J. Ind. Teach. Educ. 35(4), (1998): p. 5–28.
- 14 Mahwah, NJ Schaafstal, A., and Schraagen, J. M. Training of troubleshooting: A structured, task an- alytical approach. In Schraagen, J. M., Chipman, S. F., and Shalin, V. L. (eds.), Cognitive Task Analysis, Lawrence Erlbaum Associates, (2000), pp. 57–70.
- 15 A step toward creating a reactive learning environment. International Journal of Man-Machine Studies 7(5) Brown, J. S., Burton, R. R., Bell, A. G. (1975): 675–696.
- 16 Open vSwitch // openvswitch.org: Open vSwitch Documentation // <http://docs.openvswitch.org/en/latest/>

17 C. Pham, D.Chen, Z. Kalbarczyk, R. K. Iyer. CloudVal: A Framework for Validation of Virtualization Environment in Cloud Infrastructure. Columbia University, New York, NY, USA. 2014.

18 RFC3720 // ietf.org: Internet Small Computer Systems Interface: <https://tools.ietf.org/html/rfc3720> –.

19 СНиП РК 2.04-05-2002 «Естественное и искусственное освещение. Общие требования».

20 Catalogue of switches of the Ruba Technology company: <http://wifi.kz/> (reference date 11.04.2018)

21 Online shop for electronics. // technodom.kz: Laptop selection page. 2018: <https://www.technodom.kz/catalog/notebooks/p/67879> (reference date 18.04.2018).

22 Online shop for furniture. // zeta.kz: Computer table selection page. 2018: <http://zeta.kz/ru/product/2601-kul-132/> (reference date 18.04.2018).

23 Online shop for furniture. // zeta.kz: Chair selection page. 2018: <http://zeta.kz/ru/product/1178-izo-n/> (reference date 18.04.2018).

24 Hongyi Zeng, Peyman Kazemian, George Varghese, Nick McKeown. A Survey on Network Troubleshooting. Stanford University, Stanford, CA, USA.

25 Evgenia F. Adamopoulou, Konstantinos P. Demestichas. Enhancing Channel Estimation in Cognitive Radio Systems by means of Bayesian Networks. Springer Science+Business Media, LLC. 2008.

26 Khanafer, R., Moltsen, L., Dubreil, H., Altman, Z. and Barco, R., 2006, September. A bayesian approach for automated troubleshooting for UMTS networks. IEEE.

27 Brendan Gregg // brendangregg.com: The USE Method a performance methodology for identifying resource bottlenecks: <http://www.brendangregg.com/usemethod.html>

28 Brendan Gregg // brendangregg.com: Linux Performance: <http://www.brendangregg.com/linuxperf.htm>

29 F. Jensen, Bayesian Networks and decision graphs, Springer-Verlag, 2001.

30 Абдимуратов Ж.С., Манабаева С.Е. Безопасность жизнедеятельности. Методические указания к выполнению раздела «Расчет производственного освещения» в выпускных работах для всех специальностей. Бакалавриат – Алматы: АИЭС, 2009.

31 Базылов К.Б., Алибаева С.А., Бабич А.А. Методические указания по выполнению экономического раздела дипломной работы бакалавров для студентов всех форм обучения специальности 050719 – Радиотехника, электроника и телекоммуникации. – Алматы: АИЭС, -2009. -19 с.

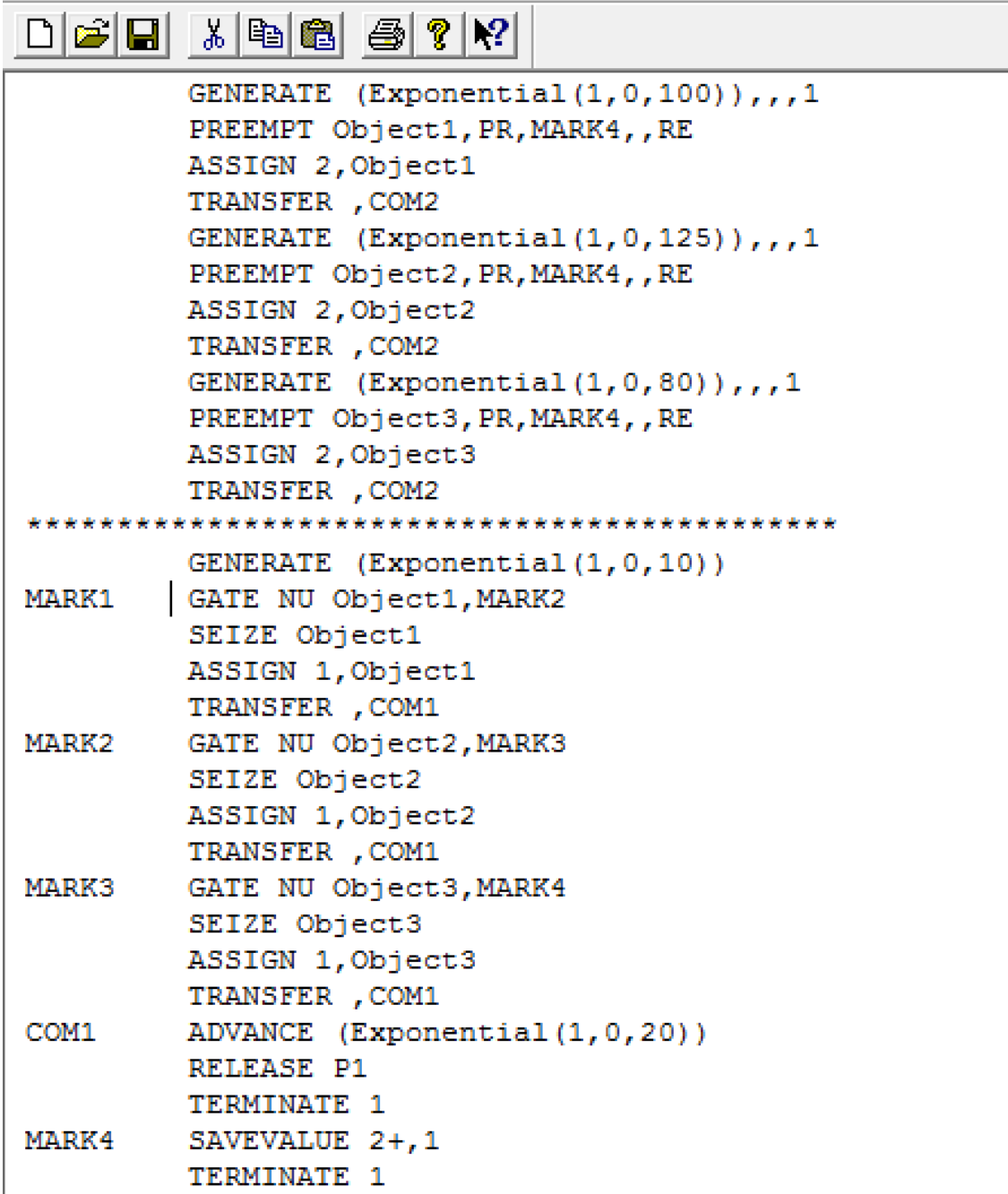
32 Голубицкая Е.А. Экономика связи: М.: -Ирмас, 2006.

Базылов К.Б., Алибаева С.А., Бабич А.А. Методические указания по выполнению экономического раздела выпускной работы бакалавров. – Алматы: АИЭС, 2009. - 44 с

33 С.В. Коньшин, А.П. Кондратович. СТ НАО 56023-1910-04-2014. Общие требования к построению, изложению, оформлению и содержанию учебно-методических и учебных работ. Алматы: АУЭС, 2014

Appendix A

Listing of the error service simulation model in the GPSS World



The screenshot shows the GPSS World software interface. At the top is a toolbar with icons for file operations (new, open, save, print, copy, paste, delete, undo, redo), a help icon, and a cursor icon. Below the toolbar is a text area containing the simulation model code. The code is organized into sections, with labels on the left side of the text area. The code includes GENERATE, PREEMPT, ASSIGN, TRANSFER, GATE, SEIZE, ADVANCE, RELEASE, and TERMINATE commands. A line of asterisks separates two main sections of the code.

```
GENERATE (Exponential(1,0,100)),,1
PREEMPT Object1,PR,MARK4,,RE
ASSIGN 2,Object1
TRANSFER ,COM2
GENERATE (Exponential(1,0,125)),,1
PREEMPT Object2,PR,MARK4,,RE
ASSIGN 2,Object2
TRANSFER ,COM2
GENERATE (Exponential(1,0,80)),,1
PREEMPT Object3,PR,MARK4,,RE
ASSIGN 2,Object3
TRANSFER ,COM2
*****
MARK1  | GENERATE (Exponential(1,0,10))
      | GATE NU Object1,MARK2
      | SEIZE Object1
      | ASSIGN 1,Object1
      | TRANSFER ,COM1
MARK2  | GATE NU Object2,MARK3
      | SEIZE Object2
      | ASSIGN 1,Object2
      | TRANSFER ,COM1
MARK3  | GATE NU Object3,MARK4
      | SEIZE Object3
      | ASSIGN 1,Object3
      | TRANSFER ,COM1
COM1   | ADVANCE (Exponential(1,0,20))
      | RELEASE P1
      | TERMINATE 1
MARK4  | SAVEVALUE 2+,1
      | TERMINATE 1
```

Figure A.1 – Listing of the simulation model

Continuation of Appendix A

```
*****
COM2      TRANSFER BOTH,SER1,SER2
SER1      QUEUE Point_1
          SEIZE Point_1
          DEPART Point_1
          ADVANCE (Exponential(1,0,30))
          RELEASE Point_1
          TRANSFER ,COM3
SER2      QUEUE Point_2
          SEIZE Point_2
          DEPART Point_2
          ADVANCE (Exponential(1,0,40))
          RELEASE Point_2
          TRANSFER ,COM3
COM3      RETURN P2
          SAVEVALUE 1+,M1
          TERMINATE
          START 5000
```

Figure A.2 –Continuation of the model listing

Appendix B

GPSS World simulation report

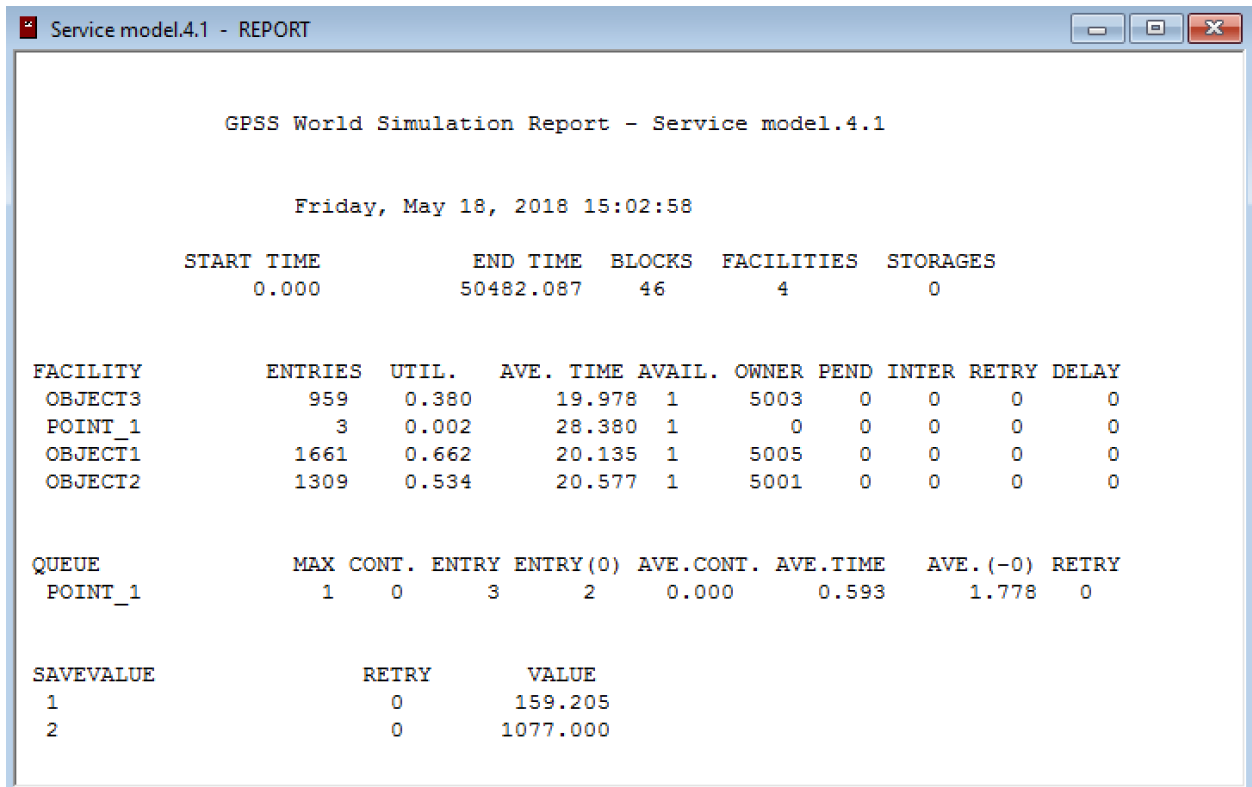


Figure B.1 – Report of the simulation

Appendix C

Anti-plagiarism report

Appendix D

Electronic version of DP and demonstration materials (CD-R)

Appendix E

Handout