My name is Timur. I am from Uzbekistan, Tashkent. I am 25 years old.

I graduated from Tashkent University of information technologies with Telecommunication Technologies specialty.

Before and while I was student I worked as IT Supporter, who fix problems with computer and other office equipment.

After I graduated from University, I worked in Huawei. My position was IT Engineer, it is something like a Systems Engineer. My responsibilities was integration, supporting, maintenance Huawei products and solutions. For example, Beeline order solution for traffic analyzing from Huawei and Huawei have this solution and offer to Beeline. After our project manager provide and organize the subcontractors. Subcontractors installing equipment, after installation I begin to configure devices. It is one of the example, also I worked with VideoConference systems, especially I maintained government videoconference. Also we had some military projects.

When I was working in Huawei, last half a year I was thinking about switching position to DevOps, because it is interesting for me. It is difficult to find DevOps job I Uzbekistan.

But I found it. So they were looking for DevOps expert with big experience in bank. They said that first time I will be QA Engineer, next I will be QA and Release Engineer. And in near future our Bank will invite Russian IT specialist from Yandex and BSS, so and this specialist will develop new infrastructure, also the will integrate real financial technology devops and teach us.

Build

Mvn install

JS – npm or yarn. Frontend - webpack

Project must have some package.json file for building app. Npm

# DevOps

is a set of practices for improving the efficiency of development (Development) and operation (Operation) of software (software) through their continuous integration and unsing of special automation tools

# Cloud Computing

Cloud computing means delivery of IT resources on demand over the Internet with paying for it. Lease of computing resources.

**SaaS. Is a prepared product that is launched and managed by a service provider. Most commonly SaaS solutions is end-user applications (such as email websites)**

**PaaS**. The user is provided with a computer platform, with an installed operating system, possibly with a software one.

**IaaS**. includes computing resources: virtual servers, storage, networks. These are something like virtual "computers" on which you can install anything: operating system, software, applications.

# Network

**Computer network**, two or more [computers](https://www.britannica.com/technology/computer) that are connected with one another for the purpose of communicating data electronically

* 10.0.0.0 – 10.255.255.255.
* 172.16.0.0 – 172.31.255.255.
* 192.168.0.0 – 192.168.255.255.

LAN – is a group of computers connected to each other in a small area such as building, office.

PAN – is used for connecting the devices of personal use.(earphones, mouse, bluetooth)

MAN - network is a network that covers a larger geographic area by interconnecting a different LAN to form a larger network.

WAN - is a network that extends over a large geographical area such as states or countries.

Star, bus, ring

**OSI**

Open Systems Interconnection model - describes the functions of a networking or telecommunication system.

1. **Physical Layer**, represents the electrical and physical representation of the system. This can include everything from the cable type, radio frequency link

2. **The Data Link Layer** provides node-to-node data transfer (between two directly connected nodes)

3. **Network layer** works for the transmission of data from one host to the other located in different networks.

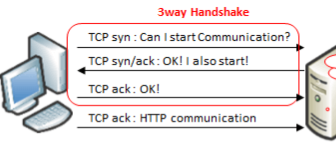
4. **Transport layer** is checking for errors, controlling flow, and sequencing.

5. **Session layer** responsible for opening and closing communication between the two devices.  It helps you to establish starting and terminating the connections between the local and remote application.. AP, sokets

6. **The presentation layer** formats or translates data for the application layer.

7. **Application layer** is closest to the end-user. It means OSI application layer allows users to interact with other software application.

**Transmission Control Protocol (TCP).** . It provides a reliable connection between applications; that is, a connection is established before data transmission begins. Retransmission of lost packets is possible in TCP. TCP doesn’t support Broadcasting. TCP is used by HTTP, HTTPs, FTP, SMTP and Telnet.



**User Datagram Protocol (UDP).**  It provides an unreliable datagram connection between applications but more faster than TCP. UDP is used by DNS, DHCP, TFTP, SNMP, RIP, and VoIP.

**UDP** is commonly **used** for applications that are “lossy” (can handle some packet loss), such as streaming audio and video

**DNS**

**A** (IPv4 Address Record) - Associates a domain name with an IPv4 host address

**AAAA** (IPv6 Address Record) - associates a domain name with an IPv6 host address (similar to an A record)

**CNAME** (Canonical Name Record) - used to redirect to another domain name

**MX** (Mail Exchange - mail exchanger) - refers to the mail server serving the domain

**NS** (Name Server) - the DNS server responsible for the domain

**TXT** - textual description of the domain. Often required to perform specific tasks (for example, confirming ownership of a domain when linking it to an email service)

**SOA(start of authority) -** The original zone record that specifies the server information that contains sample information about the domain name.

**iterative** - this is a method in which the DNS server acts as a client and queries other DNS servers starting with the root DNS servers and ending with the last, authoritative for the desired DNS zone.

**recursive** is a method in which the DNS server simply sends data from the client to another server to process the request and return the final data. (the other server can run recursively or similarly interactively)

**DHCP**

A**DHCP Server** is a network server that automatically provides and assigns IP addresses, default gateways and other network parameters to client devices

DHCP discover(broad), offer, request(broad), dhcp ack nak

**VPN**

gives you online privacy and anonymity by creating a private network from a public internet connection.

VPNs create a data tunnel between your local network and another computer in another location, which could be thousands of miles away. And it will be seem like we are in one room

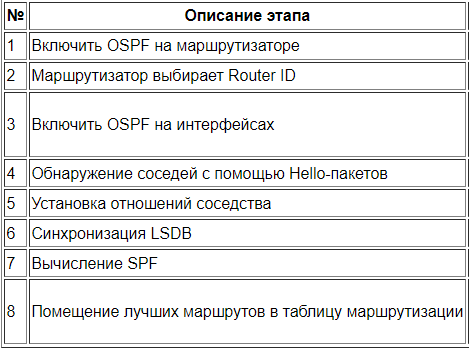
PPTP (pp tunneting protocol)- a point-to-point tunneling protocol that allows a computer to establish a secure connection with a server by creating a special tunnel in a standard, unsecured network

L2tp

**OSFP**

a dynamic routing protocol based on link-state technology and using Dijkstra's algorithm to find the shortest path.

multicast address 224.0.0.5 for normal communication and 224.0.0.6 for update to designated router(DR)/Backup Designated Router (BDR)



DR and BDR choosing

1. router with biggest priority will DR, second priority will BDR

2. by default the priorities is same on all routers = 1.

3. if Priority is same, Router with Highest Router ID will be DR

OSPF Packages:

**Hello** - packets that are used to discover neighbors, establish neighbor relations and monitor their availability (keepalive)

**DBD** - packages describing the content of LSDB

**LSR** - packets that request complete information about the LSA that is missing in the LSDB of the local router.

**LSU** - packets that transmit complete information that is contained in the LSA

**LSAck** - packets with which the receipt of other packets is confirmed

The router chooses the best route based on the smallest metric value

cost = reference bandwidth / link bandwidth

**BGP**

It is main protocol that works in Internet. BGP works between AS(autonomus system), AS is system of IP-networks and routes that managing by one or more operators.

Routers using BGP exchange info about network. With information about networks, some various network attributes will be trasmitted. And Via these attribues BPG will choose best route

**Open** - used to establish neighborhood relations and exchange basic parameters.

**Update** - Used to exchange routing information.

**Keepalive** - used to maintain neighbor relationships, to detect inactive neighbors.

**Notification** - Used when BGP errors occur. After sending the message, the session with the neighbor is terminated.

**Well-known mandatory** - all BGP routers must recognize these attributes. Must be present in all updates.

**Well-known discretionary** - All BGP routers must recognize these attributes. May be present in updates, but their presence is not required.

**Optional transitive** - May not be recognized by all BGP routers. If the router does not recognize the attribute, it marks the update as partial and sends it on to neighbors, keeping the unrecognized attribute.

**Optional non-transitive** - May not be recognized by all BGP routers. If the router does not recognize the attribute, then the attribute is ignored and discarded during transmission to neighbors.

* Well-known mandatory:
  + Autonomous system path
  + Next-hop
  + Origin
* Well-known discretionary:
  + Local preference
  + Atomic aggregate
* Optional transitive:
  + Aggregator
  + Communities
* Optional non-transitive:
  + Multi-exit discriminator (MED)
  + [Originator ID](http://xgu.ru/wiki/BGP_route_reflector)
  + [Cluster list](http://xgu.ru/wiki/BGP_route_reflector)

**AS path** - Describes what autonomous systems must be passed through to reach the destination network.

**Next-hop** – next ip address

**Origin** — указывает на то, каким образом был получен маршрут в обновлении.

**Local preference** - Tells routers inside the autonomous system how to get out of it

**Aggregator**

**Communities** – tag the routes

**Router id** - Prefer the path with the lowest BGP neighbor router ID. The router ID is based on the highest IP address

**EIGRP**

[Enhanced Interior Gateway Routing Protocol](https://www.geeksforgeeks.org/computer-network-features-enhanced-interior-gateway-routing-protocol-eigrp/) - is a dynamic routing Protocol which is used to find the best path between devices to deliver the packet

**RIP**

**Routing Information Protocol** is a dynamic routing protocol which uses hop count as a routing metric to find the best path between the source and the destination network.

Hop count is the number of routers occurring in between the source and destination network. The maximum hop count allowed for RIP is 15 and hop count of 16 is considered as network unreachable. There are three vesions of routing information protocol – **RIP Version1**, **RIP Version2** and **RIPng**. **Rip1** - Classful routing protocol , **Rip2** - Classless protocol, supports classful, Supports authentication of RIPv2 update messages.

In Distance-Vector protocols, the router learns information about the routes directly connected to the same network segment. That is, a router has information about the topology only within the boundaries of its neighboring routers and has no idea how the topology is arranged behind these routers, focusing only on metrics. In Link-state protocols, each router must have a hard time knowing the best routes to all remote networks.

**NAT**

Static NAT - one-to-one address mapping between local and global addresses;

Dynamic NAT - Many-to-many address mapping between local and global addresses;

Port Address Translation (PAT)- translates multiple private addresses to one or more public addresses. This method is also known as NAT Overload;

**VRRP**

Virtual router redudancy protocol - creating a virtual router based on physical routers or L3 interfaces of switches. For example we have 2 L3 switches, one master switch will be selected, through which all traffic will go - and the rest are classified as backup L3gfswitch and are waiting for their turn.

**VLAN**

is logical grouping of devices in the same [broadcast domain](https://study-ccna.com/collision-broadcast-domain/).

VLAN 0 and 4095 – Reserved, which cannot be used

VLAN 1 – This is default VLAN of switches. You cannot delete or edit, but it can be used

VLAN 2-1001 – It is normal VLAN range. Can be edited,created, deleted

VLAN 1002-1005 – cisco defaults for token rings and FDDI.

VLAN 1006-4094 – It is an extended range of VLANs.

A VLAN allows several networks to work virtually as one LAN. One of the most beneficial elements of a VLAN is that it removes latency in the network, which saves network resources and increases network efficiency

Flexible division of devices into groups

Typically, one VLAN is one subnet. Computers on different VLANs are isolated from each other.

Reducing broadcast traffic on the network

Each VLAN represents a separate broadcast domain. Broadcast traffic will not be broadcast between different VLANs

Increased network security and manageability

In a network divided into virtual subnets, it is convenient to apply policies and security rules for each VLAN. The policy will be applied to the entire subnet, not to an individual device.

**ACL**

it is a set of text expressions that allow or deny traffic.

Standart – can check only source address.

Extension – can filter the traffic by source and destination address, by protocols, by ports, etc..

Both standrart and Extension can be Numbered and Named.

Numbered in Standart mode can be from 1-99 and 1300-1999

Numbered in Extension mode can be from 100-199 and 2000-2699

Route **route add 10.41.0.0 mask 255.255.0.0 10.27.0.1**

netstat –tulpn (netstat --tcp --udp --listening --program –numeric)

Pathping, WinMTR

# Linux

1. BIOS or UEFI

2. MBR or GPT

3. GRUB

4. Linux Kernel

5. Init

**BIOS** is short for [Basic Input-Output system](https://www.howtogeek.com/179789/htg-explains-what-is-bios-and-when-should-i-use-it/). It’s low-level software that resides in a chip on your computer’s motherboard. The BIOS loads when your computer starts up, and the BIOS is responsible for waking up your computer’s hardware components, ensures they’re functioning properly.

**MBR** (Master Boot Record) and **GPT** (GUID Partition Table) are two different ways of storing partition information on a hard disk. This information includes both information about the principle of operation of drives and determines which partition will become bootable for the system.

**UEFI+GPT**. It was created because the BIOS has a number of disadvantages. These include working in 16-bit mode, serial device checking and booting only through the MBR. UEFI works in 32- and 64-bit modes, contains a GPT partition table, can have a beautiful GUI, EFI Shell and mouse control and equipment cheking more fster

**GRUB** is the main boot loader for most Linux distributions.The boot loader prepares the system to load the operating system kernel.

**Linux Kernel**. Next, the Linux kernel is loaded. The kernel is located in the boot folder called vmlinuz. Next, the Linux kernel is loaded. The kernel lies in the boot folder called vmlinuz

The kernel has booted up and is working. But this is not enough to work with the operating system. In order to be able to do something in Linux, the system initialization (and service management subsystem) gets control.

**Systemd** is the most popular init system. Starts the services. For example, which control sound devices, multimedia keys, launch the graphical shell

|  |  |  |
| --- | --- | --- |
| 0 | Halt | Shuts down system |
| 1 | Single-User Mode | Does not configure network interfaces, start daemons, or allow non-root logins |
| 2 | Multi-User Mode | Does not configure network interfaces or start daemons. |
| 3 | Multi-User Mode with Networking | Starts the system normally. |
| 4 | Undefined | Not used/User-definable |
| 5 | X11 | As runlevel 3 + display manager(X) |
| 6 | Reboot | Reboots the system |

each file has three categories of users. Owner,Group,Others

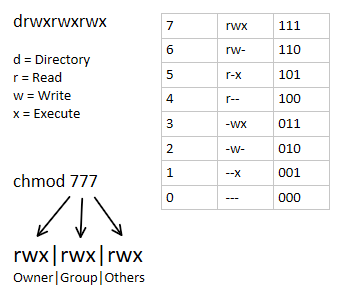
In order to allow ordinary users to run programs on behalf of the superuser without knowing his password, such a thing as SUID and SGID bits was invented.

**SUID** – user id who execute app will replaced to owner of file

**SGID** - this flag works in a similar way, the only difference is that the user is considered a member of the group the file is associated with

**Sticky-bit** - This bit is also used to create shared folders. If it is installed, then users can only create, read and execute files, but cannot delete files owned by other users.

**chmod ugo+rwx file1**



**Symlink and hardlink**

The inode is a database that describes the file/directory attributes such as metadata and the physical location on the hard drive

hardlink cannot point to a file in another file system (since an inode can only belong to one file system), but a symbolic link can.

Set – env variables

The **zombie** process is also known as the "dying" process. It is a terminating process that exited through the exit system call, but still has an entry in the process table.

Swap Space is the additional space that temporarily holds programs when the RAM does not have enough space to hold the programs.

Cat /proc/meminfo

**tar** -cvf archive.tar.gz /path/to/files

c - create an archive in linux

v - show detailed information about the process

f - file for recording the archive

**tar** -xvf archive.tar.gz

x - extract files from the archive

.**BASHRC -** This is a user-specific variable file. Loaded every time the user opens a new terminal

**.BASH\_PROFILE -** These variables take effect every time a user connects remotely via SSH. If this file is missing, the system will search for it .bash\_login or .profile.

**/ETC/ENVIRONMENT -** This file is for creating, editing, and deleting any environment variables at the system level. The environment variables created in this file are available for the entire system, for each user, and even when connected remotely.

**/ETC/BASH.BASHRC** - System bashrc. This file is executed for each user, each time they create a new terminal session. This only works for local users, when connected via the Internet, such variables will not be visible.

**ETC/PROFILE** - The system profile file. All variables from this file are available to any user in the system only if they are logged in remotely. But they will not be available when you create a local terminal session, that is, if you just open the terminal.

Route -n

Netstat -rn

Ip route show

Cat /proc/net/route

Route add -net 192.168.0.0/16 gw 10.0.0.1

Route add -net 192.168.0.0/16 dev eth1

Route add -host 192.168.1.3 gw 172.16.1.1

Ip route add 192.168.0.0/16 via 10.0.0.1

# Python

A function is a block of code which is executed only when it is called

There are mutable and immutable built-in types.

The mutable ones include:

* List
* Sets. A set is a mutable, unordered data type. The set always contains only unique elements. Set(list1)
* Dictionaries

The immutable types include:

* Strings
* Tuples. List but unchangeble List. The tuple only has read permissions. This can be a protection against accidental changes.
* Numbers. (Int, float, complex)

The difference between list and tuple is that list is mutable while tuple is not. Tuple can be hashed, for example., as a key for dictionaries.

make\_sum = lambda x,y: x+y

make\_sum(3,8)

>>> a = [1, 2, 3]

>>> a

[1, 2, 3]

>>> a.reverse()

>>> a

[3, 2, 1]

>>> L = [0,10,20,40]

>>> L[::-1]

[40, 20, 10, 0]

# Git

version control system. IT tracking changes of files

# Docker

Difference between VM and container

Docker Engine :

1. Docker Server: **Container Runtime**: pulling, managing images and containers, **Volumes**: persisting data, **Network**: configuring network for container comunication
2. API: interacting with Docker SErver
3. CLI: execute docker commands

Docker-compose build

docker container run -d -p 80:80 --name proxy nginx

docker container run -d -p 80:80 --name proxy -m 10485760 --memory-reservation 5242880 nginx

docker container exec -it <CONTAINER\_ID> /bin/sh

docker container kill my\_container #forcely

docker container kill $(docker ps -q) #kill all containers

docker container inspect <Container\_name>

FROM ubuntu:18.04

RUN apt -y update

RUN apt -y install httpd

RUN echo “Hello World from Docker!” > /var/www/html/index.html

CMD [“/usr/sbin/apache2ctl”, “-DFOREGROUND”]

EXPOSE 80

> docker image build -t abu\_bandit/my\_image:tag .

RUN instruction allows you to install your application and packages requited for it. It executes any commands on top of the current image and creates a new layer by committing the results

CMD instruction allows you to set a *default* command, which will be executed only when you run container without specifying a command. If the Dockerfile contains more than one CMD statement, all except the last CMD statement is ignored.

# Kubernetes

is an [open source](https://www.redhat.com/en/topics/open-source/what-is-open-source) container orchestration platform provide automation of many the manual processes with containers. It help deploying, managing, and scaling containerized applications.

**Components**

***Master -*** *manage kuber cluster*

Api server – it is front-end or client part kuber controller that orchestrate all operations on server.

Scheduler – help to schedule the pods on the different nodes and depends to computer resources.

Etcd - stores information about cluster

Controller - is a control plane component that runs controllers (replica controller, node controller, account, endpoint)

***Worker***

Kubectl – listens to instructions from kube-api server

Kube-proxy enable communications between services in between cluster

Container runtime engine

Pods - are the smallest deployable units of computing that you can create and manage in Kubernetes.

Service – service is responsible for exposing an interface to those pods, which enables network access

ClusterIP - Exposes the Service on an internal IP in the cluster

NodePort - Exposes port

LoadBalancer - Creates an external load balancer

ExternalName - DNS CNAME record

Deployment - is a method of launching a pod with containerized applications and ensuring that the necessary number of replicas is always running on the cluster.

Ingress - is an object that allows access to your Kubernetes services from outside the Kubernetes cluster

# Ansible

Is a configuration management tool that provides automation for application deployment, updates on workstations and servers, cloud provisioning, configuration management

An inventory file is a list to identify your managed nodes

[staging\_servers]

aws\_ubuntu1 ansible\_host=aws2

[prod\_servers]

aws\_ubuntu2 ansible\_host=aws1

Ad hoc - are commands which can be run individually to perform quick functions. Ad-hoc to send a single task on one or more managed nodes

ansible all -i production.ini -m ping

Playbooks are the files where Ansible code is written. Playbooks are written in YAML format.  They are like a to-do list for Ansible that contains a list of tasks (instruction)

- name: Install and run nginx web-server

hosts: all

become: yes

tasks:

- name: install nginx

apt: name=nginx state=latest

- name: start nginx

service: name=nginx state=started enabled=yes

**name**. This tag specifies the name of the Ansible playbook

**hosts**. This tag specifies hosts or host group. It tells Ansible on which hosts to run the listed tasks

**vars.** Vars tag lets you define the variables which you can use in your playbook

**tasks.**Tasks are a list of actions one needs to perform

**handlers** are special tasks that only get executed when triggered

**Modules** - These are small programs that perform certain work on the server

group\_vars - The file contains a set of variables, such as the database user name and password.

Role it is Framework that divide a playbook into multiple files, each file will store in separated special directory. Roles have its own directory structure that create automaticalli with command ansible-galaxy init