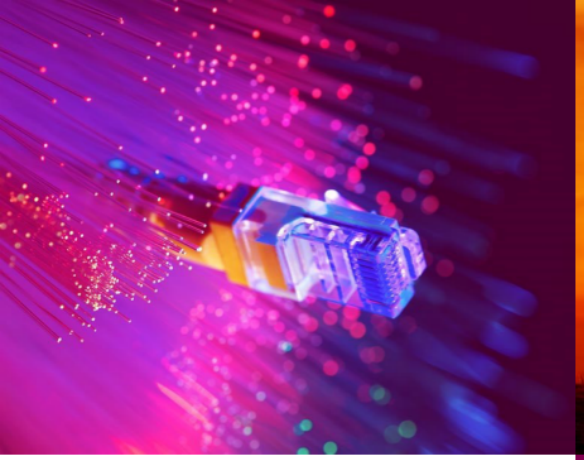


Pouta Cloud Course

Jemal Tahir, Cloud System Specialist

Alvaro Gonzalez, Cloud System Specialist

CSC – Finnish expertise in ICT for research, education and public administration

Notice

We will record this presentation

- This is to explore the idea of publishing an online video of this course
- We will cut out from the recording the Q&A sections (for GDPR and privacy reasons).
- So feel free to ask questions any time

If something **makes no sense**, you want to make a **question** or **correction**, **Please interrupt** and make your comment

Expectations

- How familiar are you with Cloud and Pouta Cloud?
- What are you expecting to learn from this course?

https://www.menti.com/al9z1gu7kw1d

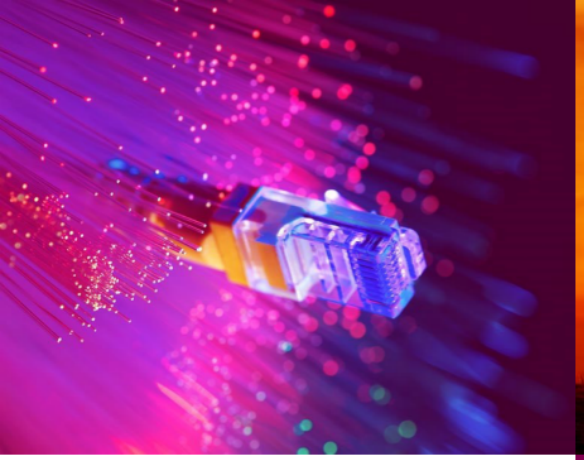
Schedule

When	What	
9:00-10:30	Introduction	Cloud computing basics and landscape
		OpenStack
		Pouta virtual resource basics
10:30-10:45	Coffee break	☕
10:45-12:00	Exercises	A
12:00-13:00	Lunch	🍽️
13:00-14:30	Theory	cPouta vs. ePouta
		Storage
		Pouta management
		Practical information, installing software
14:30-14:45	Coffee break	☕
14:45-15:00	Exercises	B and C
15:00-15:15	Closing	Documentation and contact info
15:00-16:00	Exercises	Extra time

Introduction

"This course gives you a practical introduction for using CSC's cloud services Pouta"

- The **Pouta Cloud** is CSC's Infrastructure as a Service (IaaS) offering. Based on OpenStack.
  - Allows running Virtual Machines (VMs) on CSC's Data Center infrastructure. Grants users **full control** over the OS, middleware and run time environments. On the flip side, users are responsible **manage** and **secure** their VMs.
- Provides an IaaS cloud environment for your **sensitive data** processing (ePouta).



Cloud Basics

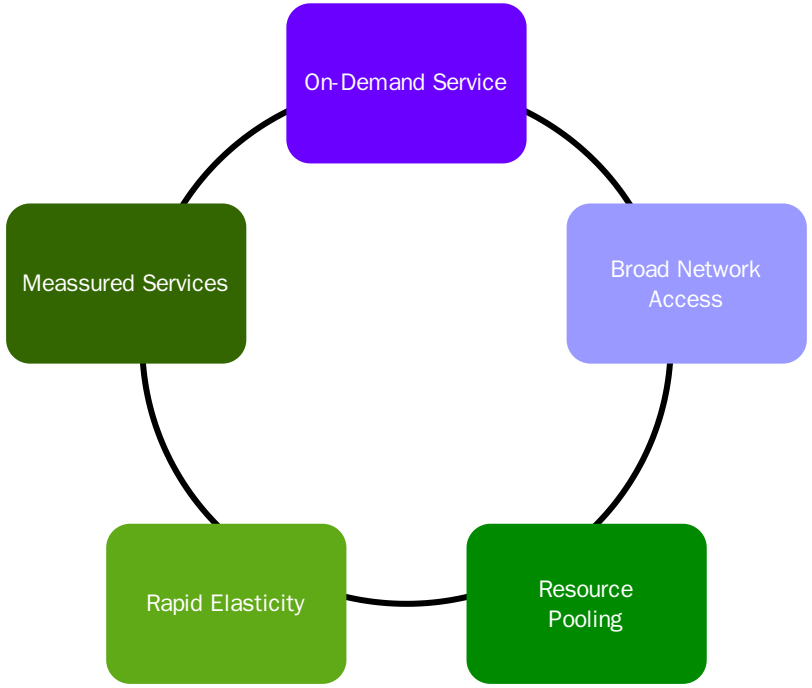
Cloud computing

"A model for enabling convenient, on-demand network access to a shared pool of configurable computing resources. Resources like networks, servers, storage, applications, and services. They can be rapidly provisioned and released with minimal management effort or service provider interaction" [1]

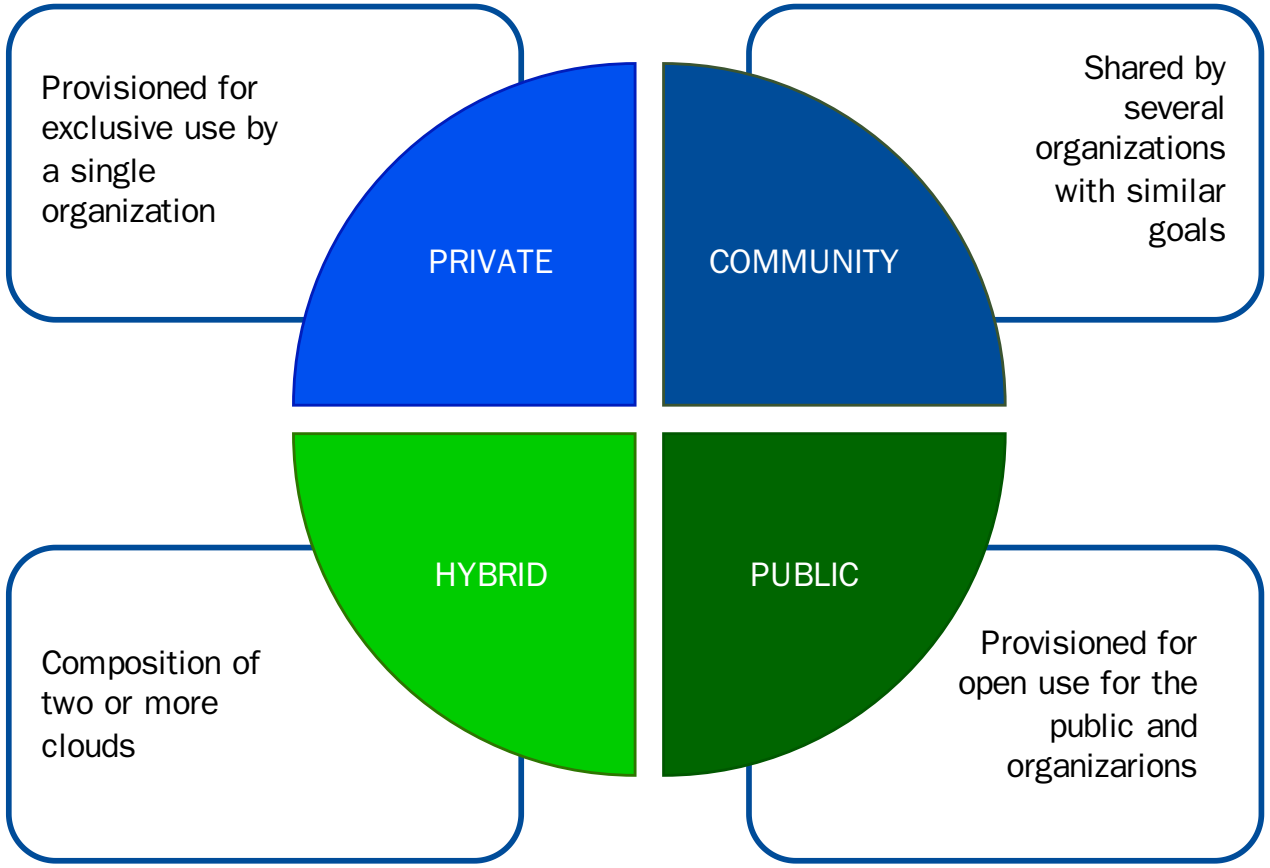
[1] https://en.wikipedia.org/wiki/Cloud_computing

Cloud computing characteristics

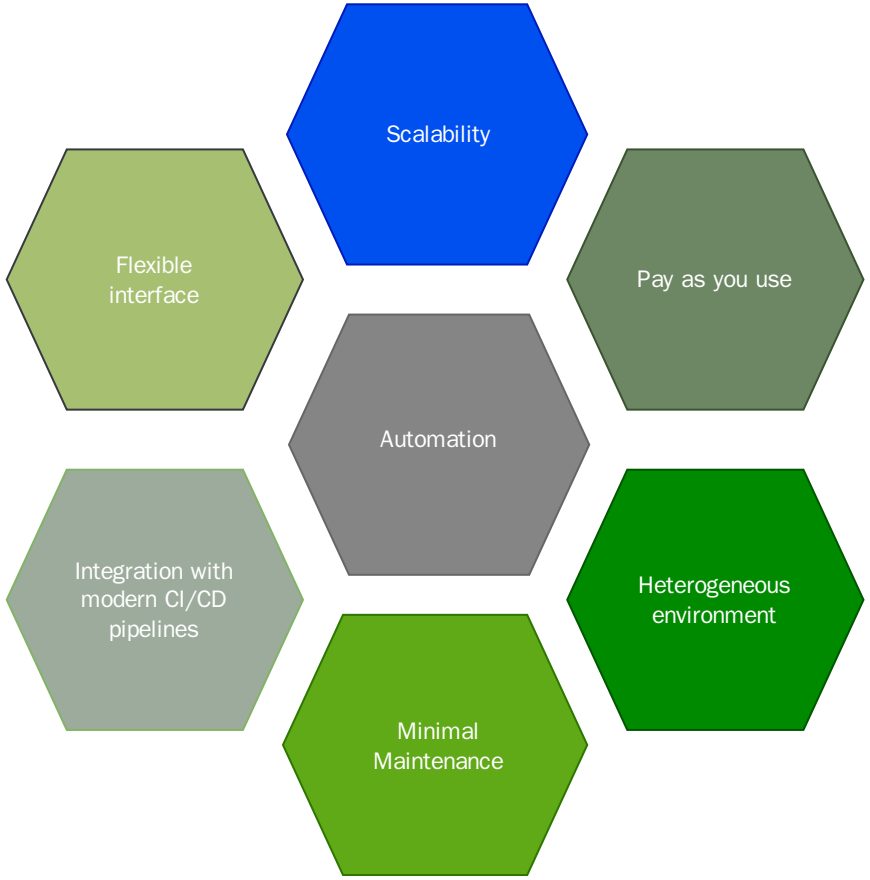
- Cloud Computing has 5 essential characterstics
  - On-demand self-service
  - Broad network access
  - Resource pooling
  - Rapid elasticity
  - Measured Service
- The trend so far
  - Colocation: rented space in shared facility, user-configured, user-managed
  - Virtualization: user-configured, provider-managed
  - Containerization: automatically managed and configured infrastructure



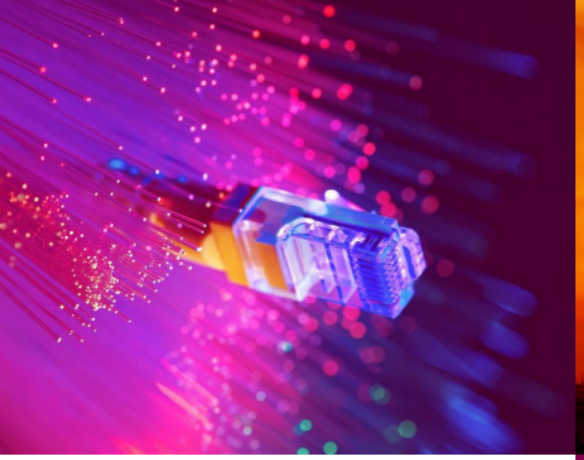
Cloud deployment models



Cloud computing wishlist

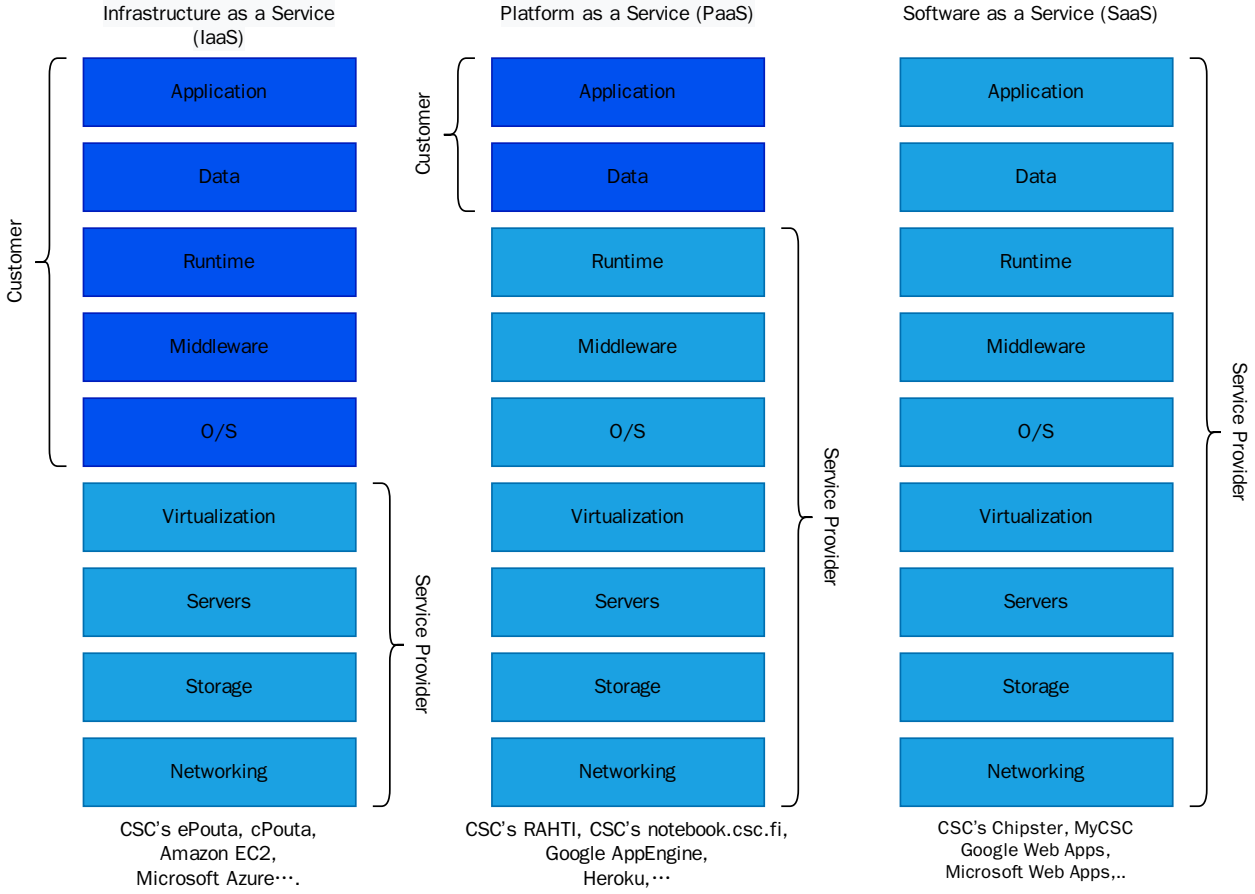


Cloud computing challenges



Cloud computing Landscape

Cloud computing: "As A Service"

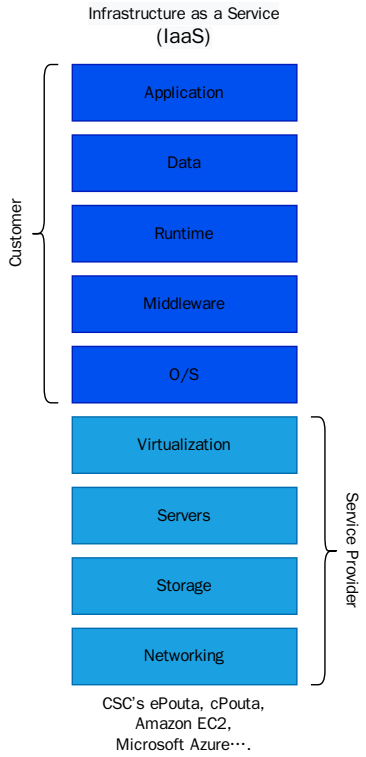


Cloud Service Landscape: IaaS

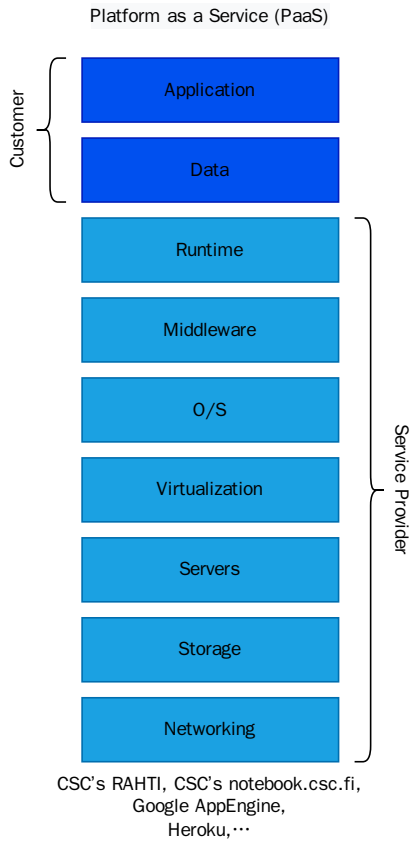
- Infrastructure as a Service:

"The network and hardware **infrastructure** is offered to you, the user, so you can just worry about running your Operating system of choice, the Software you need to run and nothing else".

- No worries about: Hardware issues
- Flexibility in scaling up and down
- Control via API
- Software defined network (**SDN**)



Cloud Service Landscape: PaaS



- Platform as a Service:

  "The network and hardware infrastructure, plus the Operating System and thge middleware is offered as a **platform** to you, the user, so you can just worry about installing, configuring and using the Software and nothing else".

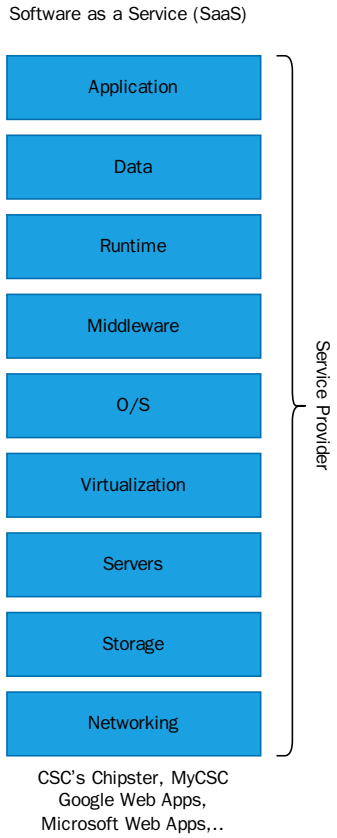
  - No worries about: Hardware issues, Operation systems patches, etc.
  - Security: Containers allow software from independent teams of people to run isolated, even though they run in the same hardware.
  - Control via API
  - QoS: Orchestration services provide assured resources

Cloud Service Landscape: SaaS

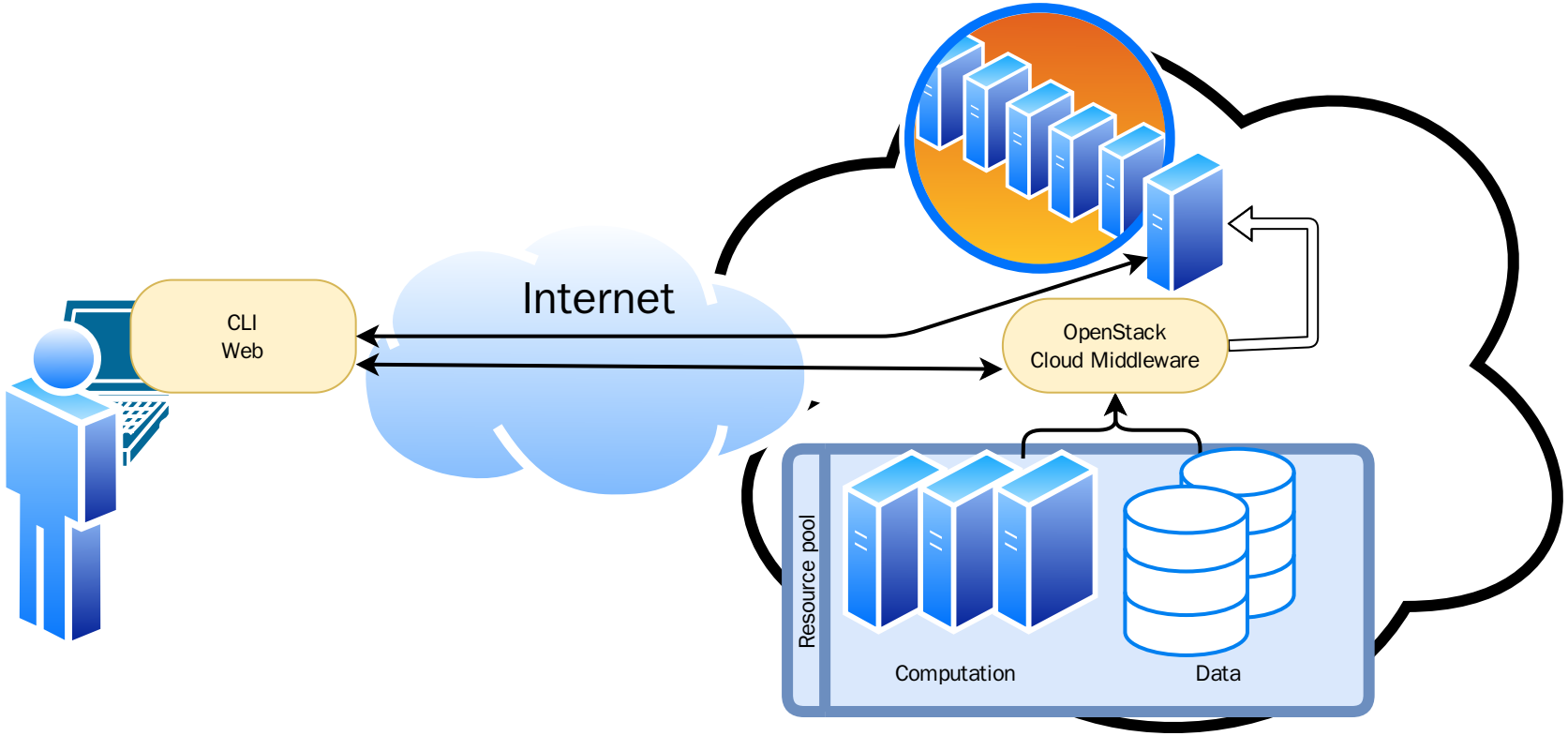
- Software as a Service

  "Software is offered to you, the user, so you can just worry about using the Software and nothing else."

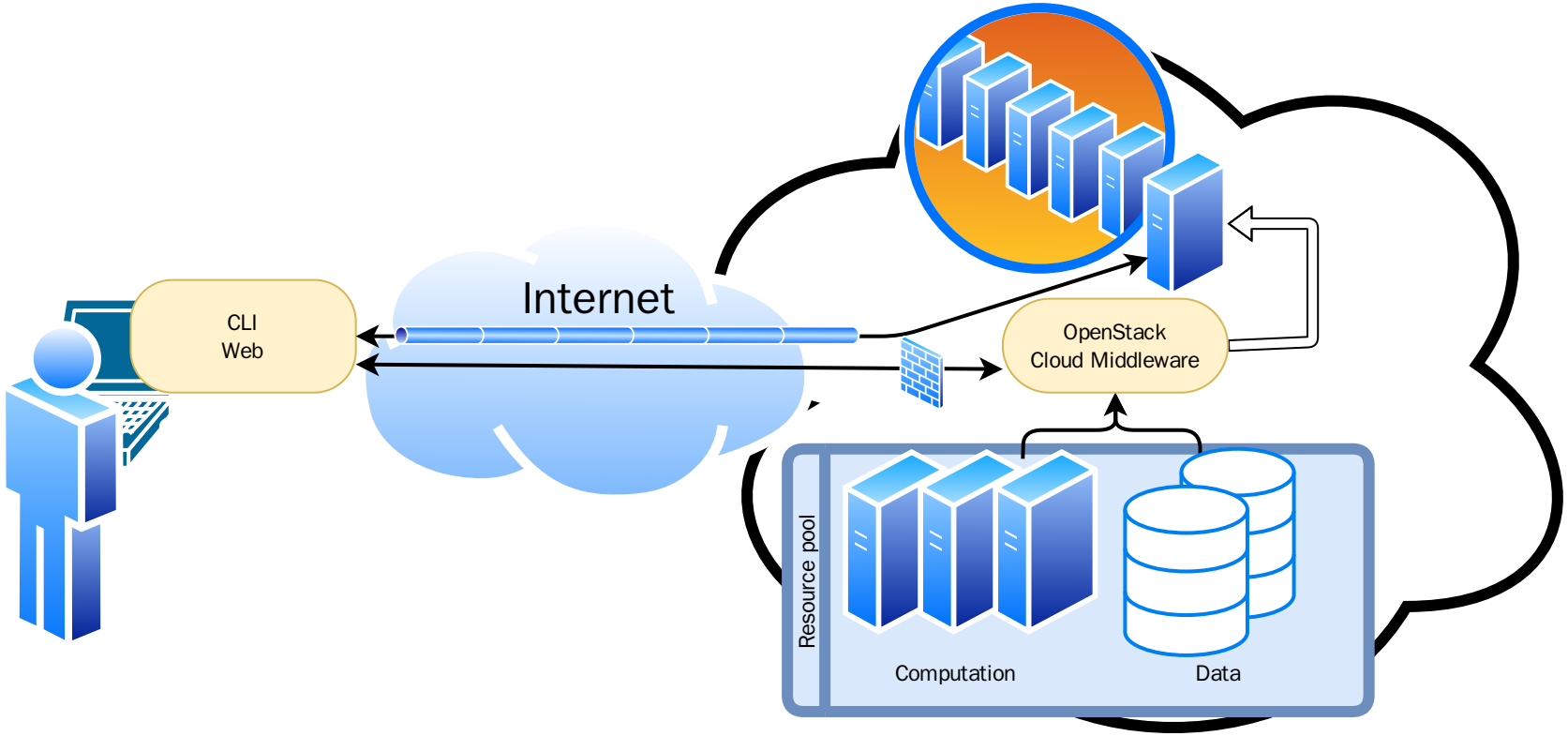
  - No worries, only use the software.
  - Examples:
    - myCSC
    - Web mail
    - Web Office platforms
    - ...



Typical IaaS Cloud Setup



Secure Cloud Setup



"A private data center"

CSC's Cloud Computing Services

cPouta

Community IaaS Cloud

General purpose

Services accessible over internet
Powered by OpenStack
ISO27001 Certified
In Production since 2013

Web UI, CLI & REST APIs supported

ePouta

Community IaaS Cloud

Sensitive data

Accessible only from customer network
Powered by OpenStack
ISO27001 Certified
In Production since 2013

Web UI, CLI & REST APIs supported

Rahti

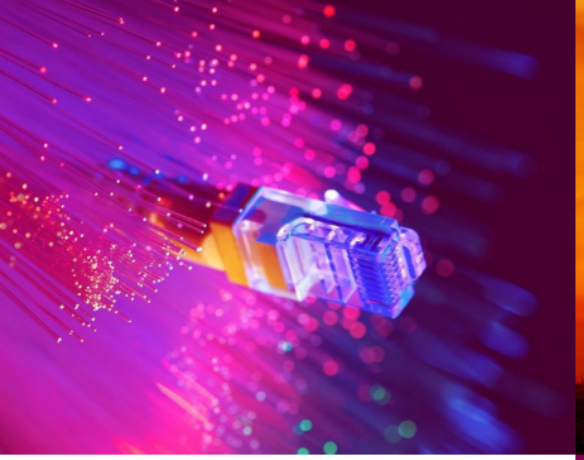
Community PaaS Cloud leveraging containers

General purpose HTTP(s) applications

Services accessible over internet
Powered by OpenShift OKD

In Open Beta*

Web UI, CLI & REST APIs supported

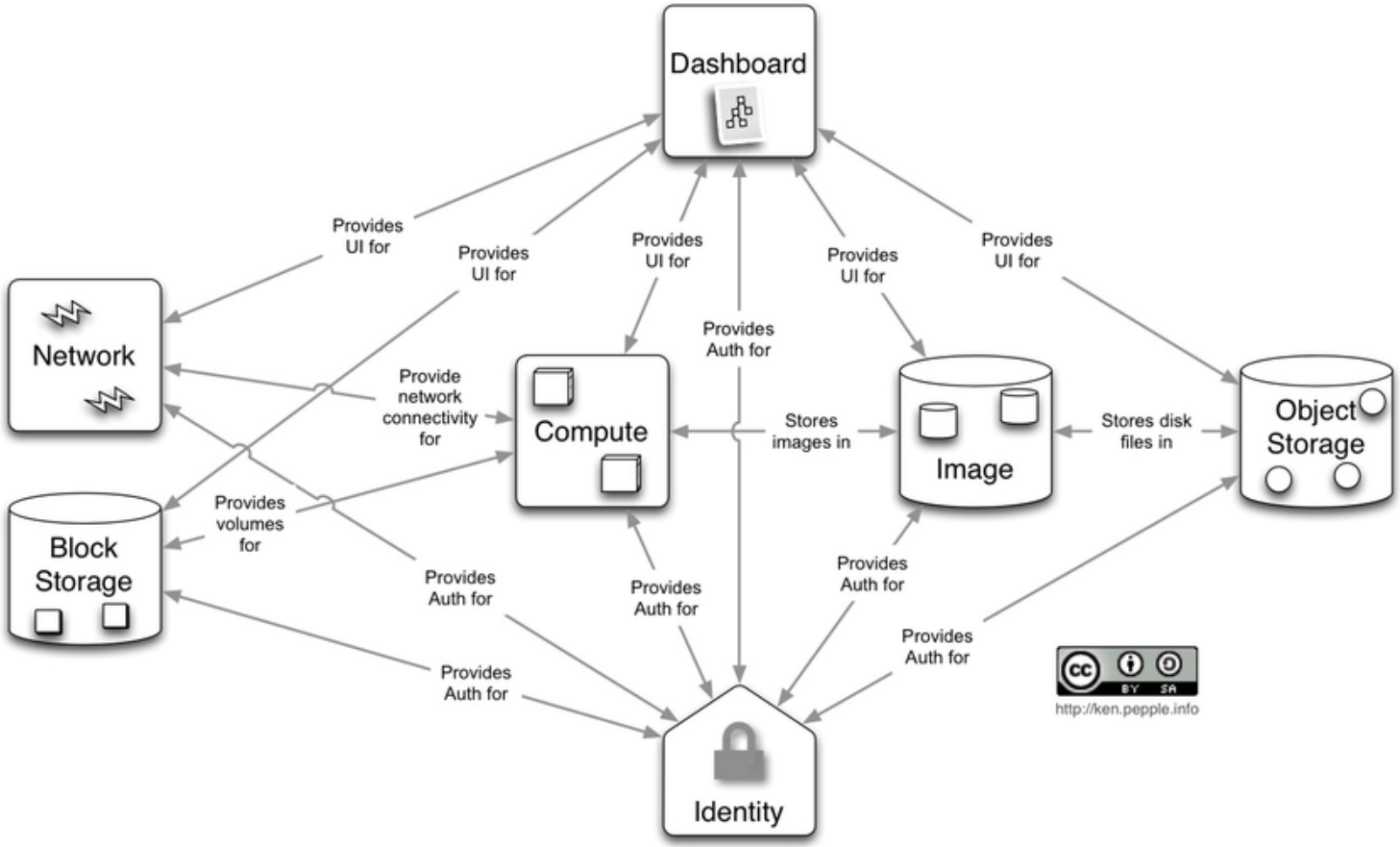


Basics of VMs

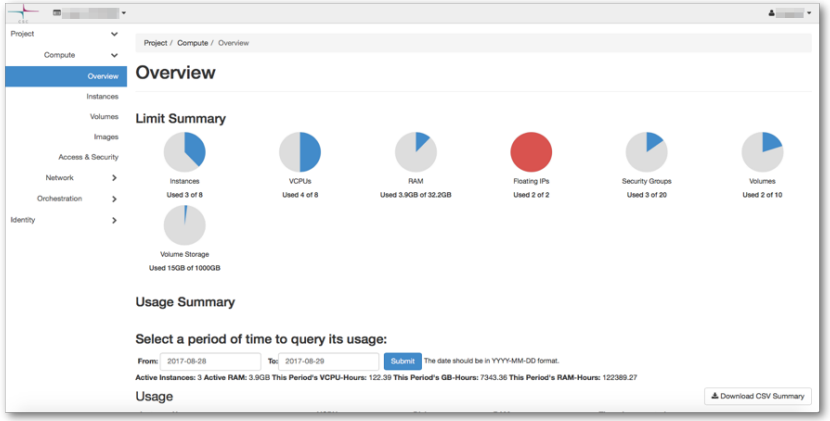
OpenStack software

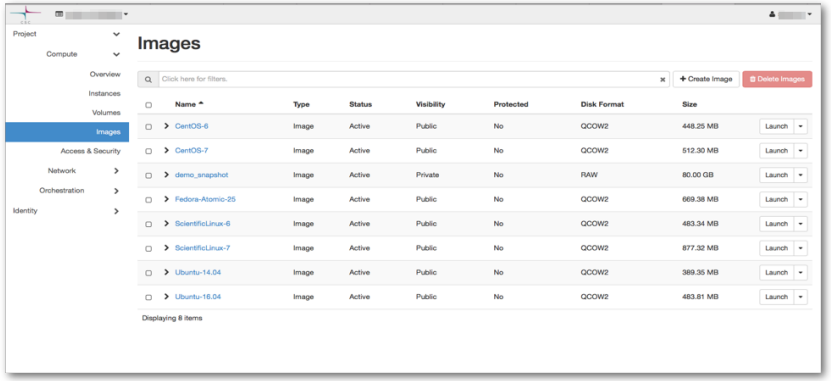
- CSC's cPouta/ePouta cloud services are powered by OpenStack.
  - Current OpenStack version used by Pouta services is Queens
- OpenStack is a cloud software that allows end user to create and use their VM instances, networks and storage.
- Virtualization is a technology that allows the creation of virtual computer resources such as CPU, storage, network, etc.
- Fast moving open source project with backing from industrial giants like: AT&T, Red Hat, IBM, Intel, HP etc.
- Flexible architecture which may support different types of scales.
- Used by many organizations from research institutes to service/content providers.
- Large customer base augments better availability of expertise, support and chances of continuity.
- Supports Web UI, CLI and REST Interfaces

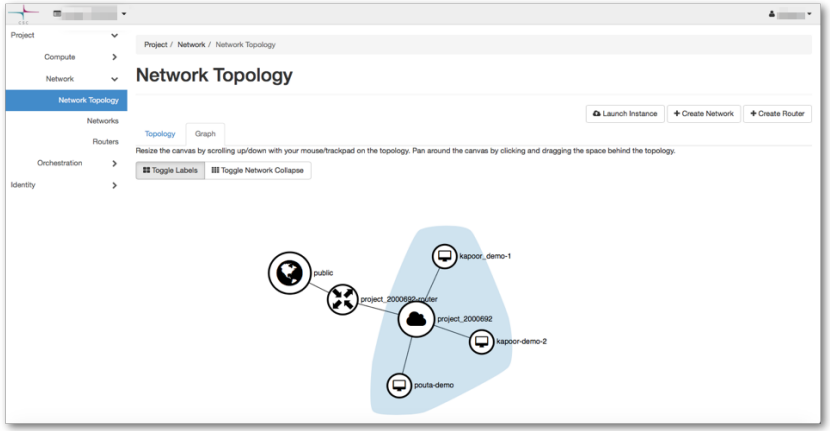
OpenStack Architecture



OpenStack WebUI





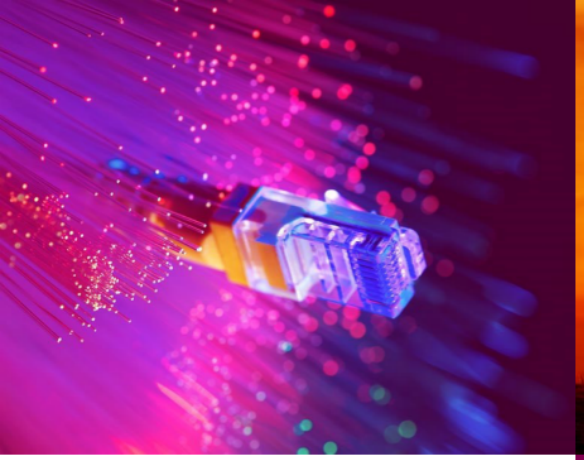


OpenStack CLI

```
$ openstack flavor list
+--------------------------------------+------------------+--------+------+-----------+-------+-----------+
| ID                                   | Name             |    RAM | Disk | Ephemeral | VCPUs | Is Public |
+--------------------------------------+------------------+--------+------+-----------+-------+-----------+
| 0143b0d1-4788-4d1f-aa04-4473e4a7c2a6 | standard.tiny    |   1000 |   80 |         0 |     1 | True      |
| 053c4852-dd1e-42dc-947a-fe4263548fa9 | hpc-gen2.48core  | 240000 |   80 |         0 |    48 | True      |
| 110eb004-f7cc-474b-8158-14bb244cb05e | hpc-gen2.24core  | 120000 |   80 |         0 |    24 | True      |
| 1792db39-f38e-43ba-ae95-96b7549b4f84 | standard.xlarge  |  16000 |   80 |         0 |     6 | True      |
| 242de907-89dc-44e9-ac45-65f27da15e80 | hpc.4.20core     |  88000 |   80 |         0 |    20 | True      |
| 3d1f8655-f0c4-4afb-98fa-e35c764da03d | gpu.1.4gpu       | 480000 |   80 |         0 |    56 | True      |
| 41ec2177-604b-492c-8f19-f2d7c2bc8c07 | io.70GB          |  10000 |   20 |        70 |     2 | True      |
| 42e7cc2b-5c46-4a76-bc97-5bc01498fdf5 | hpc.4.10core     |  44000 |   80 |         0 |    10 | True      |
| 4eb92da8-a99e-41b8-a8a7-723ebaa29732 | gpu.1.1gpu       | 120000 |   80 |         0 |    14 | True      |
| 7a1d78bf-4336-4d88-87c3-8edfd5a045d3 | hpc.4.40core     | 176000 |   80 |         0 |    40 | True      |
| a1c8d559-1b5d-47c1-b688-9606468e3419 | gpu.1.2gpu       | 240000 |   80 |         0 |    28 | True      |
| aaf7d3b1-f11b-4cfa-86d9-c6540e5efd06 | standard.xxlarge |  32000 |   80 |         0 |     8 | True      |
| af9fa76e-818a-421e-9142-0341e7818d90 | io.340GB         |  40000 |   20 |       340 |     8 | True      |
| b2e91c81-d9a8-4636-b409-aa3c32e2939e | hpc.4.80core     | 352000 |   80 |         0 |    80 | True      |
| b5df0476-cae8-424d-8d3f-f6bd4f354c22 | hpc.4.5core      |  22000 |   80 |         0 |     5 | True      |
| c1da3536-f22d-426e-bc14-ef994f1bfaa7 | io.700GB         |  80000 |   20 |       700 |    16 | True      |
| c5ffaed0-6707-4a99-9498-9ef6d34c8add | io.160GB         |  20000 |   20 |       160 |     4 | True      |
| d4a2cb9c-99da-4e0f-82d7-3313cca2b2c2 | standard.small   |   2000 |   80 |         0 |     2 | True      |
| e2b74386-3649-40f2-a46c-98390fbf0cdd | standard.3xlarge |  64000 |   80 |         0 |     8 | True      |
| e7b3364e-f70c-4e3b-8e5a-fa249759d14c | standard.large   |   8000 |   80 |         0 |     4 | True      |
| f363d088-4967-48ff-bc80-86c0d05ff418 | standard.medium  |   4000 |   80 |         0 |     3 | True      |
+--------------------------------------+------------------+--------+------+-----------+-------+-----------+
```

```
$ openstack keypair show [redacted]
+-------------+-----------------------------------------------+
| Field       | Value                                         |
+-------------+-----------------------------------------------+
| created_at  | 2020-02-27T12:39:01.000000                    |
| deleted     | False                                         |
| deleted_at  | None                                          |
| fingerprint | 71:6e:2e:4c:80:c7:75:0c:35:d0:35:9c:70:a6:88:03 |
| id          | 822736                                        |
| name        | [redacted]                                    |
| updated_at  | None                                          |
| user_id     | [redacted]                                    |
+-------------+-----------------------------------------------+
```

```
$ openstack image list
+--------------------------------------+----------------------------+--------+
| ID                                   | Name                       | Status |
+--------------------------------------+----------------------------+--------+
| 8293dd1c-7b35-41a0-9f94-17c35c01e890 | CentOS-7                   | active |
| 062458e2-88ec-4de4-9a30-e774ba63d541 | CentOS-7-Cuda              | active |
| 158be574-9ff6-410c-a344-2e281b5cad7f | CentOS-8                   | active |
| 2ca237c5-bd0a-4469-ae9f-20878dd288a9 | Fedora Cloud Base 31       | active |
| 46eb2e6c-0677-4211-883f-ec4b4077b161 | ScientificLinux-7          | active |
| 13a458e5-7a4f-484d-893c-99f13edfb3e8 | Ubuntu-16.04               | active |
| 15eac02e-c2f3-417f-b4e7-e7b4422ef12b | Ubuntu-16.04-Cuda          | active |
| 1dfbbc5e-91cb-428a-a1db-ae9d1767d2eb | Ubuntu-18.04               | active |
| 264a384f-0d7a-4476-98f1-630b9a5b39e8 | Ubuntu-20.04               | active |
+--------------------------------------+----------------------------+--------+
```



Pouta

Resources you get from Pouta Clouds

Compute
- VMs
- Oversubscribed or dedicated CPUs
- GPUs

Storage
- Volume Storage
- Object Storage [1]

Private VLAN
- 10 GbE or 40 GbE

IPv4
- With or Without NAT [2]

Images
- With Latest Security patches

API
- Full programmability or your resources

[1] S3/Switf API access. [2] Network Address Translation

cPouta

- General purpose service.
- Serving cloud computing needs of Finnish research institutes and universities since 2013.
- VMs and Control plane can be accessed via **public** internet.
- Customers may decide access to VMs by creating firewall rules at OpenStack level known as "Security Groups".
- Could be used for hosting:
  - Custom services such as Web servers, File servers, load balancer etc.,
  - Scientific applications,
  - Course computer resources for students,
  - Research Data Sharing etc.

ePouta ☁️🔒

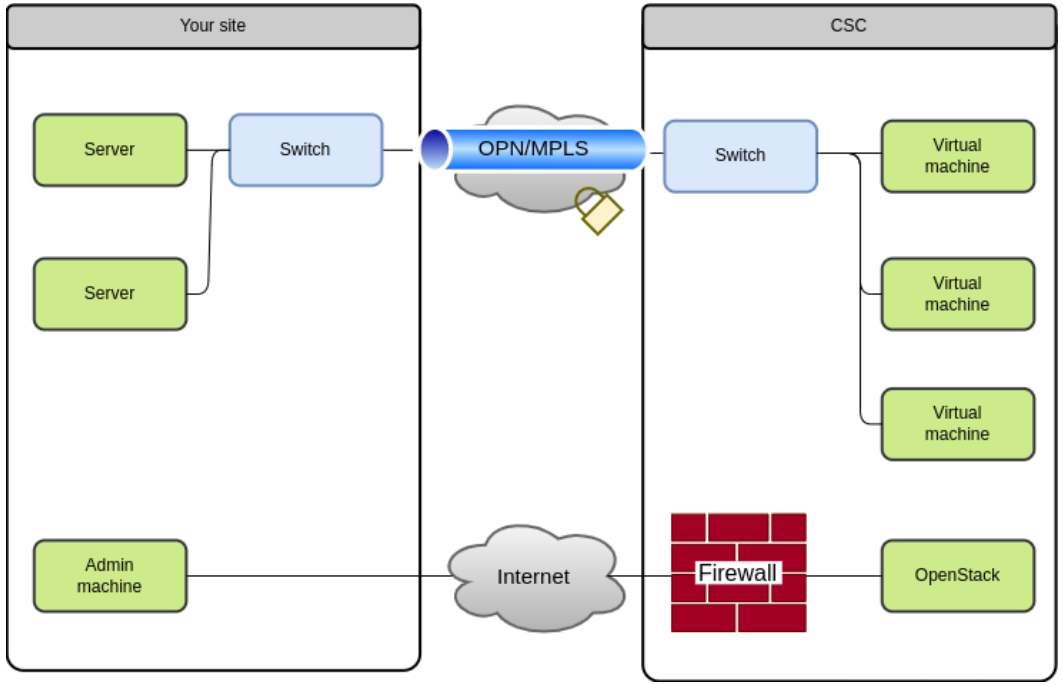
- **Sensitive Data** related services.
- Complete Isolation of VMs from rest of the world and other ePouta customers.
  - A (virtual) **private** data center.
- VMs accessible only from customer network.
- Virtual **Private** Cloud: Optical Private Network (OPN) or MPLS[1] VPN connection between the end customer and ePouta VM instances.
- Could be used for hosting:
  - Scientific applications dealing with **sensitive data**,
  - **Sensitive Data** Sharing, Archiving etc.

[1] https://en.wikipedia.org/wiki/Multiprotocol_Label_Switching

cPouta/ePouta

	cPouta public Cloud	ePouta private Cloud
Usage	General purpose	Sensitive Data
Network connection	Public Internet	Private OPN/MPLS
ISO27001 certification	✔	✔
VAHTI 2010 certification	✔	✔
Firewall, LB, VM installation, VM auto-recovery, Backups	Self-service	Self-service
Supported Operating Systems	All (commercial OSs require a license)	All (commercial OSs require a license)
OpenStack Version	Queens	Queens
GPU	✔	✔
Service availability target	99%	99%
Allas (S3/Swift) Object Storage	✔	❌

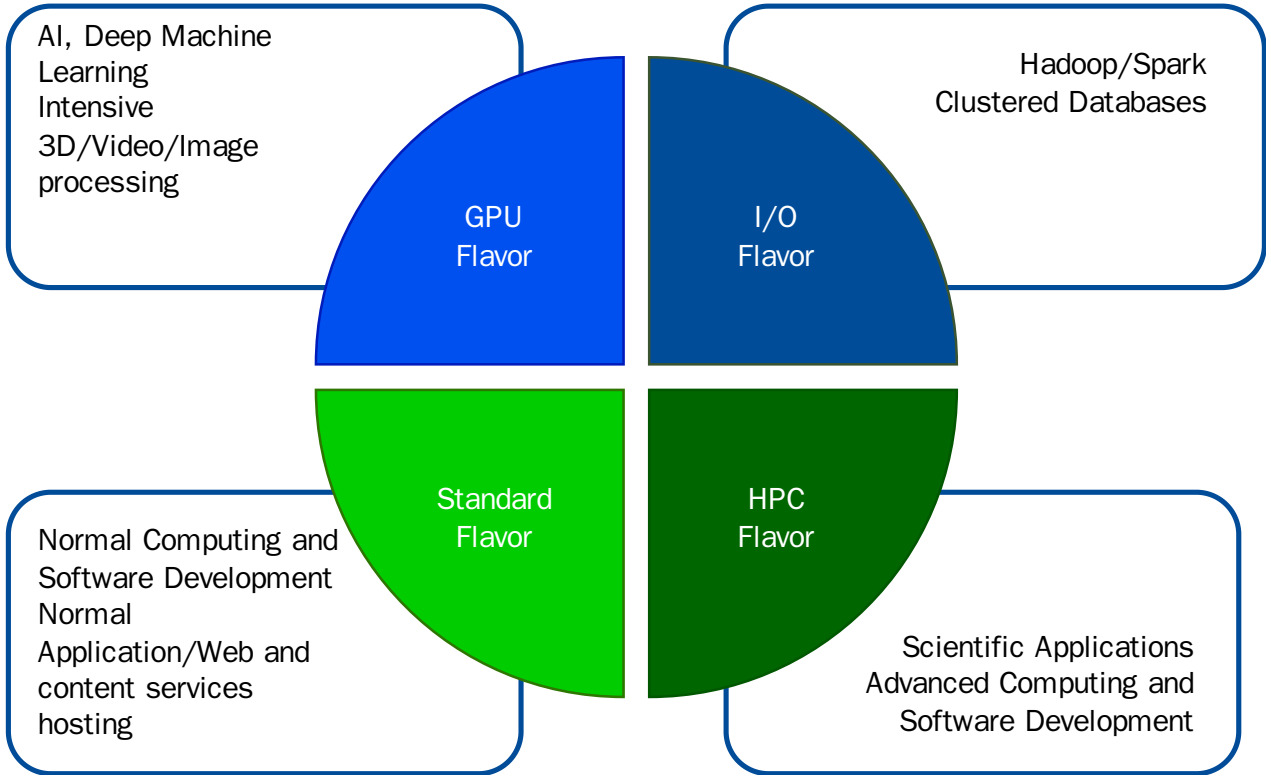
ePouta Connection



Typical VM connections between ePouta and customer's network. Such connections are normally coordinated between CSC's cloud team, Funet[1] and customers IT department for initial setup.

[1] https://en.wikipedia.org/wiki/FUNET

Hardware Options



Diverse set of hardware options to support your computing needs

VM flavors (extract)

Flavor	Cores	Memory	Disk(root)	Disk(ephemeral)	Disk(total)	Memory/core	Billing Units/h
standard.tiny	1	0.9GiB	80GB	0GB	80GB	0.9GiB	0.25
io.70GB	2	9.7GiB	20GB	70GB	90GB	4.8GiB	3
hpc.4.5core	5	21GiB	80GB	0GB	80GB	4.2GiB	8
standard.3xlarge	8	62GiB	80GB	0GB	80GB	7.7GiB	16
io.700GB	16	75GiB	20GB	700GB	720GB	4.7GiB	24
hpc.5.128core	128	464GiB	80GB	0GB	80GB	3.6GiB	160
tb.3.1470RAM	80	1470GiB	80GB	2500GB	2580GB	18GiB	320

Full list of flavor available at https://docs.csc.fi/cloud/pouta/vm-flavors-and-billing/

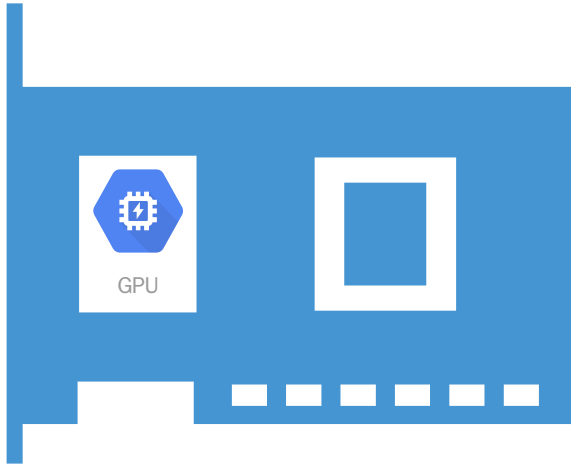
Billing units and Pricing at https://research.csc.fi/pricing

(1 BU $\approx$ 0.021 EUR)

GPU Flavors in Pouta

- GPU VM flavors provide high performance computing leveraging General-purpose computing on graphics processing units (GPGPUs). PCI passthrough is used for performance.

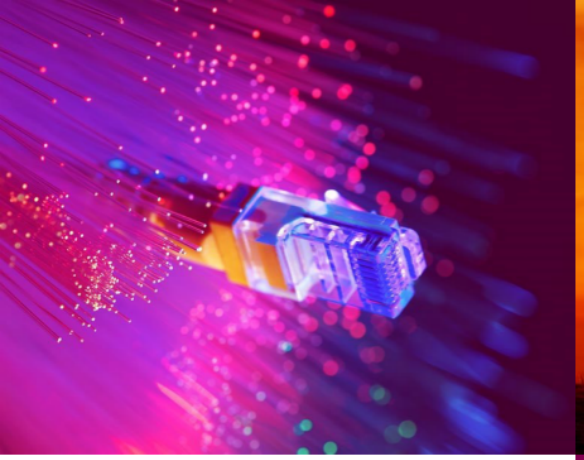
Flavor family	GPU Card	CPU	SSD
GPU 1.*	NVIDIA Tesla P100	Intel® Xeon®	✔
GPU 2.*	NVIDIA Tesla V100	Intel® Xeon®	✔
GPU 3.*	NVIDIA Ampere A100 [1]	AMD® EPYC®	✔



[1] With Multi-Instance GPU (MIG) support

- CSC's Code Optimization Service is available to help you coding.
  - https://research.csc.fi/optimization-service
- GPGPUs are also available in the batch system on Puhti:
  - https://docs.csc.fi/#computing/system/

- OS images pre-installed with latest CUDA version are available. (You may also use your own OS images by installing required libraries yourself).



Getting Access

CSC account

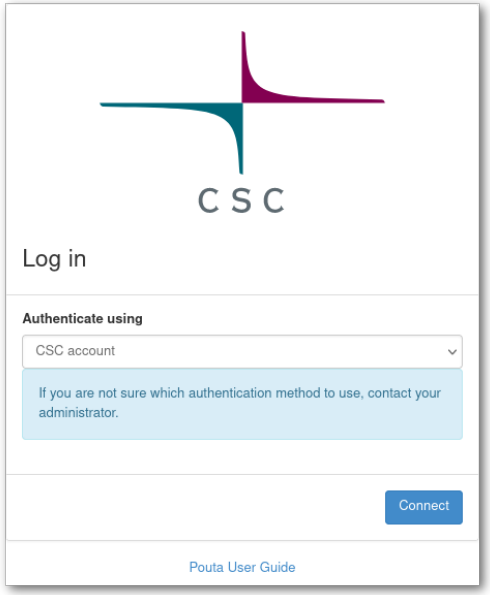
In order to access Pouta, you first need a CSC account. There are two ways to create an account:

- Getting an account with Haka or Virtu. If your home organization is a member of the Haka or Virtu federation, you can create an account yourself.
- Getting an account without Haka or Virtu, by contacting servicedesk@csc.fi

If you already have an account, you can visit:

https://my.csc.fi/myProfile

There you will be able to see and edit the account details, like change your password.

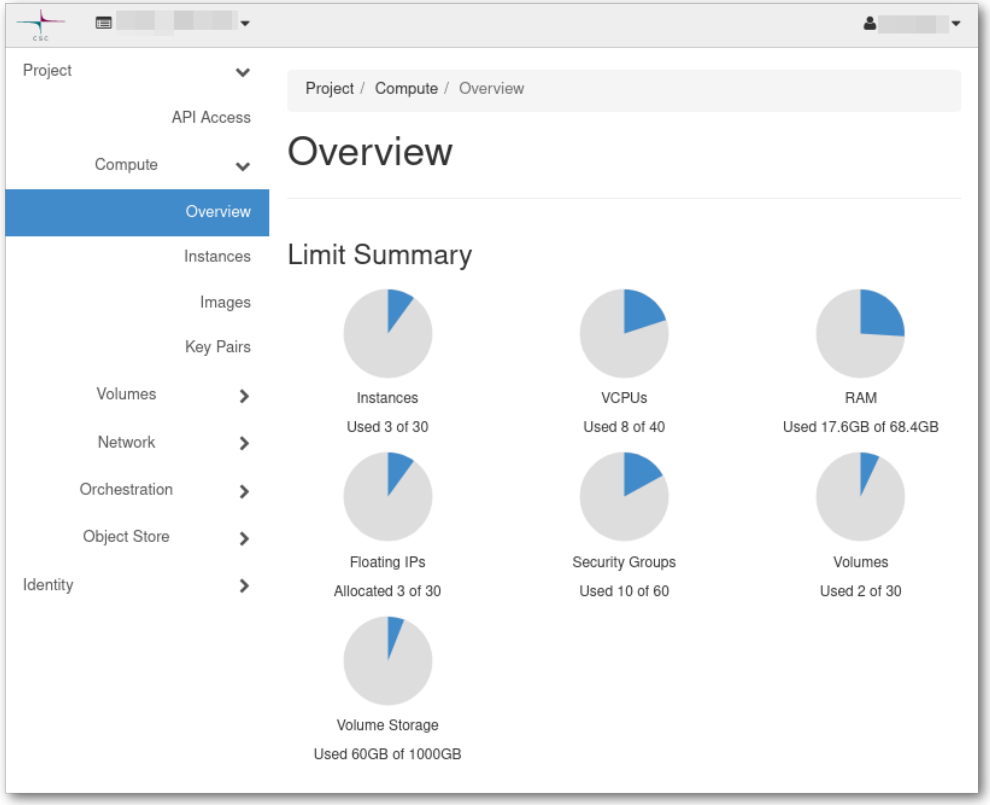


Project access

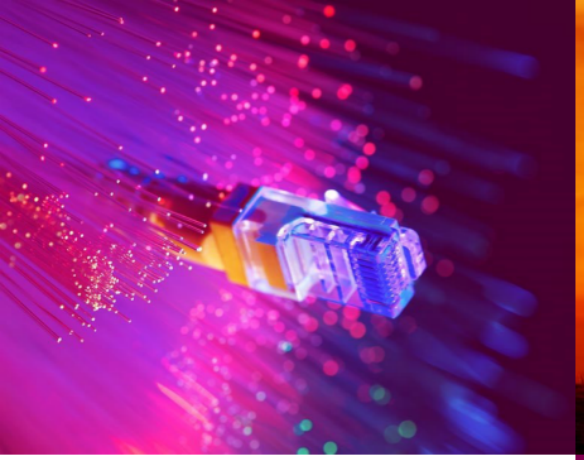
Once you have an account, you need access to a project. You can get access with two different roles:

- Project manager
- Member

Once done, you can confirm access by login in https://pouta.csc.fi [1]



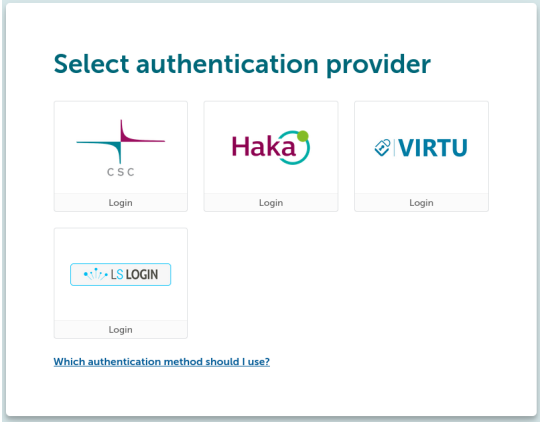
[1] You can login using **Haka** or a **CSC account**, but you still need to have an active CSC account



Creating and Configuring VMs

User Interfaces

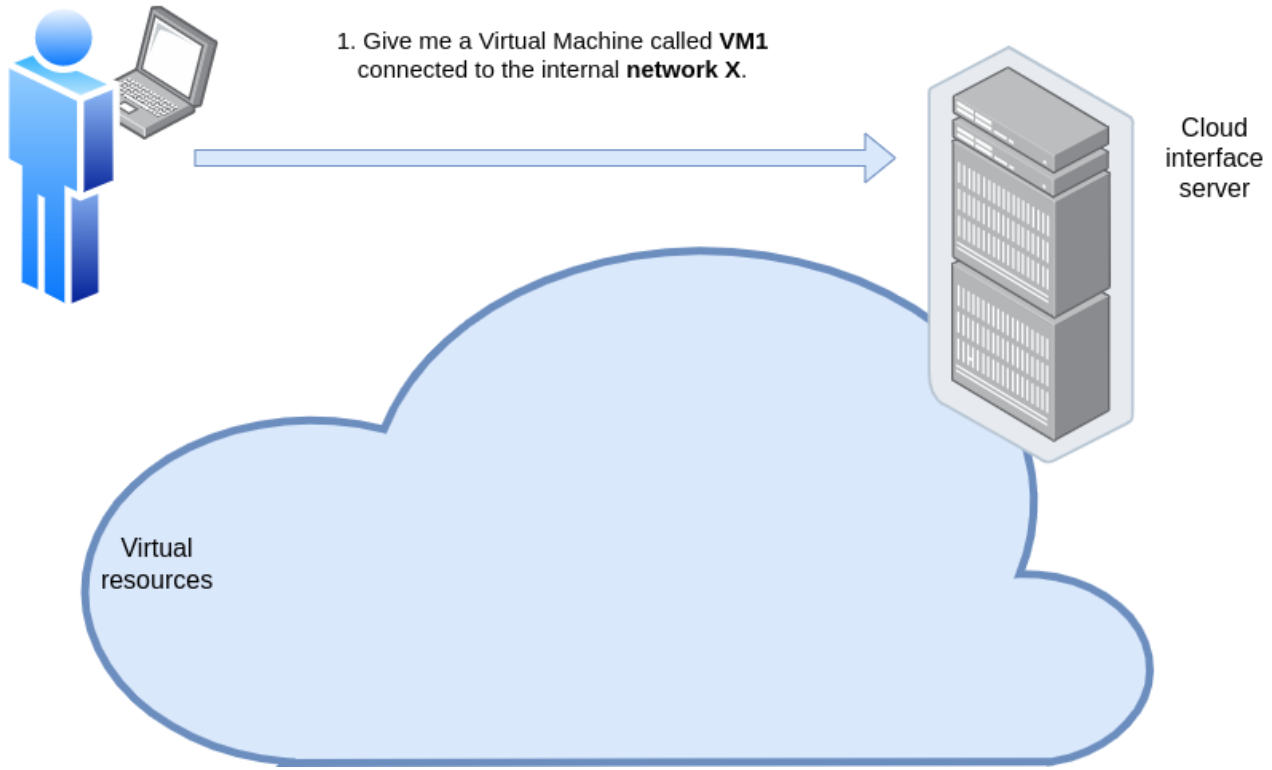
- Web User Interface
  - Suitable for administering individual VMs, keys, images, volumes...
  - Several authentication providers supported



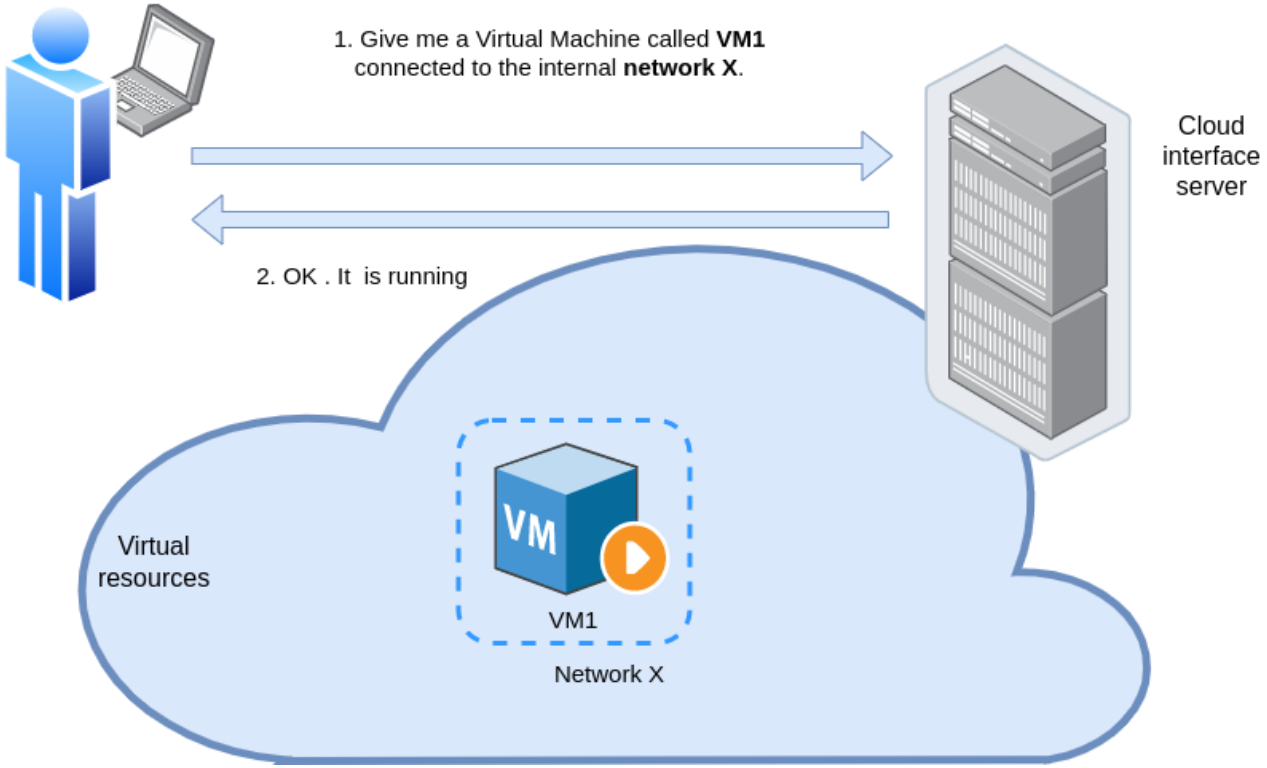
- CLI tools
  - Suitable for more elaborate resource provisioning and possibly some lightweight (scripted) software integrations
  - More info at https://research.csc.fi/pouta-install-client

- Programming APIs
  - Suitable for building very large systems and stacks
  - Support from individual services (compute, storage) to full-fledged orchestration
  - List of APIs available at https://pouta.csc.fi/dashboard/project/api_access/ (login required).

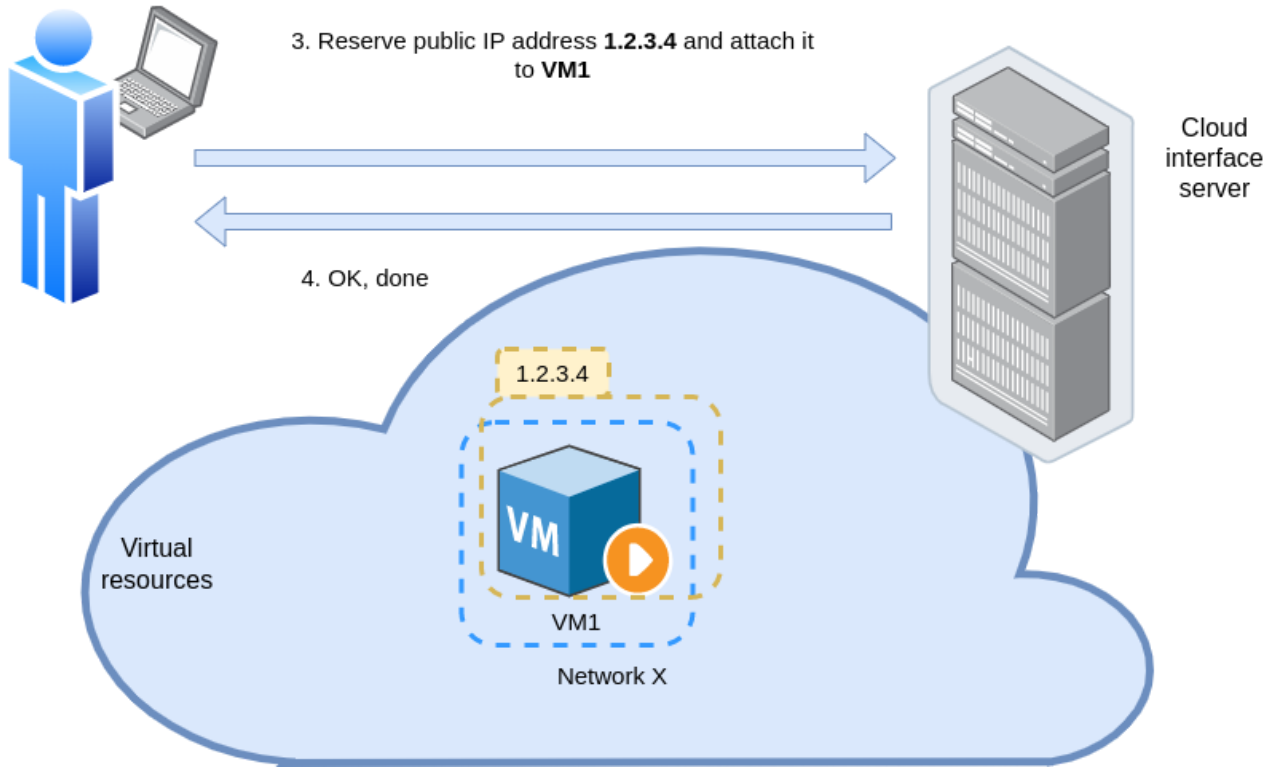
Workflow for Creating Resources I



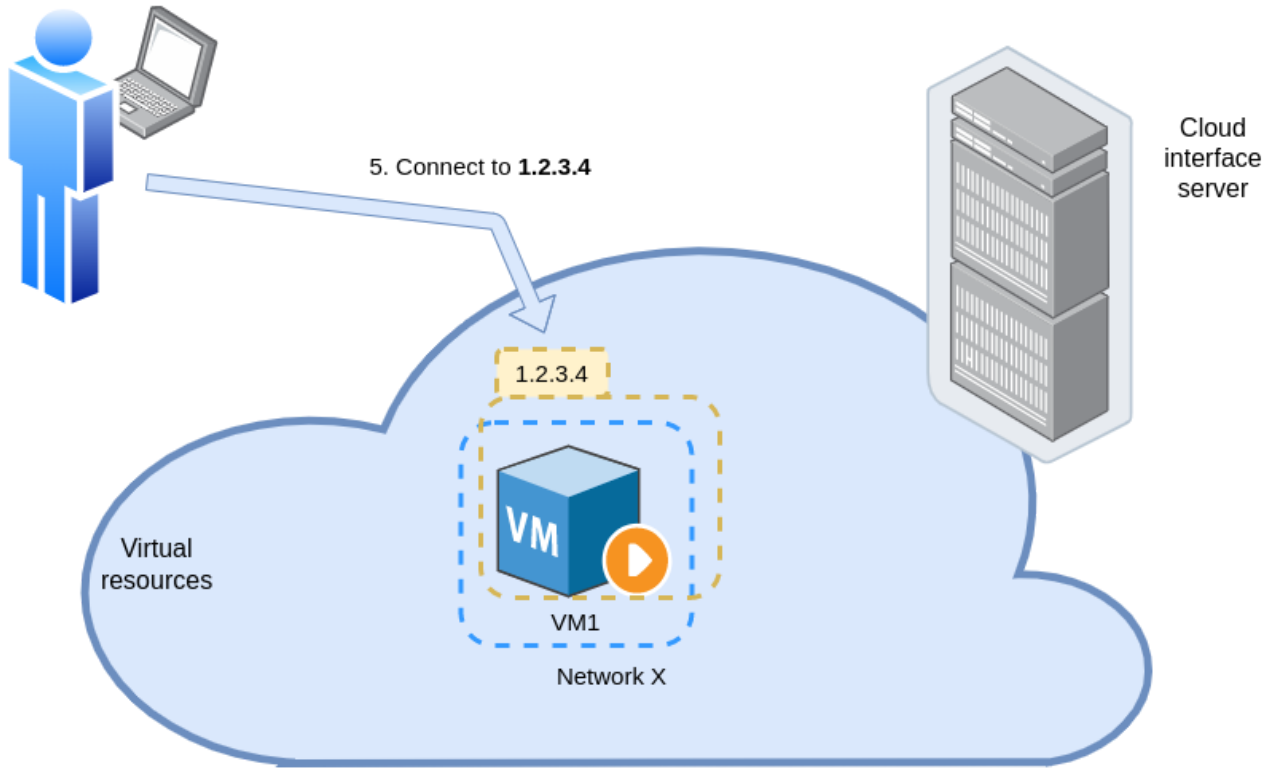
Workflow for Creating Resources II



Workflow for Creating Resources III



Workflow for Creating Resources IV



SSH

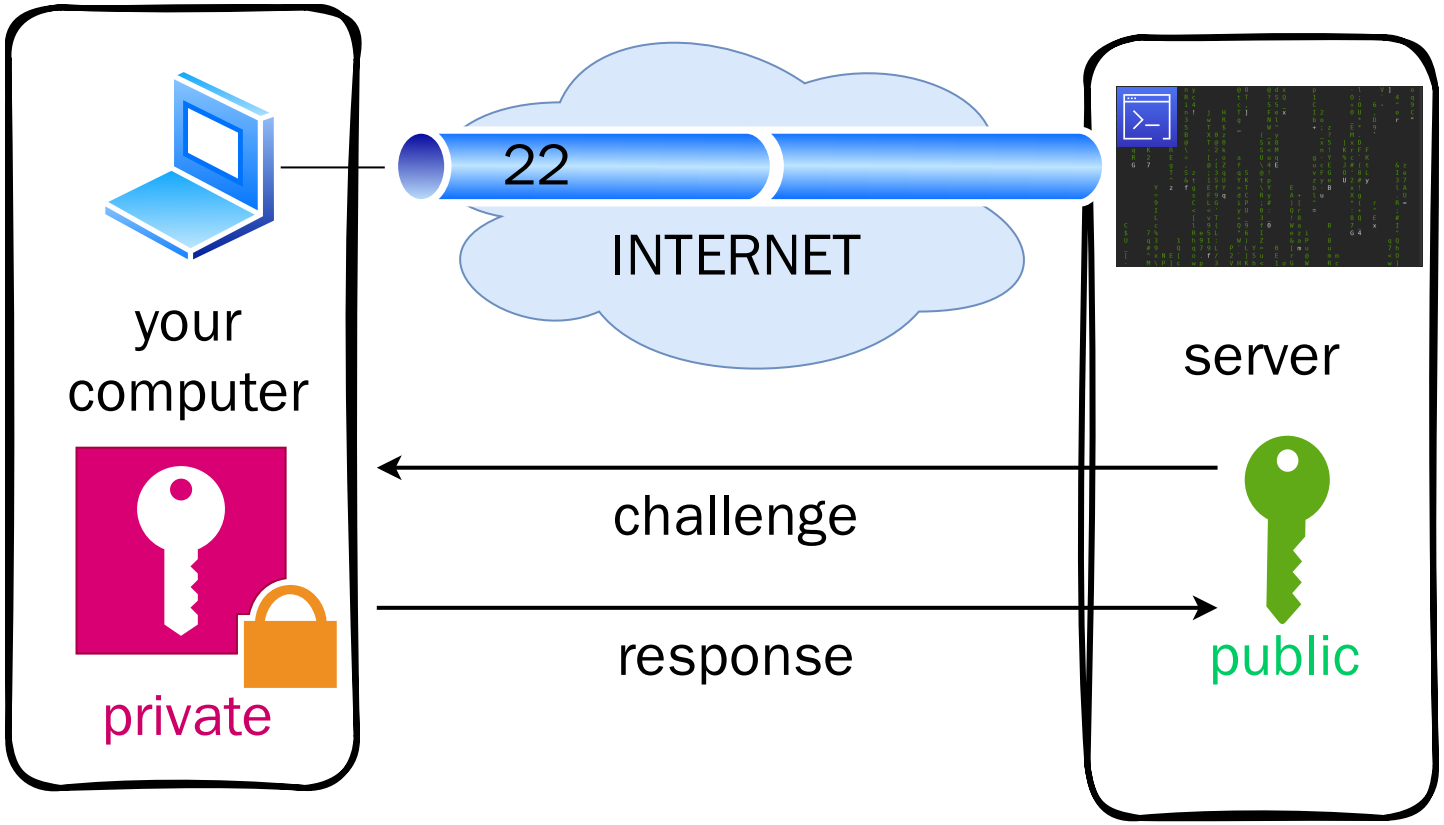
SSH or Secure Shell is a cryptographic network protocol for operating services securely over an unsecured network.[1]

SSH keys

SSH keys serve as a means of identifycation to an SSH server using public-key cryptography and challenge-response authentication. Key-based authentication is not prone to brute-force attacks and credentials are not exposed to the server.[2]

[1]https://en.wikipedia.org/wiki/SSH, [2]https://wiki.archlinux.org/index.php/SSH_keys

Diagram



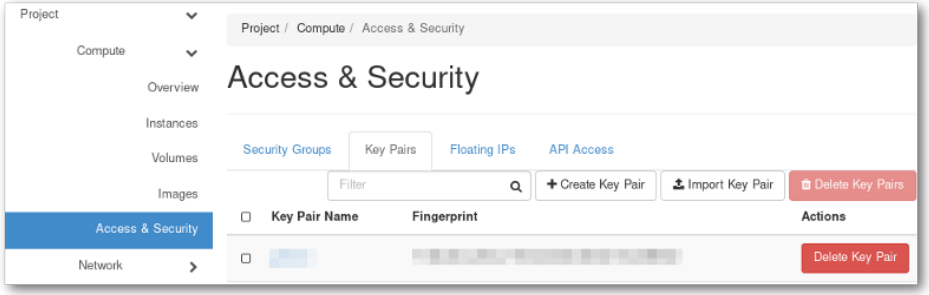
https://docs.csc.fi/cloud/pouta/launch-vm-from-web-gui/, https://docs.csc.fi/cloud/pouta/connecting-to-vm/

Creating a SSH Key pair

From the WebInterface, navigate to:

- **Compute>Access and Security>Key Pairs**

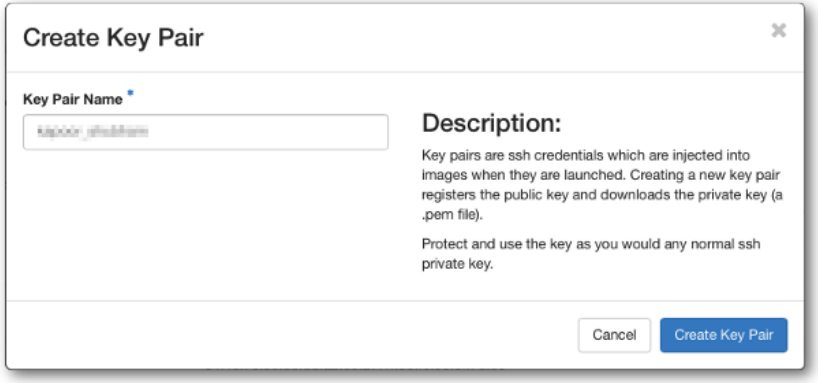
Click on create Key Pair, name key as **lastname_firstname**



Download the *private* key and store it safely. It **will not** be possible to download the private key again.

The public key will be stored in Pouta.

It is also posible to create the key using the command line tool `ssh-keygen` and then upload the public key to Pouta.



Storing a (private) Key

Linux and Mac OS X

Windows (PowerShell)

1. Create `.ssh` directory in `$HOME` if it is not there already, copy the key pair to the `.ssh` directory.

```
mkdir -p -m=700  .ssh
mv ~/Downloads/yourkey.pem ~/.ssh/
chmod 400 .ssh/yourkey.pem
```

```
mkdir ~/.ssh
mv ~/Downloads/yourkey.pem ~/.ssh/
```

2. Protect key with passphrase (Optional)

```
ssh-keygen -p -f yourkey.pem
```

```
ssh-keygen.exe -p -f yourkey.pem
```

Storing a (private) Key II

Putty (for older Windows)

1. Download Putty and Puttygen tools if you don't have them
2. Load your private key (`yourkey.pem`) into puttygen and change it to .ppk format
3. Open Putty, load .ppk file under Connection | SSH | Auth | Private key file for authentication
    1. Provide user name cloud-user
    2. Provide the password which you added to Puttygen (Optional)

Copying SSH key to server

- Automatic

```
$ ssh-copy-id -i ~/.ssh/id_ed25519.pub username@remote-server
    username@remote-server.org's password:
```

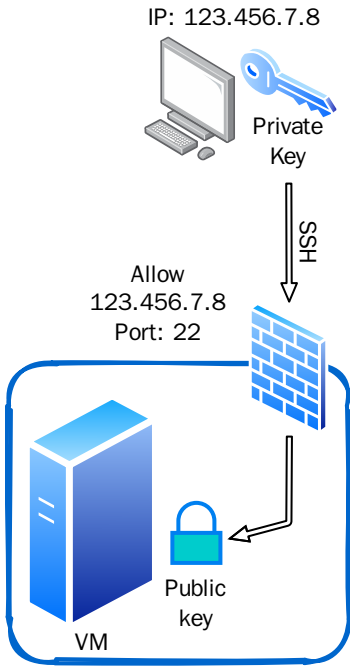
- Manual

```
$ scp ~/.ssh/id_ecdsa.pub username@remote-server.org:
$ ssh username@remote-server.org
    username@remote-server.org's password:
$ mkdir -p ~/.ssh
$ chmod 700 ~/.ssh
$ cat ~/id_ecdsa.pub >> ~/.ssh/authorized_keys
$ rm ~/id_ecdsa.pub
$ chmod 600 ~/.ssh/authorized_keys
```

Security groups

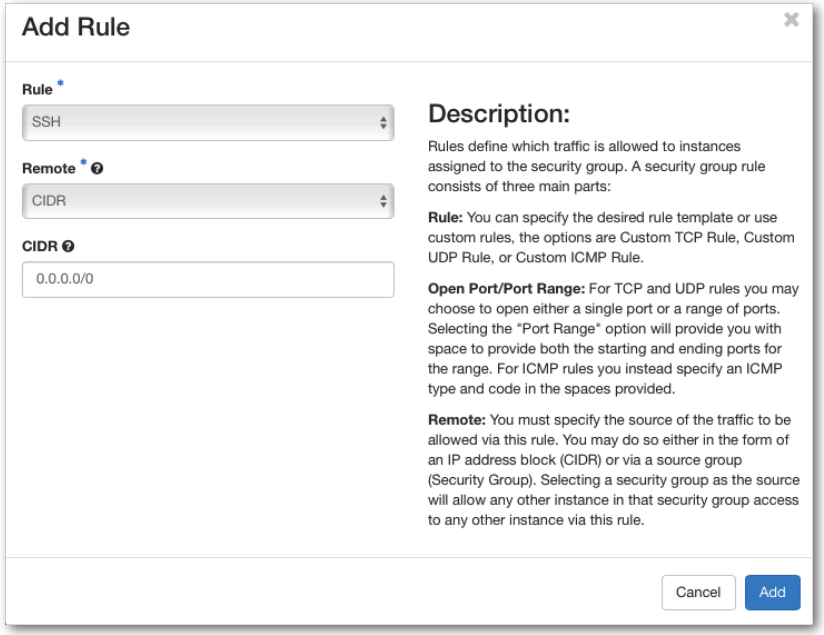
A Security Group defines a set of cloud level firewall rules for filtering traffic, typically inbound, but also outbound.

- By default a "Security Group" blocks all incoming connections to your VM,
  - and allows all outgoing connections.
- VM level firewall rules are still possible as an extra layer of security.
- One security group includes one or several several “rules”.
- One security group can be assigned to one or several Virtual machines.
- One Virtual machine can be assigned one or several security groups.



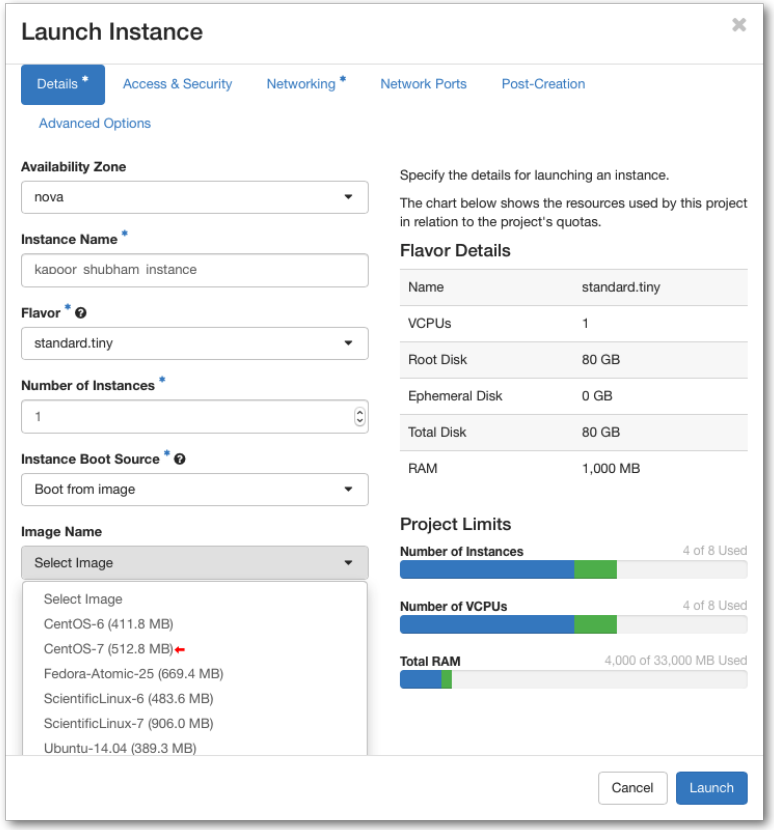
Creating a security group rules

- Navigate to:
  - **Network > Security Groups**
- First create a security group (any sensible "Name" and "Description" are valid).
- Secondly, click in "Manage Rules" and create rule opening entries.
  - You may use a predefined rule, ex. SSH, where you only need to provide the range of IPs that will be able to connect using SSH.
  - You may also create a "Custom TCP/UDP/ICMP Rule", where more fine grained rules can be defined.



Creating an Instance

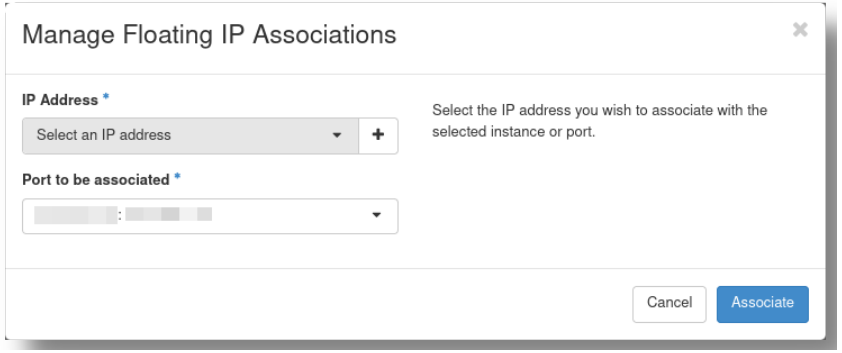
- Navigate to:
  - **Compute>Instances**
  - and click in "Launch Instance"
- Give **Instance name** as `lastname_firstname_instance`.
- Select a **Flavor** of your choice (`standard.tiny` is a good first choice)
- Select **Instance Boot Source** as "Boot from image".
- Pick an **Image Name** - any image
- Navigate to the **Access & Security** tab and select your **Key Pair**.
  - Make sure that the "SSH - World" Security Group is selected (otherwise the firewall will block the connection).

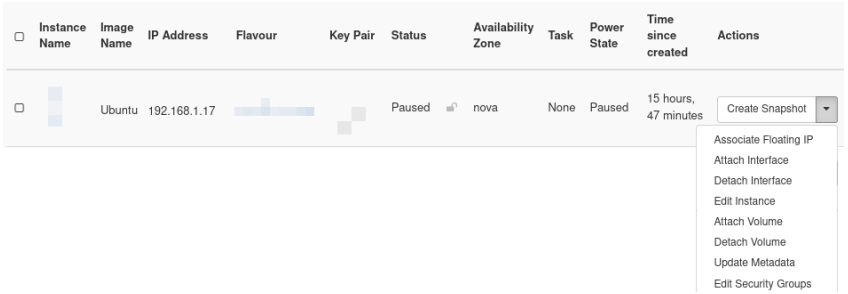


Attaching a Floating IP

By default VM get an private non-routable IP, so to get external connectivity, you need a public floating IP.

- Navigate to:
  - **Compute > Instances**
  - Under Actions click in **Associate Floating IP**
- In the dialog that appears, select an IP address. If no IP is available, click in the plus sign.





NOTE: You can move floating IPs between VMs

Connect to the VM by SSH

```
ssh <USER>@<FLOATING-IP>
```

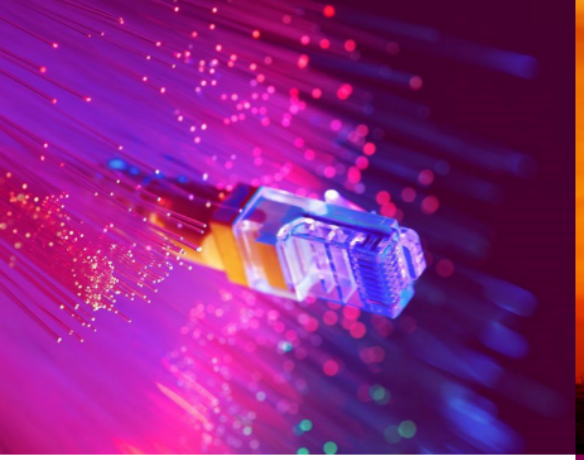
- `<FLOATING-IP>`, must be the floating ip that was set up in the previous step
- `<USER>`, must be the username suitable for the distribution used:
  - `ubuntu` for Ubuntu distributions
  - `centos` for Centos8 distributions
  - `cloud-user` for Centos7 distributions
- In most cases if you try to connect as `root`, it will fail, but you will get back the correct username:

```
$ ssh root@XXX.YYY.ZZZ.WWW
Please login as the user "centos" rather than the user "root".
```

Create and access a VM in cPouta

(Checklist)

- ☑️ Internet access
- ☑️ Access to Pouta Web UI
- ☑️ One IPv4 address - a public "Floating IP"
- ☑️ Security Group permitting access from User's computer
- Identity:
  - ☑️ SSH Key-Based Authentication (**recommended**)
  - Password (only for tests)
- ☑️ SSH client software



Coffee break

🥐 ☕ 15min

Web UI Login

- From your web browser, browse:

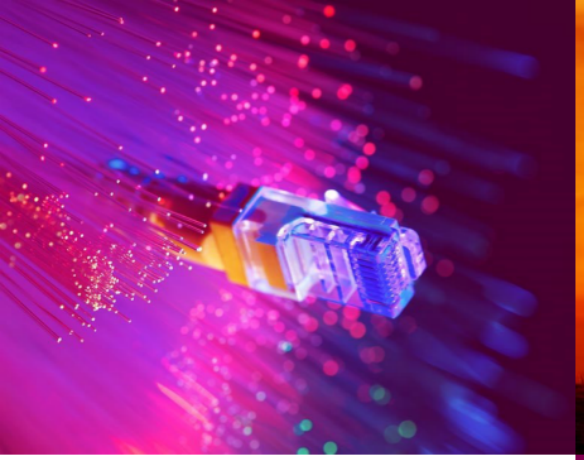
https://pouta.csc.fi

- Log in using the provided training accounts
  - Issues logging in? Please let us know and we'll help. Everybody should be able to log in to the cPouta Web UI before we start exercises

Exercise Set A

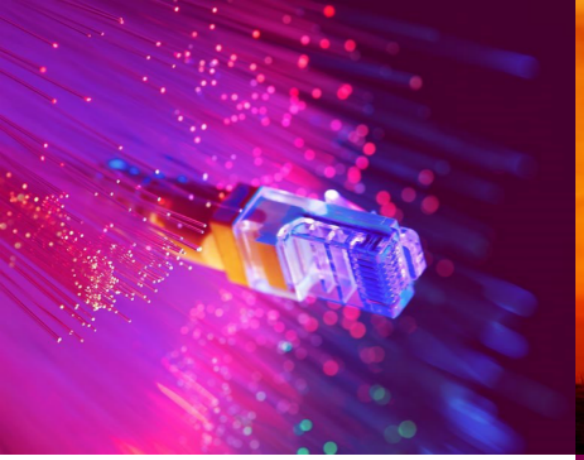
- Exercise 1 - Creating a temporary Virtual Machine for testing login
- Exercise 2 - Creating an SSH key pair for secure login to an instance
- Exercise 3 – Create your own Security Group

Go to exercise set A



Lunch break

🍽️ 60min

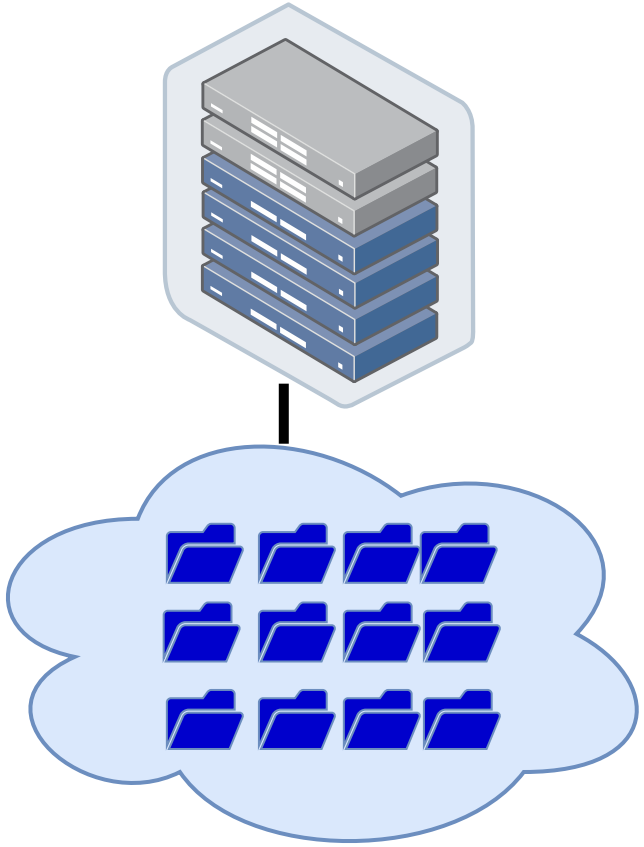


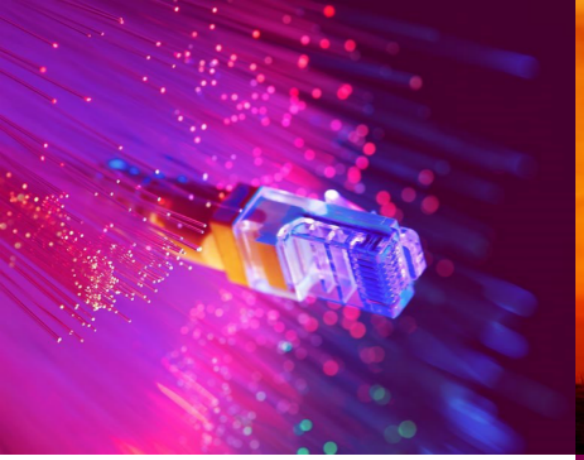
Persistent Storage Volumes

Persistent Data Volumes

A good practice is to separate the **data** from the **application** (OS and other software). Volumes are very helpful to achieve this. It is recommended to store the data in a volume, and the OS and software in another.

- Volumes are **project specific**, not user specific.
- A project can have **several volumes**.
- Volumes can be **transferred to other projects** in same cloud service.
- One volume can be **attached to one VM** at a time.
- Volumes can be management with **web interface** or **command line client**.
- Data stored in Persistent volumes is resilient to Disk Failures, Server failures, Accidental deletion of VMs, Crashing of VMs
  - But not to human errors ➡️ No Backups.
- It is easy to create and recover snapshots.

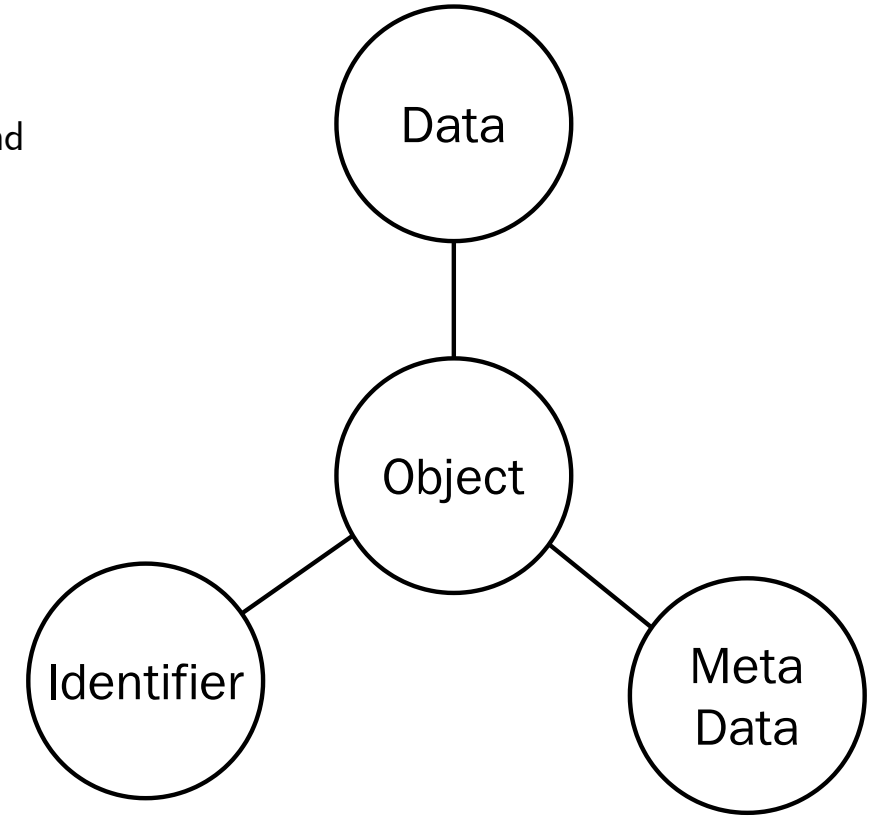




Object Storage

What is Object Storage

- Object storage is a computer data storage architecture that manages data as objects.
- Each object consists of three things: Data, Metadata and Globally unique identifier.
- Different from other data storage architectures like File Storage: Data as a file hierarchy and Block Storage: Data as blocks within sectors and tracks.
- Accessed via APIs at application-level, rather than via OS at system level.
- Scalable and Self healing storage.

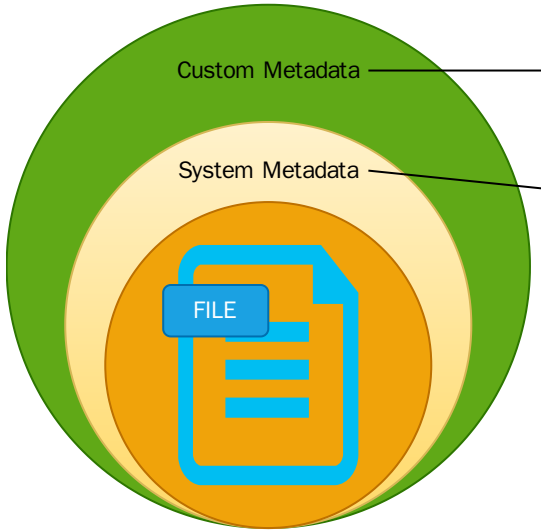


File Storage vs Object Storage

File Storage

File Name: CTSCAN_Kapoor
Created by: User1
Created on: 19-09-2017
File Type: DICOM

Object Storage



Object ID: 123456
Patient Name: Shubham
Patient ID: 23242
Physician Name: Dr. John
Prior1 : XYZ.DICOM
Self Destruct: 2 Year

File Name:
CTSCAN_Kapoor
Created by: User1
Created on: 19-09-2017
File Type: DICOM

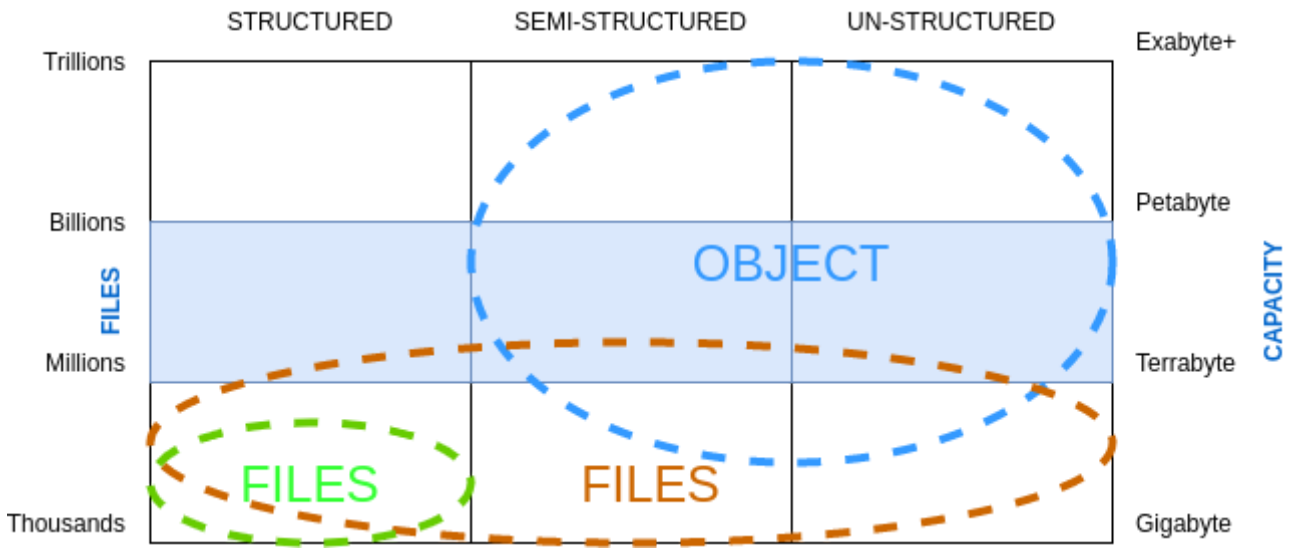
Where Object Storage Fits

On basis of Data

Type

- Storage of Unstructured/ Semi structured Data like Media files, web contents, Backup Archives etc.
- Cold Storage of structured and semi structured data like Databases, Sensor Data, Log files etc.
- Archiving files in place of local tape drives.
- Big Data, large data sets

Size



Where Object Storage does not fit

- Hot Data.
- Relational/OLTP Databases.
- Latency intolerant applications.
- Data with Strict consistency requirements.

Object Storage Around us

Amazon S3

Microsoft Azure
Blob Storage

IBM Cloud
Object Storage

Google Cloud Storage

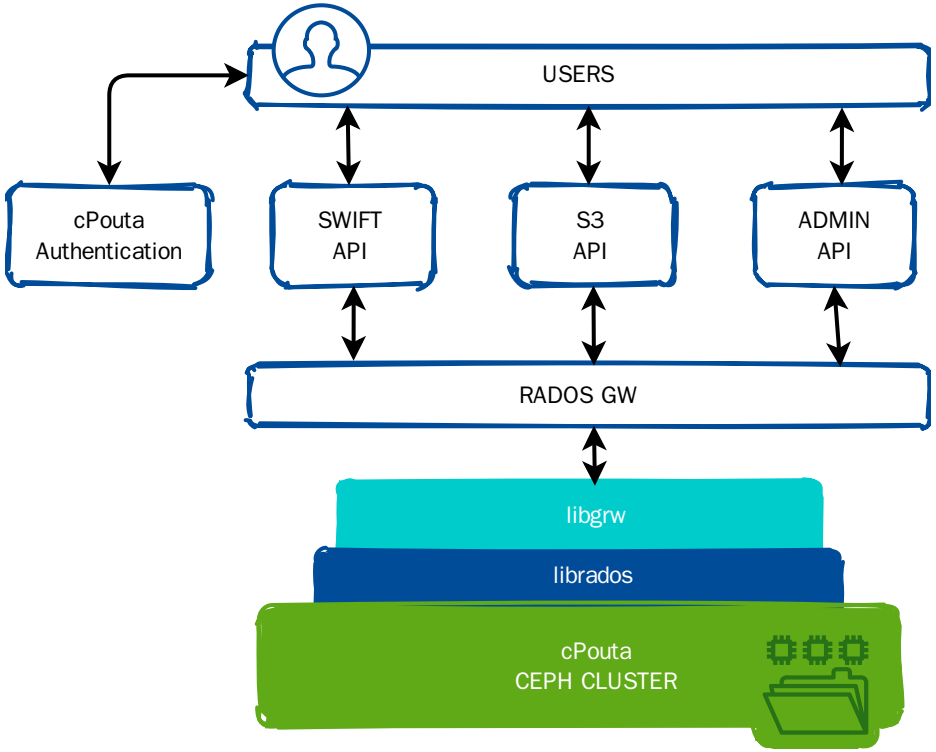
- Social Media Storage
- Big Data Analytics
- Offsite Backup/Archive
- Static Website Hosting
- Digital Archives

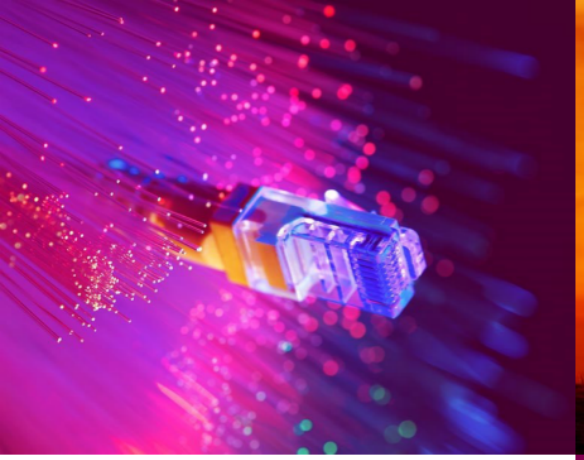
Allas: Object Storage in cPouta

- Launched for customers in 2018.
- REST API available, S3 and Swift API compatible, Supports WebUI and Swift/S3 CLI tools
- Charging on the basis of storage not transactions.
- Initial quota for object storage/project is **10 TiB**.
  - Buckets per project: **1000**
  - Objects per Bucket: **500000**
- Content Agnostic, Distributed, Scalable and Highly available Data Storage.
- Access control possible for buckets/objects.

Allas billing and quotas documentation.

Allas: Architecture





OpenStack CLI

Install OpenStack CLI

First you need to install the OpenStack command line client:

- This client is written in python.
- The latest version only support Python v3.8 or newer
- The easiest way to install it is via pip:

```
pip install python-openstackclient
```

For other installation methods, and always up to date information you can visit:

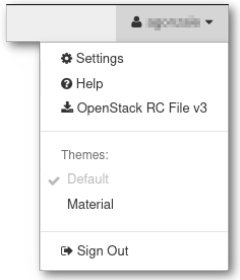
https://pypi.org/project/python-openstackclient/

To confirm the client is installed you can run:

```
$ openstack --version
openstack 6.0.0
```

Login using an OpenStack RC file

First you need to download the RC file. Go to the Pouta web UI, click in your name and then in "OpenStack RC File v3". Then you need to "source" the file you downloaded:




```
$ source <OpenStackRCFile>
Please enter your OpenStack Password for project project_y
```

The script will ask for your CSC account password and then set up the environment variables

```
$ env | grep OS
OS_AUTH_URL=https://pouta.csc.fi:5001/v3
OS_PROJECT_ID=vcItXBbYIjZDDL5RD55mQK7Wh1Qra68PQ
OS_PROJECT_NAME=project_zzzzzzz
OS_USER_DOMAIN_NAME=Default
OS_PROJECT_DOMAIN_ID=default
OS_USERNAME=xxxxxxxxx
OS_PASSWORD=yyyyyyyyyyyyyyyyy
OS_REGION_NAME=regionOne
OS_INTERFACE=public
OS_IDENTITY_API_VERSION=3
```

Flavors

The first requirement is to decide which flavor to use. Different flavors will provide different resources.

```
$ openstack flavor list
+--------------------------------------+------------------+--------+------+--------
| ID                                   | Name             |    RAM | Disk | Ephemer
+--------------------------------------+------------------+--------+------+--------
| 0143b0d1-4788-4d1f-aa04-4473e4a7c2a6 | standard.tiny    |   1000 |   80 |
| (...)                                | (...)            |  (...) | (..) |    (...
+--------------------------------------+------------------+--------+------+--------
```

The more resources, the higher will be the cost.

For a whole and updated flavor list go to https://docs.csc.fi/cloud/pouta/vm-flavors-and-billing/. There you will also find the cost associated with it.

Images

Next, you need to decide the image to use. You can get a list of images by:

```
$ openstack image list
+--------------------------------------+----------------------+--------+
| ID                                   | Name                 | Status |
+--------------------------------------+----------------------+--------+
| f505d49e-55e6-4f3f-9790-cc8250cace1b | CentOS-7             | active |
| 72251ff8-607d-451a-8769-bb2741464577 | CentOS-7-Cuda        | active |
| ea8d1ff7-1f2f-4255-ba27-62a67bc9c6bd | CentOS-8-Stream      | active |
| 2ca237c5-bd0a-4469-ae9f-20878dd288a9 | Fedora Cloud Base 31 | active |
| 2cce570c-a98d-4bab-b329-8d657c77c72e | Ubuntu-18.04         | active |
| 67f975c8-af09-4a7e-be1e-42f5a16e0cd8 | Ubuntu-20.04         | active |
| 0d952564-c0f2-4b54-ad4a-78ce6d32edb7 | Ubuntu-22.04         | active |
| 3a9aad67-0f9c-4493-b574-17fe28d40afc | cirros               | active |
| 646c6051-19ba-48e8-b8e7-397e12a55be1 | ftp-test             | active |
+--------------------------------------+----------------------+--------+
```

In that list above, there is a list of `public` and `private` (Fedora Cloud Base 31 and ftp-test) images.

It is possible to upload your own images, or create an image using snapshots. https://docs.csc.fi/cloud/pouta/images/

Network

The Virtual Machine needs to be connected to a network.

```
$ openstack network list
+--------------------------------------+------------------+-------------------------
| ID                                   | Name             | Subnets
+--------------------------------------+------------------+-------------------------
| 26f9344a-2e81-4ef5-a018-7d20cff891ee | public           |
| c55bc796-841f-4704-a1a2-8f29bb9a699a | project_xxxxxxx  | bef20f4d-015d-4eff-a120-
+--------------------------------------+------------------+-------------------------
```

By default all projects will have a network called `project_xxxxxxx` where `xxxxxxx` is the project number. If unsure, choose this network.

Security groups

Last but not least, you need to specify a security group. By default no incoming security group is created. You need to create an incoming SSH security group and a rule:

```
$ openstack security group create ssh-ip
$ openstack security group rule create --ingress --dst-port 22 \
                                       --remote-ip $(curl ifconfig.co -4) \
                                       --protocol tcp ssh-ip
```

The rule above will create an `ingress` opening, in the port `22` for your IPv4 ip [1], in the `tcp` protocol.

You can also list the existing security groups.

```
$ openstack security group list
+--------------------------------------+---------+------------------------+--------
| ID                                   | Name    | Description            | Project
+--------------------------------------+---------+------------------------+--------
| aa4243ab-120d-439a-9156-938b1202540c | default | Default security group | 384ce70
| 172921f5-4b26-42c6-8d60-0e77206823f8 | SSH-IP  | ssh-ip                 | 384ce70
+--------------------------------------+---------+------------------------+--------
```

[1] `ifconfig.co` returns the IP of the client that makes the request, your IP

SSH Key

We will use the SSH protocol to login in Virtual machines. For that, you need a ssh key pair.

To create run:

```
$ openstack keypair create any-name-is-fine >~/.ssh/any-name-is-fine
```

This will do two things, return via the command line the private key and store it at `~/.ssh/any-name-is-fine`. And store the public key into the OpenStack servers.

You can list the available keys [1] by:

```
$ openstack keypair list
+------------------+-------------------------------------------------+------+
| Name             | Fingerprint                                     | Type |
+------------------+-------------------------------------------------+------+
| any-name-is-fine | f2:b1:23:7f:58:ea:fc:e4:0a:1d:5a:f3:2f:f7:dd:35 | ssh  |
+------------------+-------------------------------------------------+------+
```

[1] After creation, you will not be able to obtain again from the server the private key

Create a Virtual Machine

Once all the previous steps were followed, the last is to create the Virtual Machine:

```
$ openstack server create --flavor standard.tiny \
                          --image cirros \
                          --nic net-id=project_2001316 \
                          --security-group ssh-ip \
                          --key-name any-name-is-fine \
                          VM-name
```

You can list the all your Virtual Machines:

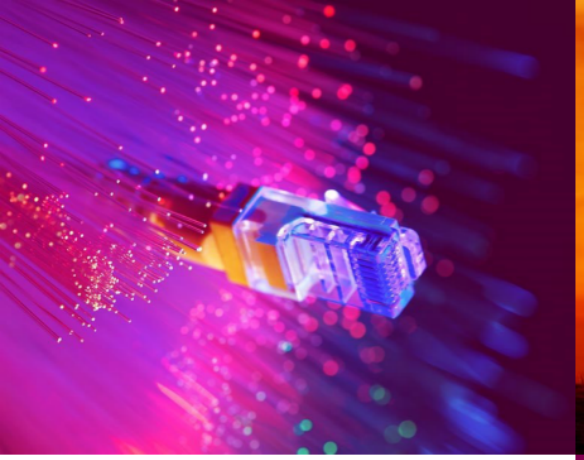
```
$ openstack server list
+--------------------------------------+---------+--------+--------------------+---
| ID                                   | Name    | Status | Networks           | Im
+--------------------------------------+---------+--------+--------------------+---
| 550fa2a8-0932-45f9-b579-dbbed4df2dc4 | VM-name | ACTIVE | project_XX=x.x.x.x | ci
+--------------------------------------+---------+--------+--------------------+---
```

Other commands

The OpenStack command line interface supports much more commands, here is a short subset of them:

- `$ openstack --help`, shows all available commands and options
- `$ openstack server`, shows all available commands on the `server` subsection.
- `$ openstack server show VM-name`, shows the information of a VM.
- `[-f {csv,json,table,value,yaml}]`, allows to change the default `table` format of the output.

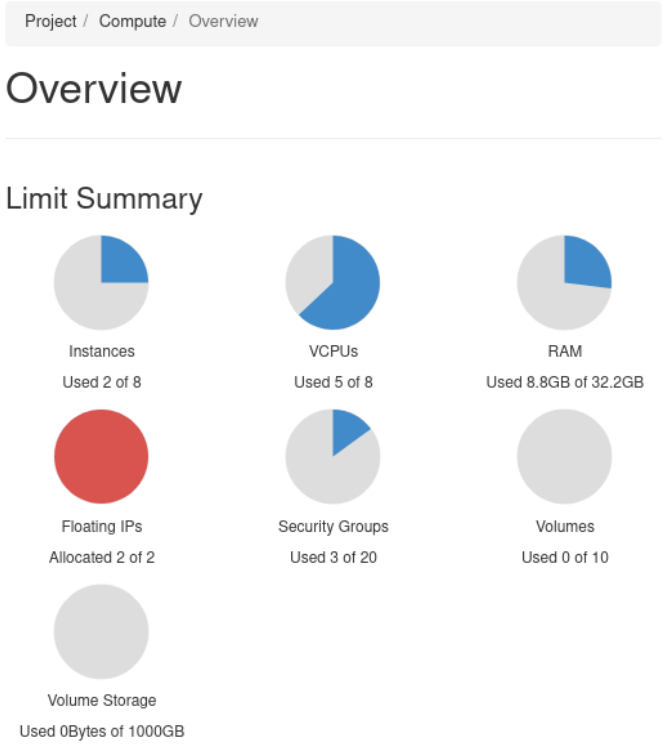
The OpenStack CLI offers all functionality that is offered by the API (or the web UI).



Pouta management

Pouta: Managing Project

- A Pouta project contain a set of resources: cores, memory, storage, ip-addresses
- A default project contains:
  - For cPouta: 8 cores, 32 GB memory, 1 TB disk space, 2 floating IP addresses.
  - For ePouta: Negotiated between customer and CSC
- If needed, you can ask for more resources for your project.
- Project members can build one or several VMs and volumes based on the granted resources.
- When VMs and Volumes are active they are consuming billing units (even if no one is using them).
- Project members can manage other members machines and volumes too.
- Your CSC account can be a member of many cPouta projects.

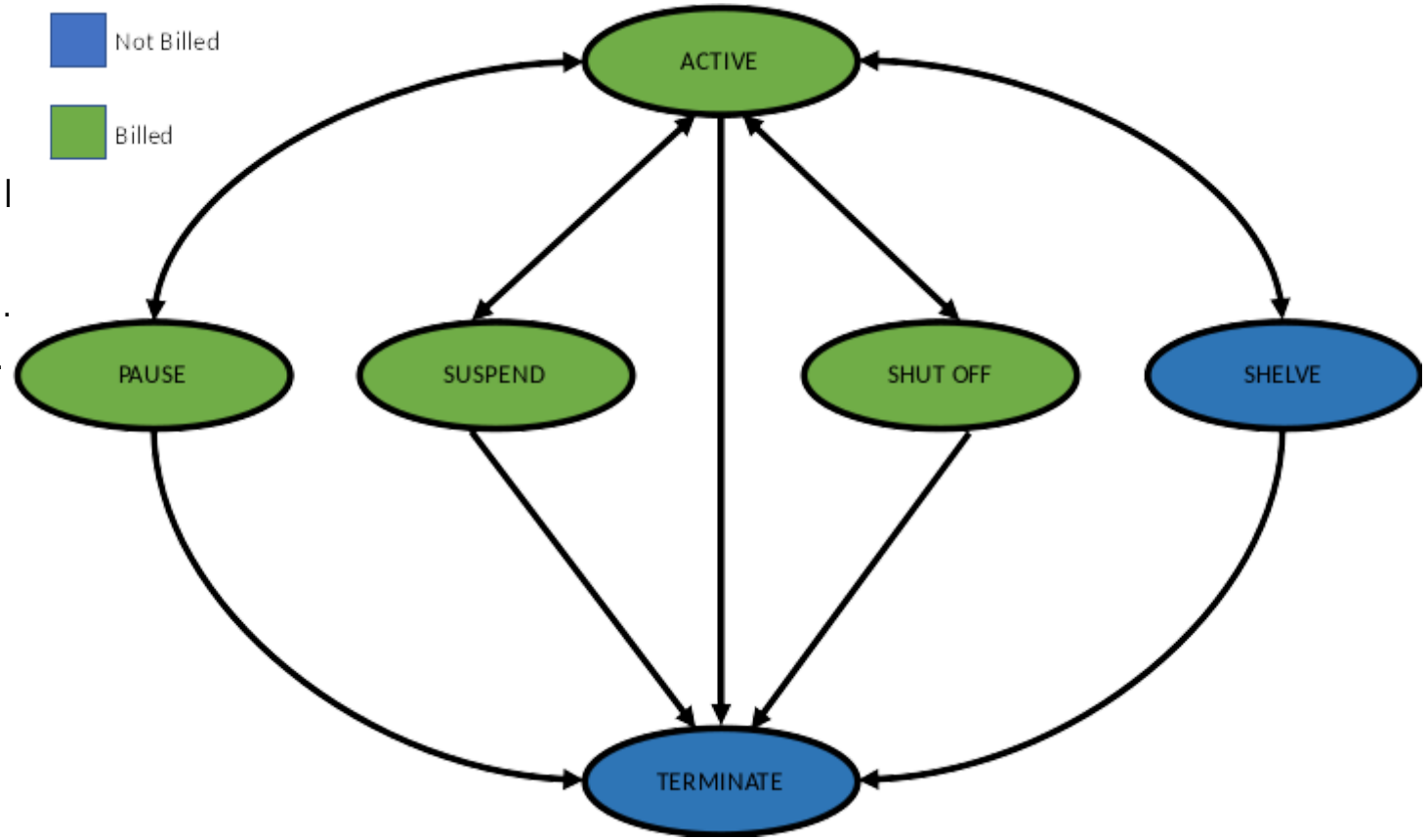


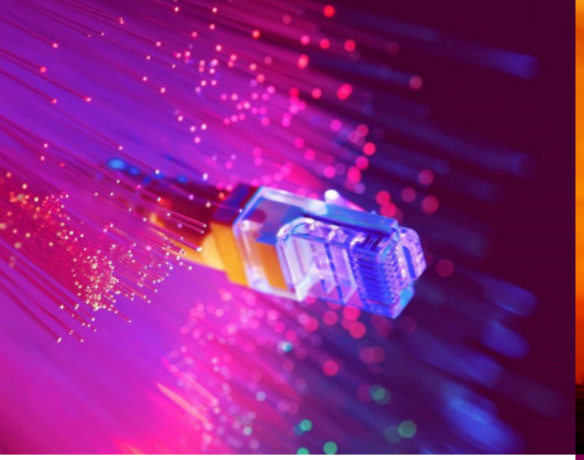
Billing

- CSC uses Billing Units (BUs) to allocate, monitor and charge for resource usage.
- Open research and educational use fall under Free-of-charge use cases.
- In Pouta you are hourly billed for
  - VM usage based upon your VM flavor.
  - Storage volumes, Object Storage and Public IPs are all accounted hourly.
- VMs start consuming BUs once you create them, they consume BU regardless you use them or not.
  - Shutting down VM **does not stop them consuming BUs**.
  - You can **Shelve**/**Terminate** your VM for stopping BU consumption.
- Object/Volume Storage start consuming BUs once you create them
  - Even if they are **not attached** to virtual machines.
- Floating IPs are billed once they are allocated, assigned to a VM or not.
- Users can monitor usage and apply for addtional BUs through My CSC.

Pouta: VM Lifecycle

- **Active** – Consumes billing units regardless of the real usage.
- **Shut off** – Not active, but still reserves the resources. Consumes still billing units.
- **Suspended** – Temporarily suspended. Current state saved. Can be revoked. Consumes billing units
- **Shelved** – VM is shut off, resources are freed & Sate is saved. Can be later on revoked if resources are free (un-shelved). Does not use billing units.
- **Terminated** – Removes the Virtual Machine and all its data.





Deploying an Application on a VM

Installing software to your VM

- The VM images provided by CSC include only just the basic linux tools.
- You can/must add the tools you need with using tools like:
  - System level repository installation. ex. `apt` or `yum`.
  - Language specific package managers, ex. `pip` or `npm`.
  - Compile from source
  - Docker
  - Conda

`sudo` command for system administration

The default user, cloud-user, does not have superuser rights, but can do admin operations with sudo (**su**peruser **do**).

```
sudo linux-command-to-execute
```

- Repository installations
- System libraries and directories
- User accounts, ex:

```
sudo reboot
sudo yum install nano
sudo nano /etc/yum.conf
sudo useradd apache
```

Repository installation: yum

In CentOS and RedHat

- List commands and options:
  - `yum help`
- Install a package from repository
  - `sudo yum install package`
- Install locally available RPM file
  - `sudo yum localinstall package.rpm`
- Update one or all packages in the system
  - `sudo yum update`
- Check what packages include the defined file
  - `yum provides filename`
- Search package names and descriptions
  - `yum search term`
- Remove the package:
  - `sudo yum remove package`

Repository installation: apt

In Debian and Ubuntu

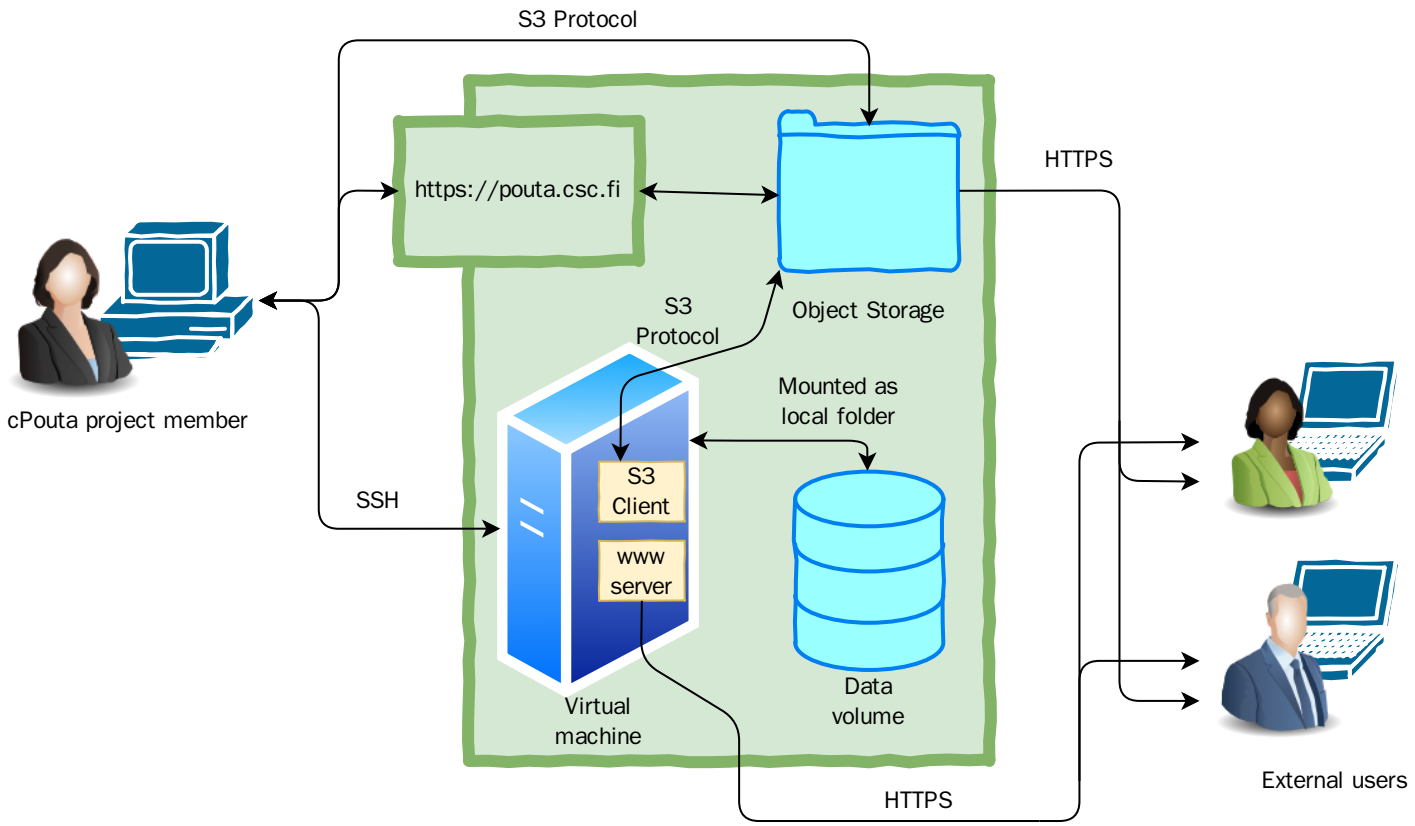
- List commands and options:
  - `apt --help`
- Install a package from repository
  - `sudo apt install package`
- Install locally available DEB file
  - `sudo apt install ./package.deb`
- Update one or all packages in the system
  - `sudo apt update`
- Check what packages include the defined file
  - `apt-file filename`
- Search package names and descriptions
  - `apt search term`
- Remove the package:
  - `sudo apt remove package`

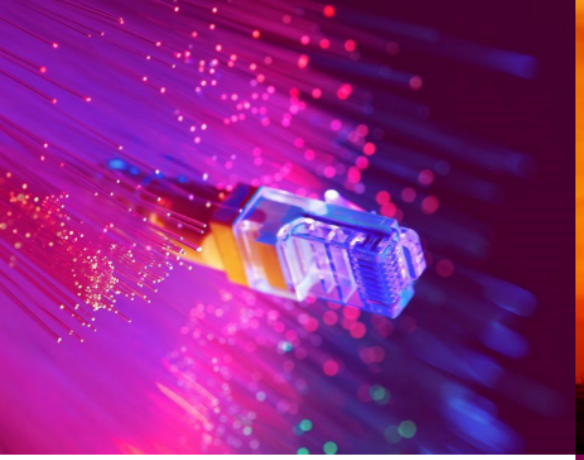
Conda / Bioconda

- Easy way to install software tools together with their dependencies
- Bioconda- repository contains over 700 bioscience tools
- Does not need superuser privileges
- For installing conda and browsing bioconda packages, check bioconda home page:
  - https://bioconda.github.io/
- Once you have conda installed, you can install application software with commands like:

```
conda create -n aligners bwa bowtie hisat star
source activate aligners
bwa
```

cPouta in action





Coffee break II

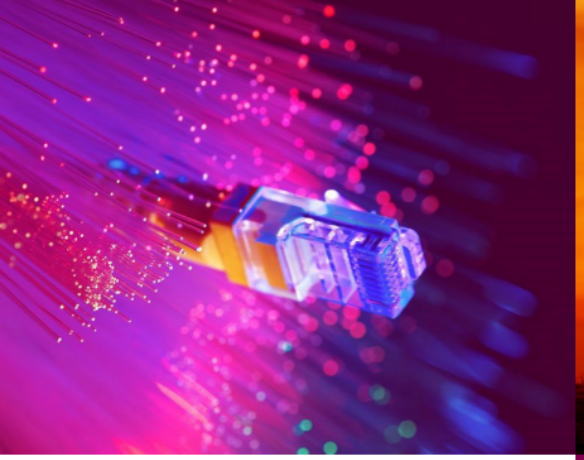
☕ 🥐 15min

Exercise Set B

Make at least 3 of Following Exercises

- Exercise 1 - Install Docker CE & run a RStudio server in Docker Container
- Exercise 2 - Build your own RStudio Server
- Exercise 3 - Install OpenStack CLI
- Exercise 4 - Create a Snapshot of a VM
- Exercise 5 - Manage your own Persistent Volume
- Exercise 6 - Create your own Bucket and Object using the WebUI
- Exercise 7 - Upload Object to your Bucket using the s3cmd client

Go to exercise set B



Advanced topics and exercises

Terraform

Terraform is an open-source **infrastructure as code** software tool created by HashiCorp. Users define and provision data center infrastructure (machines, networks, storage, authentication, ...) using a declarative configuration language[1].

Terraform has four major commands:

```
$ terraform init
$ terraform plan
$ terraform apply
$ terraform destroy
```

[1]https://en.wikipedia.org/wiki/Terraform (Software)

Exercise Set C

Optional extra exercise

- Exercise 1 - Use Terraform to create a VM

Go to exercise set C

// --- // // # Feedback needed // // Please answer this course survey

// * Enrolment key: `cloudcomputing`

Documentation Links

- The cPouta main page: pouta.csc.fi
- These slides: https://pouta-course.a3s.fi/index.html
- These slides in PDF: https://pouta-course.a3s.fi/pouta-course-slides.pdf
- e-Lena Cloud computing fundamentals course
  - Enrolment key: `cloudcomputing`.
- Pouta documentation: docs.csc.fi
- Command line tools

Accounts

- Create CSC account
- Pouta access

Contact Us

If you have any problem, request, or you just need more information:

servicedesk@csc.fi

Jemal Tahir, Cloud System Specialist,
Jemal.Tahir@csc.fi

Alvaro Gonzalez, Cloud system specialist,
Alvaro.Gonzalez@csc.fi

CSC – IT Center for Science Ltd.

https://facebook.com/CSCfi

https://twitter.com/CSCfi

https://www.youtube.com/c/CSCfi

https://www.linkedin.com/company/csc--it-center-for-science