# Notes

RabbitMQ

* If a message arrives at a queue with no subscribed consumers, the message waits in the queue.
* When a queue has multiple consumers, messages received by the queue are served in a round-robin fashion to the consumers.
* Each message is sent to only one consumer subscribed to the queue.
* A message will survive a reboot, if its "delivery mode" option is set to persistent (value of 2), be published into a durable exchange and arrive in a durable queue
* Exchanges types : direct, fanout, topic
* Master and slave paradigm: channel.basic\_qos(prefetch=1)
* Channel.queue\_declare(exclusive=True)
* Classic binding(exchange, queue), possible to add a routing key to subscribe only to a subset of the msgs: channel.queue\_bind(exchange='direct\_logs' queue=queue\_name, routing\_key=severity)

MPI

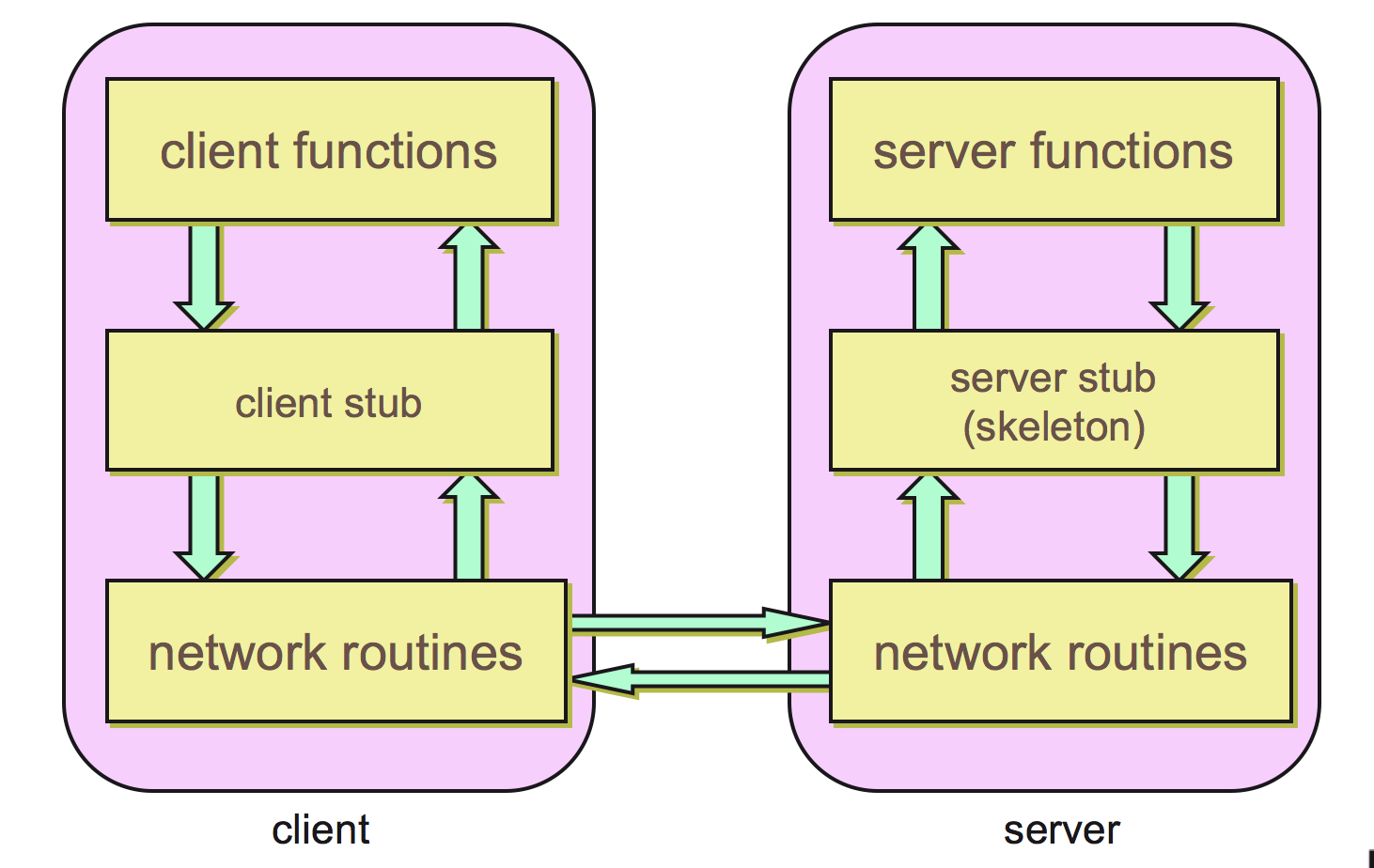
Message passing interface

MPI\_Comm\_size reports the number of processes.

MPI\_Comm\_rank reports the rank, a number between 0 and size-1, identifying the calling process

There is a default communicator whose group contains all initial processes, called MPI\_COMM\_WORLD.

RPC



Create stub functions to make it appear to the user that the call is local. Stub function contains the function’s interface

1. Client calls stub (params on stack)
2. Stub marshals params to net message
3. Network message sent to server
4. Receive message: send to stub
5. Unmarshal parameters, call server func
6. Return from server function
7. Marshal return value and send message
8. Transfer message over network
9. Receive message: direct to stub
10. Unmarshal return, return to client code

Sun RPC includes support for a single argument to a remote procedure.\*

Typically the single argument is a structure that contains a number of values (the parameters).

Each procedure is identified by:

* Hostname (IP Address)
* Program identifier (32 bit integer)
* Procedure identifier (32 bit integer)

Procedure Identifiers usually start at 1 and are numbered sequentially

Version Numbers typically start at 1 and are numbered sequentially.

Service number is coded on 32 bits. Possible user value range : 0x 2000 0000 à 0x 3fff ffff

at most one remote procedure within a program can be invoked at any given time.

If a 2nd procedure is called, the call blocks until the 1st procedure has completed

Sun RPC does not support reliable call semantics

At Least Once Semantics: if we get a response (a return value)

Zero or More Semantics: if we don't hear back from the remote subroutine

Clients ask a remote port mapper for the port number corresponding to Remote Program ID.

The portmapper is itself an RPC server!

The portmapper is available on a well-known port (111).

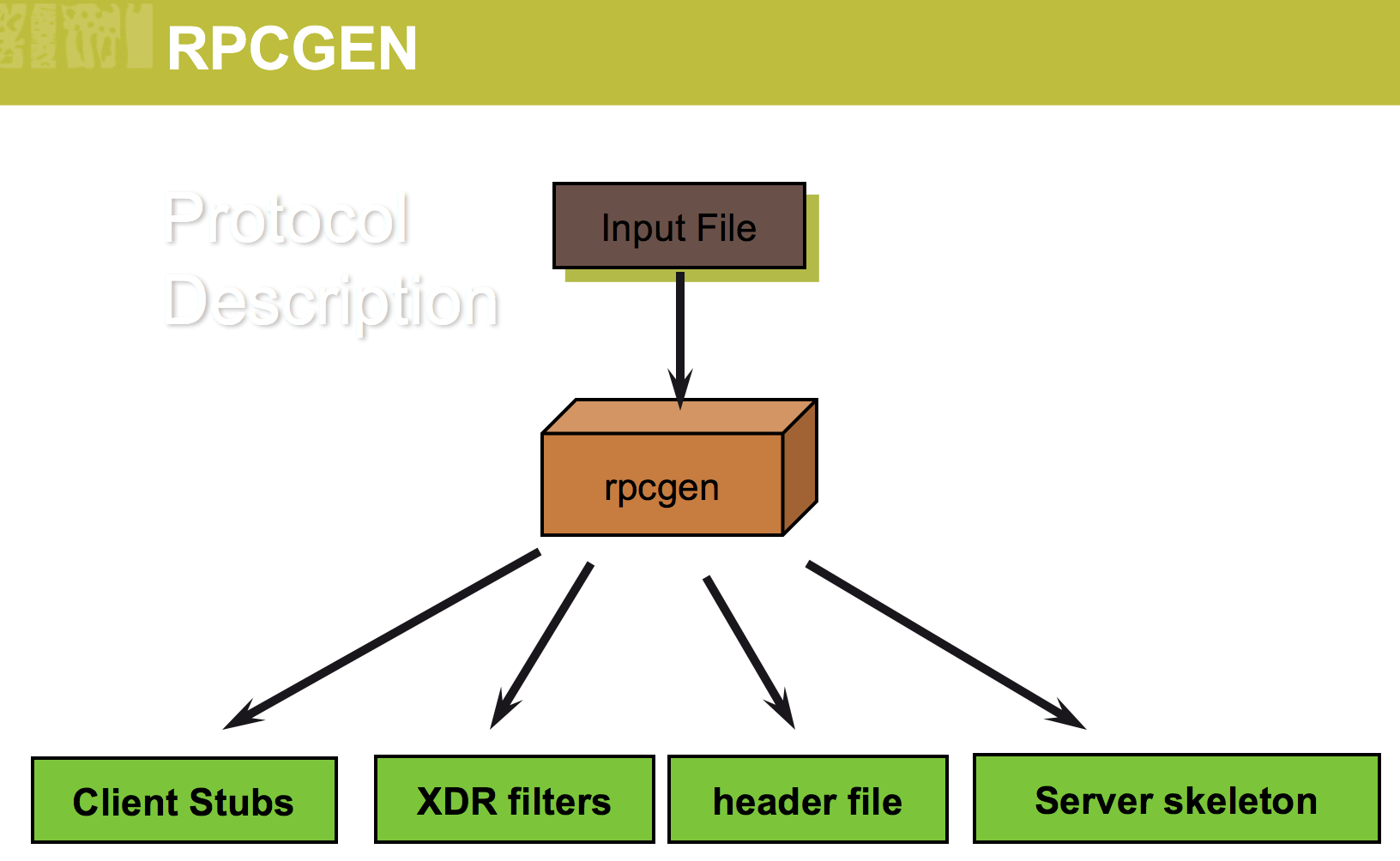
**T**here is a tool for automating the creation of RPC clients and servers.

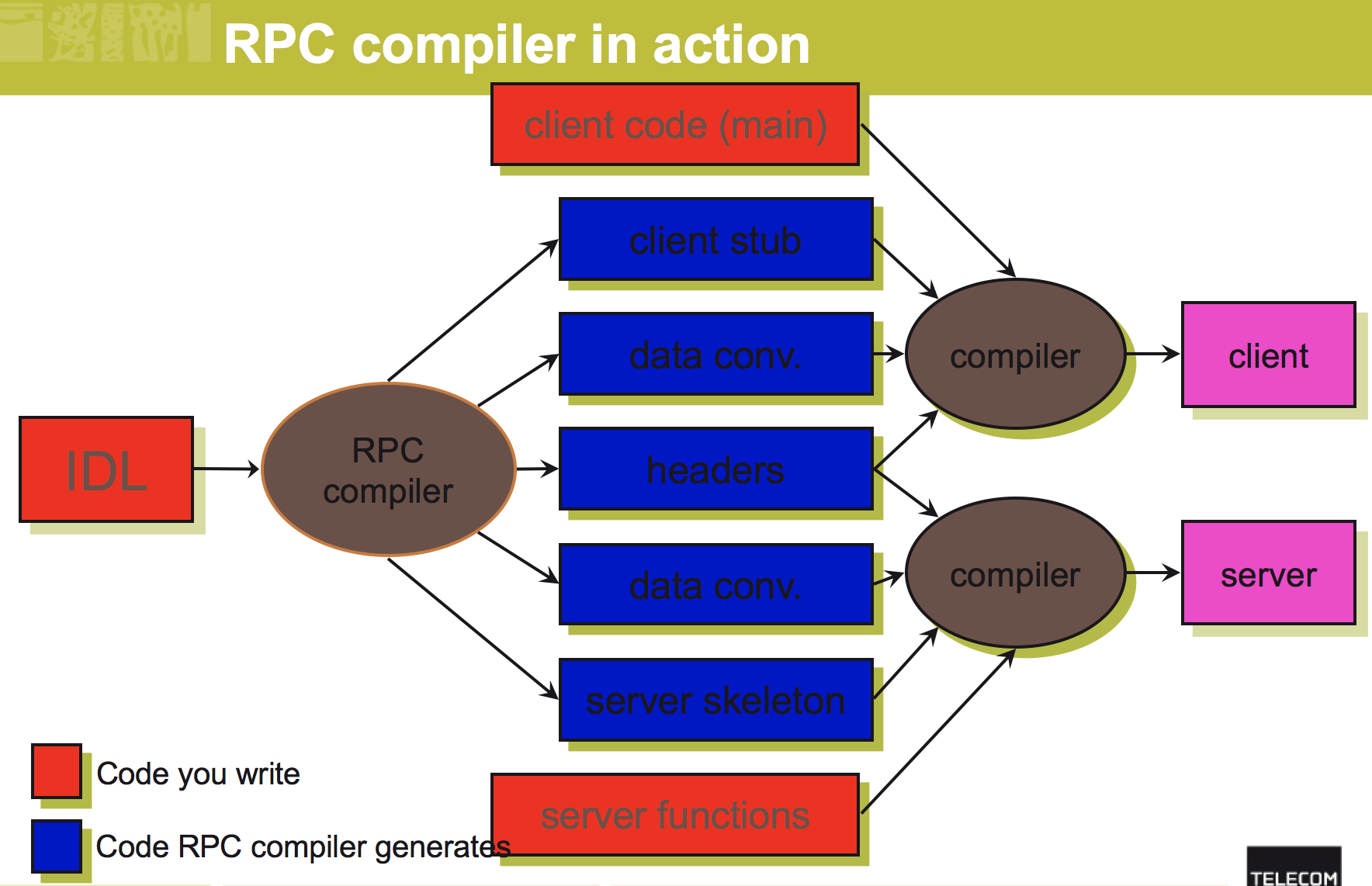
The input to rpcgen is a protocol definition in the form of a list of remote procedures and parameter types.

Ref:

<http://www.ques10.com/p/2159/explain-rpc-call-semantics-1/>

each invocation on a client results in the remote procedure being invoked zero or more times.





XDR : implicit typing, Powerful paradigm for creation and transfer of complex data structures

Iaas

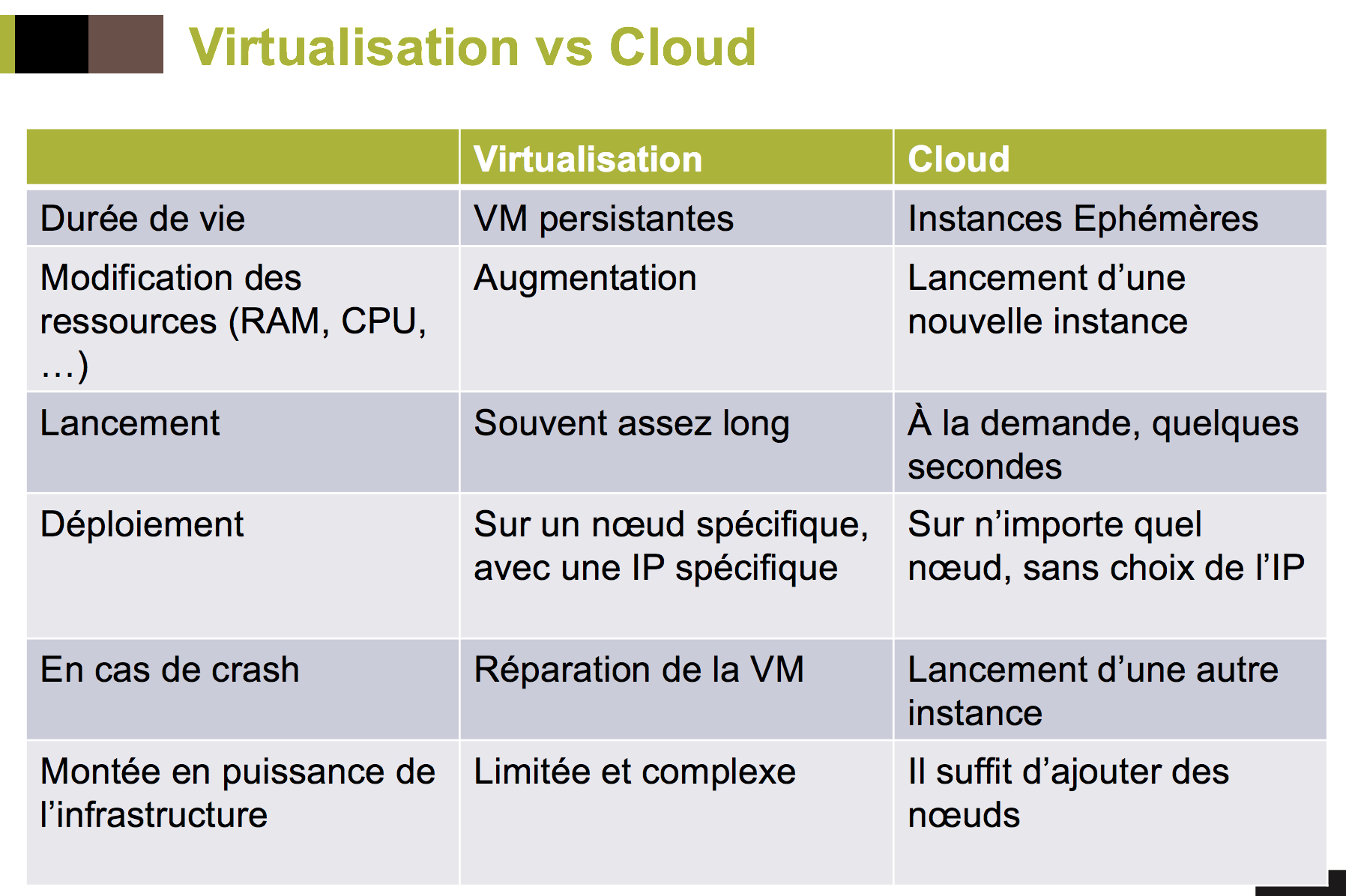
Cloud: public, privé, hybride, communautaire

Ip fixe, ip flottante, groupe de sécurité,

Quota : contrôle l’utilisation des ressources (vcpu, ram, fip, security groups,...) dans un projet

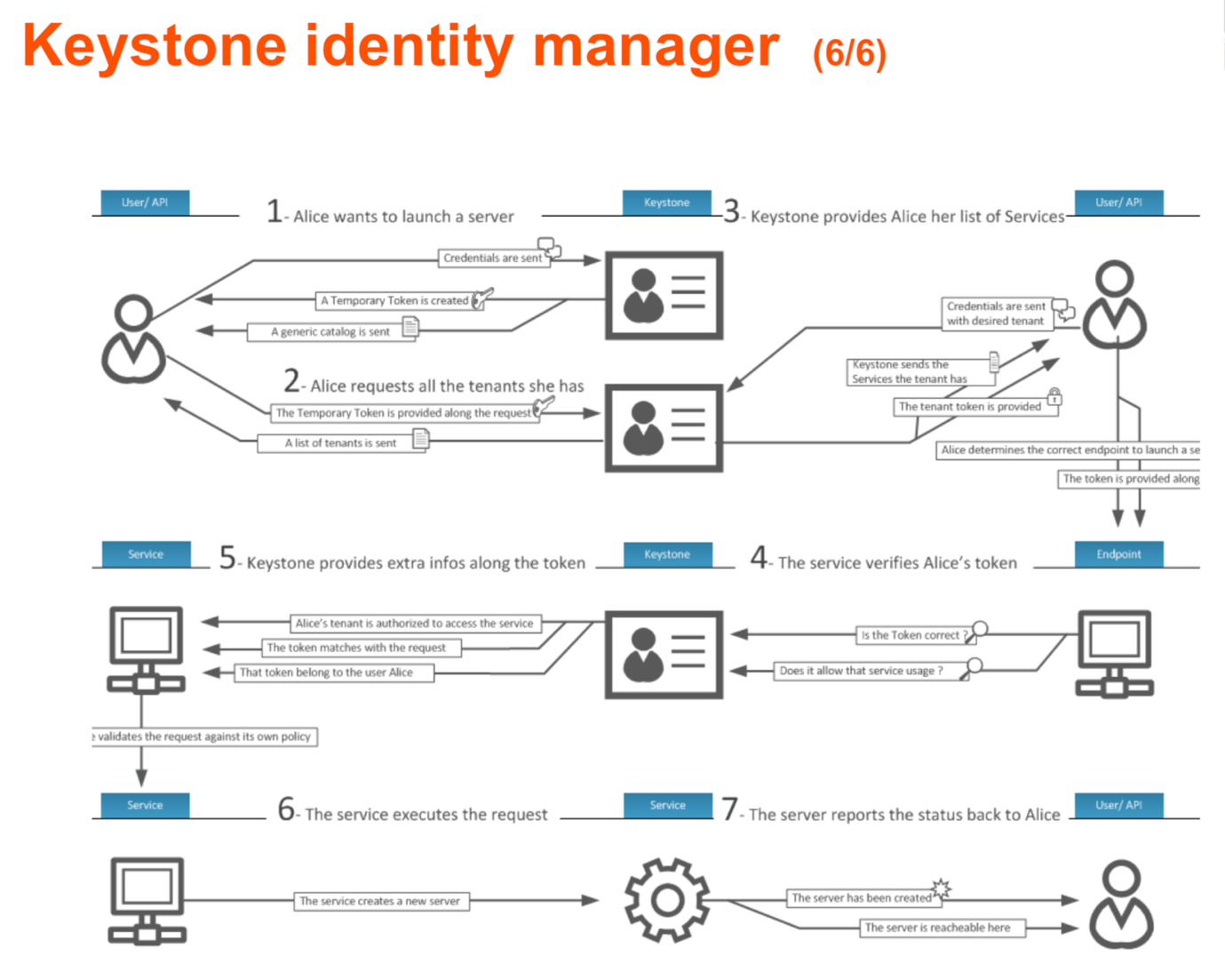
Paire de clés (Keypairs) : clé privée + clé publique permettant les connexions aux instances via SSH

Endpoint : URL permettant l’accès à une API. Un endpoint par service

  
virtualization et cloud: <https://www.elit-technologies.fr/qu-est-ce-que-virtualisation-serveurs/>

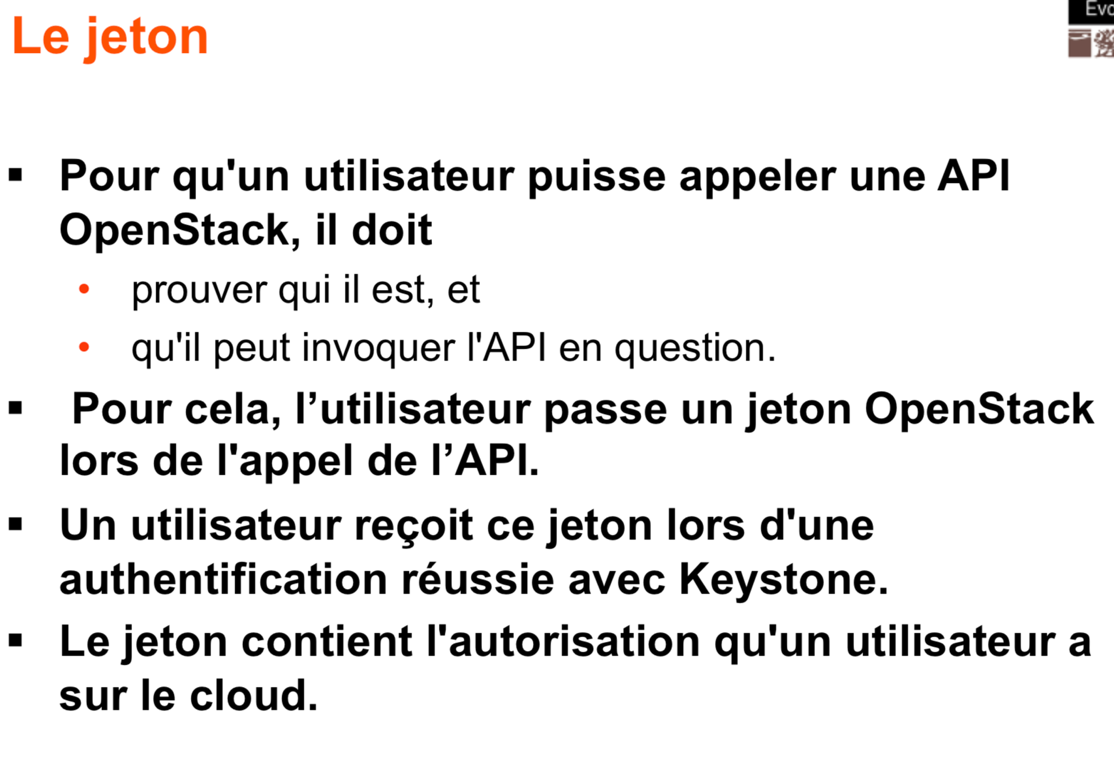
keystone

token: Un identifiant des droits associés à un utilisateur, scoped token (lié à un projet) et unscoped

token (non lié à un projet)

Un utilisateur demande un token (unscoped) en précisant ses infos de login : id et mot de passe. KeyStone lui renvoie un jeton. L’utilisateur demande quels sont ses projets (« tenants »)

L’utilisateur précise ensuite le projet et Keystone lui renvoie un token (scoped) et une liste de points d’entrées



Comment appeler un service ?

Connaitre son endpoint!

Utiliser les commandes préfixées par openstack

* https://docs.openstack.org/user-guide/cli-cheat- sheet.html
* Pensez à initialiser les variables d’environnement (ex nom du domaine, du projet, user id, ....)

Appeler le endpoint en direct avec une commande de type curl

* c’est pas très pratique....mais très explicite

RBAC : Role-Based Access Control