Client / Serveur avec UDP

UDP n’est pas connecté, c’est-à-dire on ne s’intéresse pas à si les paquets sont arrivés. L’UDP est utilisé pour les jeux vidéo (la position d’un personnage) ou bien des vidéos en direct. Car on se fiche qu’un paquet est perdu car le suivant actualise la situation.

Client / Serveur avec TCP

Sockeserver est utile pour gérer les clients, traiter le trafic. La méthode handle est une boucle infinie qui reçois l’info, la traite et la renvoie.   
Ensuite on le fait écouter en boucle : server.serve\_forever()  
Pour traiter plusieurs requêtes en même temps il faut utiliser asyncio pour utiliser le serveur de manière asynchrone

Socket avec Asyncio

Il faut définir la fonction qui traite les données. Asyncio va gérer l’ordre des tâches, on le fait boucler à l’infini

RPC & Pub/Sub

Remote Procedure Call

Publish / Subscribe

Accès SSH :

Paramiko : client ssh le plus utilisé sur python.

Netmiko : il connait plus de détail, il connait enable et le mode config alors que sur paramiko il faut y gérer à la main

NETCONFIG/YANG : librairie ncclient

Quand on a le modèle yang on connait toute la structure du modèle. Ainsi avec un diff on peut rapidement savoir ce qui a été modifié lors d’une modification.

L’interface NETCONF

gRPC :

* gRIBI : écire directement dans les tables (utiliser pour le cloud) inutile pour nous
* gNMI
* gNOI