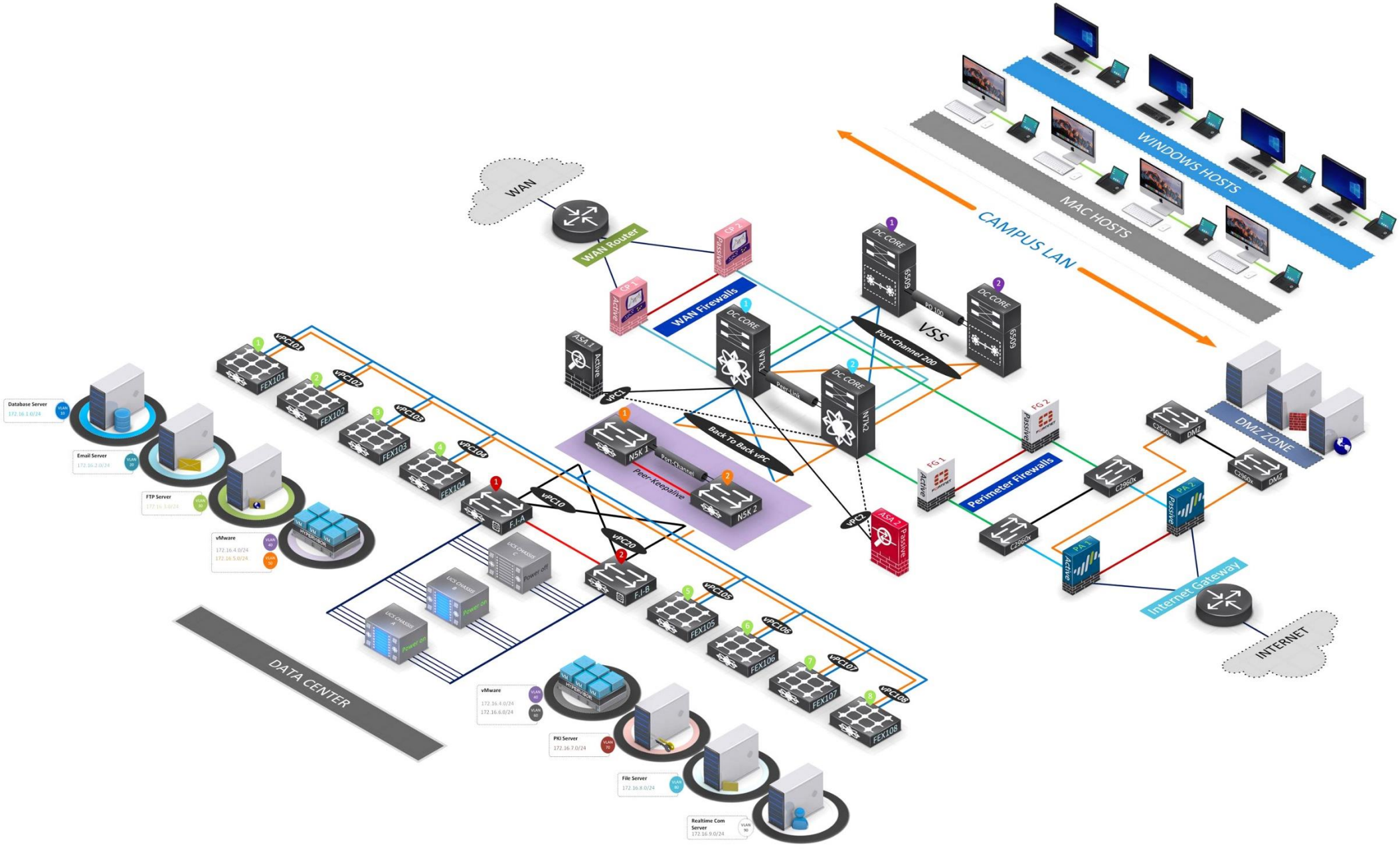
Architecture Clients/Serveur

Animé par Sylvain Labasse


WINDOWS HOSTS
MAC HOSTS
CAMPUS LAN
WAN
WAN Router
WAN Firewalls
ASA 1
Active
CSP 1
CSP 2
Passive
DC CORE
6509
VSS
Port-Channel 200
Peer-Link
CMLN
Back To Back vPC
N5K 1
N5K 2
Peer-Keepalive
Port-Channel
vPC1
vPC2
vPC10
vPC20
FEX101
FEX102
FEX103
FEX104
FEX105
FEX106
FEX107
FEX108
F.I-A
F.I-B
Database Server
172.16.1.0/24
Email Server
172.16.2.0/24
FTP Server
172.16.3.0/24
vMware
172.16.4.0/24
172.16.5.0/24
172.16.6.0/24
PKI Server
172.16.7.0/24
File Server
172.16.8.0/24
Realtime Com Server
172.16.9.0/24
DATA CENTER
FG 1
FG 2
Active
Passive
Perimeter Firewalls
C2960x
DMZ
DMZ ZONE
PA 1
PA 2
Internet Gateway
INTERNET

INTRODUCTION

En fin de module, vous saurez…

Ecrire un client TCP/IP quel que soit le protocole

Concevoir un serveur TCP/IP performant

Définir un protocole en respectant l'état de l'art

Opter pour une solution adaptée basée ou non sur TCP/IP

Pre-Requis

Public

Niveau bachelor informatique

Nécessaire

Bases de réseau

C#, Java, JS, PHP ou Python

My Learning Box – 18h

Support

Copie des slides, vidéos

Notes de cours

Réalisation des ateliers

Evaluation

Ateliers sur 4

QCM

0	1	2	3	4
Non remis	Hors sujet	<	=	>
		attentes		

L'ENVIRONNEMENT

Matériel

PC + Connexion Internet

Logiciel

IDE de votre choix

Compte GitHub ou GitLab

Architecture Clients/Serveur

Fondements du Clients/Serveur

- Enjeux
- Client
- Serveur

Fiabilisation

- Protocoles
- Cluster
- Variantes

FONDEMENTS

OBJECTIFS

Omniprésence des architectures Clients/Serveur

Socle et exigences techniques

Connexion et communication d'un client

Organisation du code pour un serveur TCP

FONDEMENTS DU CLIENTS/SERVEUR

➔ Enjeux

Client

Serveur

Synthèse

CLIENTS/SERVEUR

Communication

Asymétrique

Centralisée

Contraire : P2P

Figure 1 - src : nmedia

Déroulement

Serveur : Lancement, attente des connexions

Client : Connexion, envoi de requête

Serveur : Réponse à la requête, fermeture connexion (option)

Exemples

RFC

Telnet : FTP, TFTP, POP, IMAP, SMTP, HTTP, …

Fichiers : SMB, NFS, …

Réseau : DHCP, SNMP, LDAP, NIS, …

Applicatif

Bases de données, API Rest, SOAP, …

Média : SIP, RTP/RTCP, XMPP, …

Reseau

Transport

Nom/IPv4/IPv6 + Port

TCP : Connecté, ordre et arrivée

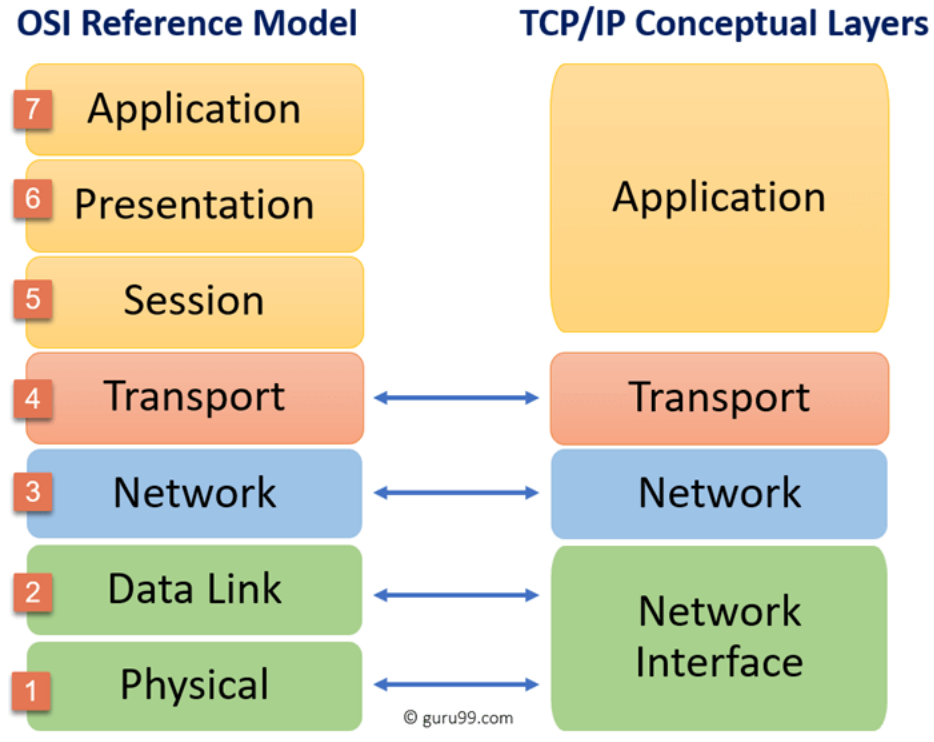
UDP : Non connecté, intégrité

Programmation

Gère Session + Présentation

TCP : 1 socket par interlocuteur

UDP : minimum 1 socket



Fondements du Clients/Serveur

- ✓ Enjeux
- ➔ Client
- Serveur
- Synthèse

Client

Déroulement

Création/Connexion du socket : Adresse, port

Préparation/envoi des données de requête

Lecture/traitement de la réponse

Autre requête ou fermeture de la connexion

Difficultés

Erreurs : (dé)connexion, timeout

Formats et évolutions

FONDEMENTS DU CLIENTS/SERVEUR

- ✓ Enjeux
- ✓ Client
- ➔ Serveur
- Synthèse

Serveur

Initialisation

Création du socket d'écoute

Attachement au port et la ou les interface(s) réseau (bind)

Mise en écoute (listen)

En boucle

Attente de connexion/récup. d'un socket de service (accept)

Création d'un thread pour com. avec le client sur socket service

Fermeture du socket de service en fin de communication (close)

Serveur - Concurrence

Echec d'attachement au port (bind)

Port réservé + droits insuffisants : 0-1023

Port utilisé par un autre serveur

Bon à savoir : Ports éphémères ≥ 32 768

Ressources partagées

Mémoire partagée entre threads (sauf pile des appels)

Vérifier si collection thread safe / mutex (cf fiabilisation)

Fondements du Clients/Serveur

- ✓ Enjeux
- ✓ Client
- ✓ Serveur
- ➔ Synthèse

Resume

Omniprésence des architectures Clients/Serveur

Socle et exigences techniques

Connexion et communication d'un client

Organisation du code pour un serveur TCP

FIABILISATION

OBJECTIFS

Structuration de la communication

Format universel des données

Résilience de la solution

Propositions et évolutions récentes

FIABILISATION

➔ Protocoles

Cluster

Variantes

Synthèse

Protocoles

Communication

HTTP : REST, récents : gRPC, anciens : SOAP, XML-RPC

TCP : Thrift, Corba/IIOP, XMPP, Pub/Sub : AMQP, MQTT

UDP : CoAP, MQTT-SN (sensor net.)

Indépendant : JSON-RPC, DDS (pub/sub)

Description d'interface

Avro, Protobuf, GraphQL, WSDL, Thrift (IDL), IDL (Corba)

JSON-RPC

Requête (id, réponse)/Notification (~~id, réponse~~)

"id" : …, "jsonrpc" : "2.0", "method" : …, "params" : …

params : by-position (tableau), by-name (objet)

Réponse

"jsonrpc" : "2.0", "result" : …, "id" : …

"jsonrpc" : "2.0", "error" : { "code": …, "message": …, "data": … },

"id" : null /* code : définis[1] ≤ -32000 */

[1] http://xmlrpc-epi.sourceforge.net/specs/rfc.fault_codes.php

IDENTITE

OAuth2

Accès res. aux app. tierses

Bearer Token/JWT

OpenID

Utilise OAuth2

Ajoute authentification

Partage d'identité / SSO

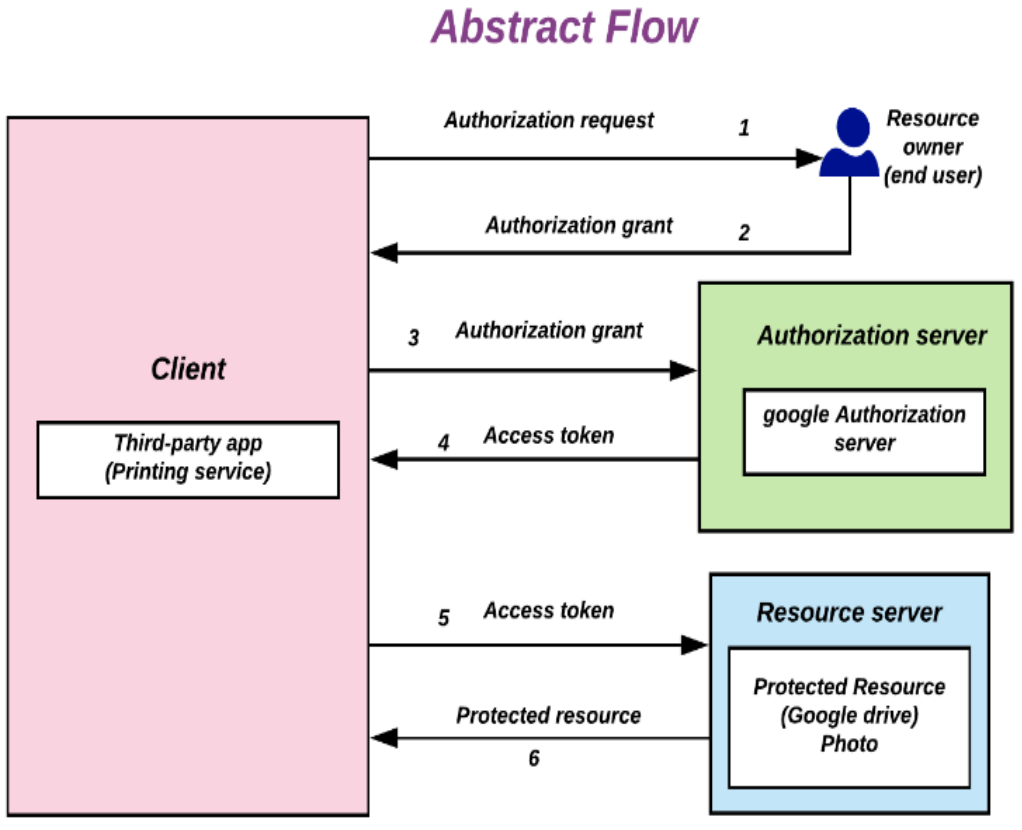


Figure 2 - src : wikipedia

FIABILISATION

- ✓ Protocoles
- ➔ Cluster
- Variantes
- Synthèse

CLUSTER

Principe

Redondance serveurs

Fiabilité/Rapidité

Maître/esclaves (conf. ou broadcast)

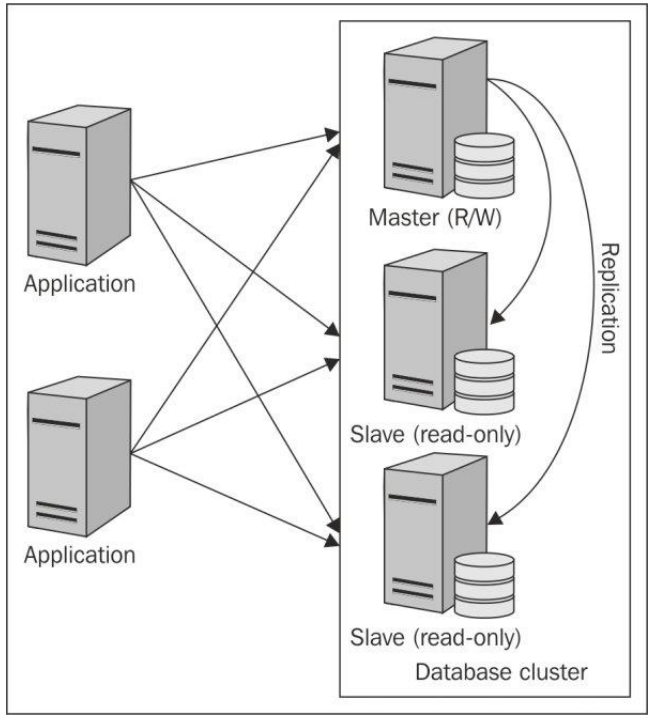


Figure 3 - src : Cassandra HA (Packt)

Implémentation

Distribution des connexions : Proxy ou maître lui-même

Esclaves : Double rôle client/serveur

Heartbeat et mécanisme d'élection d'un maître

Sharding : Répartition des requêtes par clés/groupes/volumes

FIABILISATION

- ✓ Protocoles
- ✓ Cluster
- ➔ Variantes
- Synthèse

TRANSPORT

UDP

Client/Serveur possible mais implémentation complète

Risque DDOS par amplification

Possibilité de multicast au sein d'un réseau local

Quic

Alternative TCP utilisant UDP

Multiplexage de connexions

Utilisé par Google et Facebook

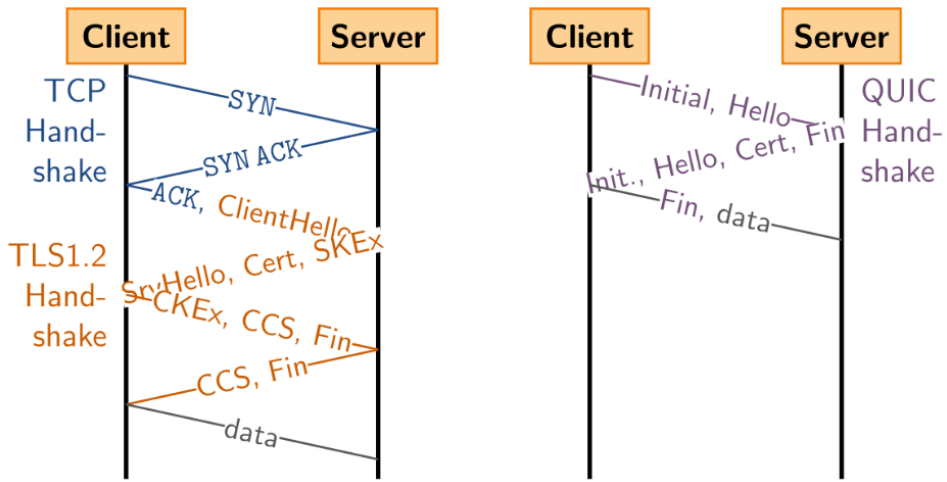


Figure 4 – Handshake TCP+TLS vs Quic src : wikipedia

SECURISATION

CRC, Codes de correction

Opération mathématique dont le résultat est joint au message

Si résultat égal : validation du message sinon rejet/correction

Chiffrement

Symétrique : 1 Clé de chiffrement/déchiffrement (AES)

Asymétrique : 1 clé chiffrement/1 clé déchiffrement (RSA)

FIABILISATION

- ✓ Protocoles
- ✓ Cluster
- ✓ Variantes
- ➔ Synthèse

Resume

Structuration de la communication

Format universel des données

Résilience de la solution

Propositions et évolutions récentes

BILAN

MAINTENANT, VOUS SAVEZ…

Ecrire un client TCP/IP quel que soit le protocole

Concevoir un serveur TCP/IP performant

Définir un protocole en respectant l'état de l'art

Opter pour une solution adaptée basée ou non sur TCP/IP