

LES PROTOCOLES DE COMMUNICATION

INTRODUCTION

PROTOCOLES INTERNET

Les protocoles de communication machine/machine jouent un rôle clé dans l'industrie 4.0, où l'interconnexion, la communication en temps réel et l'autonomie des machines sont essentielles.

IP (INTERNET PROTOCOL)

Le **protocole Internet** est la base des communications sur le réseau Internet. Il est responsable de la **transmission de paquets** entre les machines à travers les réseaux.

CARACTÉRISTIQUES

Caractéristiques	Description
Transmission de paquets	Envoie des données sous forme de paquets
Routage	Trouve le meilleur chemin pour envoyer les paquets
Adresse IP	Identifie les machines sur le réseau

IPV4

La version 4 du protocole **IP** est la version la plus utilisée actuellement.

AVANTAGES/INCONVÉNIENTS

Avantages

Facilité d'utilisation

Prise en charge native par la majorité des systèmes et équipements

Inconvénients

Nombre d'adresses limité

Pas de chiffrement intégré

POURQUOI IPV6 ?

La version 6 d'IP (IPv6) a été développée pour résoudre certains problèmes rencontrés avec l'IPv4, notamment la pénurie d'adresses.

STRUCTURE DES ADRESSES

Les **adresses IPv4** sont constituées de **4 octets** (32 bits) représentés en décimal séparés par des points. Par exemple : 192.168.1.1.

REPRÉSENTATION

Octet 1	Octet 2	Octet 3	Octet 4
192	168	1	1

Le protocole IPv6 utilise 128 bits pour les adresses, contre 32 bits pour les IPv4, permettant un plus grand nombre d'adresses uniques.

IPV6

La version 6 du protocole IP a été développée pour faire face à l'épuisement des **adresses IPv4**.

CARACTÉRISTIQUES D'IPV6

- Plus grande capacité d'adressage
- Simplification du format d'en-tête
- Meilleure prise en charge de la qualité de service (QoS)
- Amélioration de la sécurité et de la confidentialité

IPV4 VS IPV6

Comparaison	IPv4	IPv6
Adressage	32 bits	128 bits
Adresses	4 milliards	340 undécillions

STRUCTURE DES ADRESSES

Les adresses **IPv6** sont constituées de 8 groupes de 4 chiffres **hexadécimaux** séparés par des deux-points.
Par exemple : **2001:0db8:85a3:0000:0000:8a2e:0370:7334**.

REEMPLACEMENT

Les adresses IPv6 remplacent progressivement les adresses IPv4 pour faire face à l'épuisement de l'espace d'adressage IPv4.

NOTATION COMPACTE

Pour simplifier la notation des **adresses IPv6**, on peut :

- Omettre tous les zéros non significatifs d'un groupe : **2001:db8:85a3:0:0:8a2e:370:7334**.
- Remplacer une séquence consécutive de groupes de zéros par **::** :
2001:db8:85a3::8a2e:370:7334.

CLASSES D'ADRESSES

Classe	Plage	Utilisation
A	1.0.0.0 - 126.0.0.0	Grandes organisations
B	128.0.0.0 - 191.0.0.0	Moyennes entreprises
C	192.0.0.0 - 223.0.0.0	Petites entreprises, réseaux locaux (LAN)
D	224.0.0.0 - 239.0.0.0	Multicast
E	240.0.0.0 - 255.0.0.0	Réservé pour la recherche et le développement

MASQUES DE SOUS-RÉSEAU

Un **masque de sous-réseau** sert à déterminer la taille d'un **réseau local** et la plage d'adresses utilisables.
Par exemple : 255 . 255 . 255 . 0.

CLASSES

Classe d'adresse	Masque de sous-réseau par défaut
Classe A	255.0.0.0
Classe B	255.255.0.0
Classe C	255.255.255.0

Les masques de sous-réseau sont liés aux adresses IP et permettent de mieux gérer l'espace d'adressage.

TCP (TRANSMISSION CONTROL PROTOCOL)

ETABLISSEMENT DE CONNEXION

THREE-WAY HANDSHAKE

Le processus d'établissement de connexion avec **TCP** passe par un échange de trois messages : **SYN**, **SYN-ACK** et **ACK**.

ETAPES

Étape	Communication	Description
1	Client -> Serveur	Envoyer le message SYN
2	Serveur -> Client	Envoyer le message SYN-ACK
3	Client -> Serveur	Envoyer le message ACK

BUT

Le but du three-way handshake est de s'assurer que les deux côtés sont prêts à communiquer et ont reçu les informations nécessaires à l'établissement d'une connexion fiable.

TRANSFERT DE DONNÉES

SÉQUENCE ET ACCUSÉS DE RÉCEPTION

TCP utilise un système de **numéros de séquence** pour garantir l'ordre des paquets et des **accusés de réception** pour confirmer la réception.

FONCTIONNEMENT

TCP attribue un numéro de séquence unique à chaque octet de données et utilise des accusés de réception pour confirmer la bonne réception des données.

FERMETURE DE CONNEXION

La **fermeture de connexion TCP** implique un échange de messages **FIN** et **ACK** entre les deux machines.

UDP (USER DATAGRAM PROTOCOL)

STRUCTURE DES PAQUETS

Les paquets **UDP** sont plus simples que ceux de **TCP**, avec un en-tête de **8 octets** seulement et une charge utile de données.

STRUCTURE

Champs	Taille (octets)	Description
Source Port	2	Port source pour identification
Destination Port	2	Port de destination pour identification
Length	2	Longueur du paquet (en-tête + données)
Checksum	2	Somme de contrôle pour vérifier les erreurs

TRANSFERT DE DONNÉES SANS CONNEXION

UDP est un protocole **sans connexion**, ce qui signifie qu'il n'établit pas de connexion avant d'échanger des données.

UTILISATIONS COURANTES

UDP est souvent utilisé pour des applications nécessitant des **transmissions rapides** et avec **peu de latence**, comme l'audio et le **streaming vidéo**.

LES PROTOCOLES DE COMMUNICATION "MACHINE/MACHINE"

PROTOCOLES DE LA COUCHE APPLICATION

FTP (FILE TRANSFER PROTOCOL)

CONNEXION DE CONTRÔLE ET DE DONNÉES

- Deux connexions distinctes
 - Contrôle: échange de **commandes** et **informations**
 - Données: **transfert de fichiers**

COMMANDES DE TRANSFERT DE FICHIERS

- ***USER**:** authentification utilisateur
- ***PASS**:** mot de passe utilisateur
- ***PASV**:** mode passif
- ***RETR**:** télécharger fichier
- ***STOR**:** envoyer fichier
- ***LIST**:** liste des fichiers sur serveur

COMMANDES

Ces commandes sont généralement utilisées avec le protocole FTP pour transférer des fichiers entre un client et un serveur.

HTTP (HYPERTEXT TRANSFER PROTOCOL)

REQUÊTE ET RÉPONSE

- C'est un protocole de transfert hypertexte
- **Client** envoie **requête HTTP**
- **Serveur** répond avec un objet (texte, image, etc.) et un **code de statut**

MÉTHODES (GET, POST, ETC.)

- **GET**: demande d'un objet
- **POST**: envoie des données
- **PUT**: remplacement d'un objet
- **DELETE**: suppression d'un objet

Les méthodes PUT et DELETE sont moins couramment utilisées que GET et POST.

CODES DE STATUT

Les codes de statut sont des réponses standardisées qui indiquent le résultat d'une demande HTTP. Ils sont composés de 3 chiffres et sont regroupés en cinq classes en fonction du premier chiffre.

CODES DE STATUT

- 200: **OK**
- 404: **Not Found**
- 500: **Internal Server Error**

HTTPS (HTTP SECURE)

CHIFFREMENT SSL/TLS

- Protocole HTTP encapsulé dans une couche **SSL/TLS**
- Assure une connexion **chiffrée** et **sécurisée**

IMPACT

- **Évite l'interception** des données échangées
- **Authentifie** le serveur et le client (optionnel)

Les serveurs et les clients sont authentifiés via des certificats numériques pour vérifier l'identité de chaque partie.

DNS (DOMAIN NAME SYSTEM)

RÉSOLUTION DE NOMS DE DOMAINE

- Traduit les **adresses IP** en **noms de domaine** et vice-versa

DNS

DNS (Domain Name System) est utilisé pour réaliser cette résolution et donner quelques exemples courants, comme google.com traduit en 216.58.213.14 (exemple d'adresse IP).

SMTP (SIMPLE MAIL TRANSFER PROTOCOL)

ENVOI DE COURRIELS

- Échange de **commandes** et de **données** entre clients et serveurs

COMMANDES ET RÉPONSES

- **HELO**: identifie l'expéditeur
- **MAIL FROM**: adresse de l'expéditeur
- **RCPT TO**: adresse du destinataire
- **DATA**: corps du message
- **QUIT**: fermeture de la connexion

LES PROTOCOLES D'ÉCHANGE DE DONNÉES

JSON (JAVASCRIPT OBJECT NOTATION)

SYNTAXE ET STRUCTURE

JSON est un format de données léger qui utilise la notation d'objet JavaScript pour représenter les données structurées.

EXEMPLE

```
{  
    "key": "value",  
    "array": ["element1", "element2"],  
    "object": {  
        "key": "value"  
    }  
}
```

UTILISATION

JSON est couramment utilisé pour les échanges de données entre le client et le serveur. Les fichiers JSON ont généralement l'extension .json.

UTILISATION AVEC DES API

- **Facile à lire** et à écrire pour les **humains** et les **machines**
- Utilisé fréquemment pour les **API REST** pour communiquer de manière **structurée** entre le client et le serveur

XML (EXTENSIBLE MARKUP LANGUAGE)

SYNTAXE ET STRUCTURE

XML est un langage de balisage qui définit un ensemble de règles pour encoder des documents de manière à ce qu'ils soient lisibles à la fois par des **humains** et par des **machines**.

EXEMPLE

```
<root>
  <element attribut="valeur">
    <sub-element>valeur</sub-element>
  </element>
</root>
```

XML VS JSON

Les formats XML et JSON sont deux exemples courants de données structurées utilisées dans les API. JSON est généralement plus populaire en raison de sa simplicité et de sa facilité d'utilisation.