MINISTÈRE
DE L'ÉDUCATION
NATIONALE,
DE LA JEUNESSE
ET DES SPORTS

Liberté
Égalité
Fraternité

EDE ISI 1

SESSION 2021

CONCOURS EXTERNE DU CAPET, CAFEP ET TROISIÈME CONCOURS CORRESPONDANTS

Section : ÉCONOMIE ET GESTION

Option : INFORMATIQUE ET SYSTÈMES D'INFORMATION

COMPOSITION DE SCIENCES DE GESTION

Durée : 5 heures

Le lexique SQL, sans commentaire ni exemple d'utilisation des instructions, est autorisé.
La règle à dessiner les symboles informatiques est autorisée.
L'usage de tout autre ouvrage de référence, de tout dictionnaire et de tout matériel électronique (y compris la calculatrice) est rigoureusement interdit.

Si vous repérez ce qui vous semble être une erreur d'énoncé, vous devez le signaler très lisiblement sur votre copie, en proposer la correction et poursuivre l'épreuve en conséquence. De même, si cela vous conduit à formuler une ou plusieurs hypothèses, vous devez la (ou les) mentionner explicitement.

NB : Conformément au principe d'anonymat, votre copie ne doit comporter aucun signe distinctif, tel que nom, signature, origine, etc. Si le travail qui vous est demandé consiste notamment en la rédaction d'un projet ou d'une note, vous devrez impérativement vous abstenir de la signer ou de l'identifier.

INFORMATION AUX CANDIDATS

Vous trouverez ci-après les codes nécessaires vous permettant de compléter les rubriques figurant en en-tête de votre copie.

Ces codes doivent être reportés sur chacune des copies que vous remettrez.

CONCOURS EXTERNE DU CAPET

► **Enseignement public** :

Concours	Section/option	Epreuve	Matière
EDE	8031E	101	7392

► **Enseignement privé** :

Concours	Section/option	Epreuve	Matière
EDF	8031E	101	7392

TROISIEME CONCOURS DU CAPET

► **Enseignement public** :

Concours	Section/option	Epreuve	Matière
EDV	8031E	101	7392

Structure du sujet

Le sujet est composé de 3 dossiers

Dossier 1 – Management du système d'information (20 points)

Dossier 2 – Développement des applications (40 points)

Dossier 3 - Évolution de l'infrastructure réseau (40 points)

La documentation est structurée en fonction des dossiers

Dossier 1 :

- Document 1.1 : Extraits de documentations Microsoft Autopilot, Intune et Azure AD

Dossier 2 :

- Document 2.1 : Schéma logique de données
- Document 2.2 : Diagramme des classes métier de l'application TramFret
- Document 2.3 : Description des classes de l'application TramFret
- Document 2.4 : Synthèse de l'entretien avec le responsable de produit (*product owner*)

Dossier 3 :

- Document 3.1 : Schéma simplifié du réseau et plan d'adressage
- Document 3.2 : Commutateur hpe aruba 2530-8 (préconisé dans la baie de brassage d'un point d'interface)
- Document 3.3 : Principes de fonctionnement du Wi-Fi centralisé
- Document 3.4 : Les VLAN dédiés au Wi-Fi
- Document 3.5 : Fonctionnement du logiciel Heartbeat
- Document 3.6 : Extrait de la table de filtrage existante du routeur pare-feu RT-FW
- Document 3.7 : Exemple ajout de règle dans Snort

IMPRIMERIE NATIONALE – 21 0219 – D'après documents fournis

Présentation du contexte

TramFret

Figure 1 – Cargo Tram, Dresde - Allemagne (source : Wikimédia)

Le raccourcissement des délais de livraison des biens et services, souvent dans les 24h, pose la problématique du « *dernier kilomètre*[1] » afin de les acheminer au plus vite. Cette contrainte de temps doit répondre à des impératifs d'ordre environnementaux, économiques et de qualité. L'empreinte écologique du transport routier de marchandises pousse les grandes enseignes commerciales à se saisir de la problématique et à communiquer auprès des clients sur leurs actions vertueuses. La Commission européenne a décidé en 2018 de légiférer sur les normes applicables aux émissions de CO2 des poids lourds avec un objectif contraignant de 30% de réduction des émissions d'ici 2030.

Les métropoles sont très sensibles aux aspects environnementaux et sociétaux liés au transport des biens et des personnes, qui sont pour celles-ci des enjeux politiques. La circulation des véhicules de livraison multiplie les problématiques d'embouteillages et de pollution auxquels les usagers sont de plus en plus sensibilisés.

Pour les grandes enseignes commerciales, la logistique urbaine est donc un point central de la chaîne d'approvisionnement. Elle ne se limite pas à la simple gestion des marchandises mais elle doit prendre en compte de nombreux paramètres comme notamment le confort des citoyens, la gestion des infrastructures et la livraison du dernier kilomètre. Cette étape est la dernière de la chaîne logistique qui souffre le plus de la congestion au sein des villes et à ce titre, elle nécessite une attention particulière afin de coordonner les différentes activités d'une aire urbaine.

C'est dans ce cadre qu'une métropole du centre de la France a réfléchi à des solutions afin d'obtenir la meilleure coordination possible des différents modes de transports. Au niveau du trafic routier, la métropole est le milieu urbain le plus congestionné du département. Le camionnage est le mode de transport le plus utilisé et contribue fortement à la congestion aux abords de la ville. De plus, le camionnage nécessite des aménagements et des mesures spéciales au sein de la métropole, ce qui cause des désagréments dans la vie des citadins. En outre, les élus ont peu d'emprise sur le transport routier de marchandises, sauf à mettre en place des conditions favorables au bon fonctionnement des systèmes de transport de marchandises de façon harmonieuse avec le transport des voyageurs.

1 Le **dernier kilomètre** est une expression désignant l'ensemble des agents, opérations et équipements associés et mis en œuvre dans les derniers segments de la chaine de distribution finale des biens ou services (source Wikipédia)

La métropole s'est lancée dans un projet de réseau structurant qui vise à établir une cohésion entre les différents modes de transports comme l'autobus, le tramway, la voiture et le vélo. Plus particulièrement, le projet TramFret a pour objectif d'utiliser les voies du tramway pour faire circuler les marchandises depuis la périphérie de la métropole, les grands pôles routiers et différents sites de la ville. Sans impacter le trafic voyageur, des rames de tramway vont transporter les marchandises sur des arrêts dédiés.

L'idée d'utiliser les rails urbains n'est pas nouvelle, l'un des plus anciens projets (2002) de tramway de marchandises est le CargoTram utilisé par un constructeur automobile pour son usine de Dresde. En France, un groupe de supermarchés a déjà expérimenté, en 2017 à Saint-Étienne, la livraison de marchandises en milieu urbain par tramway.

Pour la métropole, une ligne en cours de construction, qui comportera à la fois des stations « voyageurs » et des stations « fret » traversera toute l'agglomération. Les tronçons seront communs, les différents types de rames circuleront de façon à ce que le trafic voyageur soit toujours privilégié. Depuis un **point d'injection**, les rames achemineront des chariots de marchandises à des **points d'interface** destinés à l'approvisionnement des commerces de détails (ou points de vente). Pour les grandes enseignes commerciales comme pour la métropole, l'objectif est de limiter le transport routier de marchandises intramuros.

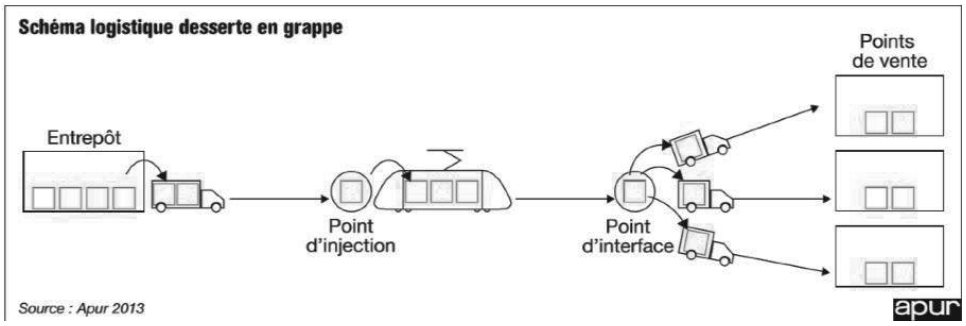


Figure 2 – Exemple de fonctionnement d'un « tram-cargo » (source : Apur)

En plus des plateformes logistiques existantes situées de part et d'autre de la ligne de tramway (au point d'injection), quatre stations « fret » (points d'interface) seront réparties à divers endroits de la ligne qui traverse la métropole et constitueront des pôles d'échange vers les points de vente. Les rames « fret » circuleront sur des créneaux horaires définis suite à des études sur la fréquentation des transports « voyageur ».

Les grandes enseignes de la métropole sont impliquées dans ce projet à la fois pour l'acheminement dans les entrepôts à proximité des points d'injection et la prise en charge depuis les stations de fret. Dans ce cadre, un groupement d'intérêt économique (GIE) composé des enseignes (telles qu'Auchan, Carrefour, Intermarché, Casino …) permet de mutualiser les moyens matériels et humains ainsi que les rames transportant les marchandises. La régie de transports urbains de la métropole est également membre de ce GIE auquel elle fournit l'infrastructure ferroviaire et ouvre son système d'information afin de coordonner et fluidifier le transport de voyageurs et de marchandises.

Dossier 1 : Management du système d'information

Ouverture du système d'information de la régie de transports urbains dans le cadre du GIE

Le système d'information de la régie de transports urbains de la métropole est actuellement entièrement articulé autour du trafic voyageur et devra intégrer celui lié au fret des marchandises. En tant que membre du GIE, les échanges d'informations entre la régie et les enseignes seront nécessaires pour permettre l'acheminement du fret.

Aujourd'hui, plusieurs applications sont utilisées, pour la gestion du trafic voyageur, au sein d'une solution *cloud* de type logiciel en tant que service (*software as a service - SaaS*). Ce sont principalement celles de gestion commerciale et comptable (SAP), de gestion de flotte, de gestion de ressources humaines et de gestion des infrastructures.

Travail à faire

1.1 Expliquer, dans le cadre du GIE, les opportunités et risques qui sont liés à l'ouverture du système d'information de la régie de transports urbains de la métropole.

1.2 Donner les avantages et inconvénients de recourir à une solution de type *SaaS*.

Solutions applicatives mises en œuvre

Les personnels des plateformes logistiques (point d'injection) seront chargés du contrôle qualité lors de la réception des marchandises, de la détection des produits dangereux, de la préparation et de l'ordonnancement des livraisons vers les points d'interface.

Les personnels des points d'interface assureront le déchargement des chariots transportés dans les rames de tramway et l'expédition des marchandises pour le dernier tronçon. Ils seront chargés de la collecte et des retours des chariots vides vers les points d'injection.

Pour réaliser ce travail, les personnels des plateformes logistiques aux points d'injection utiliseront au moins trois applications métier : TransQual pour la gestion de la qualité des réceptions, TransRisk pour la détection des produits dangereux et TransOrd pour l'ordonnancement des expéditions.

Seuls les personnels habilités qualité pourront utiliser l'application TransQual. Ces personnels pourront se déplacer pour des contrôles sur les points d'interface.

L'application *web* dédiée au fret, utilisée par les personnels sur tout le parcours des chariots, s'appelle TramFret. Celle-ci est encore en cours de développement et permettra *in fine* de suivre et gérer l'approvisionnement des points de vente. L'affectation des personnels dans un point d'interface, pourra être différente chaque jour, en fonction de la charge de travail prévue.

Le matériel mobile utilisé pour permettre aux employés de réaliser ces tâches sera affecté soit à une plateforme logistique soit à un point d'interface.

Lors de sa prise de fonctions le matin, chaque employé prendra une tablette et devra retrouver les applications nécessaires à la réalisation de son travail.

Le GIE pense choisir une solution d'annuaire basée dans le *cloud*.

Travail à faire

1.3 Expliquer dans le cadre du GIE, l'intérêt de la solution Microsoft Azure AD.

1.4 Préciser l'incidence du choix des solutions Microsoft Autopilot et Intune sur le coût total de possession du patrimoine informatique.

Dossier 2 : Développement des applications

Les enseignes qui utiliseront le transport de fret seront désignées comme étant les « clients ». L'application *web* TramFret sera utilisée pour tout ce qui concerne l'approvisionnement des points de vente des enseignes clientes à partir des entrepôts, via les rames de tramway. Chaque client, une enseigne commerciale qui utilise les points d'injection et d'interface, prépare ses propres chariots puis les dépose à un point d'injection où ils sont pris en charge et déclarés sur TramFret. Depuis un point d'injection, les rames acheminent ensuite les chariots vers les points d'interface destinés à l'approvisionnement des points de vente.

Lorsque le chariot d'un client est déposé à un point d'injection, il est enregistré et son statut est mis à jour. Il est théoriquement affecté au point d'interface du client. Néanmoins, pour certaines raisons, il se peut qu'il soit affecté à un autre point d'interface. On mémorise donc dans la base de données le point d'interface réel auquel le chariot est destiné.

Le statut d'un chariot permet d'assurer son suivi tout au long de sa prise en charge sur le réseau. Les différents statuts possibles pour un chariot sont :

- 1 : en route vers le point d'injection
- 2 : refusé (chariot arrivé au point d'injection en mauvais état, ...)
- 3 : arrivé au point d'injection
- 4 : transport planifié
- 5 : transport en cours
- 6 : arrivé au point d'interface
- 7 : livré
- 8 : en incident

Suivi des chariots et des clients

Dans une démarche d'amélioration continue, l'application TramFret doit permettre de disposer de tableaux de bord permettant d'optimiser le suivi des chariots et celui des clients. La construction de ces tableaux de bord nécessite d'extraire de la base de données des informations sur les clients et les chariots transportés, et plus particulièrement :

A. Identifiant, dimensions et poids des chariots pesant plus de 100 kg.
B. Nombre de chariots en incident.
C. Identifiant, dimensions et poids des chariots livrés pour le client « Deckea SA ».
D. Identifiant, nom et adresse des points d'interface auxquels moins de 10 clients sont rattachés.
E. Identifiant, dimensions et poids des chariots dont le point d'interface ne correspond pas au point d'interface par défaut du client concerné.

Travail à faire

2.1 Rédiger, dans un langage de communication associé à une base de données, les requêtes correspondant aux informations recherchées notées ci-dessus A, B, C, D, E.

Traçabilité du transport des chariots

Au-delà du suivi en temps réel des chariots que permet la base de données, l'application TramFret doit permettre de mettre en place une traçabilité du transport des chariots sur le réseau.

La solution retenue consiste à enregistrer, dans un fichier d'évènements au format JSON, chaque changement de statut d'un chariot.

Travail à faire

2.2 Proposer et décrire une solution à mettre en œuvre au niveau de la base de données pour journaliser automatiquement chaque changement de statut d'un chariot.

2.3 Présenter les avantages de stocker les données dans un format tel que JSON pour enregistrer les évènements de changement de statut des chariots.

Amélioration de la livraison aux points de vente

Certains clients disposent de plusieurs points de vente dispersés dans la ville avec des superficies différentes. Ils souhaitent pouvoir faire livrer automatiquement chaque chariot au point d'interface le plus proche du point de vente concerné.

Travail à faire

2.4 Expliquer en quoi la base de données actuelle ne permet pas de répondre à ce besoin.

2.5 Proposer une solution, en utilisant la représentation de votre choix, permettant de répondre à ce besoin.

Développement des fonctionnalités de suivi des chariots dans l'application TramFret

Quelques développements sont encore nécessaires sur l'application TramFret, notamment la gestion des événements sur les chariots.

Plusieurs récits utilisateurs (*user story*) ont été répertoriés afin de décrire les besoins et déterminer les différents développements à réaliser.

Le diagramme de classes et la description des classes nécessaires au traitement de cette partie figurent dans les ressources documentaires de ce dossier.

Les membres du GIE ont tous émis leurs besoins sur les fonctionnalités attendues de l'application, conduisant à l'élaboration des récits suivants :

Soit le récit utilisateur (*user story*) suivant :

En tant que client je souhaite connaître le statut d'un chariot afin de prévoir son arrivée en point de vente.

La méthode suivante devra être implémentée :

Client::getLibelleStatutChariot(Chariot unChariot) qui retourne le libellé du statut du chariot passé en paramètre.

Travail à faire

2.6 Écrire la méthode correspondant au récit utilisateur.

2.7 Écrire le constructeur de la classe PointInjection tel qu'il est présenté dans le document 2.3.

Soit le récit utilisateur suivant :

En tant que superviseur, je souhaite pouvoir mettre à jour le statut de l'ensemble des chariots concernés par un incident de circulation.

La méthode majChariotsIncident de la classe SuiviController devra mettre à jour tous les chariots impactés par l'incident, c'est-à-dire ceux qui n'ont pas encore été livrés. La méthode retournera le nombre de chariots mis à jour.

Le paramètre lesTransports contient la liste de tous les transports impactés par l'incident en cours.

Si le paramètre lesTransports ne contient aucun élément, une exception de la classe TramfretException est déclenchée avec le message « Aucun chariot à mettre à jour suite à l'incident ».

Travail à faire

2.8 Écrire la méthode majChariotsIncident de la classe SuiviController correspondant à ce récit utilisateur.

Soit le récit utilisateur suivant :

En tant que responsable d'un point d'injection, je souhaite obtenir la liste des chariots d'un client, non affectés sur leur plateforme théorique.

En effet, lors de la création d'un chariot, on lui affecte le point d'interface rattaché au client destinataire. Pour diverses raisons, notamment une indisponibilité temporaire, il se peut qu'on soit amené à changer le point d'interface automatiquement attribué.

La méthode getChariotsReAffectes de la classe PointInjection devra retourner la liste des chariots qui ont été affectés sur un autre point d'interface que le point d'interface dont dépend le client destinataire.

Travail à faire

2.9 Écrire la méthode getChariotsReAffectes de la classe PointInjection correspondant à ce récit utilisateur.

Soit le récit utilisateur suivant :

En tant que client je souhaite pouvoir afficher dans mes applicatifs le statut actuel d'un chariot afin de planifier la réception des marchandises.

Travail à faire

2.10 Rédiger une courte note à destination du responsable de produit (*product owner*) proposant une solution permettant à un client externe au SI du GIE d'accéder au statut d'un chariot.

Modélisation de la fonctionnalité de planification des transports

Le responsable de produit (*product owner*) vient de rajouter le récit utilisateur suivant dans liste des travaux (ou *backlog*) à réaliser sur le développement en cours, avec une priorité élevée :

En tant que superviseur je souhaite planifier les transports afin de garantir les meilleurs délais de livraison en toute sécurité pour les matériels et les personnes.

Vous êtes chargé de participer au développement de cette nouvelle fonctionnalité, en vous appuyant sur la synthèse de l'entretien avec le responsable de produit (document 2.4).

Travail à faire

2.11 Modifier le diagramme des classes permettant de prendre en compte cette nouvelle fonctionnalité.

Dossier 3 : Évolution de l'infrastructure réseau

Conception des locaux techniques des points d'interface

Les tramways transportant du fret empruntant les mêmes voies que le trafic voyageur, le centre de régulation du trafic devra connaitre l'emplacement et l'heure de départ prévue du tramway du point d'interface. Ainsi, le fonctionnement des applications de fret devra avoir un taux de disponibilité de 100% dans la plage de fonctionnement des tramways voyageurs (4h40 – 0h40).

Chaque point d'interface possèdera un local technique dans lequel se trouvera une baie (ou armoire) de brassage où arrivera une double liaison fibre optique en provenance du réseau métropolitain (*metropolitan area network - MAN*).

Le réseau Wi-Fi de chaque point d'interface sera connecté à des commutateurs (modèle décrit dans le dossier documentaire).

Travail à faire

3.1 Argumenter sur le choix du commutateur proposé.

3.2 Proposer un schéma de connexion physique à appliquer dans chaque baie de brassage de chaque point d'interface en précisant les précautions à prendre pour éviter une tempête de diffusion.

Conception de l'architecture sans-fil

Les personnels des points d'interface utiliseront une application métier TramFret, qui est encore en cours de développement, via des tablettes connectées en Wi-Fi.

Par ailleurs, il sera créé un réseau sans-fil dédié au personnel de maintenance du système.

Le réseau sans-fil actuel du trafic voyageur comporte un ensemble de points d'accès Wi-Fi répartis dans les wagons. Ces points d'accès sont coordonnés par des contrôleurs Wi-Fi Cisco 2504.

Ces contrôleurs n'étant plus disponibles à l'achat actuellement, il a été décidé d'acheter le contrôleur sans fil Catalyst 9800 pour les deux nouveaux réseaux sans-fil, dédiés respectivement aux personnels des points d'interface et aux personnels de maintenance du système.

Pour des raisons de disponibilité, deux bornes seront connectées dans chaque point d'interface (une sur chaque commutateur), configurées à l'identique hormis le canal utilisé.

Un contrôleur, positionné au niveau du site central (qui sera également redondé), suffit pour l'ensemble des bornes.

Ce dernier supporte la fonctionnalité « FlexConnect » (décrite dans le dossier documentaire), qui désigne la capacité d'un point d'accès (*access point* - AP) à décider si le trafic des appareils sans fil est placé directement sur le réseau au niveau du point d'accès (commutation locale), ou si le trafic est centralisé vers le contrôleur Catalyst 9800 (commutation centrale).

Il n'a pas encore été décidé si cette fonctionnalité allait être activée ou non.

Travail à faire

3.3 Préciser, dans un tableau conforme à la présentation ci-dessous, en justifiant vos propositions, les éléments de configuration des ports sur lesquels vont être connectés les bornes et les contrôleurs, ainsi que les ports d'interconnexion entre le site central et les points d'interface dans le cas où :

a. La fonctionnalité FlexConnect n'est pas activée.

b. La fonctionnalité FlexConnect est activée.

Ports	Ports étiquetés avec le protocole 802.1Q (OUI/NON)	Numéro (s) éventuel (s) de VLAN configuré
Sur lequel est branché le contrôleur		
Sur lequel est branchée la borne légère		
D'interconnexion entre le commutateur du point d'interface et celui du site central		

3.4 Proposer, en justifiant vos choix, le canal configuré sur chacune des deux bornes.

Tolérance de panne de l'application *web* TramFret

Lorsque les tramways arrivent dans les points d'interface, des techniciens sont chargés de réaliser diverses opérations de vérification et d'enregistrement des chariots. Ces opérations sont réalisées à l'aide de tablettes connectées en Wi-Fi à l'application *web* TramFret. Cette application est hébergée dans la zone démilitarisée (DMZ).

Afin d'augmenter la disponibilité de cette application, la direction met à l'étude la solution Heartbeat**.**

L'étude porte sur la configuration d'un *cluster* de deux serveurs *web* :

Nom DNS du serveur web	Adresse IP réelle	Adresse IP virtuelle
web-trans1	10.20.30.11/24	10.20.30.14/24
web-trans2	10.20.30.12/24	

Chaque serveur *web* hébergera l'application TramFret. Le second serveur *web* sera donc un serveur de secours si le serveur actif tombe en panne. Les données de l'application sont hébergées sur d'autres serveurs en dehors de la DMZ. Chaque serveur *web* sera sous l'application Apache avec une carte réseau nommée enp0s2 (système Ubuntu).

Travail à faire

3.5 Expliquer pourquoi la configuration de la solution Heartbeat doit comporter la configuration d'une adresse IP virtuelle.

3.6 Proposer un paramétrage des fichiers de configuration *ha.cf* et *haresources* permettant de répondre au besoin de disponibilité de l'application TramFret.

Gestion des flux d'administration des serveurs

Le site central de gestion des tramways comporte un VLAN dédié à la DSI. Les techniciens de la DSI doivent pouvoir administrer le serveur hébergeant l'application TramFret depuis leur poste, via le protocole SSH, et via l'interface *web* d'administration de l'application qui accepte seulement des connexions HTTPS. La DSI doit pouvoir aussi tester la connectivité vers les serveurs *web* de cette application. Toutes les machines de la DSI doivent pouvoir réaliser ces tâches d'administration excepté la machine du secrétariat qui a pour adresse IP 192.168.50.10.

Travail à faire

3.7 Expliquer le rôle de la règle n°3 de la table de filtrage existante du routeur pare-feu RT-FW.

3.8 Écrire les règles de filtrage à ajouter sur le routeur pare-feu RT-FW afin de permettre à la DSI de réaliser les tâches d'administration de l'application TramFret.

Analyse de sécurité : attaques et vulnérabilités

Le réseau de chaque point d'interface est maintenant opérationnel.

Afin de se prémunir d'éventuelles futures menaces, un test d'intrusion est prévu sur les éléments d'un point d'interface et ceux du site central de gestion de tramway.

Ce test d'intrusion, ou encore *pentest,* consistera à analyser les réseaux visés en jouant le rôle d'un attaquant et tenter de trouver un moyen de s'infiltrer dans ceux-ci.

Lors de ce *pentest*, l'attaquant testera :

- Le déni de service sur les serveurs de la DMZ.
- Les attaques *web* par « *cross-site scripting* » (ou XSS), les injections SQL sur les serveurs *web*.
- La prise de contrôle à distance du contrôleur Wi-Fi.
- La prise de contrôle à distance des commutateurs du point d'interface.

Travail à faire

3.9 Présenter le principe de fonctionnement de chaque attaque et évaluer le risque qu'elle pourrait engendrer en cas de succès.

3.10 Présenter les précautions à prendre pour se prémunir de chaque attaque.

L'outil de détection d'intrusions Snort est installé dans le réseau de la DMZ.

Un premier scan de vulnérabilité a permis de remonter plusieurs alertes sur les serveurs de la DMZ.

```
# cat /var/log/snort/alert
[**] [1:0001:0] nouvelle connexion FTP [**]
[Priority: 0]
08/05-04:23:35.977348 10.20.30.35:46234 -> 10.20.30.14:21
[**] [1:0002:0] tentative de connexion SSH [**]
[Priority: 0]
08/05-04:23:45.977358 10.20.30.35:44644 -> 10.20.30.14:22
```

Travail à faire

3.11 Écrire la règle Snort qui lancera une alerte pour toute tentative de connexion SSH sur un serveur de la DMZ.

3.12 Proposer des solutions complémentaires à Snort qui permettraient de prévenir les éventuelles futures attaques.

Documentation du dossier 1

Document 1.1 : Extraits de documentations Microsoft Autopilot, Intune et Azure AD

Microsoft Autopilot

Windows AutoPilot est un ensemble de technologies utilisées pour configurer et préconfigurer de nouveaux appareils et les préparer pour une utilisation productive. Vous pouvez également utiliser Windows AutoPilot pour réinitialiser, réaffecter et récupérer des appareils. Cette solution permet à un service informatique d'obtenir les éléments ci-dessus avec peu ou pas d'infrastructure à gérer, avec un processus facile et simple.

Microsoft Intune

Gérez en toute sécurité les appareils iOS, Android, Windows et macOS à l'aide d'une solution unique de gestion des points de terminaison. Simplifiez et automatisez le déploiement, l'approvisionnement, la gestion des stratégies, la livraison des applications et les mises à jour.

Restez à jour avec une architecture hautement évolutive de services cloud distribués à l'échelle mondiale. Tirez parti de l'intelligence du *cloud* pour recueillir des enseignements et des lignes de base pour vos stratégies de sécurité et vos paramètres de configuration.

Microsoft Azure AD

Le service d'identité d'entreprise Azure Active Directory (Azure AD) fournit l'authentification unique et l'authentification multi-facteur pour vous aider à protéger vos utilisateurs contre 99,9 % des attaques de cybersécurité.

Des outils de développement facilitent l'intégration de l'identité dans vos applications et services.

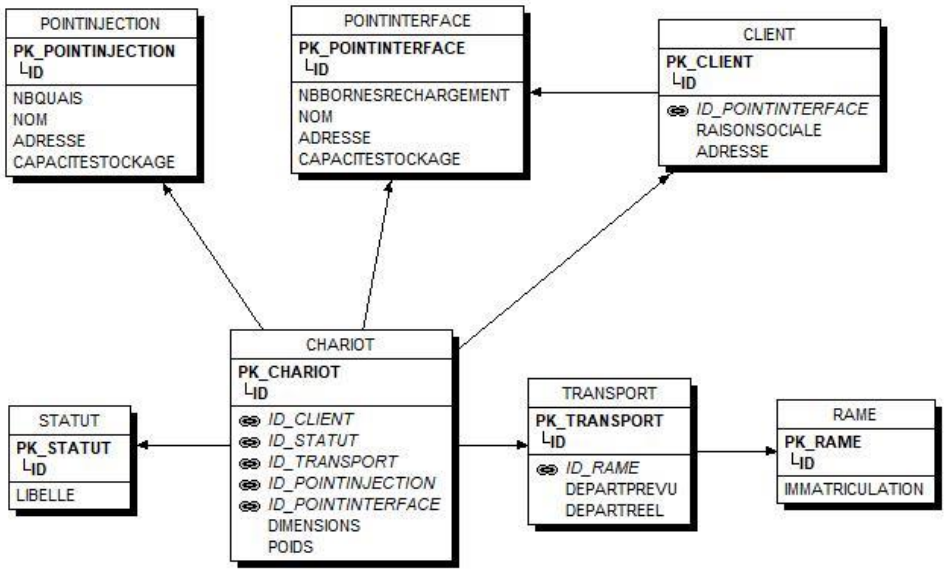
Que les utilisateurs soient sur site ou à distance, offrez-leur un accès transparent à toutes leurs applications afin qu'ils puissent rester productifs où qu'ils se trouvent.

Simplifiez l'authenti-fication unique. Azure AD prend en charge plus de 2 800 applications *Software as a Service* (*SaaS*) pré-intégrées.

Documentation du dossier 2

Document 2.1 : Schéma logique de données

Représentation graphique



Représentation textuelle

PointInjection(id, nbQuais, nom, adresse, capaciteStockage)
- id : clé primaire

PointInterface(id, nbBornesRechargement, nom, adresse, capaciteStockage)
- id : clé primaire

Client(id, raisonSociale, adresse, id_pointInterface)
- id : clé primaire
- id_pointInterface : clé étrangère en référence à PointInterface

Rame(id, immatriculation)
- id : clé primaire

Transport(id, id_rame, departPrevu, departReel)
- id : clé primaire
- id_rame : clé étrangère en référence à Rame

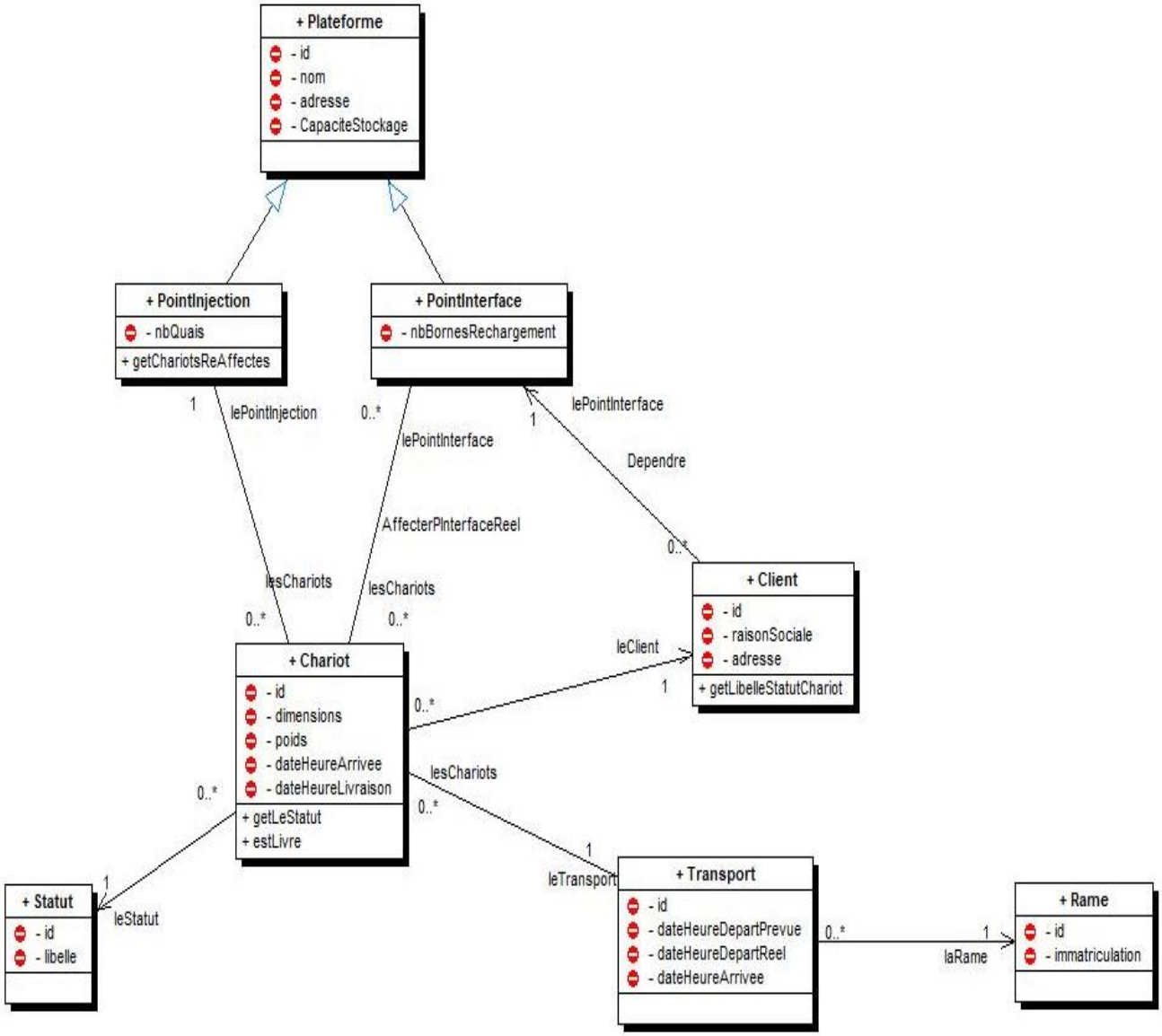
Statut(id, libelle)
- id : clé primaire

Chariot(id, id_client, id_statut, id_transport, id_pointInjection, id_pointInterface, dimensions, poids)
- id : clé primaire
- Id_client : clé étrangère en référence à Client
- Id_statut : clé étrangère en référence à Statut
- Id_transport : clé étrangère en référence à Transport
- id_pointInjection : clé étrangère en référence à PointInjection
- id_pointInterface : clé étrangère en référence à PointInterface

Document 2.2 : Diagramme des classes métier de l'application TramFret

Seules les propriétés et les méthodes utiles à l'application TramFret apparaissent dans le diagramme de classes et la description des classes.

Dans le diagramme des classes, les constructeurs, accesseurs et mutateurs n'ont pas été représentés.



Document 2.3 : Description des classes de l'application TramFret

Classes métier :

```
public class Client {
        private int id;
        private String raisonSociale;
        private String adresse;
        PointInterface lePointInterface; // point d'interface le plus près de son adresse
        // Constructeur
        public Client(int id, String raisonSociale, String adresse, PointInterface lePointInterface) { … }

        public String getLibelleStatutChariot(Chariot unChariot) {  //  à compléter }

        public int getId() { … }
        public String getRaisonSociale() { … }
        public void setRaisonSociale(String raisonSociale) { … }
        public PointInterface getLePointInterface() { …  }
        public void setLePointInterface(PointInterface lePointInterface) { … }
}
public class Statut {

        private int id;
        private String libelle;
        //Constructeur
        public Statut(int id, String libelle) { … }
        public int getId() { … }
        public String getLibelle() { … }
        public void setLibelle(String libelle) { … }
}

public class Rame {
        private int id;
        private String immatriculation;

        // Constructeur
        public Rame(int id, String immatriculation) { … }
        public int getId() { … }
        public String getImmatriculation() { … }
        public void setImmatriculation(String immatriculation) { … }
}
public class Chariot {
        private int id;
        private String dimensions;
        private double poids;
        private Date dateHeureArrivee;
        private Date dateHeureLivraison;
        private PointInjection lePointInjection;
        private PointInterface lePointInterface;  // réel, celui vers lequel le chariot sera acheminé
        private Statut leStatut;
        private Transport leTransport;
        private Client leClient;

        // Constructeur
        public Chariot(int id, String dimensions, double poids, Date dateHeureArrivee,
              PointInjection lePointInjection, Client leClient) {  //à compléter }
```

```
        public Boolean estLivre() { … }

        public int getId() { … }
        public String getDimensions() { … }
        public double getPoids() { … }
        public Date getDateHeureArrivee() { … }
        public Date getDateHeureLivraison() { … }
        public void setDateHeureLivraison(Date dateHeureLivraison) { … }
        public PointInjection getLePointInjection() { … }
        public PointInterface getLePointInterface() { … }
        public void setLePointInterface(PointInterface lePointInterface) { … }
        public Statut getLeStatut() { … }
        public void setLeStatut(Statut leStatut) { … }
        public Transport getLeTransport() { … }
        public void setLeTransport(Transport leTransport) { … }
        public Client getLeClient() { … }
        public void setLeClient(Client leClient) { … }
}

public  class Transport  {

        private int id;
        private Date dateHeureDepartPrevu;
        private Date dateHeureDepartReel;
        private Date dateHeureArrivee;
        private Rame laRame;
        private ArrayList<Chariot> lesChariots;

        //Constructeur
        public Transport(int id, Date dateHeureDepartPrevu, Rame laRame,
                        ArrayList<Chariot> lesChariots) { … }
        public int getId() { … }
        public Date getDateHeureDepartPrevu() { … }
        public void setDateHeureDepartPrevu(Date dateHeureDepartPrevu) { … }
        public Date getDateHeureDepartReel() { … }
        public void setDateHeureDepartreel(Date dateHeureDepartreel) { … }
        public Date getDateHeureArrivee() { … }
        public void setDateHeureArrivee(Date dateHeureArrivee) { … }
        public Rame getLaRame() { … }
        public void setLaRame(Rame laRame) { … }
        public ArrayList<Chariot> getLesChariots() { … }
}

public class Plateforme {
        private int id;
        private String nom;
        private String adresse;
        private int capaciteStockage;

        //Constructeur
        public Plateforme(int id, String nom, String adresse, int capaciteStockage) { … }
        public int getId() { … }
        public String getNom() { … }
        public void setNom(String nom) { … }
        public String getAdresse() { … }
        public void setAdresse(String adresse) { … }
        public int getCapaciteStockage() { … }
        public void setCapaciteStockage(int capaciteStockage) { … }
}
```

```
public class PointInterface extends Plateforme {
        private int nbBornesRechargement;
        private ArrayList<Chariot> lesChariots;

        // Constructeur
        public PointInterface(int id, String nom, String adresse, int capaciteStockage,
                int nbBornesRechargement) { …}
        public int getNbBornesRechargement() { … }
        public void setNbBornesRechargement(int nbBornesRechargement) { … }
}

public class PointInjection extends Plateforme {
        private int nbQuais;
        private ArrayList<Chariot> lesChariots;

        // Constructeur
        public PointInjection(int id, String nom, String adresse, int capaciteStockage,
                int nbQuais) { // à compléter }
        public ArrayList<Chariot> getChariotsReAffectes(Client unClient) {  // à compléter }
}
```

Autres classes de l'application :

```
public class SuiviController {
        public void enregistrerArriveePlateformeLogistique() { … }
        public int majChariotsIncident(ArrayList<Transport> lesTransports, Statut unStatut)
                { // à compléter }
        public void SuiviController_enregistrerArriveeTransport() { … }
        public static ArrayList<Chariot> SuiviController_getChariotsDate(PointInjection unPointInjection, Date
                uneDate) { … }
}


public class TramFretException extends Exception {
        public TramFretException() {… }

        // Constructeur
        public TramFretException(String message) { … }
        }
}
```

Classe technique :

Classe ArrayList<T>

Cette classe représente une liste fortement typée d'objets. T représente la classe des objets de la liste.

Cette classe fournit des méthodes de recherche, de tri et de manipulation de listes dont :

Public void add(<T> unObjet) : ajoute l'objet unObjet de la classe <T> à la fin de la liste

Public int size() : retourne le nombre d'objets de la classe <T> dans la liste.

Instanciation d'une liste :

```
List<T> uneListe = new ArrayList<T>()
```

Parcours d'une liste :

```
for (<T> unObjetDeT : uneListe) { … }
```

Document 2.4 : Synthèse de l'entretien avec le responsable de produit (*product owner*)

Certains chariots sont considérés comme dangereux s'ils contiennent au moins une matière dangereuse.

Les matières dangereuses sont classifiées selon le modèle en vigueur pour le transport routier. En voici la description :

Catégories des matières dangereuses

1	Matières et objets explosibles
2	Gaz
3	Liquides inflammables
4	Solides inflammables
5	Comburants ou peroxydes
6	Matières toxiques
7	Matières radioactives
8	Matières corrosives
9	Matières dangereuses diverses, provoquant une réaction violente spontanée

Le transport des matières dangereuses, avec la meilleure sécurité possible, nécessite des rames spécialisées. Ces rames spécialisées sont tracées avec des informations qui leur sont propres :

- La liste des catégories de matières dangereuses transportables,
- La date de l'agrément pour le transport de ces matières dangereuses,
- Le nom du cabinet qui a délivré l'agrément.

Afin de limiter les effets d'un accident non prévisible, la municipalité a demandé au GIE de limiter dans le temps le transport des matières dangereuses. Il a donc été défini, pour chaque matière dangereuse, un ou plusieurs créneaux horaires pendant lesquels elle peut être transportée.

Liste des créneaux horaires à prévoir :

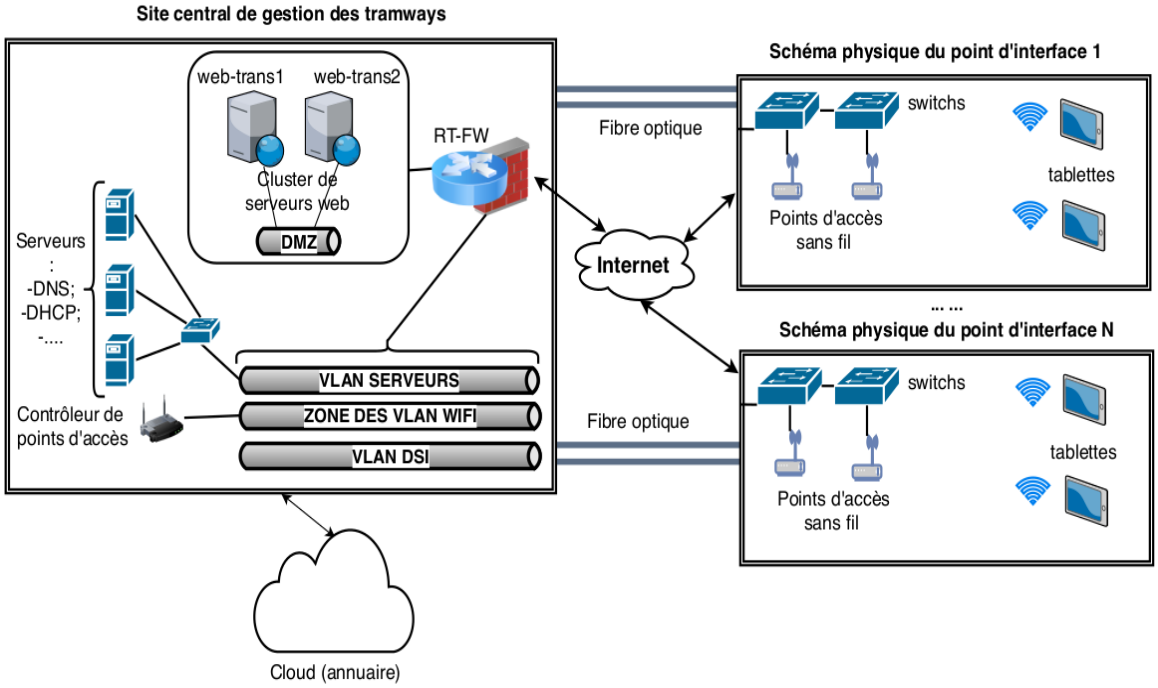
1	Journée	6h - 17h59
2	Soirée	18h00 - 23h59
3	Nuit	0h00 – 5h59

Lors de l'enregistrement d'un chariot le client devra préciser la quantité de chaque catégorie de matière dangereuse qu'il contient.

Documentation du dossier 3

Document 3.1 : Schéma simplifié du réseau et plan d'adressage

Schéma simplifié du réseau



Plan d'adressage actuel (extrait) :

RESEAU	ID VLAN	ADRESSE
DMZ		10.20.30.0/24
VLAN DSI	50	192.168.50.0/24
VLAN SERVEUR	60	192.168.60.0/24

Les VLAN Wi-Fi restant à créer sont décrits dans le document 3.4.

Document 3.2 : Commutateur hpe aruba 2530-8 (préconisé dans la baie de brassage d'un point d'interface)

Caractéristiques :

- Connectivité : 8 ports 10/100 + 2 x combo Gigabit Ethernet / SFP Gigabit.
- Débit (taille des paquets 64 octets) : 4,1 Mpps. Capacité de commutation : 5,6 Gbps.
- Protocole de gestion à distance : SNMP 1, RMON 1, RMON 2, RMON 3, RMON 9, Telnet, SNMP 3, SNMP 2c, HTTP, TFTP, SSH-2, CLI.
- Algorithme de chiffrement : SSL.
- Méthode d'authentification : Radius, TACACS+, Secure Shell v.2 (SSH2).
- Caractéristiques : Contrôle du flux, compatible DHCP, prise en charge de BOOTP, prise en charge d'ARP, liaisons, prise en charge du réseau local (LAN) virtuel, auto-uplink (MDI/MDI-X auto), mise en miroir des ports, prise en charge d'IPv6, mode semi-duplex, mode duplex intégral, prise en charge du protocole SNTP, sFlow, prise en charge du protocole Multiple Spanning Tree Protocol (MSTP), assistance Access Control List (ACL), qualité de service (QDS), STP Root Guard, prise en charge LLDP, Class of Service (CoS), Generic Attribute Registration Protocol (GARP).
- Normes : IEEE 802.3, IEEE 802.3u, IEEE 802.1D, IEEE 802.1Q, IEEE 802.3ab, IEEE 802.1p, IEEE 802.3x, IEEE 802.3ad (LACP), IEEE 802.1w, IEEE 802.1x, IEEE 802.1s, IEEE 802.1ab (LLDP), IEEE 802.3az
- Processeur : 1 x ARM: 800 MHz, RAM : 256 Mo DDR3 SDRAM, Mémoire flash : 128 Mo

Document 3.3 : Principes de fonctionnement du Wi-Fi centralisé

Les solutions sans-fil traditionnelles attribuent l'ensemble des fonctions de gestion du trafic, de contrôle radiofréquence, de sécurité et de mobilité au point d'accès. On parle de point d'accès autonome à configurer individuellement (*access point* ou *AP*).

Une alternative consiste en une architecture Wifi centralisée. Elle est constituée :

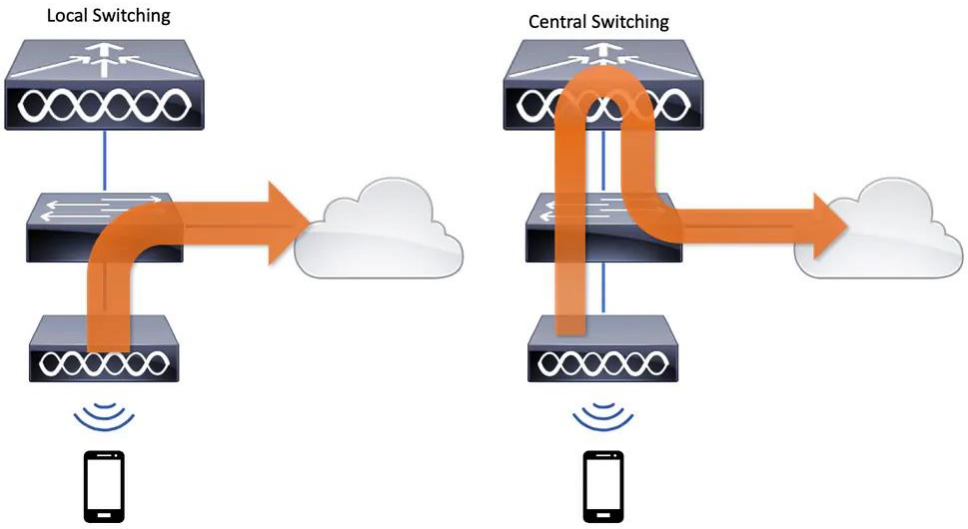
- D'une part d'un ensemble de points d'accès appelés bornes légères (*lightweight access point*) ;
- D'autre part d'un contrôleur chargé de gérer la configuration de ces bornes.

Le contrôleur contrôle à distance toutes les bornes d'accès ; toute la configuration s'y trouve.

Le fonctionnement simplifié de la borne est le suivant :

1. Elle démarre électriquement ;
2. Son interface filaire va envoyer une requête DHCP pour récupérer une adresse IP ;
3. Une fois reçue, elle va contacter son contrôleur ;
4. Le contrôleur va lui envoyer sa configuration minimale avec tous les paramètres nécessaires au bon fonctionnement (SSID – un ou plusieurs -, puissance, canal, …).

Le Catalyst 9800 a deux modes de fonctionnement :



Source : https://www.cisco.com/c/fr_ca/support/docs/wireless/catalyst-9800-series-wireless-controllers/213945-understand-flexconnect-on-9800-wireless.html

Dans le cas par défaut où la fonctionnalité FlexConnect n'est pas activée (image de droite), le trafic est centralisé vers le contrôleur 9800 (commutation centrale).

La borne légère (ou point d'accès) est chargée de faire le lien entre une communication sans-fil et le réseau filaire :

- d'un côté, elle reçoit/envoie des trames dans les airs pour les clients Wi-Fi ;
- de l'autre côté, elle est connectée à un commutateur et reçoit/envoie des trames depuis/vers un boîtier intelligent, le contrôleur (WLC – *Wireless LAN Controller)*.

Chaque borne est branchée sur un port associé au VLAN de management Wi-Fi pour pouvoir échanger avec le contrôleur.

Le contrôleur, quant à lui, est branché sur un port qui doit être en mesure d'envoyer des trames sur le ou les VLAN adéquats. C'est ce dernier qui va commuter/router les trames vers les bonnes destinations et non plus les bornes elles-mêmes.

Une fois configurée, la borne envoie tout le trafic des clients Wi-Fi au contrôleur qui se chargera de les envoyer vers les bonnes destinations.

Dans le cas où la fonctionnalité FlexConnect est activée (image de gauche), le trafic des clients sans fil est placé directement sur le réseau au niveau du point d'accès (commutation locale).

La borne légère (ou point d'accès) :

- d'un côté, elle reçoit/envoie des trames dans les airs pour les clients Wi-Fi ;
- de l'autre côté, elle est connectée à un commutateur et reçoit/envoie des trames directement sur les réseaux (via les VLAN associés aux SSID configurés).

Le contrôleur, quant à lui, est branché sur un port associé au VLAN de management Wi-Fi.

Document 3.4 : Les VLAN dédiés au Wi-Fi

Les VLAN suivants seront créés :

Le VLAN 100, dédié à l'administration du Wi-Fi :

N° VLAN	VLAN	Service	Adressage IP
100	WIFI-Mgmt_PI	VLAN dédié au management du Wi-Fi (contrôleur + bornes) des points d'interface.	10.22.100.0/24

Le VLAN 90, dédié au Wi-Fi du personnel :

N° VLAN	VLAN	Service	Adressage IP
90	WIFI_PI	VLAN dédié au Wi-Fi du personnel.	10.22.90.0/24

Le VLAN 91, dédié au Wi-Fi du personnel de maintenance :

N° VLAN	VLAN	Service	Adressage IP
91	WIFI_ MAINT	VLAN dédié au Wi-Fi.	10.22.91.0/24

Document 3.5 : Fonctionnement du logiciel Heartbeat

Heartbeat (qui signifie battement de cœur) est un logiciel installé sur le serveur maître et sur chaque serveur secondaire. L'**ensemble des serveurs** remplissant le même rôle de façon redondante se nomme un ***cluster*** (une grappe). **Chaque serveur** du *cluster* est un ***node*** (un nœud de la grappe).

Le serveur secondaire (**passif**) va surveiller en permanence les battements de cœur du serveur principal (**actif**) et prendre le relais automatiquement en cas d'arrêt des battements. Il deviendra alors actif.

La configuration de Heartbeat **doit être rigoureusement identique sur tous les serveurs de la grappe**. Elle repose sur trois fichiers :

- /etc/ha.d/ha.cf → Configuration de Heartbeat proprement dite ;
- /etc/ha.d/authkeys → Clé partagée entre les serveurs de la grappe ;
- /etc/ha.d/haresources → Liste des ressources (adresses et/ou services) fournies par la grappe.

Exemples de fichiers de configuration de la solution Heartbeat (documentation) :

Fichier /etc/ha.d/ha.cf

```
#Configuration du cluster
#Interface émettant et recevant les battements de cœur
bcast eth0
#Temps, en secondes, avant que le serveur soit déclaré mort
deadtime 5
#Intervalle, en secondes, entre deux battements de cœur
keepalive 1
#liste des nœuds (serveurs) formant le cluster
node  srv-web1   srv-web2
```

Fichier authkeys

```
#numéro de méthode d'authentification (1 = 1ère méthode)
auth 1
#type de hachage et mot de passe à utiliser pour la méthode 1
1 sha2 motdepasse
```

Fichier /etc/ha.d/haresources

```
#Ressources fournies par le serveur actif
srv-web1 Ipaddr::192.168.200.5   apache2
#Signification des paramètres de ce fichier :
#srv-web1 → indique le serveur actif par défaut (nom DNS du maître du #cluster)
;
#IPaddr::192.168.200.5 → Adresse flottante (virtuelle) attribuée au serveur
actif ;
#apache2 → Service à lancer sur le serveur lorsqu'il devient actif #(apache2
pour le service web Apache).
```

Document 3.6 : Extrait de la table de filtrage existante du routeur pare-feu RT-FW

Filtrage en entrée sur l'interface connectée au VLAN DSI (sous-interface) :

Règle	IP source	Port source	IP destination	Port destination	Protocole	Décision
1	VLAN DSI	*	Internet	80/443	TCP	Accepté
2	VLAN DSI	*	VLAN serveur	*	UDP/TCP/ICMP	Accepté
3	*	*	*	*	*	Refusé

Le pare-feu est de type *Stafeful Packet Inspection* (ou « à état »), il garde en mémoire l'état de connexions réseau.

Document 3.7 : Exemple d'ajout de règle dans Snort

```
alert tcp $EXTERNAL_NET any → $HOME_NET 21(msg:«nouvelle connexion FTP»; flags:S; SID:0001;)
```

Explication : la règle générera une alerte pour toute connexion FTP (21) réussie (flags : S pour synchronisée) depuis un poste extérieur ($EXTERNAL_NET) sur le réseau de la DMZ $HOME_NET.

La variable EXTERNAL_NET est définie à "any".