BUT1 RT – 2024-2025

SAE203 :

Mettre en place une solution informatique pour entreprise.

RAPPORT CYBERSECURITE

Louis DESCHAMPS
Pawel ZAJAC

(TDA-G2)

Table des matières

Introduction

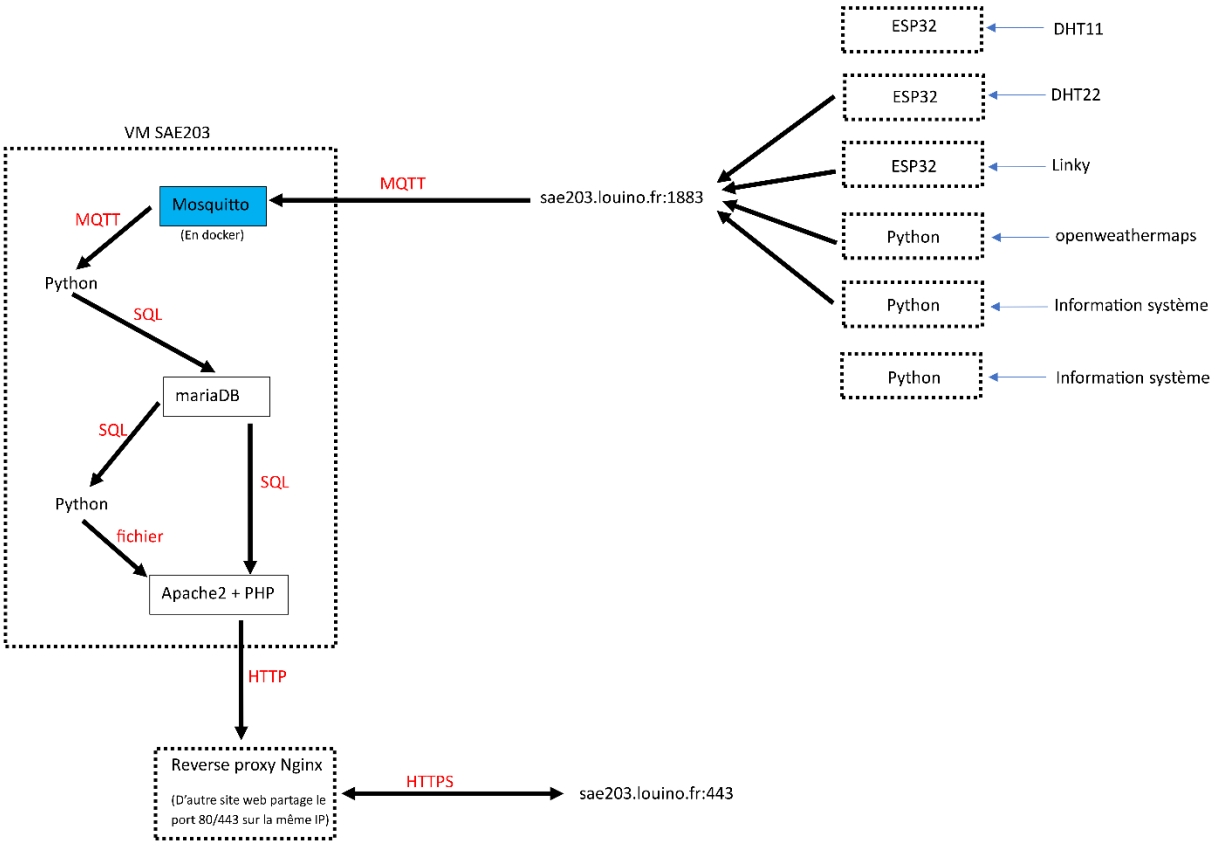
Cette SAE 203, nous a permis concevoir et déployer une plateforme de gestion des informations IoT reposant sur le protocole MQTT et divers programmes (Python, PHP, bases de données) pour collecter, stocker, traiter et présenter des données issues d'objets connectés.

Également, ce projet intègre une dimension de cybersécurité, visant à garantir la confidentialité, l'intégrité et la disponibilité des flux de données.

Cette partie du rapport a pour objectif de :

- Identifier les vulnérabilités potentielles de l'architecture (réseau, systèmes, information).
- Analyser l'exploitation de ces failles pour en mesurer l'impact sur la plateforme IoT.
- Proposer et mettre en œuvre des mesures de sécurité adaptées

Diagramme du système d'information mis en place



Analyse

Chaque élément, et chaque liaison peuvent être compromis. Nous allons faire une analyse en suivant la chaine d'information.

Pour les capteurs physiques, la liaison entre le capteur (DHT11/22 pour la température et l'humidité) et le microcontrôleur (ESP32) n'est pas sécurisé. Si on personne a un accédé physique au capteur, le bus de donnée (oneWire,i2c,uart,spi…) peut être compromis. Des fausses informations peuvent être envoyée.

- Pour cela, il n'y a pas de solution pour protégé cela. On peut la limiter au maximum en posant les capteurs dans les endroits difficile d'accès (en hauteur par exemple) et/ou dans les endroits interdit aux publics.

Pour les capteur logicielle (scripts python) les informations récoltées sur internet depuis différant serveur API peuvent être compromis. Des fausses informations peuvent être envoyée.

- On pourrait vérifier l'authenticité du serveur avec le qu'elle on communique. Notamment utilisée le SSL/TLS, pour chiffrer et authentifier (certificat) le serveur pour réduire les risques.

Le broker MQTT mis en place utilise de l'authentification pour y accéder. On ne peut pas lire ou écrire des valeurs dans les identifiants. Cependant, la connexion au broker MQTT n'est pas chiffer. Si la

connexion est interceptée, on peut extraire les identifiant et donc, on pourrait interagir avec le serveur, et envoyer de fausse information.

- On pourrait chiffer la communication avec le serveur. En utilisant le SSL/TLS, pour chiffrer et authentifier (certificat) le serveur pour éviter l'exfiltration des identifiant.

Le broker MQTT utilise le même identifiant pour tous les capteurs. Si un capteur est compromis, toutes les données sont compromises et peuvent être manipulée.

- Utilisée des identifiant séparer et mettre en place des ACL pour éviter qu'un utilisateur interagis avec les données d'un autre capteur.

Les capteurs (notamment physique) sont connectés sur un (ou des) réseau wifi. Diverses attaques sont possibles. Premièrement sur la couche physique, en brouillant le wifi, empêchant la communication des capteurs. Simple à mettre en place et efficace, cependant elle ne permet pas de récupérer les identifiant de connexion (Aujourd'hui les reseau wifi sont chiffrée avec une clé, selon la norme WPA2 ou WPA3). On ne pas éviter ce risque, sauf en utilisant un réseau câblé, ce qui est inadaptée dans la plupart des cas de système IoT.

Ensuite, en supposant que l'attaquant a accès au réseau LAN des capteur, a cause d'un réseau non sécurisé (exemple, en utilisant des sécurités obsolètes comme le WEP), un accès physique ou autre. Cette attaquant pourrais exploitée des attaques sur les couches réseau supérieure. Par exemple, un attaque MaM (man-in-middle) en utilisant un empoisonnement de la table ARP (arp spoofing) pour rediriger le trafic d'un capteur a l'ordinateur du l'attaquant plutôt qu'a la passerelle réseau. Dans le but d'intercepter le trafic (et de bloquée si l'envie le prend). Une fois le trafic intercepté, les identifiant peuvent être extrait et des fausses informations peuvent être envoyé aux broker MQTT.

- Séparé les capteur IoT du reste des équipement réseau (VLAN, ACL). Utilisé des normes wifi récent (comme le WPA3).
- Chiffrer la communication avec le serveur. En utilisant le SSL/TLS, pour chiffrer et authentifier (certificat) le serveur pour éviter l'exfiltration des identifiant, malgré une attaque MaM.

Sur le serveur :

MariaDB utilise de l'authentification avec un utilisateur (restreint a localhost), il ne peut pas se connectée depuis ailleurs sur réseau.

La machine virtuelle n'a pas de service d'accès à distance, SSH est désactivé.

De plus, des règles de filtrage (*Proxmox VE Firewall*) sont appliqué, toutes les connexion entrantes sont bloquées (DROP ALL en input), exceptant le port entrant 1883/TCP pour le broker MQTT (sur docker) et 80/TCP pour apache2.

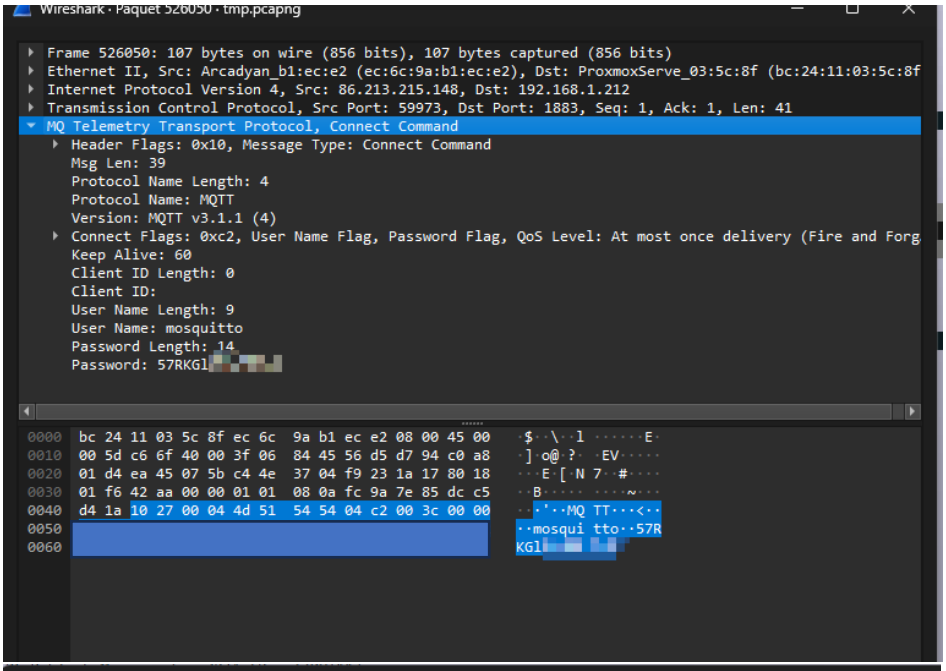
Le transit internet vers le reverse proxy (NGINX) n'est pas sécurisé. Il faudrait un avoir un accès sur le réseau interne entre les deux serveurs. Si c'est le cas, cela permettrais de de manipulé les informations affichées sur la page web.

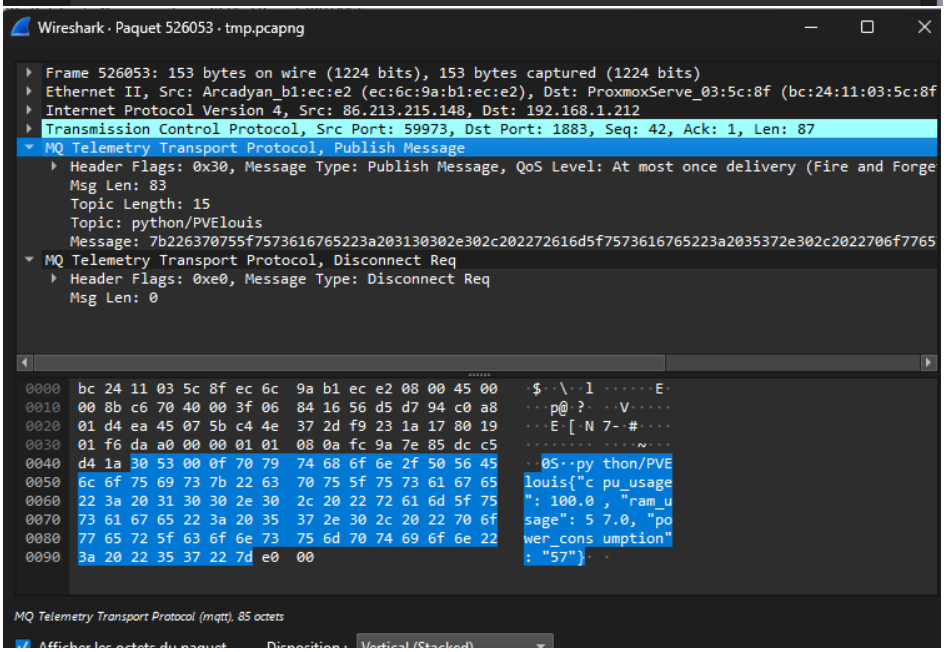
Le reverse proxy applique du chiffrement HTTPS et utilisant un certificat (Let's encrypt). De plus, il force la migration des connexion http vers https. Cela limite le risque d'interception du trafic et sa manipulation entre le serveur et le client (navigateur internet).

Expérimentation

Supposons que nous récupérions le trafic réseau entre le capteur et le serveur. Plusieurs méthodes sont possibles pour cela (écoute Wi-Fi, attaque ARP, faux point d'accès Wi-Fi, TAP Ethernet, etc.).

On peut inspecter le trafic MQTT non chiffré :

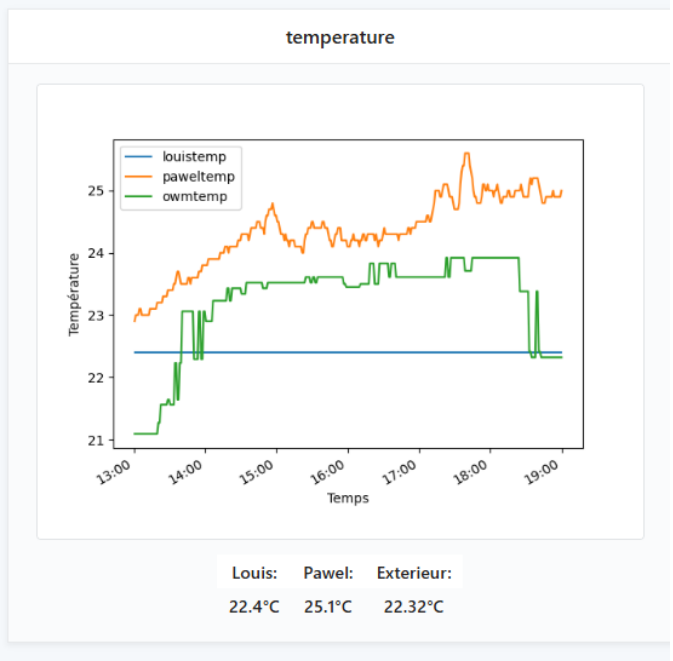




Il est alors facile d'extraire les identifiants, le topic et la structure des messages json. Il suffit donc de publier soi-même un message avec les mêmes identifiants et le même topic pour envoyer de fausses informations.

En cas de fausses informations relevées par le capteur :

- Si ce sont des valeurs (float) avec des données aberrantes (par exemple une température à 1000 °C ou une humidité à 150 %), le serveur les stocke et les affiche sur la page sans distinction.
- Si les valeurs sont autre chose que des float (injection de chaînes de caractères), le serveur rejette le message et conserve la valeur précédente (ou NULL s'il n'y avait pas de valeur précédente).



Si le capteur ne renvoie pas de message, le serveur conserve la valeur précédente (ou NULL s'il n'y avait pas de valeur précédente).

Dans cet exemple, le capteur de température « louistemp » était débranché, il conserve donc la même valeur.

Conclusion

Nous avons identifié les principales vulnérabilités de notre plateforme IoT :

Sécurisation des capteurs :

- Les capteurs physiques doivent été installés dans des zones restreintes pour limiter l'accès physique non autorisé.
- Les échanges entre scripts Python et API externes, ainsi que la connexion au broker MQTT, doivent être chiffrés et authentifiés via SSL/TLS, empêchant l'interception des identifiants/donnée et la falsification des messages.

Renforcement de l'authentification et des contrôles d'accès :

- Chaque capteur doit disposer de ses propres identifiants MQTT, associés à des listes de contrôle d'accès (ACL) fines, afin de limiter le capteur a la publication sur son topic et pas ailleurs.

Segmentation et filtrage du réseau

- Les objets IoT sont doivent sur un réseau dédié. (wifi séparé, utilisation de vlan…)
- Utilisée des réseaux sans fils sécurisé (exemple : WIFI en wpa2 ou wpa3, Zigbee, LoraWAN). Et évité les protocole obsolète ou non chiffrée (certain des appareille en 433MHz ou 868MHz avec le qu'elle on peut bricoler très facilement avec un Flipper Zero)

Robustesse de l'infrastructure et continuité de service :

- Utilisation de HTTPS avec un certificat pour la page web.
- Mise à jour régulière.
- Filtrage réseau (Parfeux)
- Sauvegardes régulière (notamment de la base de données)

Ces mesures permettraient de garantir la confidentialité, l'intégrité et la disponibilité de notre solution IoT. En s'appuyant sur des bonnes pratiques en informatique : chiffrement systématique, authentification forte, segmentation réseau.