Aerus+

Introduction

Dans l'aviation loisir les intepéries sont souvent la causes d'inquéitudes pour les pilotes. Aerus+ est un système designé pour les propritétaires d'aerodrôme et d'aeroport afin d'installer sur chaque vois d'accès un capteur permettant des remontée d'information en temps réel sur les conditions météorologiques telle que la temperatures, la vitesse du vent, l'humidité et la pression atmosphérique. ces données sont transmise en wifi ou en lora au serveur cloud AWS. Les données sont ensuite traitées et affichées sur une interface web. Les utilisateurs peuvent ainsi consulter les données en temps réel et être informé des conditions météorologiques en temps réel.

Cahier des charges

Le cahier des charges est decomposé en 3 parties: Wifi, Lora et interface web.

Wifi

Un premier device doit être capable de se connecter à un réseau wifi et de transmettre les données en temps réel au serveur cloud AWS. Les données doivent être lue sur différents capteurs. A des fins de secutité, l'appareil doit ce connecter au cloud en utilisant des certificats de sécurité. Un premier certificat doit être installé sur l'appareil à l'usine. Ce certificat doit être utilisé pour se connecter et recuperer un second certificat qui sera utilisé pour les communications futures. Cela permet de garantir que les personnes travaillant à l'usine ne peuvent pas accéder aux données des clients.

Lora

Web

Architecture

Conception

Wifi

Pour commencer, nous avons utilisé un ESP32-S3 pour les communications wifi. Pour les capteurs, nous avons utilié un BME680 pour la température, l'humidité et la pression atmosphérique. Pour la vitesse du vent, nous avons utilisé un anémomètre. Ensuite, à l'aide de diffentes librairies, nous avons pu lire les données des capteurs et se connecter wifi.

Demande de certificat

Premièrement, il est nécessaire de crée un premier objet sur AWS. Pour ce faire, il faut aller dans: IOT Cores -> all devices -> Things -> Create Things. Cette appareil est un appareil temporaire qui permet de se connecter à AWS pour récupérer un certificat. La création de cet appareil peut être automatisé en utilisant l'API d'AWS mais cela n'a pas été fait dans le cadre de ce projet. Cet apprareil possede un certificat avec une police bien spécifique.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:eu-west-
1:137068254498:topic/$aws/certificates/create/*",
        "arn:aws:iot:eu-west-1:137068254498:topic/$aws/provisioning-
templates/ma_iot_template/provision/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": [
        "arn:aws:iot:eu-west-
1:137068254498:topicfilter/$aws/certificates/create/*",
        "arn:aws:iot:eu-west-1:137068254498:topicfilter/$aws/provisioning-
templates/ma_iot_template/provision/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:RegisterThing",
      "Resource": "arn:aws:iot:eu-west-
1:137068254498:topicfilter:thing/${iot:Connection.Thing.ThingName}"
    }
  ]
}
```

Cete politique permet à l'appareil de se connecter à AWS, de publier sur le topic `$aws/certificates/create/json`qui permet de récupérer un certificat et de s'abonner au topic`$aws/certificates/create/json/accepted` qui permet de récupérer un certificat. Enfin, l'appareil peut s'enregistrer en tant qu'objet sur AWS.

Il est important de noter que le topic `$aws/certificates/create/json` est un topic spécifique qui permet de récupérer un certificat. Il n'est pas possible de s'abonner à ce topic. Le certificat transmis sur le topic `$aws/certificates/create/json/accepted` est reçu uniquement par l'appareil qui a demandé le certificat. C'est à dire que les messages n'est pas visible sur les autres appareils abbonnés à ce topic même le client de test d'AWS.

Au niveau de l'esp32, il faut que cette authentification ce fasse uniquement au premier démarrage. Pour ce faire, nous avons utilisé la mémoire flash de l'esp32 pour stocker le certificat. Lors du premier démarrage, l'esp32 se connecte à AWS et récupère un certificat. Ce certificat est ensuite stocké dans la mémoire flash. Un champ `already_registered` est également stocké dans la mémoire flash. Lors des démarrages suivants, l'esp32 utilise ce certificat pour se connecter à AWS.