Lerat - Mackay - Convert - Benazzouz - Herbras

Security project : Monitoring our City - Vehicloud

I] Presentation of the project and security threats

Our Iot solution named Vehicloud is at the stage of a proof of concept. As such, this solution does not include many of the industry standard cyber-security measures put into place to stop most attacks. Our project, is a work in progress, bare bones project for a complete IoT solution, that creates data thanks to sensors, data which is sent through BLE to an internet connected gateway who communicates data to an API, which in turn stores this data in an online database before using a third party planning software to display this data.
Hence, our system has many weak points that malicious third parties could attack through a multitude of attacks, an issue inherent to IoT with security flaws that may appear at many levels. We will list them here and describe the solutions that need to be put into place to address these issues. In the best case scenario, we would implement measure to address : Authentication • Authorization • Confidentiality • Integrity • Non-repudiation

At one end of our chain, we have the data creation step. Here, attackers could create erroneous data to inject into our data processing chain, making the results wrong. This is only possible if the attackers can get access to our data pipeline. As of right now, this is entirely possible, since our system only relies on the HTTP post method to send a JSON data packet to our API. it is good practice to never trust client side data and inputs. We can prevent some erroneous/malicious data entry by applying more security to our API. We need to filter data in terms of values (eliminating data points that are outside of the city our solution is implemented in and eliminating packets that have impossible values, such as INSA being 1092°C). This filtering is not a proper security measure though, we need to implement authentication. If the API can verify through a secret code the authenticity of the sender, this already makes our system more secure. Establishing the keys though is an issue. Here the main issues are data integrity and non-repudiation, sender's authentication and authorization.

Another point of attack is the physical data transfer from our bike-borne device, to our gateway through BLE. Here, multiple issues are to be considered. Indeed, Man In The Middle attacks, sniffing and spoofing are all possible at this crucial link in our data processing chain. One solution would be data encryption, though this solution has it's faults as well. Here the main issues are data integrity and confidentiality, and the possible compromising of our authentication information at the input of the chain. An attacker can gain information about our system, and act like a legitimate device to send in erroneous data.

A crucial aspect of our project is privacy. Should attackers gain access to our PostgreSQL database, hosted online on AWS, they could associate GPS location information, data timestamps and bicycle IDs to identify a specific trip. Once trips can be mapped out, the attackers gain knowledge on our user's daily habits. To address this issue, we need to find a way to anonymise data even better, by dissociating location tags with bike IDs and by

making sure the database is secure. To properly hide the privacy of every user, we need to find a way to make sure that it is not easy to deduce a trip from datapoints. We cannot encrypt all data that is stored, since encryption affects the size of the data we store in a considerable way. Our API and our data is stored on a free online PaaS, which has its own security concerns. We can imagine many attacks to these services which would negatively affect our ability to provide our service (DDoS of the Heroku platform, flooding our API, flooding our database that can only process 10.000 data messages and has a fixed memory allocation…etc).

Therefore, attacks can be at the physical layer as well as the software level, with threats ranging from service errors and downtime, reaching all the way to invasion of privacy. As of right now, our solution does not implement nearly enough security measures to stop these attacks, seeing as we are trying to prove the concept. For further development, it is imperative that these security flaws be addressed.

II] Presentation of an ideal protocol for send messages

In this part we will focus on the protocol we want to use in order to send the sensitive data (GPS) from the sensor to the gateway.
The idea is to associate a BikeID to a MAC address and then authorise the send through an exchange of BLE frames. First the gateway scans only for the MAC addresses he has in memory, after that the sensor gives his bike ID and then the gateway will send him a beacon to tell him to send his data. We can encrypt all that information in the channel c with the sharedkey k.
The following proverif code try to implement this protocol
set verboseClauses = short.

```
free c : channel.

fun enc(bitstring, bitstring): bitstring.
equation forall m: bitstring, k:bitstring;
   dec(enc(m,k),k) = m.
event sensor(bitstring).
event gateway(bitstring).

free k:bitstring [private].
free header1: bitstring.

let processConnect =
   new mac_adress: bitstring;
   event gateway(mac_adress);
   new bike_id: bitstring ;
   out(c, enc((header1, bike_id), k));
   in(c, x:bitstring);
   let (=header2, =bike_id, rb: bitstring) = x in
   out(c,enc((header3, rb, mac_adress), k)).

let processData
```

```
    in(c, x1: bitstring) ;
    let (=header1, bike_id: bitstring) = dec(x1, k) in
    new rb: bitstring ;
    out(c, enc((header2, bike_id, rb), k)) ;
    in(c, x2: bitstring) ;
    let (=header3, =rb, mac_adress: bitstring) = dec(x2,k) in
    event sensor(data).

query secret gps
query secret data
query attacker(k).
query m :bitstring ; inj-event(gateway(m)) ==> inj-event(sensor(m)).

process
    processConnect | processData
```