R0: va bene

R1: riconfigurazione per minimizzare overhead e allungare il ciclo di vita delle sentinelle

R2: non da fare -> Adattamento su soglia: riconfigurare dopo aver superare la soglia

C’è molto in letteratura

R3 Va bene: per capire se c’è un guasto basta applicare tattiche: ping echo (ping alla macchina e se non si riceve echo la macchina è silente) e geart bit (path di interazione)

Capire perché la sentinella non risponde.

Sui gateway si fa pingh echo o heart bit.

Guasto anche su gateway

* Funzionalità di autodiagnosi di guasti o nodi silenti

Nodo: sensore su sentinella, sentinella o gateway.

Sensore non funzionante se dato out of range e non arriva il dato -> controllo da fare in edge sul gateway

Oss: chiedersi sempre chi esegue la funzionalità: componente (edge, sentinelle, cloud) o insiemi di componenti (tipo ping echo).

Nodo gateway e upserver installa serie di layer del chipstark

* Server
* Strato mqtt per comunicare con server
* Percorso uplink da sentinelle a cliud (supportato)
* Percorso downlink da server cloud a gateway a sentinelle (supportato)

Veicolare i messaggi da realizzare (usando code topiche del chipstark) o c’è già in una API di chipstark

Esposizione API di flussi in MQTT, ma si può accedere alle code topiche con REST o gRPC: come si possono sfruttare per veicolare le funzionalità.

Bozza iterazione 0: casi d’uso e topologia

Aggiustare documento

Modello concettuale per dire gli attori con i loro attributi e come interagiscono: pre-processing.

Sfruttare specifiche smart\_E e guardare il vocabolario dei termini di W3C nell’ambito del monitoraggio ambientale. Passo .1 per prendere confidenza con il dominio.

https://www.w3.org/TR/wot-architecture/

Tipo figura 1 DING.net.

Per ogni funzionalità da realizzare bisogna realizzarla con il contributo di una o più componenti che girano un po’ su server e su gateway e su sentinella.

Le funzionalità core

Emulazione sentinella con software che manda dati al gateway.

Figura 2: rappresentazione software delle caratteristiche da monitorare.

Basta mettere un application server (MQTT server astratto perché interessano i messaggi di send, publish,… Si può scrivere MessageServer).

Interessa la dependability.

Altre entità possono apparire dal consorzio W3C (gateway sostituito da edge node in termini di nome).

Nella knowledge si devono memorizzare gli stati delle sentinelle.

Quando si definirà l’architettura e si capisce come è fatto il sistea edge, per ogni goal delle sentinelle si può introdurre un loop di controllo mape-loop k e si introdurranno i vari componenti.

Magari non stanno tutti su gateway, ma può consistere in una configurazione master-slave (spiegherà prossima volta).

Requisiti: funzionalità monitoraggio con tecniche di dependability come EB e PE (più ricerca di altre).

Risalire ultima versione del documento W3C (11) e tracciare le specifiche (elenco puntato).

* Funzionalità o su gateway o su server usando la documentazione di W3C
* Inglese per grafici e codice (sentinella = watchdog

Iterazione -1

* Leggere documento 11 w3c e selezionare le parti d’interesse in un elenco
* Figura 1 DingNet e adattarlo al nostro sistema.

Tralasciare MQTT nei nomi e restare più astratti (non considerare un messaggio come quello MQTTMessage sotto MQTTServer)

Iterazione 0

* Sistemare diagramma hardware
* Figura 2 DingNet da adattare

Iterazione da fare ogni volta:

* Guardare API chirpstack