Piotr Król
Rafał Korszuń

HOW TO BUILD IOT SOLUTIONS USING CLOUD INFRASTRUCTURE ?

Agenda

- About Us
- Embedded Systems in IoT
- Cloud Infrastructure
- Demo

About Us

About
Piotr Król

Embedded Systems Consultant
freedom and liberty enthusiast
Open Source and Linux fan
chess player
productivity maniac and blogger

Twitter: @pietrushnic
Email: piotr.krol@3mdeb.com
Web: http://3mdeb.com
http://lmgtfy.com/?q=3mdeb

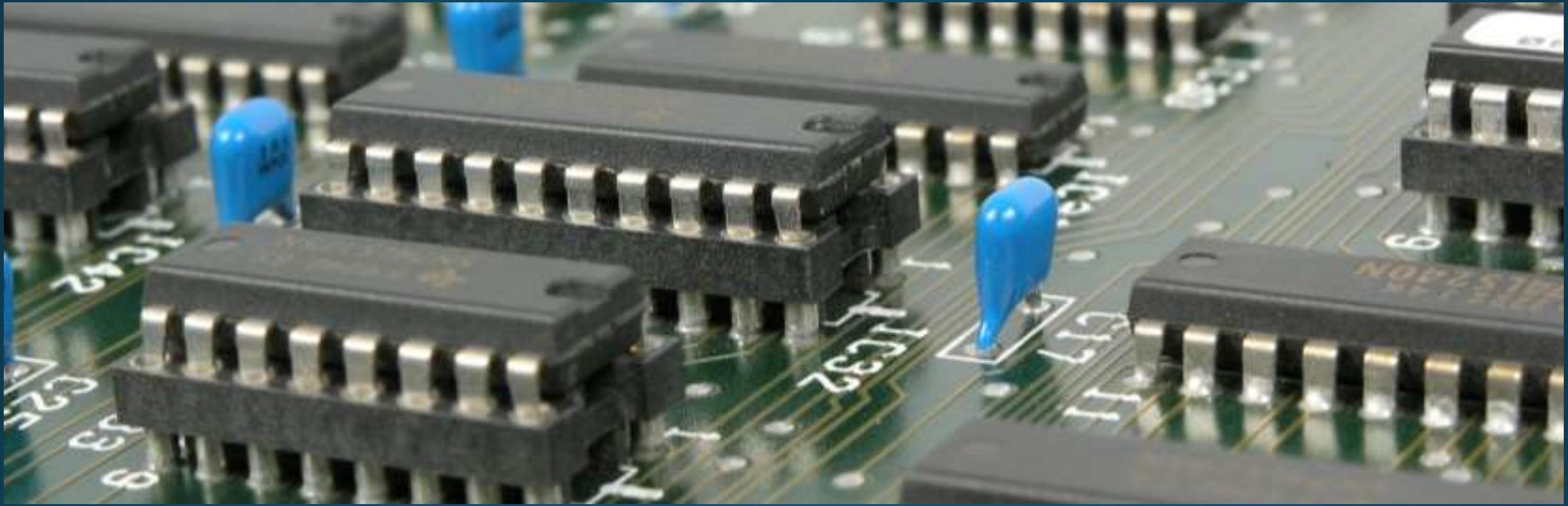
About
Rafał Korszuń

Software Architect / CEO at Kleder
Machine learning enthusiast
Cloud solutions evangelist
Excellent design maniac

Twitter: @rafalkorszun
Email: rafal.korszun@kleder.co
Web: http://kleder.co

Definitions

- Internet of Things (IoT)
  - Computing **concept** that describes a future where everyday physical objects will be connected to the Internet and be able to identify themselves to other devices.
- Embedded System
  - Is a special-purpose **computer system** designed to perform one or a few dedicated functions, often with real-time computing constraints.
- Cloud computing
  - Cloud computing is a general term for the delivery of **hosted services** over the Internet



Embedded Systems in IoT

Endpoint architecture - design

Key Factors

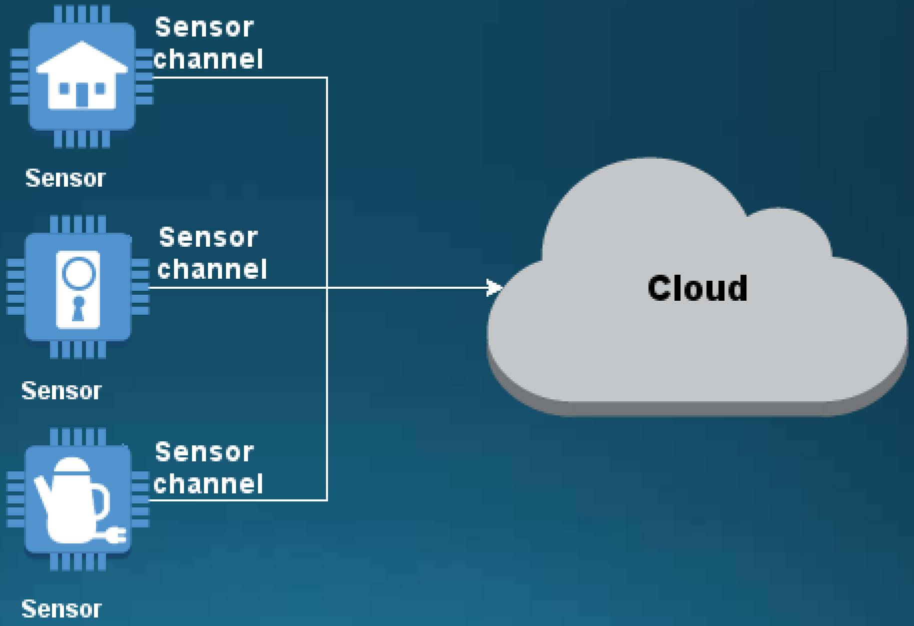
- The customer is always right
  - Cost
  - Use cases/requirements
  - Bill of materials
  - Time to market
  - Power source
- Experience
- Openness
- Regulation

Key Problems

- Security
- Firmware upgrade
- Connectivity
- Power consumption
- Provisioning

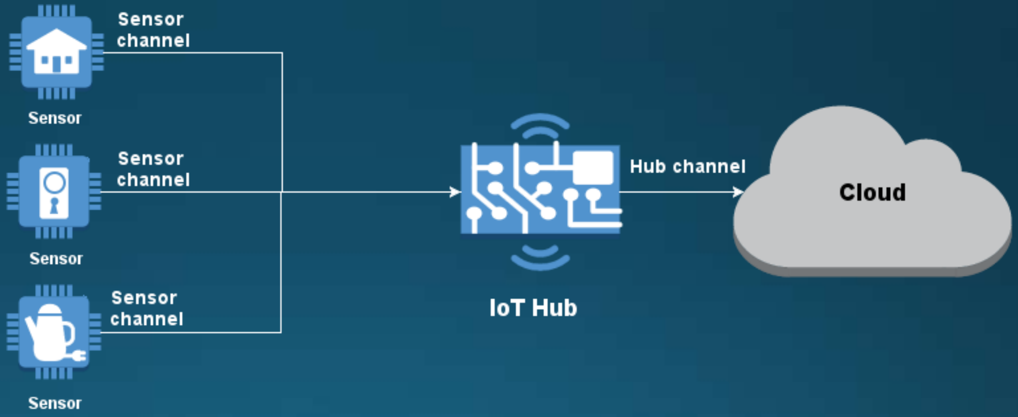
Endpoint architecture

- Expensive sensors
- Ready to use product
- Custom hardware
- Built-in:
  - Connectivity
  - Storage
  - MCU performance
- Complex firmware
  - NAT
  - ISP restrictions
  - OTA update



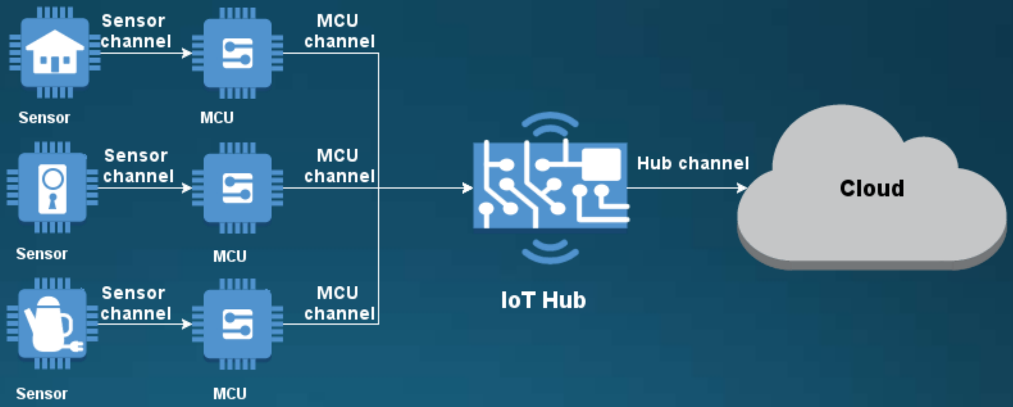
Endpoint architecture

- Existing infrastructure
- Plug and play
- Fixed sensor capability



Endpoint architecture

- Most popular
- Most flexible
- Dedicated hardware
- Lot of software



IoT - components

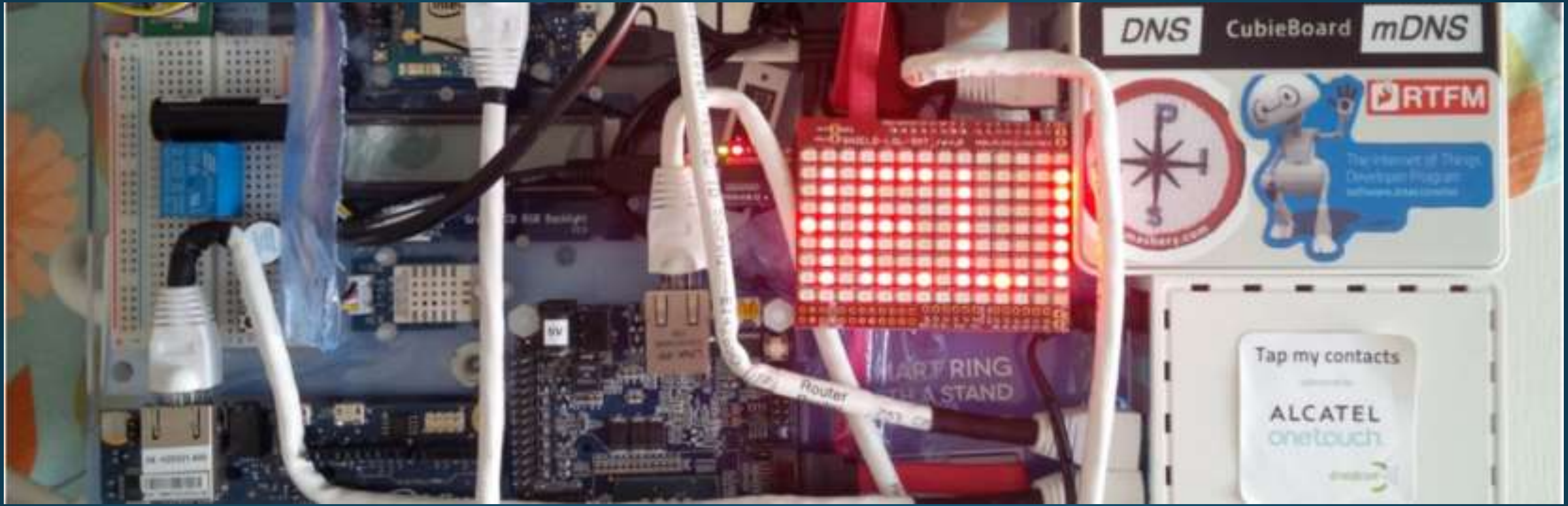
•Sensors

Sensor channel

MCU

IoT Hubs/Transceivers

IoT Hub channel

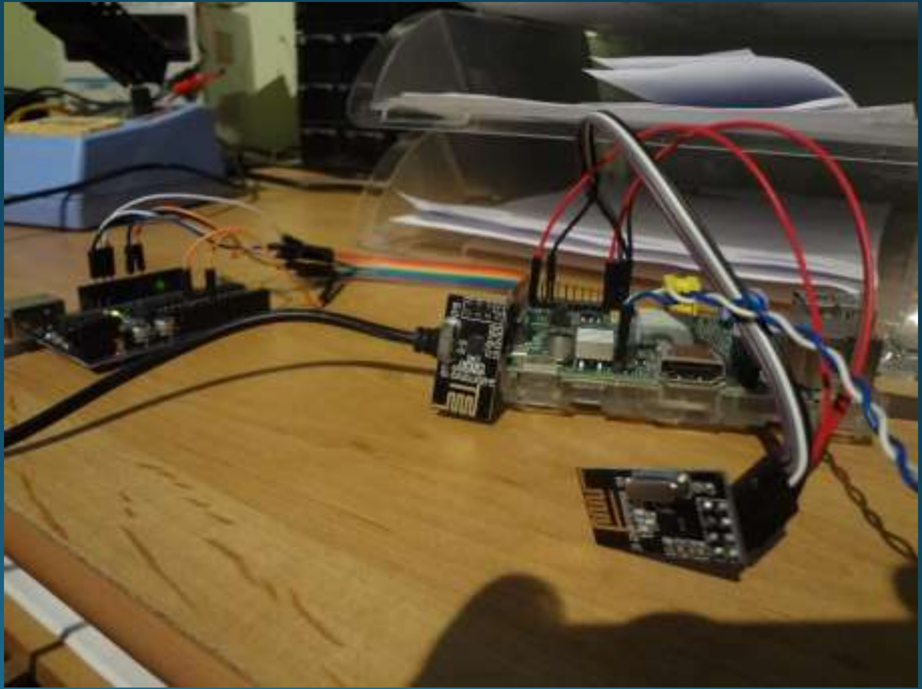


How IoT really look like ?

Credits to Michael Schloh von Bennewitz: https://fosdem.org/2016/schedule/event/deviot19/

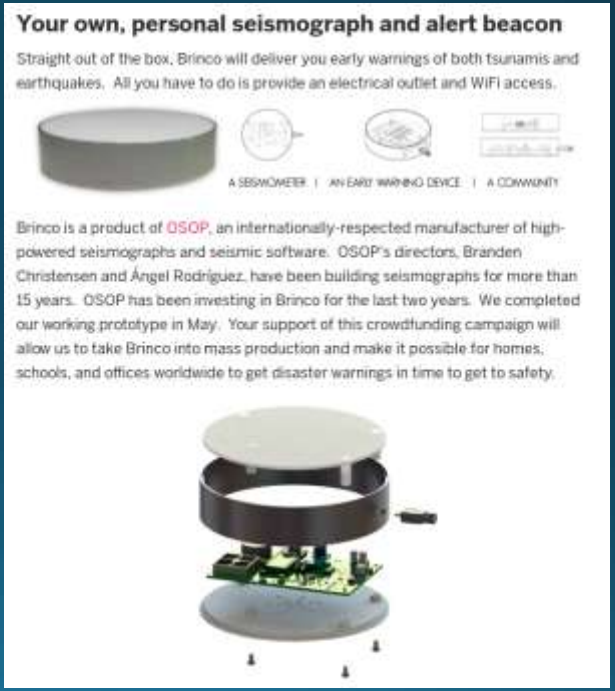
Hardware and applications

- NRF24L01-based beacons
- Arduino and Raspberry Pi as controllers
- Location detection



Hardware and applications

- TI CC3200
- Tsunami and earthquake early detection system



Your own, personal seismograph and alert beacon

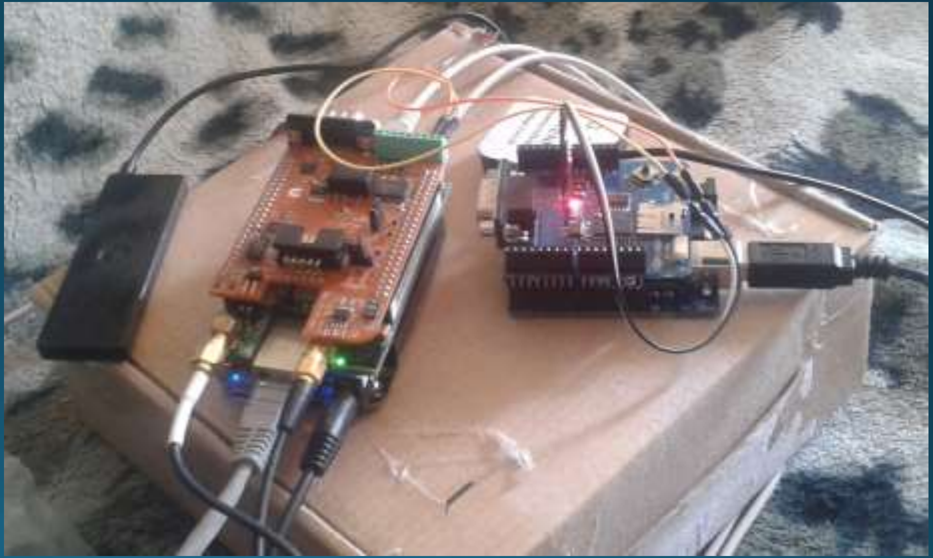
Straight out of the box, Brinco will deliver you early warnings of both tsunamis and earthquakes. All you have to do is provide an electrical outlet and WiFi access.

A SEISMOMETER | AN EARLY WARNING DEVICE | A COMMUNITY

Brinco is a product of OSOP, an internationally-respected manufacturer of high-powered seismographs and seismic software. OSOP's directors, Branden Christensen and Ángel Rodríguez, have been building seismographs for more than 15 years. OSOP has been investing in Brinco for the last two years. We completed our working prototype in May. Your support of this crowdfunding campaign will allow us to take Brinco into mass production and make it possible for homes, schools, and offices worldwide to get disaster warnings in time to get to safety.

Hardware and applications

- BeagleBone Black (TI AM3358)
- GPRS/GPS cape
- CAN shield
- Arduino as car emulator
- Logistics system (car location and diagnostics)



Hardware and applications

- ESP8266
- Retail selling/notification system

Cloud Infrastructure

Cloud-Oriented Architecture (COA)

- Scalability

Security

Readiness

Interoperability

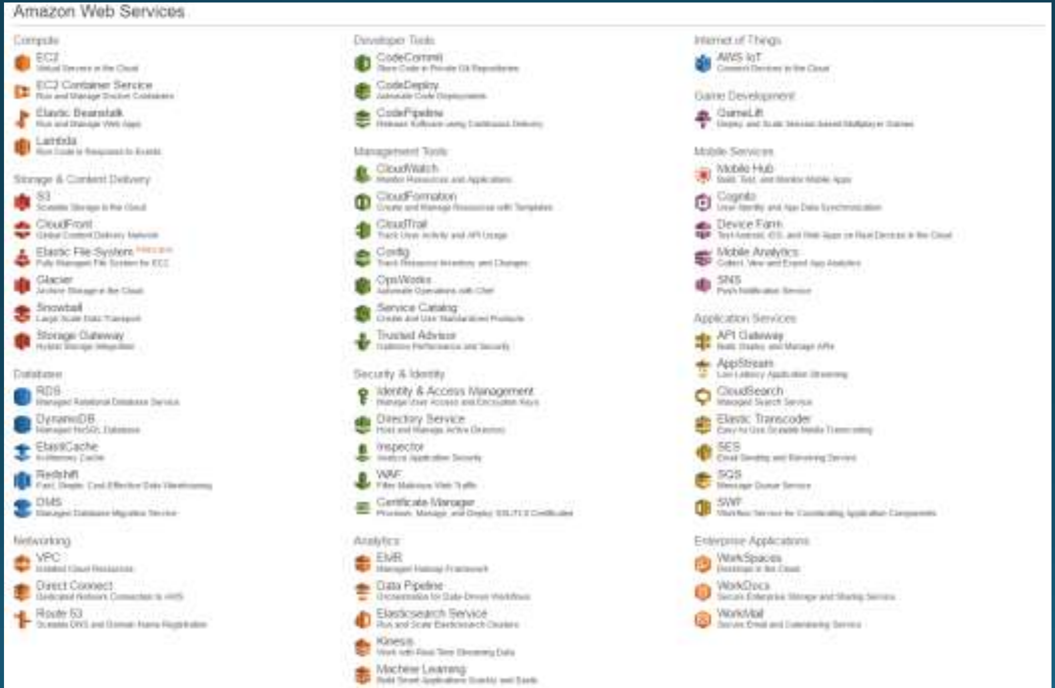
Microservices principles

- Domain Driven Design
- Hide Implementation Details
- Decentralization
- Failure Isolation
- Continuous Delivery through DevOps Culture

Cloud Top 10 Risks (OWASP)

- Accountability & Data Risk
- User Identity Federation
- Regulatory Compliance
- Business Continuity & Resiliency
- User Privacy & Secondary Usage of Data
- Service & Data Integration
- Multi-tenancy & Physical Security
- Incidence Analysis & Forensics
- Infrastructure Security
- Non-production Environment Exposure

AWS Microservices Example



Amazon Web Services

Compute

- EC2
  Virtual Servers in the Cloud
- EC2 Container Service
  Run and Manage Docker Containers
- Elastic Beanstalk
  Run and Manage Web Apps
- Lambda
  Run Code in Response to Events

Storage & Content Delivery

- S3
  Scalable Storage in the Cloud
- CloudFront
  Global Content Delivery Network
- Elastic File System Preview
  Fully Managed File System for EC2
- Glacier
  Archive Storage in the Cloud
- Snowball
  Large Scale Data Transport
- Storage Gateway
  Hybrid Storage Integration

Database

- RDS
  Managed Relational Database Service
- DynamoDB
  Managed NoSQL Database
- ElastiCache
  In-Memory Cache
- Redshift
  Fast, Simple, Cost-Effective Data Warehousing
- DMS
  Managed Database Migration Service

Networking

- VPC
  Isolated Cloud Resources
- Direct Connect
  Dedicated Network Connection to AWS
- Route 53
  Scalable DNS and Domain Name Registration

Developer Tools

- CodeCommit
  Store Code in Private Git Repositories
- CodeDeploy
  Automate Code Deployments
- CodePipeline
  Release Software using Continuous Delivery

Management Tools

- CloudWatch
  Monitor Resources and Applications
- CloudFormation
  Create and Manage Resources with Templates
- CloudTrail
  Track User Activity and API Usage
- Config
  Track Resource Inventory and Changes
- OpsWorks
  Automate Operations with Chef
- Service Catalog
  Create and Use Standardized Products
- Trusted Advisor
  Optimize Performance and Security

Security & Identity

- Identity & Access Management
  Manage User Access and Encryption Keys
- Directory Service
  Host and Manage Active Directory
- Inspector
  Analyze Application Security
- WAF
  Filter Malicious Web Traffic
- Certificate Manager
  Provision, Manage, and Deploy SSL/TLS Certificates

Analytics

- EMR
  Managed Hadoop Framework
- Data Pipeline
  Orchestration for Data-Driven Workflows
- Elasticsearch Service
  Run and Scale Elasticsearch Clusters
- Kinesis
  Work with Real-Time Streaming Data
- Machine Learning
  Build Smart Applications Quickly and Easily

Internet of Things

- AWS IoT
  Connect Devices to the Cloud

Game Development

- GameLift
  Deploy and Scale Session-based Multiplayer Games

Mobile Services

- Mobile Hub
  Build, Test, and Monitor Mobile Apps
- Cognito
  User Identity and App Data Synchronization
- Device Farm
  Test Android, iOS, and Web Apps on Real Devices in the Cloud
- Mobile Analytics
  Collect, View and Export App Analytics
- SNS
  Push Notification Service

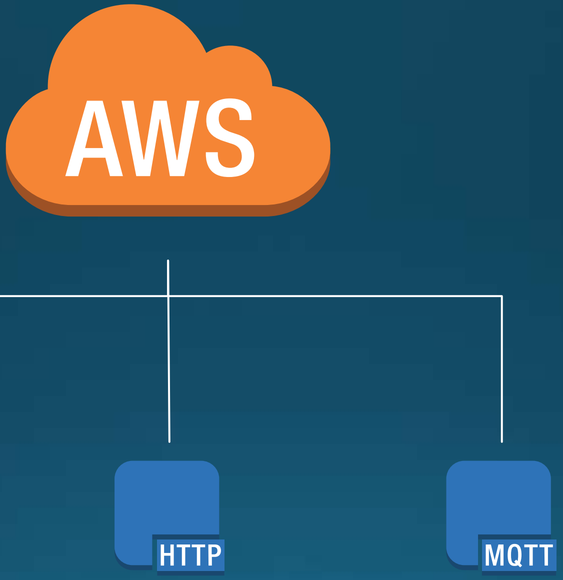
Application Services

- API Gateway
  Build, Deploy and Manage APIs
- AppStream
  Low Latency Application Streaming
- CloudSearch
  Managed Search Service
- Elastic Transcoder
  Easy-to-Use Scalable Media Transcoding
- SES
  Email Sending and Receiving Service
- SQS
  Message Queue Service
- SWF
  Workflow Service for Coordinating Application Components

Enterprise Applications

- WorkSpaces
  Desktops in the Cloud
- WorkDocs
  Secure Enterprise Storage and Sharing Service
- WorkMail
  Secure Email and Calendaring Service

AWS communication



IoT HTTP/2 protocol

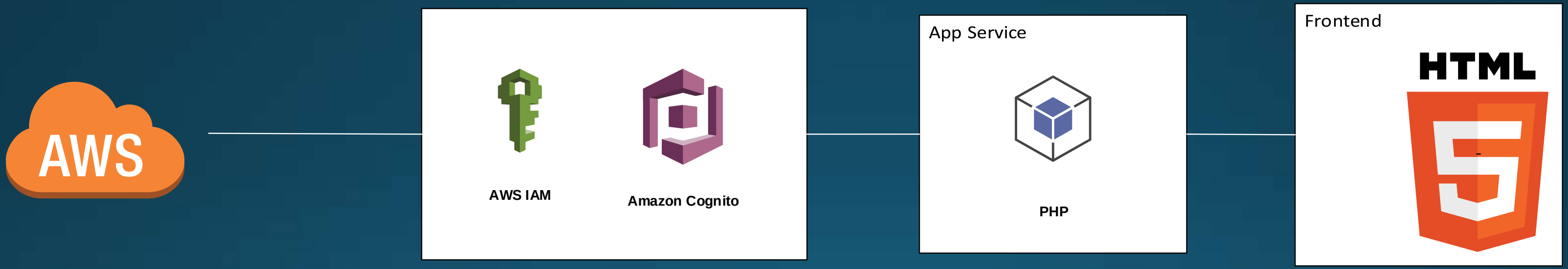
IoT HTTP protocol

IoT MQTT protocol

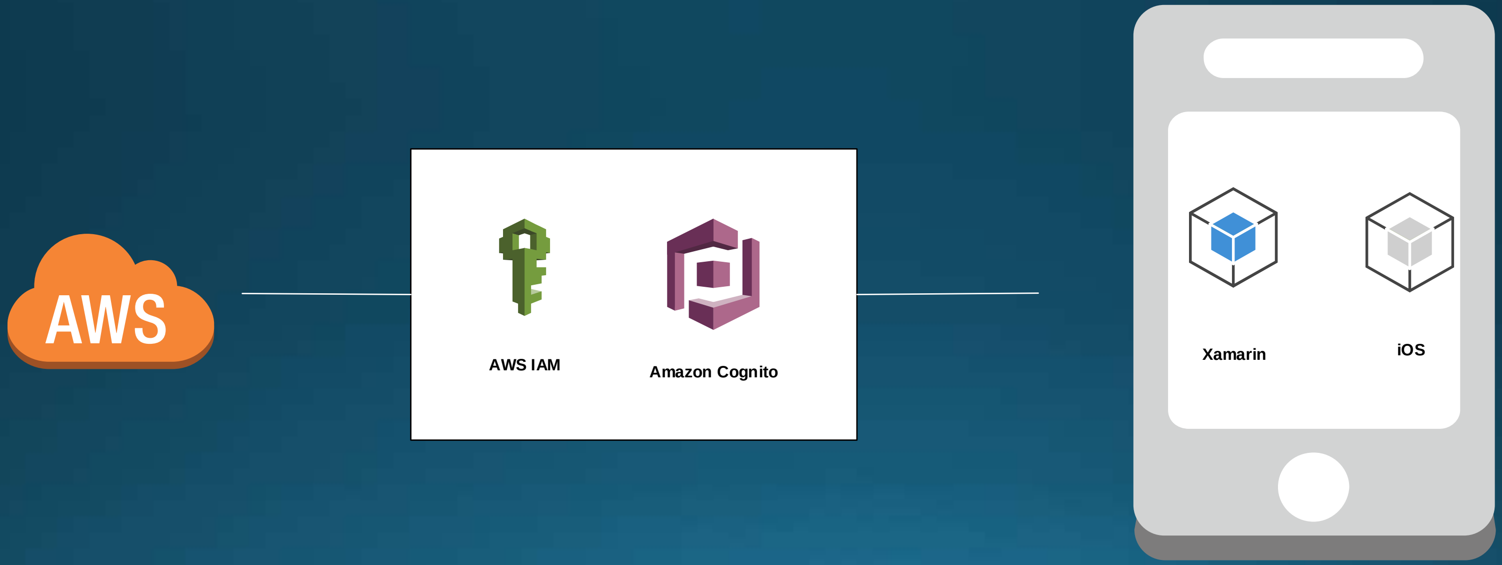
AWS SDK’s

- Android
- Browser (JavaScript)
- iOS
- Java
- .NET
- Node.js (JavaScript)
- PHP
- Python
- Ruby
- Go
- C++

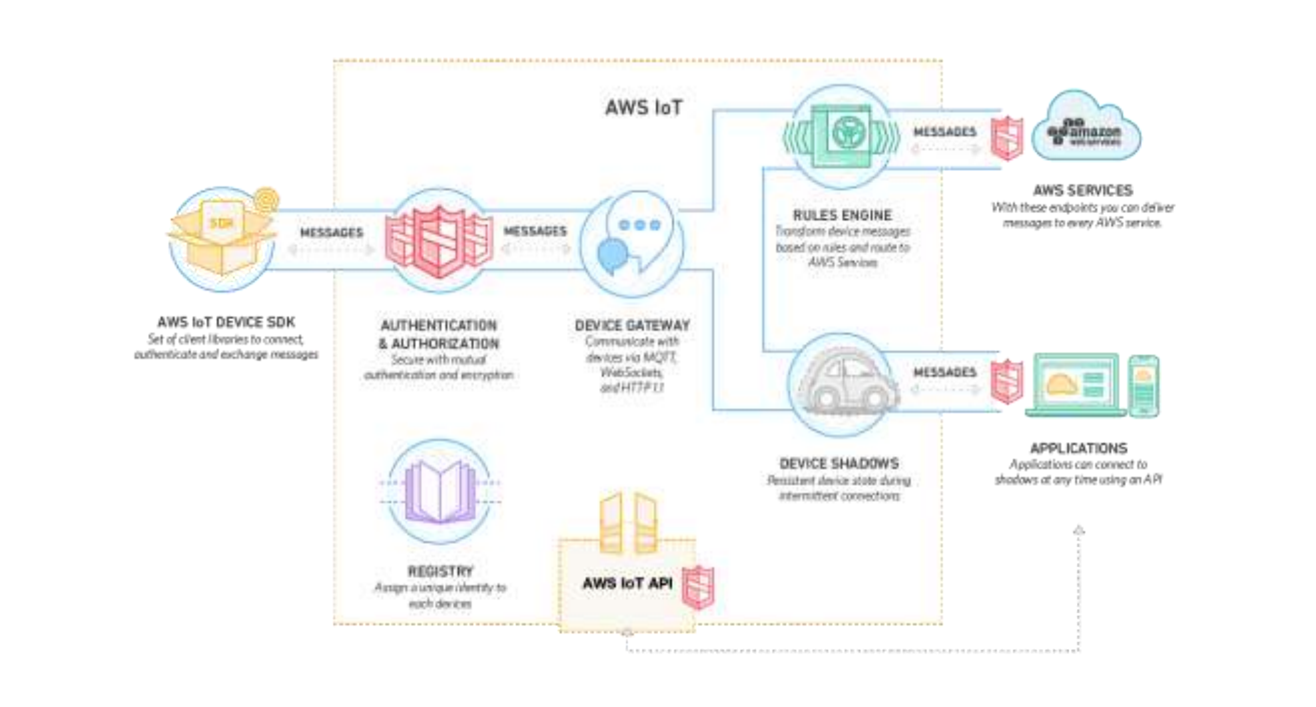
Cloud & Web interfaces



Cloud & Mobile Clients



Amazon IoT Model


AWS IoT
AWS IoT DEVICE SDK
Set of client libraries to connect, authenticate and exchange messages
MESSAGES
AUTHENTICATION & AUTHORIZATION
Secure with mutual authentication and encryption
MESSAGES
DEVICE GATEWAY
Communicate with devices via MQTT, WebSockets, and HTTP 1.1
RULES ENGINE
Transform device messages based on rules and route to AWS Services
MESSAGES
amazon web services
AWS SERVICES
With these endpoints you can deliver messages to every AWS service.
DEVICE SHADOWS
Persistent device state during intermittent connections
MESSAGES
APPLICATIONS
Applications can connect to shadows at any time using an API
REGISTRY
Assign a unique identity to each devices
AWS IoT API

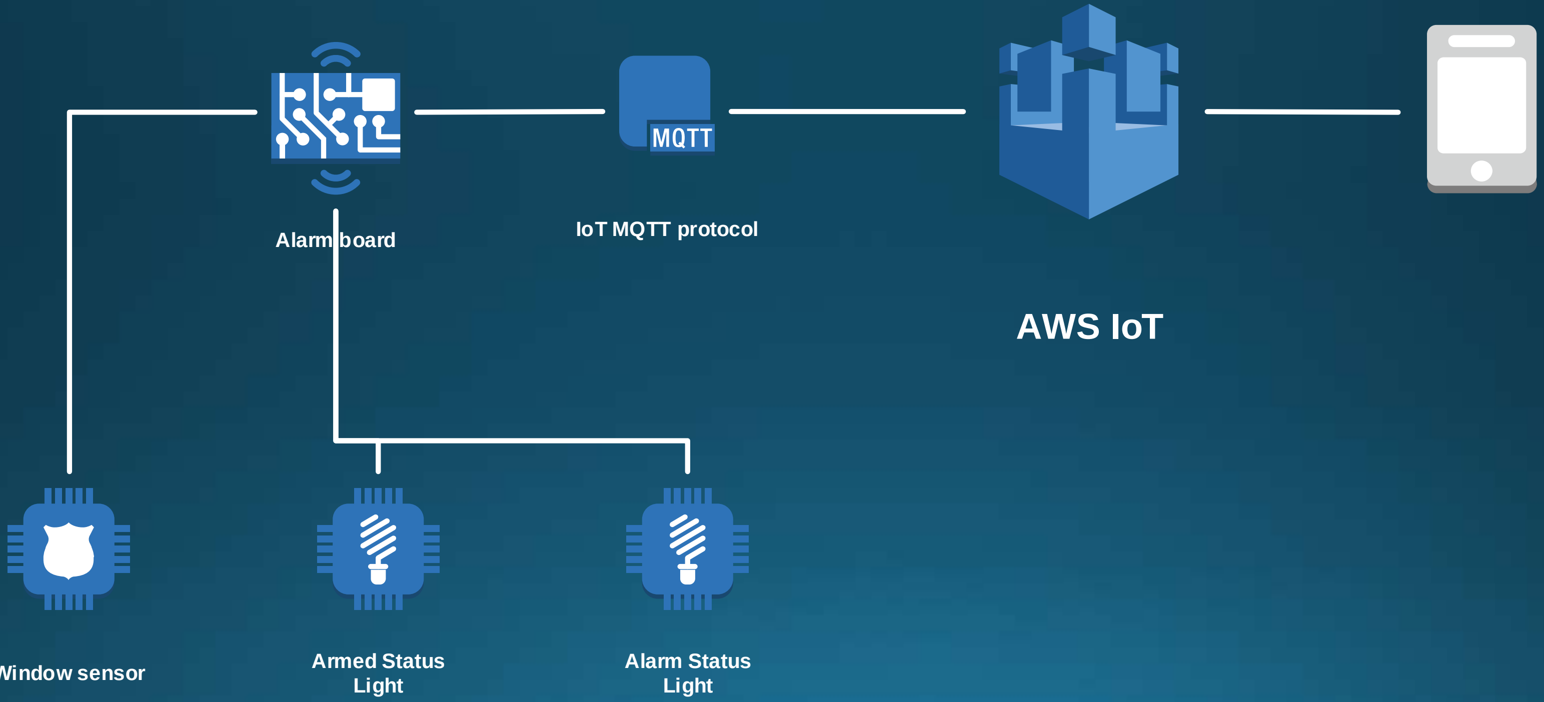
Credits https://aws.amazon.com/iot/how-it-works/

Demo

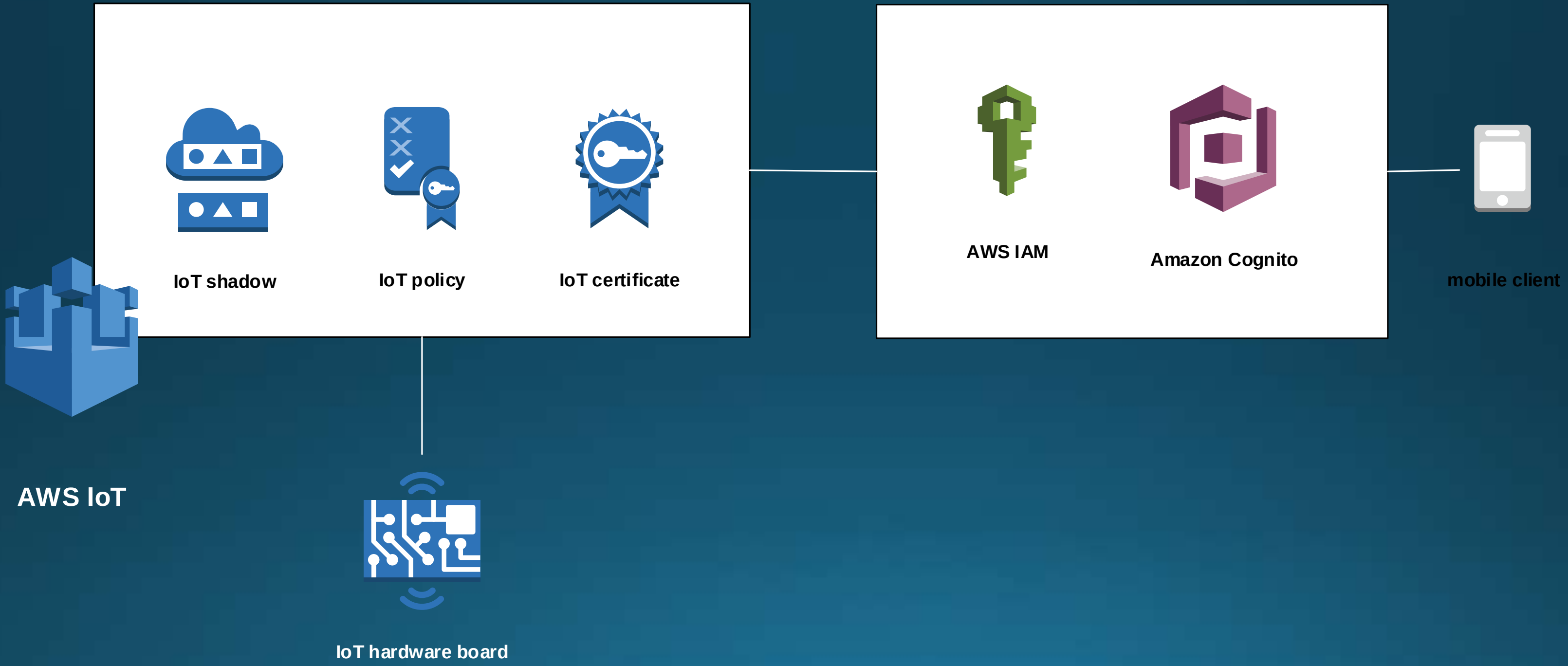
Use case

- We have alarm installation in home
- There is only one sensor for window
- Alarm can be armed
- We want to use mobile device to control alarm system

Use case – High Level Architecture



Use case – Cloud infrastructure



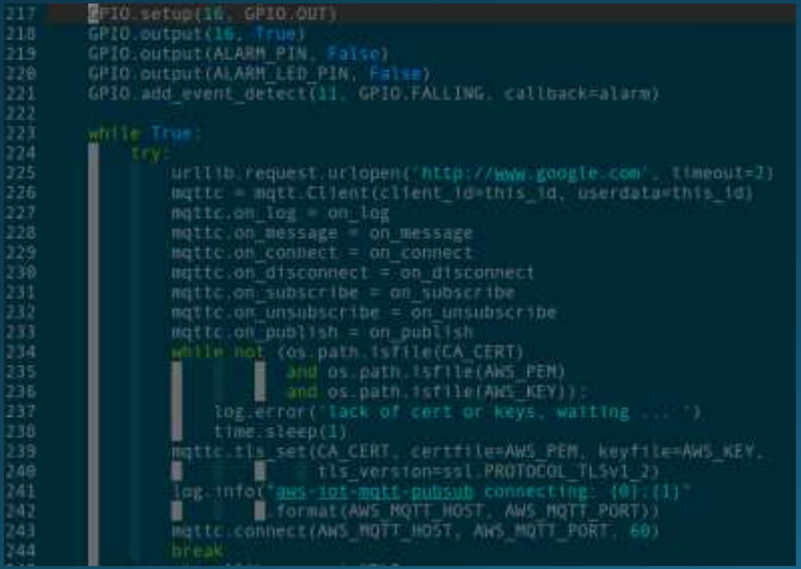
Alarm Exchange Data

- window
  - 1 – close
  - 0 - open
- lock
  - 0 – disarmed
  - 1 – armed
- alarm
  - 0 – OFF
  - 1 – ON
- alarm_reset
  - Turn OFF alarm button indicator (0 – normal, 1 – pushed, 2- processed)

```
while (again) {
    iN = -1;
    again = false;
    getline(cin, sInput);
    system("cls");
    stringstream(sInput) >> dblTemp;
    iLength = sInput.length();
    if (iLength < 4) {
        again = true;
        continue;
    } else if (sInput[iLength - 3] != '.') {
        again = true;
```

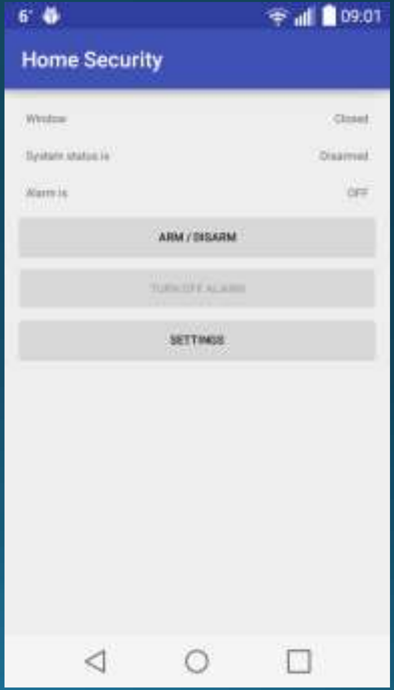
Live Demo

Solution Explanation



```
    GPIO.setup(16, GPIO.OUT)
    GPIO.output(16, True)
    GPIO.output(ALARM_PIN, False)
    GPIO.output(ALARM_LED_PIN, False)
    GPIO.add_event_detect(11, GPIO.FALLING, callback=alarm)

    while True:
        try:
            urllib.request.urlopen('http://www.google.com', timeout=2)
            mqttc = mqtt.Client(client_id=this_id, userdata=this_id)
            mqttc.on_log = on_log
            mqttc.on_message = on_message
            mqttc.on_connect = on_connect
            mqttc.on_disconnect = on_disconnect
            mqttc.on_subscribe = on_subscribe
            mqttc.on_unsubscribe = on_unsubscribe
            mqttc.on_publish = on_publish
            while not (os.path.isfile(CA_CERT)
                       and os.path.isfile(AWS_PEM)
                       and os.path.isfile(AWS_KEY)):
                log.error('lack of cert or keys, waiting ... ')
                time.sleep(1)
            mqttc.tls_set(CA_CERT, certfile=AWS_PEM, keyfile=AWS_KEY,
                          tls_version=ssl.PROTOCOL_TLSv1_2)
            log.info("aws-iot-mqtt-pubsub connecting: {0}:{1}"
                      .format(AWS_MQTT_HOST, AWS_MQTT_PORT))
            mqttc.connect(AWS_MQTT_HOST, AWS_MQTT_PORT, 60)
            break
```

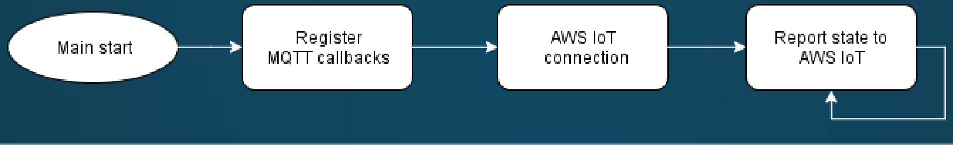


Window Closed

System status is Disarmed

Alarm is OFF

Demo IoT code flow



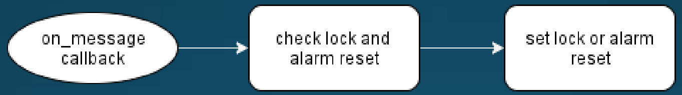
Main start

Register MQTT callbacks

AWS IoT connection

Report state to AWS IoT

MQTT on_message callback

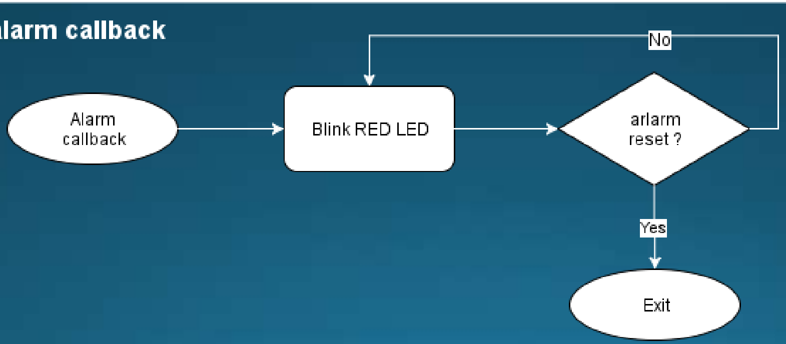


on_message callback

check lock and alarm reset

set lock or alarm reset

GPIO alarm callback



Alarm callback

Blink RED LED

arlarm reset ?

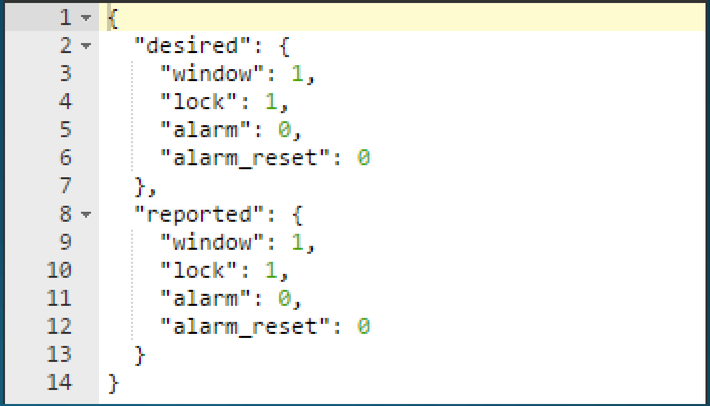
No

Yes

Exit

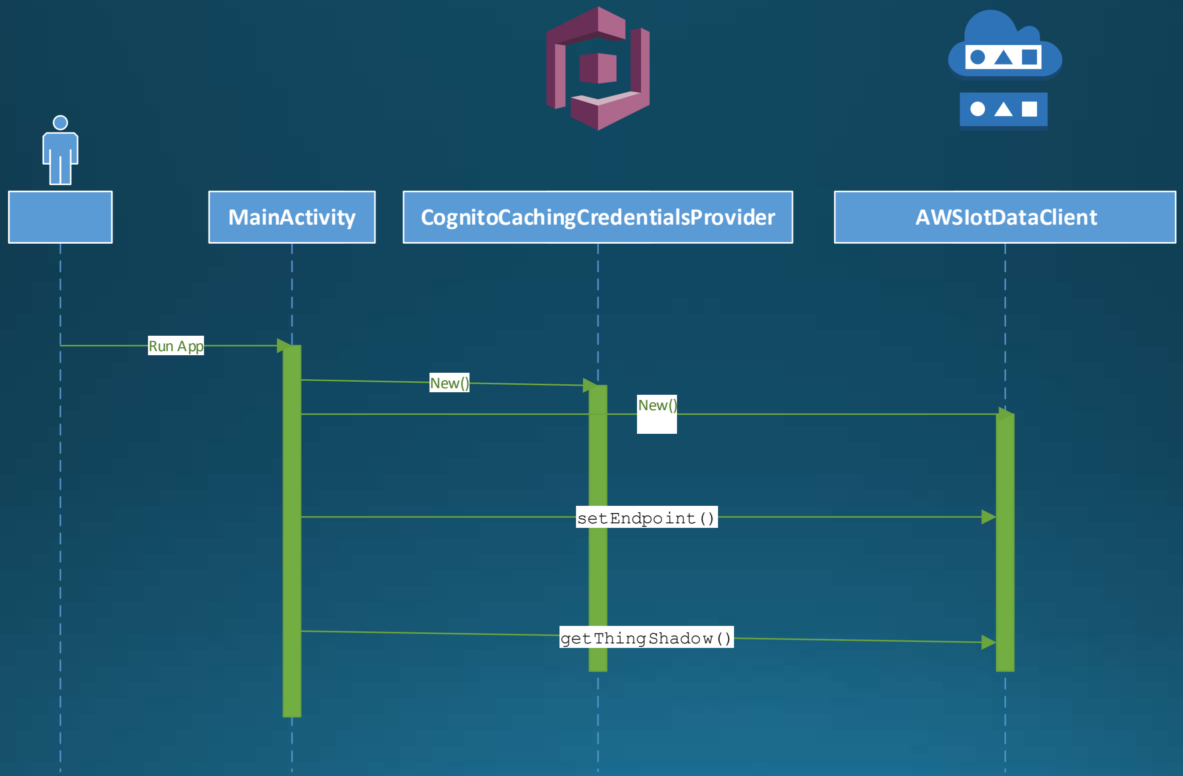
Data JSON

- Outcome data published in “desired”
- Incoming data from “reported” node
- Lock – green LED
- Alarm – red LED



```
{
  "desired": {
    "window": 1,
    "lock": 1,
    "alarm": 0,
    "alarm_reset": 0
  },
  "reported": {
    "window": 1,
    "lock": 1,
    "alarm": 0,
    "alarm_reset": 0
  }
}
```

Android Code Flow



Career Opportunity @ 3mdeb

- Embedded Systems Intern
- Embedded Systems Administrator
- Embedded C Developer
- Embedded Python Developer

- Sent CV to contact@3mdeb.com

Links

- Source code
  - Embedded device code: https://github.com/3mdeb/aws-iot-mqtt-pubsub
  - Mobile Application: https://github.com/rafalkorszun/awsIotExample

- Contact
  - http://kleder.co
  - http://3mdeb.com

Q&A

Thank You!