An Internet of Things (IoT) System Development and Implementation for Automatic Weather Station (AWS) of BMKG based on MQTT Protocol

Ariffudin
Center for Instrumentation, Calibration & Engineering of BMKG, Indonesia

CIMO TECO 2018 Amsterdam, 8 – 11 October 2018

OUTLINE

- Introduction
- AWS of BMKG
- IoT Technology
- MQTT Protocol
- Implementation IoT on AWS
- Resume

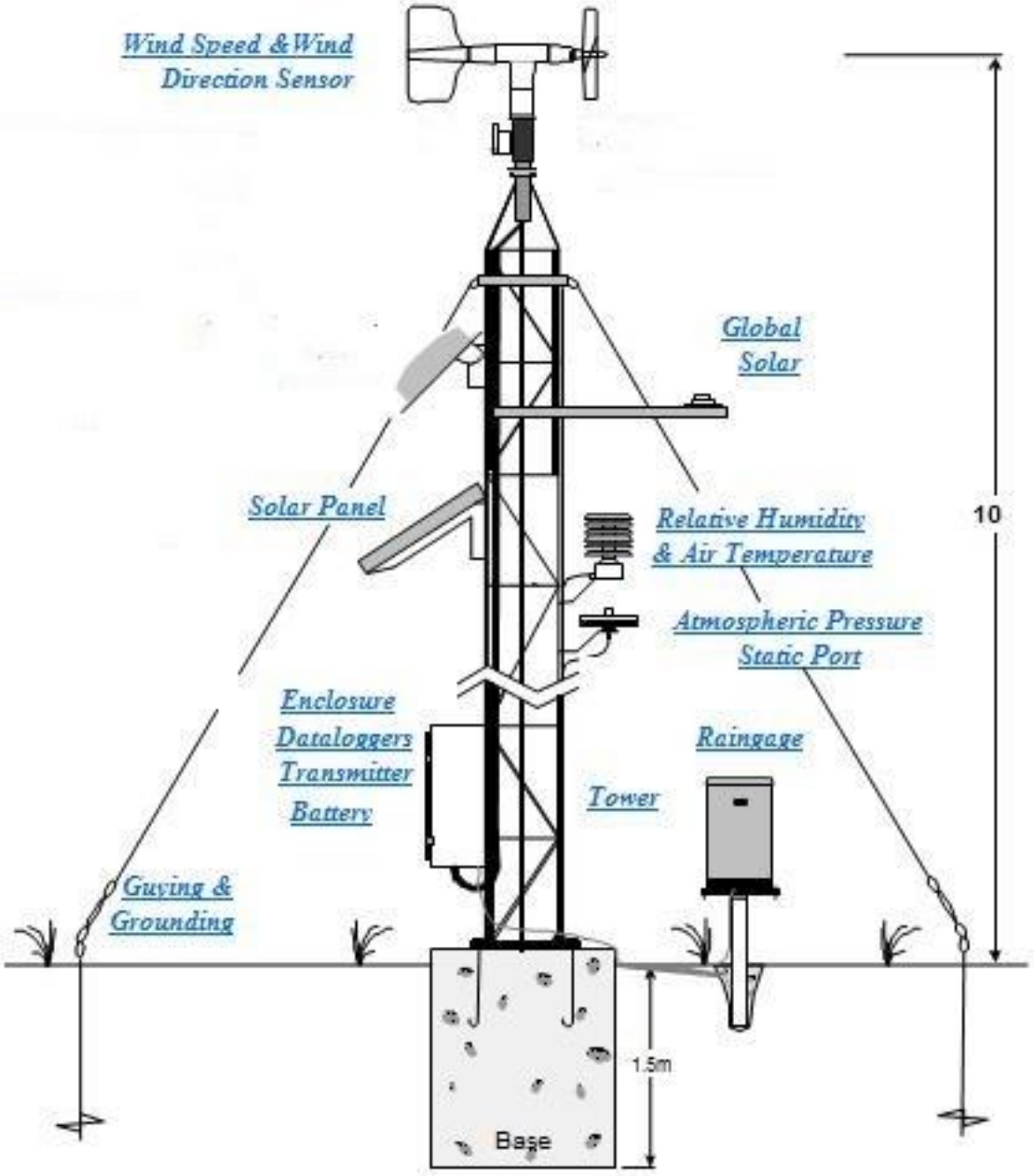
Introduction

BMKG has operated AWS equipment about 430 sites, each with seven sensor parameter using FTP protocol for transfer data to server. System using FTP / HTTP consume more power had a lesser efficiency of transmission and could not utilize system bandwidth efficiently. Traditional web communication technologies, such as FTP and HTTP, provide a uni-directional link and request/response message exchange model. To solve these problems we need adopting the MQTT protocol for the IoT.

AWS (Unmanned) of BMKG

CIMO TECO 2018, Amsterdam

Automatic Weather Station



AWS Network at BMKG

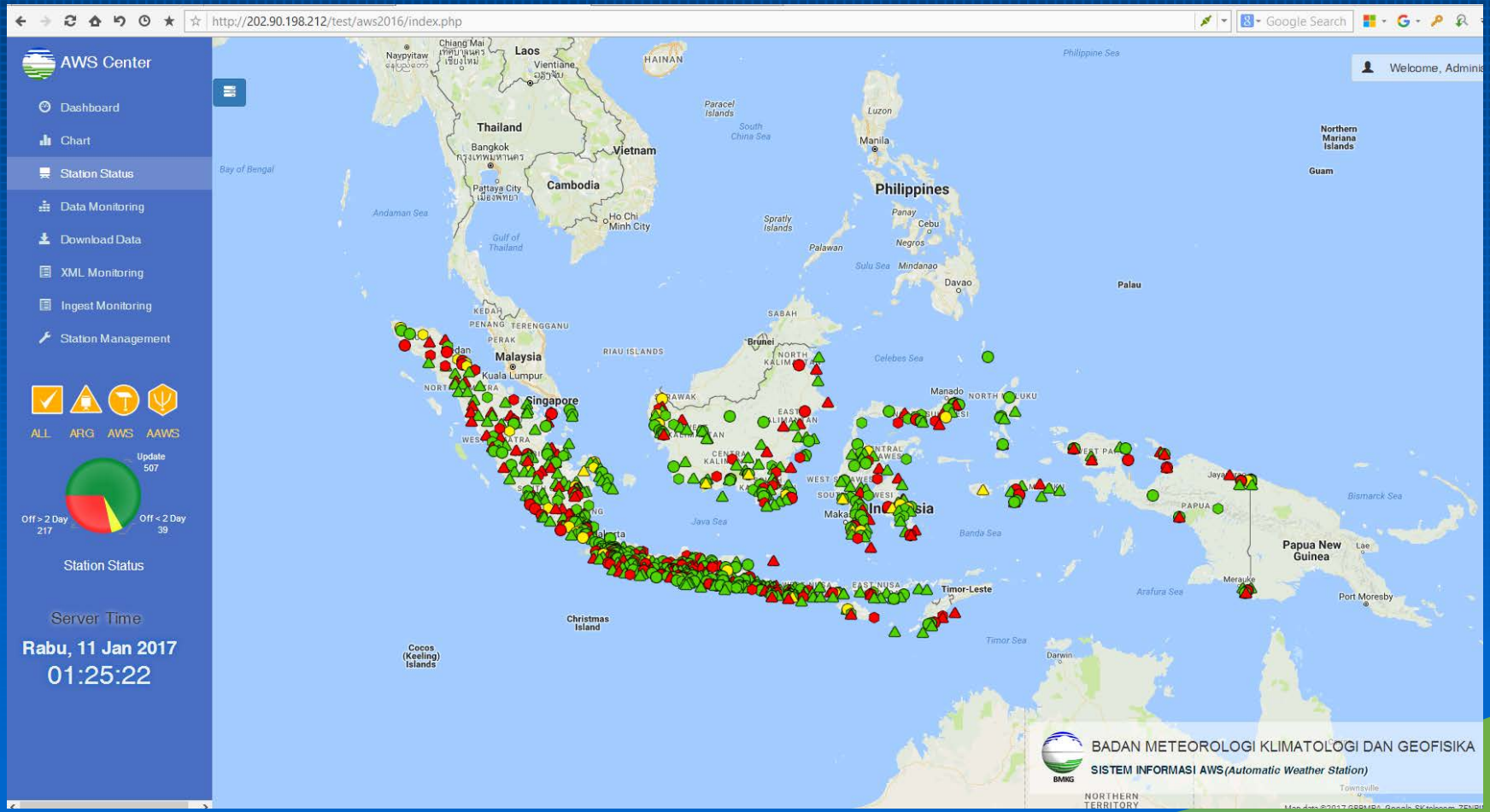


Diagram Network AWS BMKG Existing

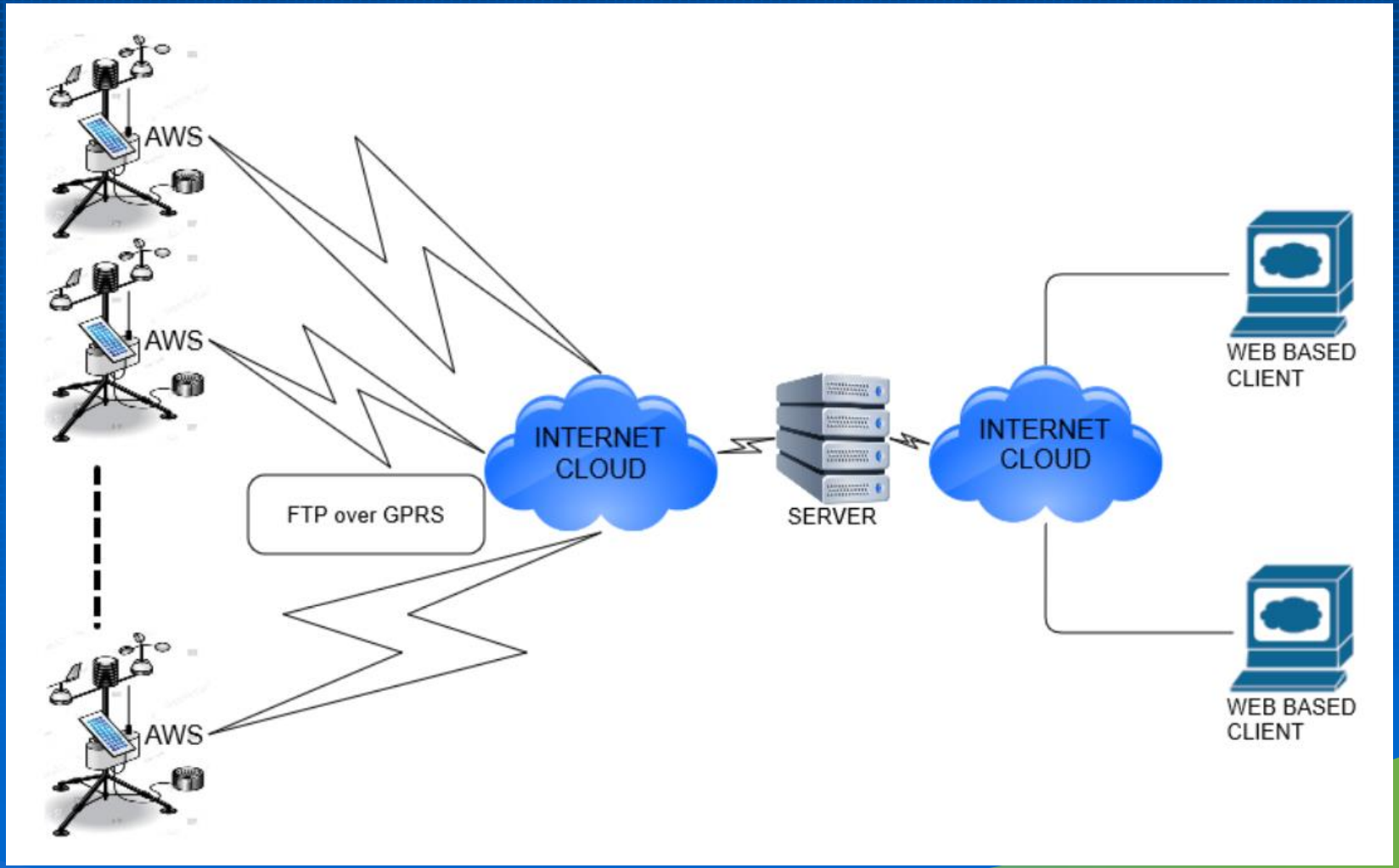
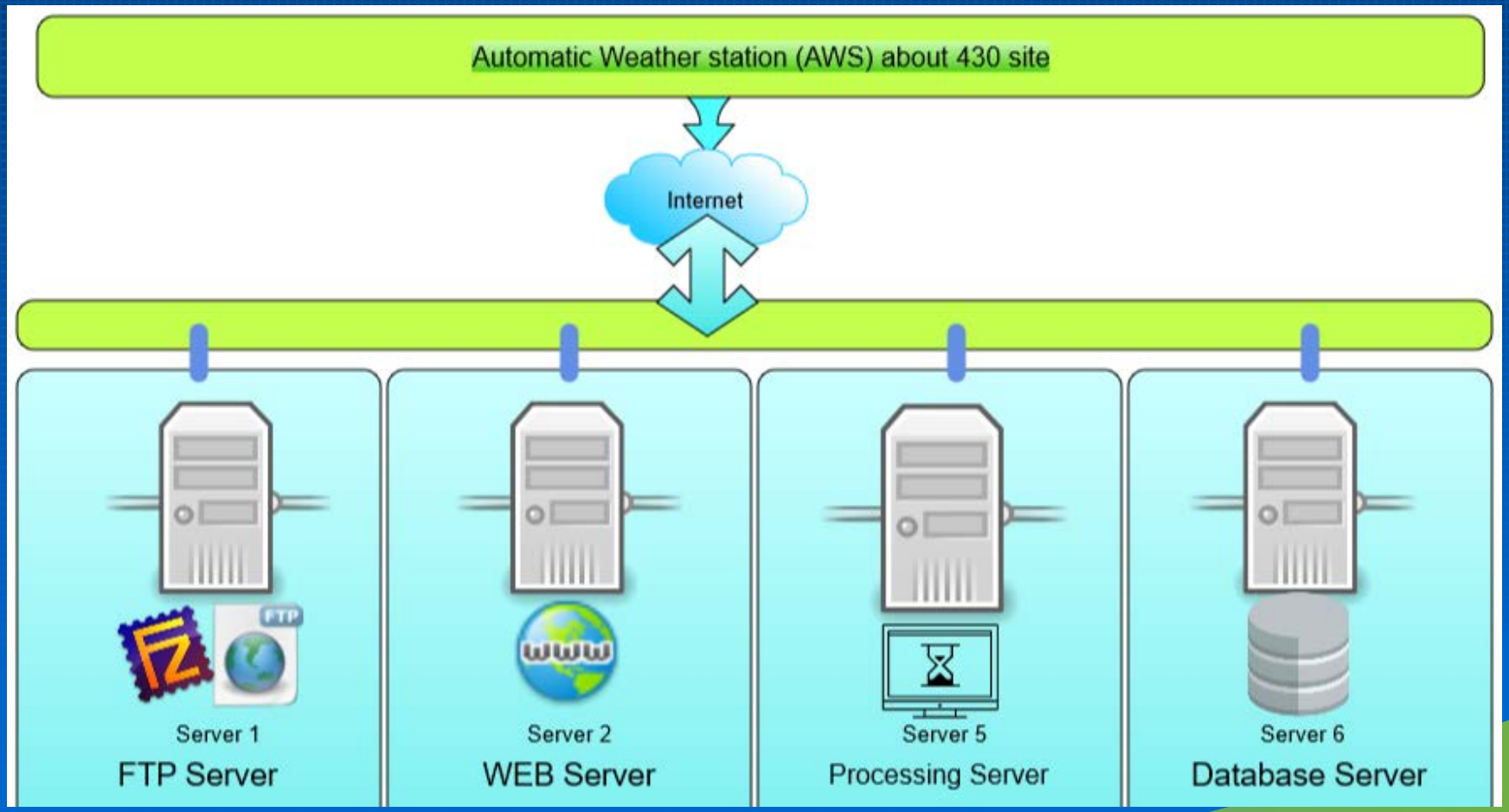
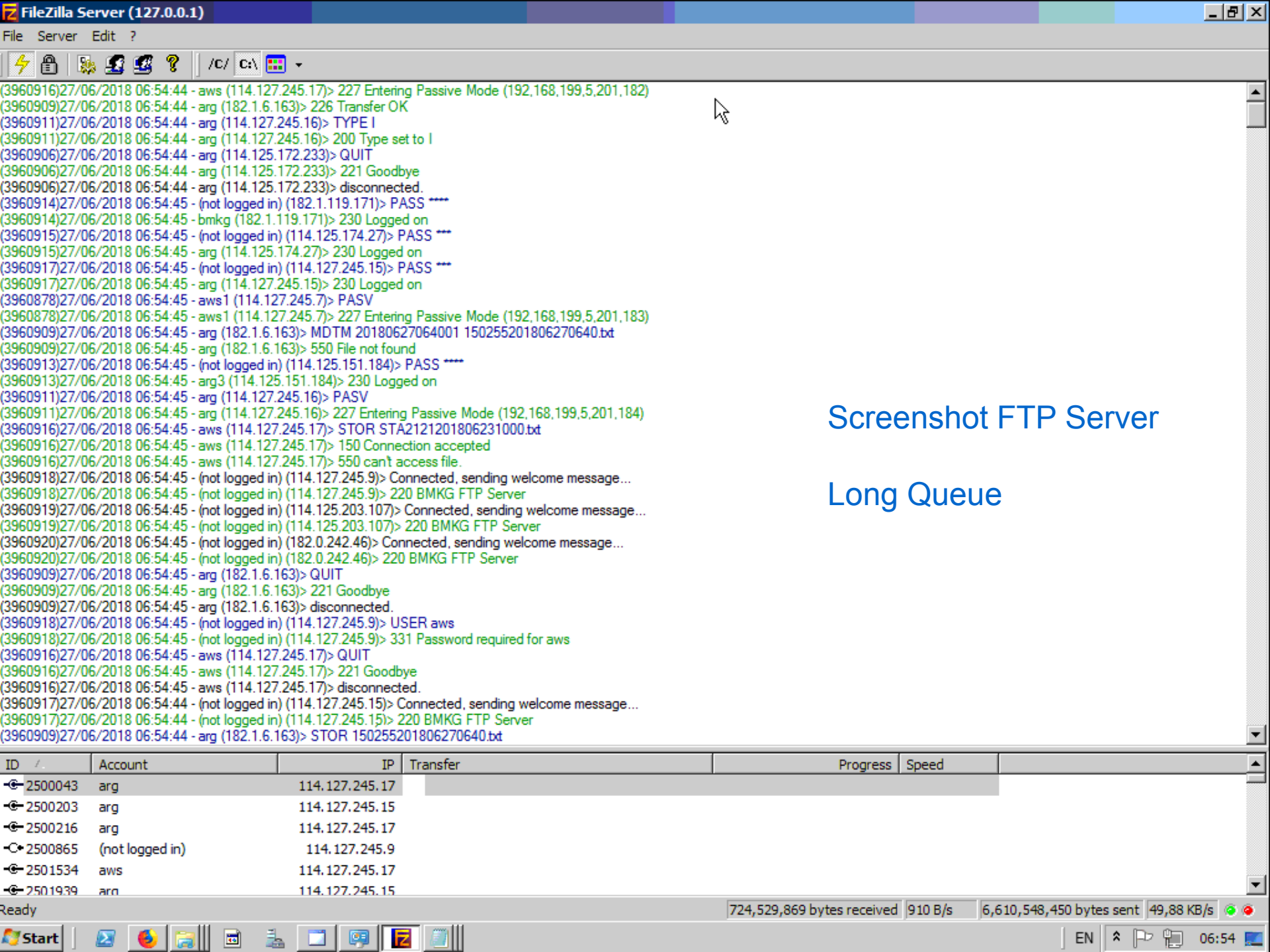


Diagram of AWS server (ftp)




FileZilla Server (127.0.0.1)
File Server Edit ?
(3960916)27/06/2018 06:54:44 - aws (114.127.245.17)> 227 Entering Passive Mode (192,168,199,5,201,182)
(3960909)27/06/2018 06:54:44 - arg (182.1.6.163)> 226 Transfer OK
(3960911)27/06/2018 06:54:44 - arg (114.127.245.16)> TYPE I
(3960911)27/06/2018 06:54:44 - arg (114.127.245.16)> 200 Type set to I
(3960906)27/06/2018 06:54:44 - arg (114.125.172.233)> QUIT
(3960906)27/06/2018 06:54:44 - arg (114.125.172.233)> 221 Goodbye
(3960906)27/06/2018 06:54:44 - arg (114.125.172.233)> disconnected.
(3960914)27/06/2018 06:54:45 - (not logged in) (182.1.119.171)> PASS ****
(3960914)27/06/2018 06:54:45 - bmkg (182.1.119.171)> 230 Logged on
(3960915)27/06/2018 06:54:45 - (not logged in) (114.125.174.27)> PASS ***
(3960915)27/06/2018 06:54:45 - arg (114.125.174.27)> 230 Logged on
(3960917)27/06/2018 06:54:45 - (not logged in) (114.127.245.15)> PASS ***
(3960917)27/06/2018 06:54:45 - arg (114.127.245.15)> 230 Logged on
(3960878)27/06/2018 06:54:45 - aws1 (114.127.245.7)> PASV
(3960878)27/06/2018 06:54:45 - aws1 (114.127.245.7)> 227 Entering Passive Mode (192,168,199,5,201,183)
(3960909)27/06/2018 06:54:45 - arg (182.1.6.163)> MDTM 20180627064001 150255201806270640.txt
(3960909)27/06/2018 06:54:45 - arg (182.1.6.163)> 550 File not found
(3960913)27/06/2018 06:54:45 - (not logged in) (114.125.151.184)> PASS ****
(3960913)27/06/2018 06:54:45 - arg3 (114.125.151.184)> 230 Logged on
(3960911)27/06/2018 06:54:45 - arg (114.127.245.16)> PASV
(3960911)27/06/2018 06:54:45 - arg (114.127.245.16)> 227 Entering Passive Mode (192,168,199,5,201,184)
(3960916)27/06/2018 06:54:45 - aws (114.127.245.17)> STOR STA2121201806231000.txt
(3960916)27/06/2018 06:54:45 - aws (114.127.245.17)> 150 Connection accepted
(3960916)27/06/2018 06:54:45 - aws (114.127.245.17)> 550 can't access file.
(3960918)27/06/2018 06:54:45 - (not logged in) (114.127.245.9)> Connected, sending welcome message...
(3960918)27/06/2018 06:54:45 - (not logged in) (114.127.245.9)> 220 BMKG FTP Server
(3960919)27/06/2018 06:54:45 - (not logged in) (114.125.203.107)> Connected, sending welcome message...
(3960919)27/06/2018 06:54:45 - (not logged in) (114.125.203.107)> 220 BMKG FTP Server
(3960920)27/06/2018 06:54:45 - (not logged in) (182.0.242.46)> Connected, sending welcome message...
(3960920)27/06/2018 06:54:45 - (not logged in) (182.0.242.46)> 220 BMKG FTP Server
(3960909)27/06/2018 06:54:45 - arg (182.1.6.163)> QUIT
(3960909)27/06/2018 06:54:45 - arg (182.1.6.163)> 221 Goodbye
(3960909)27/06/2018 06:54:45 - arg (182.1.6.163)> disconnected.
(3960918)27/06/2018 06:54:45 - (not logged in) (114.127.245.9)> USER aws
(3960918)27/06/2018 06:54:45 - (not logged in) (114.127.245.9)> 331 Password required for aws
(3960916)27/06/2018 06:54:45 - aws (114.127.245.17)> QUIT
(3960916)27/06/2018 06:54:45 - aws (114.127.245.17)> 221 Goodbye
(3960916)27/06/2018 06:54:45 - aws (114.127.245.17)> disconnected.
(3960917)27/06/2018 06:54:44 - (not logged in) (114.127.245.15)> Connected, sending welcome message...
(3960917)27/06/2018 06:54:44 - (not logged in) (114.127.245.15)> 220 BMKG FTP Server
(3960909)27/06/2018 06:54:44 - arg (182.1.6.163)> STOR 150255201806270640.txt
Screenshot FTP Server
Long Queue
ID Account IP Transfer Progress Speed
2500043 arg 114.127.245.17
2500203 arg 114.127.245.15
2500216 arg 114.127.245.17
2500865 (not logged in) 114.127.245.9
2501534 aws 114.127.245.17
2501939 arg 114.127.245.15
Ready
724,529,869 bytes received
910 B/s
6,610,548,450 bytes sent
49,88 KB/s
Start
EN
06:54

IOT TECHNOLOGY WITH MQTT

IBM Define IoT as

The Internet of Things represents an evolution in which objects are capable of interacting with other objects.

The Internet of Things helps enable proactive data access from any connected device.

Why IoT

General reasons - why IoT ?

- Connect both inanimate and living things
- Use sensors for data collection (data streams)
- Change what types of item communicate over an IP Network
- Benefits for today:
  - Faster knowledge/stats sharing with M2M
  - Better quality in control and automation
  - Cost Savings (Machine Learning, Analytics)

IoT protocols landscape

AMQP

CoAP

MQTT

XMPP

DDS

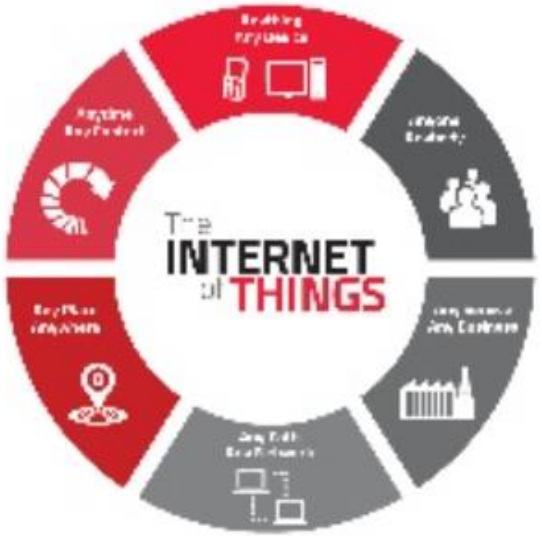
STOMP

MQTT Use Cases

- Push instead of Poll
- Low Latency
- Minimal Bandwidth is important
- Constrained devices
- Unreliable networks

Challenges

- *Emitting one to many*
- *Listening continuously for events*
- *Minimal data in huge volumes*
- *Working over unreliable networks*



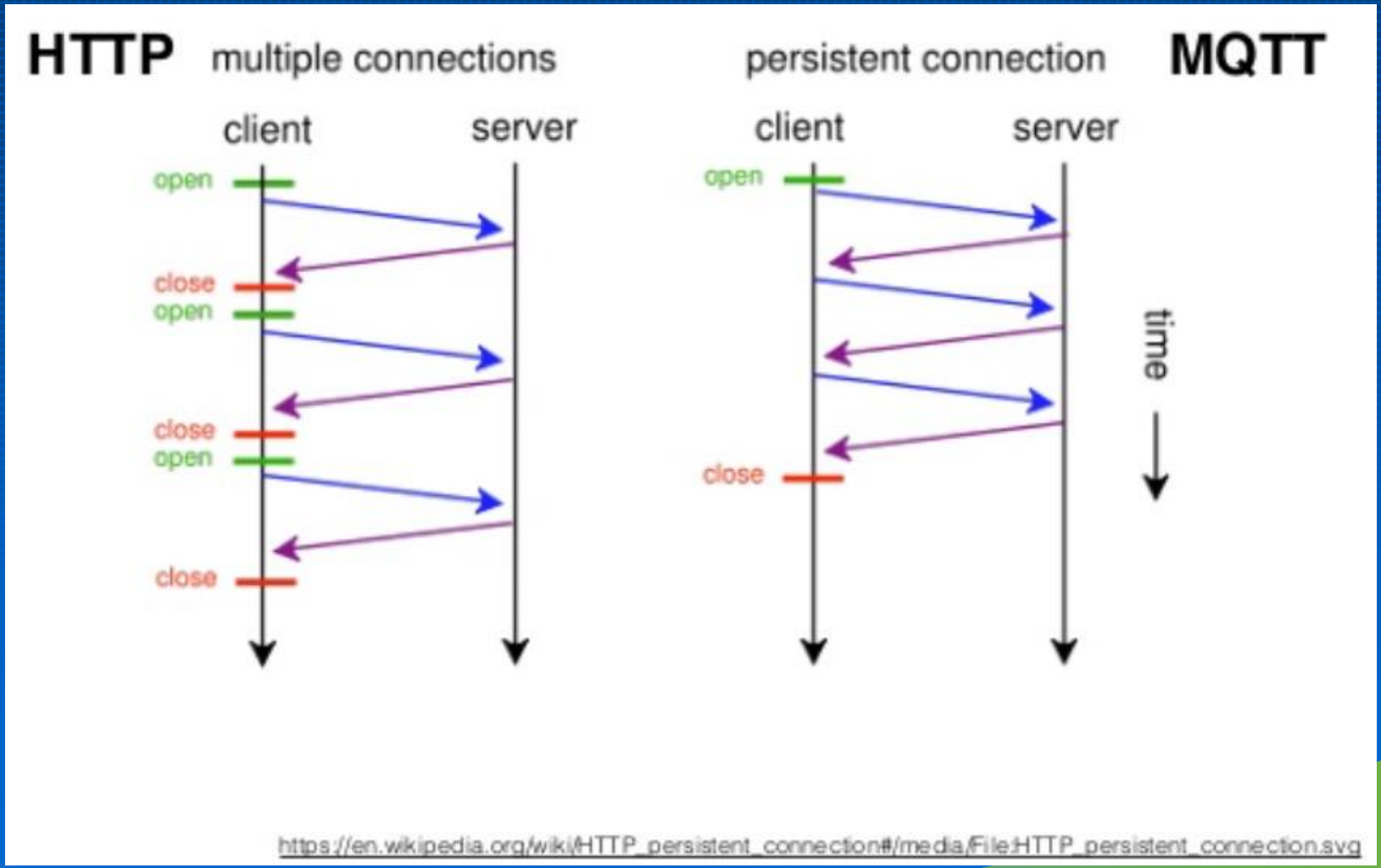
and for mobile or M2M apps:

- *Volume of data = cost (4G data plans)*
- *Power consumption*
- *Security and privacy…*
- *Scalability*

MQTT – Key things to remember

- Capabilities
  - One-many publish / Subscribe
  - Reliable store+forward
  - Bidirectional communications
  - Long-running connections
- Decoupling
  - Easy to add new message producers or consumers
- Simplicity
  - Small protocol, small clients (kBytes)
  - Implementable on low power devices
- Network efficiency
  - Small headers
  - Avoids polling
- Event-orientation
  - Near real-time notification of events

Response Times



https://en.wikipedia.org/wiki/HTTP_persistent_connection#/media/File:HTTP_persistent_connection.svg

Low battery Use

Initial connection to server

% Battery Used			
3G		Wifi	
HTTPS	MQTT	HTTPS	MQTT
0.02972	0.04563	**0.00228**	0.00276

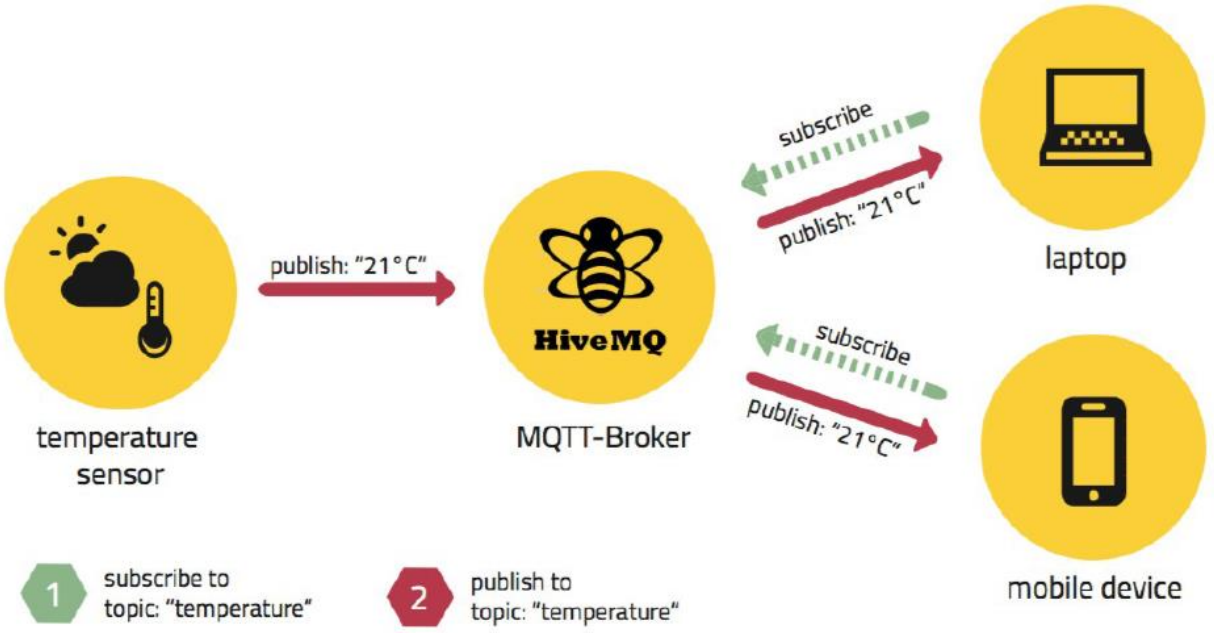
Credit: Stephen Nicolas
http://stephendnicholas.com/archives/1217

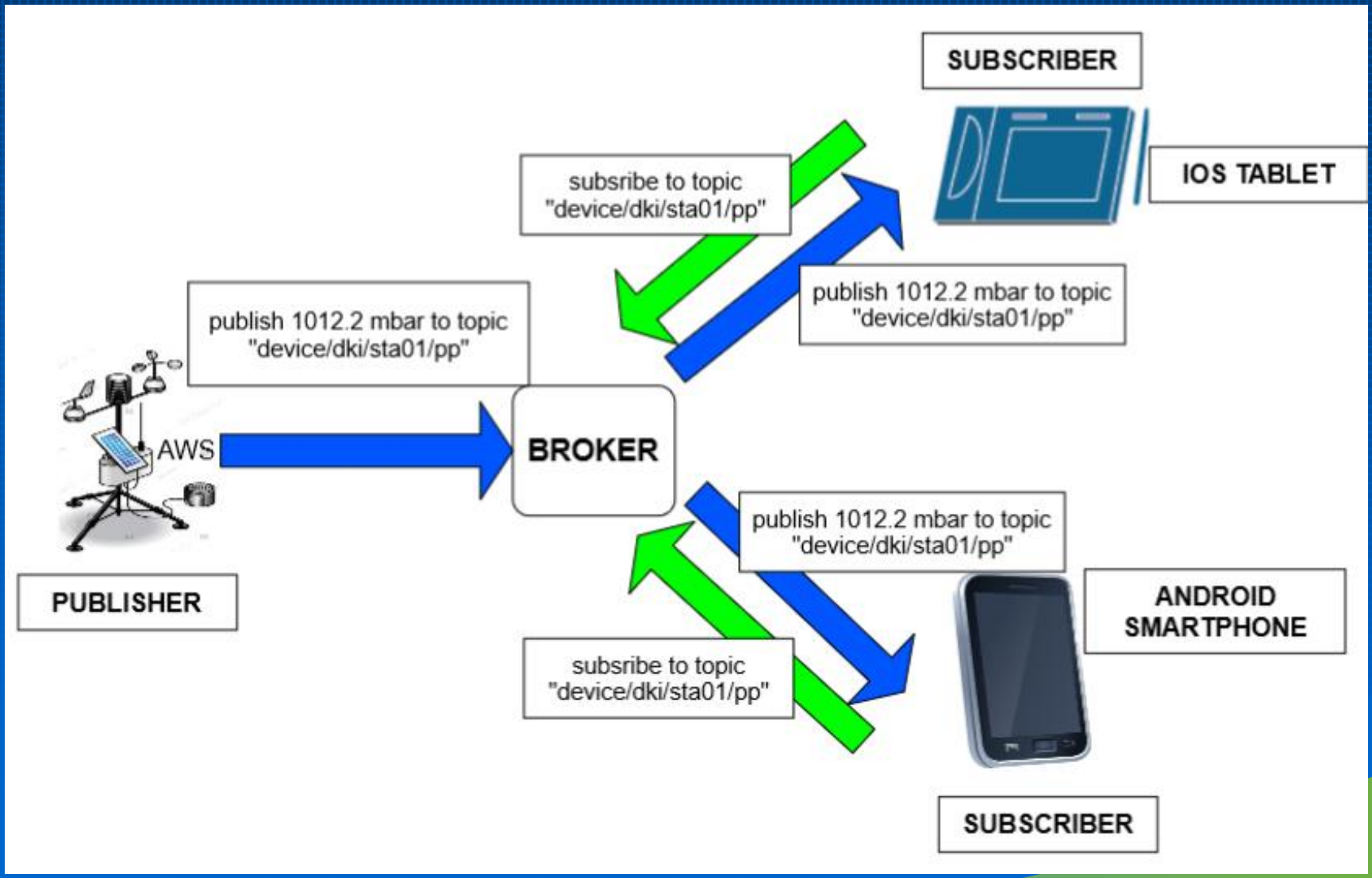
Scalability

"a single 1u-form-factor appliance can handle up to 1 million sensors and 13 million concurrent messages"

link to article

Publish / Subscribe




SUBSCRIBER
IOS TABLET
subsribe to topic
"device/dki/sta01/pp"
publish 1012.2 mbar to topic
"device/dki/sta01/pp"
publish 1012.2 mbar to topic
"device/dki/sta01/pp"
AWS
BROKER
PUBLISHER
publish 1012.2 mbar to topic
"device/dki/sta01/pp"
ANDROID
SMARTPHONE
subsribe to topic
"device/dki/sta01/pp"
SUBSCRIBER

Topics

- Case sensitive
- UTF-8
- Wildcards
  - Single level: building/+/humidity
    - building/room_4/humidity
    - building/room_67/humidity
    - building/room_78/humidity
  - Multiple level (only at the end): building/room_number/#
    - building/room_4/wall/temperature
    - building/room_4/wall/humidity
    - building/room_4/ceiling/temperature
    - building/room_4/ceiling/humidity

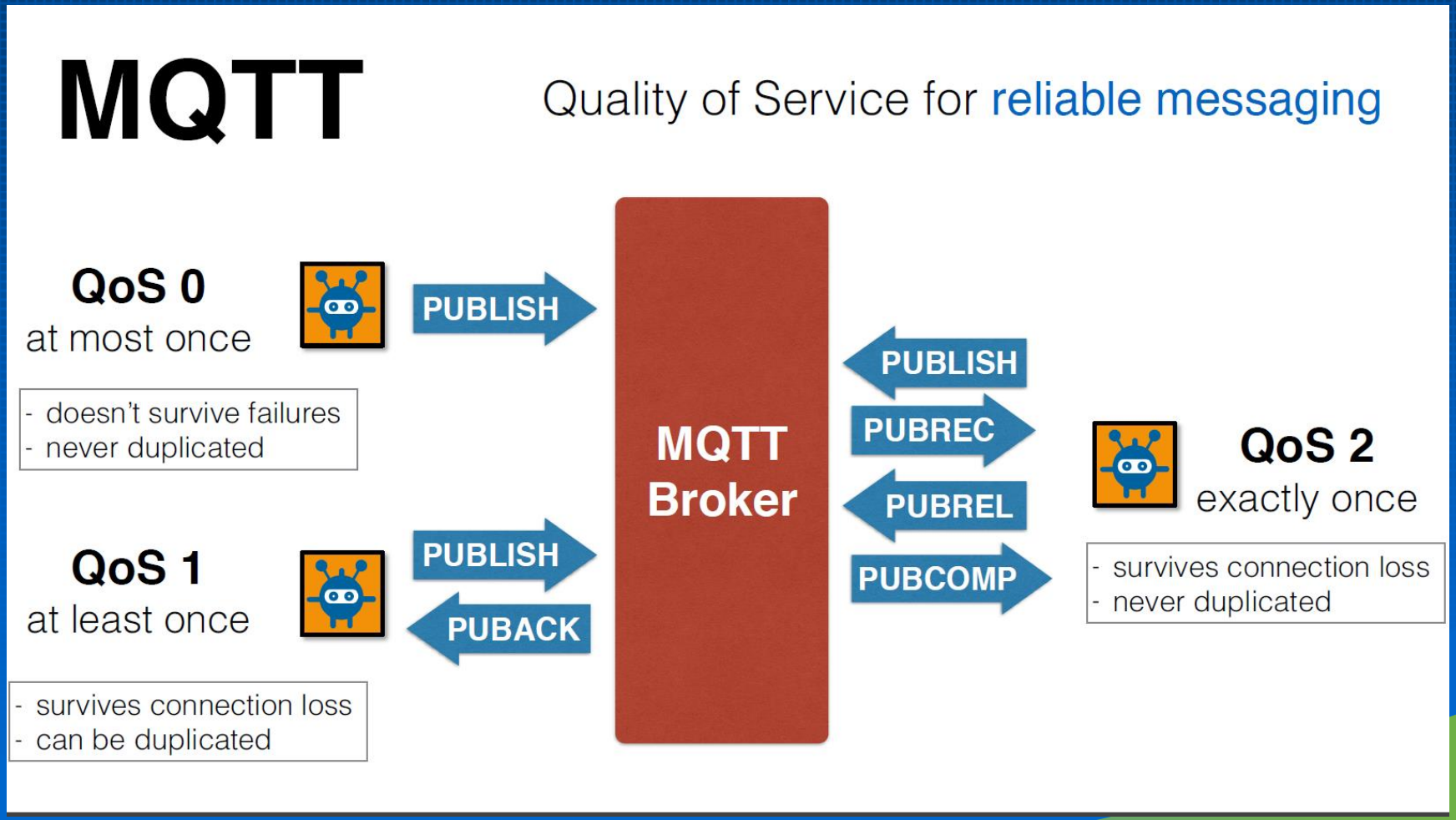
TOPICS

- Case sensitive
- Wildcards for filtering
  - device/province/site_id/humidity
  - Single level : device/jakarta/+/humidity
    - Device/jakarta/aws01/humidity
    - Device/jakarta/aws02/humidity
    - Device/jakarta/aws03/humidity
  - Multiple level : device/jakarta/#
    - Device/jakarta/aws01/pressure
    - Device/jakarta/aws02/temperature
    - Device/jakarta/aws04/rain

Quality of Service

Makes communication in unreliable networks a lot easier because it handles retransmission and guarantees the delivery of the message.

QOS


MQTT
Quality of Service for reliable messaging
QoS 0
at most once
- doesn't survive failures
- never duplicated
PUBLISH
QoS 1
at least once
- survives connection loss
- can be duplicated
PUBLISH
PUBACK
MQTT Broker
PUBLISH
PUBREC
PUBREL
PUBCOMP
QoS 2
exactly once
- survives connection loss
- never duplicated

Security

SSL/TLS TCP/IP

CONNECT with **username / password**

- MQTT spec doesn't define security model aside from username/password authorization on connection
- Brokers *can* implement support for SSL/TLS and policies for connection and messaging

ex. organize topic space by "group"
username associated with a group

bboyd is in group "IBM" and can pub/sub `IBM/bboyd/#`

MQTT Broker

BROKER MQTT

The Broker is responsible for:

- *Receiving all messages and filtering*
- *Decide who is interested*
- *Send the message to all subscribed clients*
- *Authentication and authorisation of clients*

It also holds the session of all persisted clients including subscriptions and missed messages

The broker is the central hub, which every message has to pass

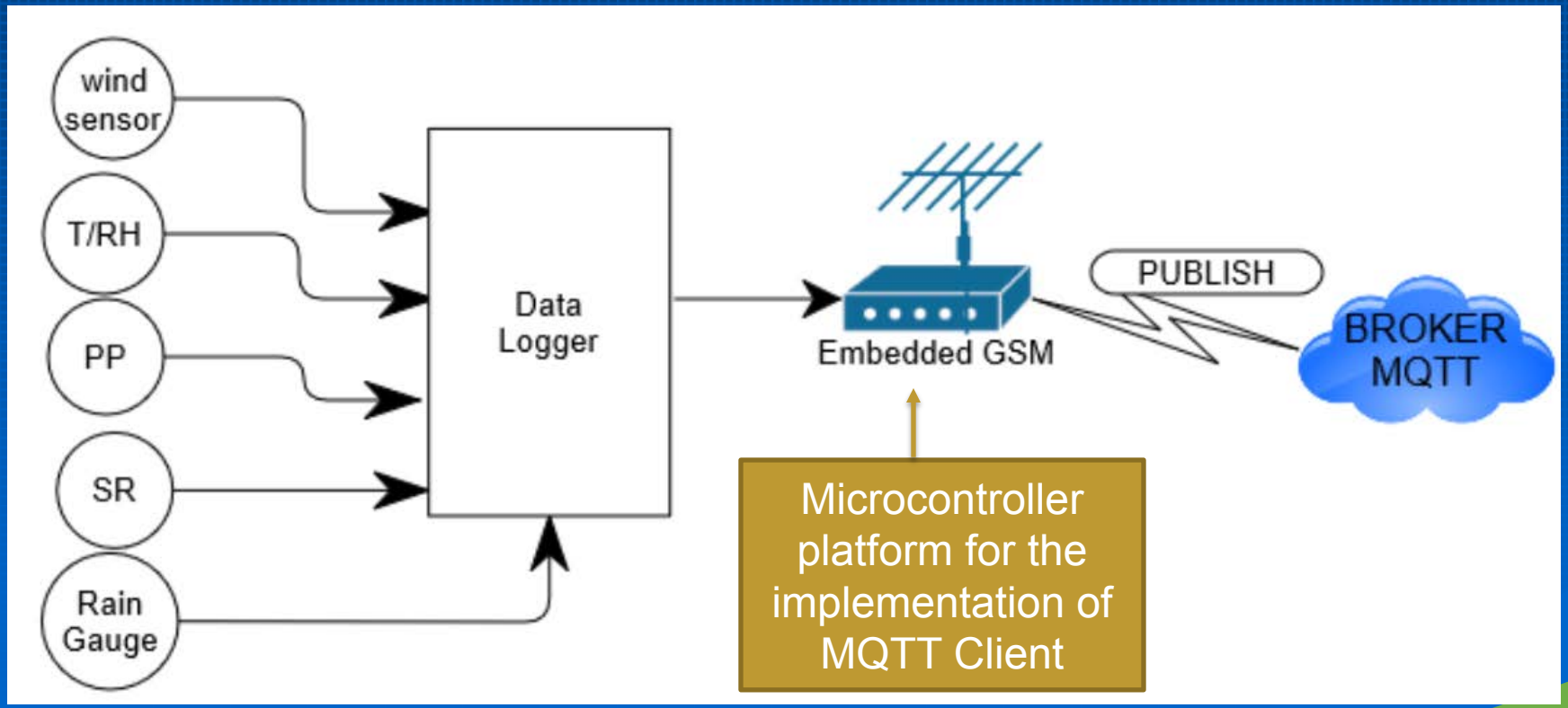
Facebook Messenger

*"One of the **problems** we experienced was **long latency** when sending a message. The method we were using to send was reliable but **slow**, and there were **limitations** on how much we could improve it. With **just a few weeks** until launch, we ended up building a new mechanism that maintains a persistent connection to our servers. To do this without **killing battery life**, we used a protocol called **MQTT** that we had experimented with in Beluga. MQTT is specifically designed for applications like sending telemetry data to and from space probes, so it is **designed to use bandwidth and batteries sparingly**. By maintaining an MQTT connection and routing messages through our chat pipeline, we were able to often achieve **phone-to-phone delivery in the hundreds of milliseconds, rather than multiple seconds**."*

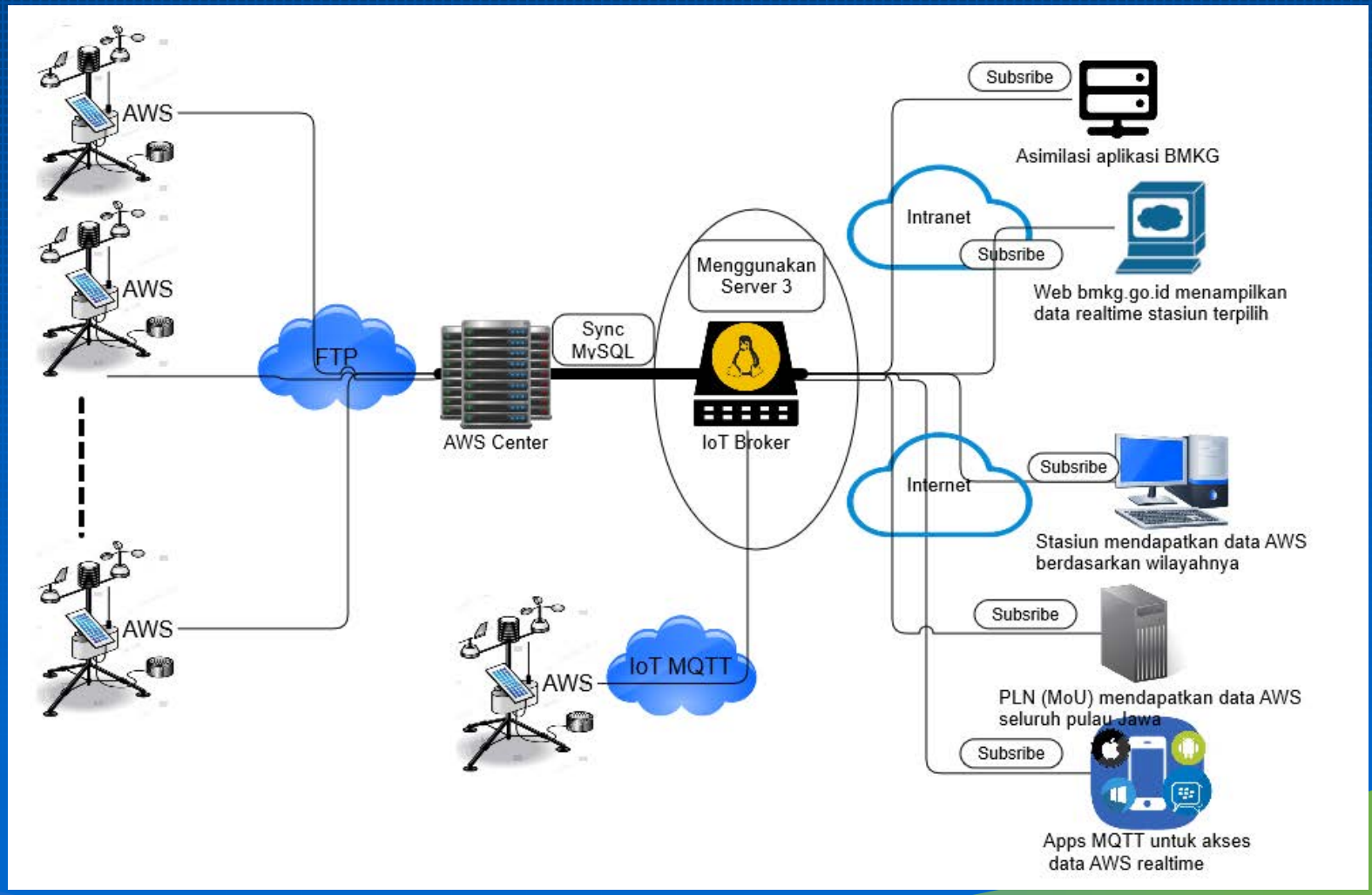
-Lucy Zhang, Facebook Engineer 10/19/11 www.facebook.com/lucyz
(MQTT used by their 350M mobile users, 475 mobile operators)

http://www.youtube.com/watch?v=aJo5jG0eKtI&hd=1

Layout of AWS for MQTT

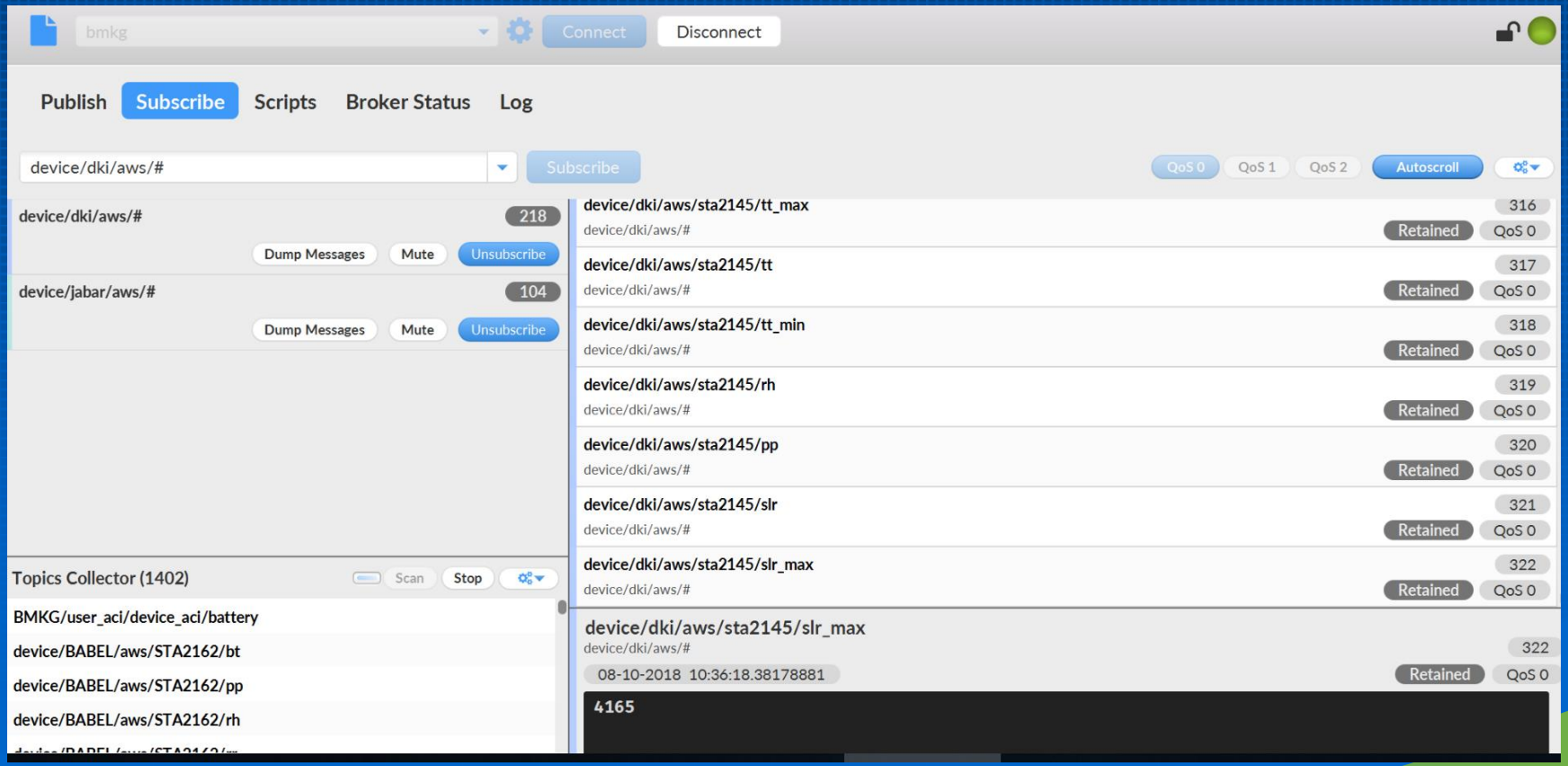


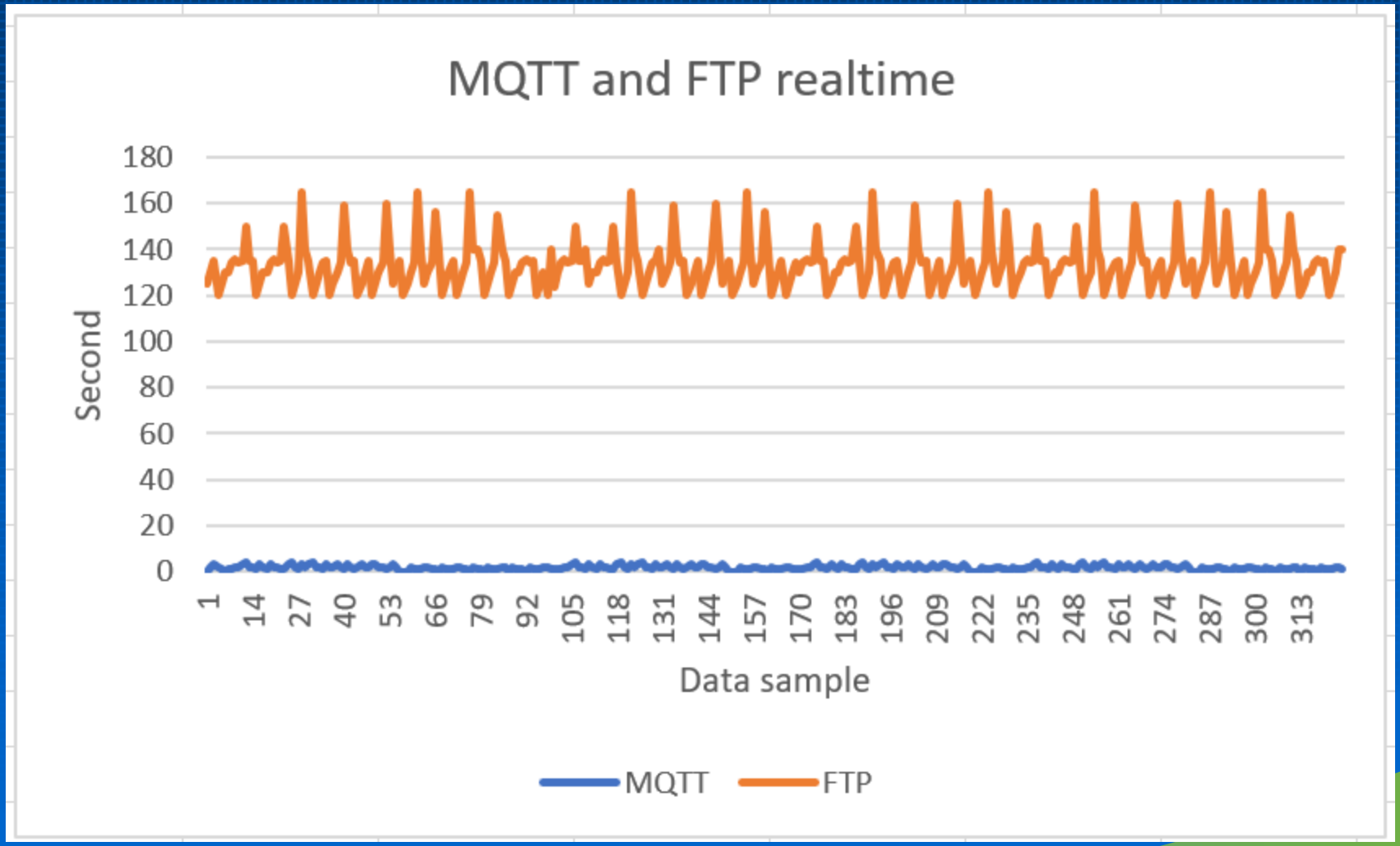
AWS - IoT BMKG


AWS
AWS
AWS
AWS
FTP
AWS Center
Sync MySQL
Menggunakan Server 3
IoT Broker
IoT MQTT
Subsribe
Asimilasi aplikasi BMKG
Intranet
Subsribe
Web bmkg.go.id menampilkan data realtime stasiun terpilih
Internet
Subsribe
Stasiun mendapatkan data AWS berdasarkan wilayahnya
Subsribe
PLN (MoU) mendapatkan data AWS seluruh pulau Jawa
Subsribe
Apps MQTT untuk akses data AWS realtime

Android Client with MQTTDash

Screenshot mqtt.fx client on Windows OS




MQTT and FTP realtime
Second
180
160
140
120
100
80
60
40
20
0
1
14
27
40
53
66
79
92
105
118
131
144
157
170
183
196
209
222
235
248
261
274
287
300
313
Data sample
MQTT
FTP

Resume

- This study has implemented the use of MQTT protocol to build weather station system application with an embedded modem communication interface. The test result indicates that MQTT protocol can transfer data faster than FTP protocol.
- AWS with IoT enabling device to be connected easily with internet and corresponding information can be accessible globally.
- Streamline the existing AWS system

Thank You
Terima Kasih

I Don’t Always do M2M Communicatrion
But when i do, i use MQTT

Email : ariffudin@bmkg.go.id