IQRF Architecture

Present and Vision

Hynek Syrovátka
Rostislav Špinar

January 11, 2017, Prague

What will we cover?

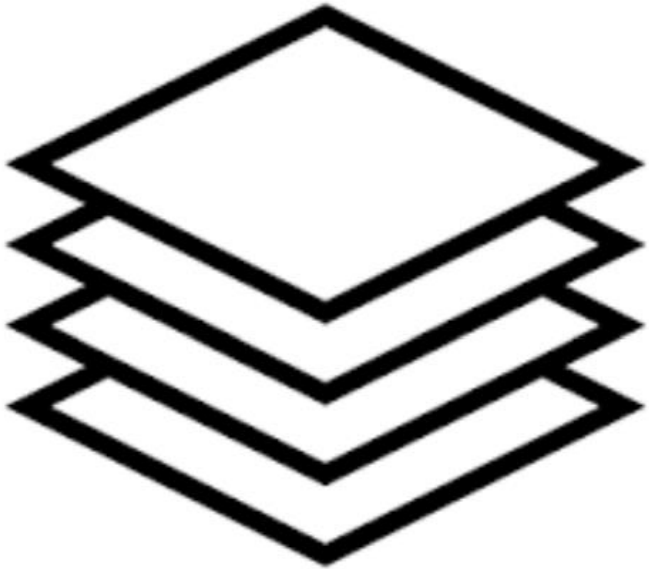
- IQRF architecture from a high level view
- Layer details
- News of upcoming OS and DPA
- Future proposals

IQRF Architecture

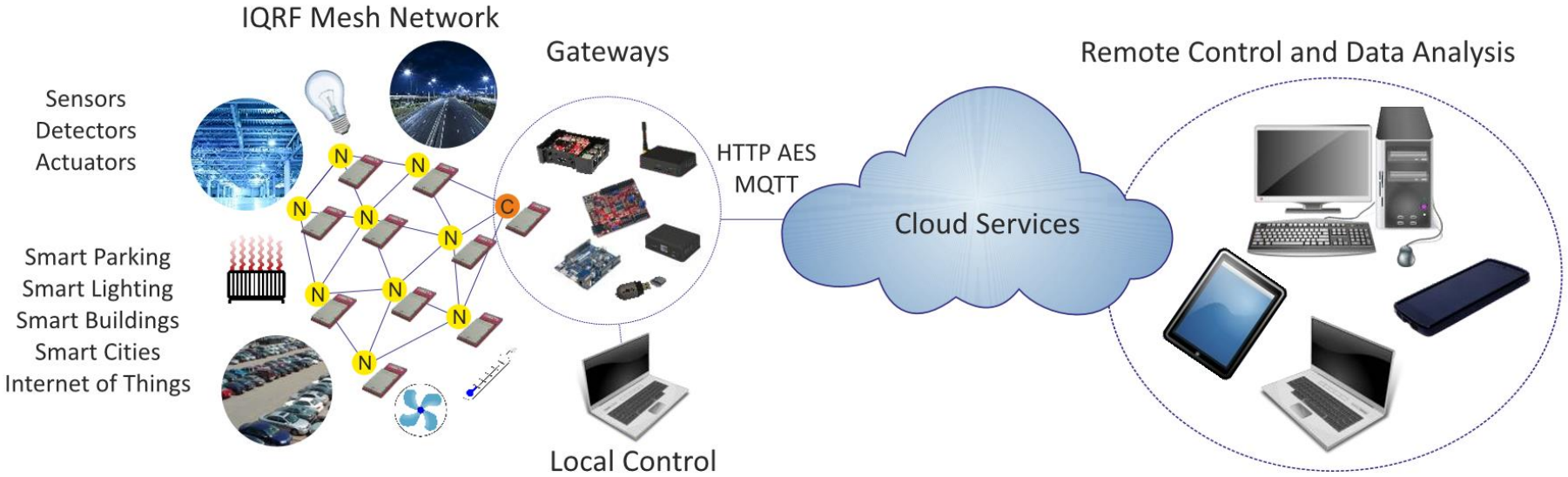
- One of key IQRF ecosystem elements
- When well and completely designed then it is an enabler for
  - Device compatibility
  - Device interoperability
  - Integration simplicity

IQRF Architecture

- Layered
- High level view
  - Module
  - Device
  - Gateway
  - Cloud & Application

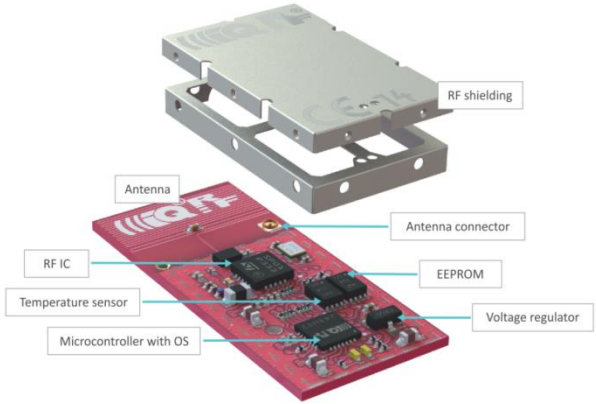


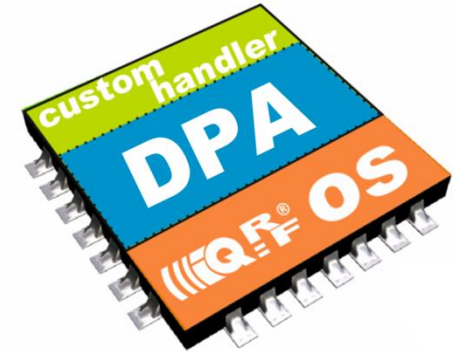
IQRF Architecture



Module Layer

- IQRF OS
  - RF & IQMESH network
  - Security, HW and basic services
  - Development means
- DPA
  - IQRF OS encapsulation/transparency
  - DPA protocol => device compatibility
  - High level services and peripherals
  - Device software customization
  - OTA



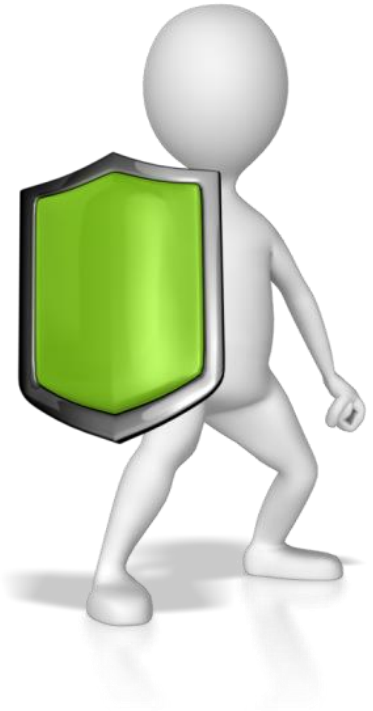


IQRF OS 4.00 News

- Improved security
- Node authorization based on full 4 byte MID
- Bonding data extended to 4 bytes
- Deep sleep mode
- External EEPROM reorganization

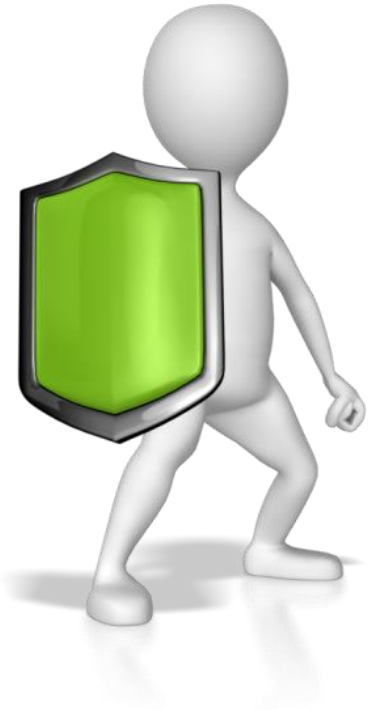
IQRF OS Security Principles

- Simplicity
- Complexity
- Adequacy
- Understandability
- Predictability
- AES-128



IQRF OS Security Features

- Bonding security
- Data consistency protection
- Network communication encryption
- Networks communication isolation
- Custom data encryption



DPA 3.00 News

- TR5x support dropped
- Demo DPA and [CN] device not released
- IQRF OS security support
- More user peripherals and commands available
- Backup, Restore and DSM secured by AES-128
- Remote bonding up to 7 nodes per device
- OTA update of both IQRF OS & DPA at the same time

Device Layer

- Defines device behavior
- Uses DPA protocol to ensure compatibility
- Subject to wide heterogeneity and diversity
- Standard HWP behavior standardized & described today
  - Using text-based description

Device Layer - Standardization

- Target group: integrators, developers, coders, …
- Unified device description style
- Available from a global internet IQRF repository to applications
- Standardized device classes to ensure interoperability
- Description style?
  - Usual “literally literary” style

    ≠

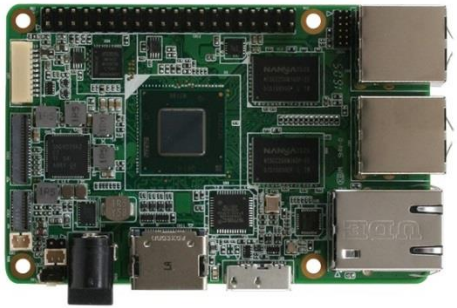
  - Structured & procedural machine readable style = “driver”
- Standardization body = IQRF Alliance - technical committee

„Driver“ kind of Device Description

Connect Device

=

Download Driver



Gateway - Goals

1. **Any robust HW** board running Linux/Windows
   (AAEON boards, Zyxel devices, Raspberry PI, etc. for quick prototyping)
2. **IQRF SDK** libraries and components as a basis
   (GitHub.com/IQRFSDK)
3. **Additional GW services** come from cooperation with partners in Alliance
   (Management services, deployment tools)

Gateway – Steps

1. **Any robust HW** board running Linux/Windows
   (AAEON boards, Zyxel devices, Raspberry PI, etc. for quick prototyping)
2. **IQRF SDK** libraries and components as a basis
   (GitHub.com/IQRFSDK)
3. **Additional GW services** come from cooperation with partners in Alliance
   (Management services, deployment tools)

1. **Establish working group**
   (IQRF standardization working group)
2. **Define standard interfaces/protocols/channels** to IQRF and Servers
   (SPI, CDC, UDP, JSON, MQ, MQTT)
3. **Define additional components and services** for GW layer
   (Management, deployment, etc.)

Invitation

1. **Any robust HW** board running Linux/Windows
   (AAEON boards, Zyxel devices, Raspberry PI, etc. for quick prototyping)
2. **IQRF SDK** libraries and components as a basis
   (GitHub.com/IQRFSDK)
3. **Additional GW services** come from cooperation with partners in Alliance
   (Management services, deployment tools)

1. **Establish working group**
   (IQRF standardization working group)
2. **Define standard interfaces/protocols/channels** to IQRF and Servers
   (SPI, CDC, UDP, JSON, MQ, MQTT)
3. **Define additional components and services** for GW layer
   (Management, deployment, etc.)

1. **Cooperation** among members of Alliance to deliver GW solution
   (MICRORISC, O2ITS, AAEON, Zyxel, IQHome, RehiveTech and more)

IQRF Architecture

Technical Committee

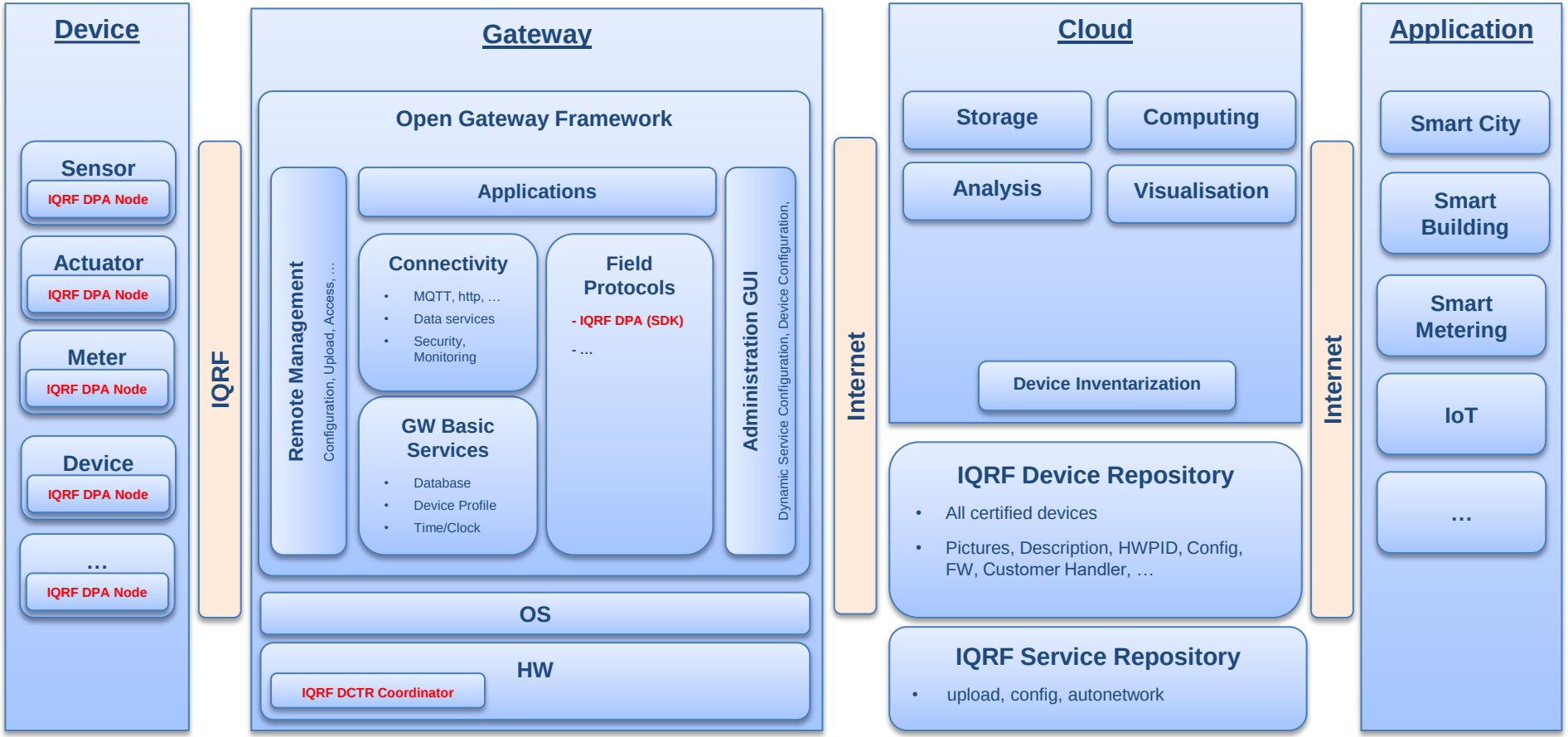
Hynek Syrovátka
Rostislav Špinar

January 11, 2017, Prague

Agenda

1. Module layer - IQRF DPA and OS – practical demonstration
2. Device layer – Standard Description and Commands
3. Gateway layer – IQRF SDK and Middleware
4. Discussion

IQRF Architecture



Practical demonstration

- IQRF OS 4.00 and DPA 3.00

Device

Device layer – Standardization and Description

- Goals
  - Unification
  - Device compatibility
  - Device interoperability
  - Integration simplicity
- Means
  - Structured device (class) description
  - Including procedural code
  - Standardized device classes
  - Global internet repository

Device Layer - Description

- HWPID uniquely identifies the device
- HWPID is a query key at IQRF description repository
- Device description
  - Version, date, name, manufacturer, URLs to picture, documentation, …
  - DPA handler
  - Supported device classes
    - ClassID (for standard one)
    - Version, date, name, …
      - Requests - procedures
      - FRCs - procedures

Device Layer – Description Usage

1. Application (middleware or cloud) queries repository with HWPID
2. Downloads description
3. Finds out (standardized) device classes to use
4. Uses prepared procedures to prepare device requests
5. Uses prepared procedures parse device responses
6. Same with FRCs
7. Alternatively the developer implements the code based on understanding of the description

Device Layer – Description

- Discussion…

Gateway

Goals

1. Introduce IQRF Software Development Kit (SDK)
    A. SDK for **Embedded gateways and devices**
    B. SDK for **Linux/Windows gateways**
    C. SDK for **Linux/Windows servers**

Basic motivation: Ease the task of integration DPA into your devices and systems

2. Introduce IQRF Gateway based on SDK
    A. IQRF daemon **introduction**
    B. IQRF daemon **practically**
    C. Standardization – **working group**

Basic motivation: Ease the task of prototyping, piloting DPA systems
Reference design for IQRF gateways coming from Alliance partners

IQRF SDK

SDK Introduction

1. **Embedded gateways and devices**

- https://github.com/iqrfsdk/clibdpa-mcu
- C based DPA library for no-OS devices
- Examples, Auto-Network, Upload

- Arduino, chipKIT, bare metal
- IQRF SPI and DPA HDLC UART
- IQRF-SHIELD-03, IQRF-BB-01, IQRF-BEE-01

DEPLOYED:

- Quick prototyping
- Actissim.com

SDK Introduction

2. Linux/Windows gateways

- https://github.com/iqrfsdk/clibcdc
- https://github.com/iqrfsdk/clibspi
- https://github.com/iqrfsdk/cutils
- https://github.com/iqrfsdk/clibdpa

- C++ based libraries
- IQRF SPI, IQRF CDC interfaces
- Examples

- AAEON UP2, Raspberry PI, Linux/Windows platforms
- KON-RASP-01, IQRF-BB-01, GW-USB-06

DEPLOYED:

- IQRF daemon

SDK Introduction

2. Linux/Windows gateways

- https://github.com/iqrfsdk/jlibcdc
- https://github.com/iqrfsdk/jlibspi
- https://github.com/iqrfsdk/jutils
- https://github.com/iqrfsdk/jlibdpa
- https://github.com/iqrfsdk/jsimply

- Java based libraries
- IQRF SPI, IQRF CDC, IQRF UDP, DPA HDLC UART interfaces
- Examples

- AAEON UP2, Raspberry PI, Linux/Windows platforms
- KON-RASP-01, IQRF-BB-01, GW-USB-06

DEPLOYED:

- TCPCloud.cz
- O2ITS.cz
- Austyn.sk

SDK Introduction

3. Linux/Windows servers

- https://github.com/iqrfsdk/jsimply

- Java based DPA framework
- MQTT channel to the GW
- Examples, Auto-Network, Upload

- Linux/Windows platforms

MQTT channel
IQRF JSON protocol

TODO:

- IQRF JSON protocol

IQRF Gateway

Linux/Windows boards

IQRF Gateways

1. IQRF daemon introduction

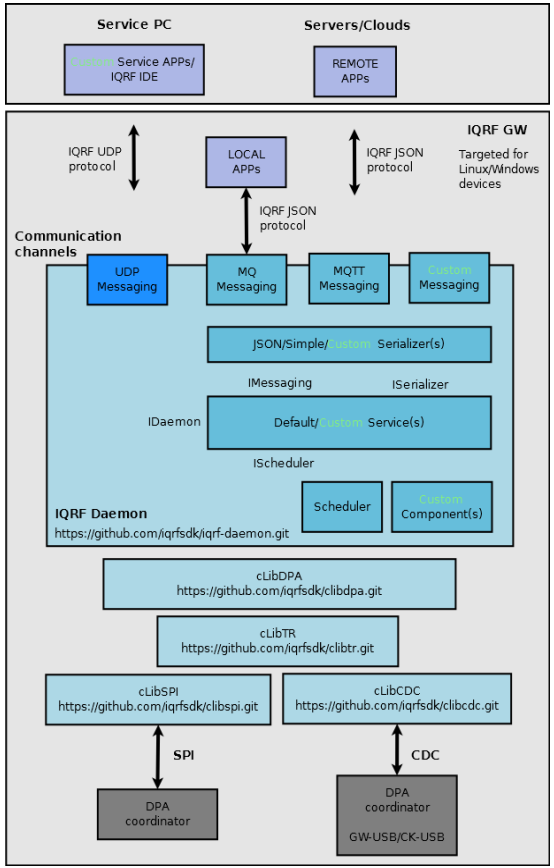
- C++ based solution
- Modular extensible design
- Linux/Windows platforms

- Channel - UDP/MQ/MQTT
- Protocols - IQRF UDP/JSON
- Interfaces - IQRF SPI/CDC
- Components – Scheduler
  Repository lookup

- Custom extensions

To be USED/DEPLOYED:

- UbiWorx.com
- RehiveTech.com
- Eon.cz

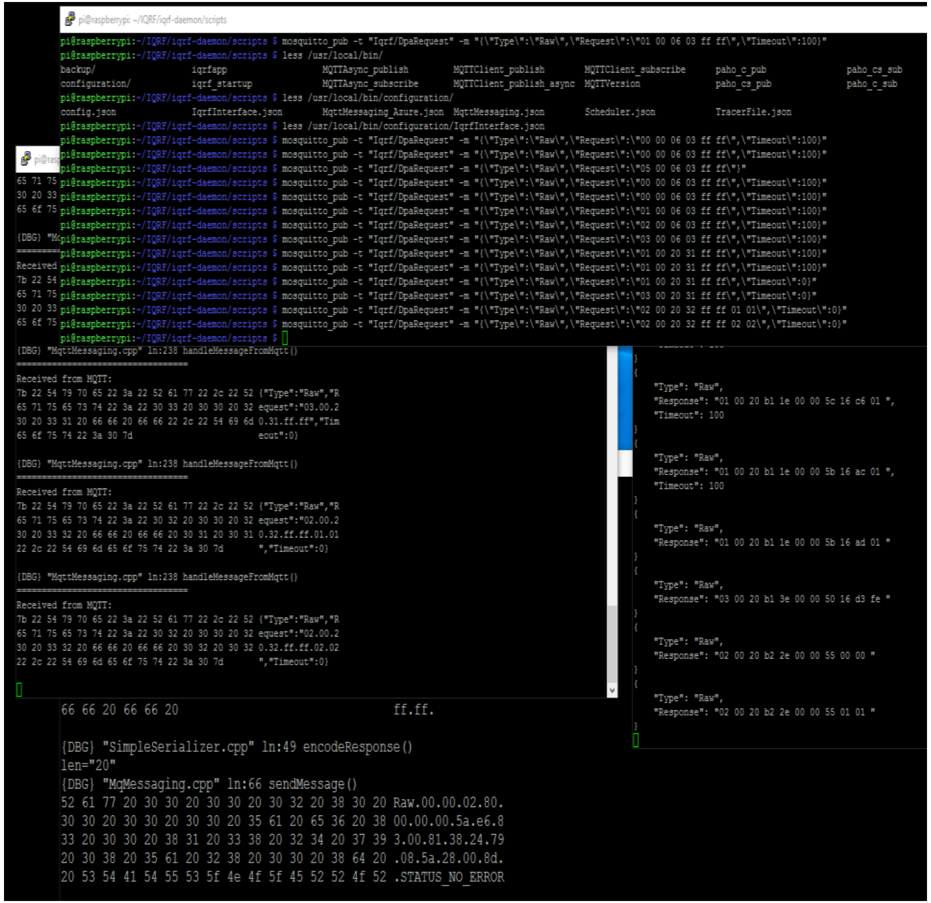


IQRF Gateways

2. IQRF daemon in action

- UDP channel – IQRF IDE
- MQ channel – IQRF APP
- MQTT channel – Mosquitto
  Azure IOT Hub

- SPI interface
- JSON protocol
- Scheduler component



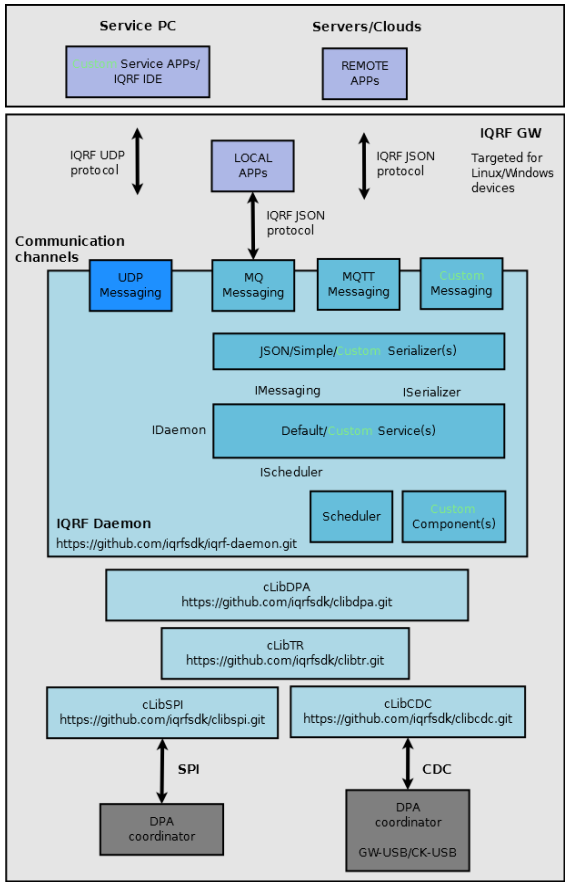
IQRF Gateways

3. Standardization

- Channels - UDP/MQ/MQTT
- Protocols - IQRF UDP/JSON
- Interfaces - IQRF SPI/CDC
- Components - Scheduler
  - Repository lookup

Working group

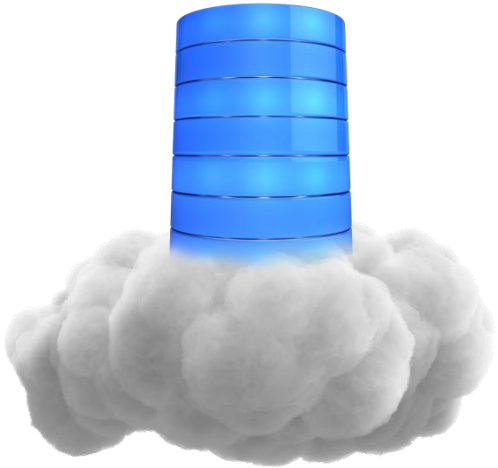
- Regular Skype calls
- Mailing list



IQRF SDK and GWs

Q&A

Means of communication



Storage:
OneDrive

Conf. calls:
Skype for Business

On-line chat:
Slack