Féidearthachtaí as Cuimse
Infinite Possibilities

Week 4
Large-scale architectures

Fundamentals of IoT
Dr. Eoin Rogers (eoin.rogers@tudublin.ie)

OLLSCOIL TEICNEOLAÍOCHTA
BHAILE ÁTHA CLIATH
TU DUBLIN
TECHNOLOGICAL
UNIVERSITY DUBLIN

Lesson Outline

- Large-scale systems and their importance to modern IoT systems
- System architectures common in IoT
- Communication protocols commonly used in IoT systems
- Cloud and fog computing

Large-scale systems

Software systems can grow extremely large

Large scale systems

- Large scale systems are common in modern software development

•Think of Google, Facebook, YouTube and other services

•Although few systems grow this big, it is important to design systems in such a way that they can scale!

Ultra-large scale systems

- Systems on the level of Google or TikTok would qualify as **ultra-large scale systems** (**ULSS**)
- This is used in fields like CS, software engineering and systems engineering to refer to systems with unprecidented:
  - Amounts of hardware
  - Lines of code
  - Numbers of users
  - Volumes of data

Related fields

- **Systems analysis** is concerned with identifying the goals and functionality of processes to create systems that can help achieve them
  - Often, this involves breaking a system into parts that interact to acomplish a purpose (**stepwise refinement**)
- **Software engineering** is the branch of engineering focused on software applications and systems

Complex systems

- A **complex system** is a system composed of many interacting components
- These systems are **intrinsically difficult to model** due to the complexity of the interactions of the components
- They often exhibit **emergent behaviour**

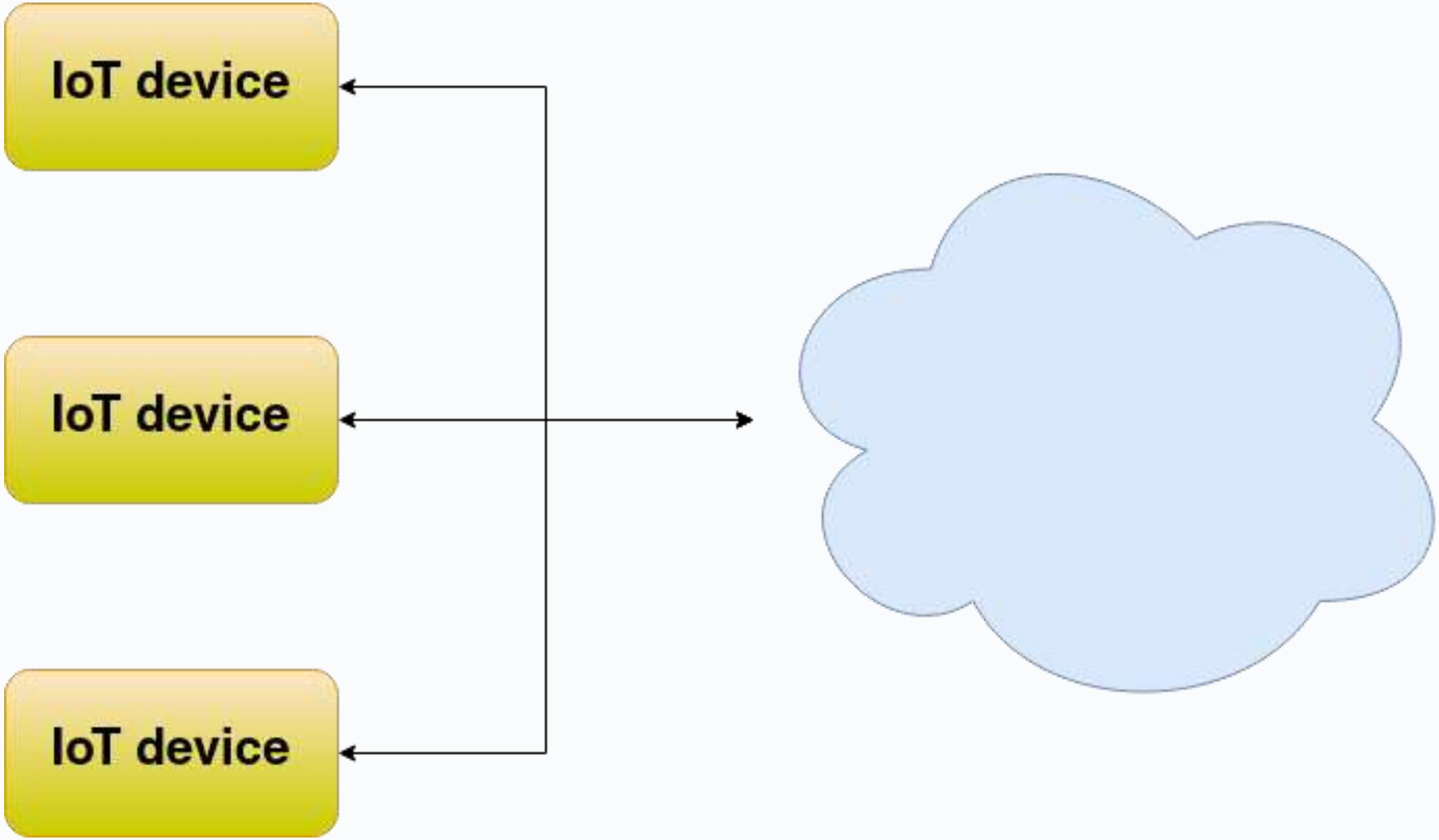
IoT system architectures

IoT systems often qualify as large-scale systems

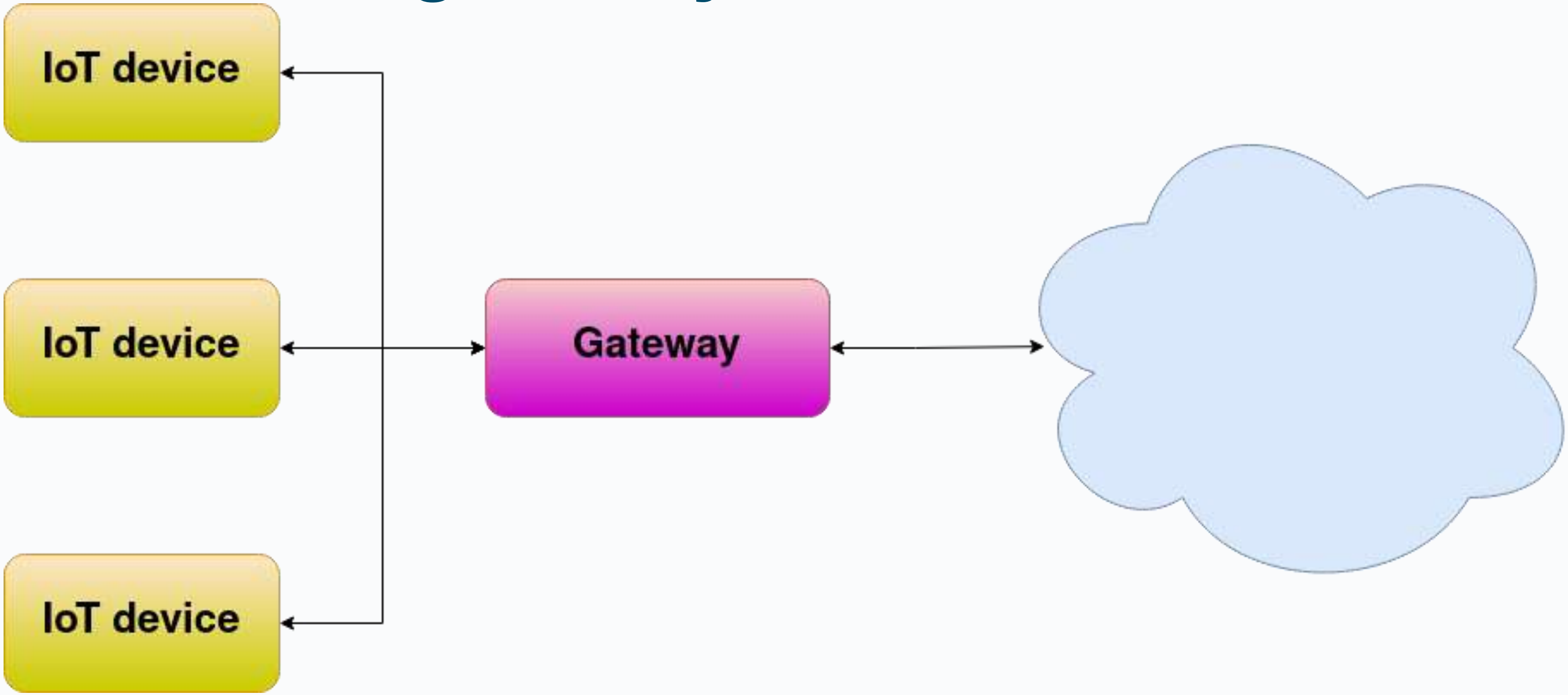
IoT system architectures

- Because IoT devices are often small and low-end, they are often insufficient for any sort of complex code
- We can get around this by **outsourcing computation** to an external system or environment
- Therefore, it is useful to have some knowledge about how to categorise **basic architectural patterns**

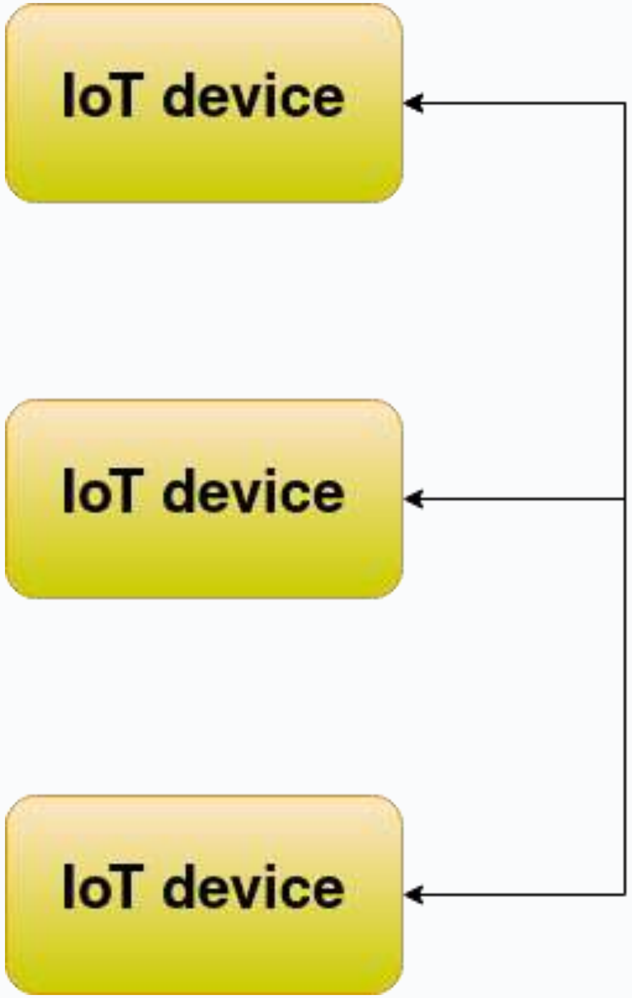
Device-to-cloud



Device-to-gateway



Device-to-device



A **mesh network** (or simply **meshnet**) is a local area network topology in which the infrastructure nodes (i.e. bridges, switches, and other infrastructure devices) connect **directly, dynamically and non-hierarchically** to as many other nodes as possible and cooperate with one another to efficiently route data to and from clients.

Communication protocols

Common ways for IoT devices to talk to the cloud

Communication protocols

- The choice of communication protocol is often important – for example, it can have a major impact on both performance and security
- Developing your own protocol is possible, but it is often frowned upon
  - Can you think of why this is?

Protocols used in IoT

- Two protocols are particularly commonly used in IoT systems:
  - REST/RESTful APIs
  - MQTT

Representational state transfer (REST)

- Commonly used in both webdev and IoT
- A **stateless protocol** making use of the **client-server model**
- Runs on top of existing, common technologies
- Typical example:
  - **JSON** to serialise messages
  - **HTTP**(**S**) to transmit them

REST basics

- You talk to the server by sending **JSON objects inside HTTP**
- The server will return JSON objects in its response
- The protocol is **connectionless/stateless** (you **need to authenticate each time**, the server won't remember you!)
  - The server itself will usually have **internal state**!

Other properties of REST

- The connectionless nature allows for **caching**
- Commonly used with **load-balancing**, since it is a good fit for this
- A **uniform interface** allows components to evolve independently of each other
- Can use a trick called **code-on-demand**, where a server can send JS code to a client in response, so client-side computation can be extended dynamically 😫

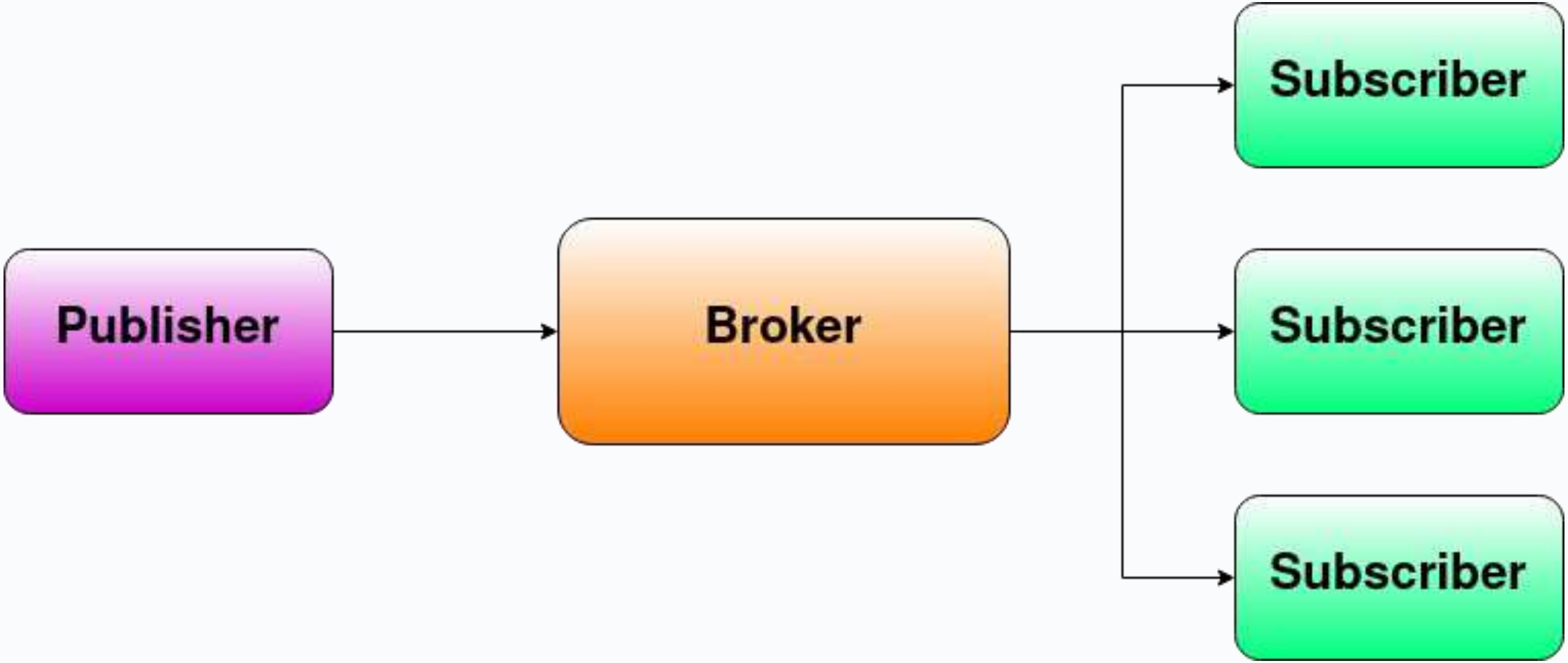
Message queue telemetry transport (MQTT)

- Originally developed for satellite communications (!)
- Now commonly used for IoT
- Works a bit like a **WhatsApp group or mailing list** – allows messages to be sent over a network between a peer machines, facilitated by a central server
  - However, communication is generally one-way – one member of the group sends messages and others just read them!

MQTT terminology

- **Broker** – The **central server** that runs the MQTT service
- **Topic** – A **single MQTT channel**, much like a WhatsApp group or a mailing list
- **Publisher** – A client in a topic that **sends** messages
- **Subscriber** – A client in a topic that **receives** messages

MQTT diagram



Cloud computing

IoT often (but not always!) utilises cloud computing

Cloud computing

- With cloud computing, most processing occurs on a **centralised server**
- This could be **on-prem**, or could be running on a **third-party cloud service**
  - What might be some advantages and disadvantages of each approach?

Fog computing

- **Edge computing** is a reaction to cloud computing, where computation is carried out locally on **edge devices** as much as possible
- **Fog computing** is the use of architectures that make extensive use of edge computing
- In fog computing, services are hosted close to where they are used: at the **network edge** or even at **end devices**!

Issues with cloud computing

- Security
- Privacy
- GDPR
- Vendor lock-in

Issues with fog computing

- Higher up-front device cost
- Higher power consumption (for users)
- Higher device complexity (need to deal with firmware updates etc.)
- More e-waste

Summary

- Gave an overview of large-scale software systems, and why they are difficult to work with/on
- Covered three types of architectural pattern common in IoT (device-to-cloud, -gateway and -device)
- Covered both REST and MQTT as exemplar communication protocols
- Talked about cloud vs. fog computing and advantages of each

That’s all for this week

Thanks for your attention!