Building IoT Data Pipelines With Python

Logan Wendholt

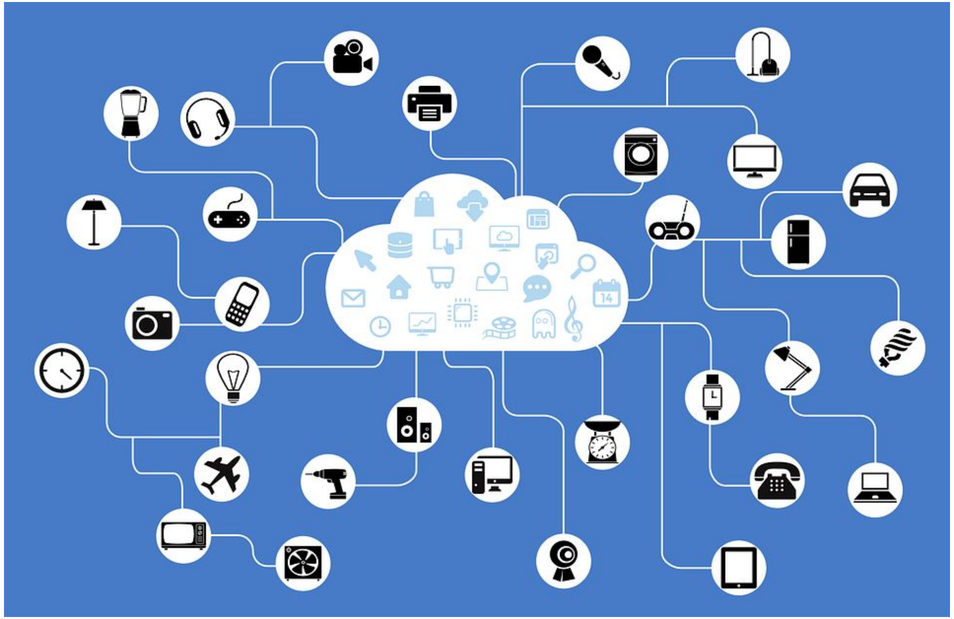
IoT Technical Lead, Bastian Solutions

Overview

- IoT in 10 seconds
- Pipeline Basics
- Common Hardware Deployments
- Pipeline Design Patterns and Examples
- Cloud-based Pipelines / “Serverless”
- Hybrid Architectures

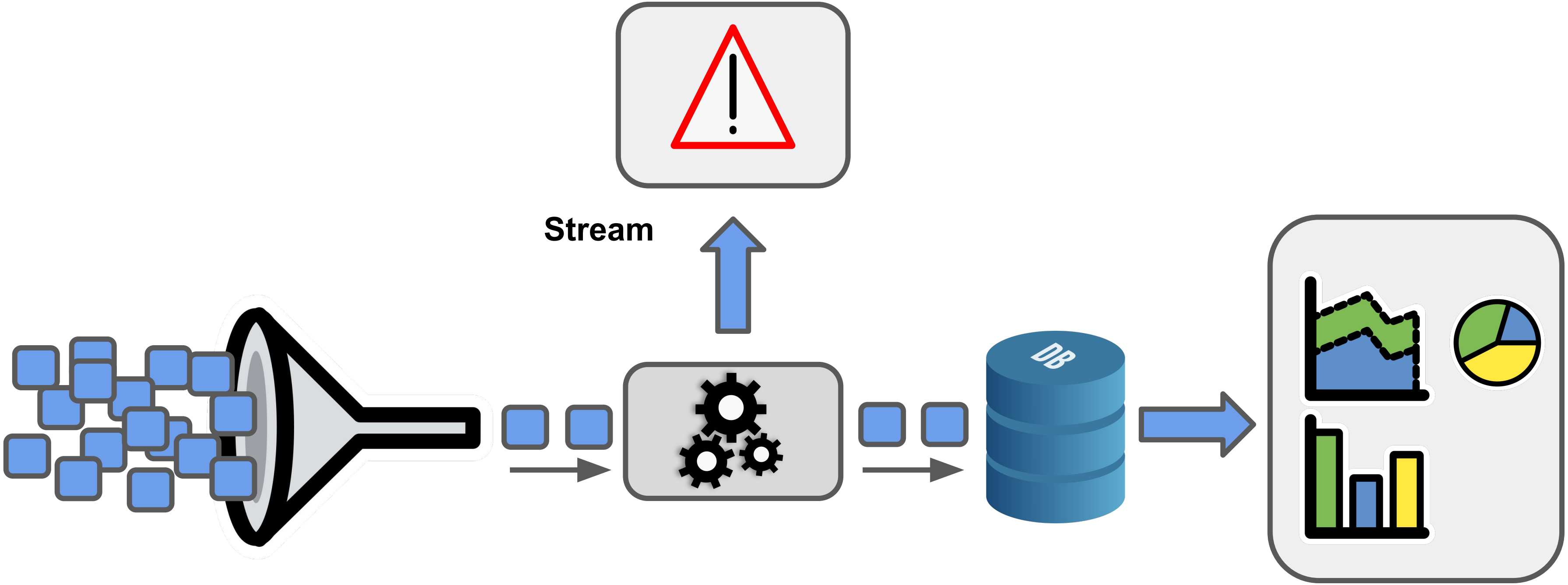
Internet of Things

- Connect things… to the internet
  - Or maybe just your private network



- Why?
  - **DATA**

What’s in a Pipeline?



Stream

Collect

Transform

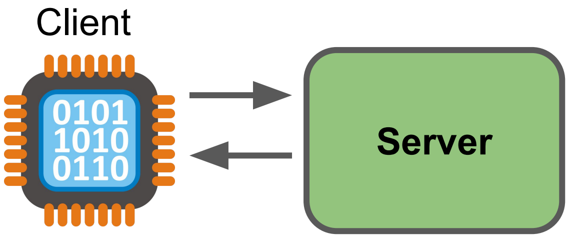
Store

Analyze

Things to Consider

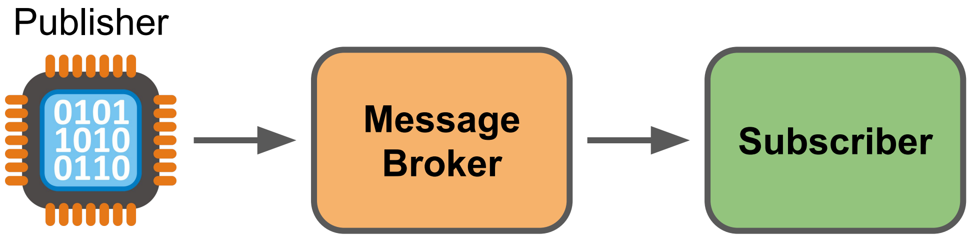
- What kind of data?
- How much?
- How fast?
- What type of “flow”?

- Hardware limitations?



Client

Server



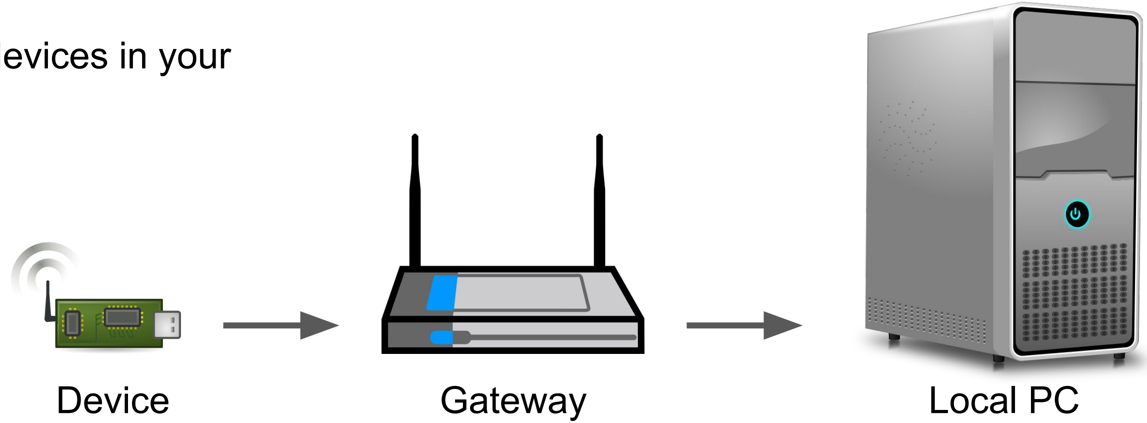
Publisher

Message Broker

Subscriber

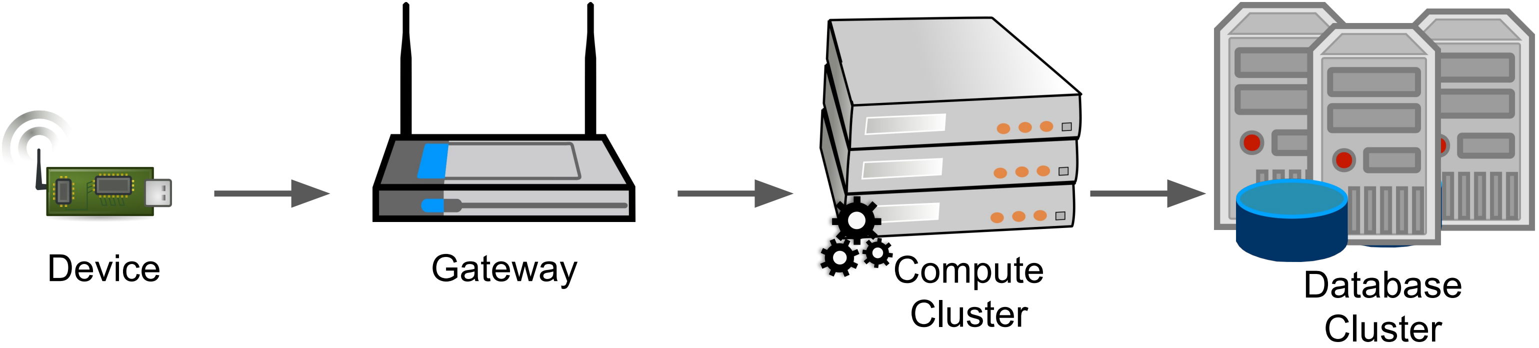
Things to Consider

A handful of devices in your apartment?



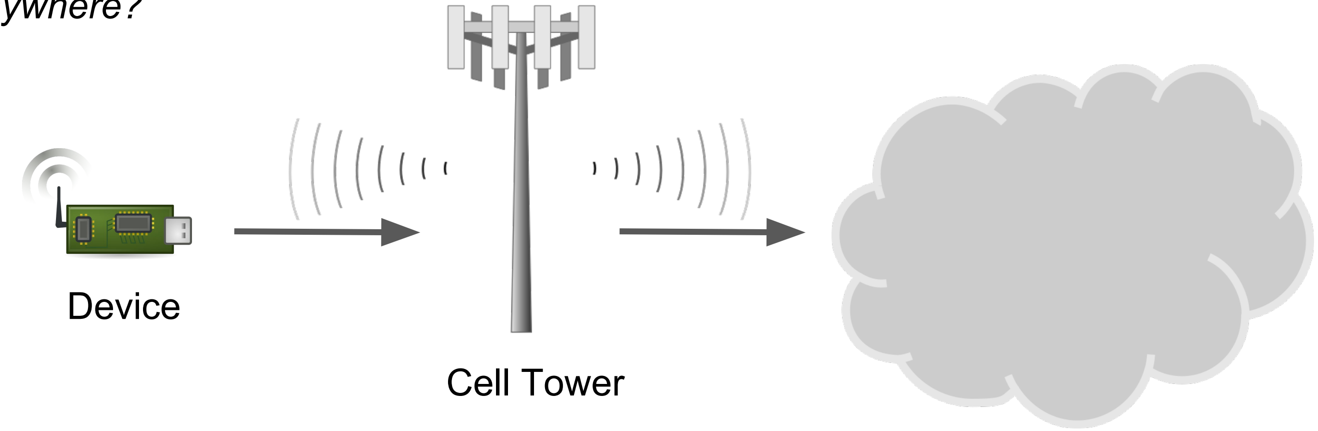
Things to Consider

Hundreds of sensors in your factory?



Things to Consider

Thousands of sensors deployed *anywhere?*



Things to Consider

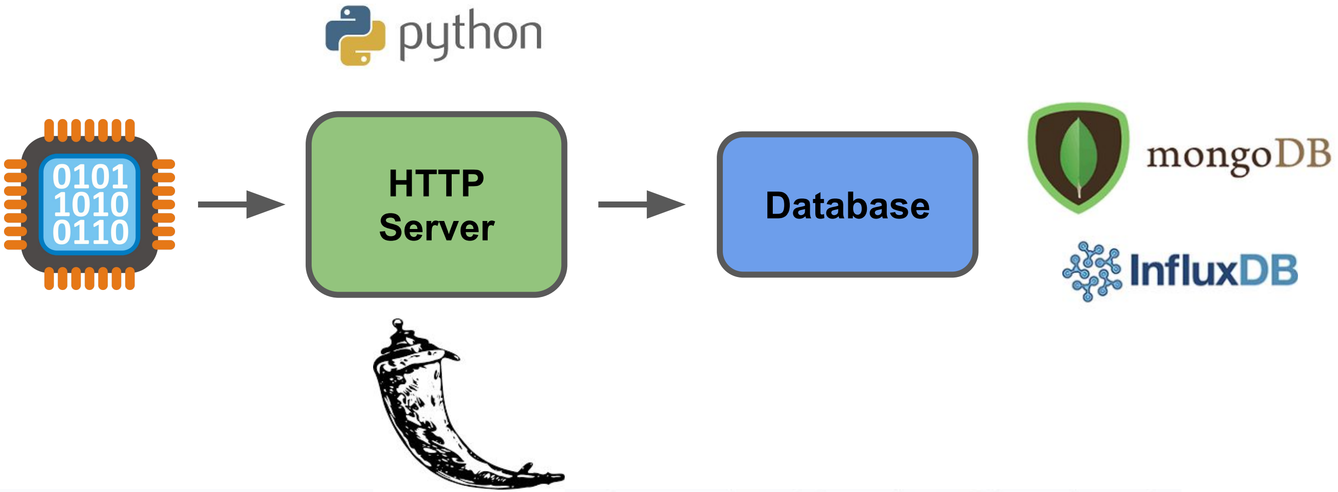
How do I choose?

Client-Server Architecture

- May want to use a client-server pattern if...
    - Devices natively use HTTP or other client-server protocol
    - Data flow pattern is better suited for request-response format
        - Receive input -> send to cloud -> receive response -> perform action
    - Need to integrate with existing architecture
        - Can’t spin up a new server, and access to database is restricted to a REST API?

Client-Server Architecture

If you're using HTTP:



pallets / flask

Watch 2,222 Star 39,290 Fork 11,406

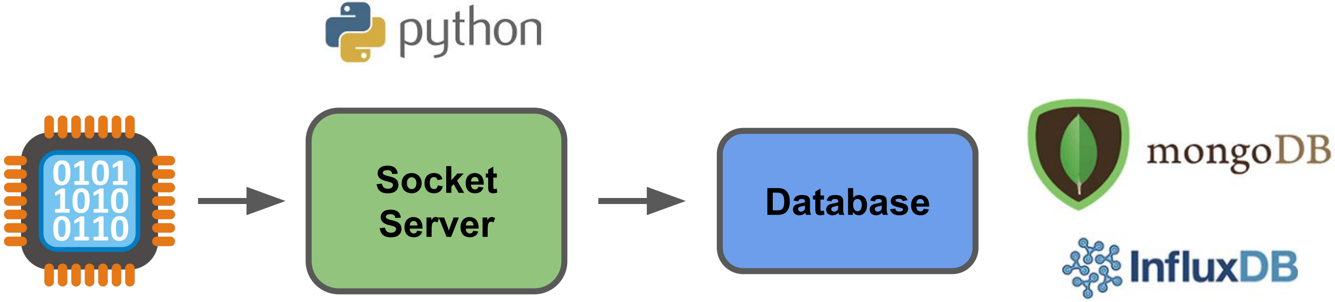
<> Code Issues 37 Pull requests 15 Insights

The Python micro framework for building web applications. https://www.palletsprojects.com/p/flask/

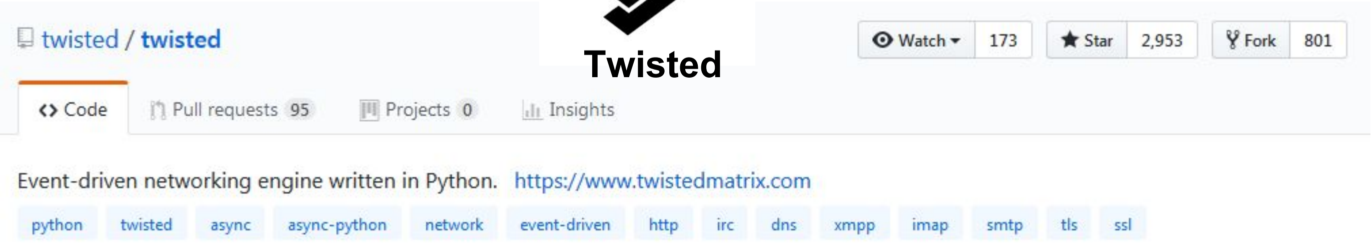
python flask wsgi web-framework werkzeug jinja

Client-Server Architecture

If not:



Twisted


twisted / twisted

Event-driven networking engine written in Python. https://www.twistedmatrix.com

Client-Server Architecture

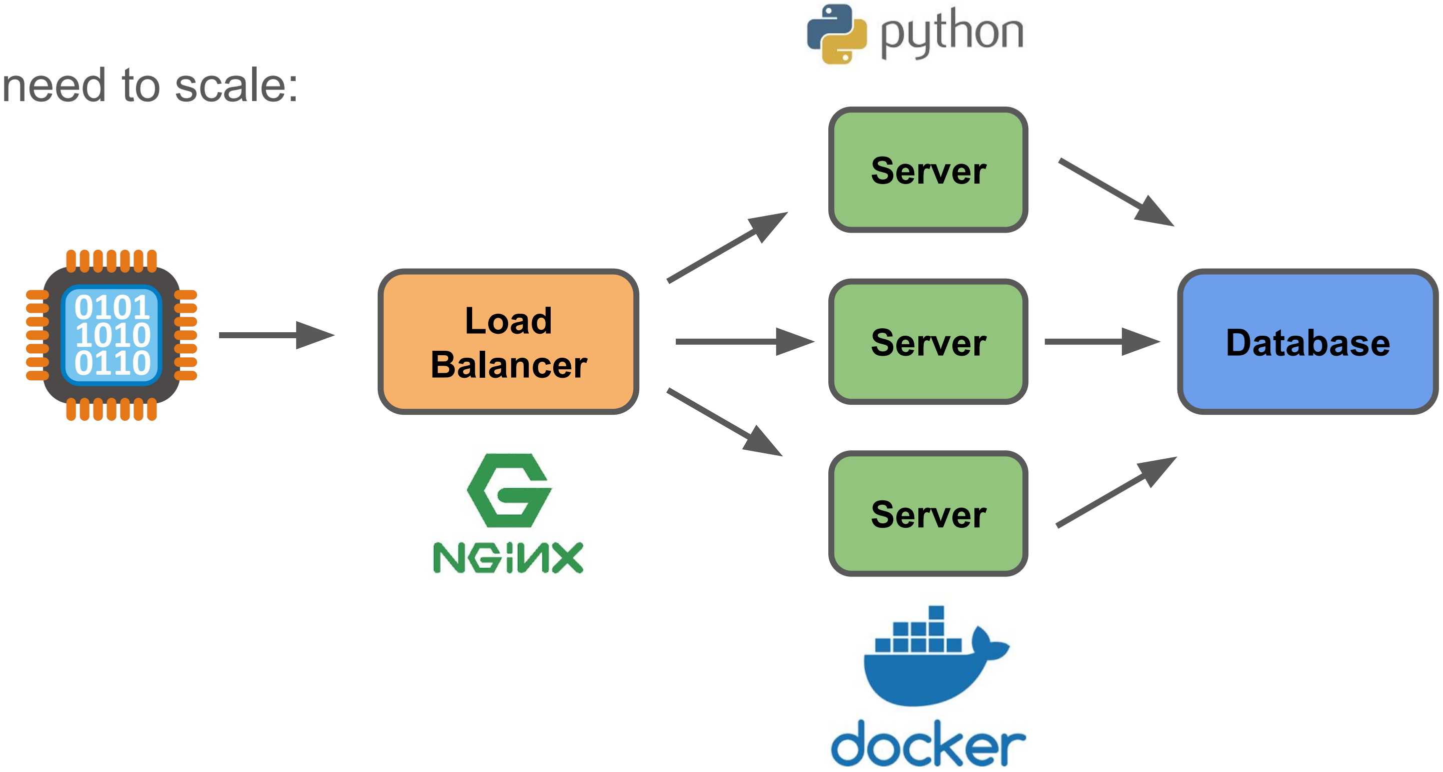
- Can everything run on a single PC? ✅

Local PC

But… what if you need more resources?

Client-Server Architecture

If you need to scale:



Client-Server Architecture

- Pros:
  - Familiar concepts and tools (web dev)
  - Can support HTTP via Flask or custom protocols via Twisted

- Cons:
  - Custom protocols can be problematic
  - HTTP can be inefficient for low-resource IoT devices
  - Need additional code to support real-time data streaming

Need a different approach?

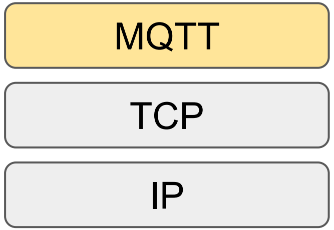
Publish-Subscribe Architecture

- May want to consider pub/sub if…
    - Sending data asynchronously
    - Small data packets (sensor data)
    - Power or bandwidth are a concern

- De facto IoT pub/sub standard: MQTT

What’s MQTT?

- Message Queuing Telemetry Transport
  - Publish / subscribe messaging protocol
  - Application layer of network stack -- replaces HTTP



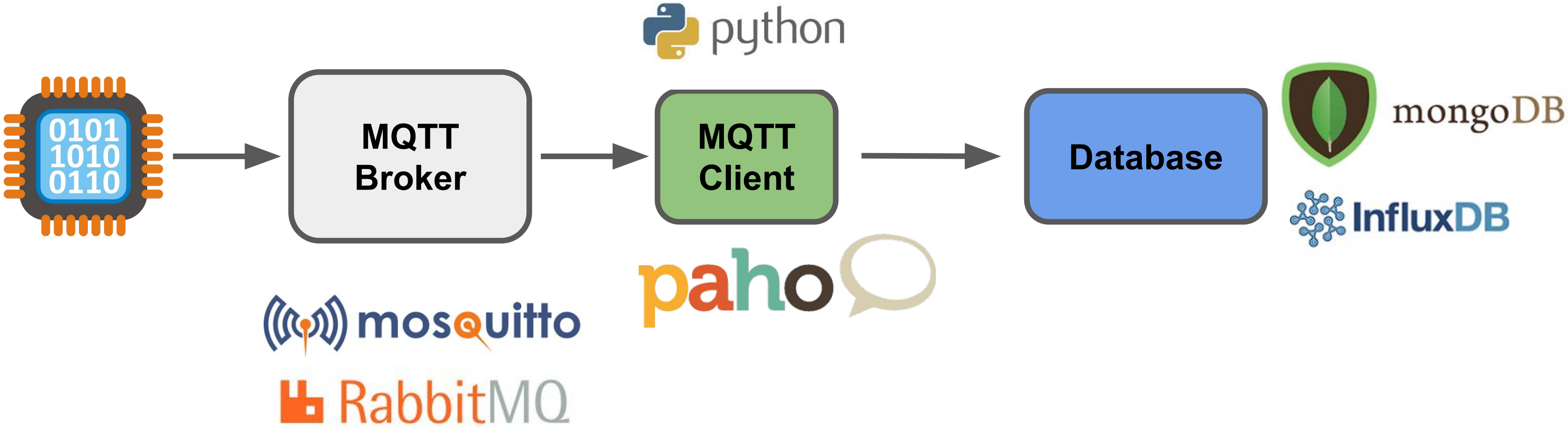
MQTT

TCP

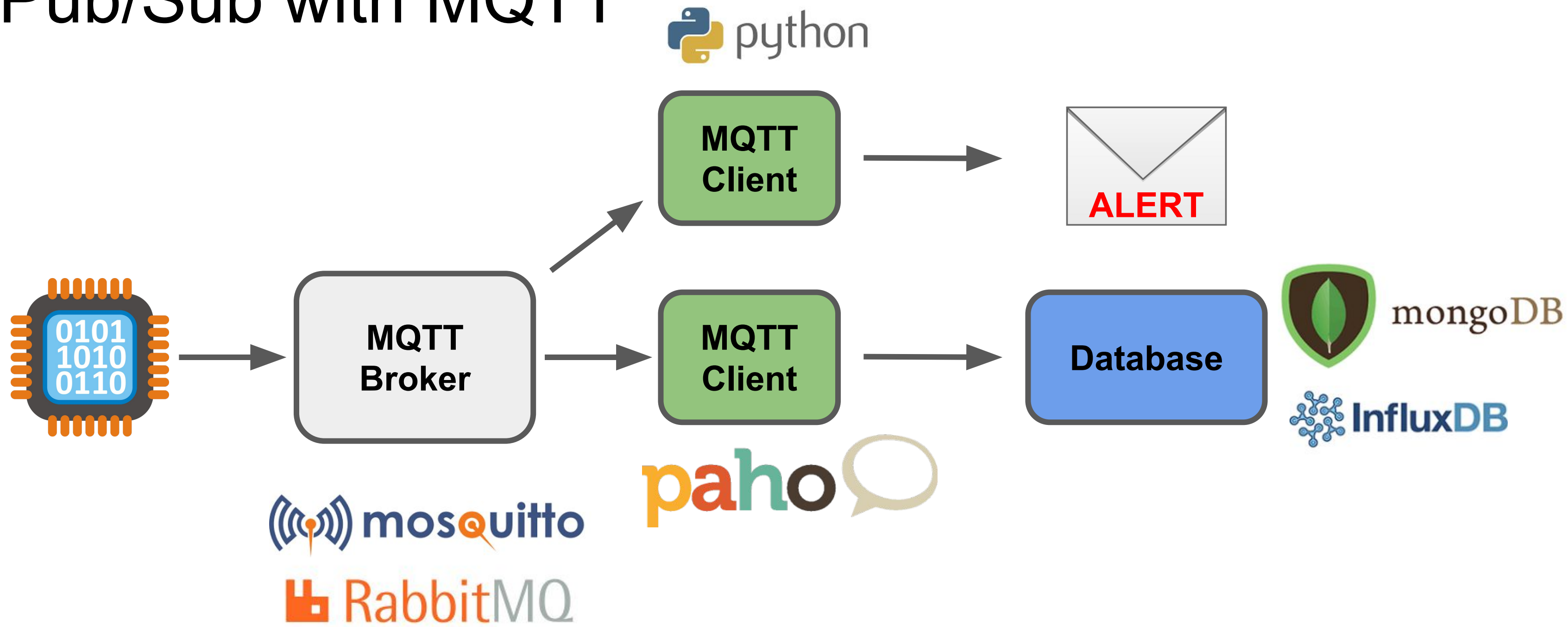
IP

- Popular option for IoT devices
  - Very little data overhead
  - Well-suited to transmitting sensor data
    - Wake up
    - Publish
    - Go back to sleep

Pub/Sub with MQTT



Pub/Sub with MQTT



MQTT Broker

MQTT Client

MQTT Client

ALERT

Database

Pub/Sub with MQTT

- Pros:
  - Pub/Sub makes it easy to stream data to multiple endpoints
    - Alerts system?
    - Real-time visualizations?
  - Decouple data processing from message routing
    - Can swap out message broker if necessary
- Cons:
  - Devices need to speak MQTT!
    - If you have to translate, it’s probably not worth it
  - Typically limited to one client working as “database adaptor”
    - Can lead to bottlenecks in high-throughput scenarios

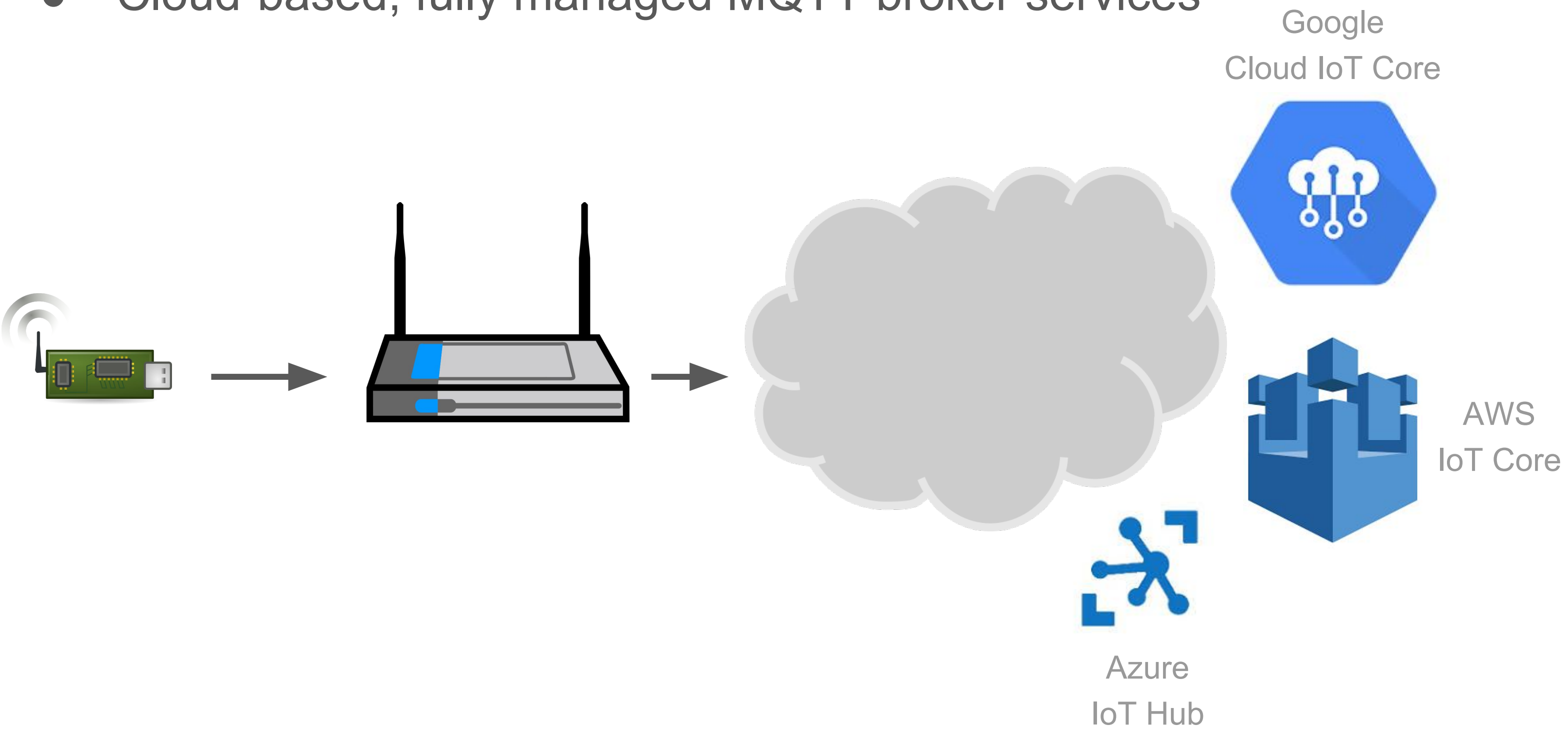
Pub/Sub with MQTT

- How to solve bottleneck / scaling issues?

Use the cloud!

IoT + Cloud Pipeline

- Cloud-based, fully managed MQTT broker services



AWS Lambda

Google Cloud Functions

Azure Functions

IoT + Cloud Pipeline

- Function-as-a-Service (FaaS)
  - Self-contained functions (Python!)
  - Trigger based on events, inputs, etc.
  - Use cases:
    - ETL
    - Alerts
    - Analytics
    - Storage

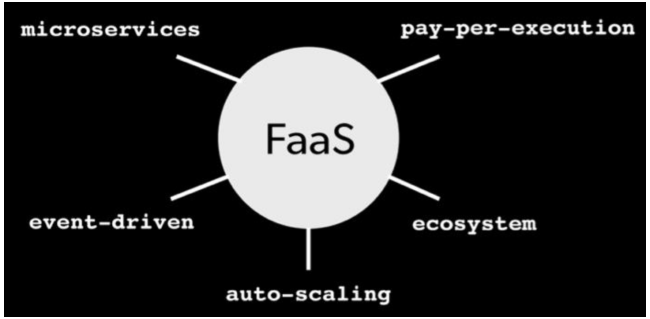


Image Credit: Austen Collins - ServerlessConf Austin ‘17

  - Cloud provider manages deployment, scaling, maintenance -- you just provide code!

IoT + Cloud Pipeline

- Managed database solutions
  - Have the cloud provider run the database for you
  - Only charged for the storage and requests
  - Scale up or down fast!

Google Big Query

DynamoDB

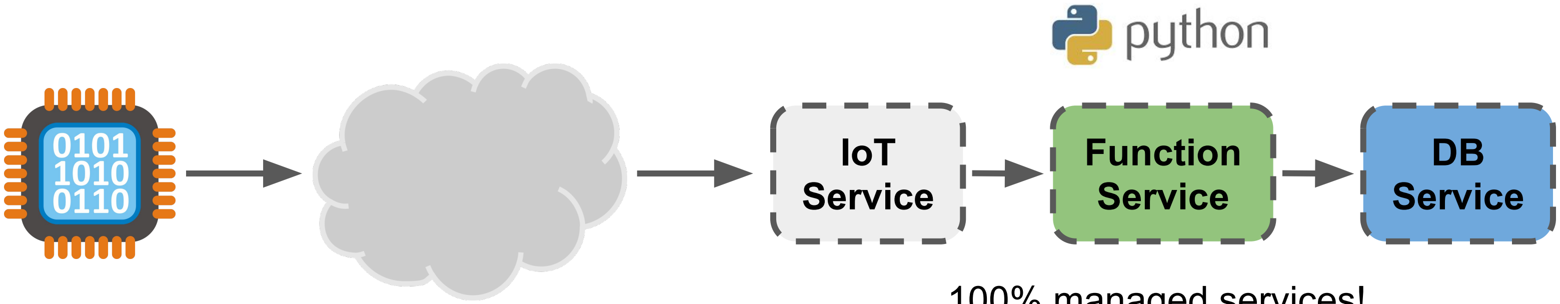
amazon web services™

Azure Cosmos DB

Serverless Architecture

Managed MQTT broker + FaaS + Managed Database = **Serverless**

- Exclusively use managed services rather than setting up VMs or servers
  - Let the cloud providers handle maintenance, installation, security updates, etc.
  - Scale up or down almost instantly
  - Only pay for what you use (requests, compute time, storage)
  - Focus development efforts where it counts: on the code!



100% managed services!

Serverless Architecture

Managed MQTT broker + FaaS + Managed Database = **Serverless**

- Infrastructure-as-Code
  - Create code templates that describe the resources you need
    - Databases, compute functions, other services
  - “Serverless Framework”
    - Platform-agnostic infrastructure as code
    - Easily deploy and scale pipelines
    - Less cloud provider lock-in

Serverless Opportunities

- Easy to hook serverless pipeline into additional managed services
    - AWS IoT Analytics, Kinesis, Redshift
    - Google ML Engine, Dataflow, Bigquery
- Powerful analytics, machine learning, data warehousing options
- Beyond the scope of this talk

Go explore!

Serverless Costs

Scenario:

- Want to monitor temperature changes throughout a large warehouse
  - Place temperature sensors near all the doors
  - Send out a push alert to smartphone when temperature increases rapidly

- Want to use cloud solution for all processing and storage
  - How about AWS?
    - AWS IoT Core, Lambda, DynamoDB, Simple Notification Service

Serverless Costs

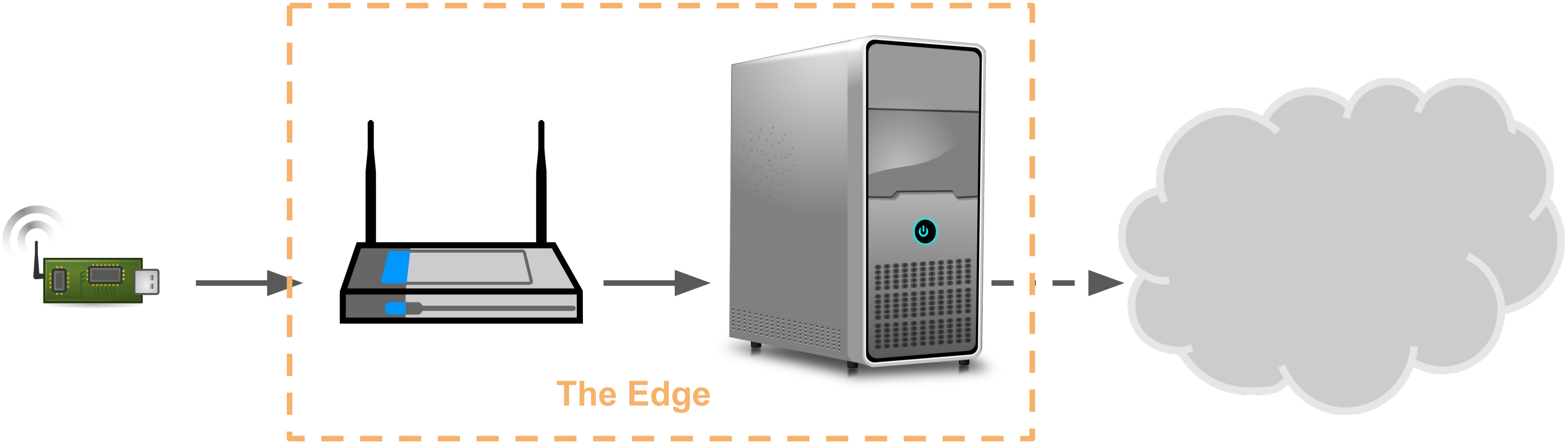
Straight to the cloud:

- 20 devices, report every 5 minutes: FREE
- 20 devices, report once per minute: $1
- 20 devices, report every 30 seconds: $3
- 20 devices, report once per second: $110

Lots of requests -> lots of $$$

Edge Processing

- Process data locally -> only talk to cloud when necessary
  - Also works with unreliable network connection scenarios
  - AWS Greengrass, Google Cloud IoT Edge, Azure IoT Edge



Back to the Scenario...

- Use AWS Greengrass to process data at edge
  - Local Lambda functions running on a Raspberry Pi -- still using Python!

RaspberryPi

AWS Greengrass

python

- If temperature anomaly is detected, send request to cloud to send an alert
  - Store all temp data locally -- no need to stream to cloud
  - Maybe periodically send a backup of historical data?

Serverless Costs

Straight to the cloud:

- 20 devices, report every 5 minutes: FREE
- 20 devices, report once per minute: $1
- 20 devices, report every 30 seconds: $3
- 20 devices, report once per second: $110

Serverless Costs

~~Straight to the cloud:~~ Edge + Cloud

- 20 devices, report every 5 minutes: FREE
- 20 devices, report once per minute: ~~$1~~ FREE*
- 20 devices, report every 30 seconds: ~~$3~~ FREE*
- 20 devices, report once per second: ~~$110~~ FREE*

*Perform bulk of compute at edge and remain within free tier on requests

Edge Processing Recap

- Dramatically reduce costs by doing pre-processing locally
- Only send data to cloud when necessary
- Great compromise
  - flexibility of the cloud
  - low operational costs of running on-prem
- Can still use Python functions as the primary method of computation

To wrap it up...

- Wide range of possibilities for building data pipelines in Python
    - Flask / Twisted for client-server architectures
    - Paho MQTT client + RabbitMQ or Mosquitto for pub/sub architectures
    - Managed IoT services from Google, Microsoft, Amazon for cloud-based pub/sub
- Keep the cloud in mind
    - Serverless architectures can be very powerful!
    - Be careful with calculating your costs
    - Cloud-based pipeline with edge pre-processing: best of both worlds?
- Use the best tools for the job
    - Consider your use case, existing resources, and experience before deciding on an approach
    - Don’t get cut by the cutting-edge!

Thanks!