The Cloud

Overview

- What is the Cloud?
- Cloud options

What is the Cloud?

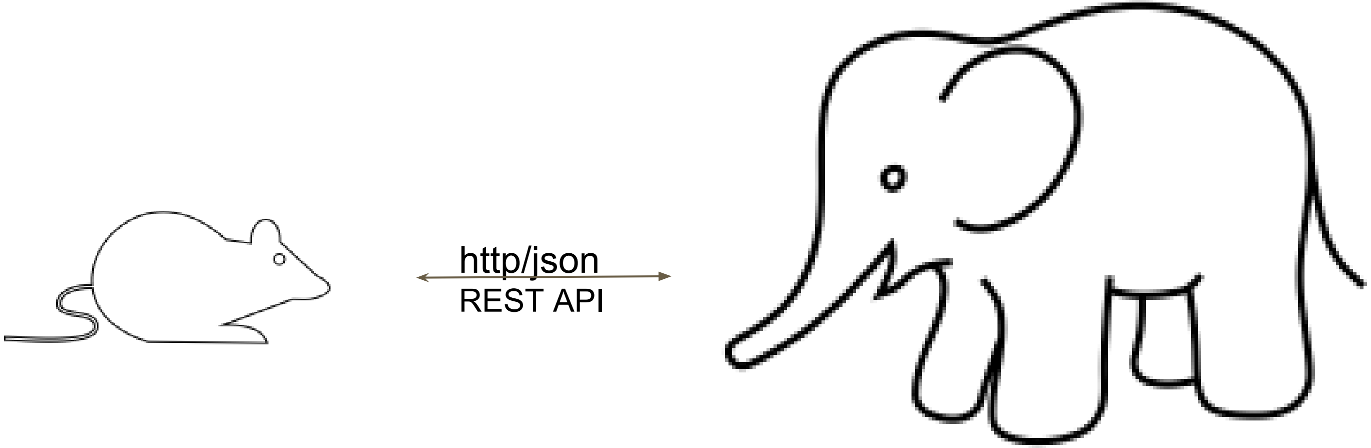
- Software runs on the internet instead of running on your computer
  - Google Docs
  - Web Mail

- Custom software can be created without worrying about resources
  - On Premise

    - 70% Time to manage hardware & 30% Develop Business logic
  - Cloud Infrastructure (Flips this)

    - 30% Time to manage hardware & 70% Develop Business logic

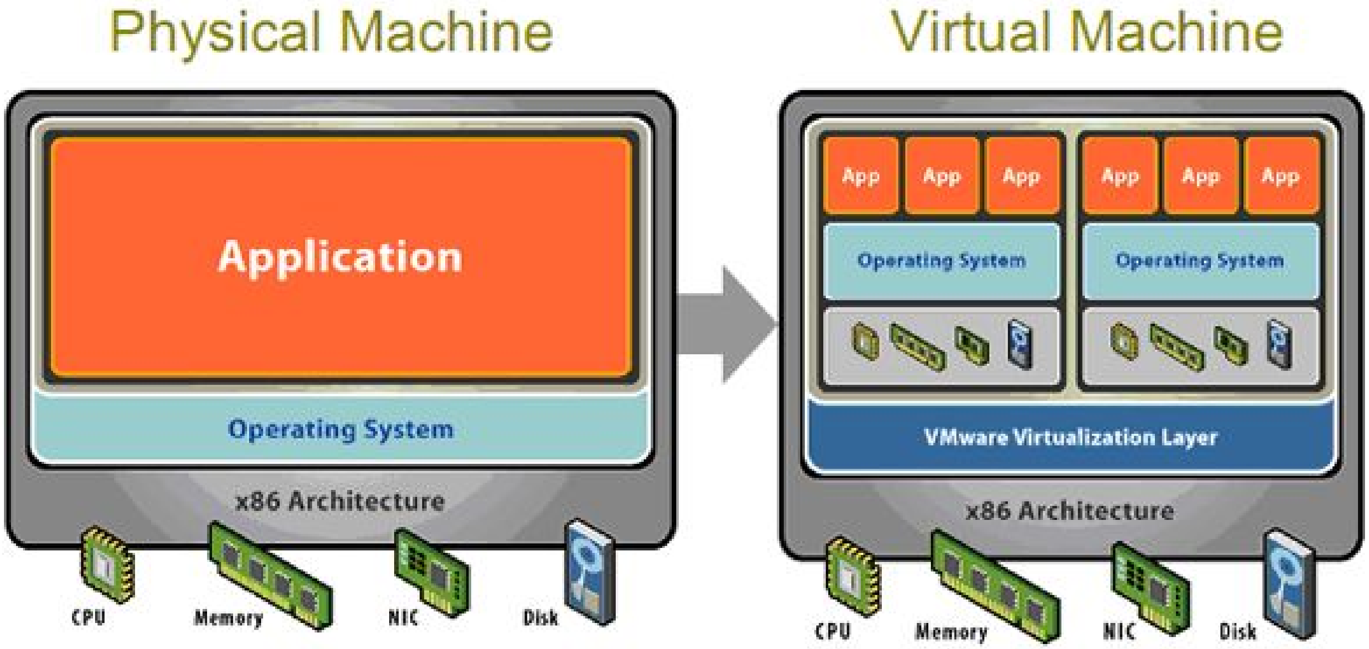
Embedded vs Cloud



http/json
REST API

Virtual Machine

Emulation of a computer



Flavors of Cloud Offerings

- IAAS - Infrastructure As A Service
- PAAS - Platform As A Service
- SAAS - Software As A Service

SAAS - Software as a Service

Microsoft Office Installed on your PC (On Premise)

Vs

Google Docs on the Cloud (SAAS)

IAAS - Infrastructure as a Service

- Virtual machine , storage and networking capabilities are hosted by a service provider and offered to customers on-demand

- Customers can provision the hardware via Web or SSH

PAAS - Platform as a Service

- Lies between SAAS & IAAS Models

- Hardware + OS + Middleware + DB services are provided

Flavors of Cloud Offerings

- IAAS - Infrastructure As A Service
  - Rent Kitchen Equipment like stove & oven
  - You buy Ingredients & Cook

- PAAS - Platform As A Service
  - Rent Kitchen Equipment & Ingredients
  - You cook

- SAAS - Software As A Service
  - You just eat out

Advantages

- Pay based on usage
- No capital costs
- Elastic scaling, On demand
- Focus on business logic and not infrastructure

Google vs Microsoft vs Amazon

- Google Cloud Platform
    - Slightly more capabilities on Big Data and Machine Learning
    - Cheaper (minute based!)
- Microsoft Azure
    - Focused on Microsoft products
- Amazon AWS
    - API's are most evolved
    - Has the most services. No SAAS
    - Documentation and community is most mature

We picked Amazon AWS

- Free Tier Available in most countries
- Offers an iOT SDK
- You are free to choose your own platform

Amazon Snowmobile - ExaByte

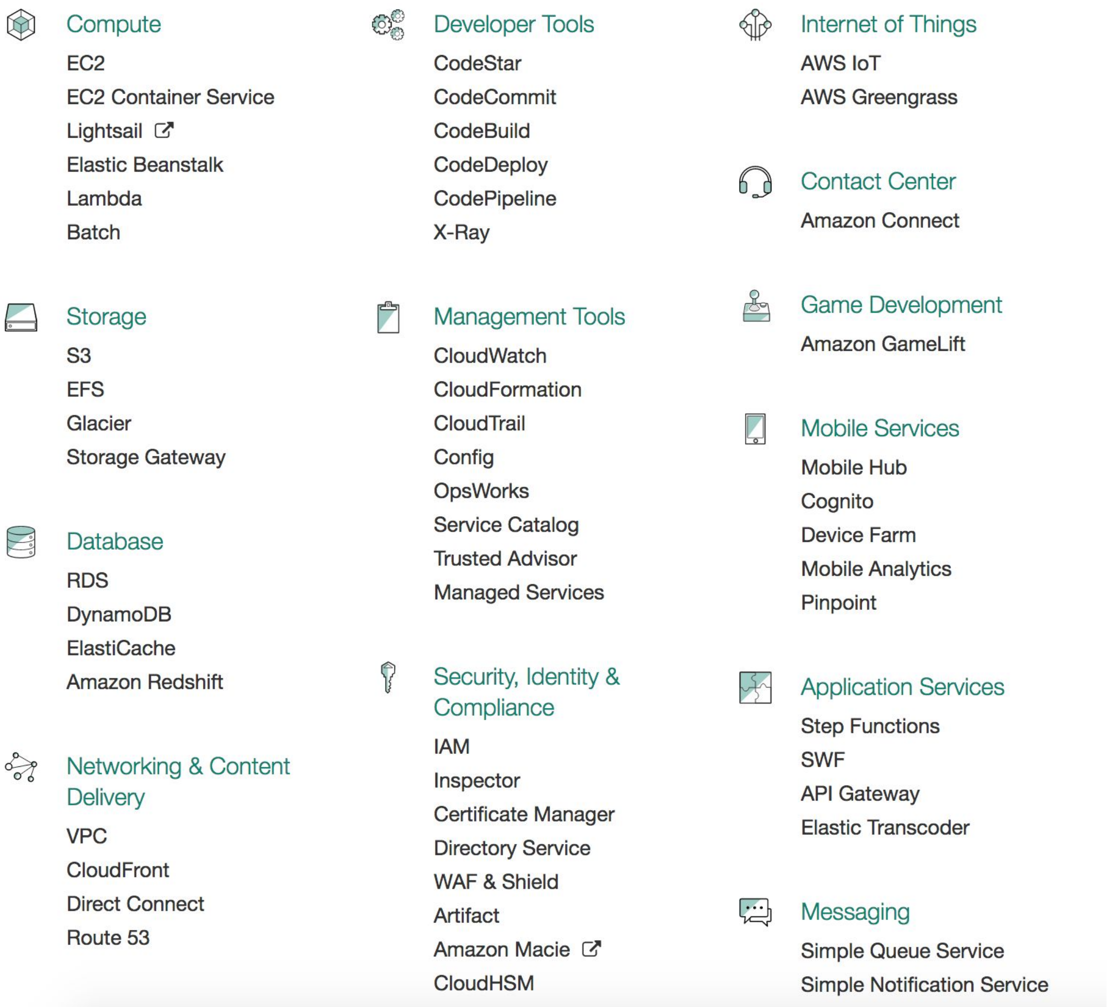
Google Cloud Vision

- https://cloud.google.com/vision/

Summary

- No reason to run your own servers anymore
- Learn to leverage the power of the cloud services
- Hands on projects in this course based using AWS

AWS Services



Compute

- EC2
- EC2 Container Service
- Lightsail ↗
- Elastic Beanstalk
- Lambda
- Batch

Storage

- S3
- EFS
- Glacier
- Storage Gateway

Database

- RDS
- DynamoDB
- ElastiCache
- Amazon Redshift

Networking & Content Delivery

- VPC
- CloudFront
- Direct Connect
- Route 53

Developer Tools

- CodeStar
- CodeCommit
- CodeBuild
- CodeDeploy
- CodePipeline
- X-Ray

Management Tools

- CloudWatch
- CloudFormation
- CloudTrail
- Config
- OpsWorks
- Service Catalog
- Trusted Advisor
- Managed Services

Security, Identity & Compliance

- IAM
- Inspector
- Certificate Manager
- Directory Service
- WAF & Shield
- Artifact
- Amazon Macie ↗
- CloudHSM

Internet of Things

- AWS IoT
- AWS Greengrass

Contact Center

- Amazon Connect

Game Development

- Amazon GameLift

Mobile Services

- Mobile Hub
- Cognito
- Device Farm
- Mobile Analytics
- Pinpoint

Application Services

- Step Functions
- SWF
- API Gateway
- Elastic Transcoder

Messaging

- Simple Queue Service
- Simple Notification Service

Important Services

EC2

Elastic Cloud Compute

S3

Simple Storage Service

RDS

Relational Data Service

Compute Resources

EC2	Virtual Machine
ELB	Load Balancer
Lambda	Modular Service
EC2 Container Service	Docker

Databases

RDS	Relational Database Management System (RDBMS)
DynamoDB	NoSQL database
ElastiCache	Memcached
S3	FTP Server

DevOps

CodeCommit	Git
CodeDeploy	Jenkins
CodePipeline	Continuous Integration
OpWorks	Orchestration (Chef)

Big Data

Elastic MapReduce	MapReduce
Data Pipeline	Spark
Kinesis	Kafka

Takeaway or Resources

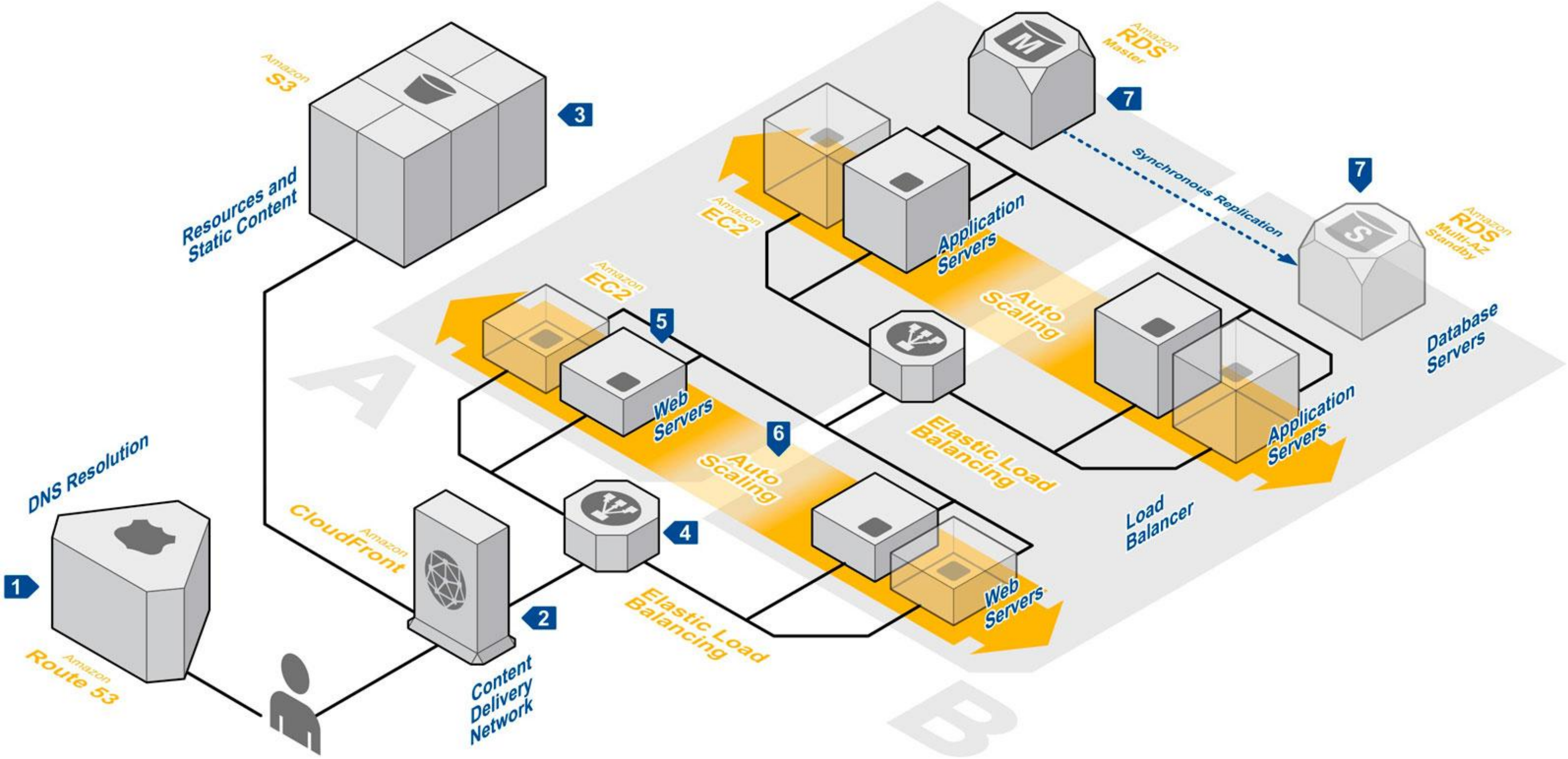
- Aws has a multitude of services that anyone can take advantage of
- https://www.expeditedssl.com/aws-in-plain-english

Systems Architecture

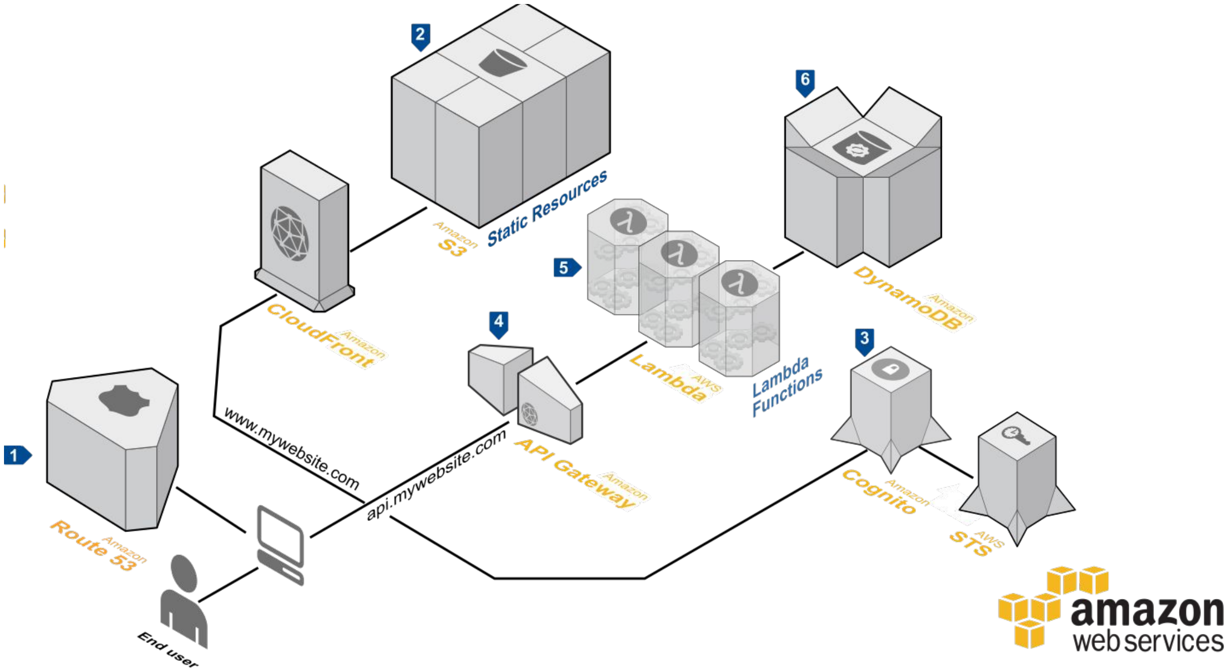
Overview

- Basic Architecture overview
- IOT-based Architecture

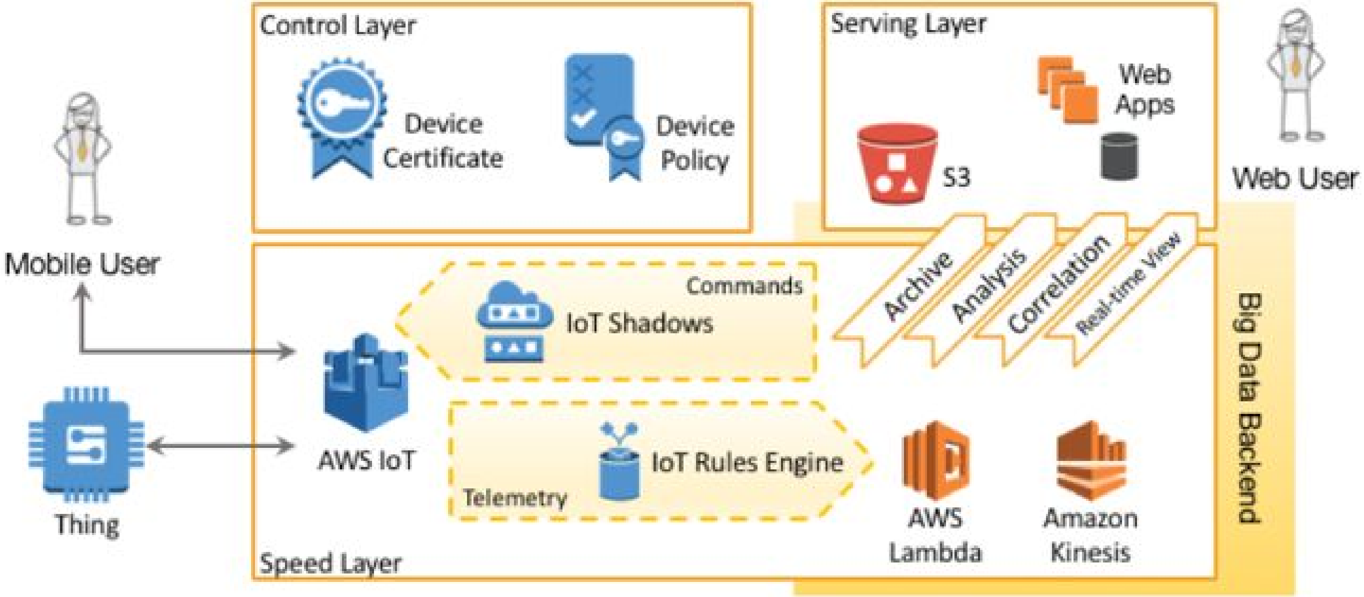
Web Architecture



Web Architecture



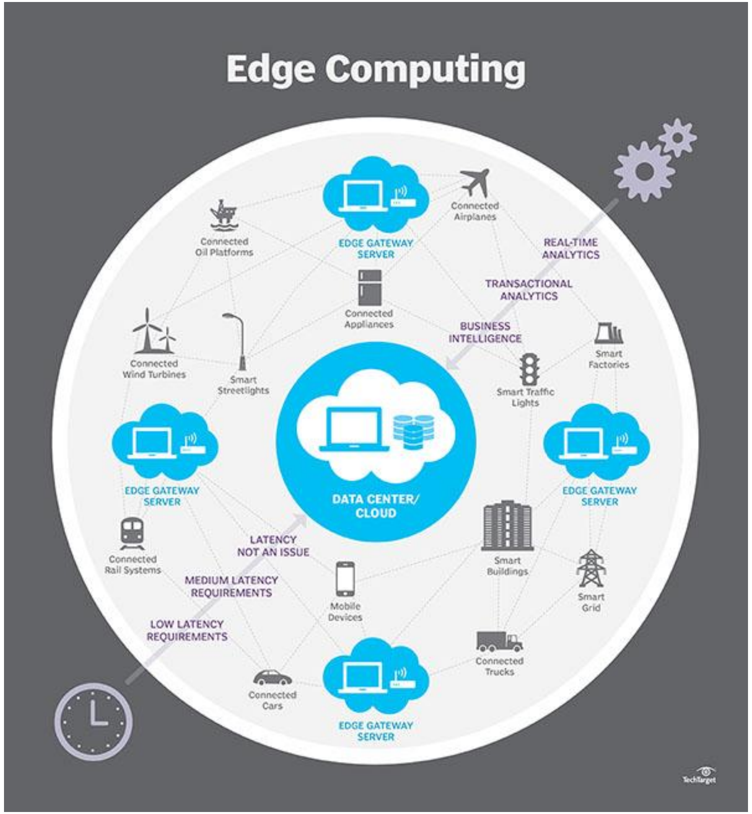
IoT



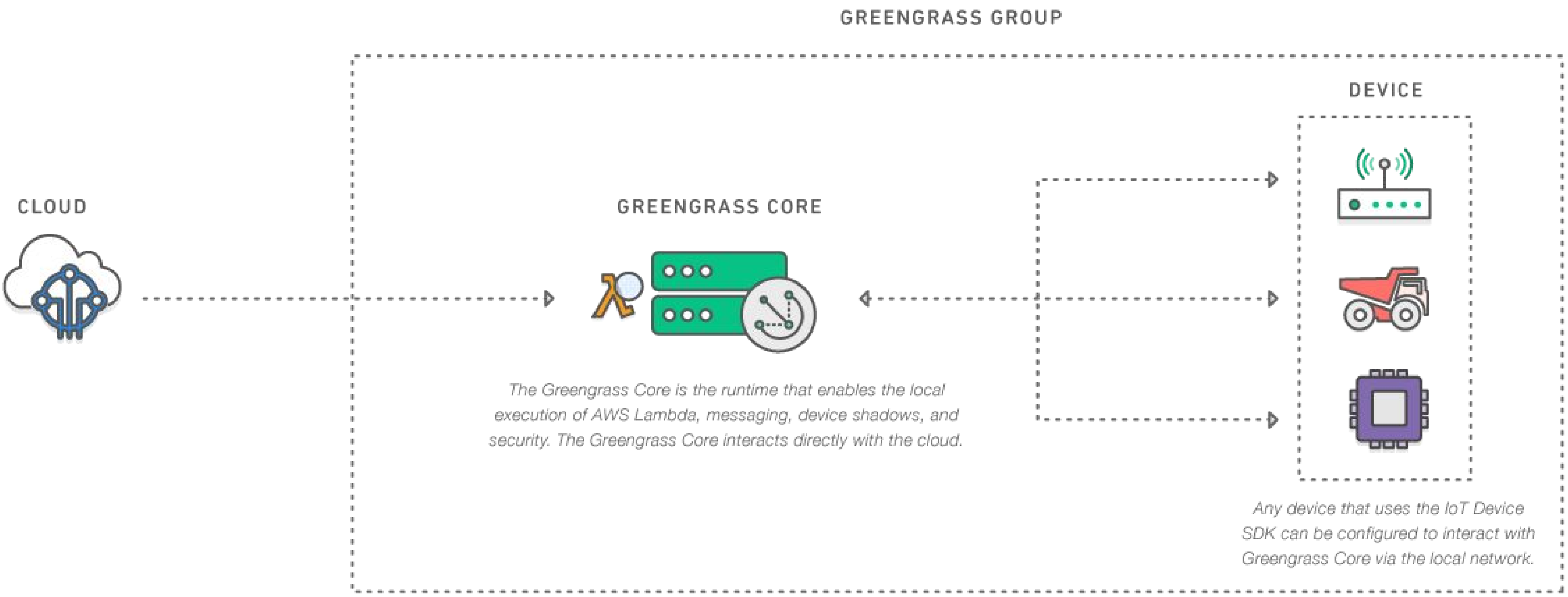
Control Layer
Device Certificate
Device Policy
Serving Layer
Web Apps
S3
Web User
Mobile User
Commands
IoT Shadows
Archive
Analysis
Correlation
Real-time View
AWS IoT
Thing
Telemetry
IoT Rules Engine
AWS Lambda
Amazon Kinesis
Big Data Backend
Speed Layer

Edge Computing

- Instead of all devices connecting to one cloud, multiple edge clouds manage devices
- Each edge is in charge of its own devices
- Offloads compute power to the edge
- Edge brings lower latency to high priority tasks
- Cloud handles low priority, but compute intensive tasks.



Edge Computing



The Greengrass Core is the runtime that enables the local execution of AWS Lambda, messaging, device shadows, and security. The Greengrass Core interacts directly with the cloud.

Any device that uses the IoT Device SDK can be configured to interact with Greengrass Core via the local network.

A defined group of Greengrass Cores and other devices that are configured to communicate with one another. A Greengrass Group may represent one floor of a building, one truck, or one home.

Takeaway or Resources

- https://aws.amazon.com/architecture/
- https://s3.amazonaws.com/awsmedia/architecturecenter/AWS_ac_ra_web_01.pdf
- https://s3.amazonaws.com/awslambda-serverless-web-refarch/RefArch_BlogApp_Serverless.png
- https://d0.awsstatic.com/diagrams/product-page-diagrams/Diagrams_greengrass-core.png
- http://searchdatacenter.techtarget.com/definition/edge-computing

Homework

- Virtual Machines
  - Read about them and see how they differ from your typical computers
- Check out Azure and Google Cloud Platform
  - See what services they have to offer
  - See how they differ from AWS
- Create your own EC2 Instance

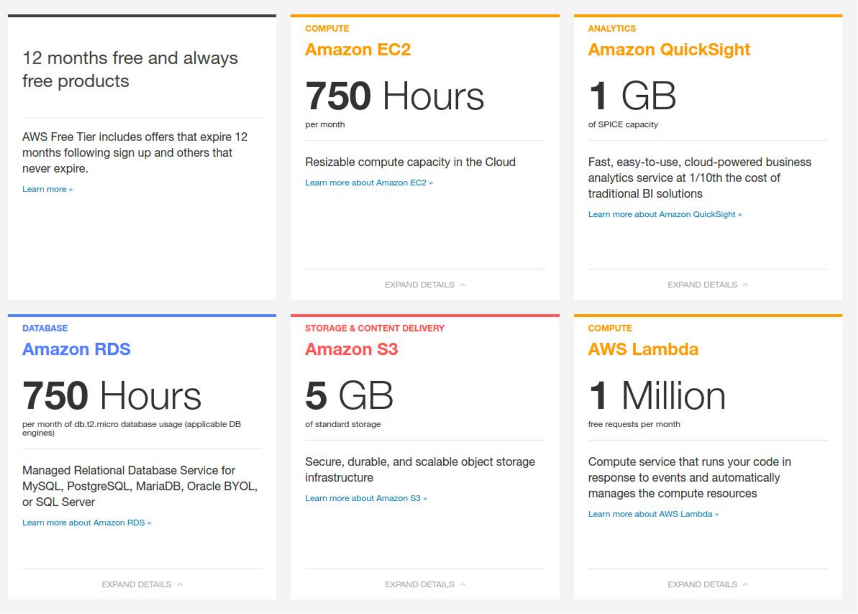
Create an AWS Account

Requirements

- What you need to sign-up for an account:
  - Email
  - Debit/Credit Card
    - You will not charged as long as you stay within usage limits
  - Phone Number

AWS Free Tier

- What you get for free for 12 months:
  - EC2 - 750 Hours
  - RDS - 750 Hours
  - S3 - 5 GB
  - And more!!
- What you get for free forever:
  - CodeCommit - 5 active users/month
  - Lambda - 1 Million requests/month
  - DynamoDB - 25 GB
  - Even more!!



12 months free and always free products

AWS Free Tier includes offers that expire 12 months following sign up and others that never expire.

Learn more »

COMPUTE

Amazon EC2

750 Hours

per month

Resizable compute capacity in the Cloud

Learn more about Amazon EC2 »

EXPAND DETAILS

ANALYTICS

Amazon QuickSight

1 GB

of SPICE capacity

Fast, easy-to-use, cloud-powered business analytics service at 1/10th the cost of traditional BI solutions

Learn more about Amazon QuickSight »

EXPAND DETAILS

DATABASE

Amazon RDS

750 Hours

per month of db.t2.micro database usage (applicable DB engines)

Managed Relational Database Service for MySQL, PostgreSQL, MariaDB, Oracle BYOL, or SQL Server

Learn more about Amazon RDS »

EXPAND DETAILS

STORAGE & CONTENT DELIVERY

Amazon S3

5 GB

of standard storage

Secure, durable, and scalable object storage infrastructure

Learn more about Amazon S3 »

EXPAND DETAILS

COMPUTE

AWS Lambda

1 Million

free requests per month

Compute service that runs your code in response to events and automatically manages the compute resources

Learn more about AWS Lambda »

EXPAND DETAILS

AWS Educate

- What is it?
  - ○ Amazon’s initiative to provide students with resources to learn how to use the cloud
  - ○ Provides credits to use on AWS services not available to free tier
- Who is eligible?
  - ○ Educators
  - ○ Academic Researchers
  - ○ Students
- What you get:
  - ○ Up to $150 in credits
  - ○ Training Course
  - ○ More!!

AWS Educate Starter Account

- Types of Educate Accounts
  - AWS Account
  - AWS Educate Starter Account
- AWS Account
  - More credits
  - Requires credit card
  - Account persists after credits run out
- AWS Educate Starter Accounts
  - Less credits
  - No credit card required
  - Account closes after credits run out

Resources

- AWS resources provided in the readings!
  - AWS Free Tier
  - AWS Educate

AWS Permissions: IAM Identities

Identity and Access Management (IAM) Identities

- What are they?
  - Ways to organize permissions for diferrent resources
- What's the point?
  - So that only admins have full access while new users have restricted access
  - Ex: Preventing a new hire from nuking your entire database

Identity and Access Management (IAM) Identities

- IAM Users
  - Name and Password
  - Access keys for API or CLI
  - Ex: Bob Seds
- IAM Groups
  - Collection of users
  - Everyone in group inherits the policies of that group
  - Managing policies related to common groups e.g. Admins, Database Team
- IAM Roles
  - Similar to users but can be assumed by any user
  - No credentials
  - Ex: Database Manager

IAM Users vs IAM Roles

- IAM Users
  - Only person working on an account
  - Multiple users
  - Want to use the command-line interface
- IAM Roles
  - Applications
  - Temporary acess

AWS Account Root User

- Spawned when you first create your account
- **Full unrestricted access**
- Not recommended for everyday use
- Instead, create an IAM user for yourself

Takeaways

- Separate levels of access to resources
- Create an IAM user for yourself
- Don’t use the root user for everyday purposes
- Keep your credentials secure

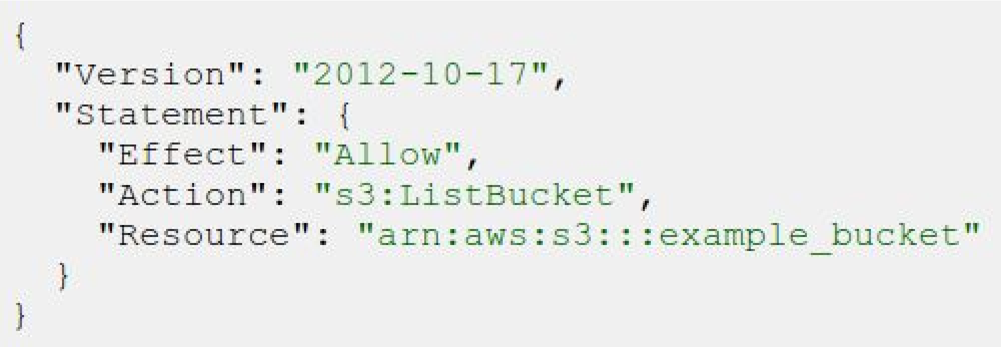
AWS Permissions: IAM Policies

Identity and Access Management (IAM) Policy

- What is it?
  - Document that defines permissions for certain users, groups, roles, or resources
- Components (Straight from AWS):
  - ***Effect*** *– whether the policy allows or denies access*
  - ***Action*** *– the list of actions that are allowed or denied by the policy*
  - ***Resource*** *– the list of resources on which the actions can occur*
  - ***Condition (Optional)*** *– the circumstances under which the policy grants permission*

Identity and Access Management (IAM) Policy

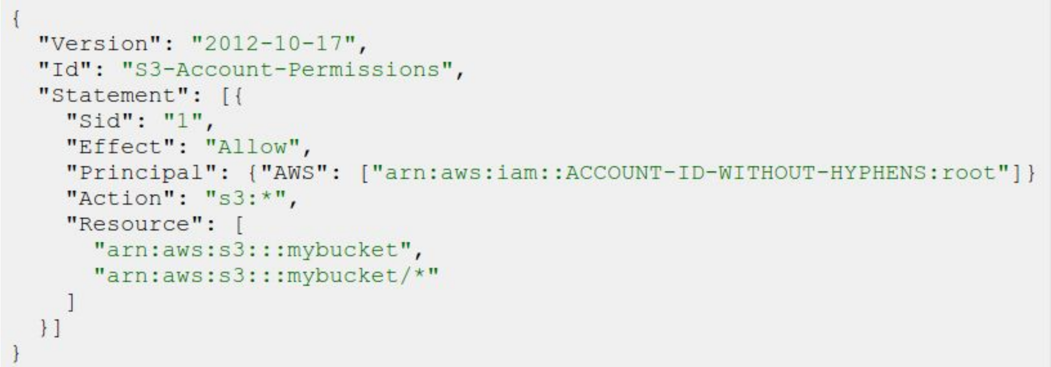
- What they look like?
    - Written in JSON



```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Identity and Access Management (IAM) Policy

- How to use them:
  - Attach to a IAM User or Group
  - Attach to a Resource
    - Needs to specify who is affected
    - Specified in **principal** field



```
{
  "Version": "2012-10-17",
  "Id": "S3-Account-Permissions",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```

Takeaways

- Assign IAM policies to specify level of access
- Great to limit the possibility of catastrophic accidents

AWS CLI and the SDK's

AWS Command Line Interface (CLI)

- Tool that provides commands to interact with AWS
- Same functionality that’s available on AWS Management Console
- Available on:
  - Linux terminal
  - MacOS terminal
  - Windows PowerShell

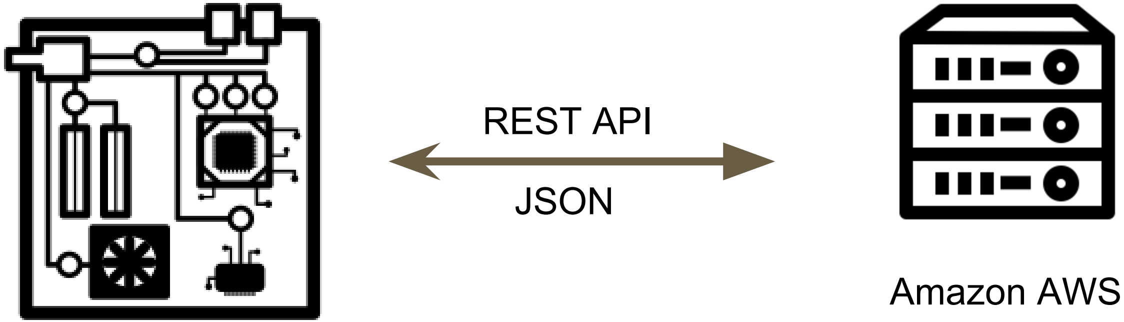
Takeaways

Homework

- Create an AWS account
- Create an IAM User with Admin Access
- Install the AWS CLI and configure it

REST API & JSON

REST



REST API

JSON

Amazon AWS

JSON

- Data Structure like XML (SOAP)
- JavaScript Object Notation

JSON

- Objects = { , , } - A collection of name/value pairs
- Arrays = [, ,] - An ordered list of values
- Object = { “firstname:Jane”, “lastname”:”Smith”}
- Array = [name1, name2, name3]

  name1 = { “firstname:Jane”, “lastname”:”Smith”}
  name2 = { “firstname:Kim”, “lastname”:”Estes”}

XML

```xml
<person>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <age>25</age>
  <address>
    <streetAddress>21 2nd Street</streetAddress>
    <city>New York</city>
    <state>NY</state>
    <postalCode>10021</postalCode>
  </address>
  <phoneNumbers>
    <phoneNumber type="home">212 555-1234</phoneNumber>
  </phoneNumbers>
</person>
```

JSON

```
 {
   "firstName": "John", "lastName": "Smith",
   "age": 25,
   "address": {
      "streetAddress": "21 2nd Street",
      "city": "New York",
      "state": "NY",
      "postalCode": "10021"
   },
   "phoneNumber": [
      {
         "type": "home",
         "number": "212 555-1234"
      }
    ]
}
```

JSON

```
person = { "firstName":"John" , "lastName":"Doe" }

phoneNumber": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
```

JSON

```
student = {  "Firstname" : "John",   "Lastname" : "Starks",     "sid" : "9343934"
}

class = [  {  "Firstname" : "John",   "Lastname" : "Starks",   "sid" : "9343934" },

              {  "Firstname" : "Jane",  "Lastname" : "Doe", "sid" : "9343934" }
            ]

        = [ student1, student2, ...., ]
```

JSON

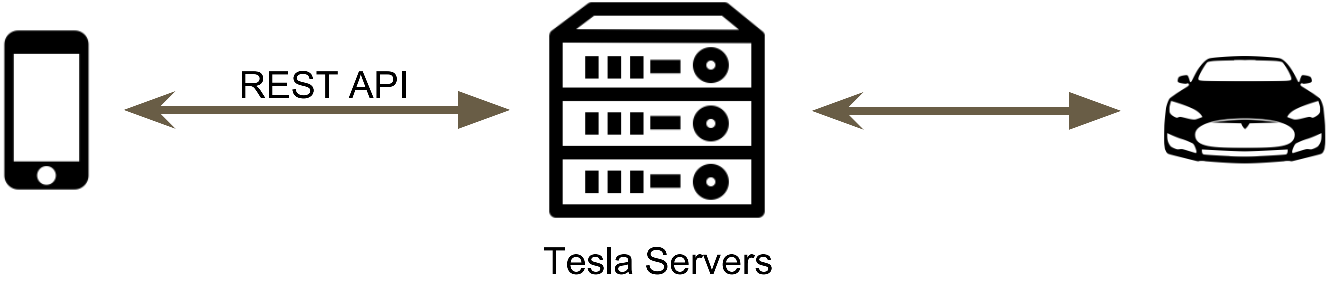
- www.json.org
- https://www.sitepoint.com/10-example-json-files/

RESTful API

- Representational state transfer
- RESTful Web services
- http://bit.ly/2gAWSK7

Tesla Motors REST Api

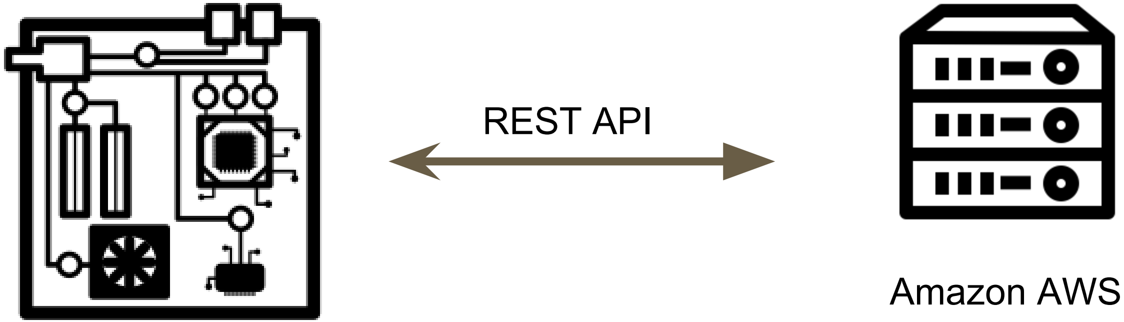
- http://docs.timdorr.apiary.io/
- http://bit.ly/2yl1p85



REST API

Tesla Servers

REST



REST API

Amazon AWS

JSON Framework

- https://github.com/miloyip/nativejson-benchmark C++
- https://github.com/google/gson - Java
- http://bsonspec.org/ - Binary JSON

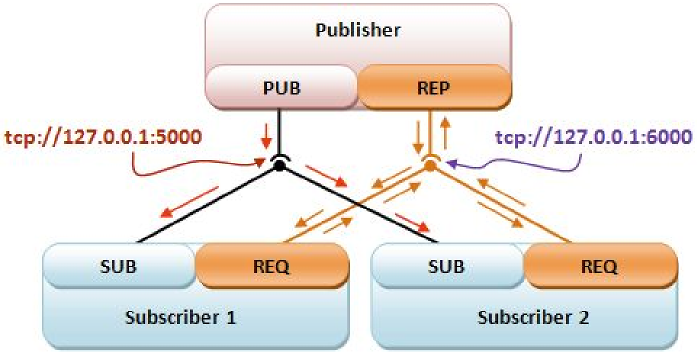
MQTT Protocol

Overview

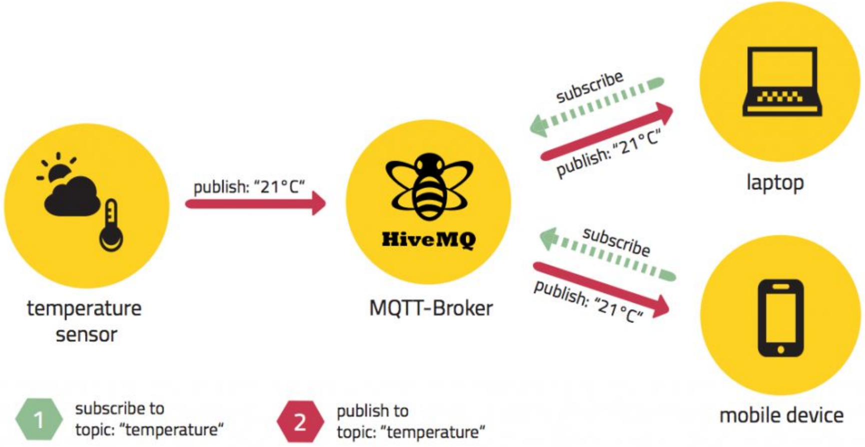
- Publisher Subscriber vs Broker
- MQTT

Publisher Subscriber Models

Pub/Sub

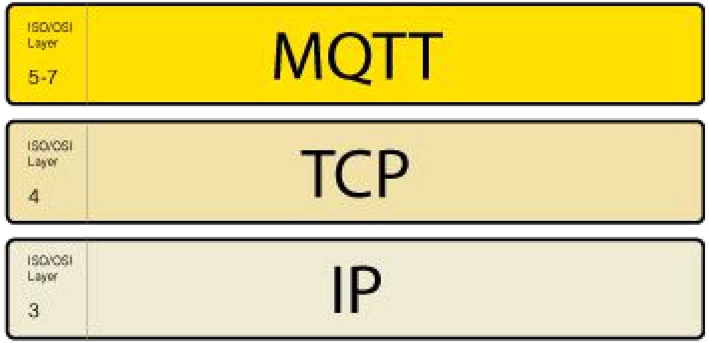


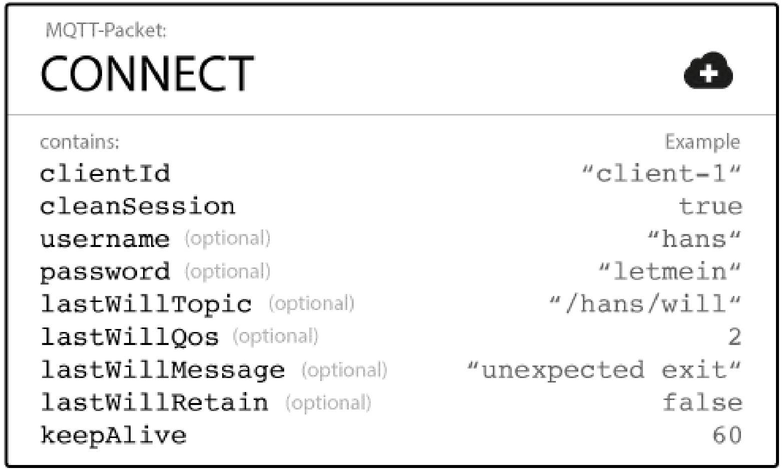
Message Broker



MQTT vs HTTP

- MQTT exists over TCP/IP, like http
- Can specify QoS
  - QoS=0 => no confirmation
  - QoS=1 => get packet recieved confirmation
  - QoS =2 => handshake, exactly one confirmation (http)
- clientId needs to be unique: use mac address
- Do not send user/pass in plaintext. Set up SSL/TLS asap

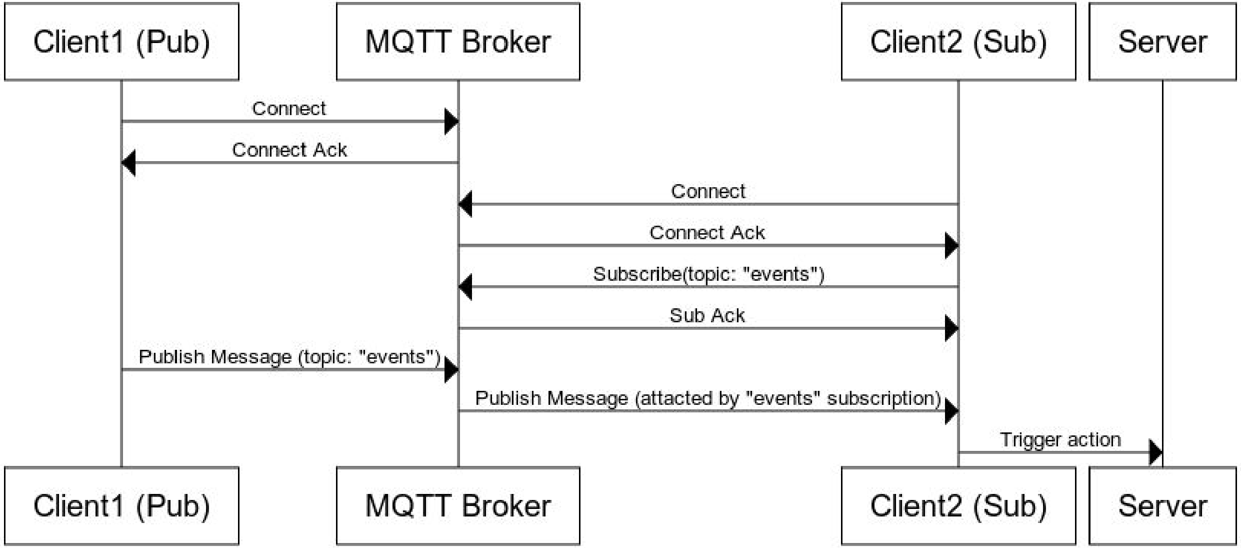



MQTT-Packet:

CONNECT

contains:	Example
clientId	"client-1"
cleanSession	true
username (optional)	"hans"
password (optional)	"letmein"
lastWillTopic (optional)	"/hans/will"
lastWillQos (optional)	2
lastWillMessage (optional)	"unexpected exit"
lastWillRetain (optional)	false
keepAlive	60

MQTT Process



Resources and Takeaways

- https://www.hivemq.com
- https://stackoverflow.com/questions/32538535/node-and-mqtt-do-something-on-message

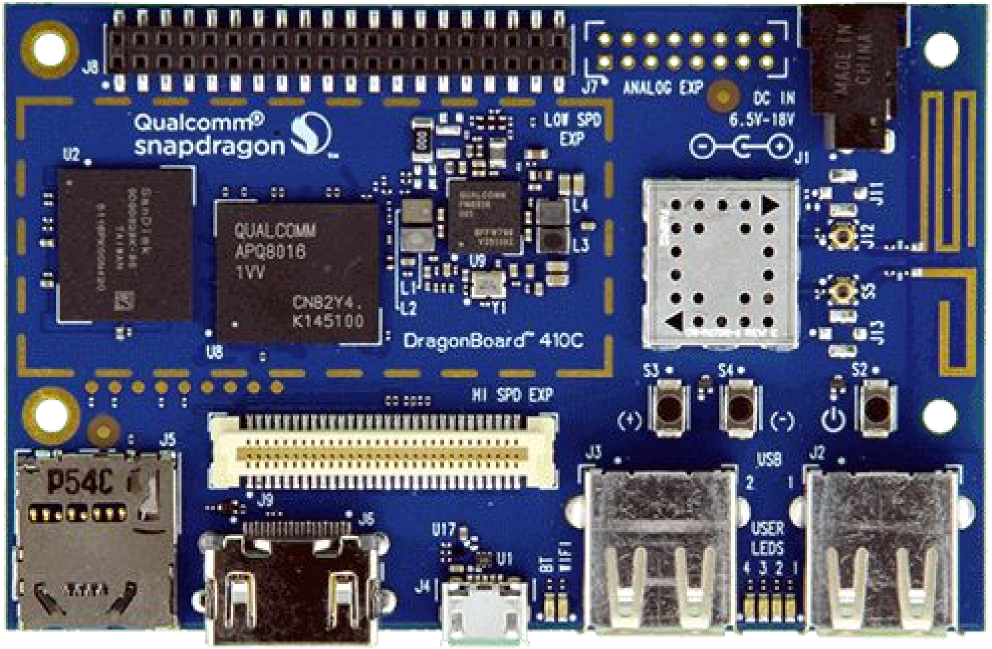
AWS IoT SDK

Overview

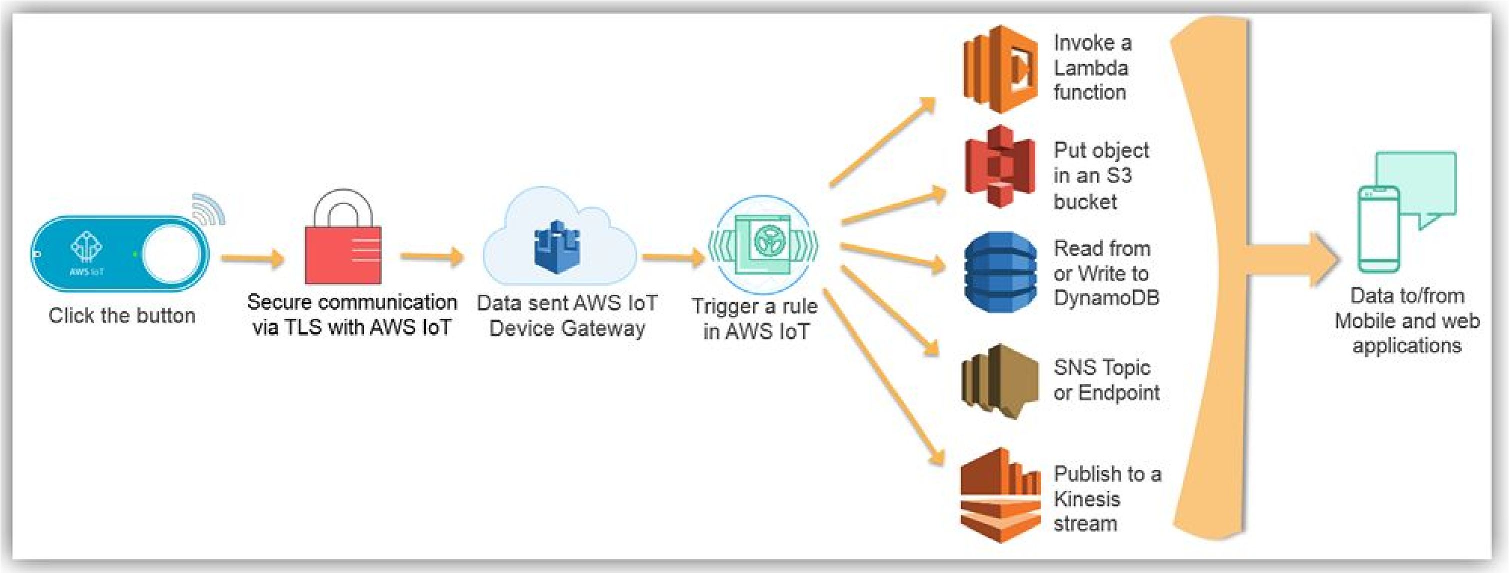
- AWS IoT
- AWS IoT SDK

IoT Devices

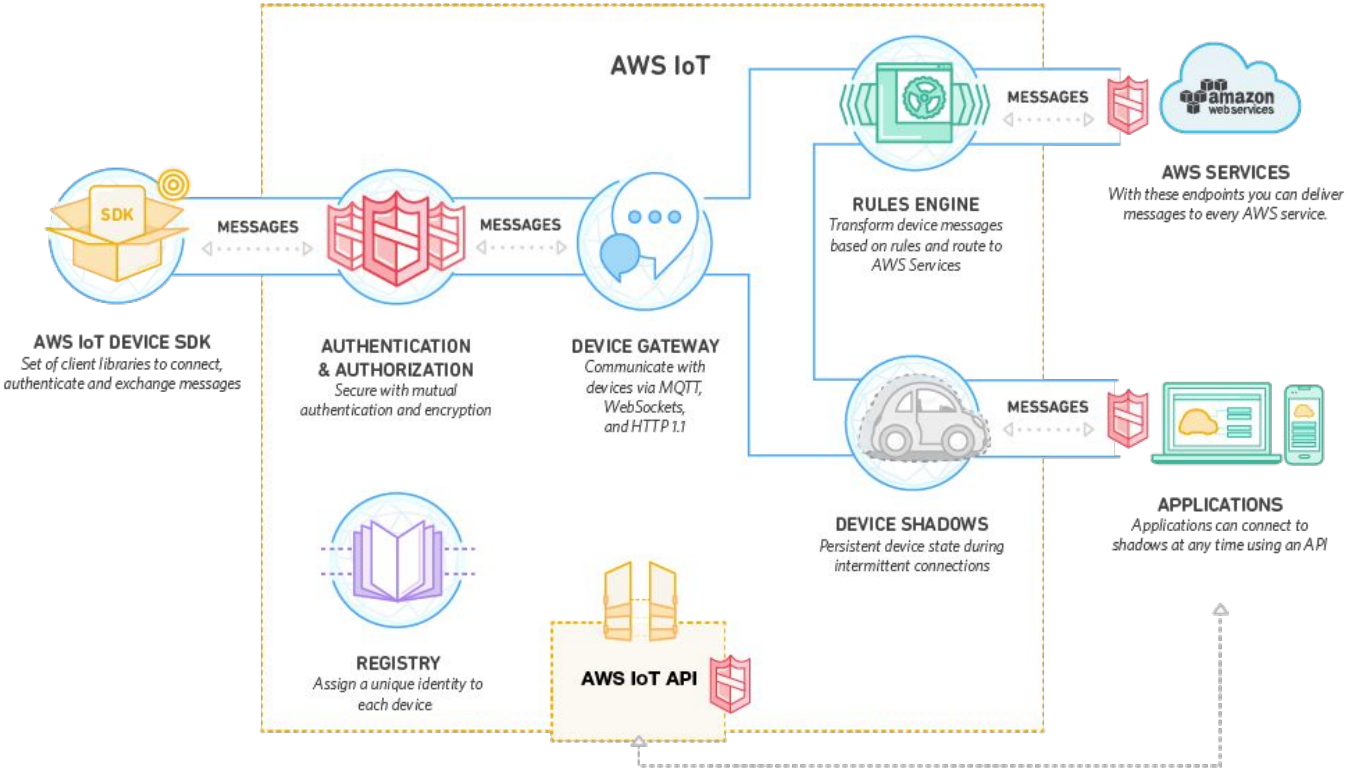
AWS IoT


Qualcomm® snapdragon
QUALCOMM APQ8016 1VV CN82Y4. K145100
DragonBoard™ 410C
LOW SPD EXP
ANALOG EXP
DC IN 6.5V-18V
HI SPD EXP
USB
USER LEDS
MADE IN CHINA

IoT Architecture


AWS IoT
Click the button
Secure communication via TLS with AWS IoT
Data sent AWS IoT Device Gateway
Trigger a rule in AWS IoT
Invoke a Lambda function
Put object in an S3 bucket
Read from or Write to DynamoDB
SNS Topic or Endpoint
Publish to a Kinesis stream
Data to/from Mobile and web applications

IoT Architecture


AWS IoT
SDK
MESSAGES
MESSAGES
MESSAGES
MESSAGES
amazon web services
AWS IoT DEVICE SDK
Set of client libraries to connect, authenticate and exchange messages
AUTHENTICATION & AUTHORIZATION
Secure with mutual authentication and encryption
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
AWS SERVICES
With these endpoints you can deliver messages to every AWS service.
DEVICE SHADOWS
Persistent device state during intermittent connections
APPLICATIONS
Applications can connect to shadows at any time using an API
REGISTRY
Assign a unique identity to each device
AWS IoT API

AWS IoT SDK

- https://github.com/aws/aws-iot-device-sdk-python

Resources and Takeaways

- https://d0.awsstatic.com/IoT/assets/awsiot_how_it_works_diagram.png
- https://media.amazonwebservices.com/blog/2017/IoTEnterprise-05-IoTButtonArchitecture-sm.png

Homework

- Read about REST API’s
- Write some JSON
- Write some XML
- Play with the GroupKt Countries API
- Read about MQTT
- Read about AWS IoT

Homework

- Read EC2 documentation
- Read about Apache
- Write some MySQL code
- Write some PHP code
- Read RDS documentation
- Read S3 documentation
- Read Code Commit documentation
- Play around with using all of the above

Rekognition

Overview

- Describe Rekognition
- Upload image via S3
- Detect labels using AWS CLI
- Detect labels using AWS SDK for Python

Polly

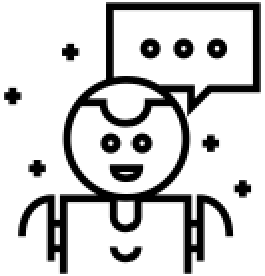
Overview

- Describe Polly
- Use Polly through Python:
  - Different phrases
  - Different speakers

Lex

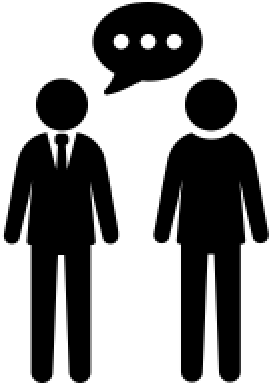
Overview

- Describe Lex
- Create a Lex chatbot on the console
- Interact with that bot through Python



Lex

- Allows for conversational interfaces
    - Essentially chat bots that help you order, schedule, etc.
- Can understand audio and text
- Responds with pre-set messages
- Lex is what powers Amazon Alexa



Important Notes

- Currently only available in certain regions:
  - N. Virginia - us-east-1
  - Ireland - eu-west-1
- Simply change your region in the console and SDK
  - On console, select your current region and change it to **US East (N. Virginia)**
  - On SDK, run **aws configure** and enter **us-east-1** for region
- Enable your IAM user to have:
  - AmazonLexFullAccess

GPU’s for Machine Learning

Reasoning

- Why use GPU's?
  - Much faster than using CPU's
- Why use AWS instead of a home computer?
  - Training machine learning models is very long

Overview

- Create a GPU instance with Deep Learning libraries pre-installed.
- Clone the Tensorflow repository
- Train a simple model for recognizing digits.

Homework

- Read about Rekognition
- Read about Lex
- Read about Polly
- Read about Machine Learning AMI's
- Play with all the above