

SERVERLESS

ARCHITECTURE

- är det nya svarta -

CADEC 2017 - Peter Larsson

SERVERLESS ARCHITECTURE

“...lets you run code without provisioning or managing servers.”

“...you pay only for the compute time you consume - there is no charge when your code is not running.”

“...you can run code for virtually any type of application or backend service - all with zero administration.”

- Amazon AWS Lambda

FaaS Providers



SERVERS INNEBÄR

- **Administration** - nätinfrastuktur, orkestrering, autoskalning
- **Kostnader** - varje miljö kräver ett minimum av aktiva servers
- **Underhåll** - servers måste hållas up-to-date
- **Sämre utnyttjandegrad** - för grovkornigt för micro-services

KOSTNADSJÄMFÖRELSE

1 000 000 transaktioner per dag @ 100 ms/trans.

2x EC2 (m4.large) \$ 5.8

Lambda (128 MB) \$ 0.4

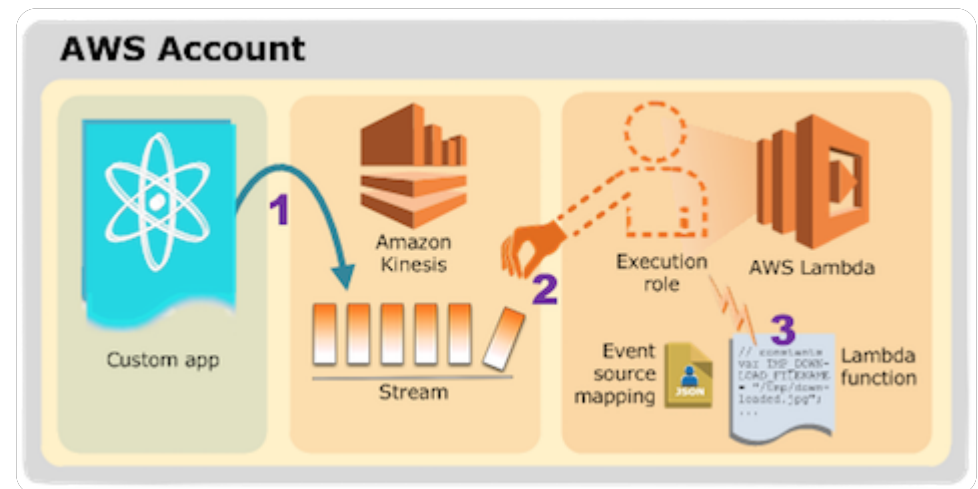
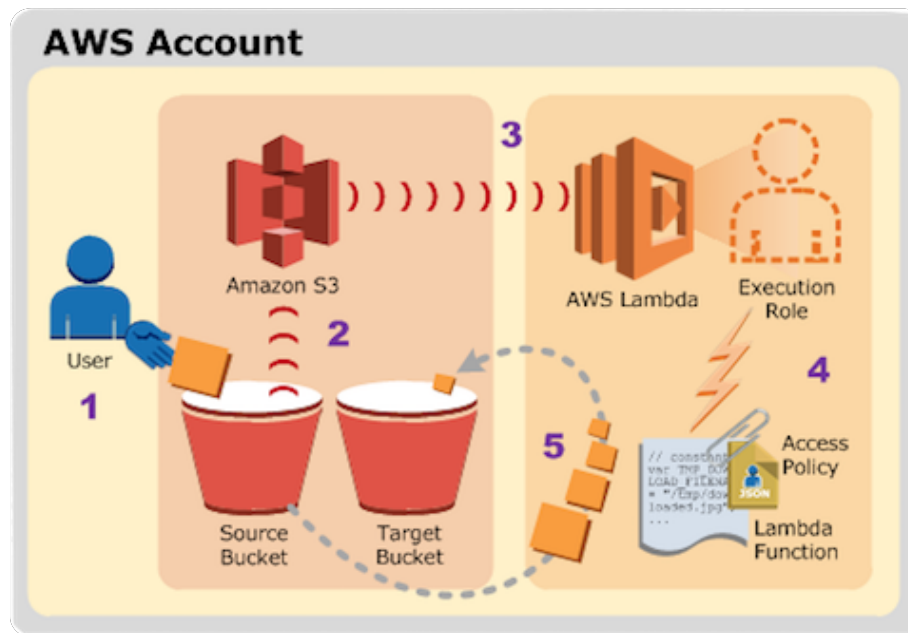
Servers kostar 10-100 ggr mer!!!

ARKITEKTUR - EXEMPEL

Serverless är typiskt eventdrivet för Mobil-, Webb- och IoT-backendtjänster.

Skapa tumnagel av bild

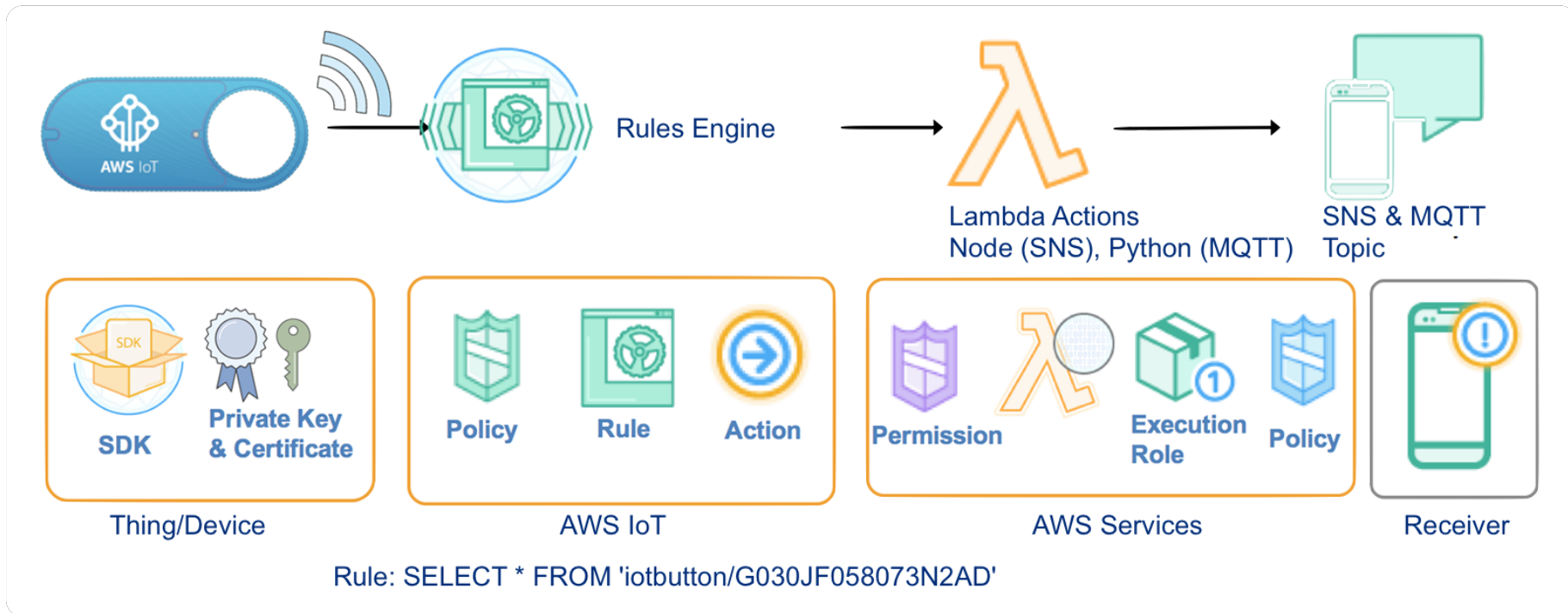
Bearbeta en dataström



AWS tillhandahåller 78 fördefinierade blueprints



DEMO - IOT BUTTON



PYTHON LAMBDA CODE (MQTT)

```
import boto3, json, socket, datetime

client = boto3.client('iot-data', region_name='eu-west-1')

def ip():
    return [l for l in ([ip for ip in socket.gethostbyname_ex(socket.getl

def lambda_handler(event, context):
    print event
    xevent = {
        'title': 'Demo event assembled by lambda python function',
        'timestamp': datetime.datetime.utcnow().isoformat() + 'Z',
        'ipAddress': ip(),
        'serialNumber': event['serialNumber'],
        'batteryVoltage': event['batteryVoltage'],
        'clickType': event['clickType']
    }
    response = client.publish(
        topic='iotprocessed',
        qos=1,
        payload=json.dumps(xevent)
    )
    return xevent
```

CLIENT - PAHO MQTT/WEBSOCKET

```
// html page callback, fragments...
IoTEventListener.initialize(function(event) {
    console.log("clicked: " + event.clickType);
    switch (event.clickType) {
        ...
    }
});
```

```
// client code, fragments...
IoTEventListener.initialize = function(cb) {
    var client = new Paho.MQTT.Client(requestUrl, clientId);
    client.onMessageArrived = function(message) {
        console.log("msg arrived: " + message.payloadString);
        var event = JSON.parse(message.payloadString);
        cb(event)
    };
    ...
    client.subscribe("iotprocessed");
```



FÖRDELAR, APP. ARKITEKTUR

- Stort utbud av färdiga designmönster och integrationer
- FaaS är en micro-service plattform
- ...och som arkitekt/utvecklare slipper man hantera:
 - CPU, cores, trådar
 - Policies och roller
 - Elastisk auto-skalning och lastbalansering
 - Uppgraderingar, discovery och konfiguration
 - Distribuerad loggning, spårning och instrumentering
 - Säkerhetsaspekter som DDoS etc
 - Upprätthålla produktionslika test- och stagemiljöer

Fokus på applikation istället för plattform!

MEN...AVVAKTA MED "ALL-IN"

- Komplex kompetensutmanande arkitektur
- Risk för så kallade "nanotjänster"
 - A nanoservice is a service whose overhead (communications, maintenance, and so on) outweighs its utility
- Saknas fortfarande beprövade metoder och verktyg för att hantera en grupp av funktioner (applikation/tjänst)
- Inlåsnings effekter på grund av beroenden till såväl plattformsspecifika SDK som stödtjänster
- Fler säkerhetsfrågor som måste besvaras/hanteras
- Detaljer (the devil is in the detail)
 - Max samtidiga exekveringar är per konto och inte miljö
 - Max exekveringstid (5 min)
 - Latens (startup, dataåtkomst)
 - Testbarhet, troligtvis molnet som gäller (utmaningar vad gäller mock-tjänster)
 - Konfiguration för olika miljöer
 - Databashantering etc...



PRAKTISKA ERFARENHETER - AWS LAMBDA/IOT

- Använd Serverless Framework
 - Förenklar applikationsprovisionering
 - Stöd för Amazon, OpenWhisk, Google och Azure
- Hitta rätt granularitet på funktioner och moduler (applikationsstackar)
 - Generellt vore det önskvärt med mindre enheter
- Var beredd på att experimentera vad gäller data-management
 - RDBMS innebär prestandautmaningar och kräver extra nätadministration (VPC)
 - AWS Kinesis medför 100-600ms latens per meddelande men är billigare än MQTT topics
- Serverless Framework, AWS Lambda, API Gateway, Javascript och Node.js fungerar alldeles utmärkt



DEMO - SERVERLESS DEPLOY (HTTP API)

```
# serverless.yml
service: aws-python-simple-http-endpoint

frameworkVersion: ">=1.2.0 <2.0.0"

provider:
  name: aws
  runtime: python2.7
  region: eu-west-1

functions:
  currentTime:
    handler: handler.endpoint
    events:
      - http:
          path: ping
          method: get
```

Serverless Framework

THE END!

LÄNKAR

AWS Lambda, IoT, SNS, API-GW, S3
(<https://aws.amazon.com>)

Azure (<https://azure.microsoft.com/services/functions>)

Google (<https://cloud.google.com/functions>)

IBM (<https://developer.ibm.com/openwhisk>)

Iron (<https://www.iron.io>)

Webtask (<https://webtask.io>)

MQTT (<http://mqtt.org>)

Serverless Framework (<https://serverless.com>)