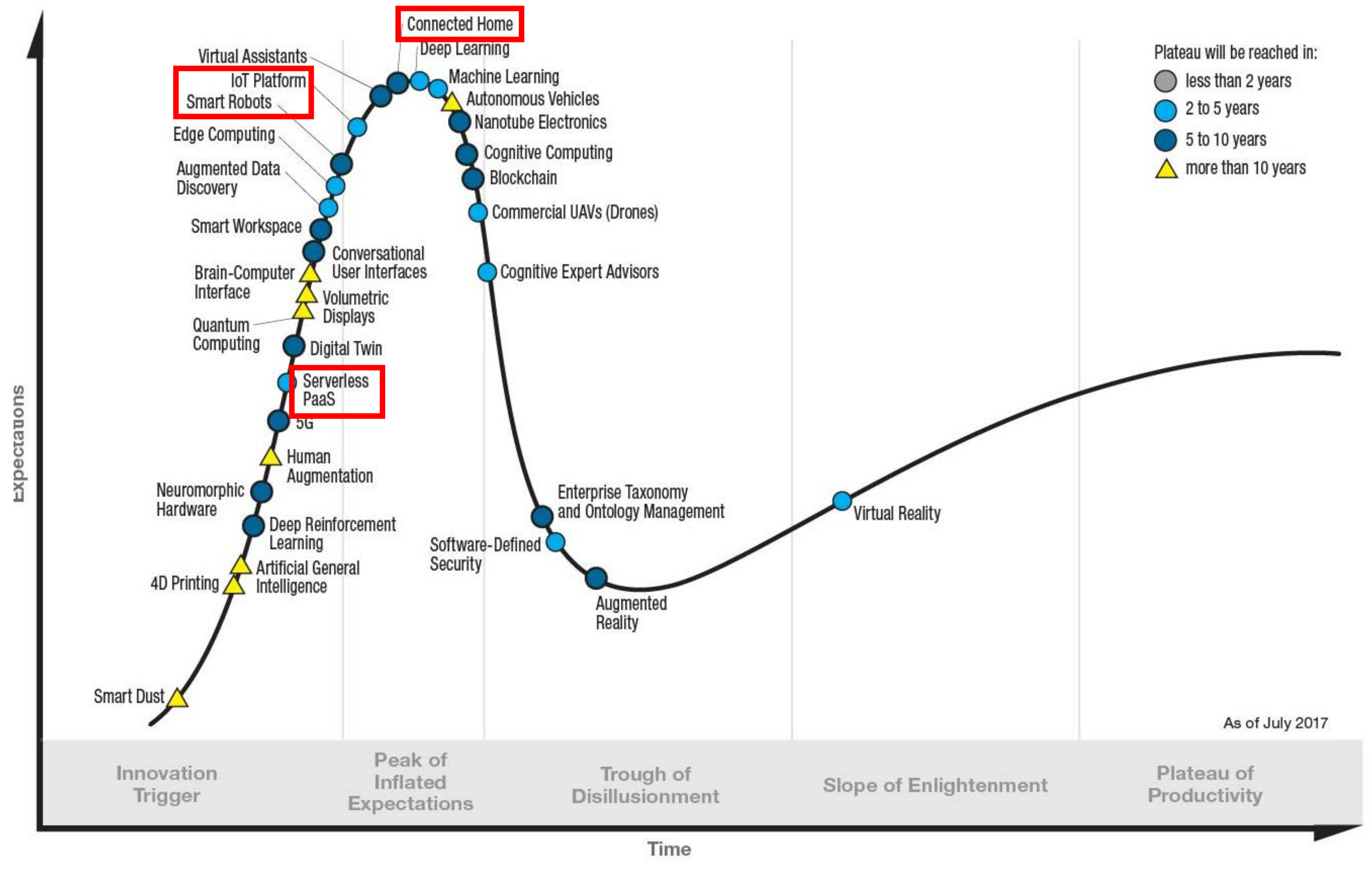
Serverless IoT at iRobot

Ben Kehoe @ben11kehoe
Cloud Robotics Research Scientist

About me

- Cloud Robotics Research Scientist at iRobot
- Serverless evangelist
- AWS Community Hero

Gartner Hype Cycle for Emerging Technologies, 2017



gartner.com/SmarterWithGartner

Source: Gartner (July 2017)

Gartner

@ben11kehoe

Cloud Robotics:

Connecting robots to the internet to help them do more and better things

1

What is serverless?

What is serverless?

- The wrong first question

What does serverless do?

- Cheaper
- Faster
- Leaner
- Better

What serverless is not

- FaaS
- Event-driven compute
- Never paying for idle
- Containers?
- Public cloud

Serverless is a spectrum

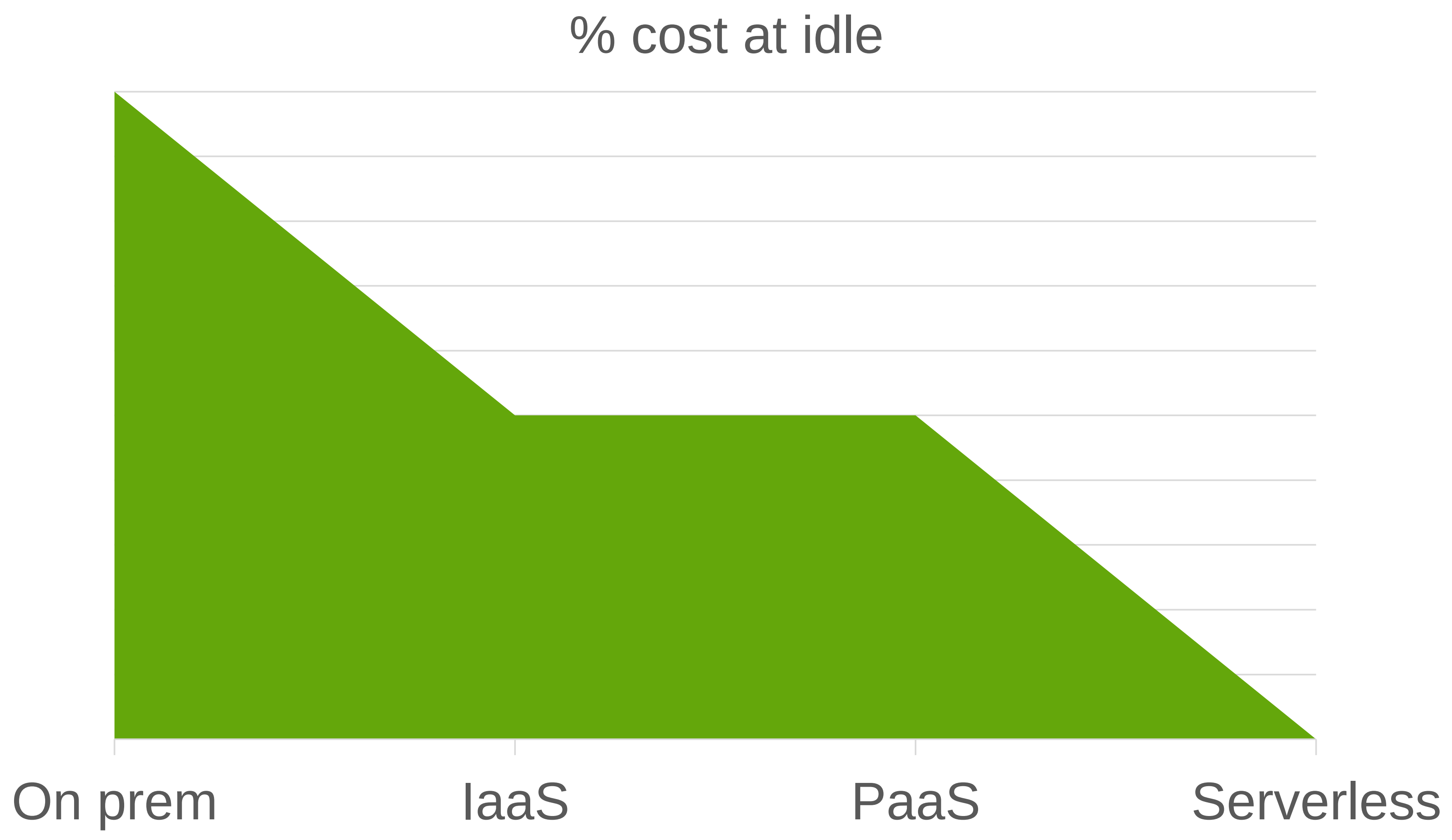
Increasing serverless-ness with:

- Service-full + emphemeral compute
- Resources billed → resources used
- Smaller, more abstract control plane

What does serverless do?

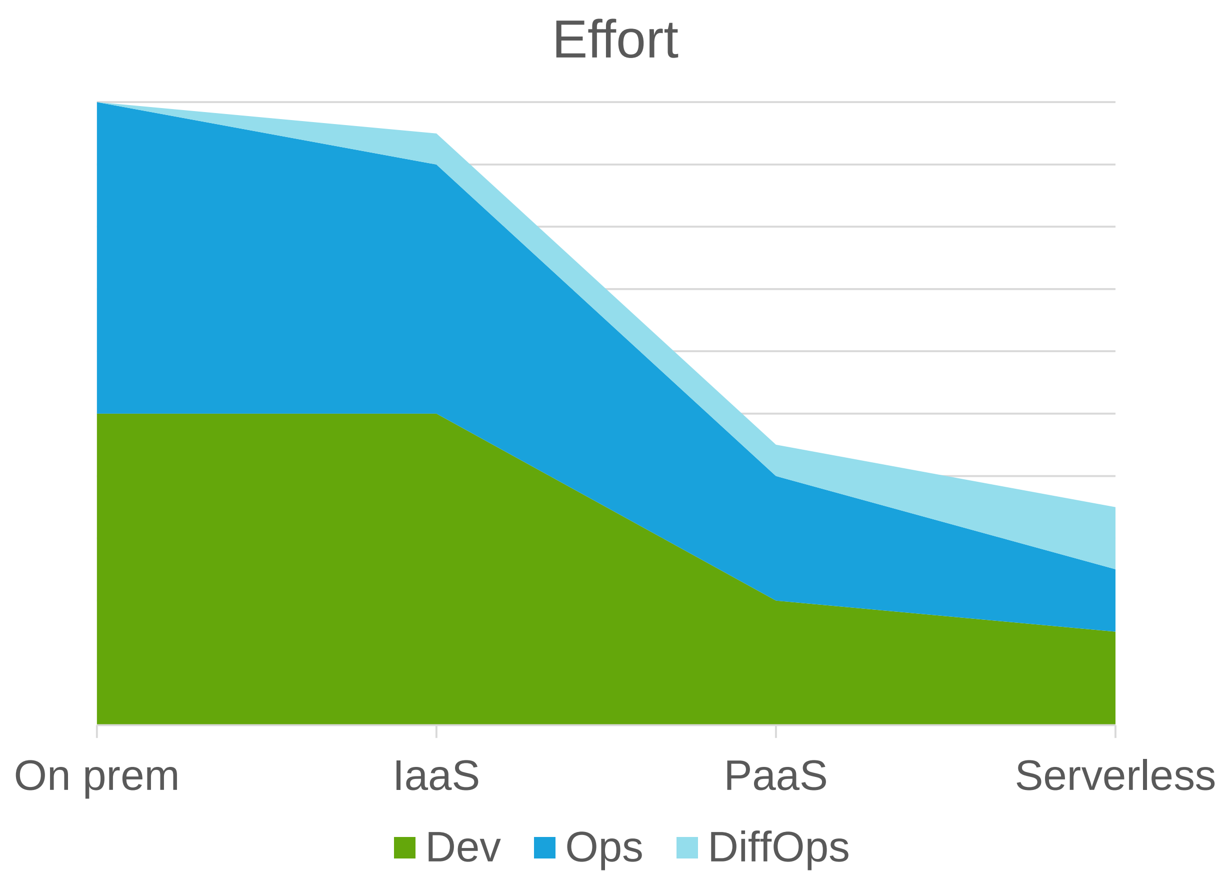
- Cheaper
- Faster
- Leaner
- Better

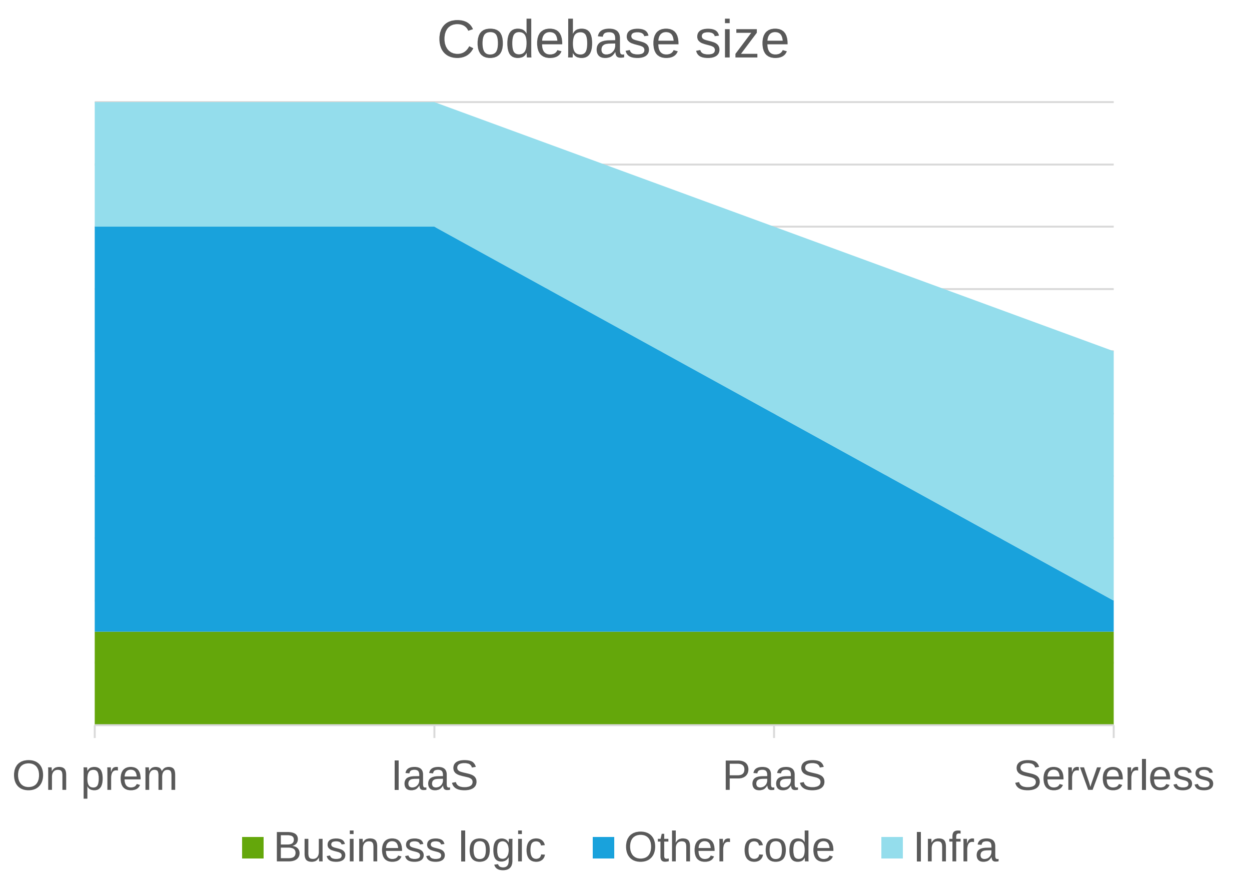
Cheaper



Not to scale

Faster, Leaner



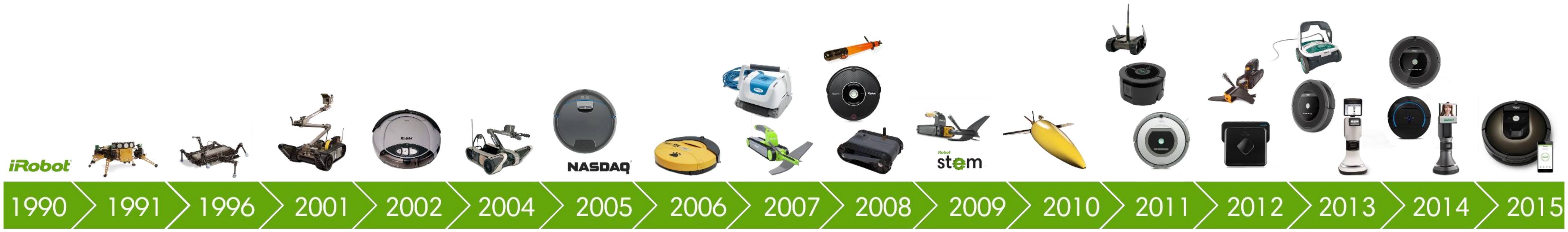


Not to scale

2

iRobot's journey

2015


iRobot
NASDAQ
stem
1990
1991
1996
2001
2002
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015

@ben11kehoe

iRobot

iRobot

Then

Roomba® 690

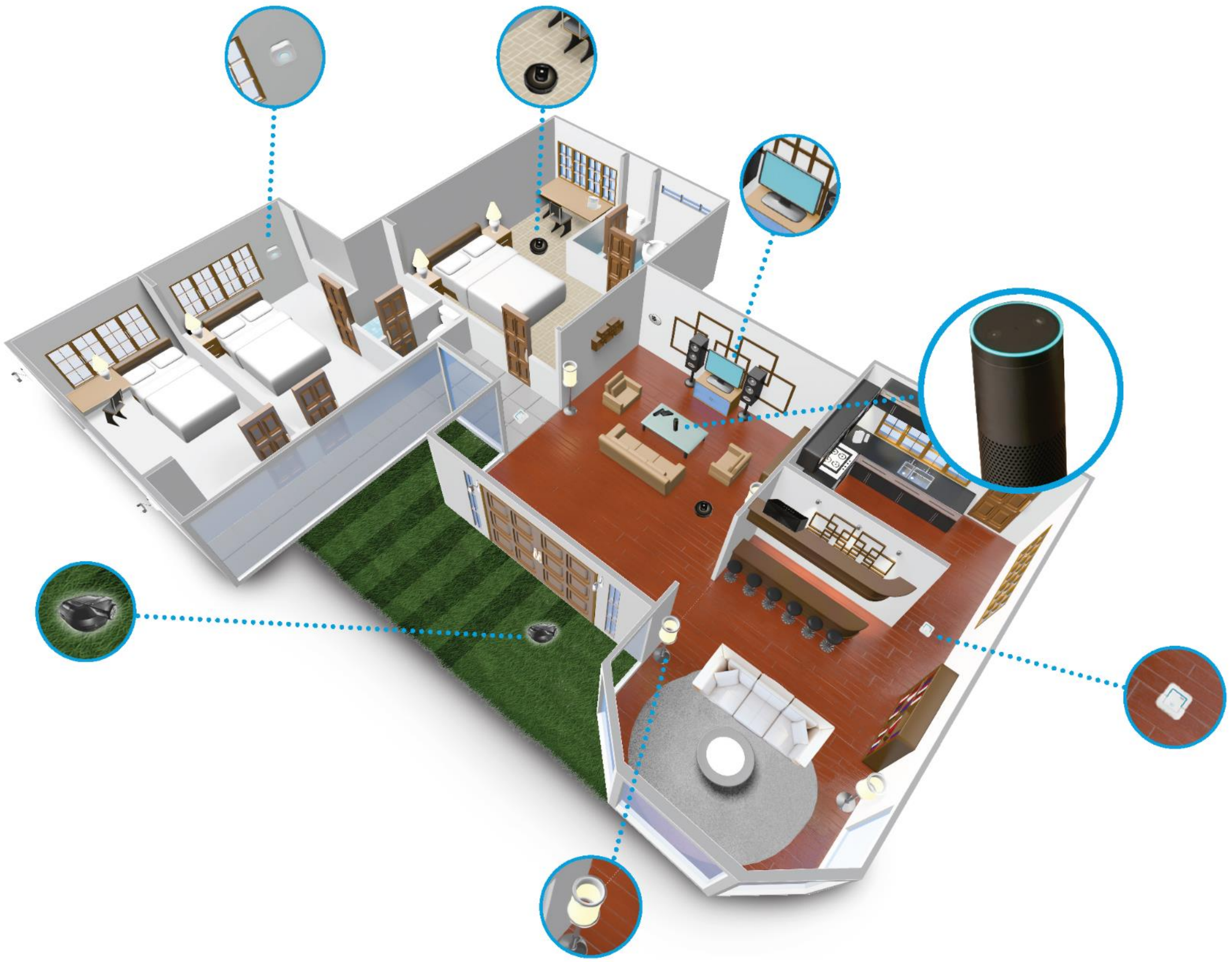
Roomba® 890

Roomba® 960

Roomba® 980

Then

Now

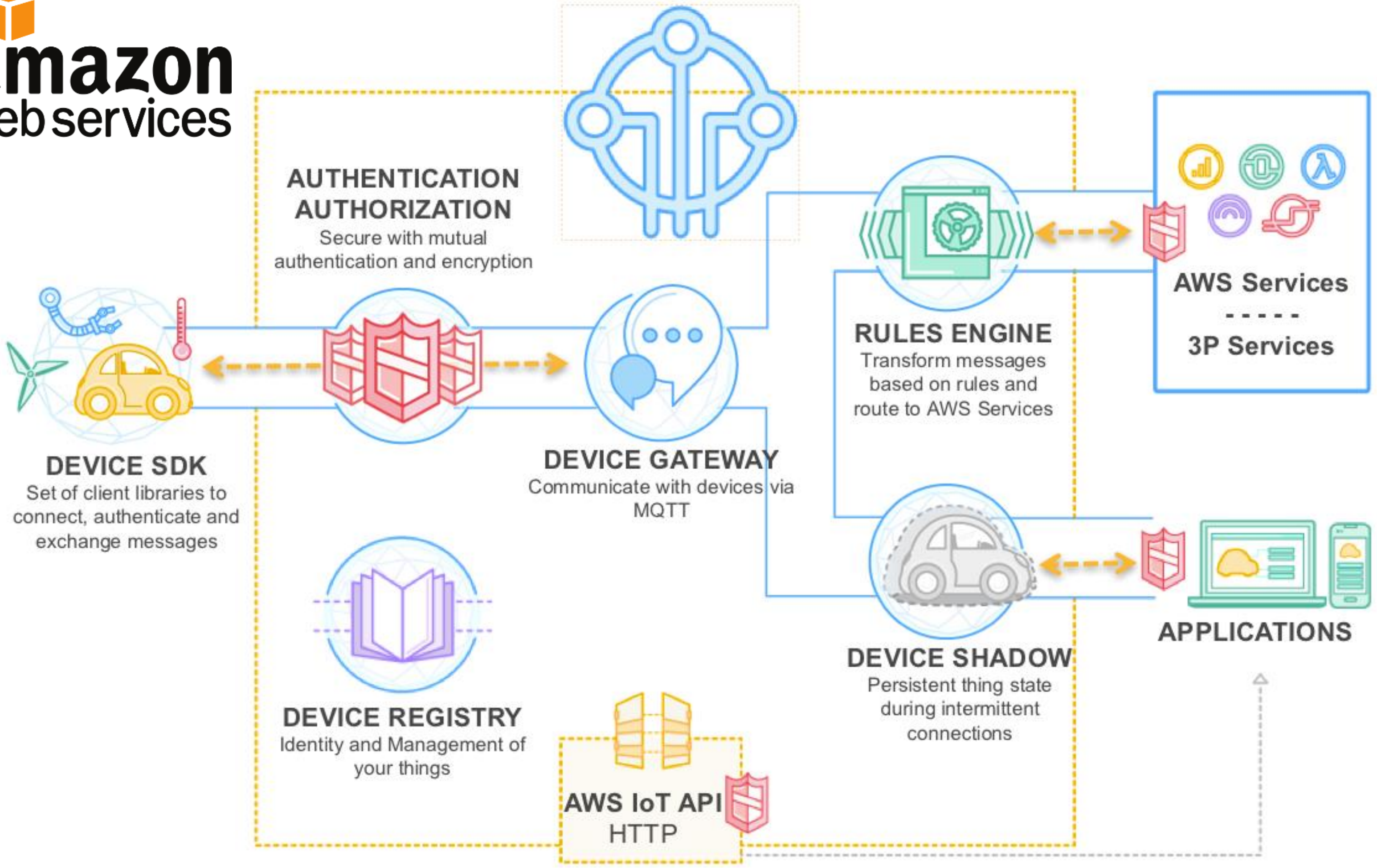


Then Now Future

@ben11kehoe

Then 2015 Now Future

AWS IoT


AUTHENTICATION
AUTHORIZATION
Secure with mutual authentication and encryption
DEVICE SDK
Set of client libraries to connect, authenticate and exchange messages
DEVICE GATEWAY
Communicate with devices via MQTT
RULES ENGINE
Transform messages based on rules and route to AWS Services
AWS Services
- - - - -
3P Services
DEVICE SHADOW
Persistent thing state during intermittent connections
APPLICATIONS
DEVICE REGISTRY
Identity and Management of your things
AWS IoT API
HTTP

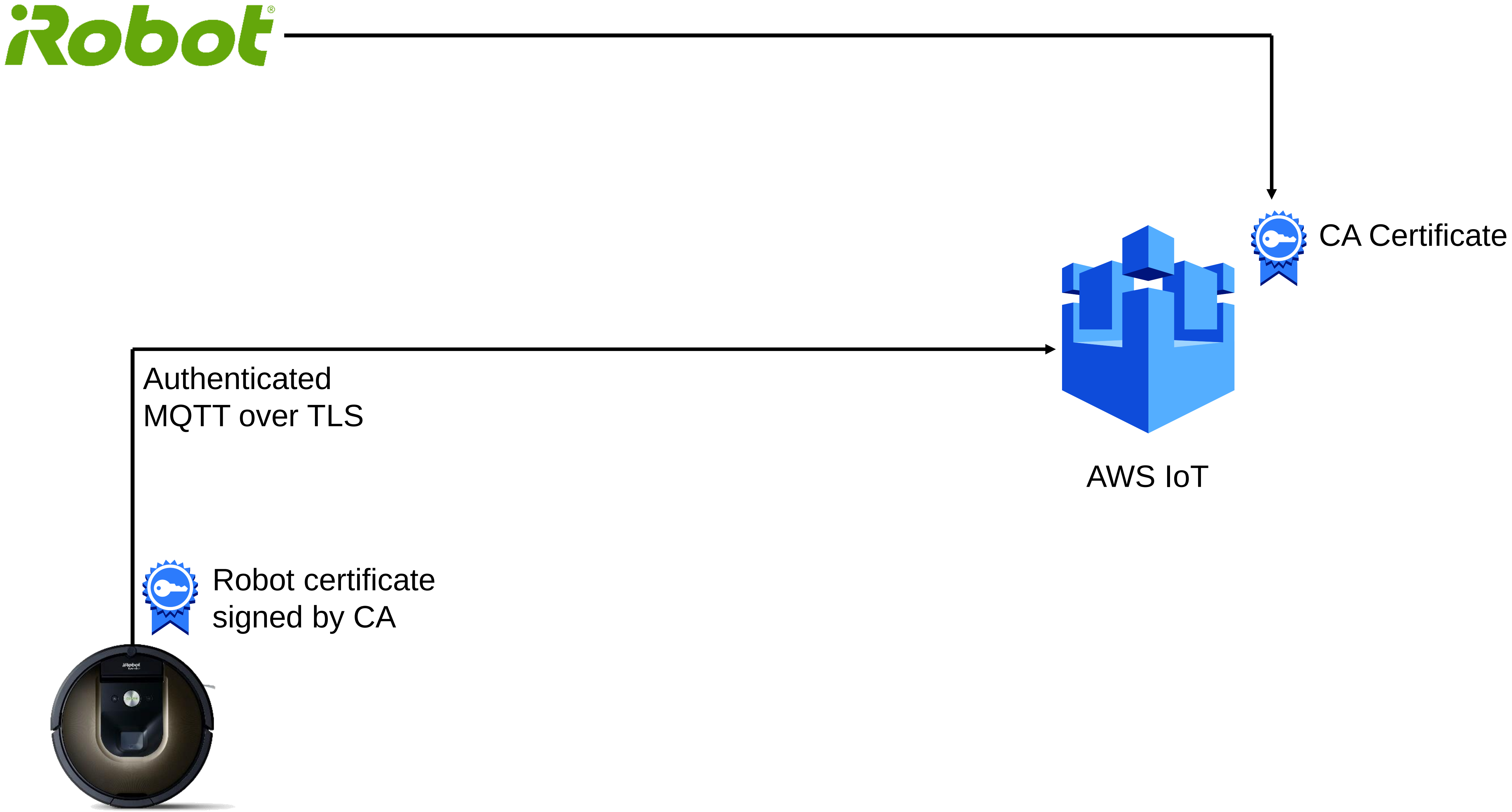
AWS IoT

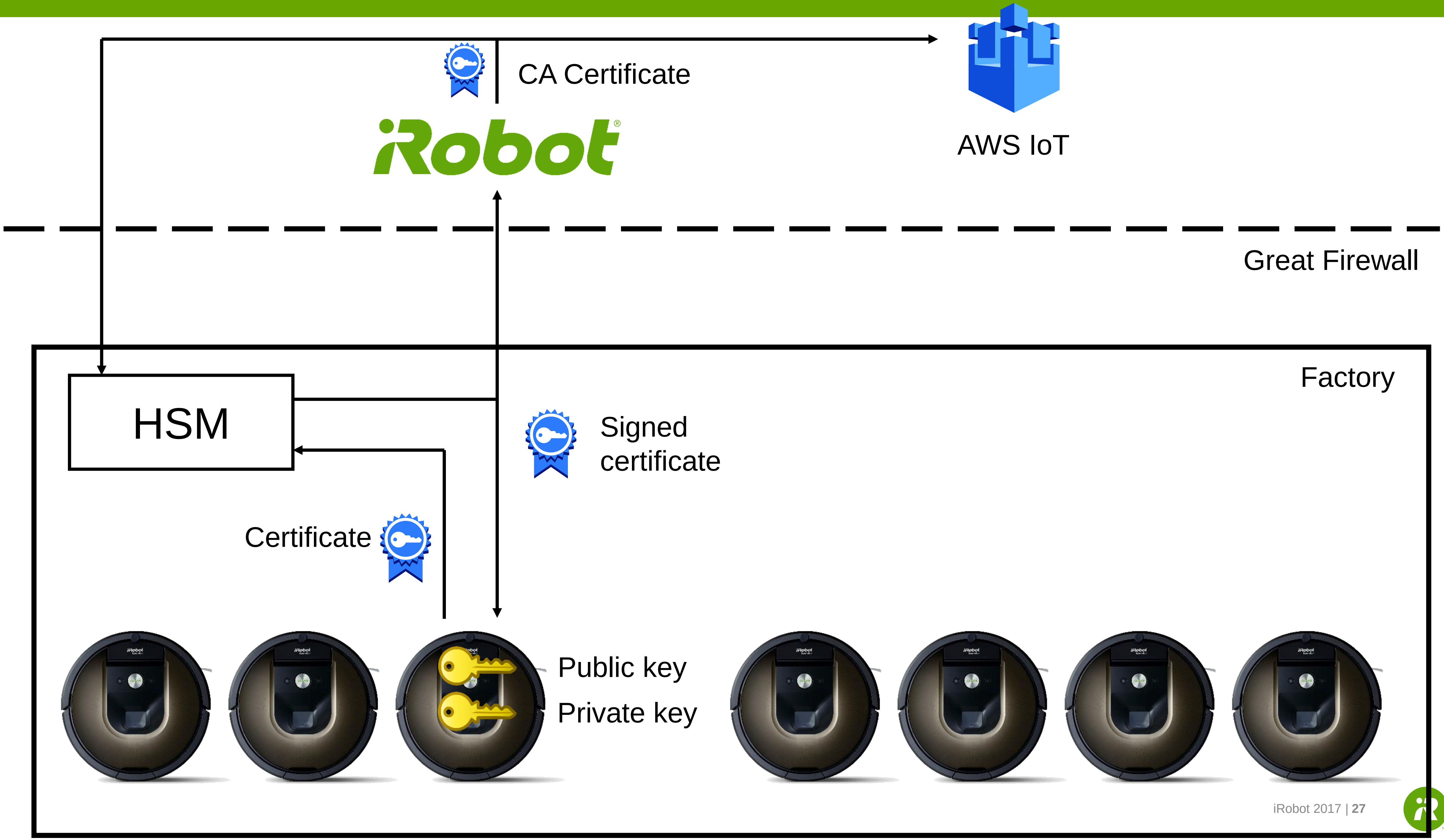
- Serverless
  - Event-driven
  - Scalable
- Integrates with AWS ecosystem
- Device shadows
- Integrates with your process

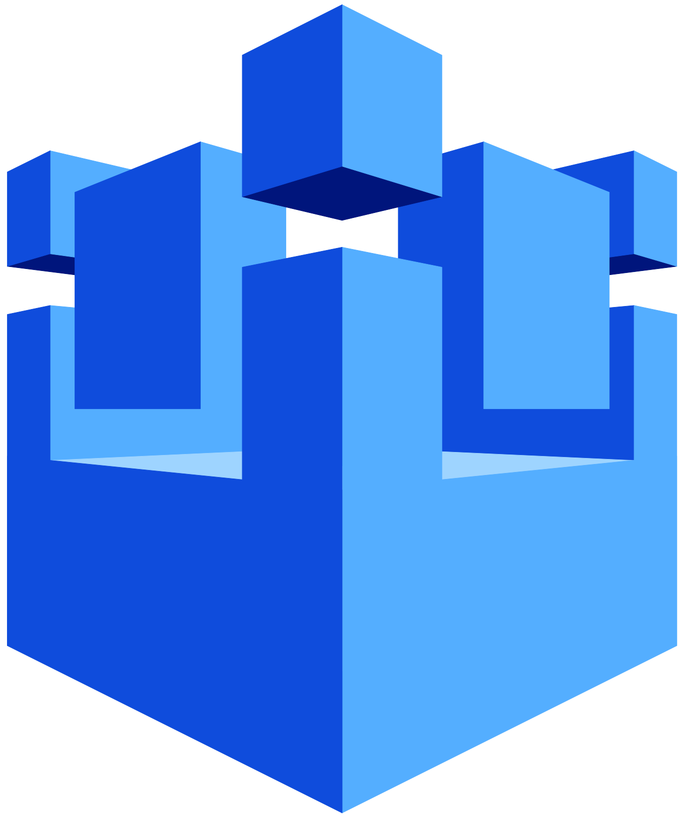
IoT + serverless

A natural fit

- Event-driven
- Scalable
- Lean for device makers
  - Reverse: AWS Greengrass
- Focus


iRobot
CA Certificate
AWS IoT
Authenticated
MQTT over TLS
Robot certificate
signed by CA


CA Certificate
iRobot
AWS IoT
Great Firewall
Factory
HSM
Signed certificate
Certificate
Public key
Private key

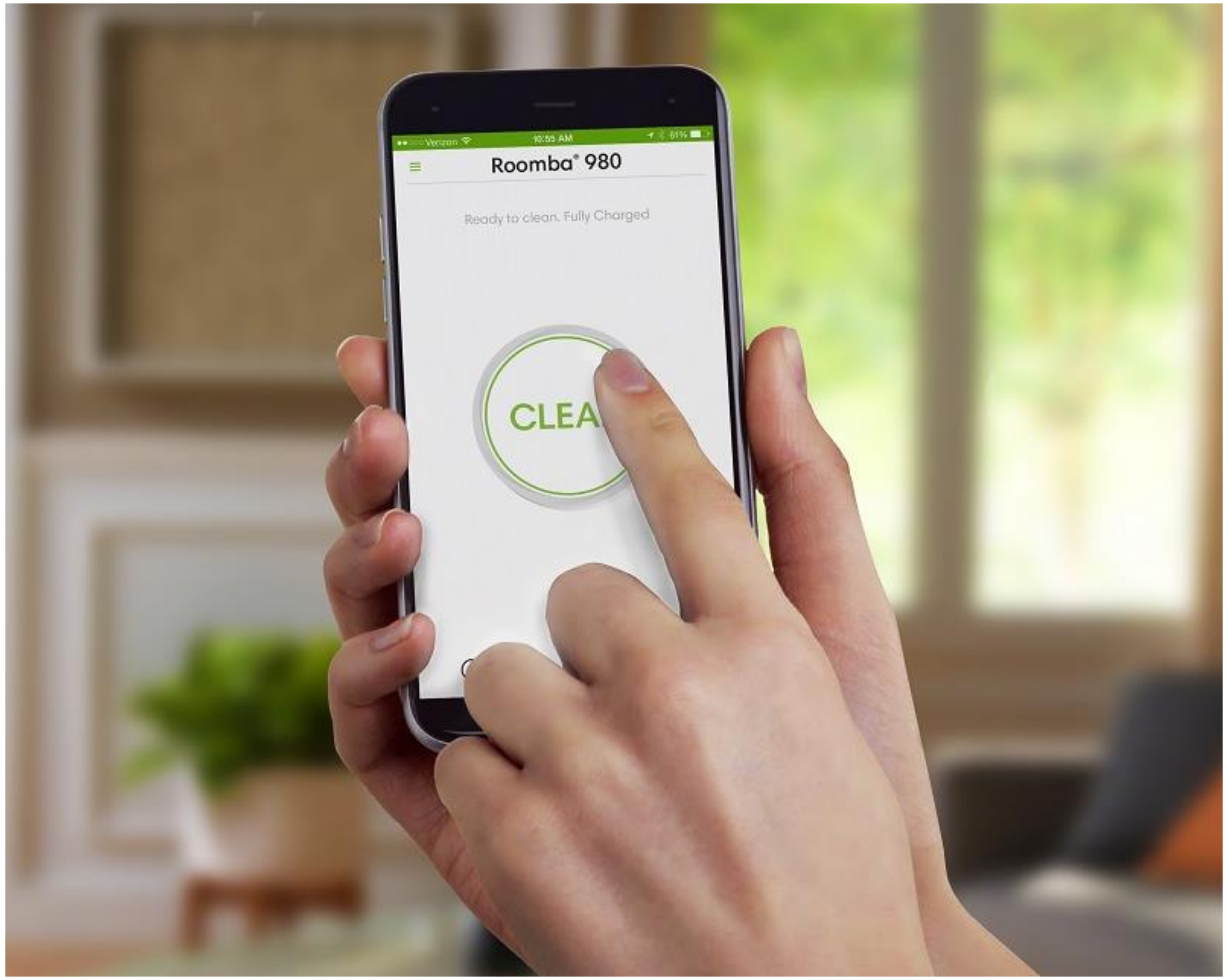


Long story short: success!

- Fully serverless production cloud
- 2 million connected robots by 2018
- Mostly serverless analytics platform
- Basis for future data-powered platform

iRobot scale

- Production application:
  - 100+ Lambda functions
  - 25 AWS services
  - 0 unmanaged EC2 instances
- AWS footprint:
  - ~50 accounts, growing constantly
  - 1000s of Lambda deploys per day
- Low single digit FTE supporting operations



Long story less short

- Architecture
- Deployment
- Operations
- Organizational

4

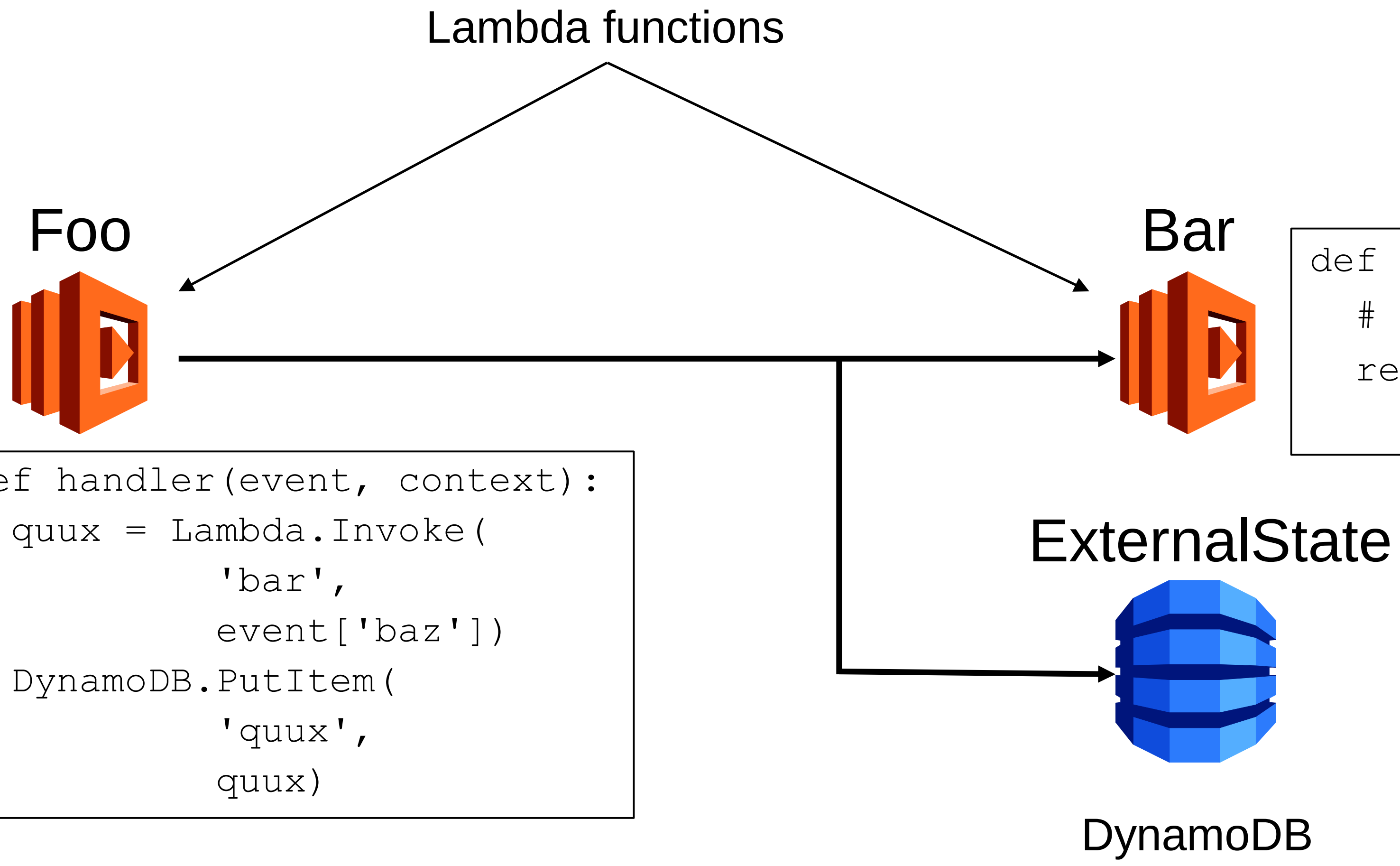
Serverless architecture at iRobot

Before serverless

aka the dark ages

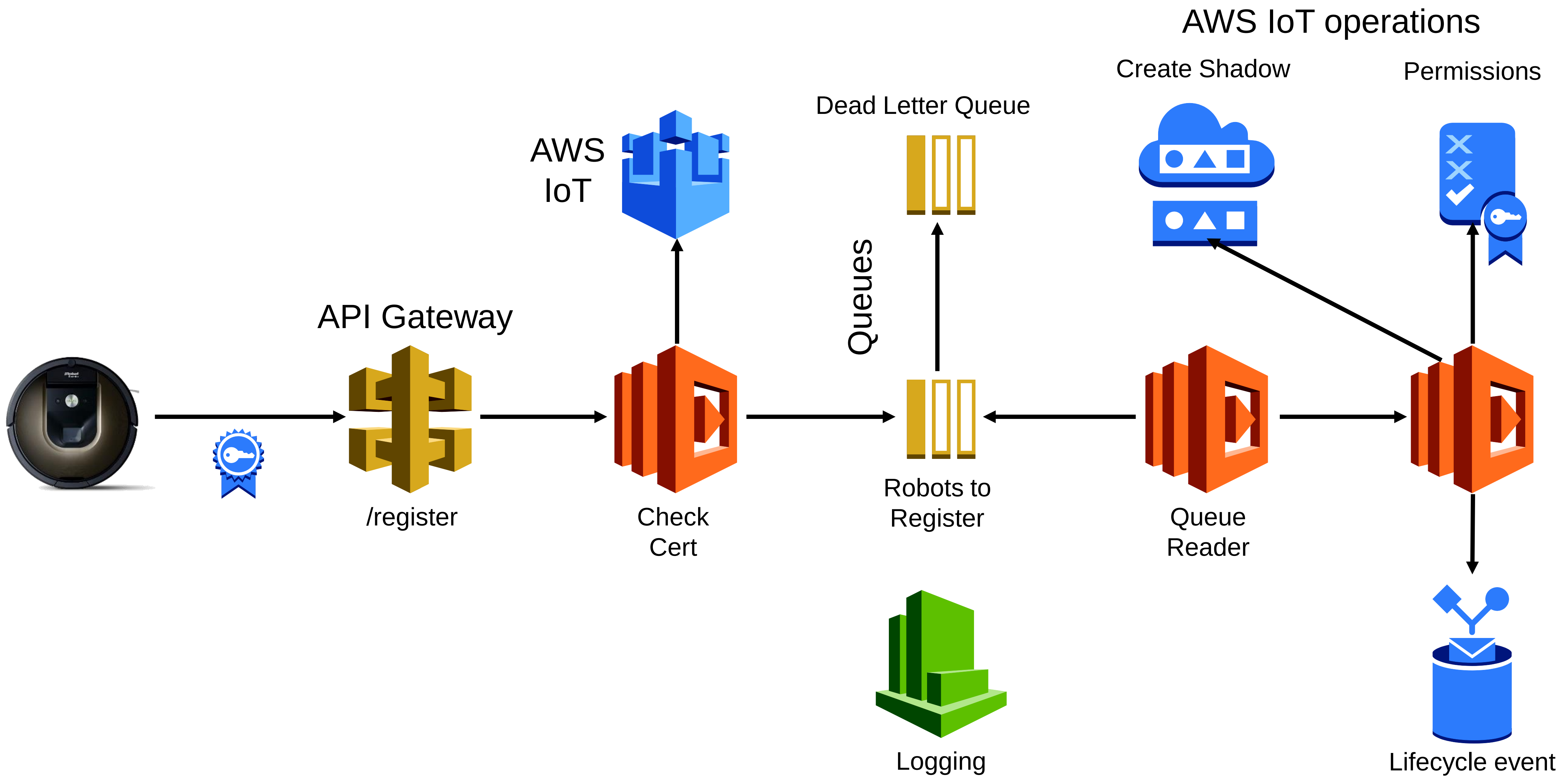
```
def foo(input):
  quux = bar(input.baz)
  internalState.quux = quux

def bar(input):
  # do work
  return result
```

EC2



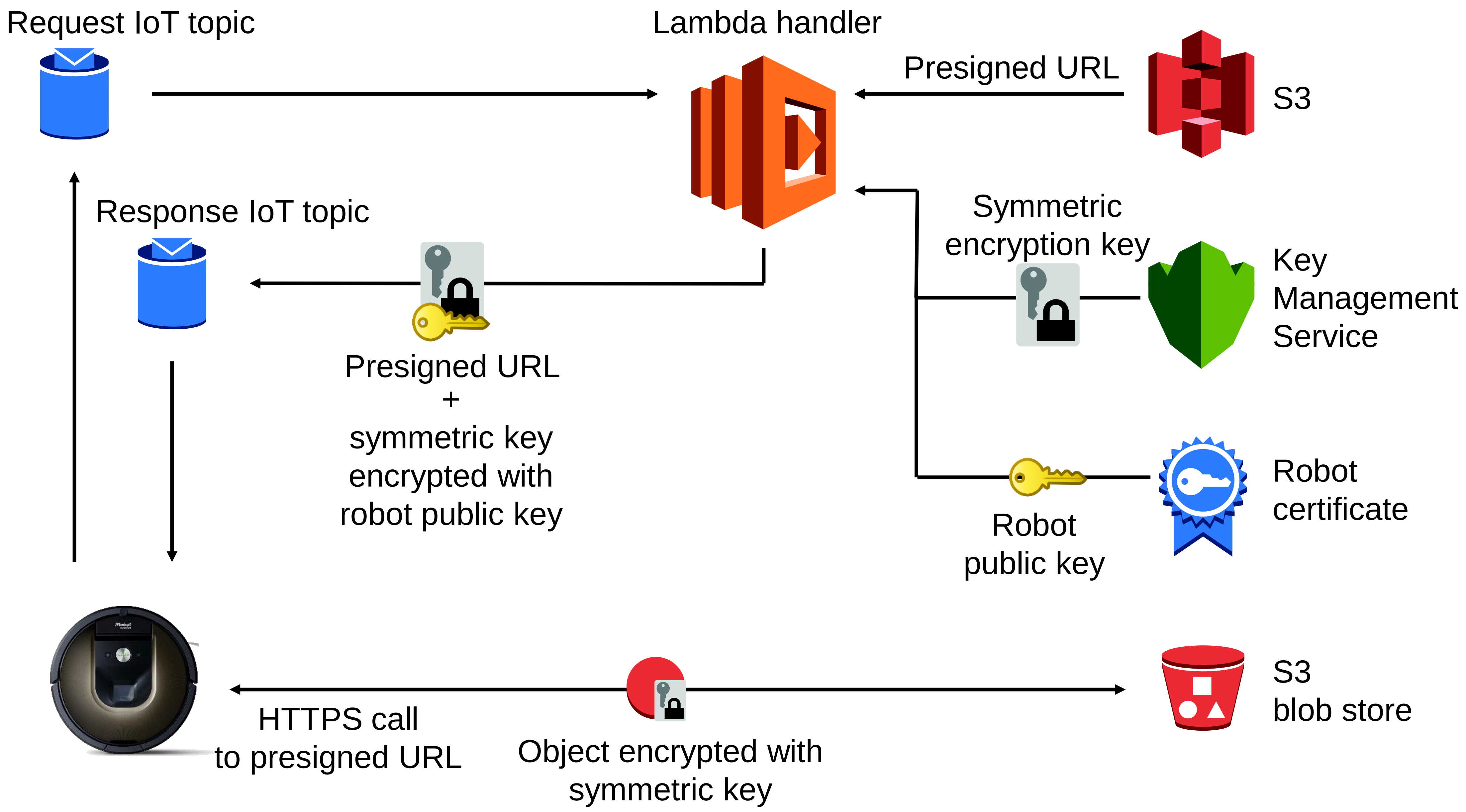
```
def handler(event, context):
  quux = Lambda.Invoke(
           'bar',
           event['baz'])
  DynamoDB.PutItem(
           'quux',
           quux)
```

```
def handler(event, context):
  # do work
  return result
```


AWS IoT operations
Create Shadow
Permissions
Dead Letter Queue
AWS IoT
Queues
API Gateway
/register
Check Cert
Robots to Register
Queue Reader
Logging
Lifecycle event

Serverless architecture

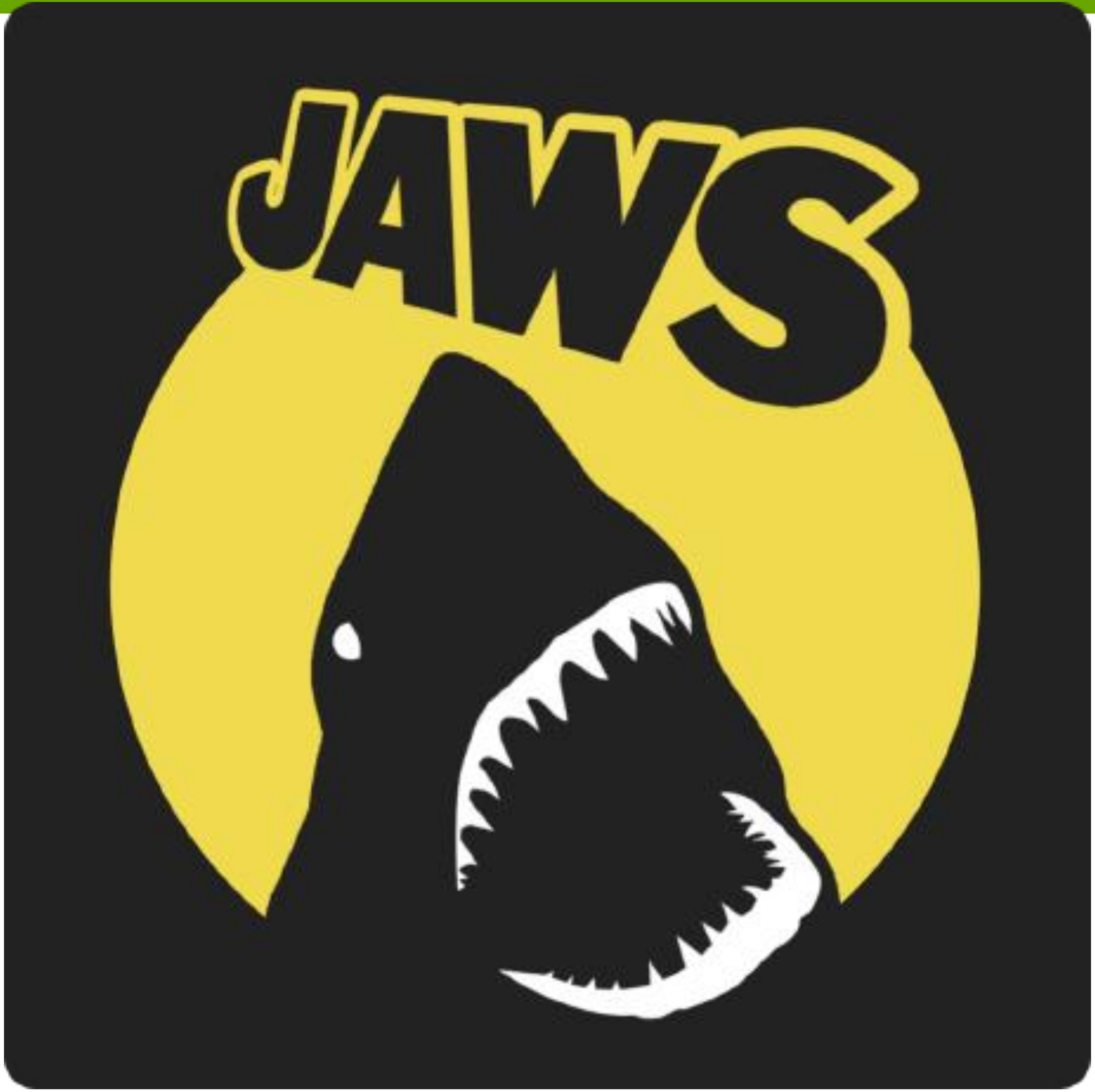
- Call graph → component graph
- Distributed system thinking
  - Traditionally occurs at system boundaries
  - Serverless: must be treated systematically
- Build robust-by-design systems
  - Better


Request IoT topic
Lambda handler
Presigned URL
S3
Response IoT topic
Symmetric encryption key
Key Management Service
Presigned URL + symmetric key encrypted with robot public key
Robot certificate
Robot public key
S3 blob store
HTTPS call to presigned URL
Object encrypted with symmetric key

3

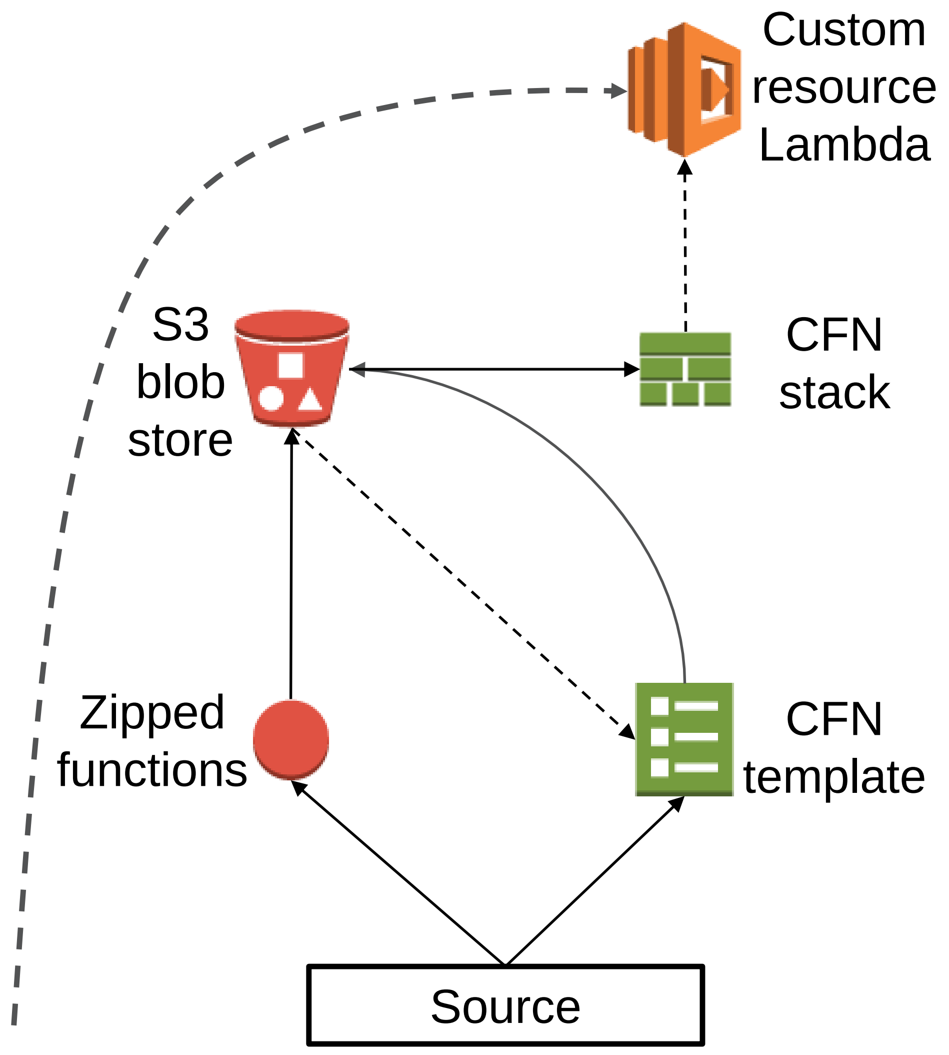
Serverless deployment at iRobot

Then 2015 Now Future

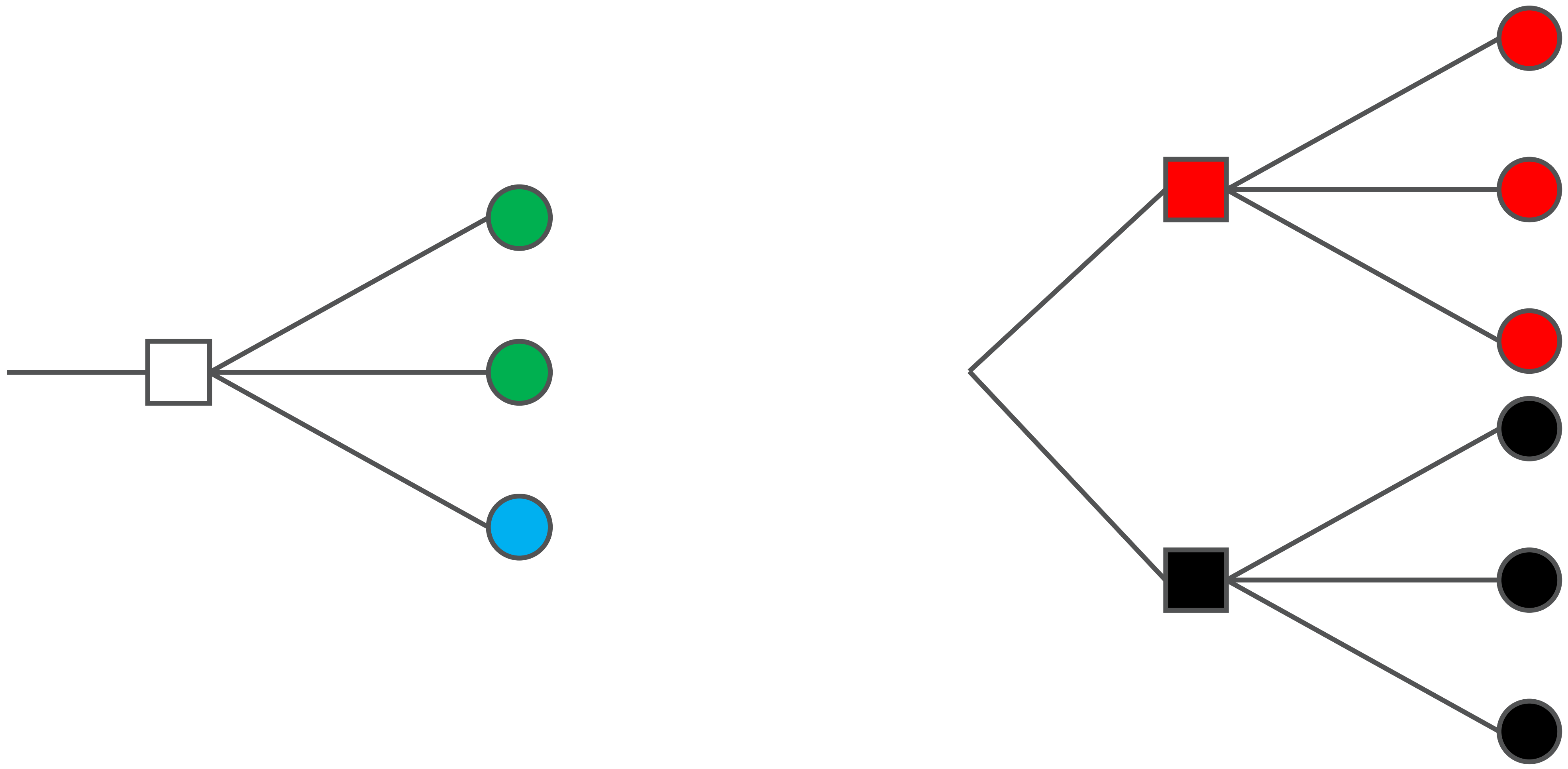

JAWS

Deployment tool: cloudr

- Collective noun for cats: "clowder"
- Designated cat herder: CloudFormation
- Custom resource Lambdas
  - Github repo: iRobotCorporation/cfnlambda



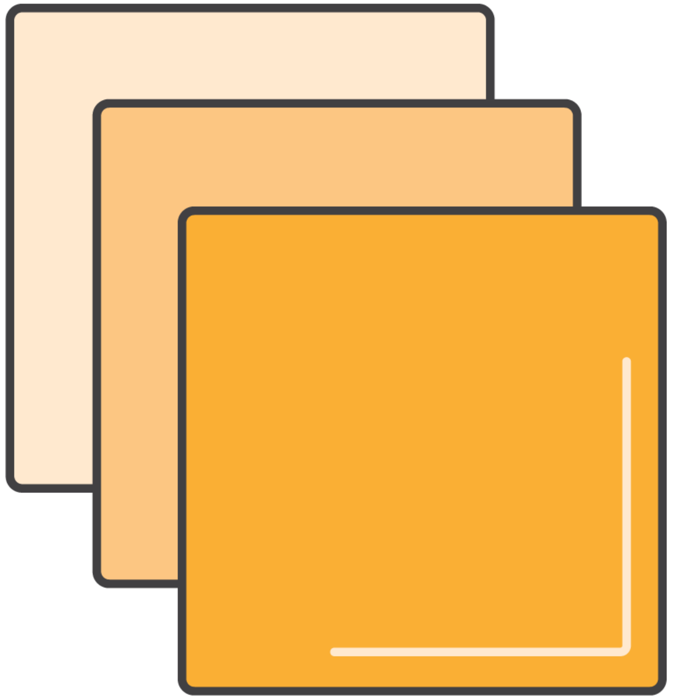
Red/Black deployments



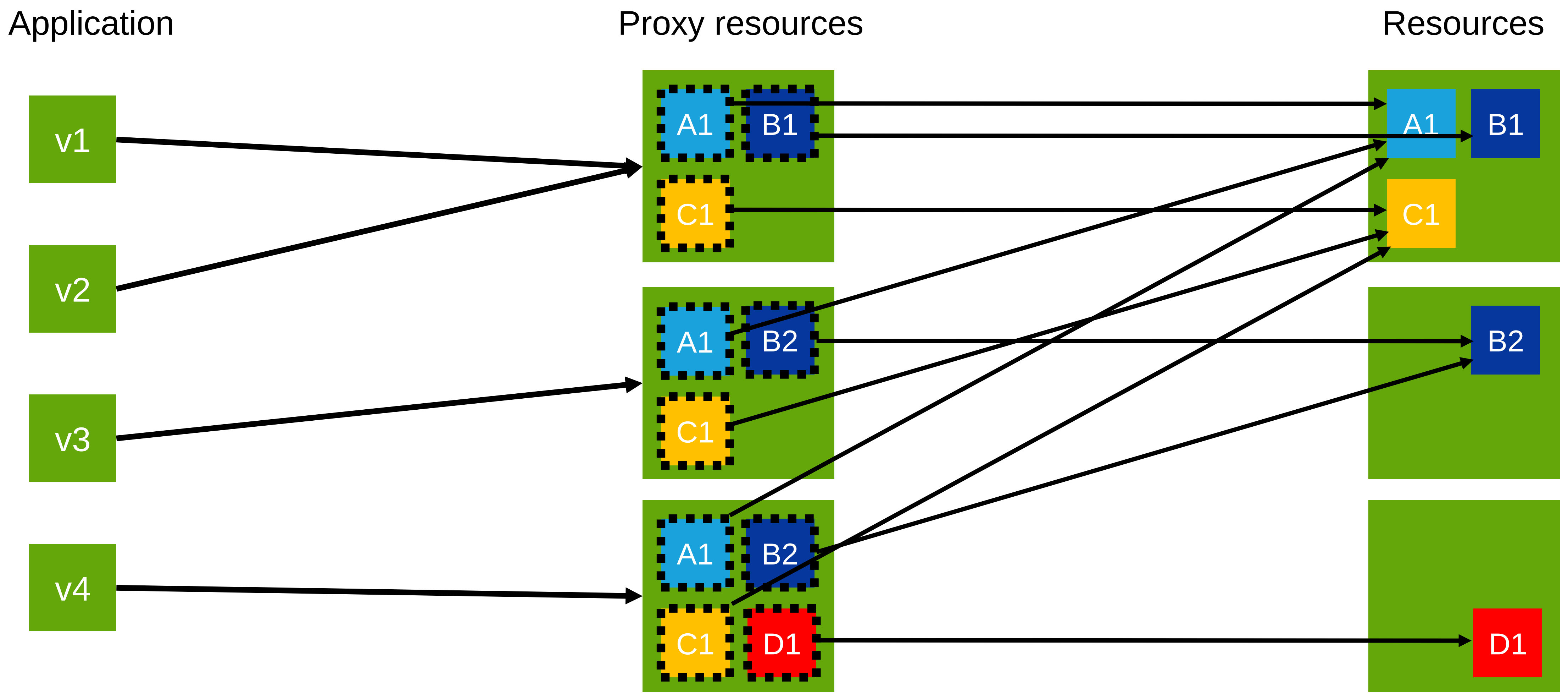
@ben11kehoe

Hosting multiple versions

- Serverless = no overhead to red/back
- IoT makes things tricky
- Data stores, etc. have separate life cycle



Deployed system architecture

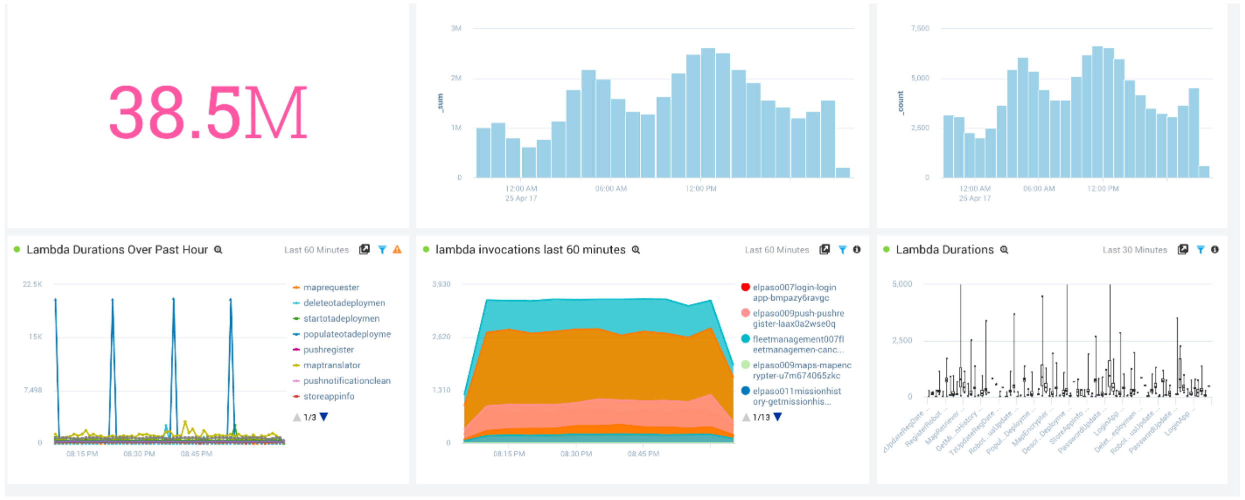


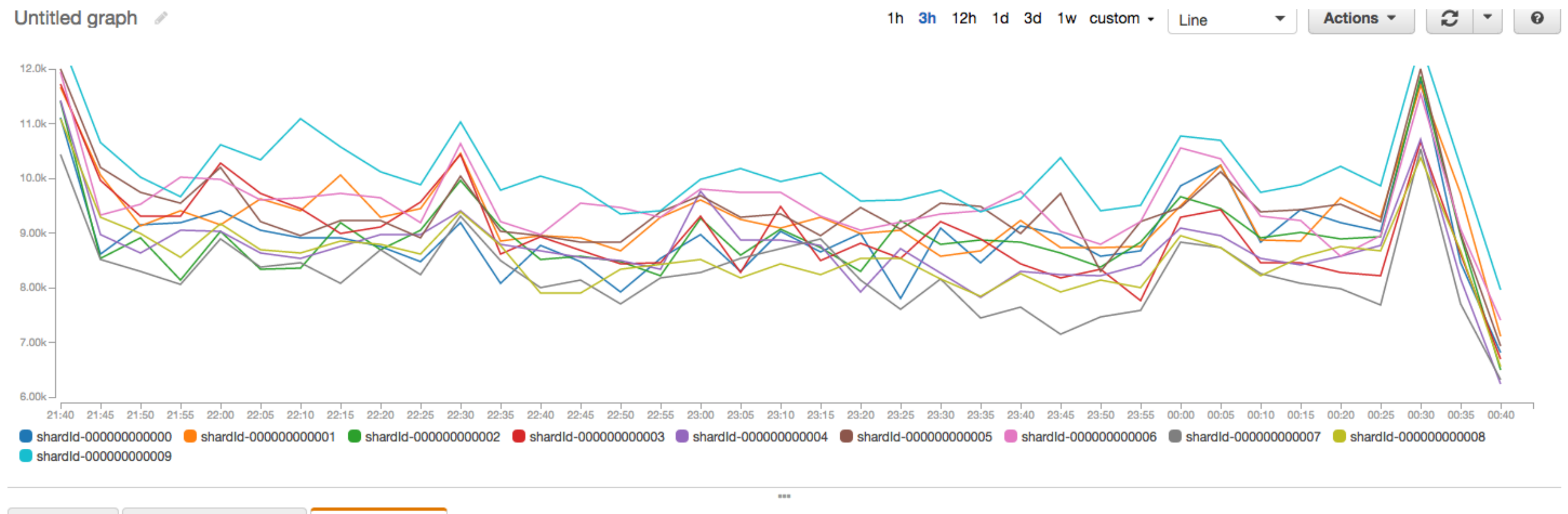
5

Serverless operations at iRobot

Monitoring

- Sumo Logic
  - Essential for log sleuthing
  - Get all data associated with an artifact immediately across all accounts
  - Provides quantitative metrics on fleet health
  - Alarms and notifications
- Extensive use of CloudWatch as well





@ben11kehoe

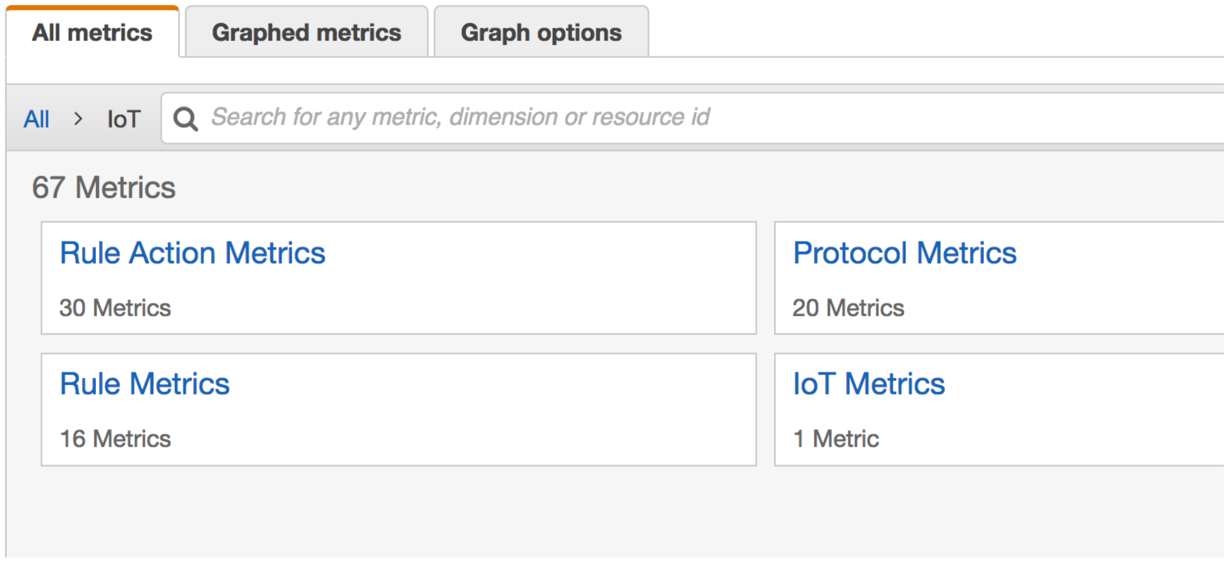
DevOps

Developers can be platform testers, canaries, and guinea pigs

- Same environment in the platform as production
- Orders of magnitude more churn
  - Exercises the account limits
  - Tests metrics to determine relevance and meaning
- Bonus – Developer activity provides additional visibility into how the platform is currently behaving
  - Higher volume of deployments in many different AWS accounts means problems found quickly
  - This can alert us prior to problems hitting prod

Visibility

- AWS IoT today has a ton of metrics
- At launch, it had <10
- Without throttling metrics, thing shadow updates, or WebSocket metrics it was hard to debug issues
  - Especially early on with small numbers of robots
  - Can I connect? How many publishes?
  - Load scale, are we over our limits?

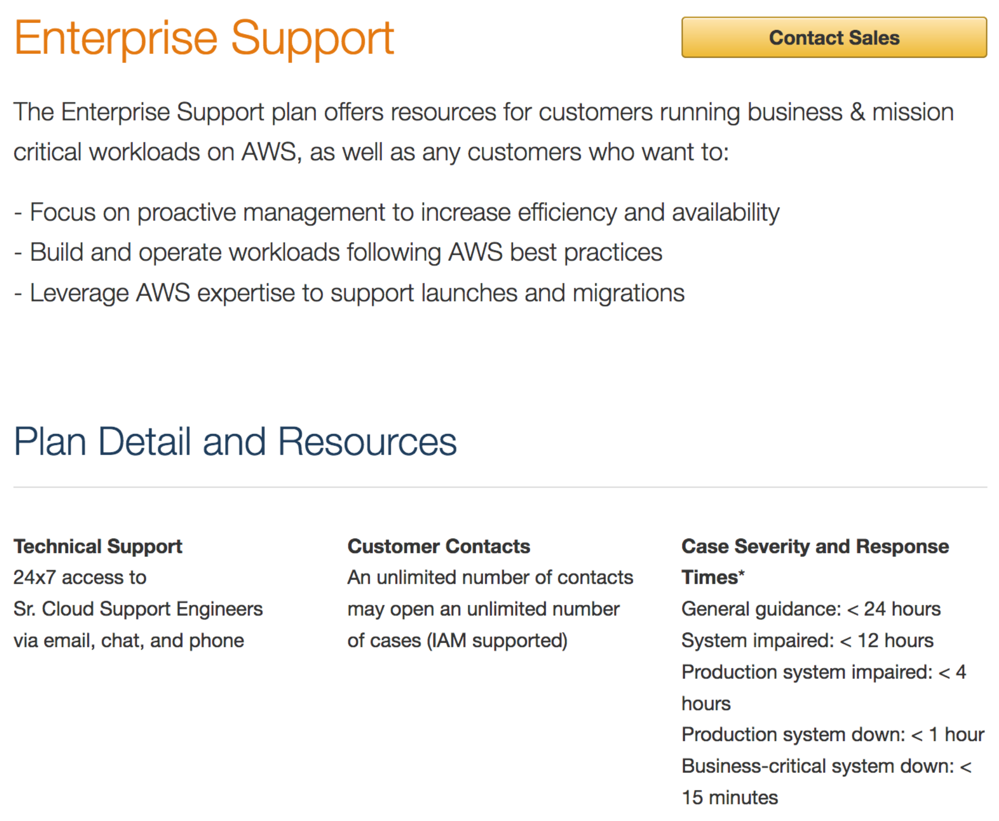


67 Metrics

Rule Action Metrics 30 Metrics	Protocol Metrics 20 Metrics
Rule Metrics 16 Metrics	IoT Metrics 1 Metric

AWS Enterprise Support

- Enterprise Support has been a valuable resource
  - They are our eyes and ears within AWS
- Engage with them to run load tests, understand account limits
- Our AWS Support team has made the effort to understand our technology choices
- All of our AWS users, company-wide, benefit from being able to create tickets



Enterprise Support

The Enterprise Support plan offers resources for customers running business & mission critical workloads on AWS, as well as any customers who want to:

- Focus on proactive management to increase efficiency and availability
- Build and operate workloads following AWS best practices
- Leverage AWS expertise to support launches and migrations

Plan Detail and Resources

Technical Support
24x7 access to
Sr. Cloud Support Engineers
via email, chat, and phone

Customer Contacts
An unlimited number of contacts may open an unlimited number of cases (IAM supported)

Case Severity and Response Times*
General guidance: < 24 hours
System impaired: < 12 hours
Production system impaired: < 4 hours
Production system down: < 1 hour
Business-critical system down: < 15 minutes

The future of improved AWS visibility

Looking toward the horizon

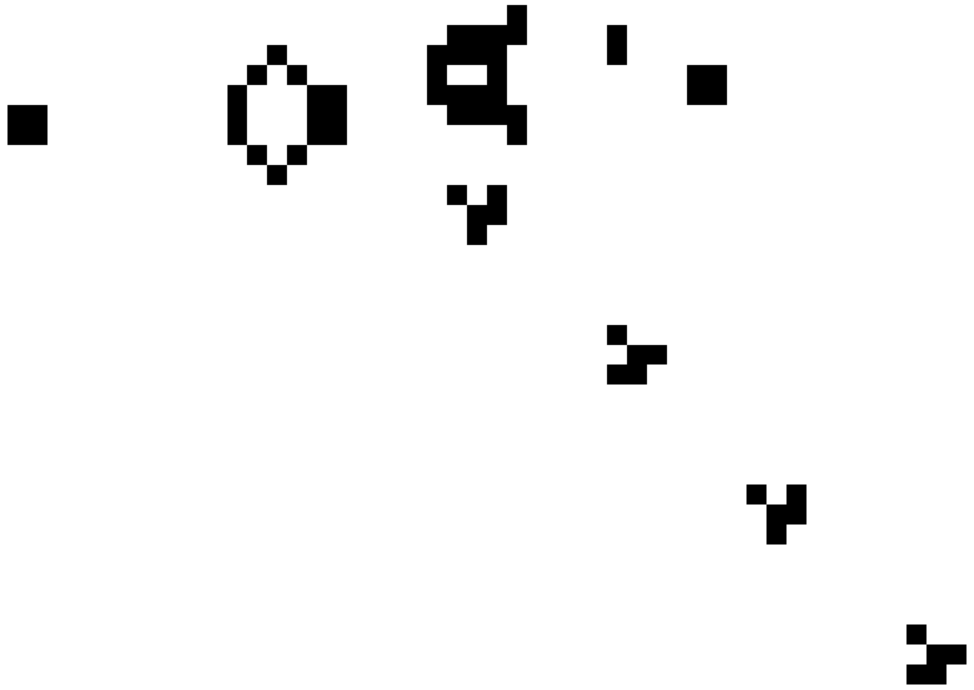
- Metrics, metrics, metrics
  - Service teams are always on the lookout for which new metrics to include – connect with them and share your requests!
  - Kinesis shard-level metrics, Lambda iterator ages, all added with user input and makes a real difference in understanding system performance
- Personal Health Dashboard
  - Per-account service health means AWS can update those affected customers more directly
  - When performance is degraded, status is important for ops to show evidence that it isn’t a problem with our software

6

Serverless organizations

Conway's Law

- Heed the warning
- Information flow is different in serverless architecture
- Organization must change for architecture to succeed


https://commons.wikimedia.org/wiki/File:Gospers_glider_gun.gif

DiffOps

- Servers → serverless is like on-prem → cloud
- Easier overall and in most respects
  - *But* also includes new challenges
- Outsourcing doesn't mean you do zero work
- Being clear about this organizationally is critical

@ben11kehoe

The cloud has weather

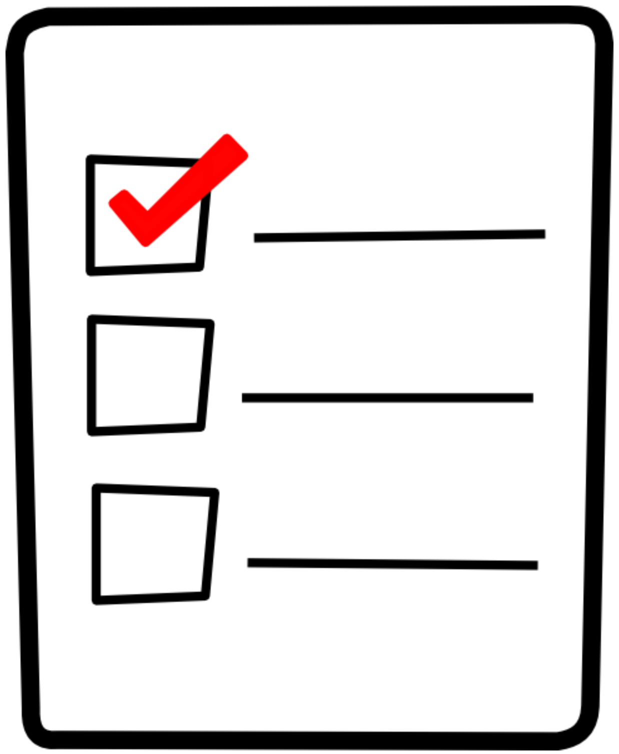
- No provider is immune to problems
- Small effects are more common than big outages
- More services = blips could be encountered more frequently
- This comes with the territory
  - Set expectations internally
  - Architecting robustly is key

Visibility

- You only know what the provider tells you
  - Architecture
  - Security
  - Operations
- How do they actually do all of the stuff they do?
- Many known unknowns and unknown unknowns
- Unknown unknown unknowns: what you don’t know *that they don’t know they don’t know*

Reacting to incidents

- First: gather data
- Root cause: our code or platform?
- Own the impact to your customers
- Diagnose your applications' handling of incident
  - Live and postmortem
- Aftermath



7

Summing up

Serverless

Serverless is:

- Cheaper
- Faster
- Leaner
- Better

Serverless-ness goes with:

- Service-full + emphemeral compute
- Resources billed → resources used
- Smaller, more abstract control plane

Serverless at iRobot

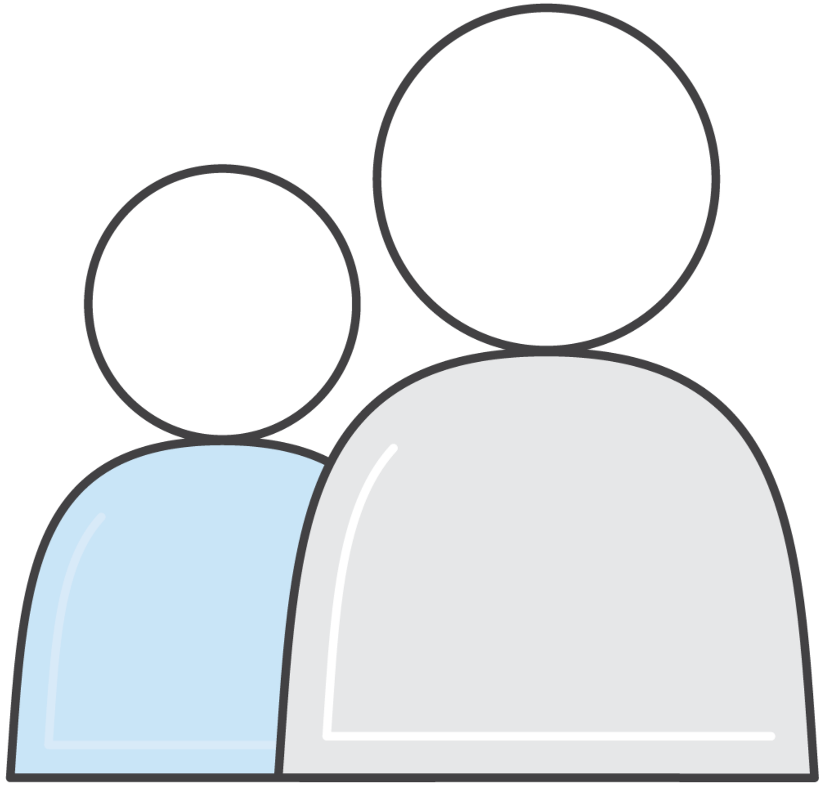
- Successfully transitioned from turnkey to application built on public cloud
- Skipped learning to build elastic cloud infrastructure
- Fully serverless production application

Lessons learned

- Serverless deployment is still not a solved problem
- Call graph → component graph
- Visibility is the biggest operations obstacle

Serverless organizations

- Conway's Law
- Cloud has weather
- Set expectations
- Focus on TCO



Questions?