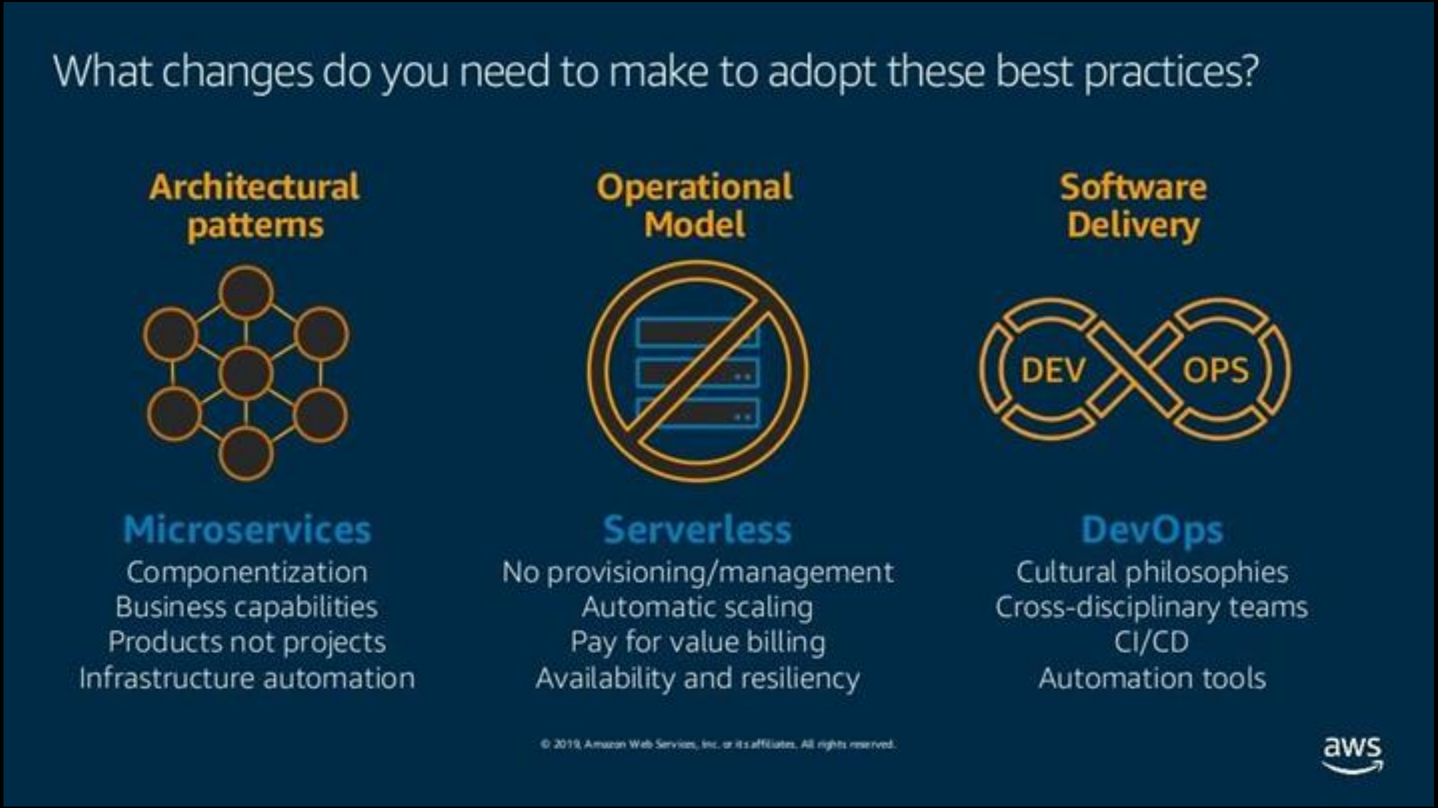
Cloud Services 2

Amazon Web Services

Observability



What changes do you need to make to adopt these best practices?

Architectural patterns

Microservices

Componentization
Business capabilities
Products not projects
Infrastructure automation

Operational Model

Serverless

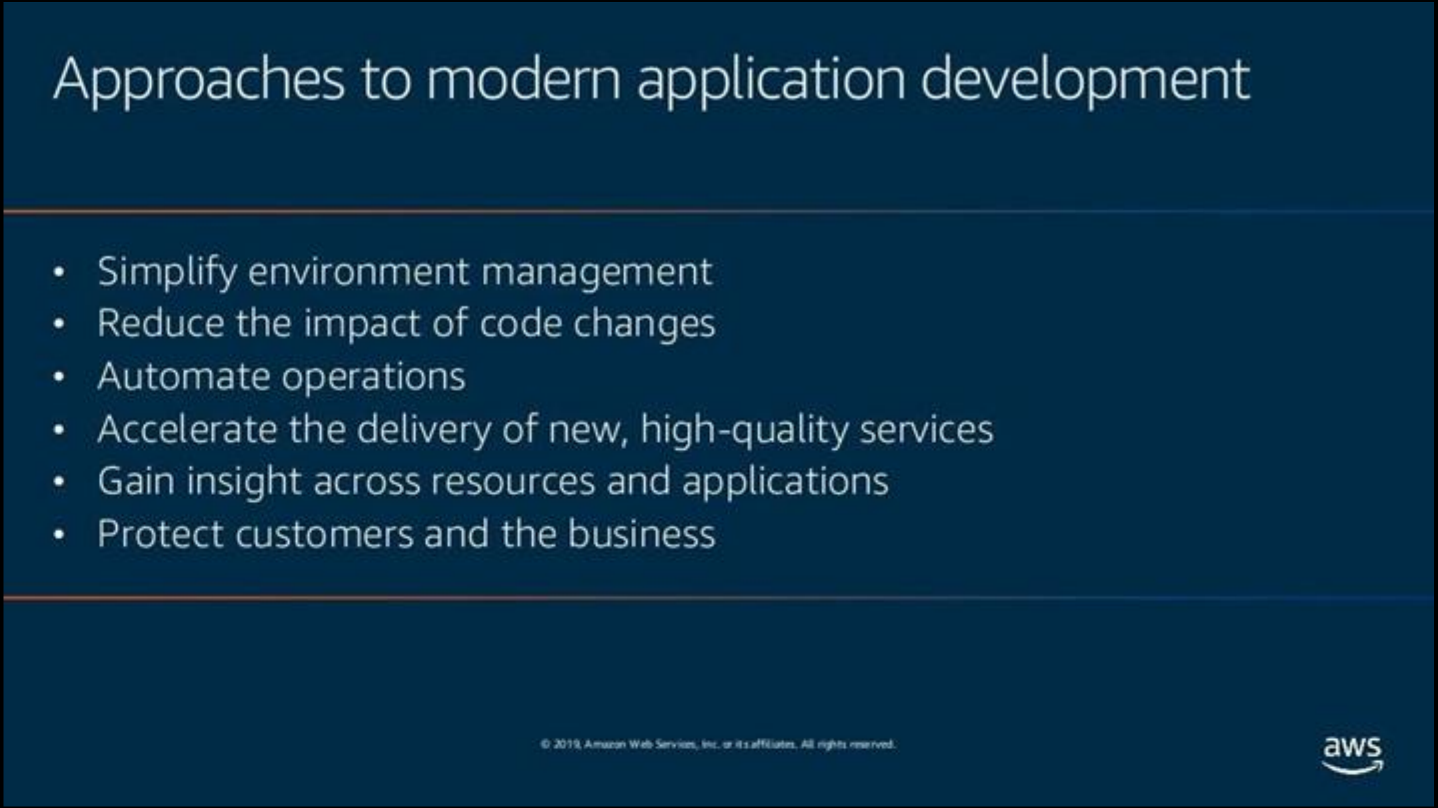
No provisioning/management
Automatic scaling
Pay for value billing
Availability and resiliency

Software Delivery

DevOps

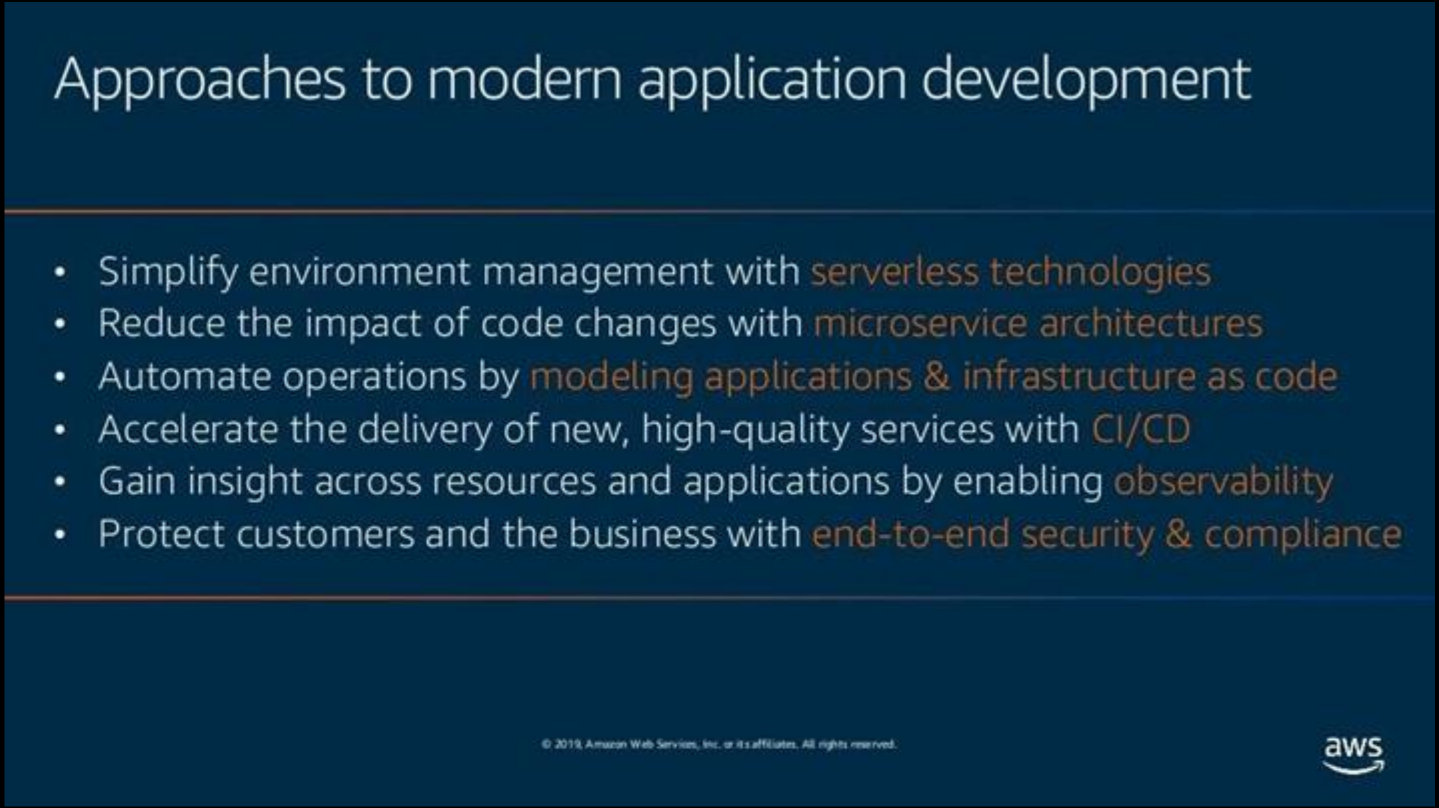
Cultural philosophies
Cross-disciplinary teams
CI/CD
Automation tools

aws



Approaches to modern application development

- Simplify environment management
- Reduce the impact of code changes
- Automate operations
- Accelerate the delivery of new, high-quality services
- Gain insight across resources and applications
- Protect customers and the business

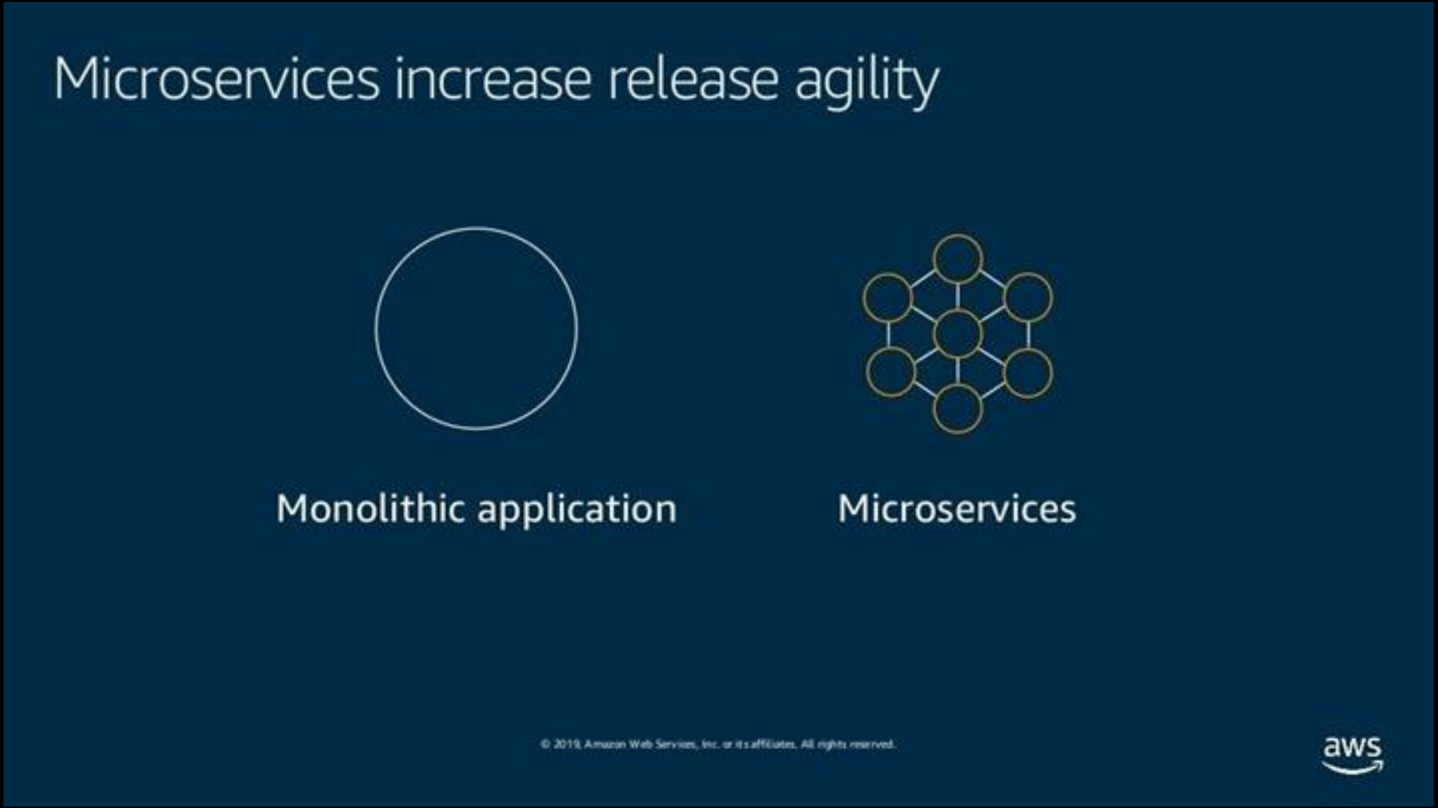


Approaches to modern application development

- Simplify environment management with serverless technologies
- Reduce the impact of code changes with microservice architectures
- Automate operations by modeling applications & infrastructure as code
- Accelerate the delivery of new, high-quality services with CI/CD
- Gain insight across resources and applications by enabling observability
- Protect customers and the business with end-to-end security & compliance

Approaches to modern application development

- Simplify environment management with serverless technologies
- Reduce the impact of code changes with microservice architectures
- Automate operations by modeling applications & infrastructure as code
- Accelerate the delivery of new, high-quality services with CI/CD
- Gain insight across resources and applications by enabling observability
- Protect customers and the business with end-to-end security & compliance



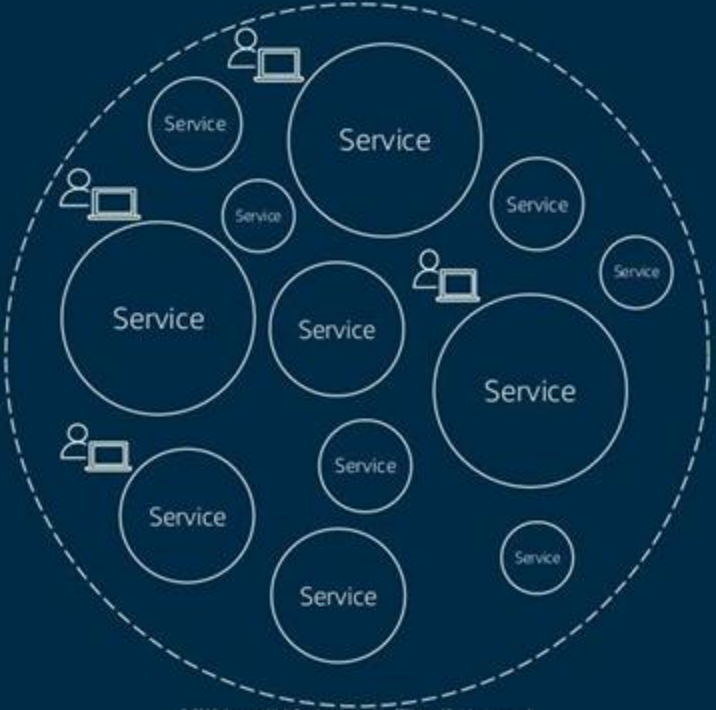
Microservices increase release agility

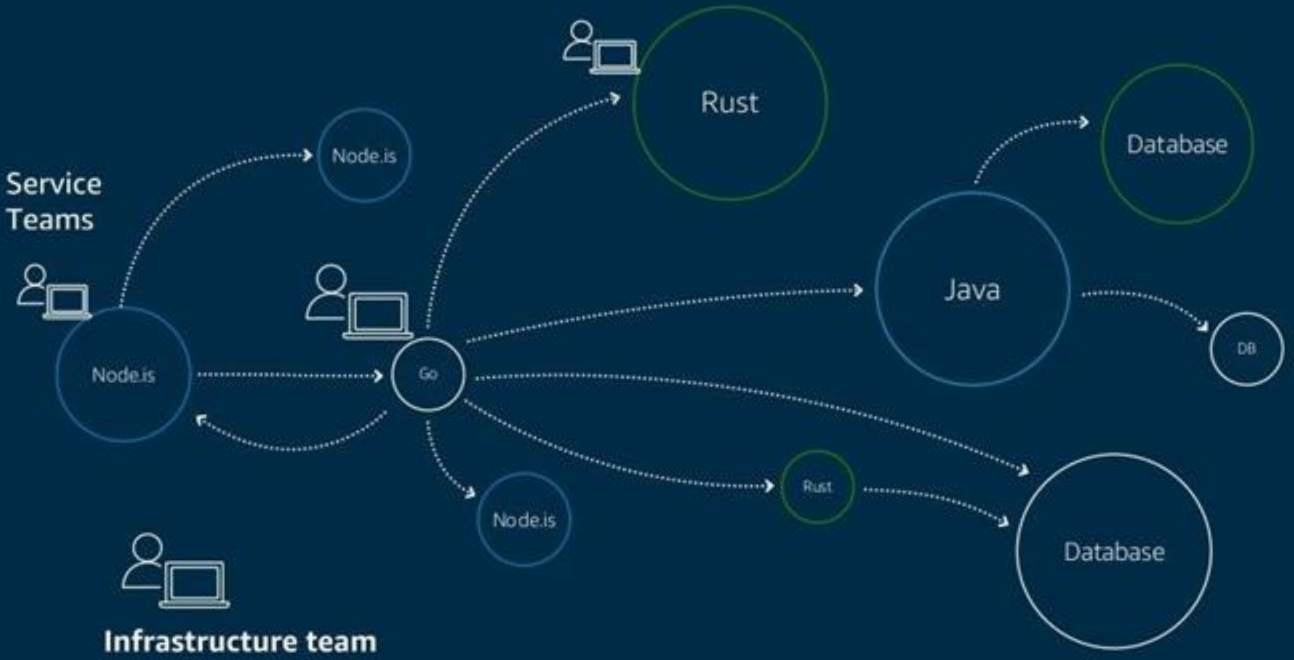
Monolithic application

Microservices

Monolith

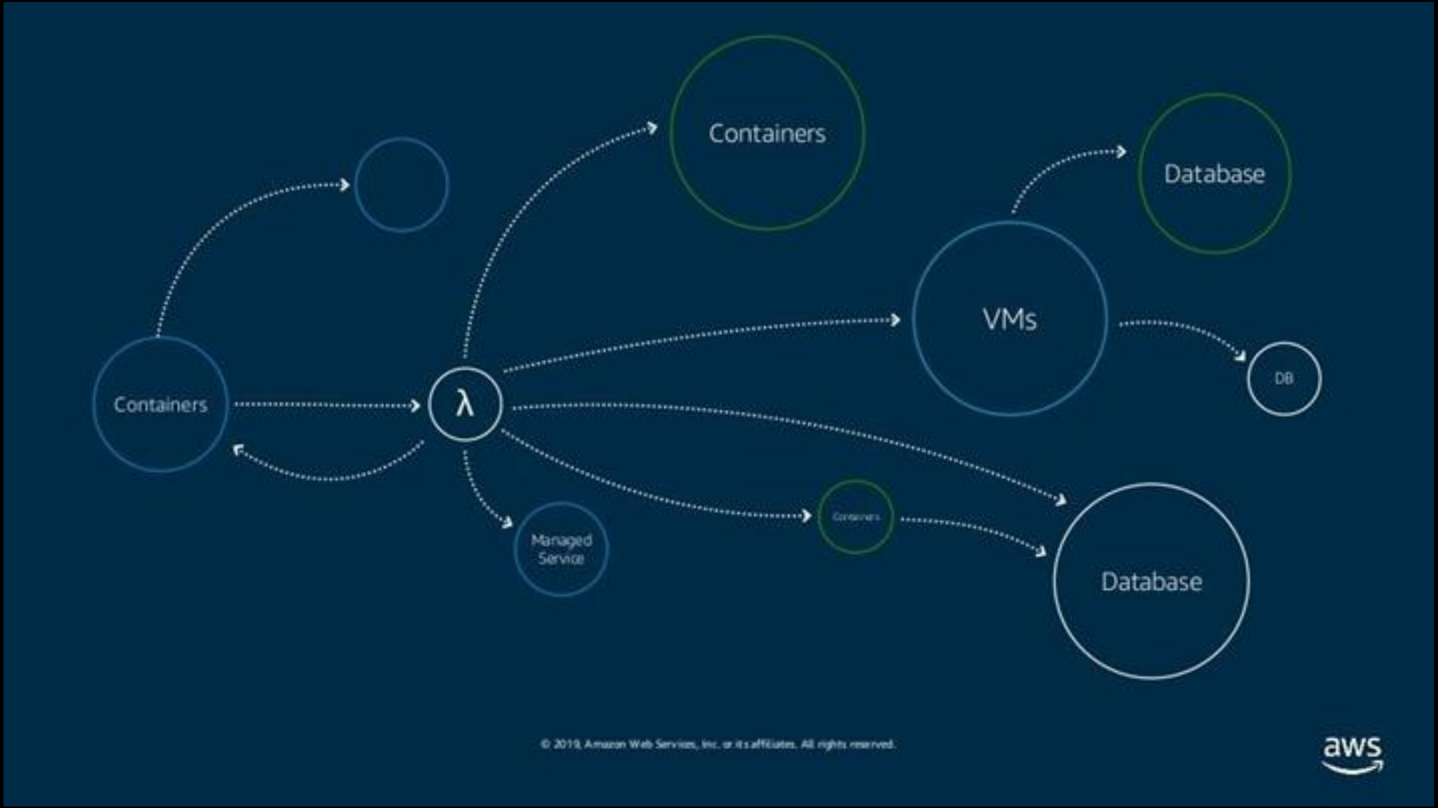
aws

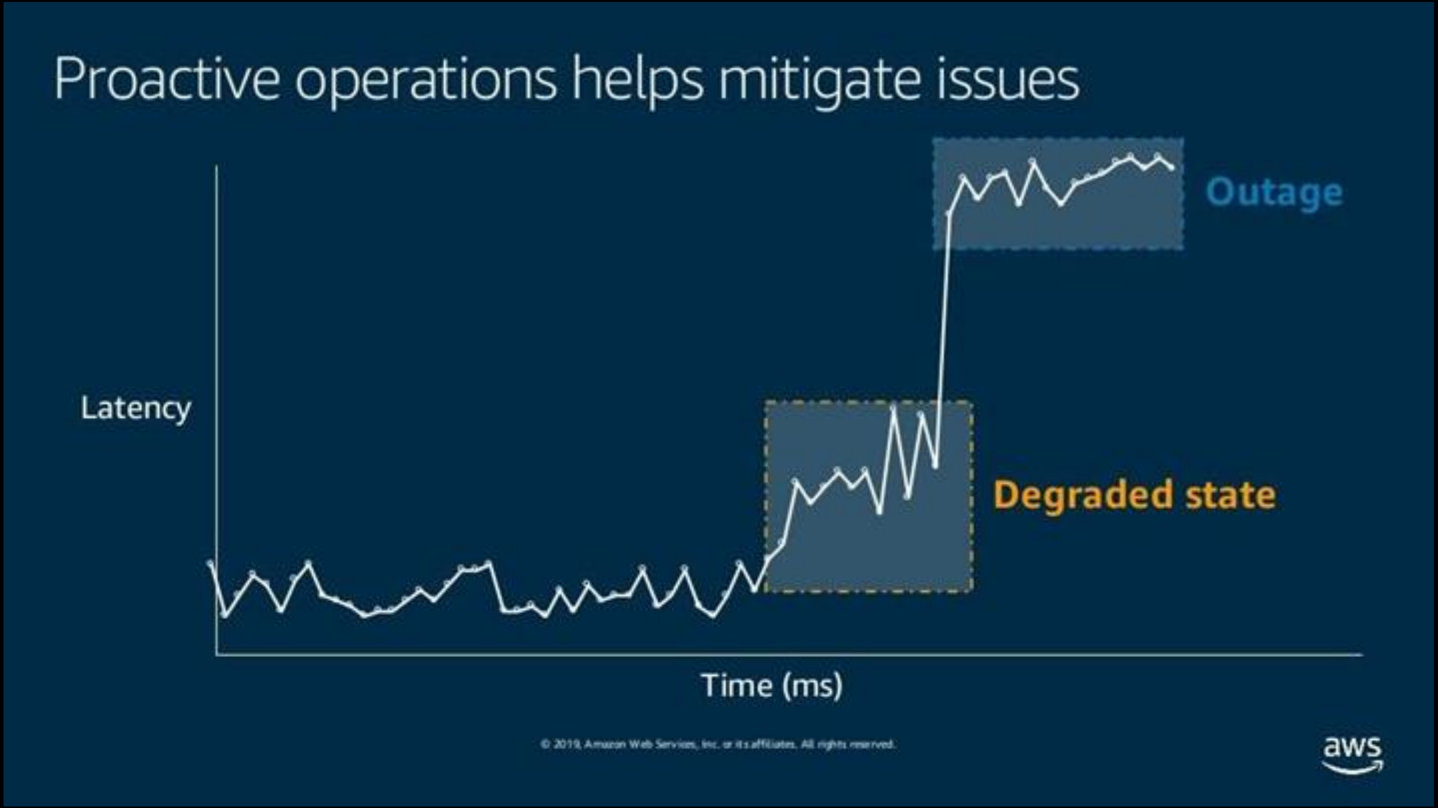

Service
Service
Service
Service
Service
Service
Service
Service
Service
Service
Service
Service



Service Teams

Infrastructure team


Containers
Database
VMs
DB
Containers
λ
Managed Service
Containers
Database
aws



Proactive operations helps mitigate issues

Observability in Control Theory

1961-62

On the General Theory of Control Systems

R. E. KALMAN

Introduction

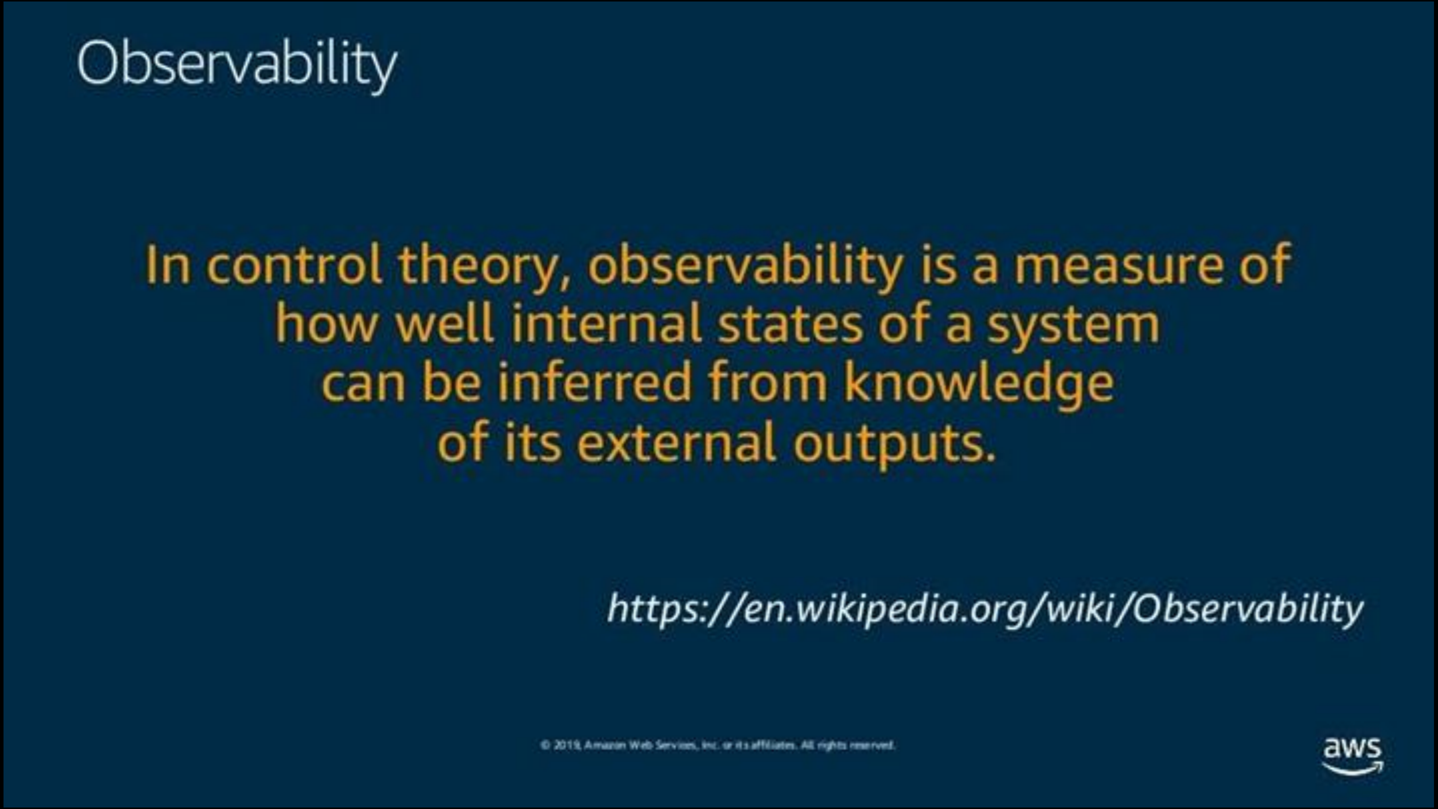
[illegible]

MATHEMATICAL DESCRIPTION OF LINEAR DYNAMICAL SYSTEMS*

R. E. KALMAN†

Abstract. [illegible]

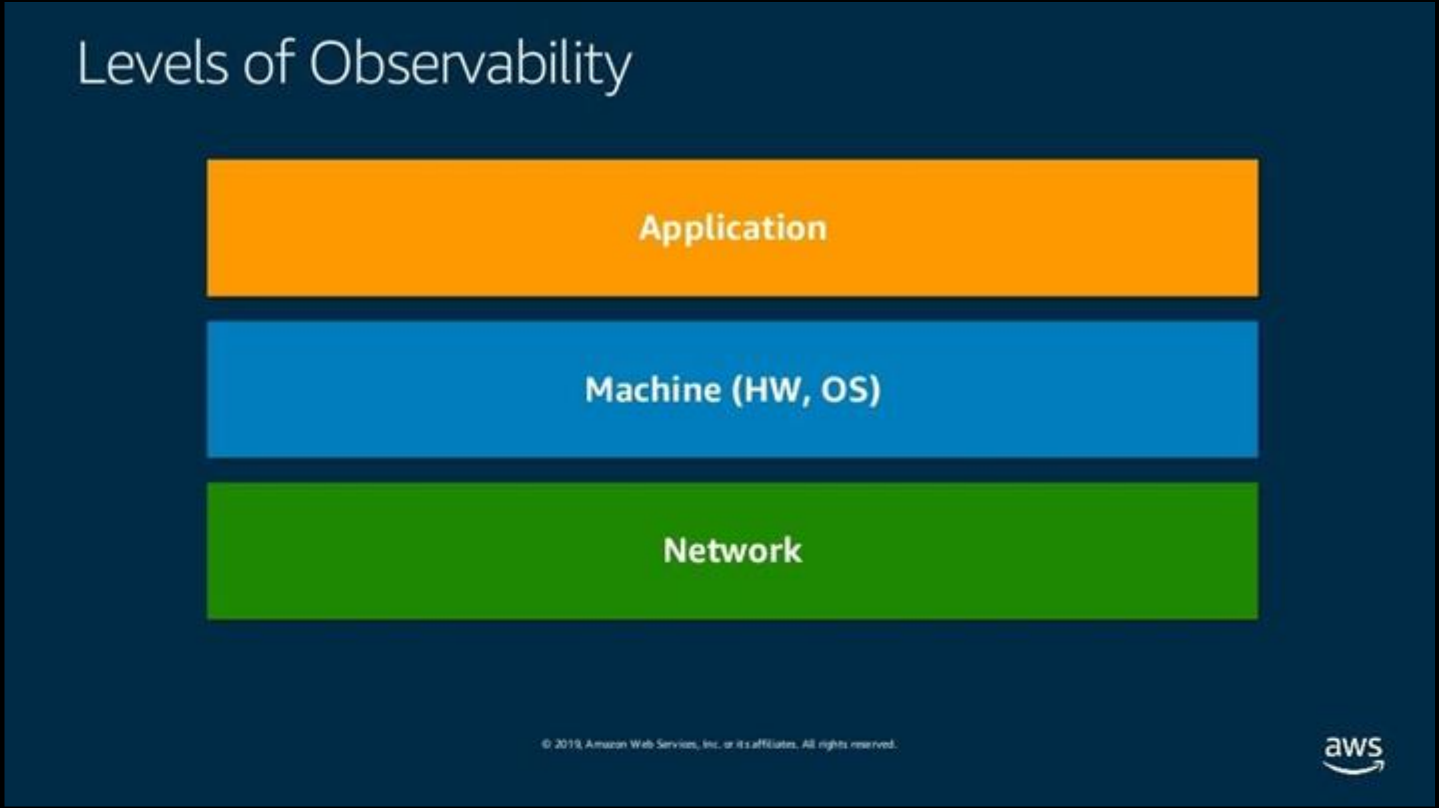
1. Introduction and summary. [illegible]



Observability

In control theory, observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs.

https://en.wikipedia.org/wiki/Observability



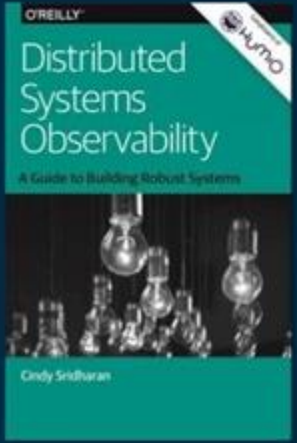
Levels of Observability

Application

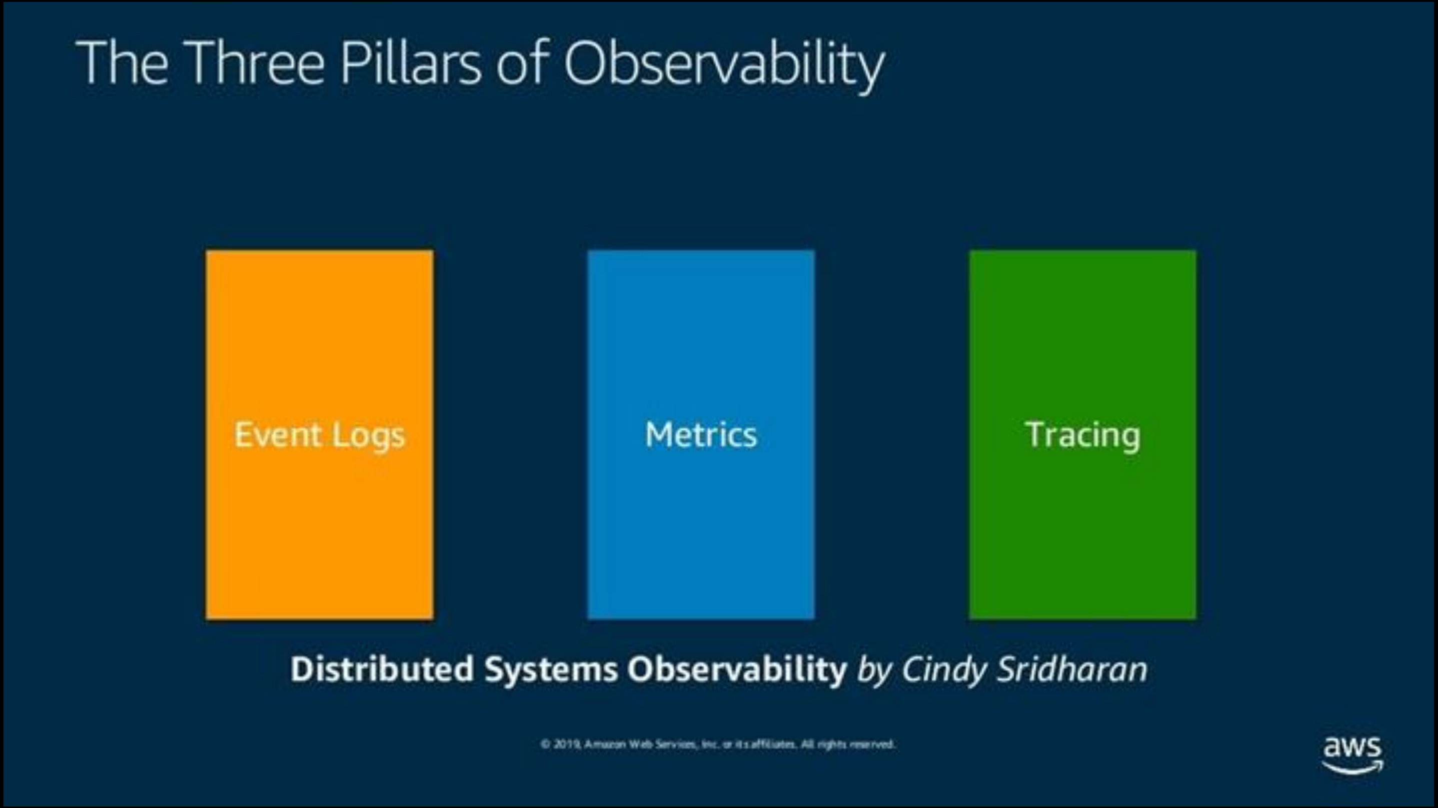
Machine (HW, OS)

Network

The Three Pillars of Observability


O'REILLY
Distributed Systems Observability
A Guide to Building Robust Systems
Cindy Sridharan

Distributed Systems Observability *by Cindy Sridharan*



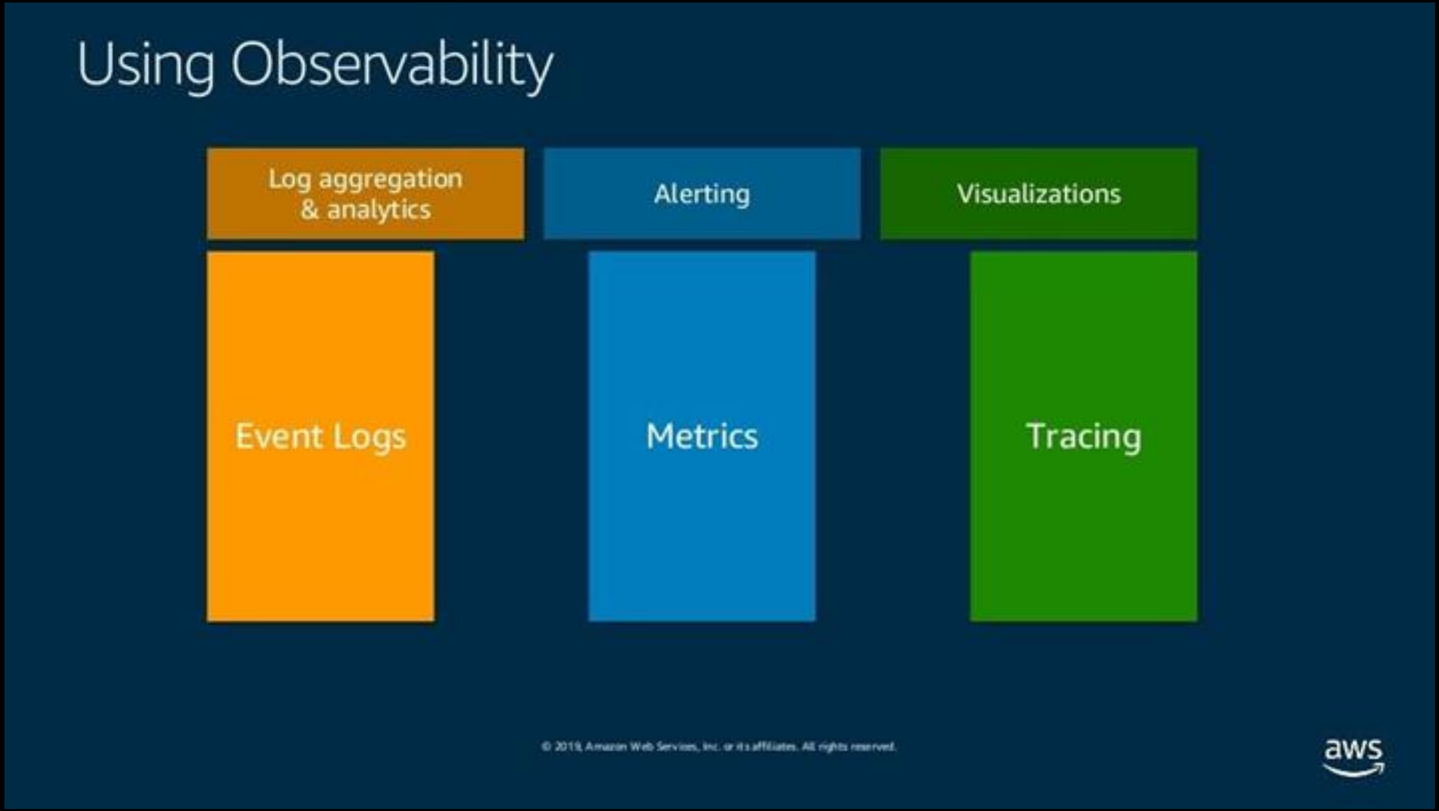
The Three Pillars of Observability

Event Logs

Metrics

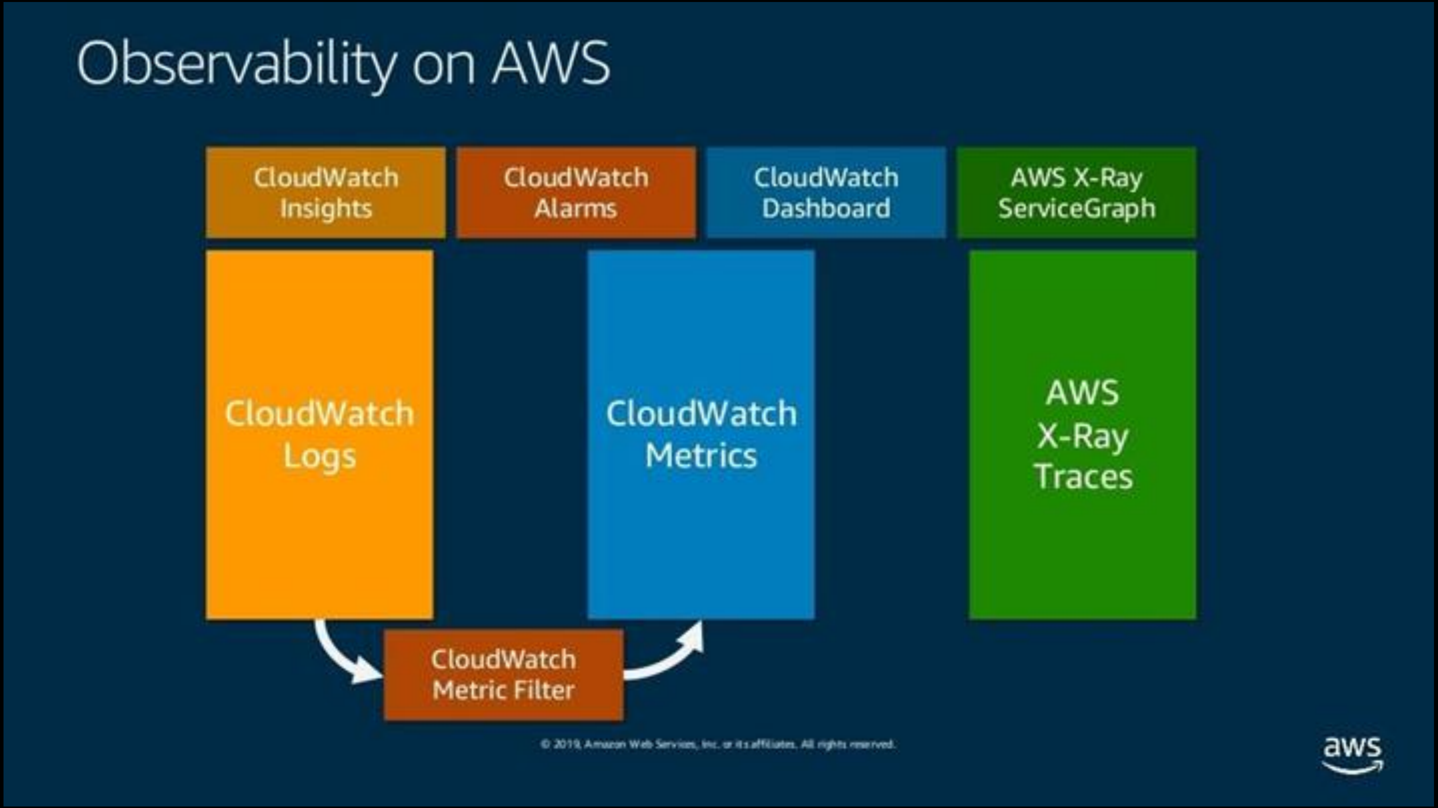
Tracing

Distributed Systems Observability *by Cindy Sridharan*



Using Observability

Observability on AWS



CloudWatch Insights

CloudWatch Alarms

CloudWatch Dashboard

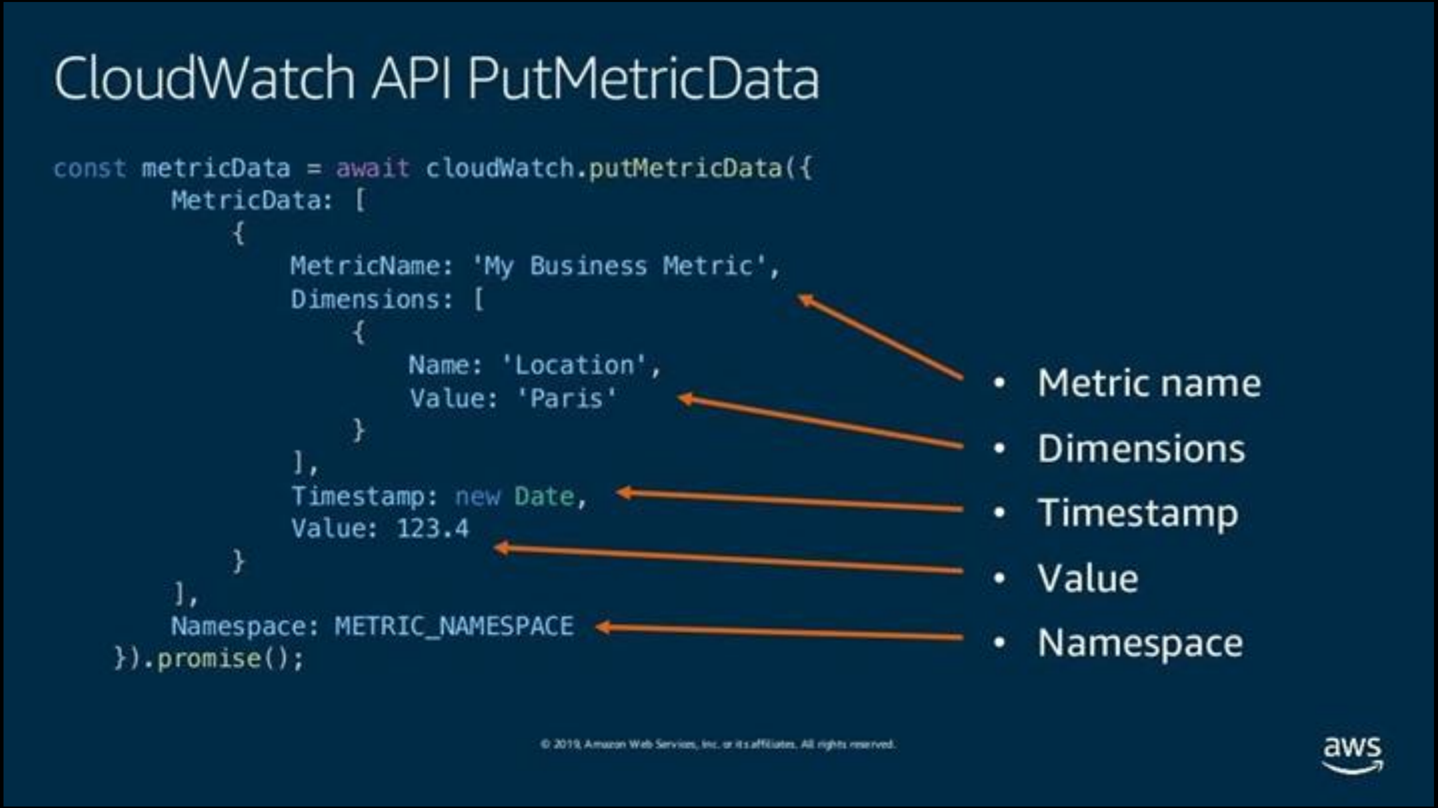
AWS X-Ray ServiceGraph

CloudWatch Logs

CloudWatch Metrics

AWS X-Ray Traces

CloudWatch Metric Filter

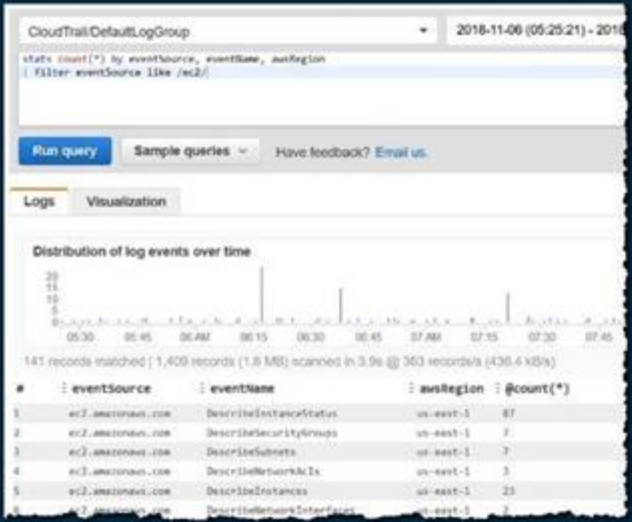


CloudWatch API PutMetricData

```
const metricData = await cloudWatch.putMetricData({
        MetricData: [
            {
                MetricName: 'My Business Metric',
                Dimensions: [
                    {
                        Name: 'Location',
                        Value: 'Paris'
                    }
                ],
                Timestamp: new Date,
                Value: 123.4
            }
        ],
        Namespace: METRIC_NAMESPACE
    }).promise();
```

- Metric name
- Dimensions
- Timestamp
- Value
- Namespace

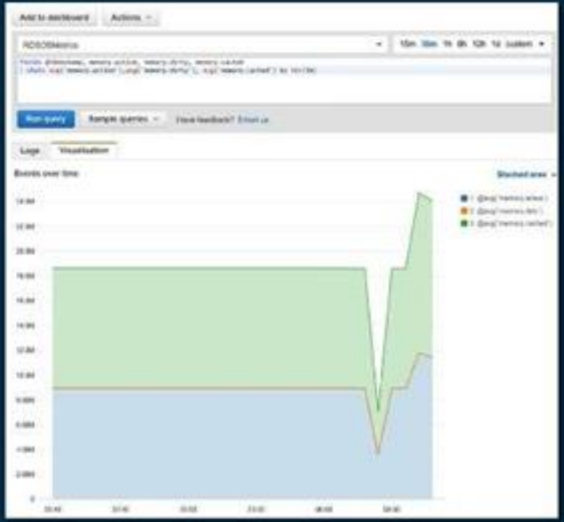
Add correlation IDs to logs – CloudWatch Logs + Insights



```
stats count(*) by eventSource, eventName, awsRegion
| filter eventSource like /ec2/
```

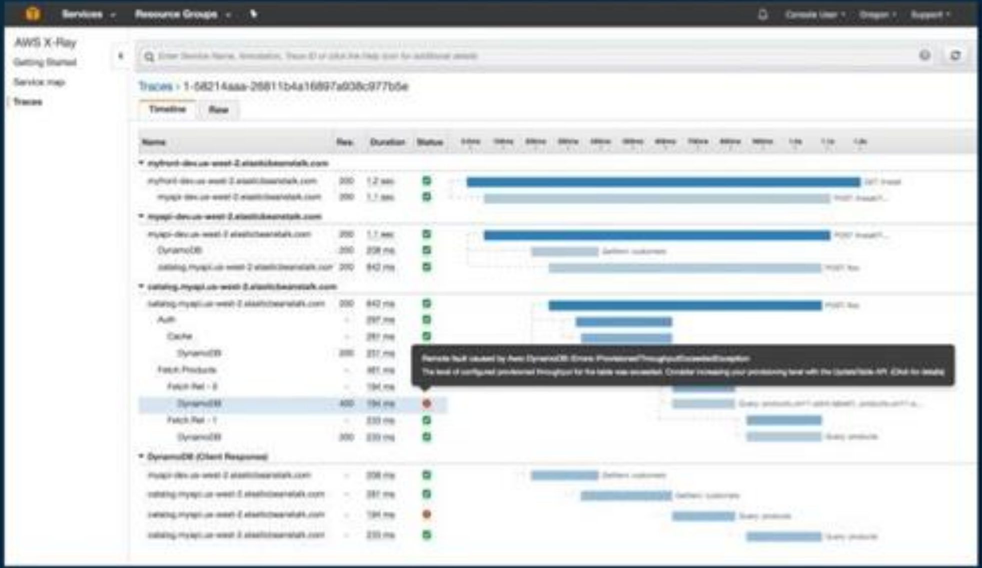
141 records matched | 1,409 records (1.8 MB) scanned in 3.9s @ 363 records/s (436.4 kB/s)

#	eventSource	eventName	awsRegion	@count(*)
1	ec2.amazonaws.com	DescribeInstanceStatus	us-east-1	87
2	ec2.amazonaws.com	DescribeSecurityGroups	us-east-1	7
3	ec2.amazonaws.com	DescribeSubnets	us-east-1	7
4	ec2.amazonaws.com	DescribeNetworkAcls	us-east-1	3
5	ec2.amazonaws.com	DescribeInstances	us-east-1	23
6	ec2.amazonaws.com	DescribeNetworkInterfaces	us-east-1	2



```
fields @timestamp, memory.active, memory.dirty, memory.cached
| stats avg(`memory.active`), avg(`memory.dirty`), avg(`memory.cached`) by bin(5m)
```

End-to-end tracing – AWS X-Ray Traces



Traces › 1-58214aaa-26811b4a16897a938c977b5e

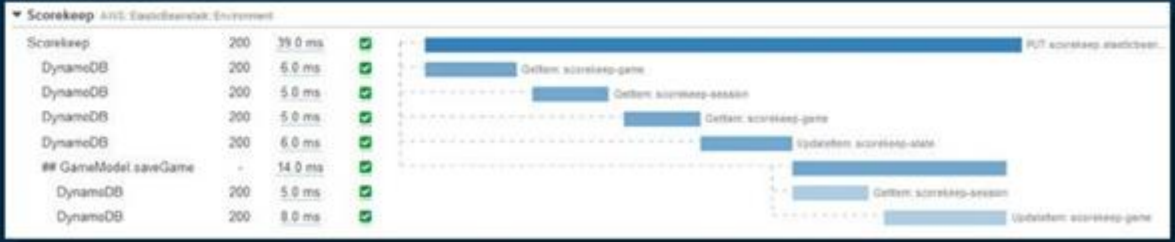
Name	Res.	Duration	Status
▾ myfront-dev.us-west-2.elasticbeanstalk.com			
myfront-dev.us-west-2.elasticbeanstalk.com	200	1.2 sec	✓
myapi-dev.us-west-2.elasticbeanstalk.com	200	1.1 sec	✓
▾ myapi-dev.us-west-2.elasticbeanstalk.com			
myapi-dev.us-west-2.elasticbeanstalk.com	200	1.1 sec	✓
DynamoDB	200	208 ms	✓
catalog.myapi.us-west-2.elasticbeanstalk.com	200	842 ms	✓
▾ catalog.myapi.us-west-2.elasticbeanstalk.com			
catalog.myapi.us-west-2.elasticbeanstalk.com	200	842 ms	✓
Auth	-	297 ms	✓
Cache	-	281 ms	✓
DynamoDB	200	237 ms	
Fetch Products	-	481 ms	
Fetch Rel - 0	-	194 ms	
DynamoDB	400	194 ms	●
Fetch Rel - 1	-	233 ms	✓
DynamoDB	200	233 ms	✓
▾ DynamoDB (Client Response)			
myapi-dev.us-west-2.elasticbeanstalk.com	-	208 ms	✓
catalog.myapi.us-west-2.elasticbeanstalk.com	-	281 ms	✓
catalog.myapi.us-west-2.elasticbeanstalk.com	-	194 ms	●
catalog.myapi.us-west-2.elasticbeanstalk.com	-	233 ms	✓

Remote fault caused by Aws::DynamoDB::Errors::ProvisionedThroughputExceededException

The level of configured provisioned throughput for the table was exceeded. Consider increasing your provisioning level with the UpdateTable API. (Click for details)

AWS X-Ray Key Concepts

Segments

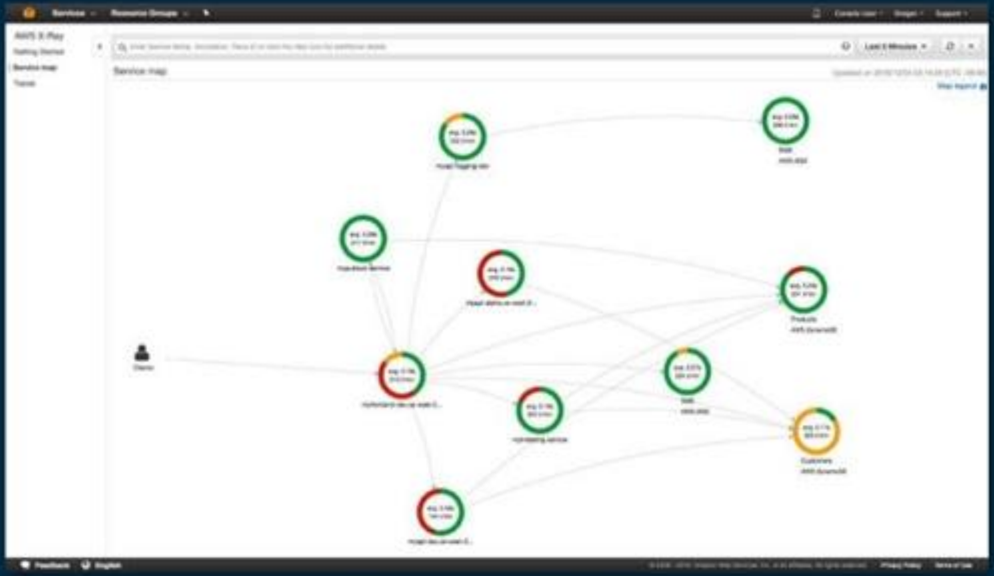


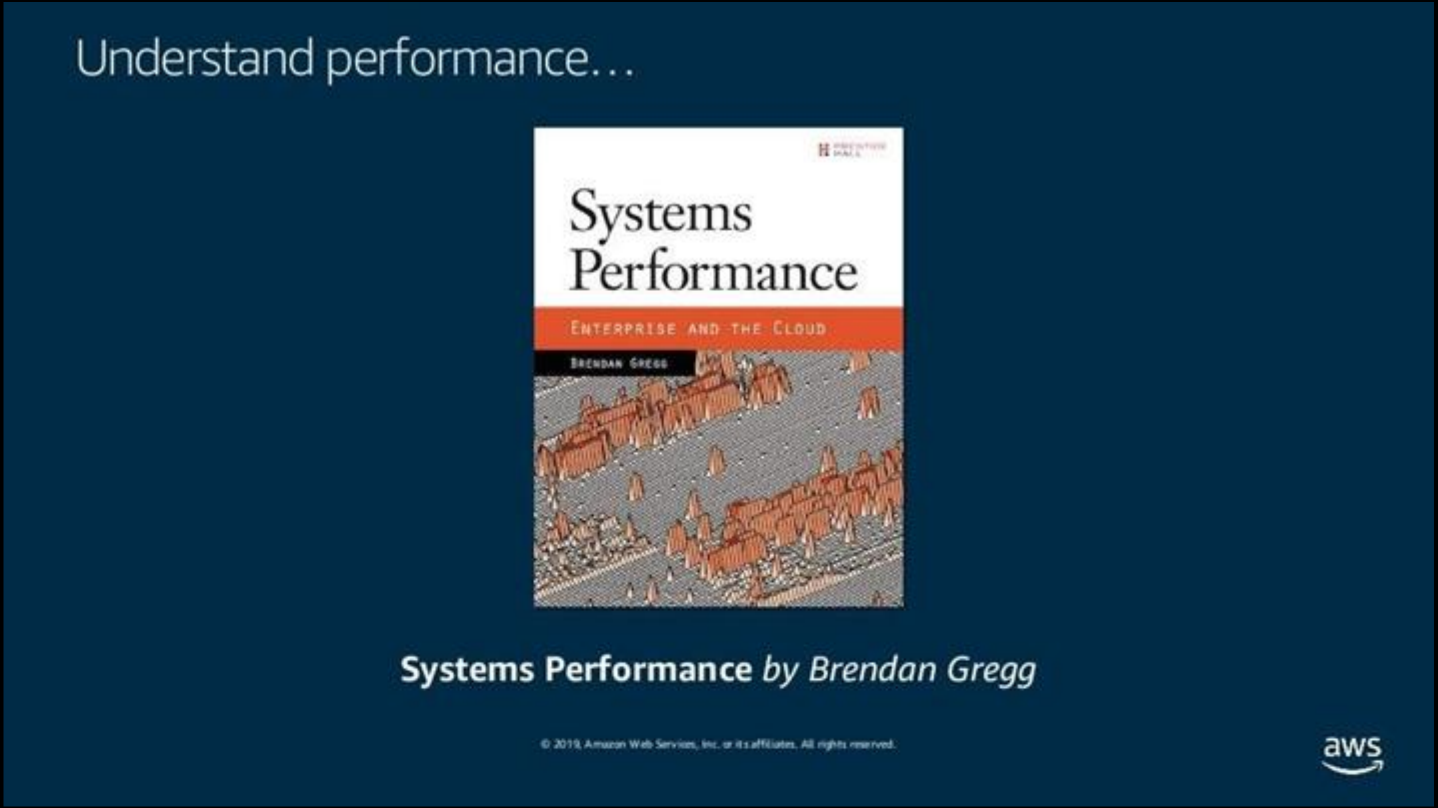
▾ **Scorekeep** AWS::ElasticBeanstalk::Environment

Scorekeep	200	39.0 ms	✓	PUT scorekeep.elasticbean...
DynamoDB	200	6.0 ms	✓	GetItem: scorekeep-game
DynamoDB	200	5.0 ms	✓	GetItem: scorekeep-session
DynamoDB	200	5.0 ms	✓	GetItem: scorekeep-game
DynamoDB	200	6.0 ms	✓	UpdateItem: scorekeep-state
## GameModel.saveGame	-	14.0 ms	✓	
DynamoDB	200	5.0 ms	✓	GetItem: scorekeep-session
DynamoDB	200	8.0 ms	✓	UpdateItem: scorekeep-game

Subsegments

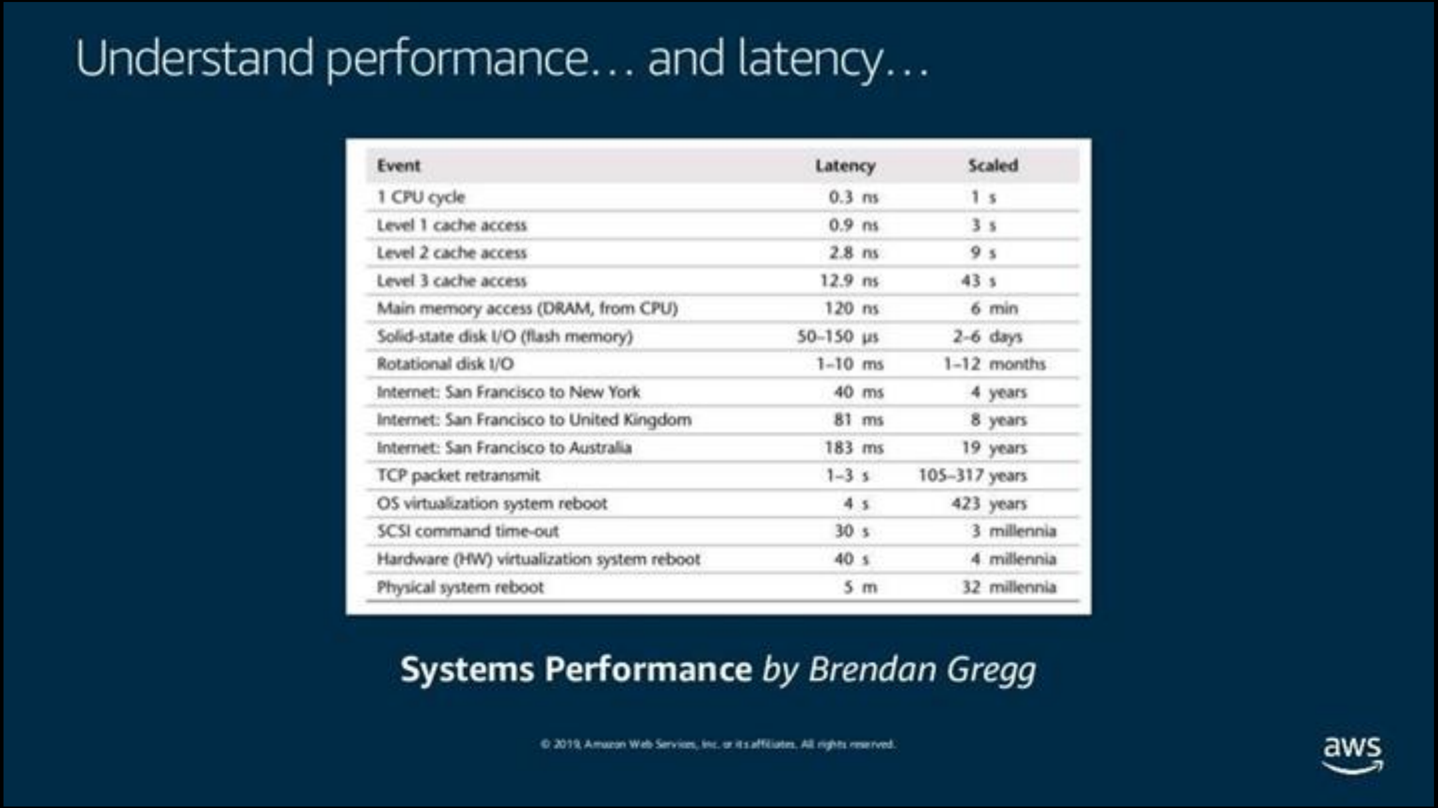
End-to-end tracing – AWS X-Ray Service Map





Understand performance…

Systems Performance *by Brendan Gregg*



Understand performance… and latency…

Event	Latency	Scaled
1 CPU cycle	0.3 ns	1 s
Level 1 cache access	0.9 ns	3 s
Level 2 cache access	2.8 ns	9 s
Level 3 cache access	12.9 ns	43 s
Main memory access (DRAM, from CPU)	120 ns	6 min
Solid-state disk I/O (flash memory)	50–150 μs	2–6 days
Rotational disk I/O	1–10 ms	1–12 months
Internet: San Francisco to New York	40 ms	4 years
Internet: San Francisco to United Kingdom	81 ms	8 years
Internet: San Francisco to Australia	183 ms	19 years
TCP packet retransmit	1–3 s	105–317 years
OS virtualization system reboot	4 s	423 years
SCSI command time-out	30 s	3 millennia
Hardware (HW) virtualization system reboot	40 s	4 millennia
Physical system reboot	5 m	32 millennia

Systems Performance *by Brendan Gregg*

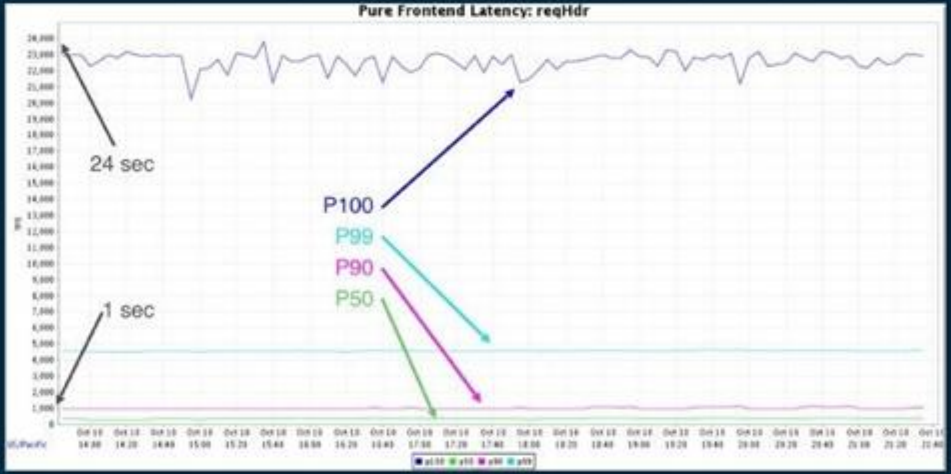
Understand performance… and latency… and percentiles!

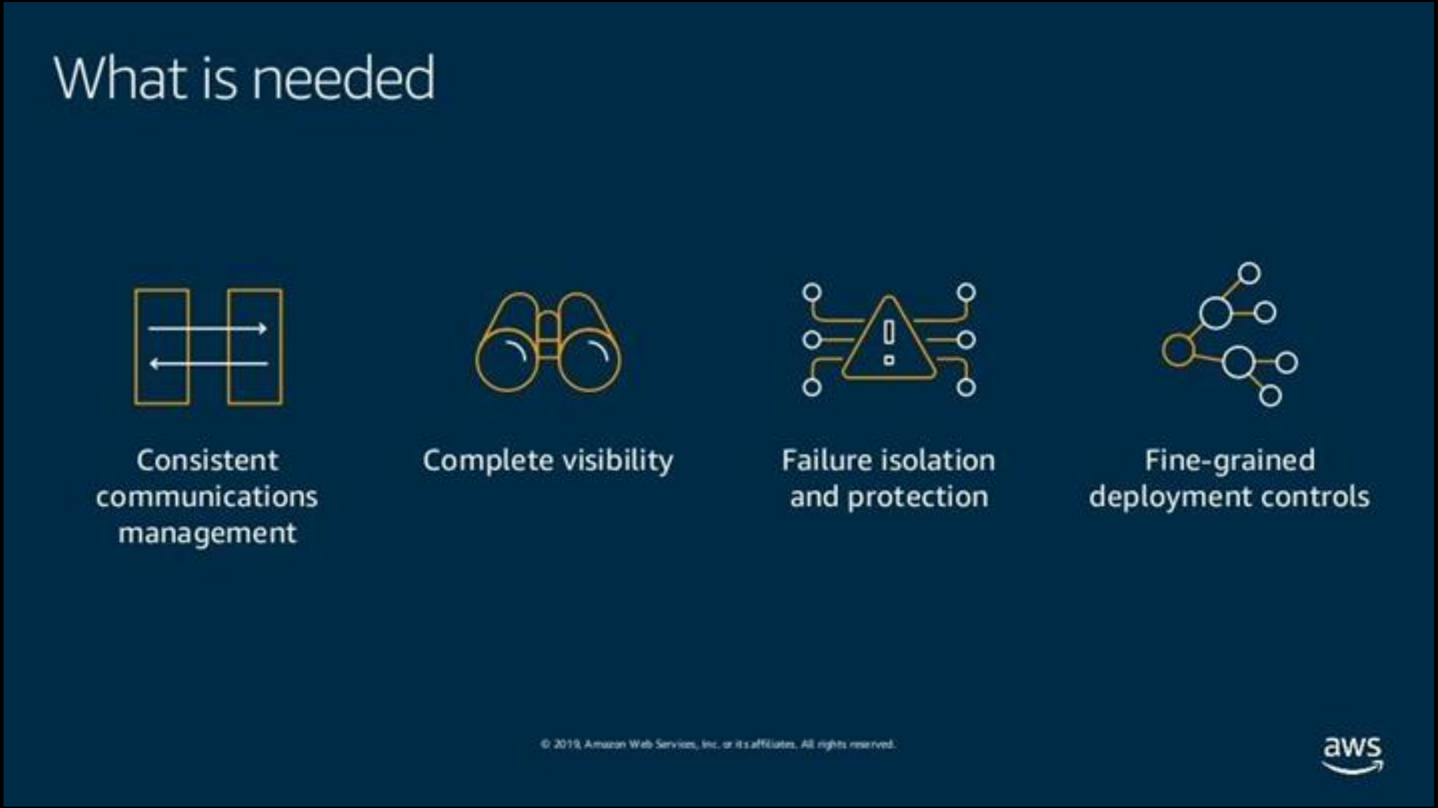
P50

P90

P99

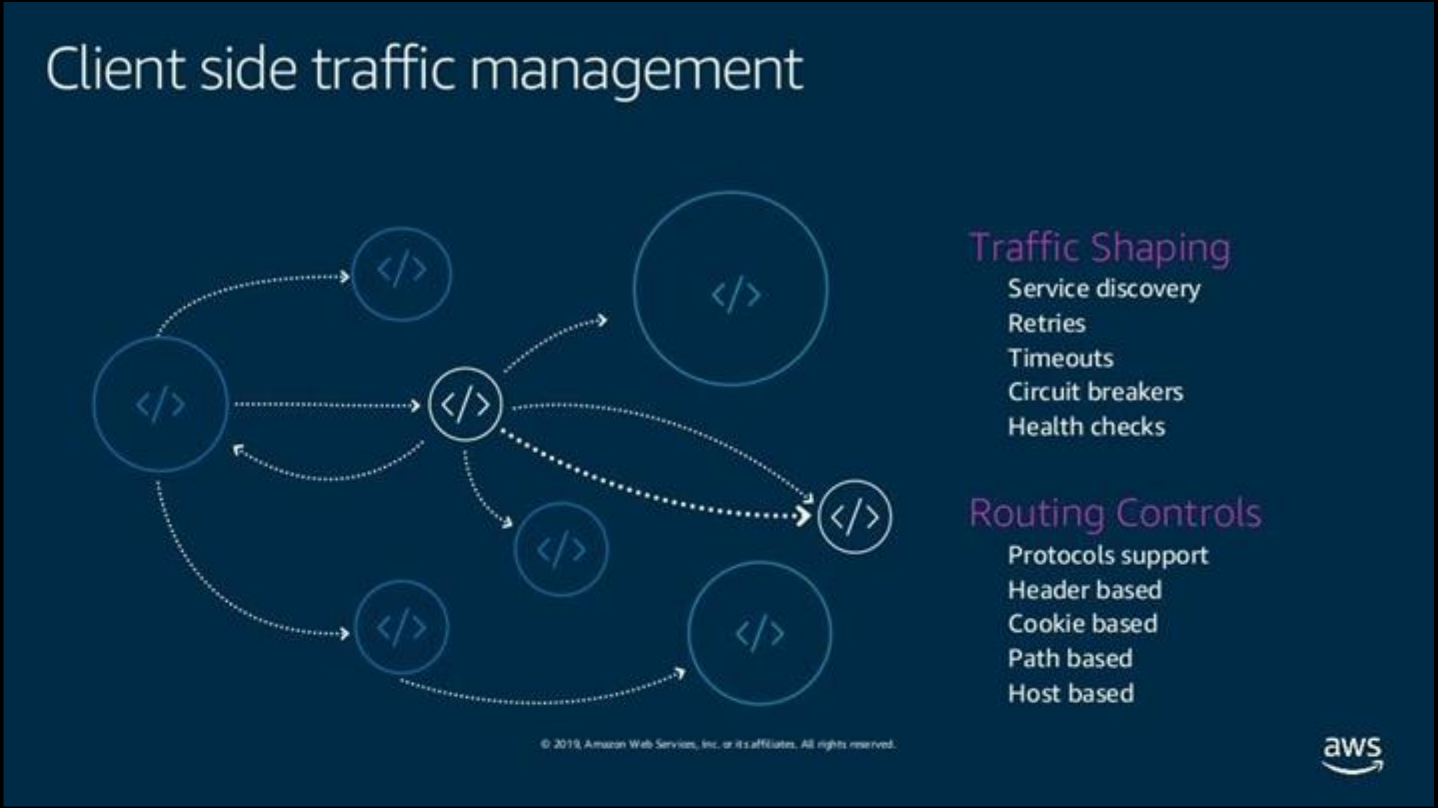
P100





What is needed

- Consistent communications management
- Complete visibility
- Failure isolation and protection
- Fine-grained deployment controls



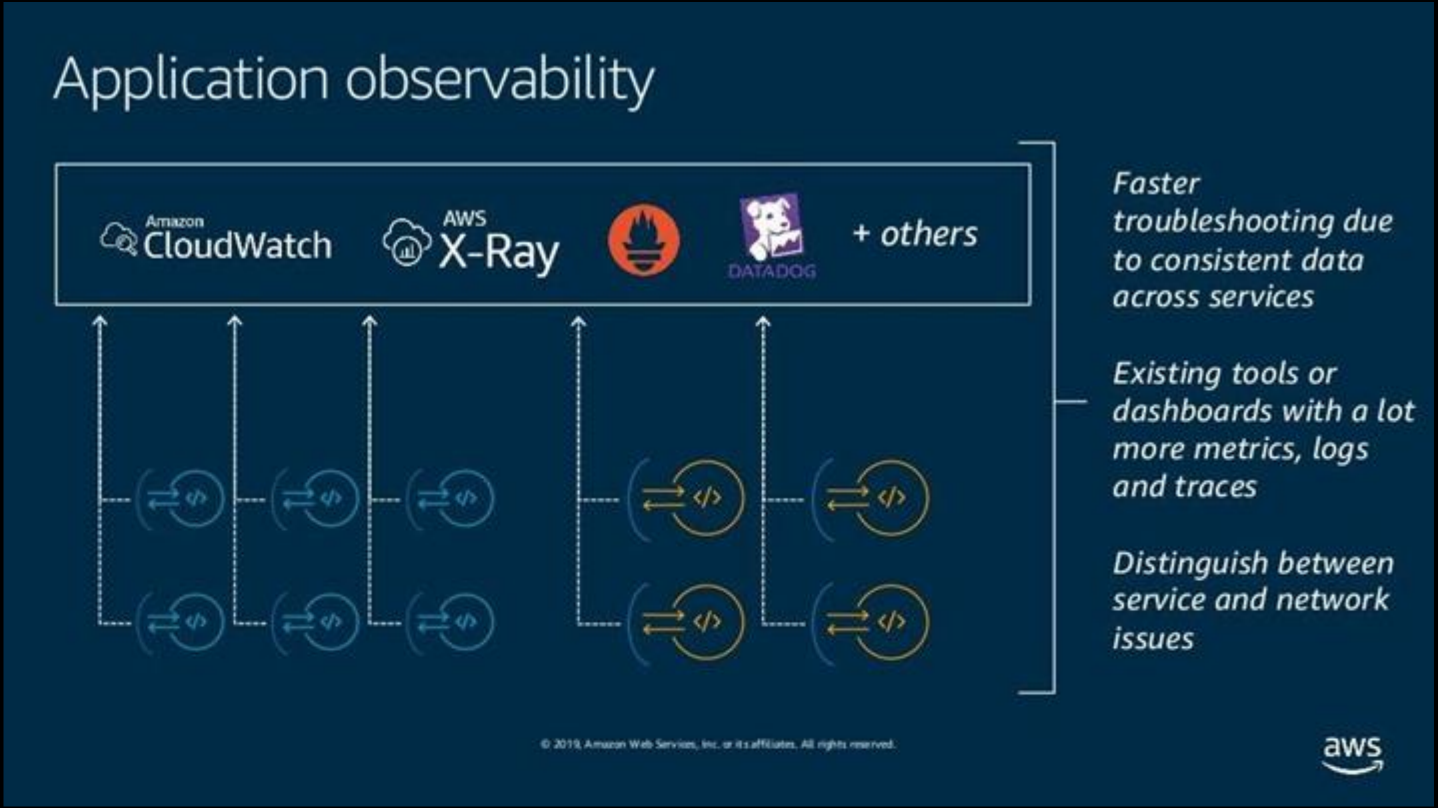
Client side traffic management

Traffic Shaping

- Service discovery
- Retries
- Timeouts
- Circuit breakers
- Health checks

Routing Controls

- Protocols support
- Header based
- Cookie based
- Path based
- Host based



Application observability

Faster troubleshooting due to consistent data across services

Existing tools or dashboards with a lot more metrics, logs and traces

Distinguish between service and network issues

Extra leermaterialen & labs

Bronvermelding slides:

- AWS Techtalk: https://www.slideshare.net/AmazonWebServices/observability-for-modern-applications