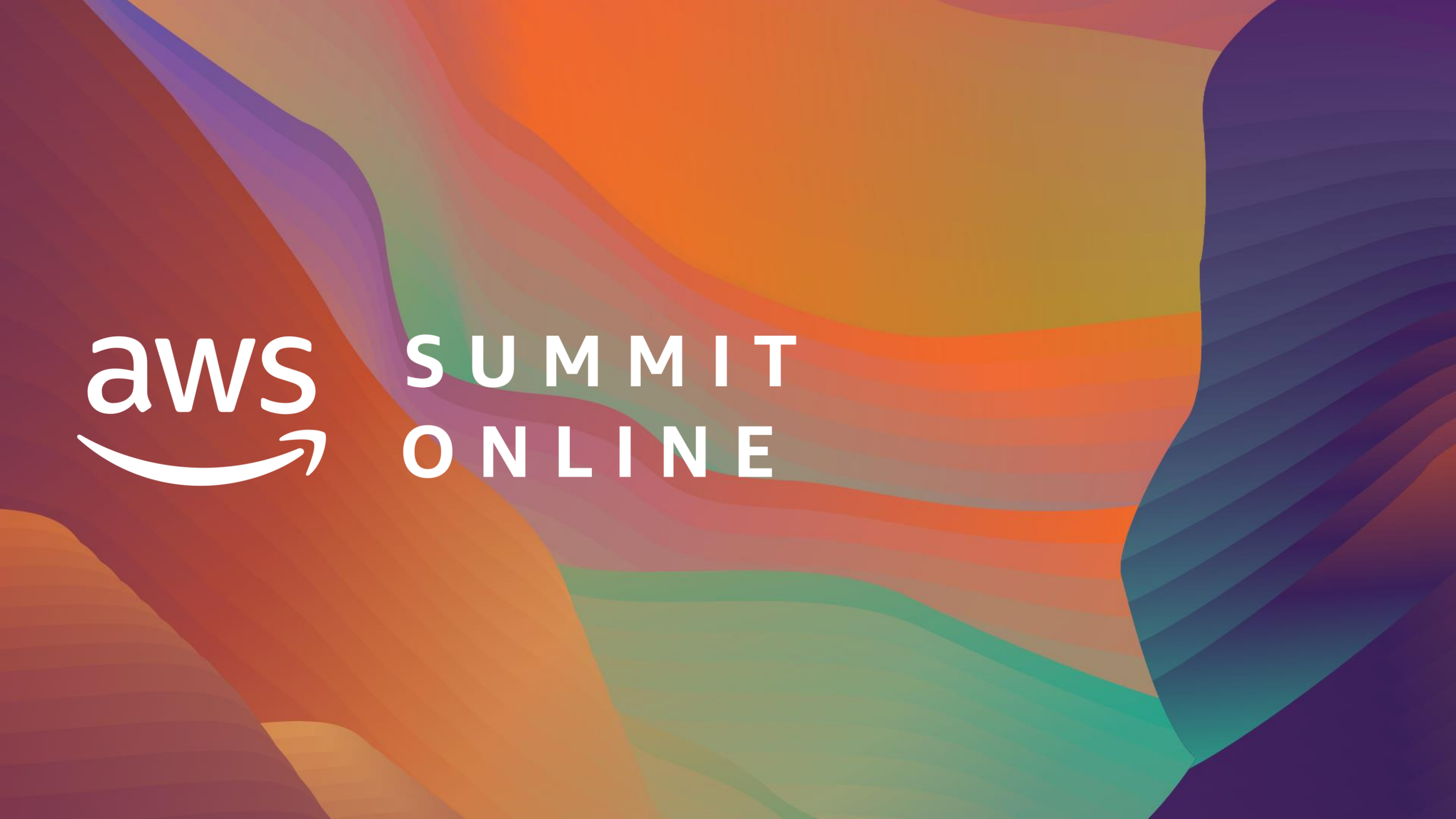

aws
SUMMIT
ONLINE

SEC08

How AFL secures real-time player tracking with encryption

Ralph Stone
Lead Architect
Media Applications
Telstra

Srichakri Nadendla
Enterprise Solutions Architect
Amazon Web Services

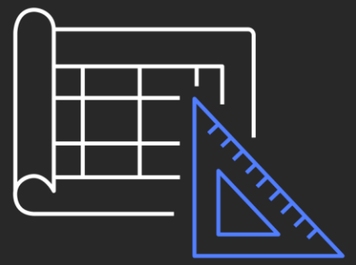
aws SUMMIT ONLINE

Product success

Desirable

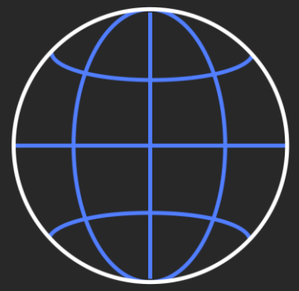
Feasible

Viable

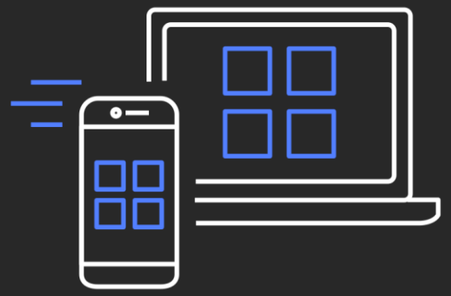


Adaptable

Security and product adoption



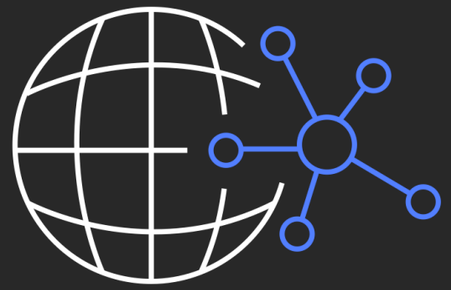
World wide web



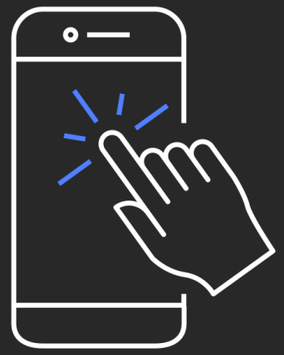
Internet banking
e-commerce

Interconnected
everything

Factors for adoption at scale



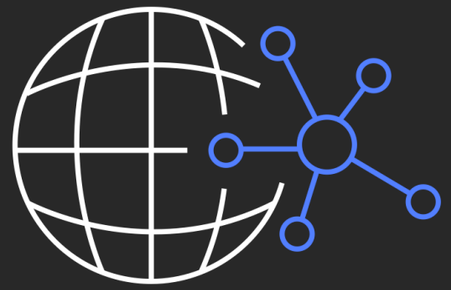
Connectivity



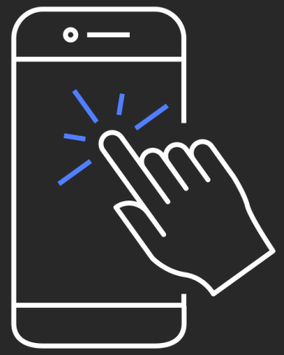
Mobile computing
(smart phones)

An unsung hero

Factors for adoption at scale



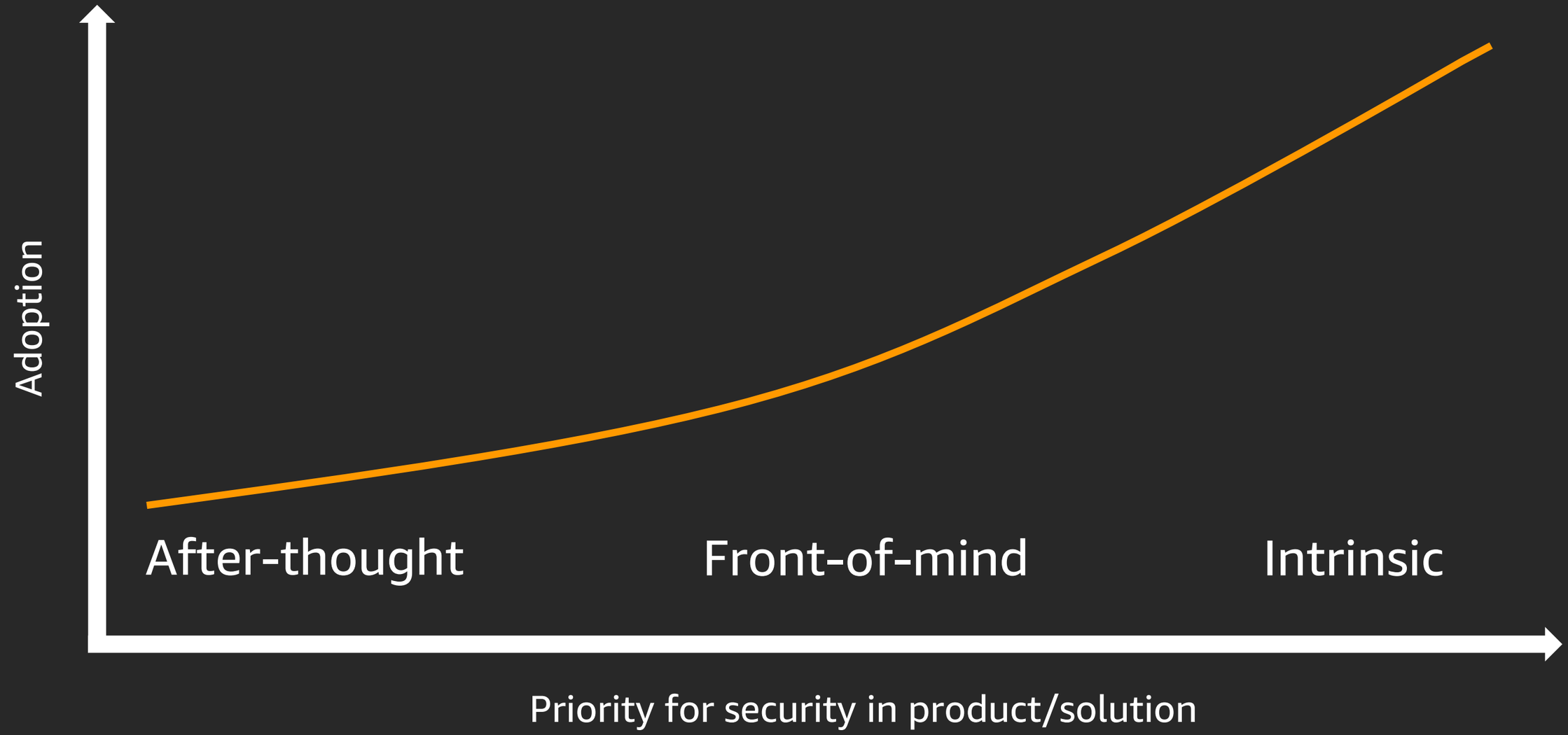
Connectivity



Mobile computing
(smart phones)

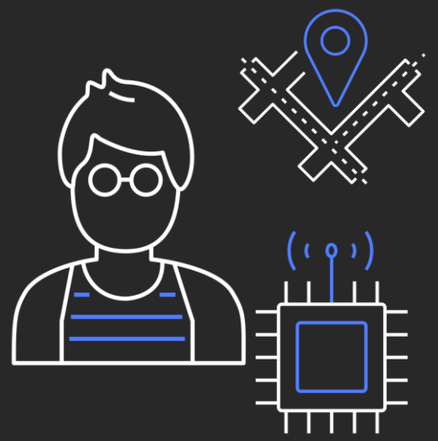
Security

Security a priority to reach adoption potential

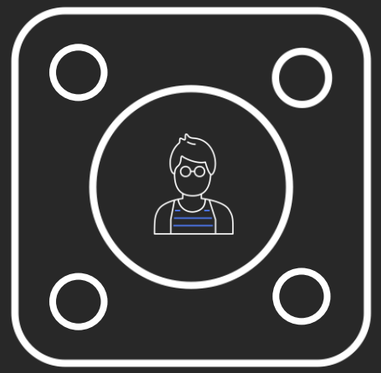


AFL player tracker

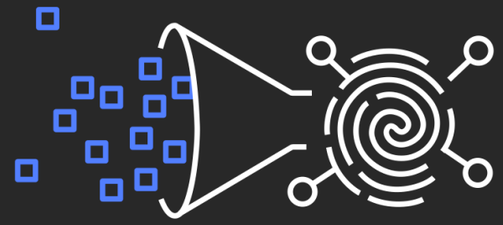
AFL player tracker



GPS and accelerometers

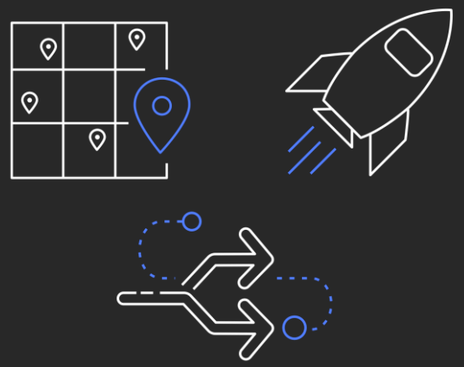


Sensors on ground

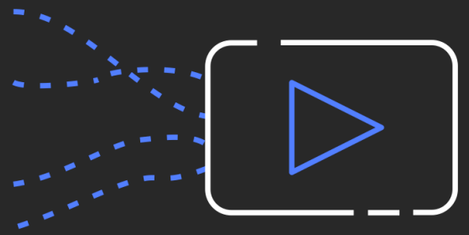


Real-time streaming

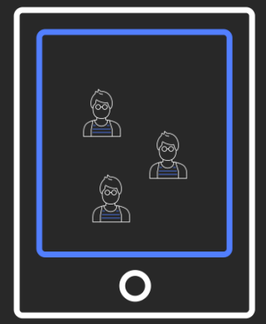
AFL player tracker



Location, speed, distance

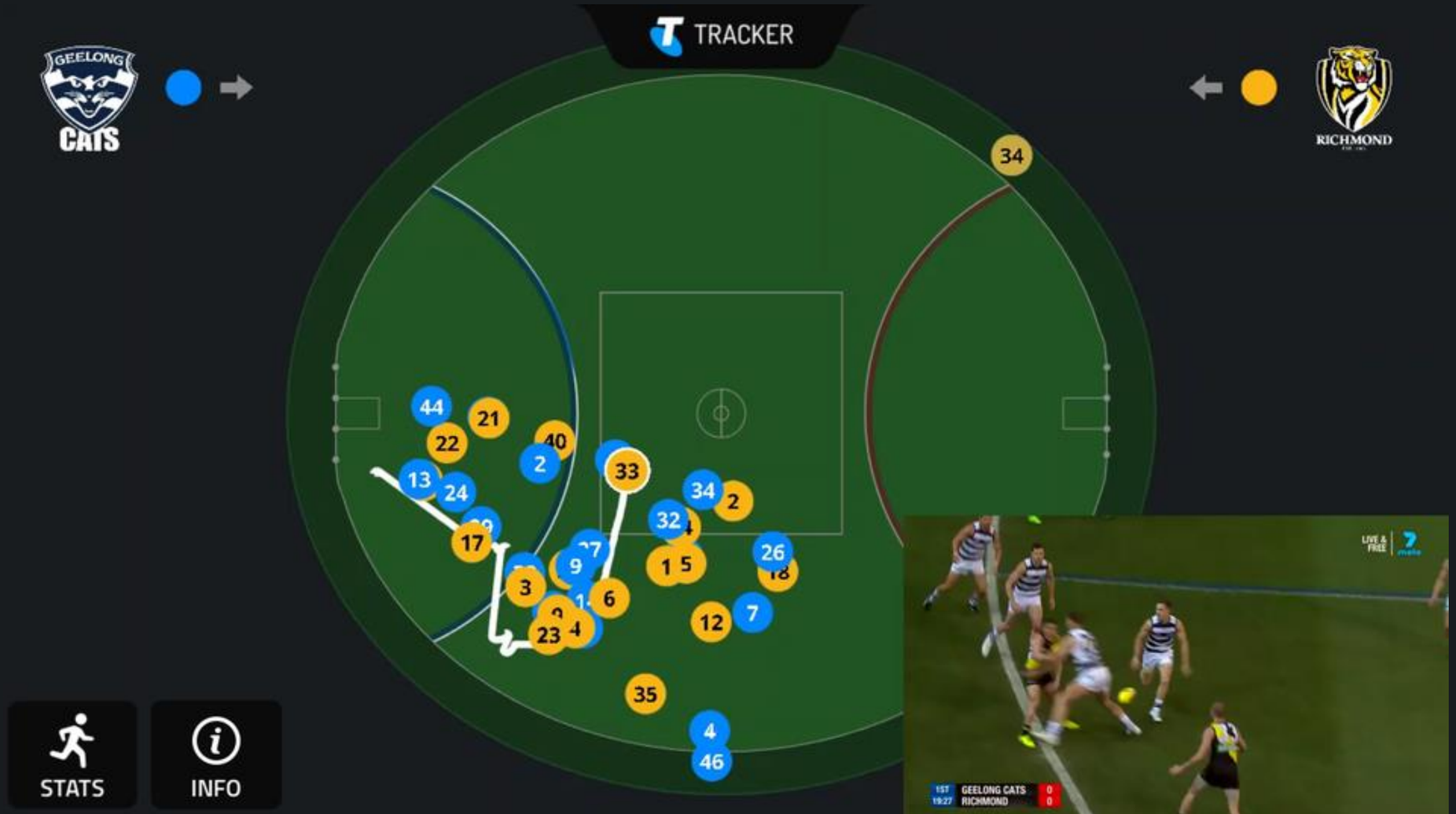


Data and video sync



Mobile based

AFL player tracker



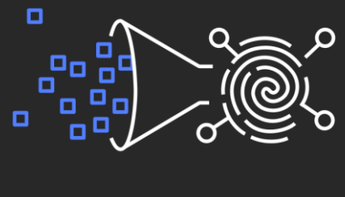
Logo use approved by the AFL and Telstra

Stakeholders, perceptions and risks

AFL

AFL Players Association

Telstra



Data partner

App Dev partner

Everyone’s lawyers

Logo use approved by the AFL and Telstra

Stakeholders, perceptions and risks

AFL

AFL Players Association

Telstra

Data partner

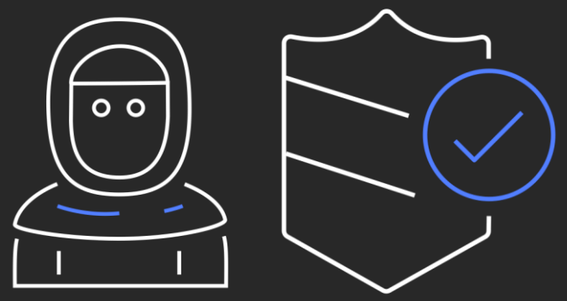
App Dev partner

Everyone's lawyers

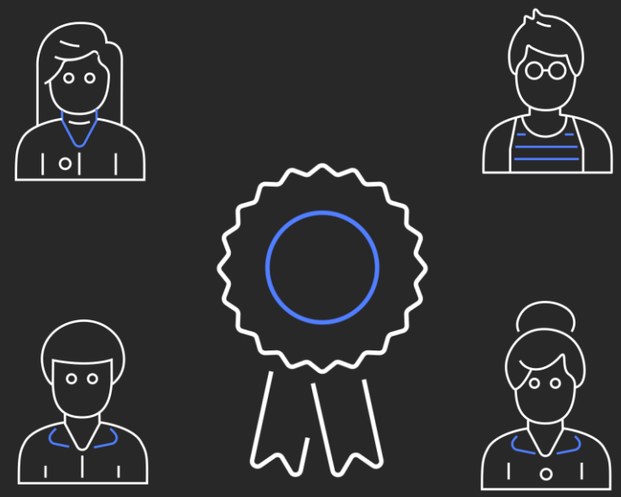
Clubs

Logo use approved by the AFL and Telstra

Beyond security - trust and confidence

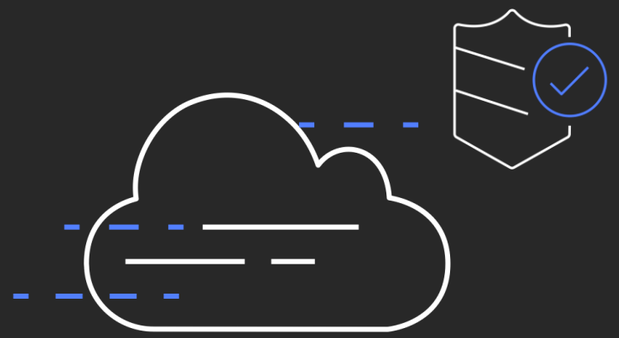


Cryptographers



Stakeholders

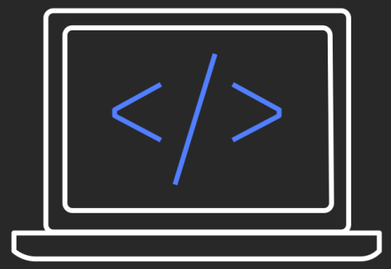
Design considerations



Encryption of streaming data

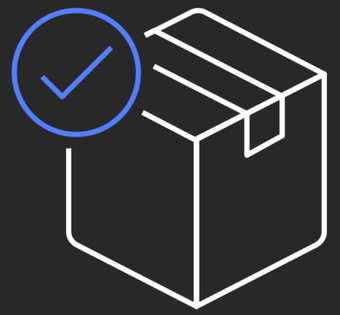
Key management

Three hard problems



Naming
things

Cache
invalidation



Cryptographic key
management

The unexpected option for key management

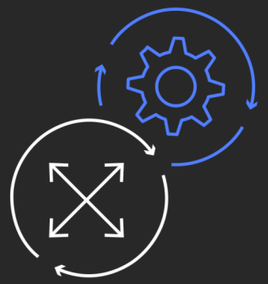
Software only solutions

AWS Key Management Service

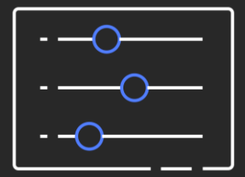
HSM

Hardware Security Module (HSM)

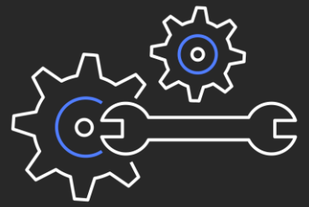
Tamper-proof hardware



Crypto functions



TRNG

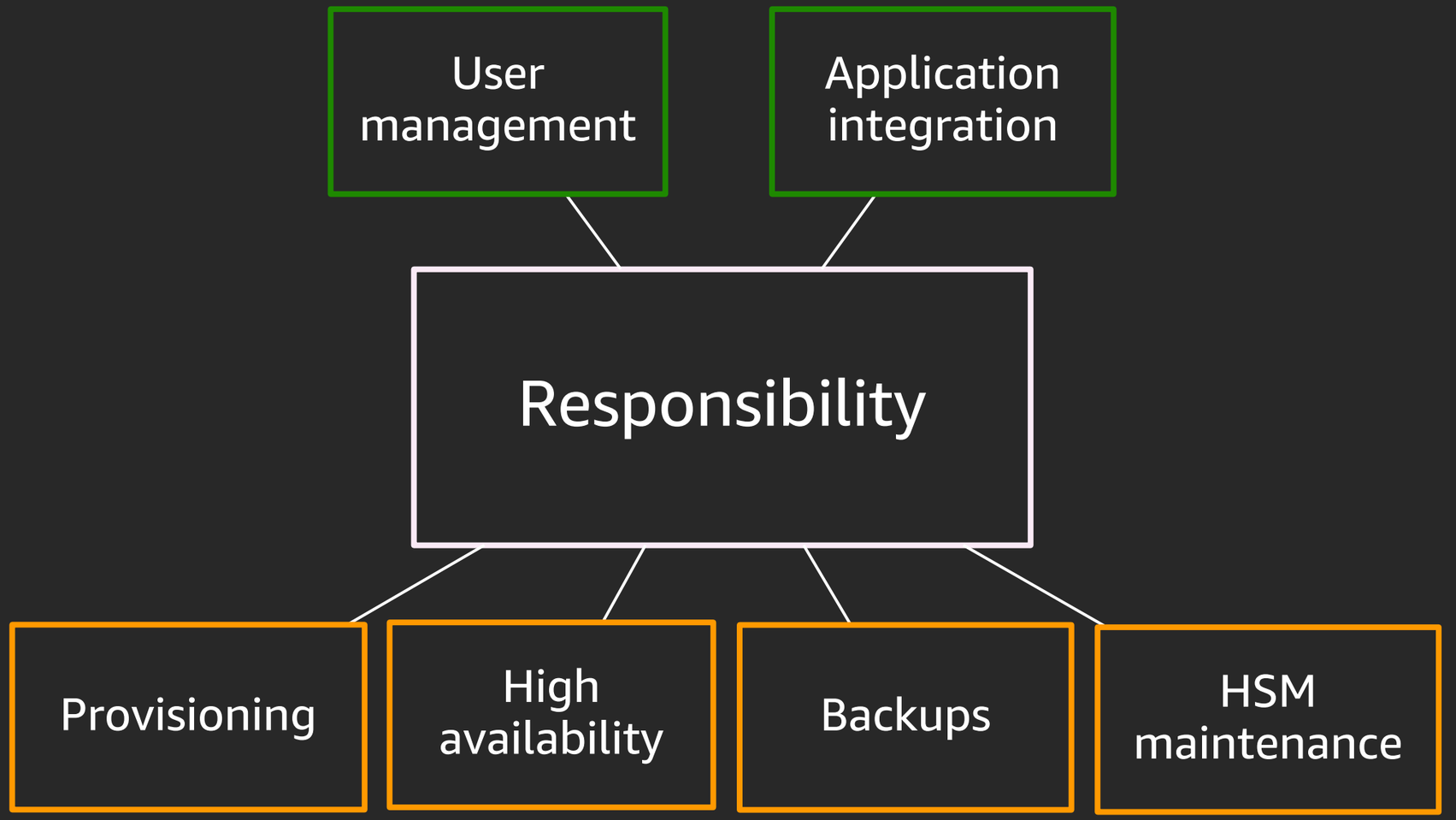


Device management (FIPS 140-2)

Trusted by critical institutions

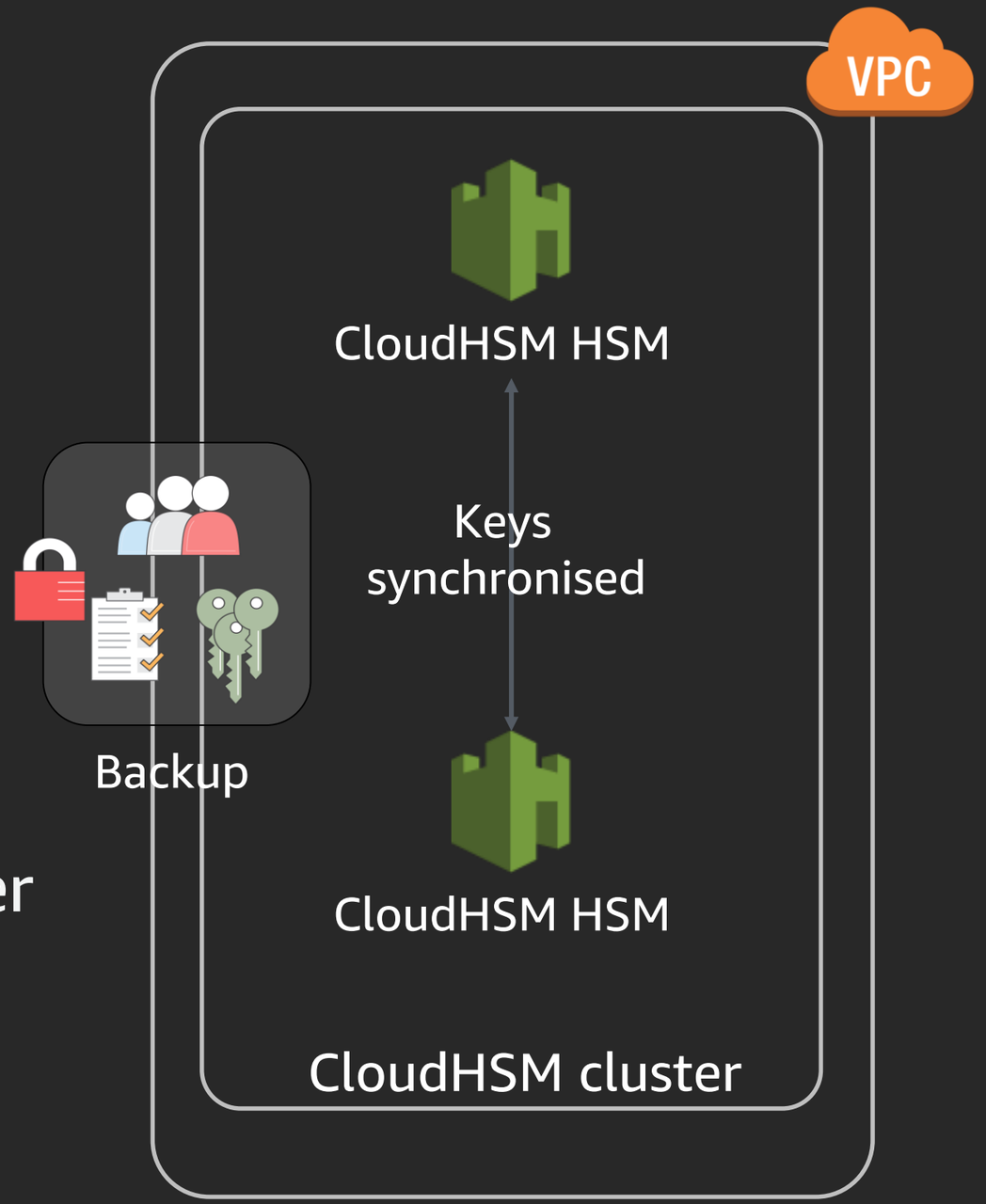
Secure protocols and standards

Shared responsibility with CloudHSM



Concepts in CloudHSM

- Cluster
- HSM
- Backup



- Higher throughput => Expand cluster
- More active keys => New cluster

CloudHSM application integration

Service API

- Console
- AWS CLI/SDK
- Shows in AWS CloudTrail

SDKs: Application development

- PKCS#11
- OpenSSL
- JCE

AWS Command Line Interface

- CloudHSM_mgmt_util – HSM administration
- Key_mgmt_util – Convenient for infrequent key operations

Client daemon

- Used by key_mgmt_util and SDKs to interact with cluster
- Handles load balancing
- Is aware of cluster configuration changes

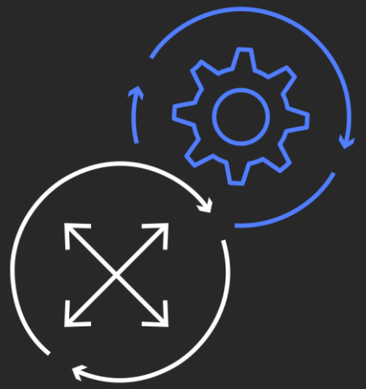
The PoC

Affordable

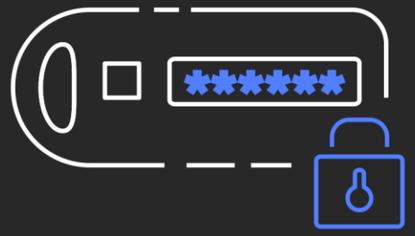
Python and
PKCS11

Key wrapping
(PKI and AES)

The solution



Cryptographic functions in CloudHSM



Random number for user challenges

Developer tasks simplified

Data flow from the ground to cache

1

grnAES [data]

4

appAES [data]

Ingestion

Crypto Server

Cache

decrypt { grnAES [data] }

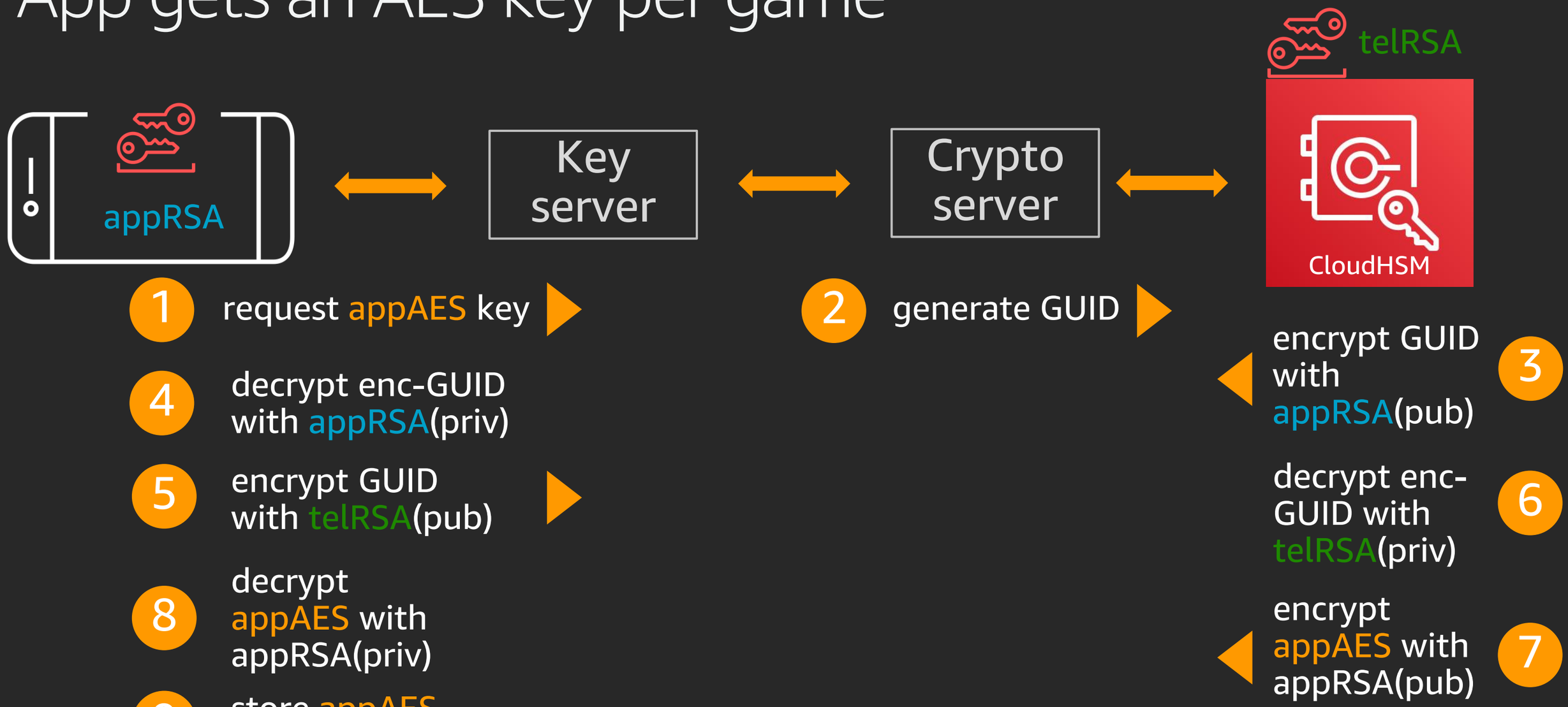
encrypt-with-appAES [data]

2

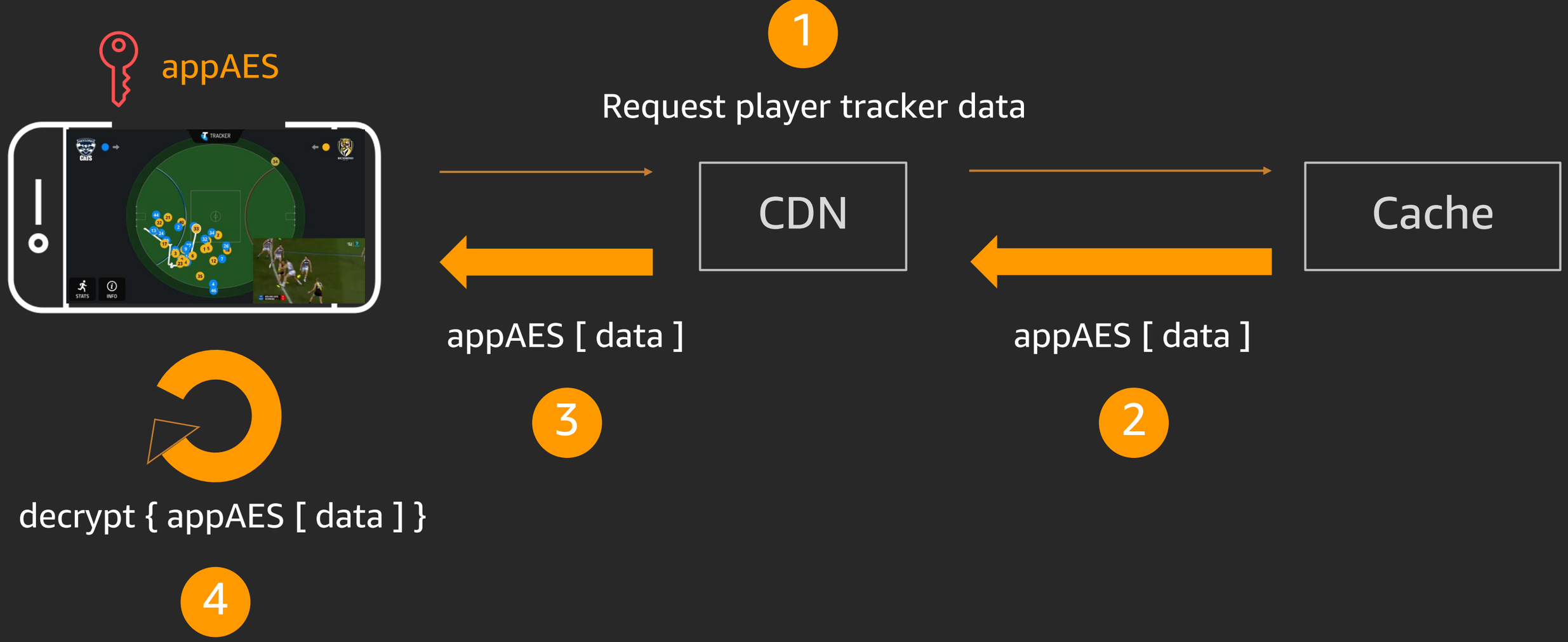
3

CloudHSM

App gets an AES key per game



Getting data from cache



Easy to use

```
# load the aws cloudhsm library

lib = pkcs11.lib('/opt/cloudhsm/lib/libcloudhsm_pkcs11_standard.so’)

# fetch the hsm PIN, and open a session to the default token (cavium)

hsmPIN = getParameter("hsmPIN")

token = lib.get_token(token_label='cavium’)

session = token.open(user_pin=hsmPIN)

# open the app’s RSA public key: we use this to wrap match AES keys

keyAppPublic = session.get_key(id=17, object_class=ObjectClass.PUBLIC_KEY, label='pAppPrivate’)

# ask the hsm to decrypt with the cKey then re-encrypt with the pKey

e = pEncryptKey.encrypt(cDecryptKey.decrypt(thisIterableData,
                                            mechanism=Mechanism.AES_CBC, mechanism_param=cIV),
                        mechanism=Mechanism.AES_CBC, mechanism_param=pIV)
```

Easy to use

```
# load the aws cloudhsm library

lib = pkcs11.lib('/opt/cloudhsm/lib/libcloudhsm_pkcs11_standard.so')

# fetch the hsm PIN, and open a session to the default token (cavium)

hsmPIN = getParameter("hsmPIN")

token = lib.get_token(token_label='cavium')

session = token.open(user_pin=hsmPIN)

# open the app's RSA public key: we use this to wrap match AES keys

keyAppPublic = session.get_key(id=17, object_class=ObjectClass.PUBLIC_KEY, label='pAppPrivate')

# ask the hsm to decrypt with the cKey then re-encrypt with the pKey

e = pEncryptKey.encrypt(cDecryptKey.decrypt(thisIterableData,
                                            mechanism=Mechanism.AES_CBC, mechanism_param=cIV),
                        mechanism=Mechanism.AES_CBC, mechanism_param=pIV)
```

Easy to use

```
# load the aws cloudhsm library

lib = pkcs11.lib('/opt/cloudhsm/lib/libcloudhsm_pkcs11_standard.so’)

# fetch the hsm PIN, and open a session to the default token (cavium)

hsmPIN = getParameter("hsmPIN")

token = lib.get_token(token_label='cavium’)

session = token.open(user_pin=hsmPIN)

# open the app’s RSA public key: we use this to wrap match AES keys

keyAppPublic = session.get_key( id=17, object_class=ObjectClass.PUBLIC_KEY,
                                label='pAppPrivate’ )

# ask the hsm to decrypt with the cKey then re-encrypt with the pKey

e = pEncryptKey.encrypt(cDecryptKey.decrypt(thisIterableData,
                                            mechanism=Mechanism.AES_CBC, mechanism_param=cIV),
                        mechanism=Mechanism.AES_CBC, mechanism_param=pIV)
```

Easy to use

```
# load the aws cloudhsm library

lib = pkcs11.lib('/opt/cloudhsm/lib/libcloudhsm_pkcs11_standard.so’)

# fetch the hsm PIN, and open a session to the default token (cavium)

hsmPIN = getParameter("hsmPIN")

token = lib.get_token(token_label='cavium’)

session = token.open(user_pin=hsmPIN)

# open the app’s RSA public key: we use this to wrap match AES keys

keyAppPublic = session.get_key(id=17, object_class=ObjectClass.PUBLIC_KEY, label='pAppP   srivate’)

# ask the hsm to decrypt with the cKey then re-encrypt with the pKey

e = pEncryptKey.encrypt(
                    cDecryptKey.decrypt( thisIterableData,
                                         mechanism=Mechanism.AES_CBC,
                                         mechanism_param=cIV ),
                    mechanism=Mechanism.AES_CBC, mechanism_param=pIV )
```

Player tracker during a game

Over 100,000

Concurrent sessions

1 second

Liveness latency

2 ⟷ 6

Scaling CloudHSMs

Managing AWS CloudHSM

Operational ease

Scalability

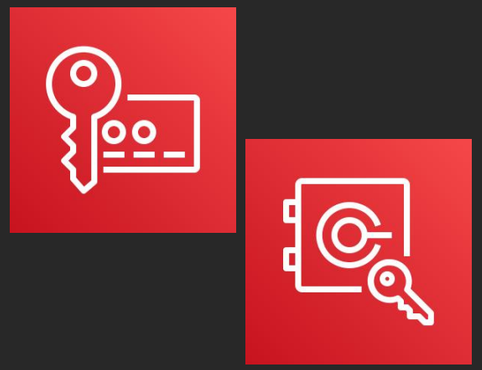
Support from AWS

Positive stakeholder sentiment

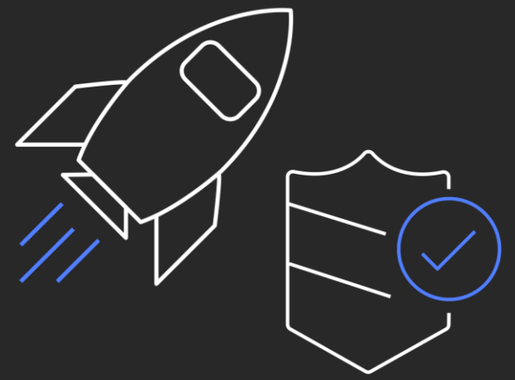
- [x] Secure player performance data [Viability]
- [x] Protected using military grade means [Feasibility]
- [x] Novel fan engagement with live data [Desirability]
- [x] No compromise on risk or user-experience [Viability]
- [x] Scales seamlessly with viewership demand [Adaptability]

Additional considerations

PKCS#11



KMS vs
CloudHSM

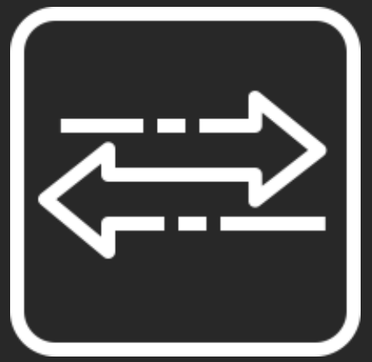


Dev speed with
security

What we learnt

Need for trust

AWS Security practice



Faster to market

Choice enables flexibility and innovation

Identity and access management

- Amazon Cognito
- AWS Directory Service
- AWS IAM
- AWS Resource Access Manager
- AWS Secrets Manager
- AWS SSO

Detective controls

- AWS Security Hub
- Amazon GuardDuty
- Amazon Inspector
- Amazon Macie
- Amazon Detective

Infrastructure protection

- AWS Shield
- AWS Shield Advanced
- AWS Web Application Firewall (WAF)
- AWS Firewall Manager

Data protection

- AWS Key Management Service (KMS)
- AWS CloudHSM
- AWS Certificate Manager

Security documentation by category

Compute

- Amazon EC2
- AWS Batch
- Amazon ECR
- Amazon ECS
- Amazon EKS
- AWS Elastic Beanstalk
- AWS Lambda
- AWS Serverless Application Repository

Databases

- Amazon Aurora
- Amazon DocumentDB
- Amazon DynamoDB
- Amazon ElastiCache
- Amazon Neptune
- Amazon RDS
- Amazon Redshift

Management and governance

- AWS CloudTrail
- Amazon CloudWatch
- AWS Command Line Interface (AWS CLI)
- AWS Control Tower
- AWS License Manager
- AWS OpsWorks
- AWS Organizations
- AWS Service Catalog
- Service Quotas
- AWS Systems Manager
- AWS Tools for Powershell
- AWS Well-Architected Tool
- AWS Chatbot

Storage

- Amazon S3
- AWS Backup
- Amazon EFS
- Amazon S3 Glacier
- AWS Snowball
- AWS Storage Gateway

Machine Learning

- AmazonComprehend
- AWSDeep Learning-AMI
- AmazonElastic Inference
- AmazonKendra
- AmazonLex
- AmazonPersonalize
- AmazonRekognition
- AmazonSageMaker
- AmazonTranscribe
- AmazonTranslate

Analytics

- Amazon Athena
- AWS Data Exchange
- Amazon Elasticsearch Service
- Amazon EMR
- AWS Glue
- Amazon Kinesis Data Analytics for Java Applications
- Amazon Kinesis Data Analytics for SQL Applications
- Amazon Kinesis Data Firehose
- Amazon Kinesis Data Streams
- Amazon Kinesis Video Streams
- Amazon MSK
- Amazon QuickSight
- Amazon Redshift

https://docs.aws.amazon.com/security/

You can bake security and have speed too

Continuous compliance

- Using AWS Management Tools
- Using AWS Security Hub

Security workshops

- Guide to AWS encryption SDK
- Securing EKS Cluster

Thank you!

Ralph Stone

Lead Architect
Media Applications
Telstra

Srichakri Nadendla

Enterprise Solutions Architect
Amazon Web Services

aws SUMMIT ONLINE