AWS re:Invent

STG340

What to consider when building a data lake on Amazon S3

Ruhi Dang
Senior Product Manager, Amazon S3
Amazon Web Services

Malik Bouchet
Senior Software Engineer, Amazon S3
Amazon Web Services

Tim Harris
Principal Engineer, Amazon S3
Amazon Web Services

AWS re:Invent

aws

Agenda

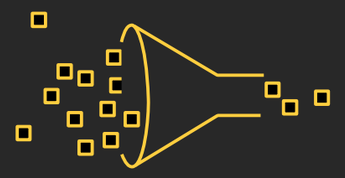
Considerations for data lakes

Recommendations

Customer examples

Q&A

S3 is the foundation of any data lake



Multiple data input sources

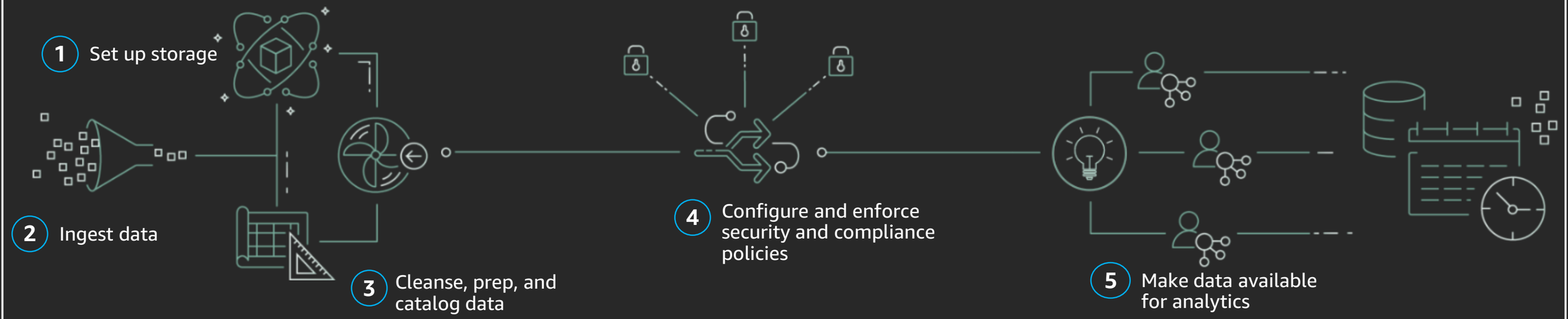
Storage scales on demand

Supports many unique users and teams

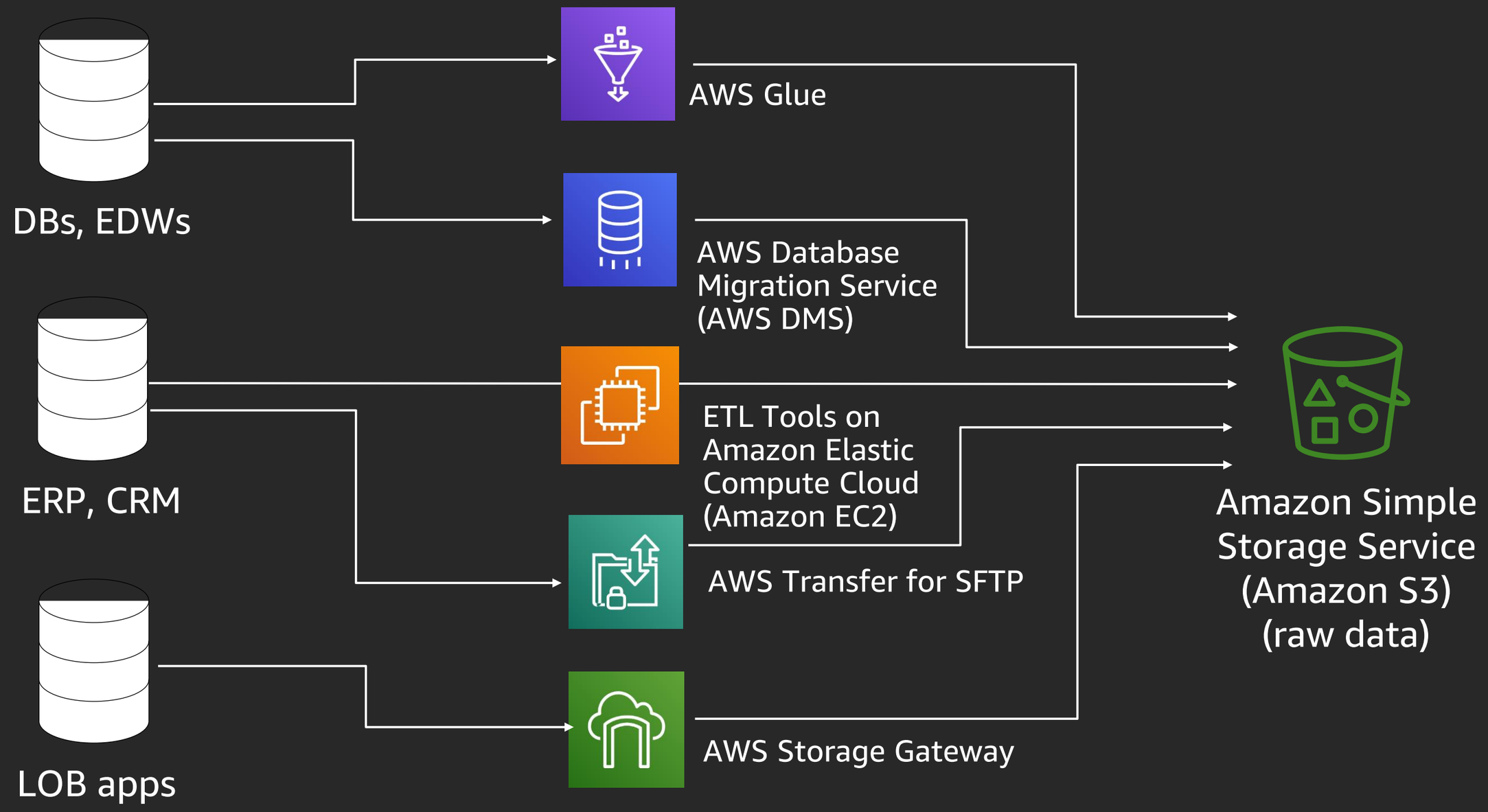
Analyzed by many applications

Data lake workflow pattern

Typical steps in building a data lake



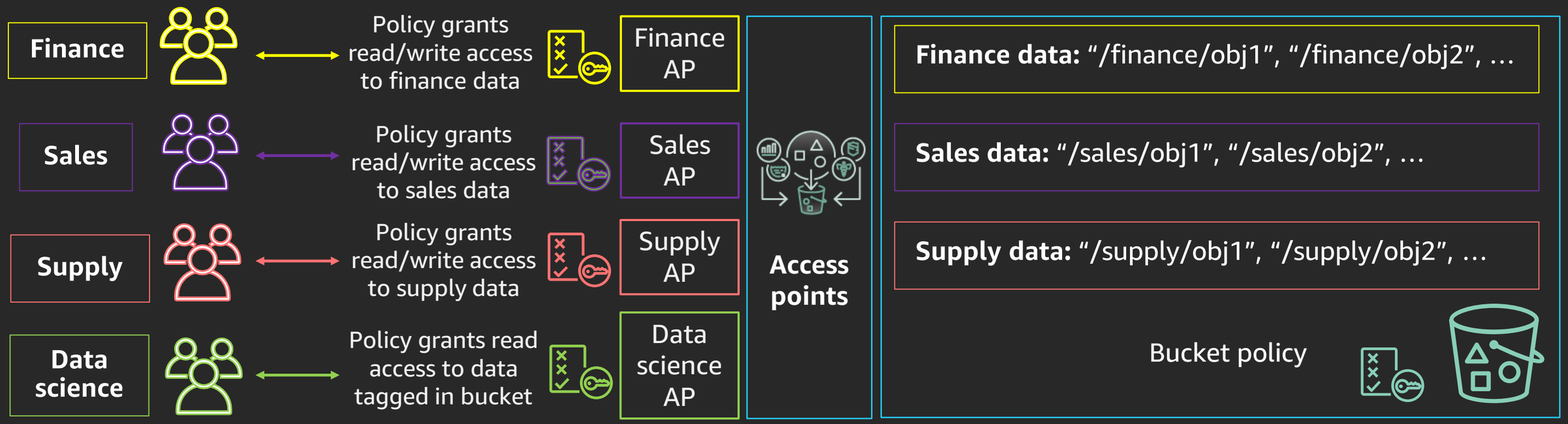
Options for structured data ingestion



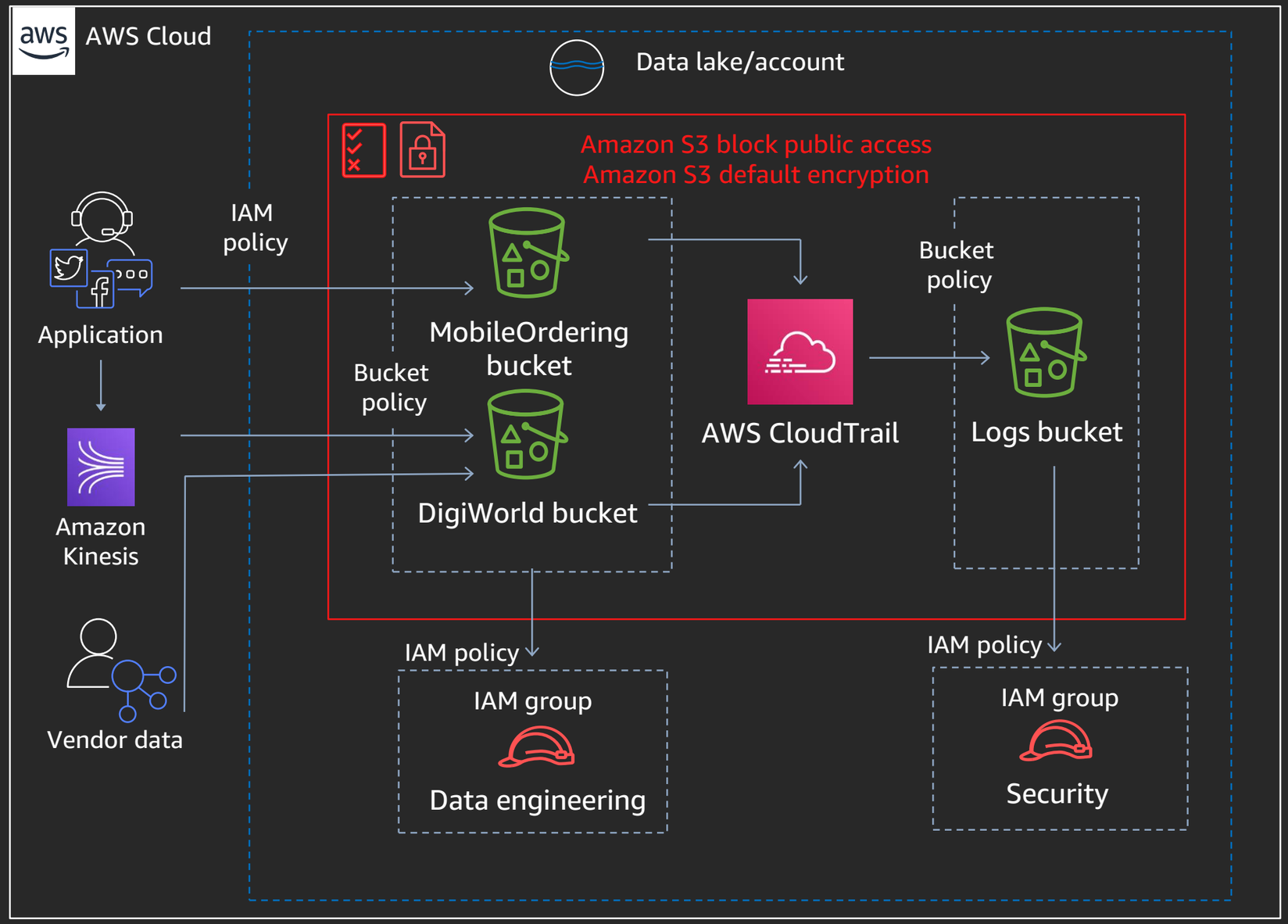
New!

Recommendation: Use Amazon S3 access points to manage your application set

Granular control for hundreds of teams accessing your data lake



Secure your data lake!



Deny access by default

Encrypt your data

Secure multiple data input sources

Provide specific access where appropriate

Support multiple unique users and teams

Recommendation: Block public access to your data lake

Four security settings to deny public access

Account level
or
Bucket level
or
Access point level

Use AWS Organizations Service Control Policies (SCPs) to prevent configuration changes

Recommendation: Encrypt data in transit and by default at rest

Encryption in transit

HTTPS/TLS enforced by policy

Encryption at rest

Server side

- SSE-S3 (Amazon S3 managed keys), SSE-KMS (AWS Key Management Service)
- Encrypt by DEFAULT AT THE BUCKET

Client side

- SSE-C (customer-provided keys)
- Encrypt with the AWS Encryption SDK

Recommendation: Data lake security best practices

- (Account) block public access: Enable
- (Bucket) default encryption: SSE or SSE-KMS
- By bucket policy, require TLS
- CloudTrail trails and S3 server access logs enable security and access audits
- VPC endpoint: Enable and require, with bucket policies limiting access
- MFA delete and object lock governance mode for permanence

Recommendation: Performance design patterns

Structure key namespace to scale

Most workloads fit in the S3 3500 PUT/5500 GET TPS per key name partition

Amazon S3 automatically creates partitions as data lake use increases

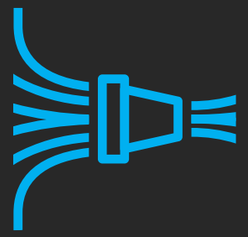
Extremely bursty workloads might require customized key name design

Consider object format and size

Use Parquet or optimized columnar format

Aim for 2–16 MB minimum object size (might require aggregation during ingest)

Perform parallel byte range accesses (included in AWS SDKs)



Use latest SDKs and software versions

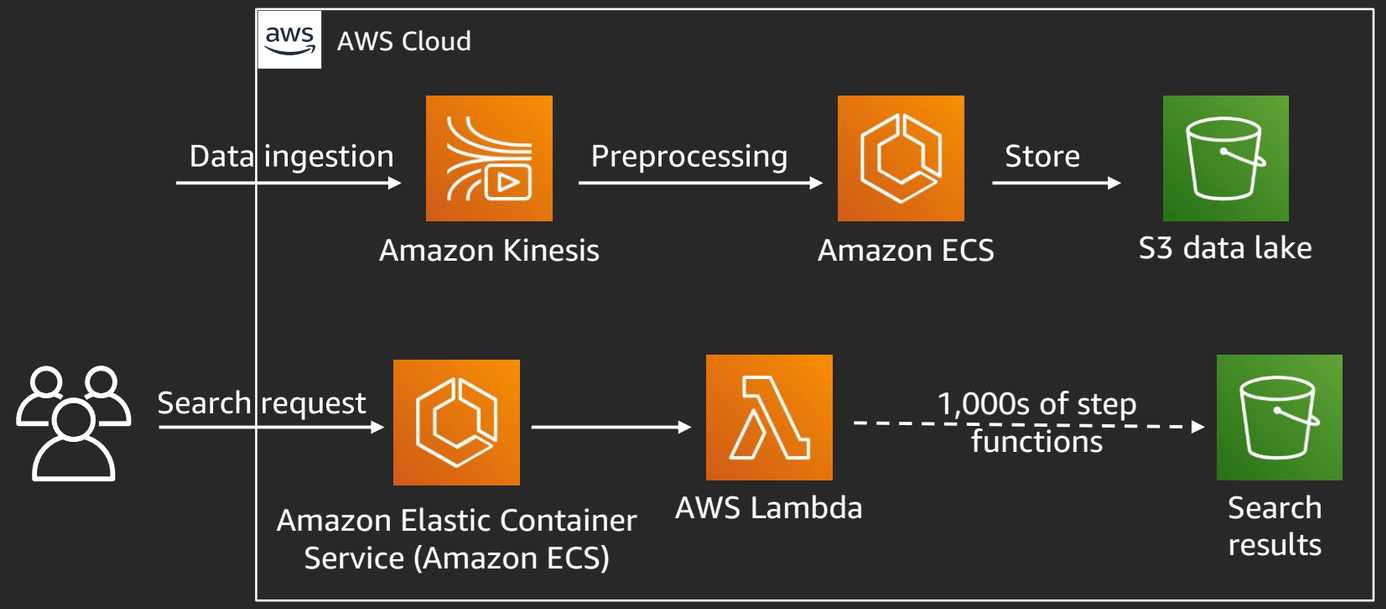
AWS SDKs include support for latest features and optimizations

Amazon EMR 5.18 and above supports S3 Select for Hive, Presto, and Spark

Alert logic: Storage and delivery of search data

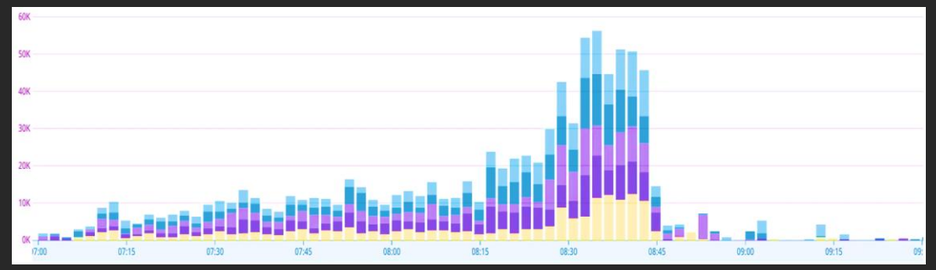
Workload: Security company processing 25 PB of end-user data, providing search functionality

Architecting on AWS for scale



Performance results

- Ran 40,000 search queries
- Scaled from 0 to 55,000 concurrent Lambda functions

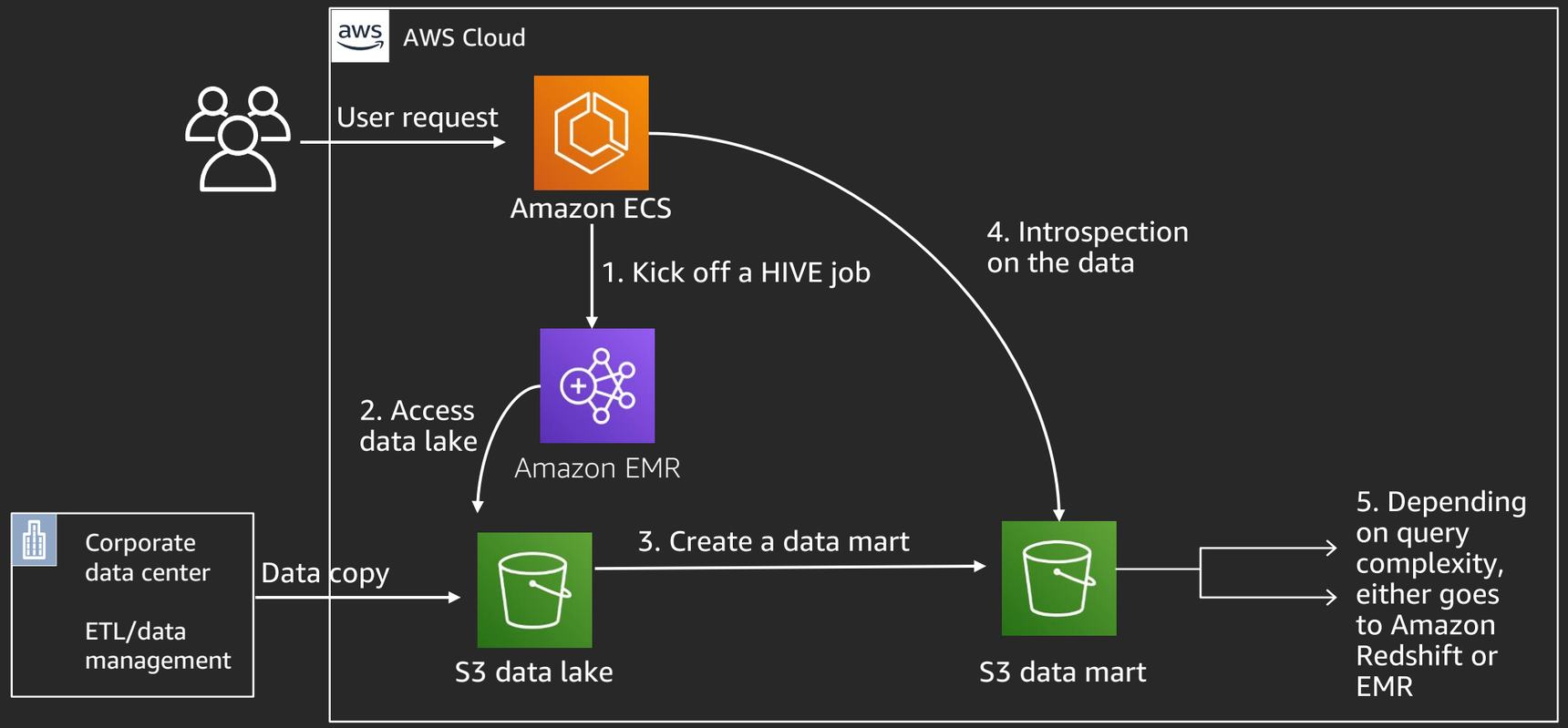


- Reached 18 million executions in an hour

FINRA: Petabyte-scale data analysis

Workload: Financial regulatory authority providing users ability to access PBs of data for analytics

Architecting on AWS for scale

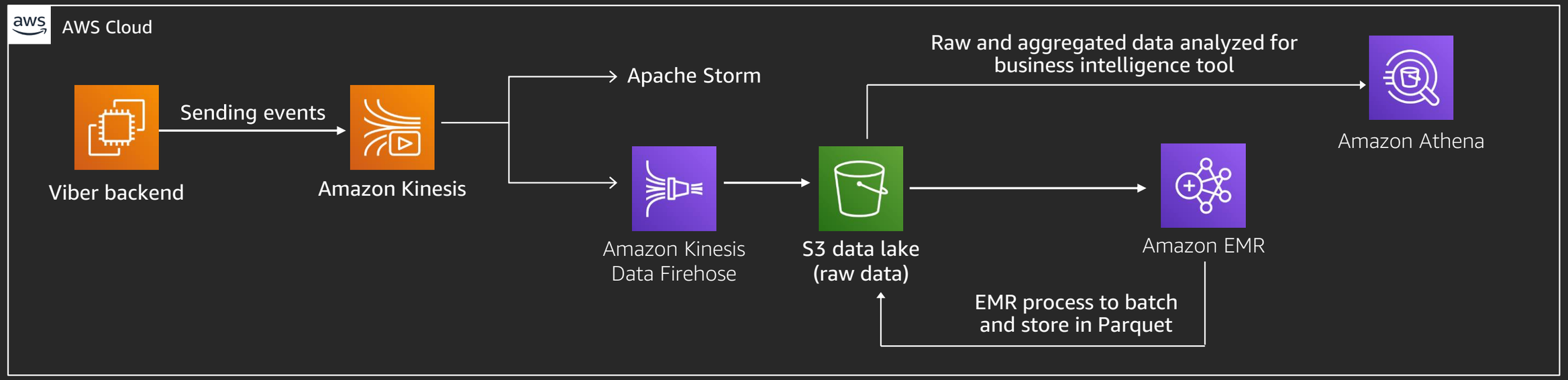


"With the new architecture, we uncovered more needs of the end user. This led us to move from one large Amazon Redshift cluster to a blend of querying engines."

Viber: Processing events on data lake

Workload: Communications platform serving a billion users worldwide

- Processing over 10–15 billion events per day
- Peaking at 300,000 events per second
- Storing many petabytes of data
- Running over 200 events on a single Kinesis stream



What are your data lake needs?

Thank you!

AWS re:Invent

aws

Please complete the session survey in the mobile app.

AWS re:Invent

aws