LESSONS LEARNED RUNNING

DATA PIPELINE ON AWS

RED | VENTURES

Majid Fatemian

mfatemian@redventures.com
@majidfn

aws CERTIFIED - SAA

RED | VENTURES

TECH & DATA-DRIVEN

CUSTOMER ACQUISITION

SALES & MARKETING

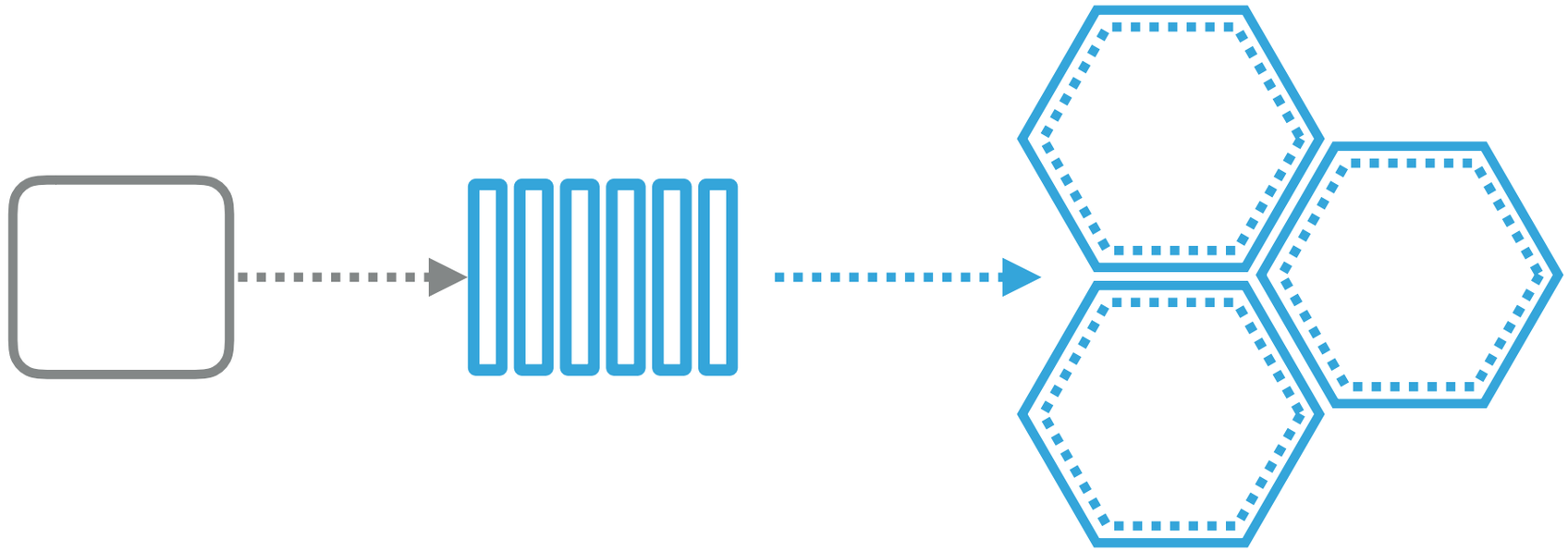
DATA


PRODUCT RECOMMENDATION
MACHINE LEARNING
DATA SCIENCE
USER EXPERIENCE
REALTIME ANALYSIS
DATA WAREHOUSE

DATA PIPELINE

RESPONSIBILITIES

- Collect
- Process
- Protect
- Access



DATA PIPELINE

DESIRED PROPERTIES

AVAILABLE

DURABLE

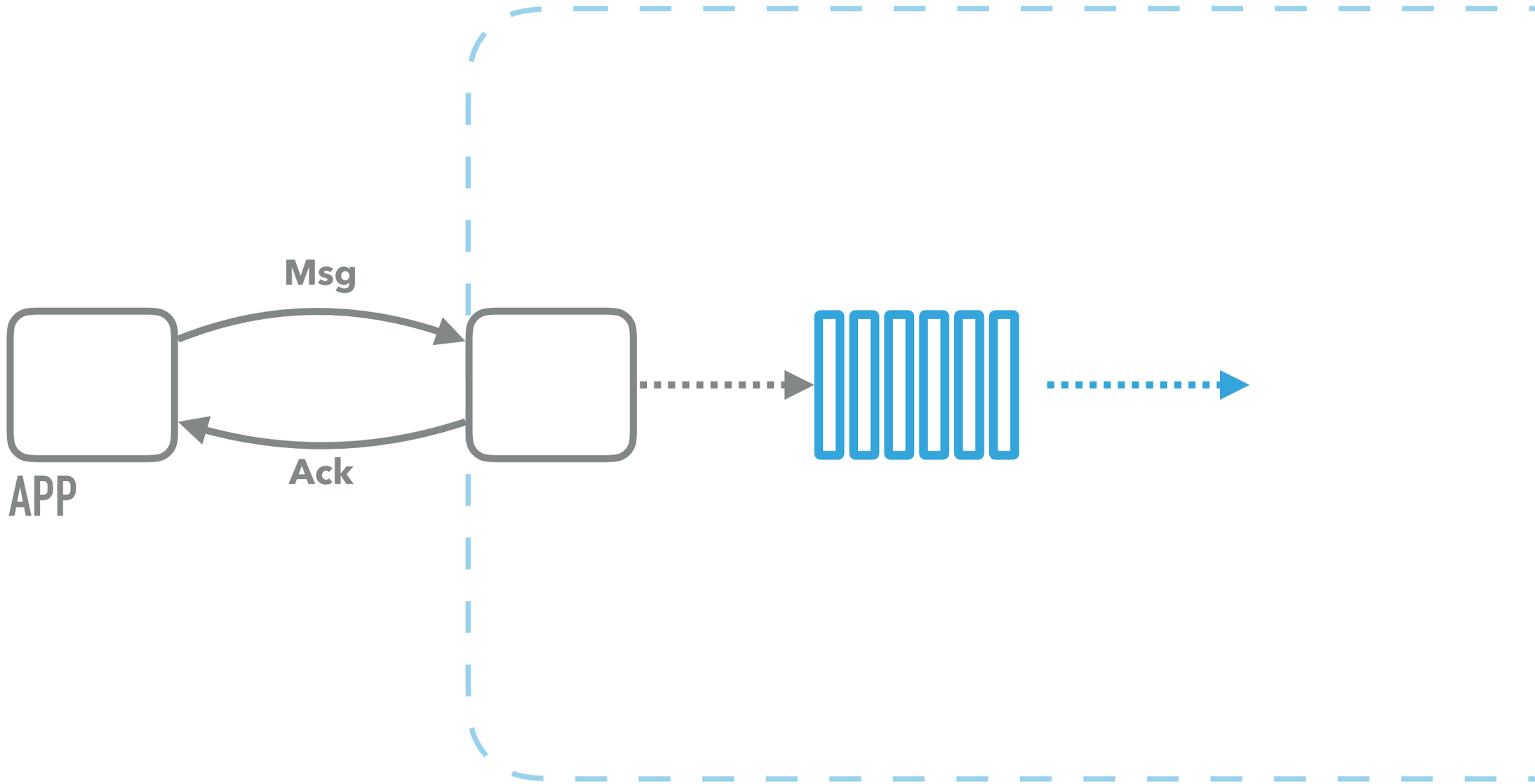
AVAILABILITY

Every request gets a response

Availability	Down time / Year
99%	3.65 days
99.9%	8.77 hours
99.99%	52.60 minutes
99.999%	5.26 minutes

DURABILITY

Guaranteed delivery when acknowledged



DATA

DESIRED PROPERTIES

IMMUTABLE

SCHEMA

IMMUTABLE DATA

- No UPDATE (Immutable data)
- Changes are stored as *sequence* of events
- Current State = Applying the history of changes

IMMUTABLE DATA

UserId	Date	City
10001	2015-07-01	Montréal
10001	2012-09-12	Toronto
10001	2009-07-26	New York
10001	2007-01-01	San Francisco

EVENT-SOURCING

- Event-Sourcing
  - User / Application events ↔ Data events
    - Reliable audit logs
    - Data consistency in multiple destinations
    - Complex queries
- One way of designing data pipeline

SCHEMA

- Enforced schema
- Data is structured
- Avoids data corruption

SERIALIZATION FRAMEWORKS

- Apache Thrift
- Protocol Buffers
- Avro
- ...

PROTOCOL BUFFERS + GRPC

- Serialization framework
- Schema
- gRPC is a binary RPC protocol

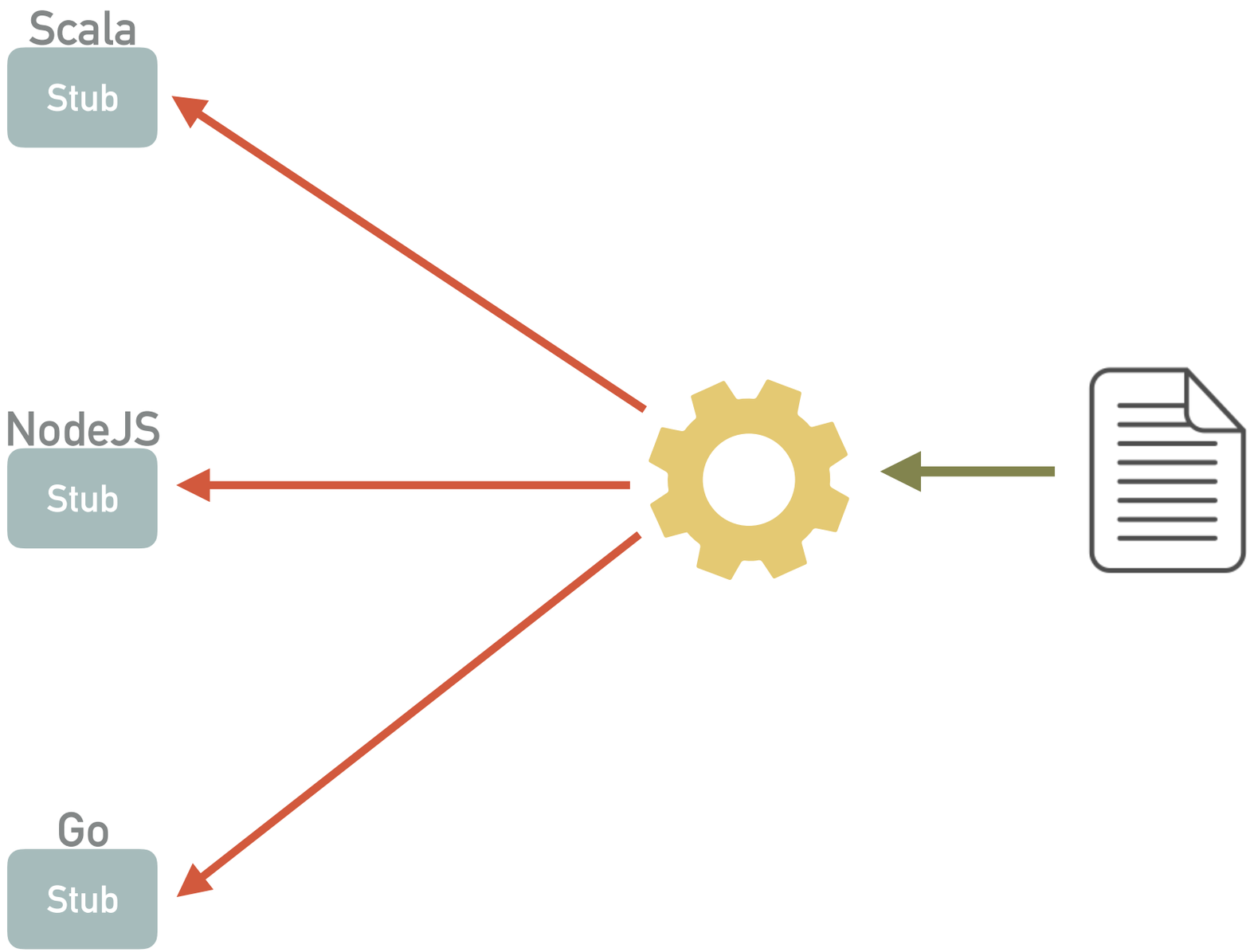
grpc.io

PROTOCOL BUFFERS

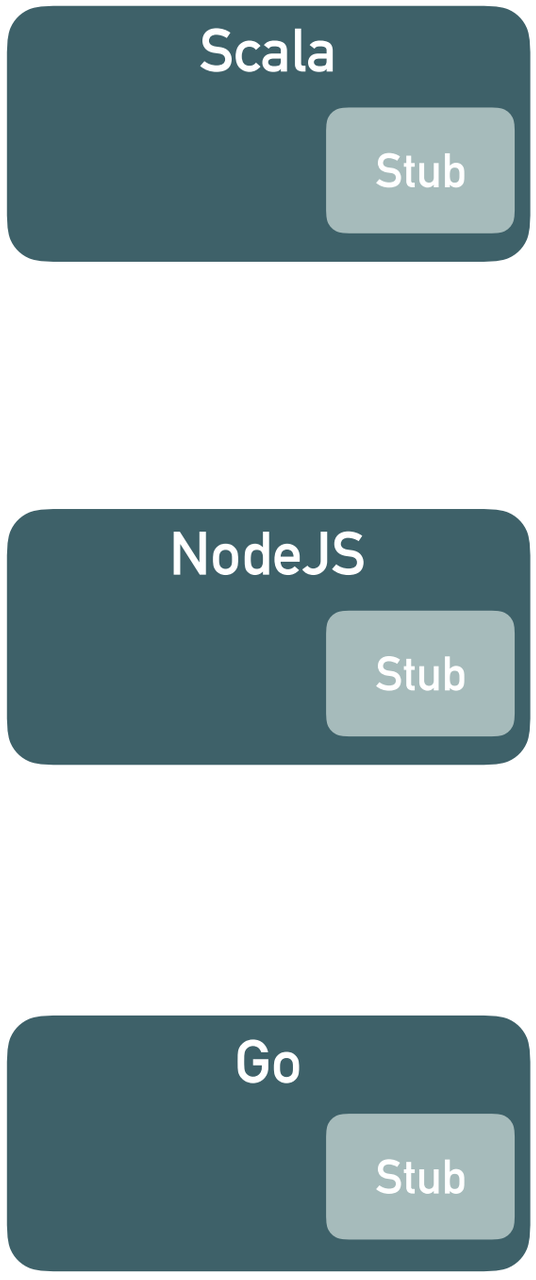
```
// Defining a Message

message Person {
  required string name = 1;
  required int32 id = 2;
  optional string email = 3;
}

// Defining a service

service SearchService {
  rpc Search (SearchRequest) returns (SearchResponse);
}
```

Protobuf v2 | developers.google.com

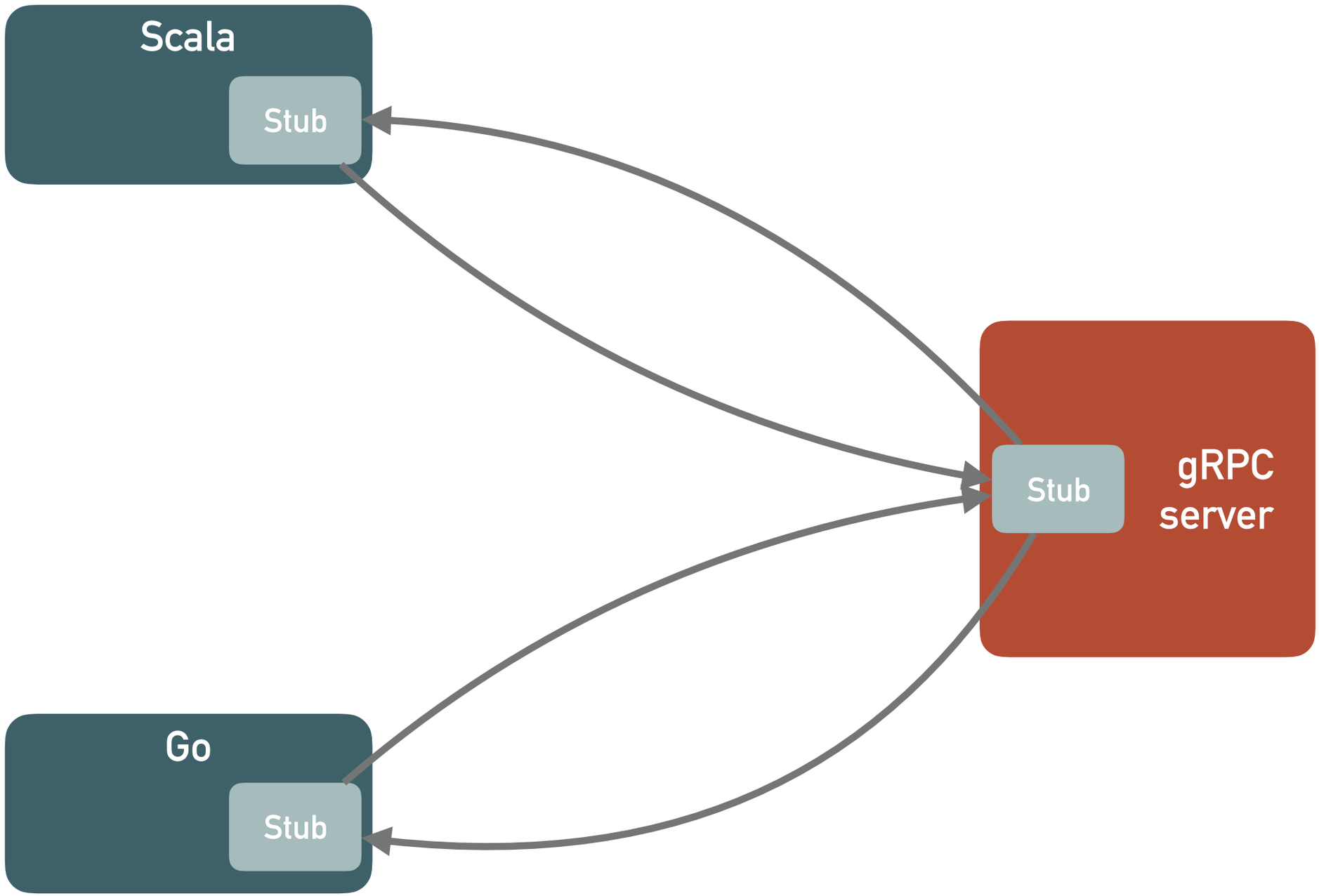
PROTOCOL BUFFERS COMPILER



PROTOCOL BUFFERS COMPILER

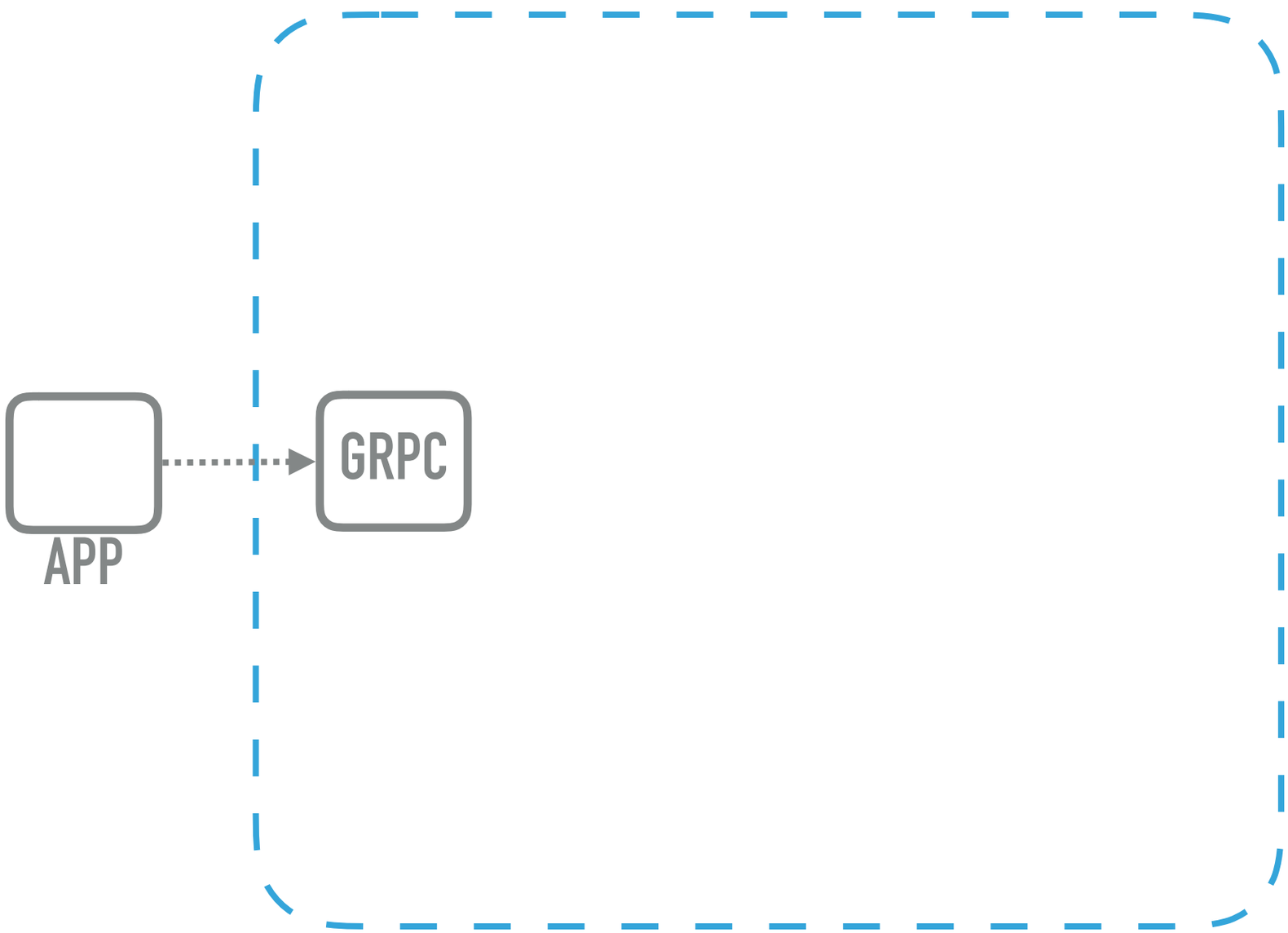


PROTOCOL BUFFERS + GRPC

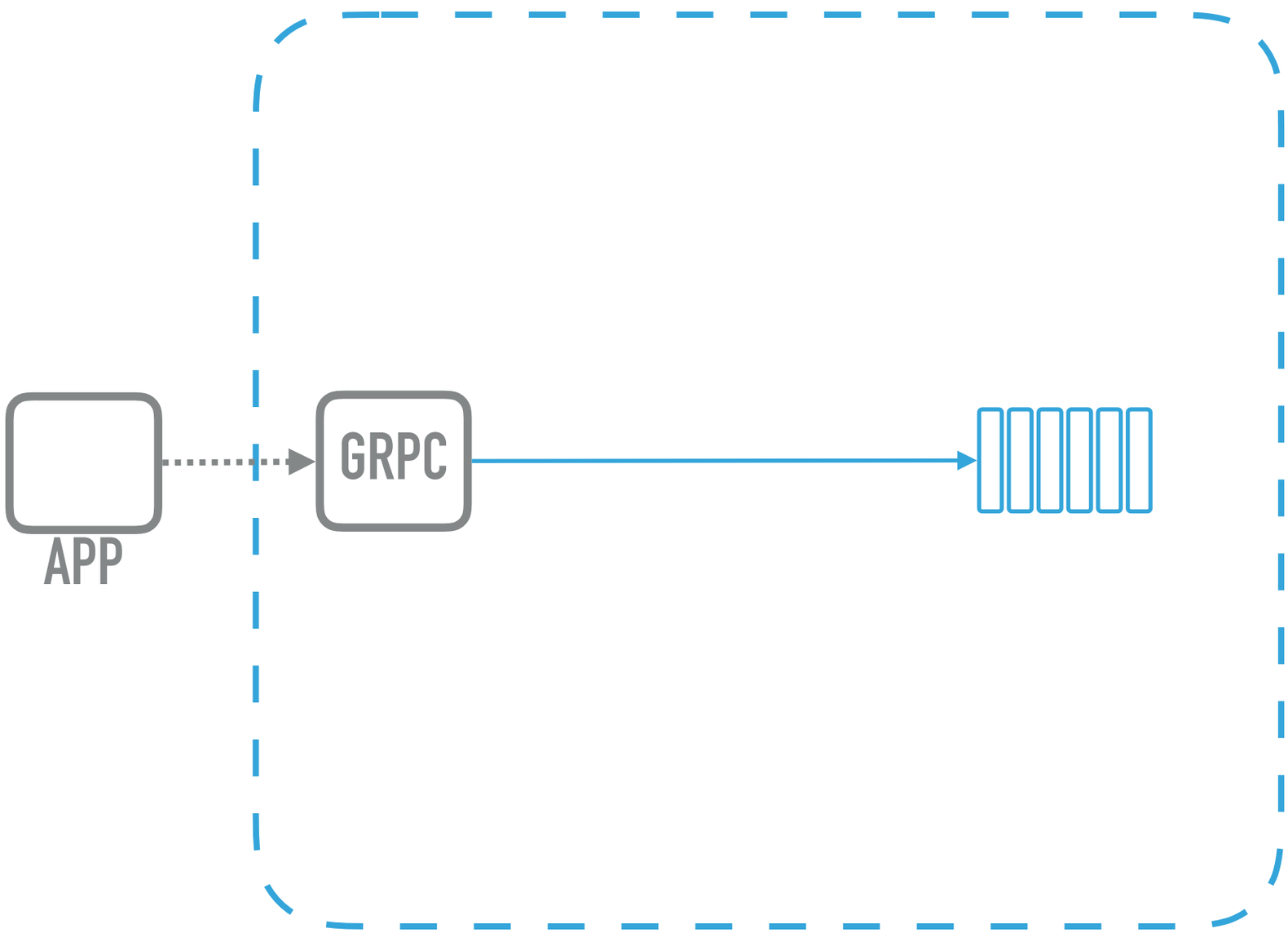


Handling gRPC traffic

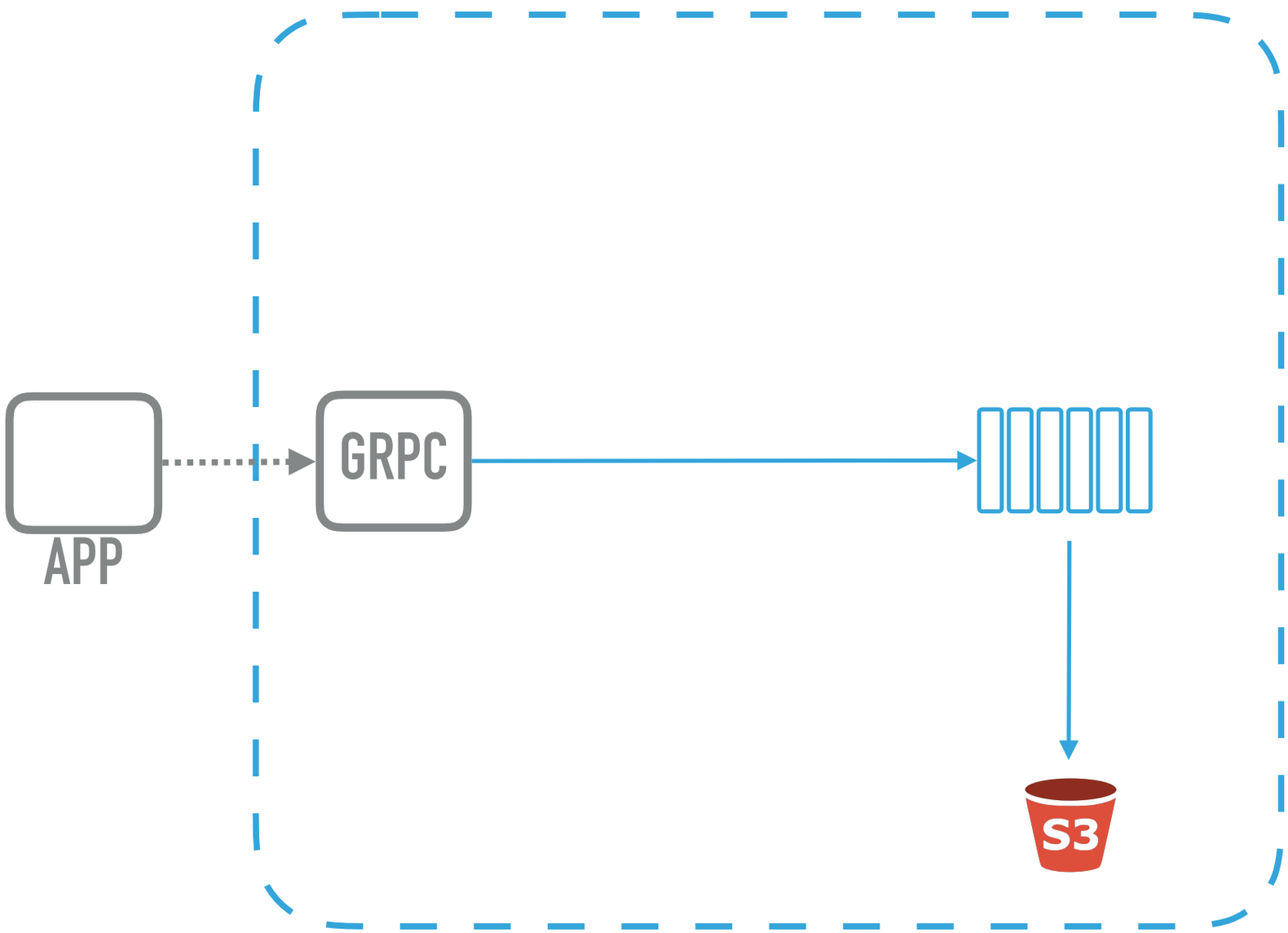
Handling gRPC traffic



Persisting gRPC traffic



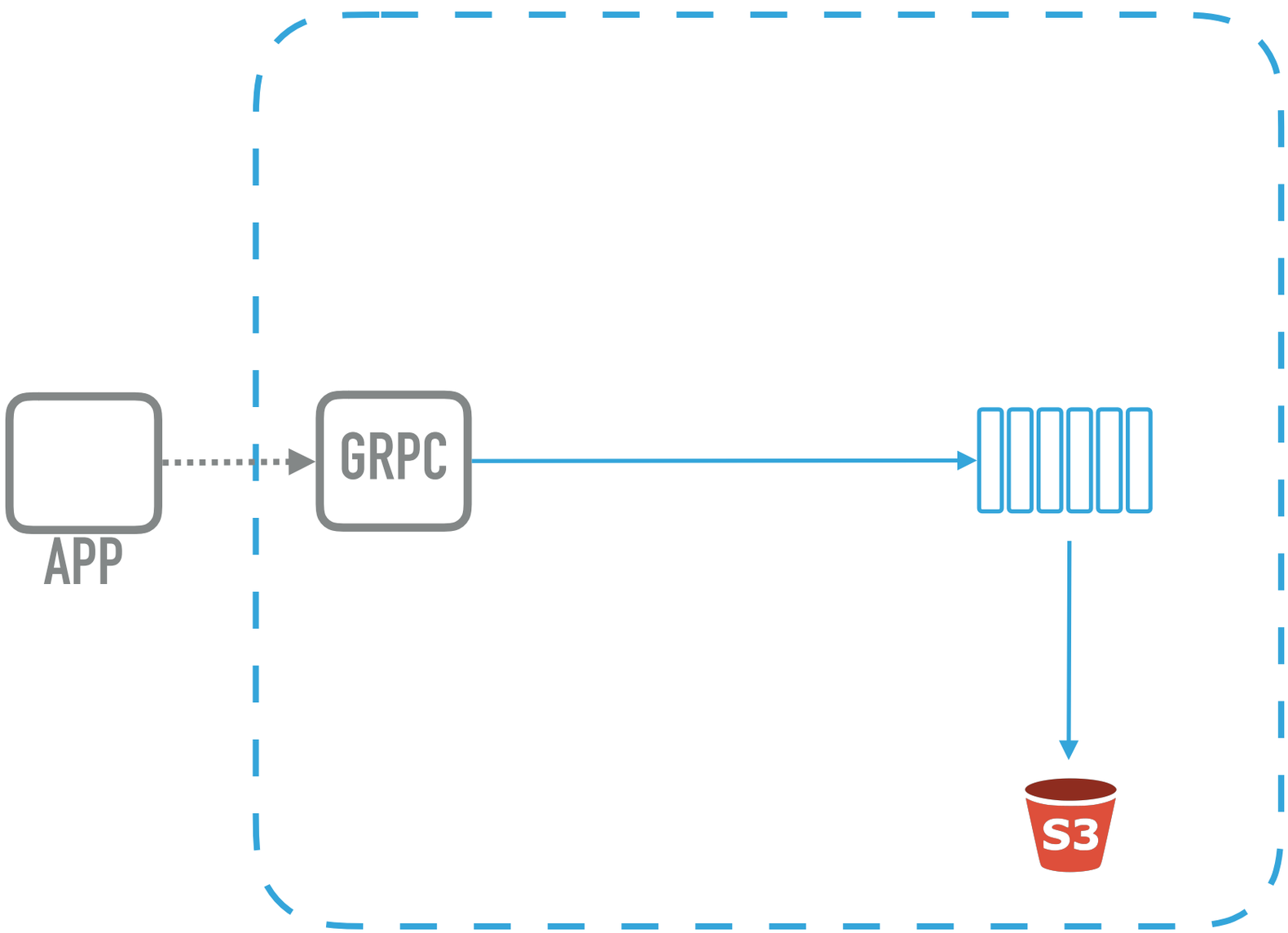
Persisting gRPC traffic



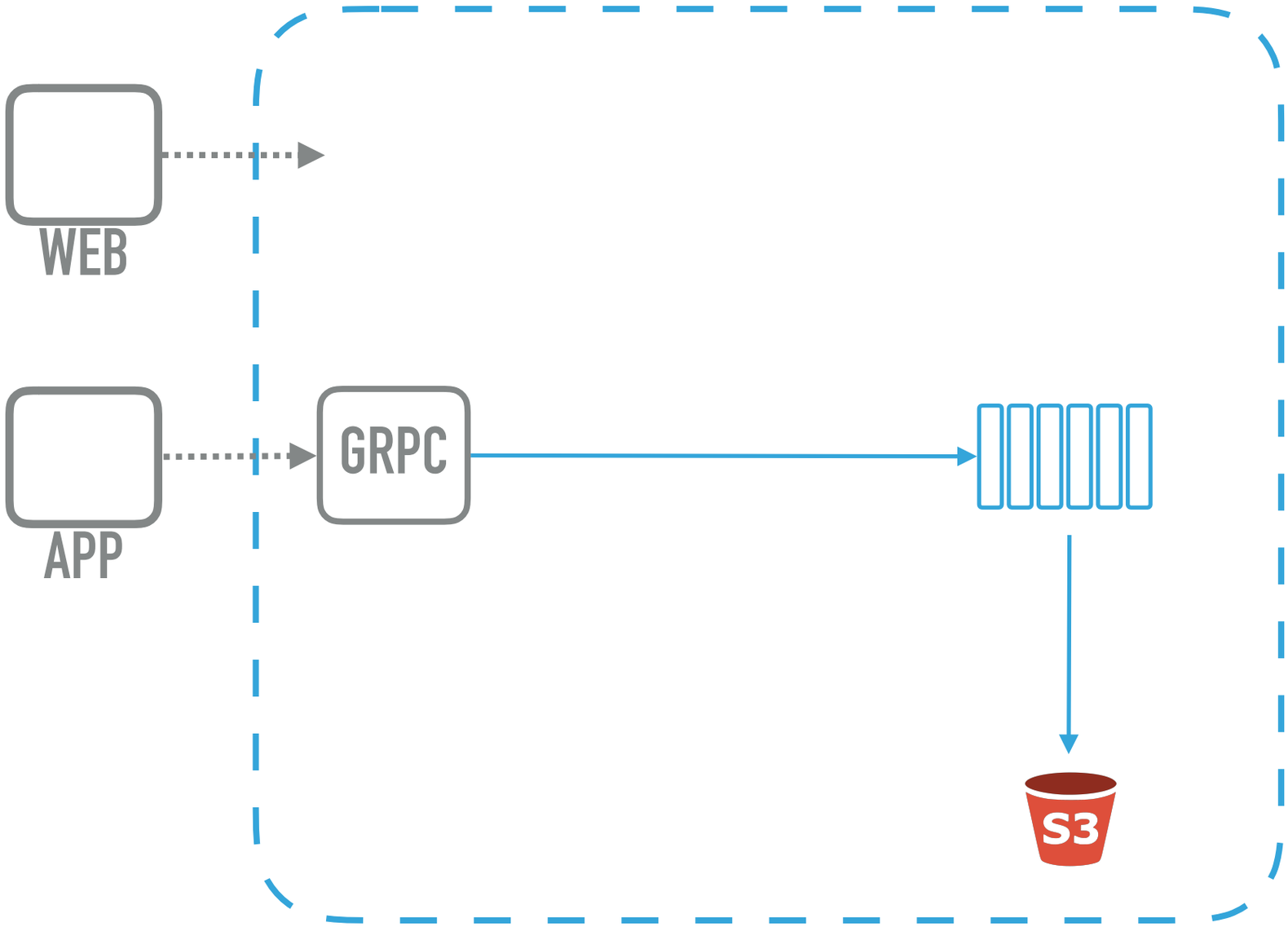
GRPC ENDPOINT – TECHNOLOGIES

- Go application
- AWS Elastic Container Service (ECS)
- Network Load Balancer (NLB)
- AWS PrivateLink
- Kinesis stream
- Kinesis Firehose to S3 (disaster recovery)

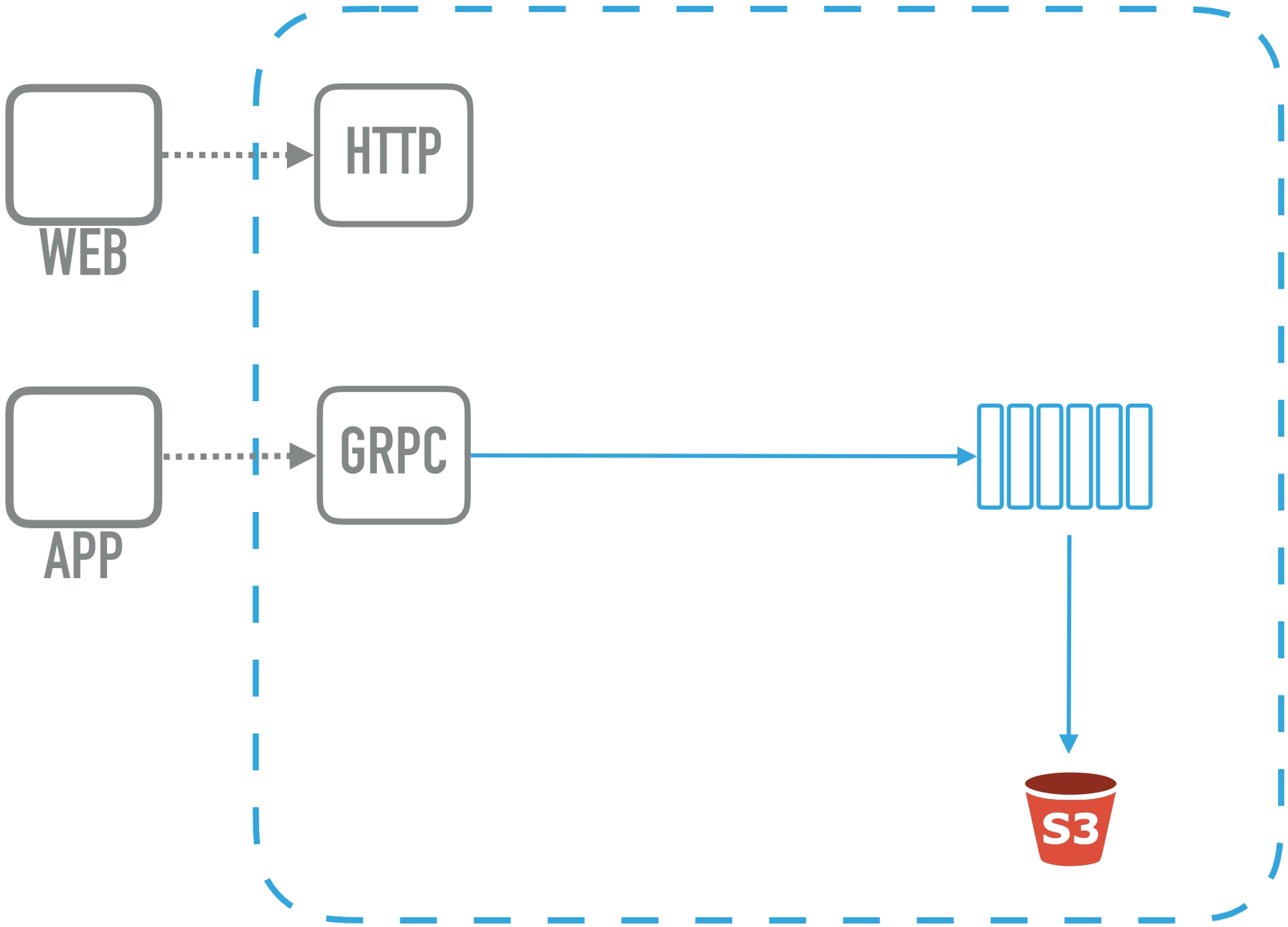
Persisting gRPC traffic



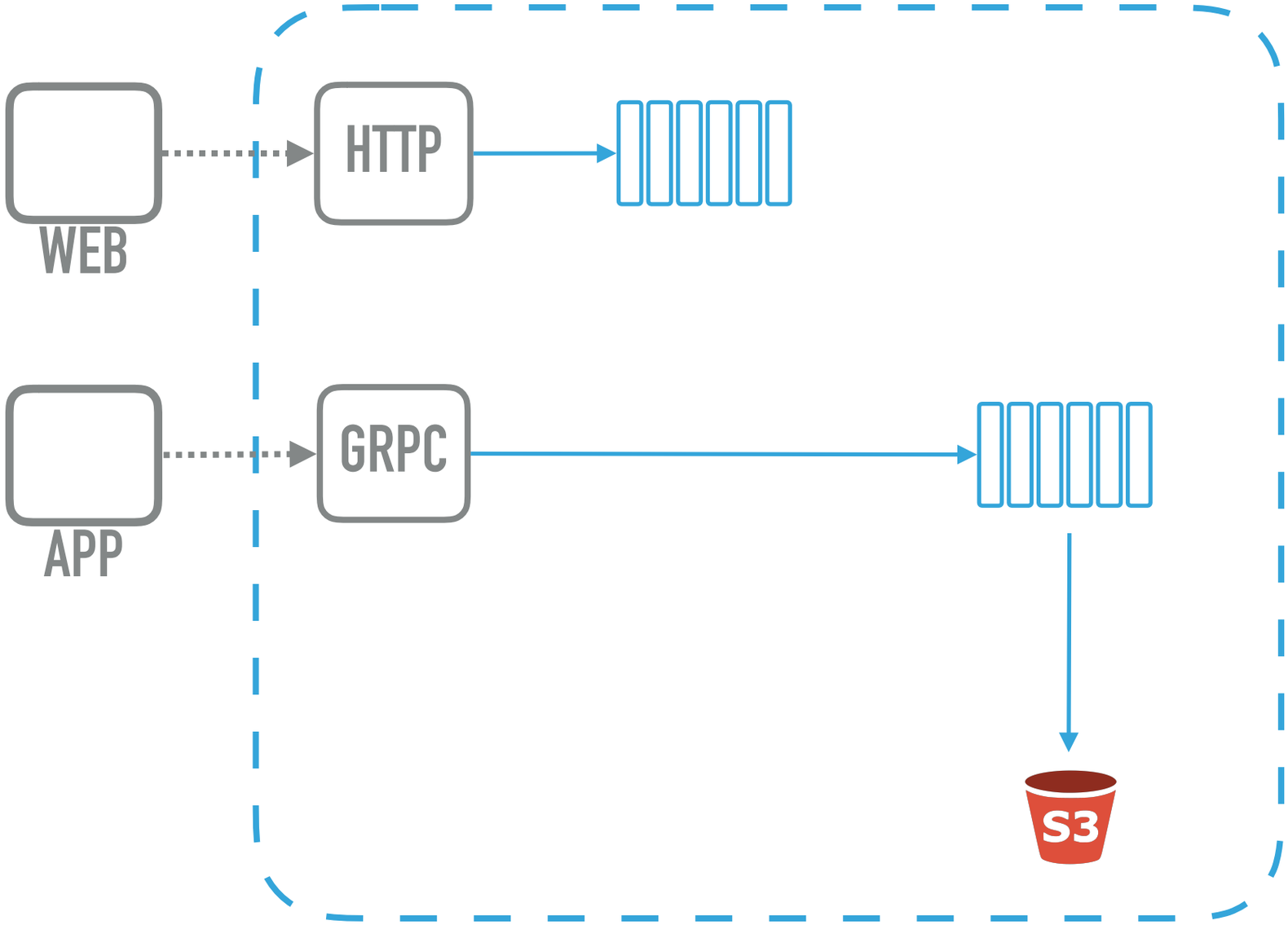
Handling web/http traffic



Handling web/http traffic



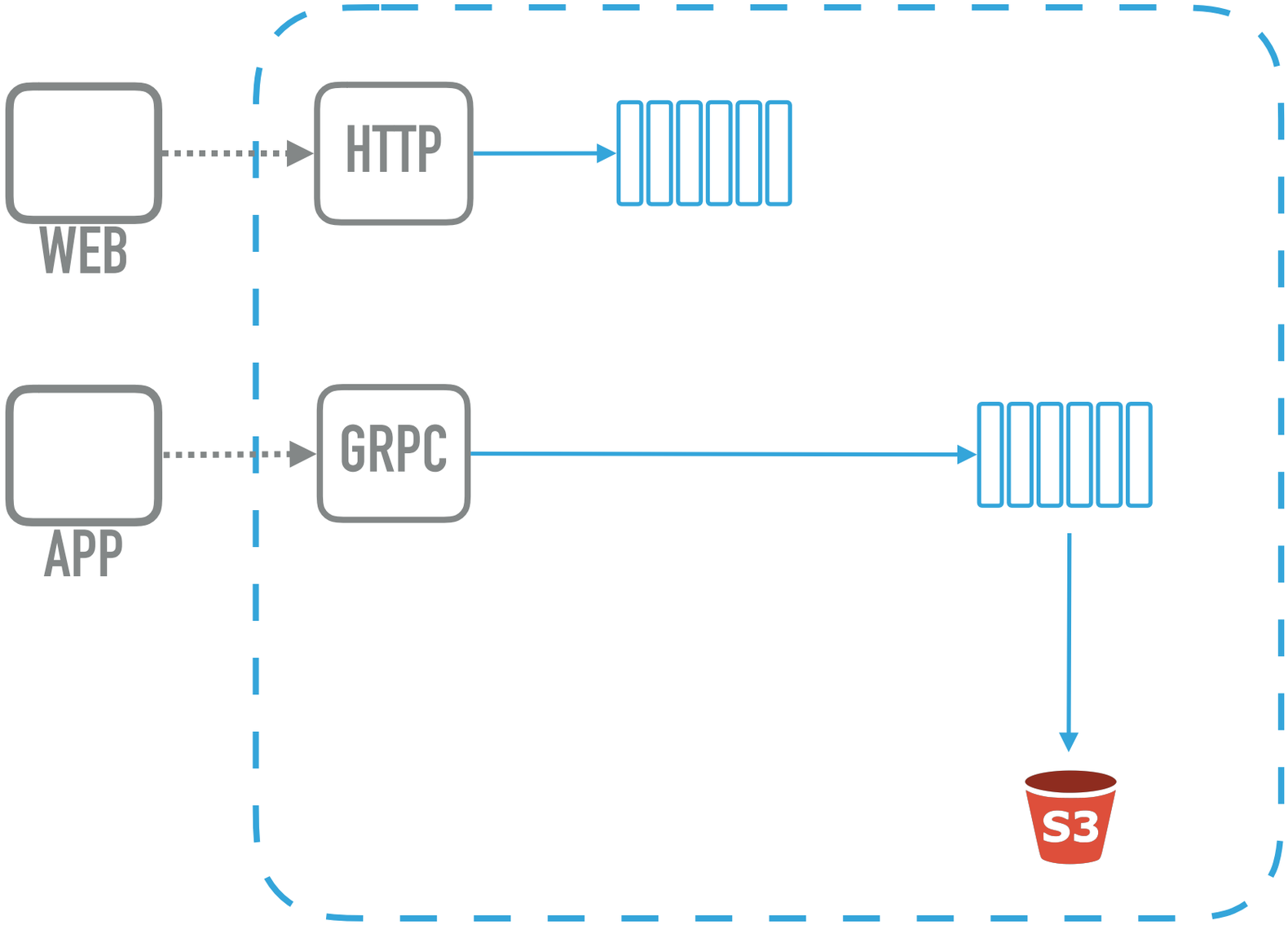
Persisting web/http traffic



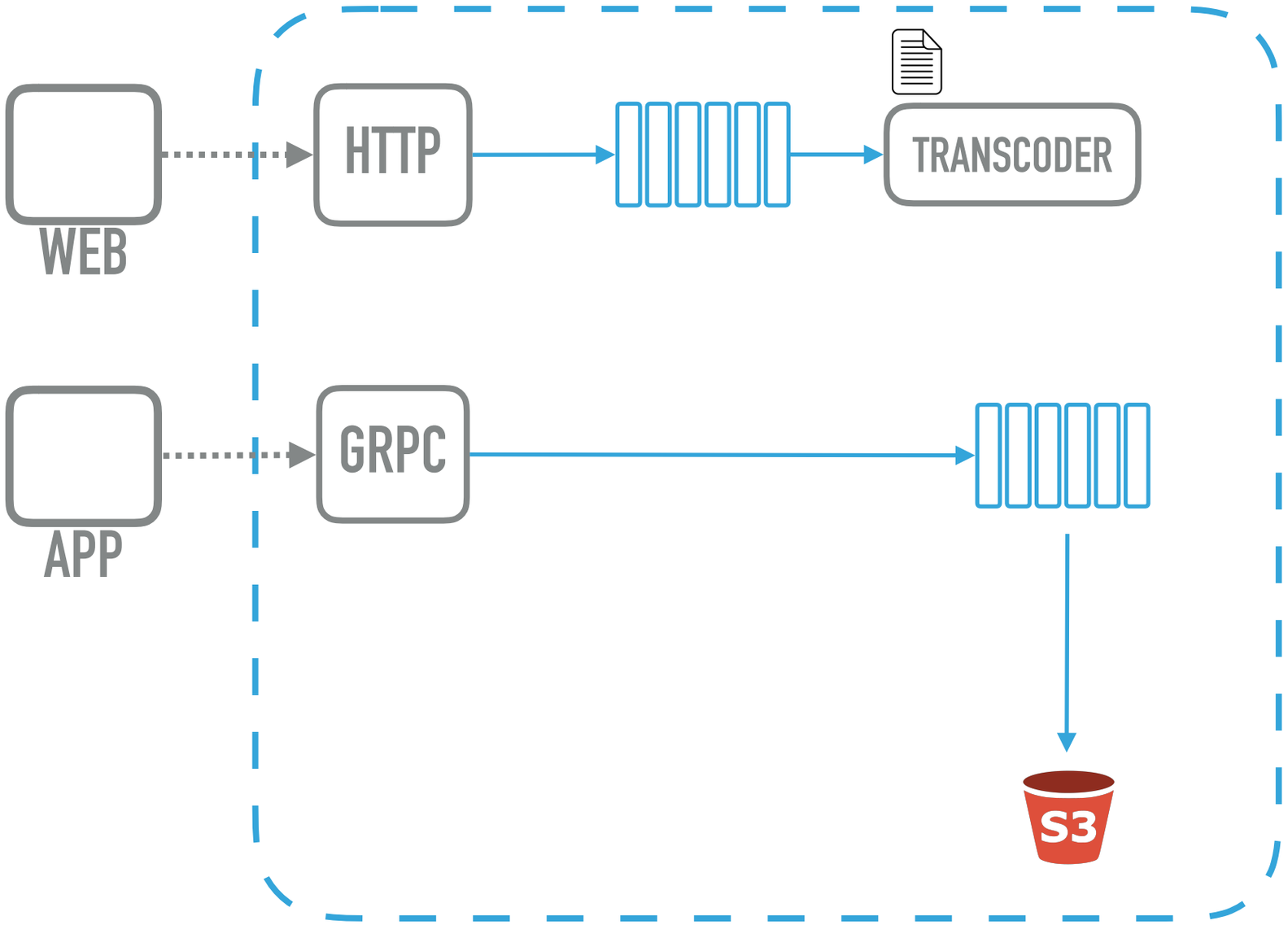
HTTP ENDPOINT – TECHNOLOGIES

- Go application
- AWS Elastic Container Service (ECS)
- Application Load Balancer (ALB)
- Kinesis stream

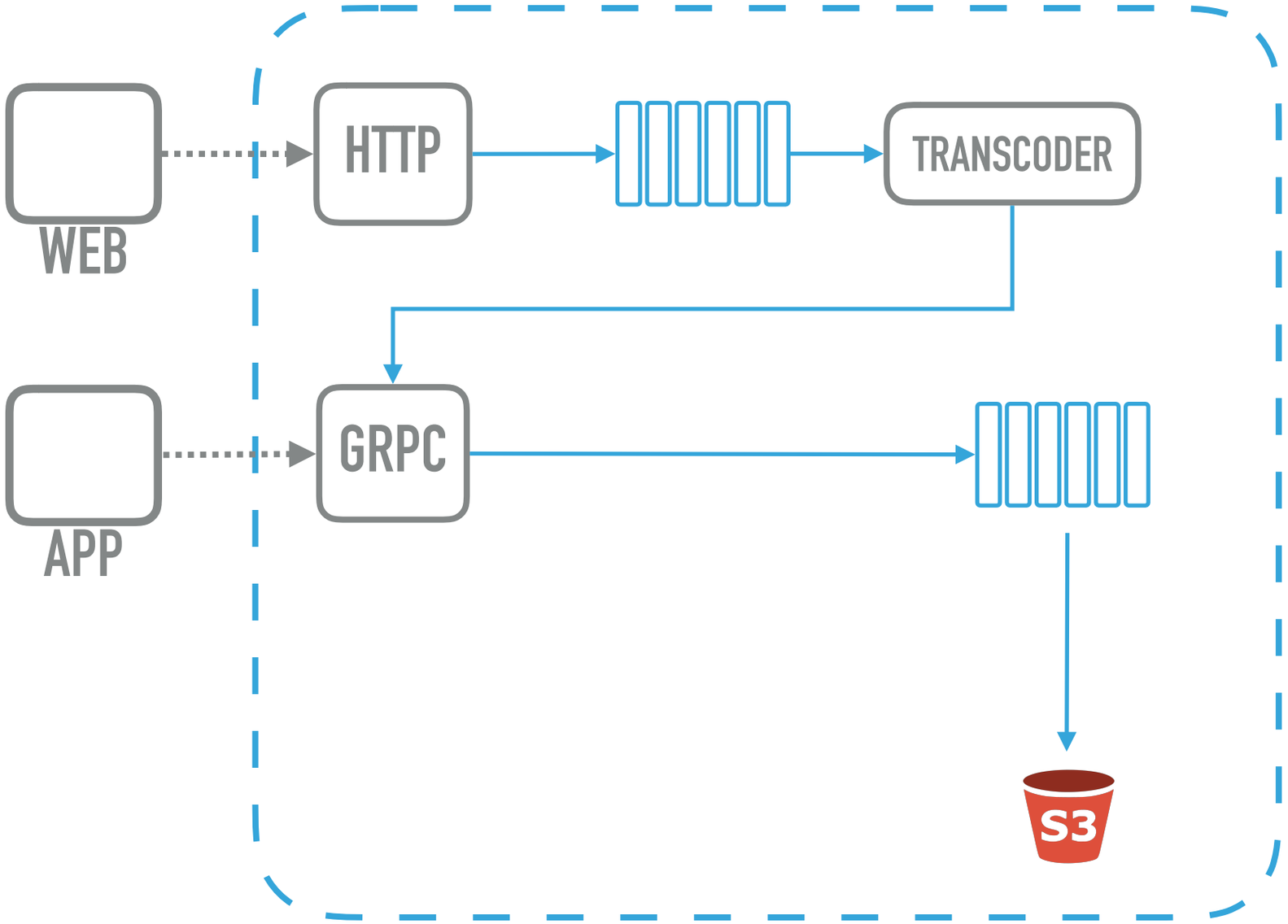
Persisting web/http traffic



Converting JSON to Protobuf



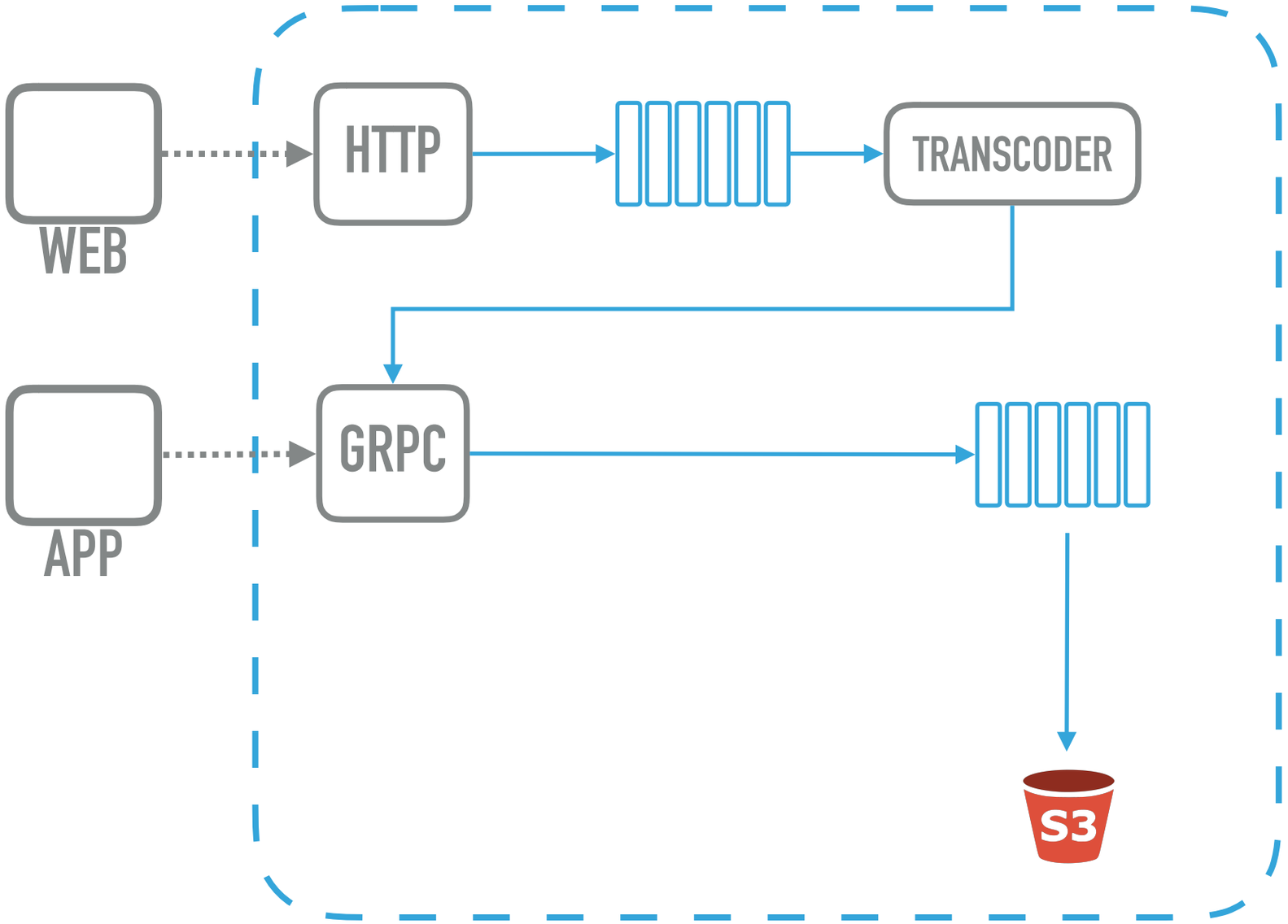
Pushing converted data to gRPC



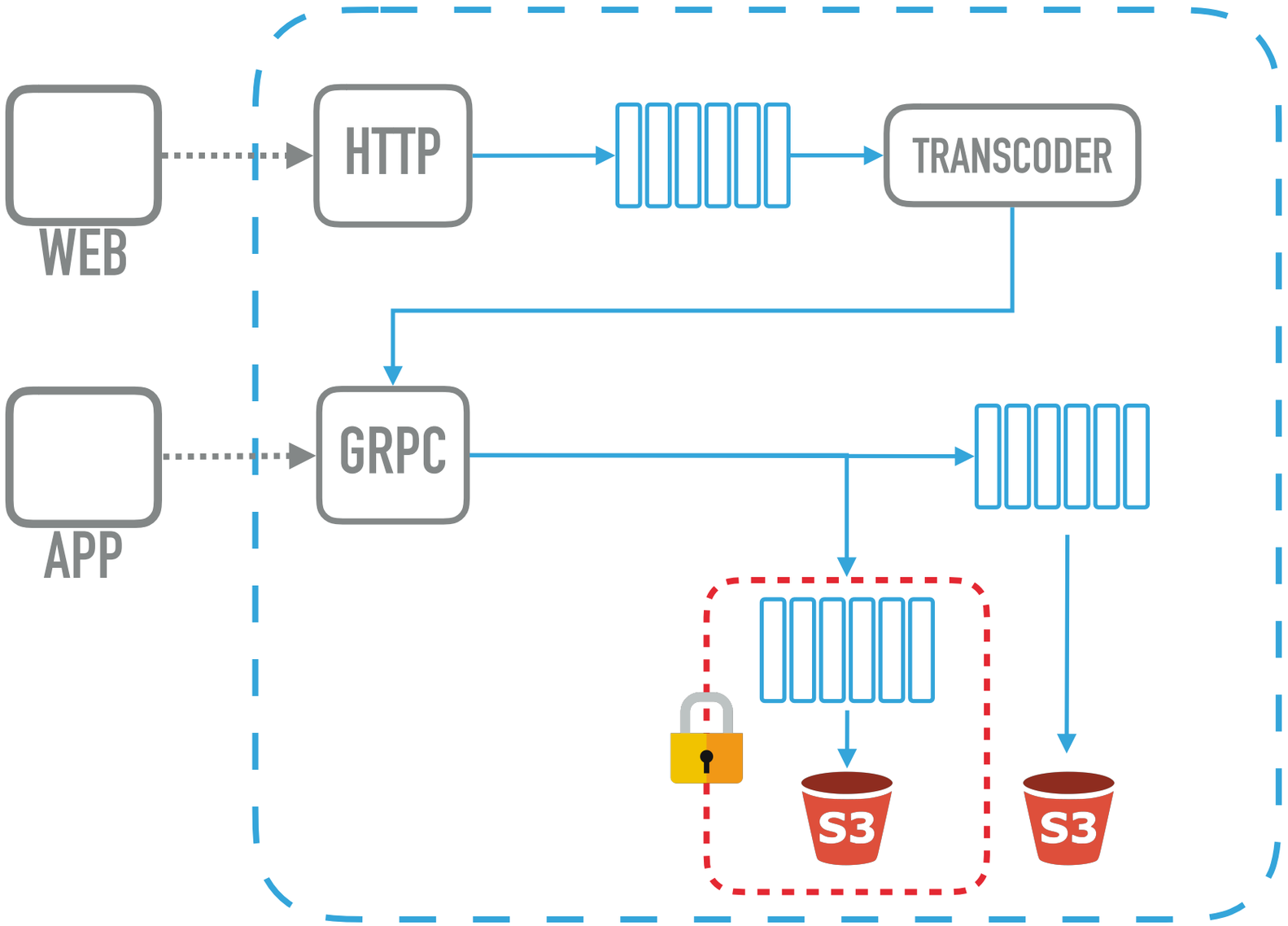
TRANSCODER – TECHNOLOGIES

- Scala application
- AWS Elastic Container Service (ECS)
- Kinesis Enhanced Fanout

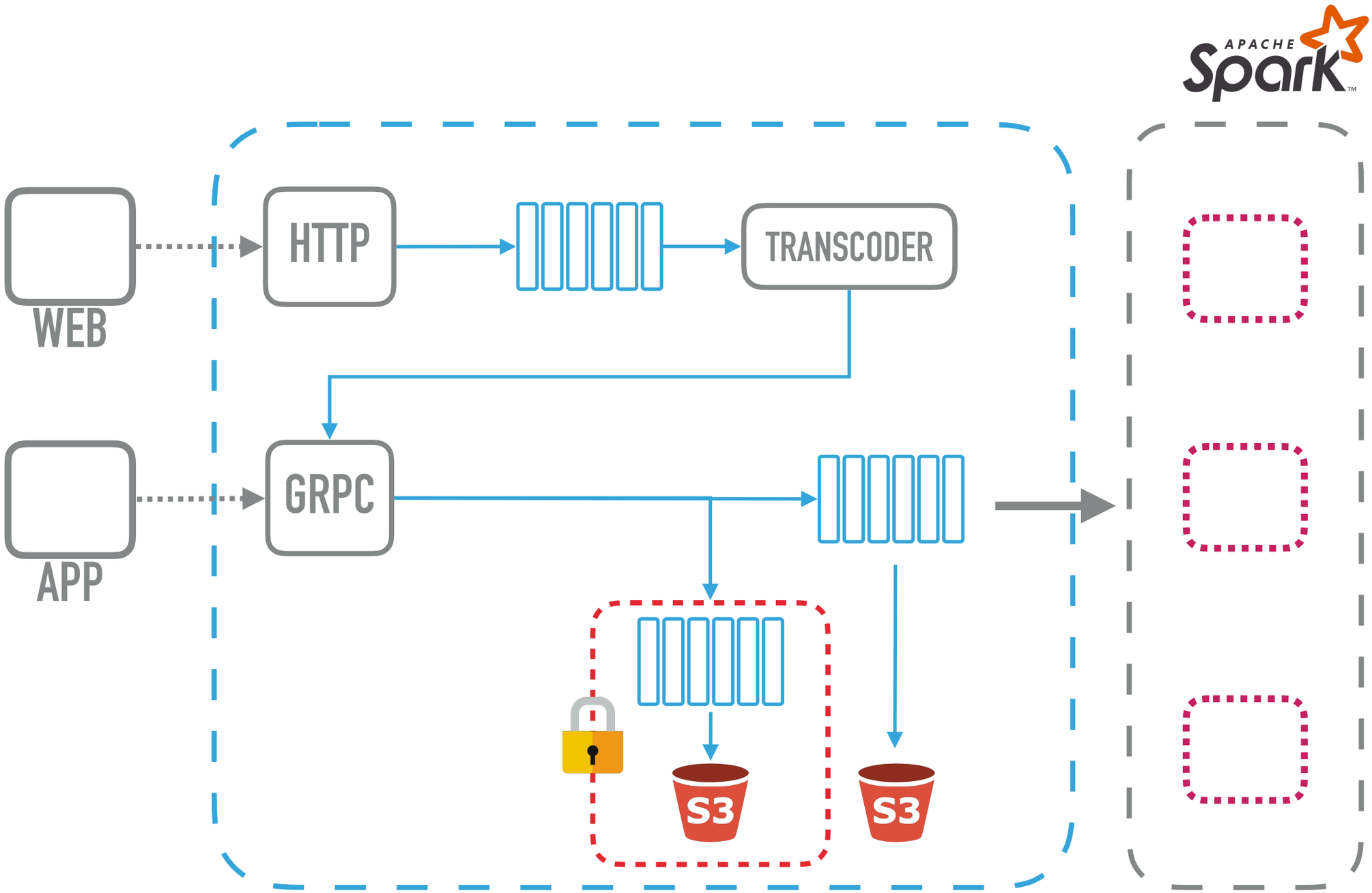
Pushing converted data to gRPC



Handling Sensitive data - Redaction

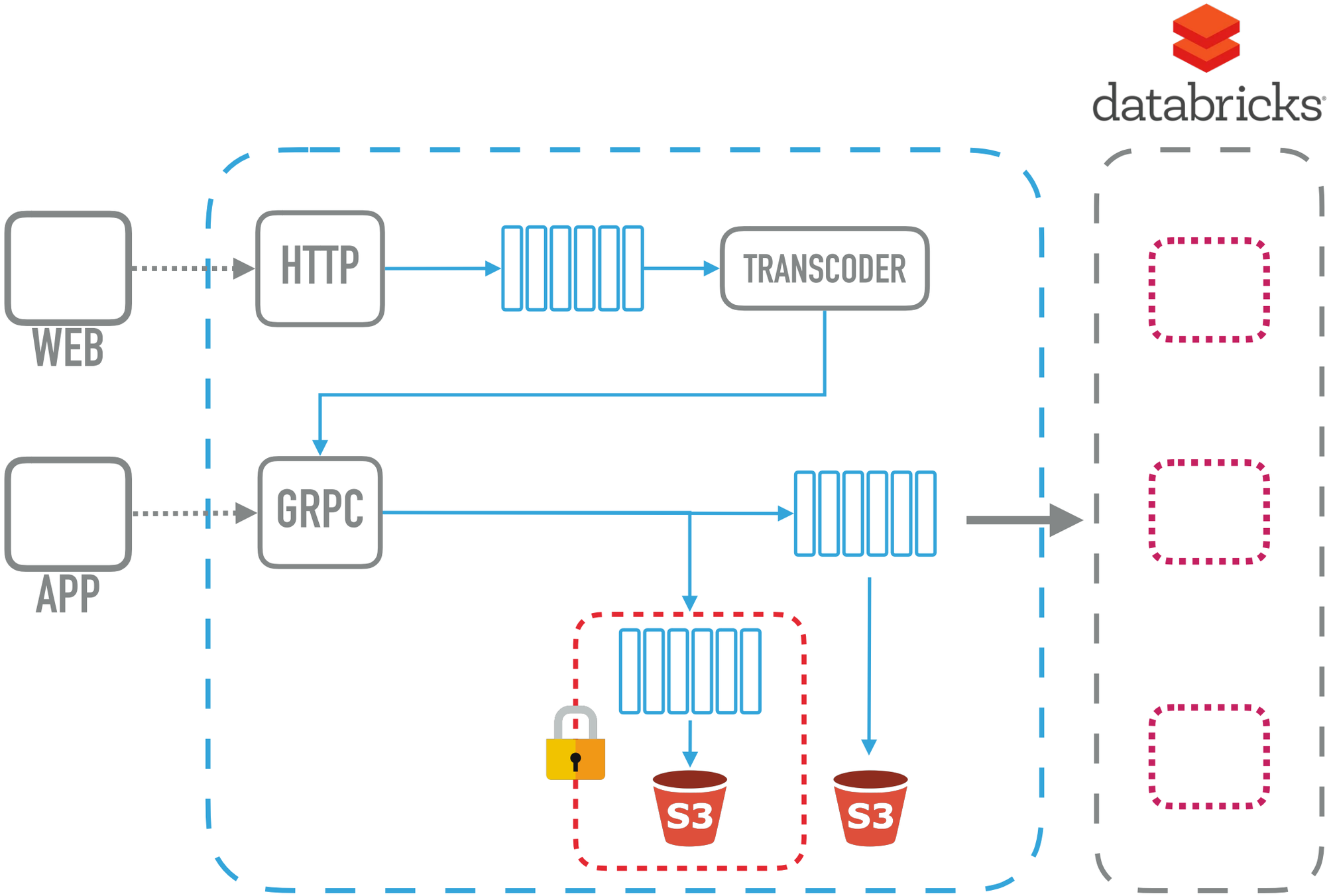


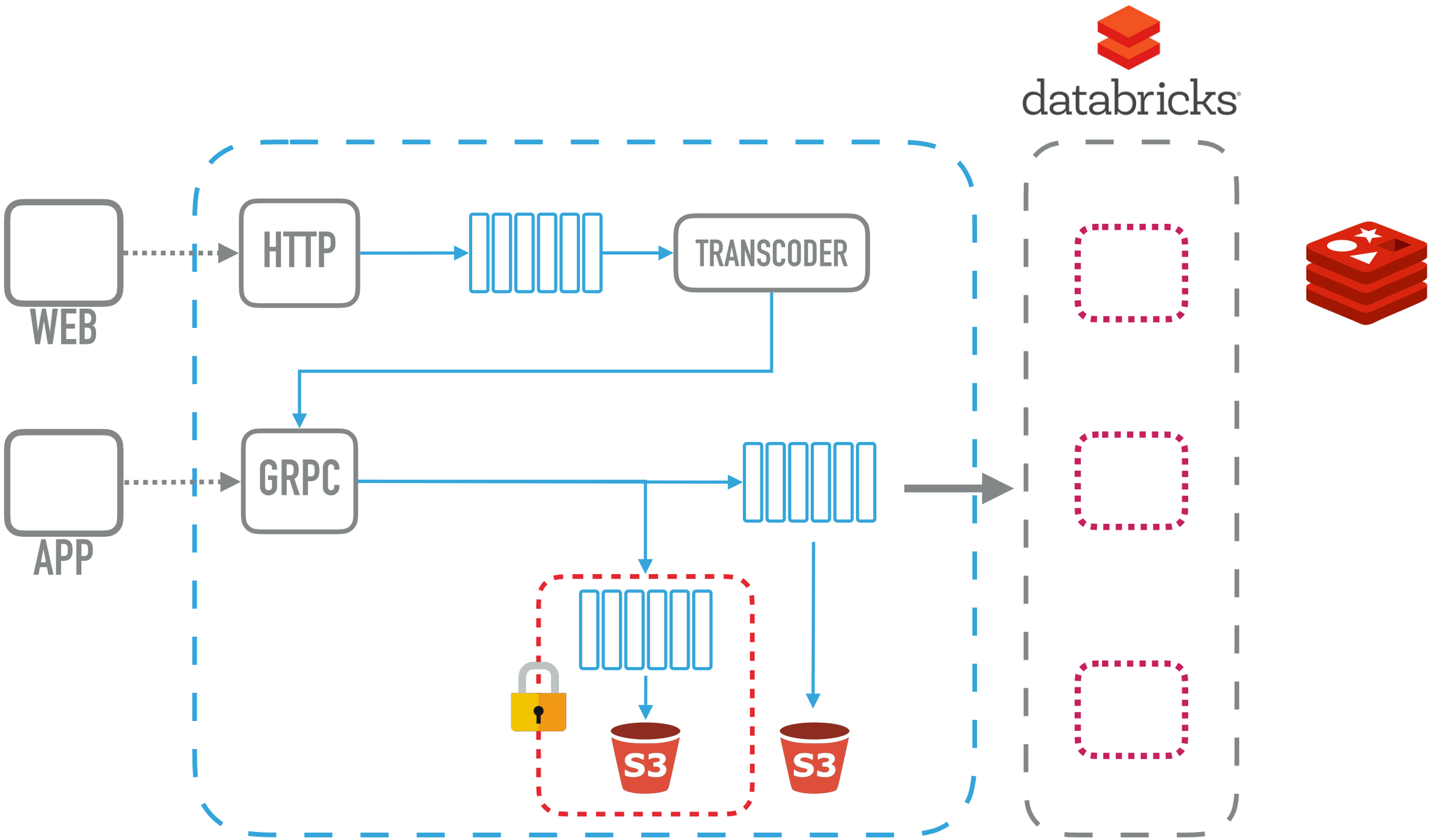
Consuming data, ETL - Apache Spark

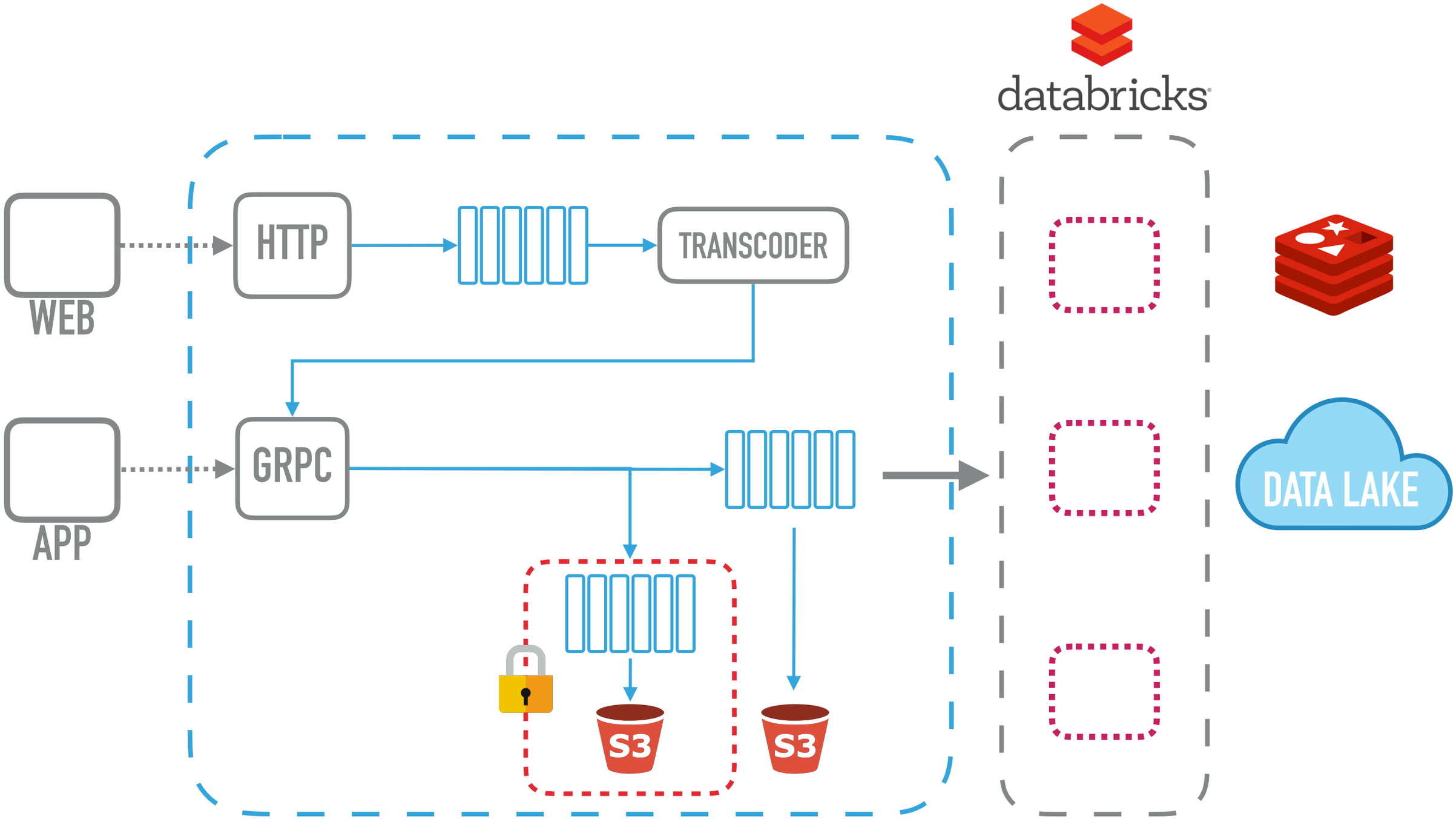


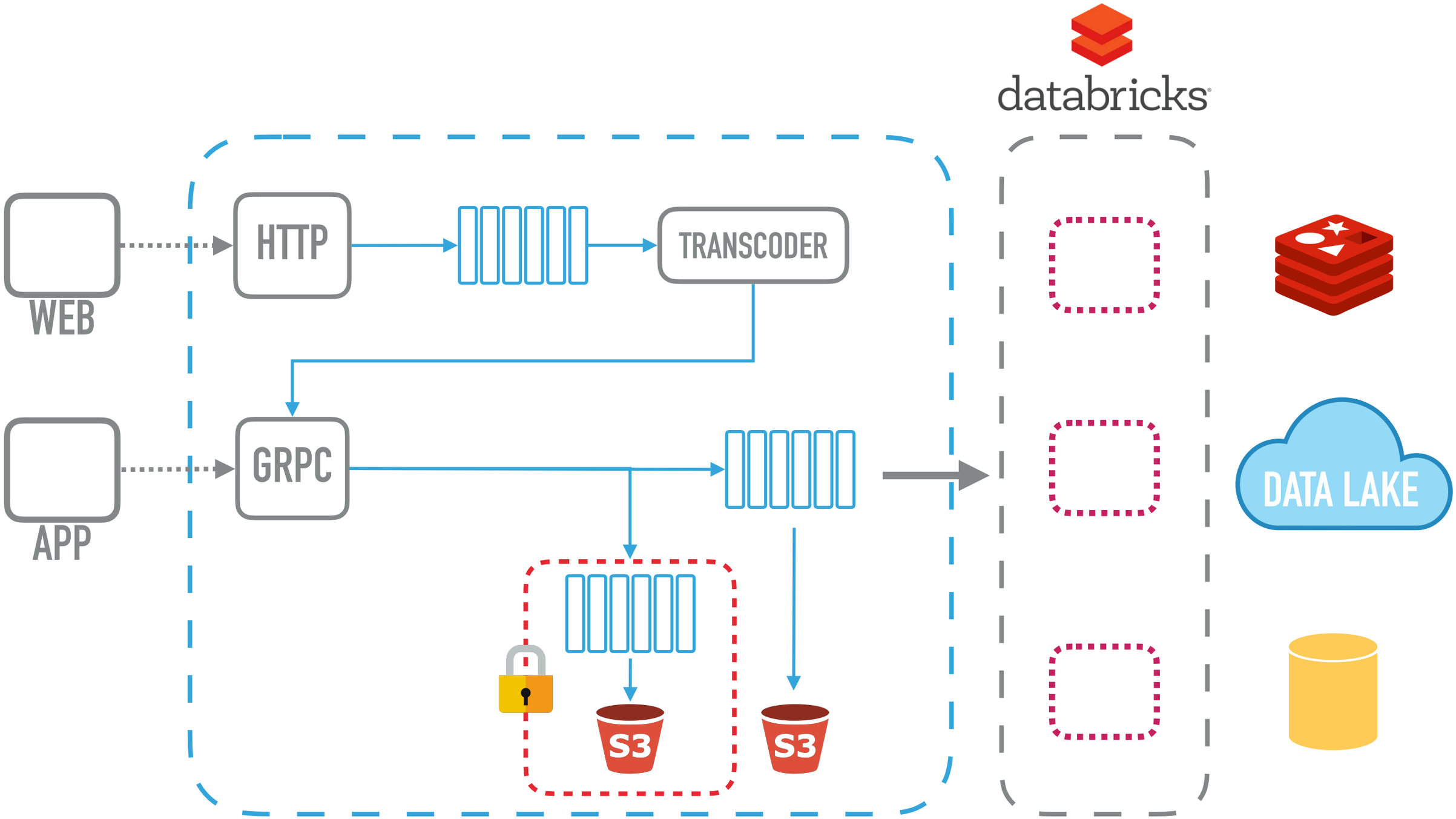
APACHE SPARK

- Apache Spark through **DataBricks**
  - Extensive tools
  - Runs on your AWS infrastructure (VPC)
  - Ability to extend with custom tools
  - Easy to use


databricks
WEB
APP
HTTP
GRPC
TRANSCODER
S3
S3

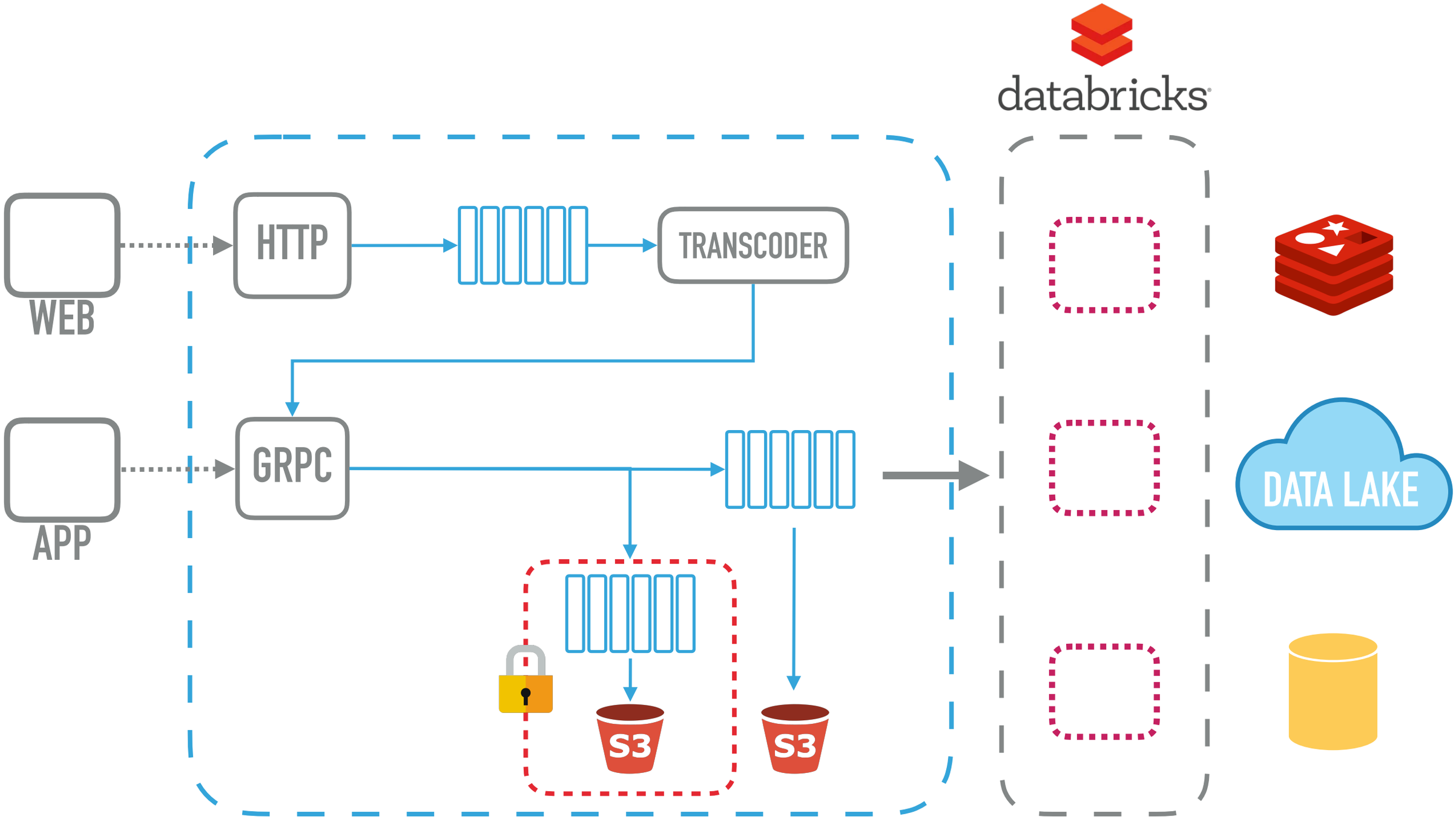

databricks
WEB
APP
HTTP
GRPC
TRANSCODER
S3
S3


databricks
WEB
APP
HTTP
GRPC
TRANSCODER
S3
S3
DATA LAKE


databricks
WEB
APP
HTTP
GRPC
TRANSCODER
S3
S3
DATA LAKE

TECHNOLOGIES

- Redis
- RedShift Spectrum
- Parquet
- S3 - multi-tenant storage
- AWS Relational Database Service (RDS)


databricks
WEB
APP
HTTP
GRPC
TRANSCODER
S3
S3
DATA LAKE

6 LESSONS

LEARNED

OBSERVABILITY

1/6

Honest Status Page
@honest_update

We replaced our monolith with micro services so that every outage could be more like a murder mystery.

7:10 PM - 7 Oct 2015

3,010 Retweets **2,627** Likes

@honest_update

MONITORING

- Golden metrics
  - Latency
  - Traffic
  - Errors
  - Saturation

MONITORING

- Analyzing long-term trends
  - Predict future capacity
  - Detect anomalies in trends
- Symptoms vs. Causes
  - What's broken & why

MONITORING

- Black box monitoring
  - Periodic test message
  - Measure end-to-end
- White box monitoring
  - Monitor individual components

DISTRIBUTED TRACING

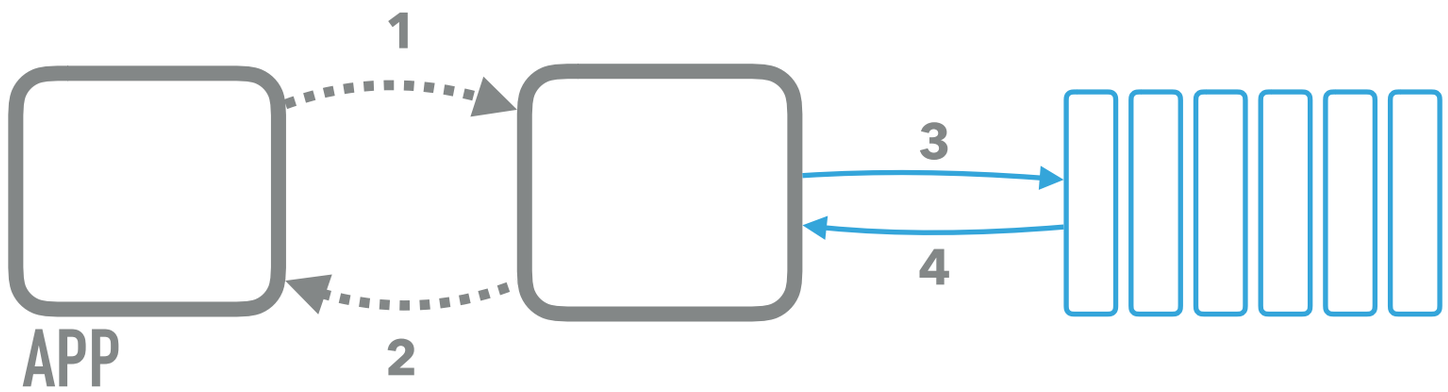
DISTRIBUTED TRACING

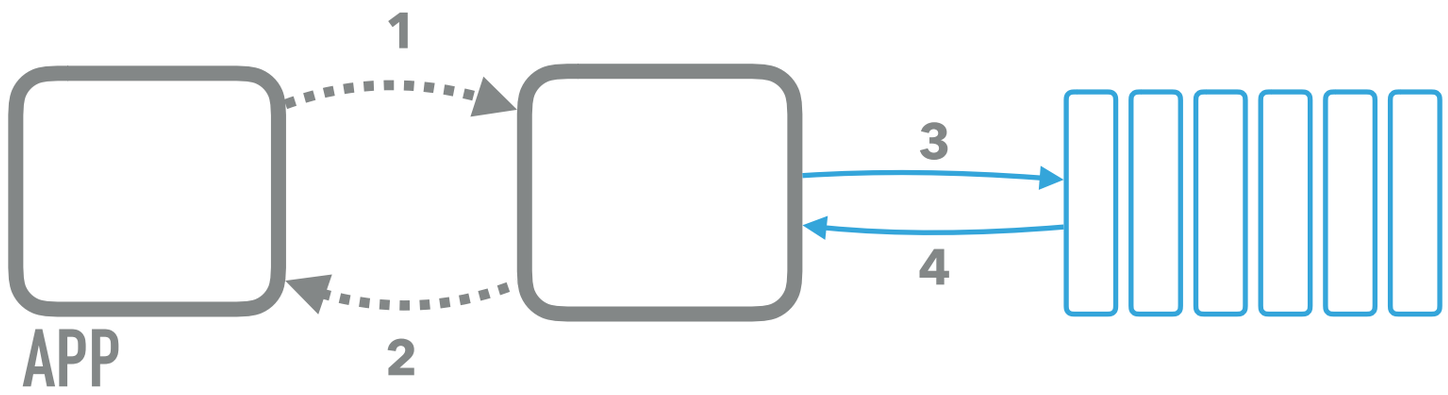
- Visibility to system over time
- Locally aggregated logs
- Centralized logs

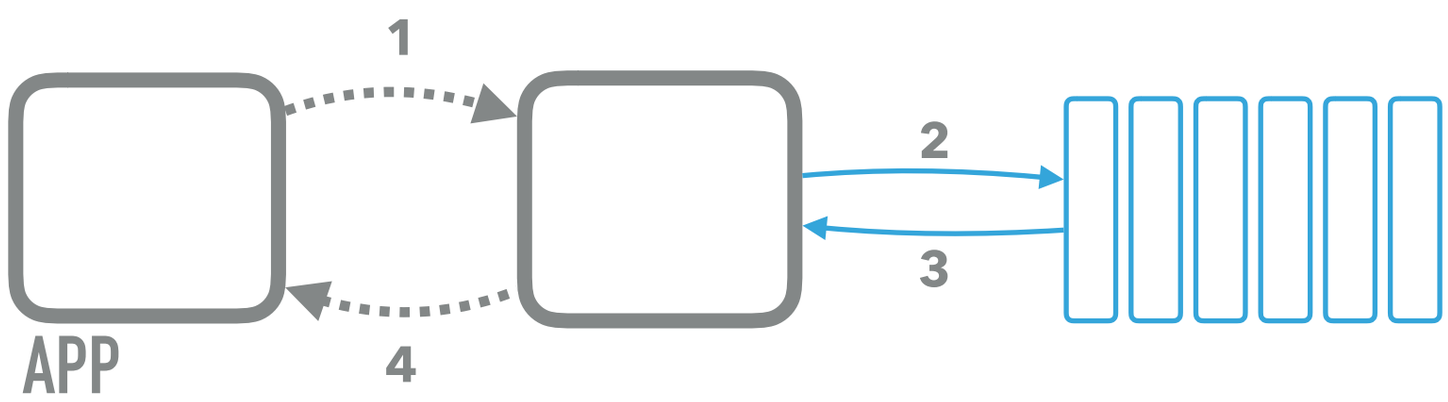
ALERTING

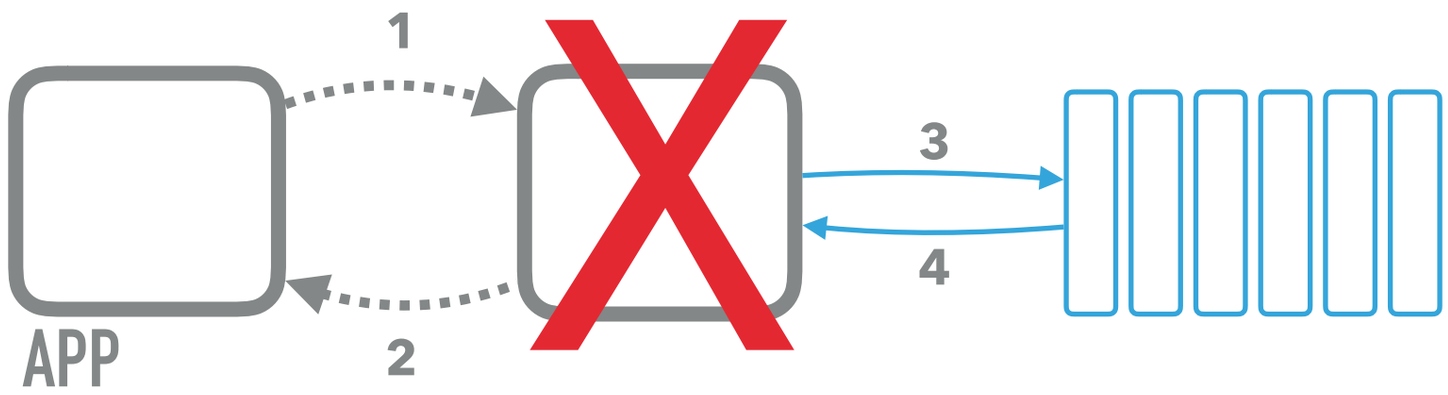
- Define thresholds
- Define appropriate resolution
- Signal vs. Noise
- Define escalation paths

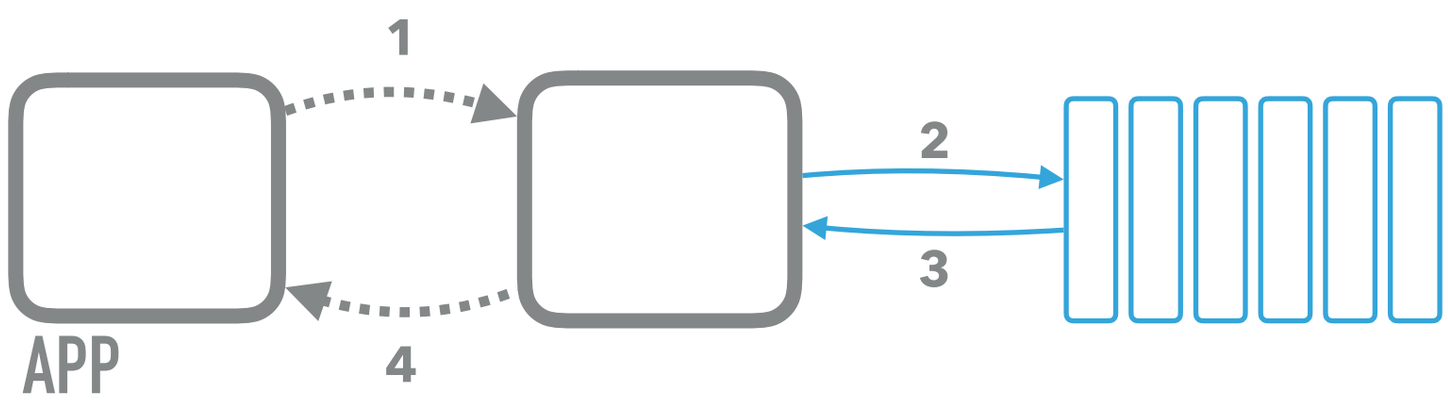
DURABILITY


1
APP
2
3
4

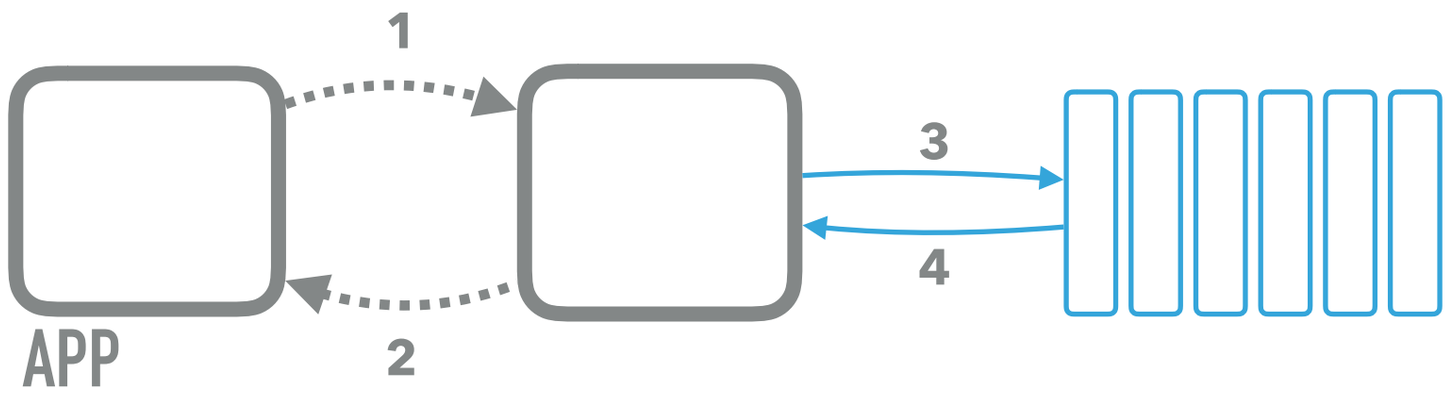

1
APP
2
3
4


1
APP
4
2
3

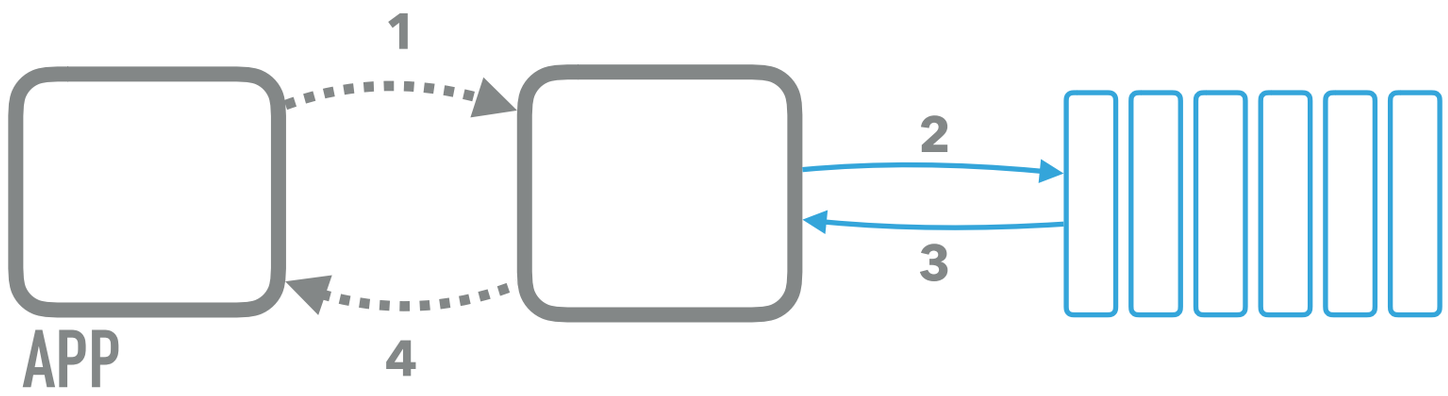

1
APP
2
3
4


1
APP
4
2
3

Fire & Forget



Guaranteed Delivery



DURABILITY

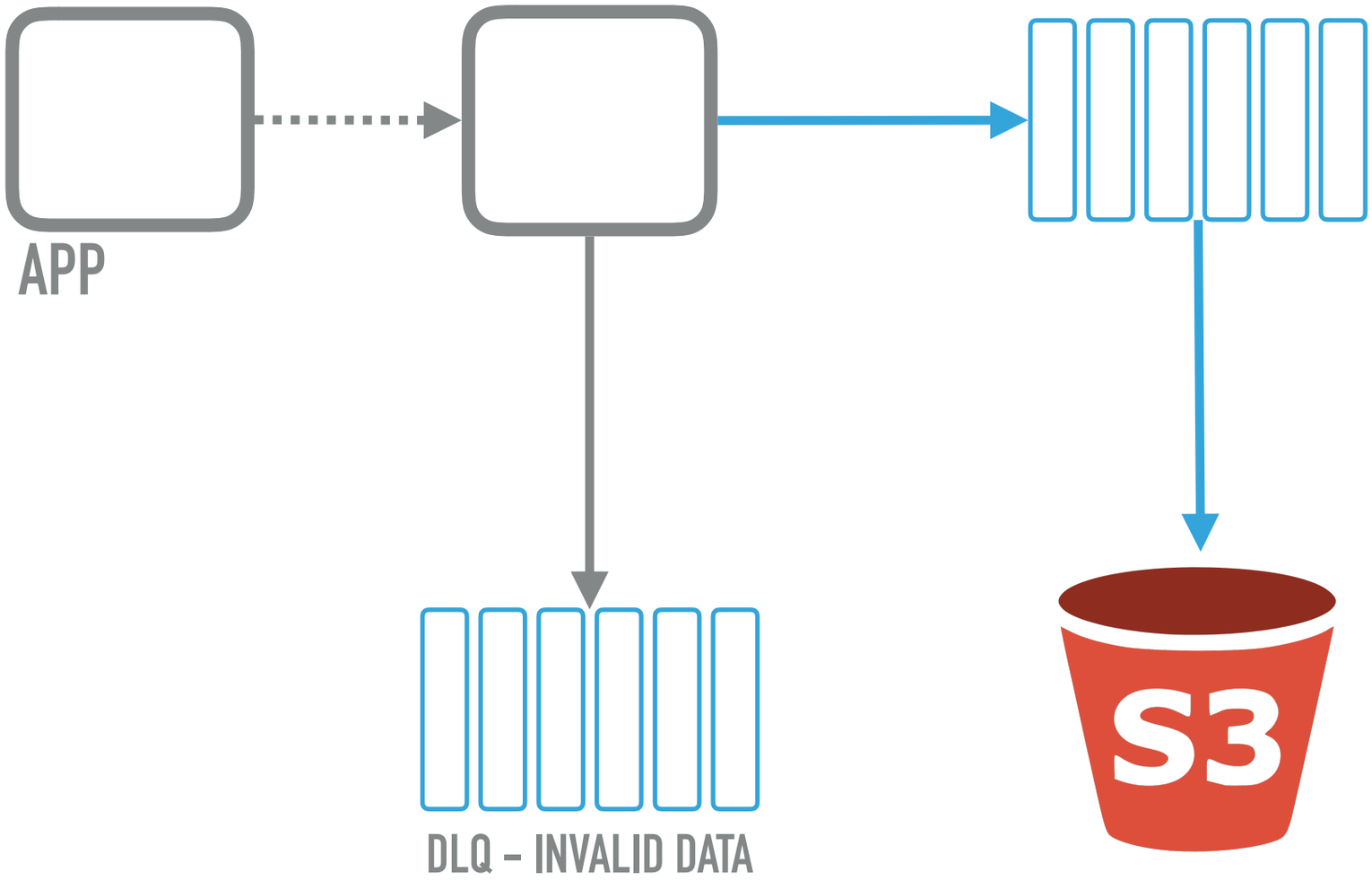
- Fire & Forget
  - Lower latency
  - Lower durability = Higher data loss
- Guaranteed delivery
  - Higher durability = Lower data loss
  - Higher latency

DURABILITY

- Delivery semantics
  - At least once
  - Exactly once
  - At most once

FAULT TOLERANCE

- Human error / Machine error
- Simple collectors
- Immutable data
- Disaster recovery / Replay of historical data
- Dead Letter Queue (DLQ) for invalid data


APP
DLQ - INVALID DATA
S3

ARCHITECTURE

& Technologies

3/6

ARCHITECTURE & TECHNOLOGIES

AS SIMPLE AS POSSIBLE,
BUT NOT SIMPLER

Albert Einstein

TECHNOLOGIES

- Understand the technologies
- Understand the trade-offs
- Build vs. Buy

TECHNOLOGIES

- Kinesis vs. Kafka
- gRPC vs. HTTP
- Apache Spark: AWS EMR vs. DataBricks
- …

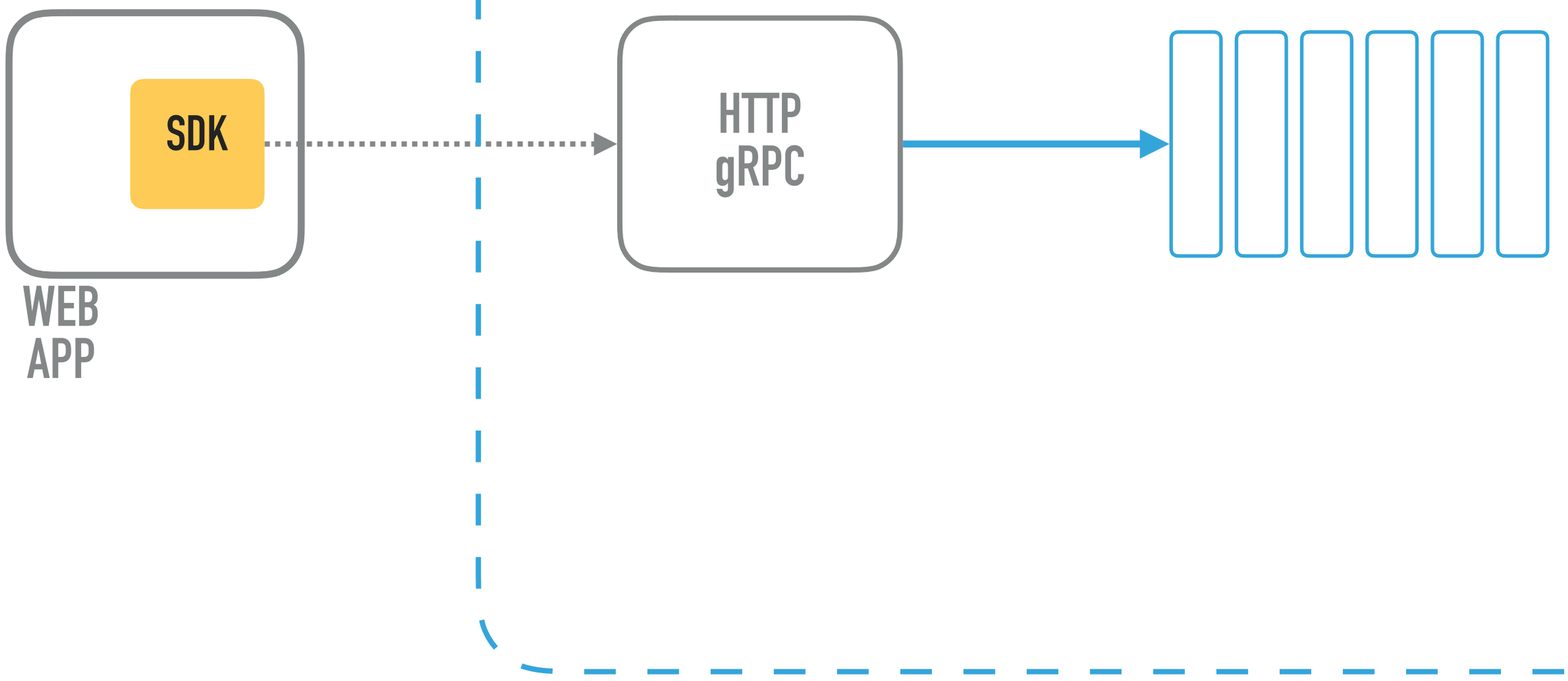
ARCHITECTURE & TECHNOLOGIES

- External facing techs
  - Consistent interface
  - Familiar techs
- Internal techs
  - Ability to experiment

SDK

Software Development Kit

4/6


SDK
WEB
APP
HTTP
gRPC

SDK

- Provide in early stages
- Easier adoption of service
- Take control
- Build resiliency
  - Retry mechanism
  - Circuit breaker

SECURITY

Everyone's Responsibility

5/6

SECURITY

- Do not collect the data you do not need

SECURITY

- Encryption: Rest vs. In-transit
- Anonymizing vs. Pseudonymizing
- Life cycle policies
- Enrichment

SECURITY

- Multiple copies of the data?
- Right to be forgotten (GDPR)

AUTOMATION

6/6

AUTOMATE

- Builds and Deployments (CI / CD)
- Automate infrastructure provisioning

INFRASTRUCTURE AS CODE

- Automatic provisioning
- Parameterized infrastructure code
- Reproducible environments
- Short-lived environments (tests)

TESTING

- Unit & integration tests
  - LocalStack (AWS services in container)
- End-to-End tests
  - Actual infrastructure (smaller capacity)

TRADE-OFFS

TRADE-OFFS

- Maintainability
- Operations
- Security
- Simplicity
- Adoption
- Performance
- …

INFORMED

DECISIONS

QUESTIONS?

RESOURCES

- Google | Site Reliability Engineering
- Designing Data Intensive Applications