
AWS
re:Invent

OPN201

Introduction to GraphQL

Robert Zhu

Principal Developer Advocate

AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- What is GraphQL?
- History of GraphQL
- Comparing GraphQL to REST
- The three superpowers of GraphQL
- Authentication and authorization
- When to use GraphQL
- How to take GraphQL into production



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is GraphQL?

- Request (GraphQL doc) → Response (JSON)
- Language specification
- Schema (domain-specific)
- Server implementation

GraphQL is not

- A client-side state management solution
- A solution for binary streams
- Facebook Graph API
- Limited to specific database(s)
- Limited to JavaScript / NodeJS on the back end
- Limited to use with react / relay / web
- Limited to HTTP

GraphQL operations

- Query (read)
- Mutation (write)
- Subscription

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

Query structure

```
query GetBooksForAuthor($id: ID!) {
  author(id: $id) {
    givenName,
    nationality,
    books {
      title
      isbn
    }
  }
}
```

History of GraphQL

- 2012 – built to support Facebook mobile apps
- 2015 – GraphQL open sourced
- 2018 – GraphQL Foundation

GraphQL vs. REST



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

API technologies

- REST
- SOAP
- gRPC
- Apache Thrift
- OpenAPI / Swagger
- JSON-LD
- HAL
- OData
- And more

REST

GraphQL

	REST	GraphQL
Shared definition	No	Yes

	REST	**GraphQL**
Shared definition	No	Yes
Conceptual model	**Resources**	**Graphs**

	REST	**GraphQL**
Shared definition	No	Yes
Conceptual model	Resources	Graphs
Organization	**Federated**	**Centralized**

	REST	**GraphQL**
Shared definition	No	Yes
Conceptual model	Resources	Graphs
Organization	Federated	Centralized
Related ops	**Yes**	**No**

	REST	**GraphQL**
Shared definition	No	Yes
Conceptual model	Resources	Graphs
Organization	Federated	Centralized
Related ops	Yes	No
Introspection	**No**	**Yes**

	REST	**GraphQL**
Shared definition	No	Yes
Conceptual model	Resources	Graphs
Organization	Federated	Centralized
Related ops	Yes	No
Introspection	No	Yes
Data typing	**Weak**	**Strong**

	REST	**GraphQL**
Shared definition	No	Yes
Conceptual model	Resources	Graphs
Organization	Federated	Centralized
Related ops	Yes	No
Introspection	No	Yes
Data typing	Weak	Strong
Real-time	**No**	**Yes**

Challenges for modern APIs

- Efficiency
- Predictability
- Versioning
- Security
- Documentation
- Tooling
- Caching
- Database integration
- Real-time
- And many more

GraphQL superpowers



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Efficiency

- Overfetching – response contained too much data
- Underfetching – response did not contain enough data; make another network request
- Networks are slow
  - Read 1 MB from SSD: 400 µs
  - Packet US <-> EU: 150 ms

Efficiency

- Avoid overfetching and underfetching
- Clients to declare all the data they need in a single network request
- Reduces the need for client-side joins, error handling, and retry logic

Type safety

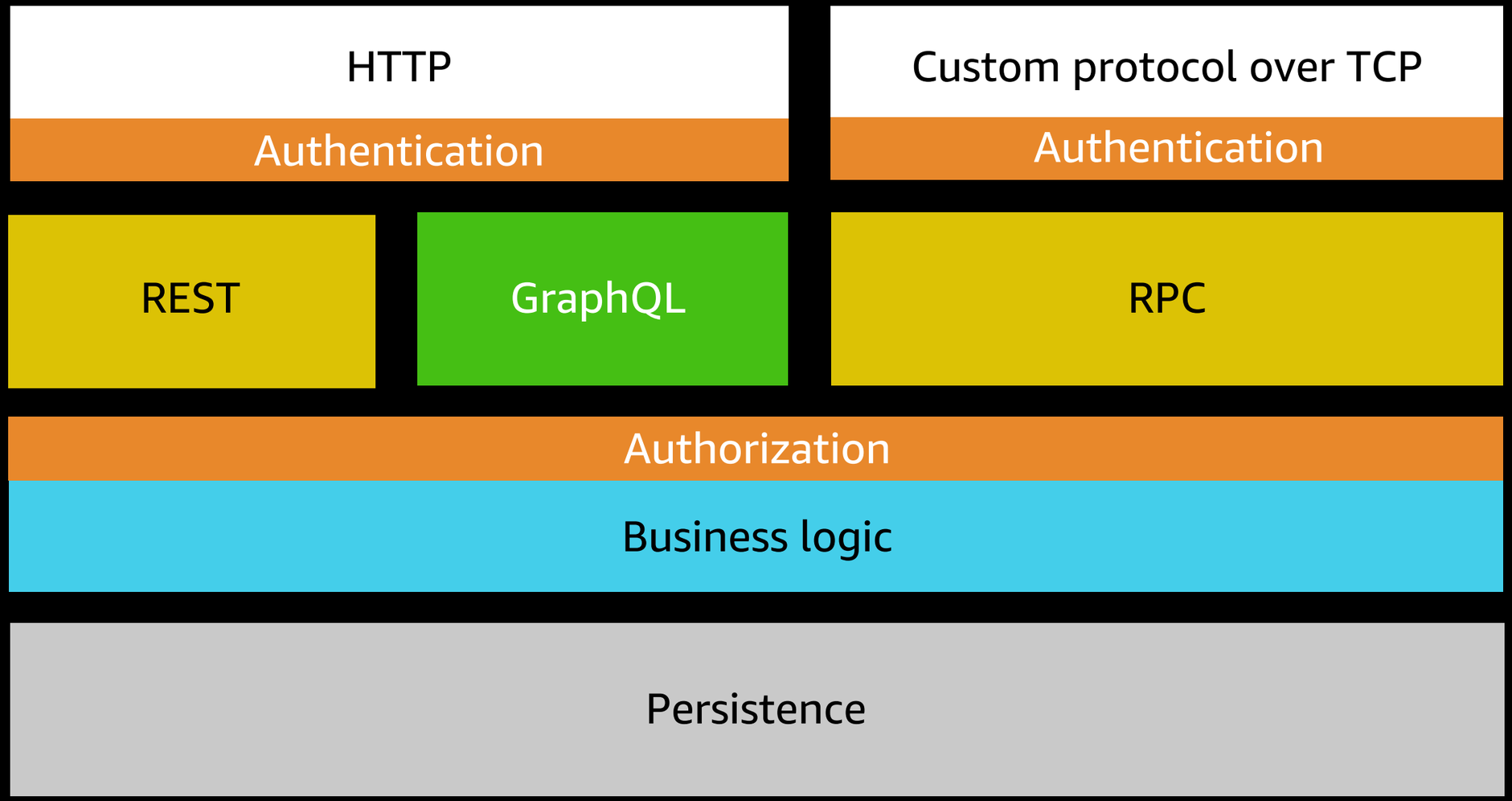
- Predictability
- Native clients
- Code gen

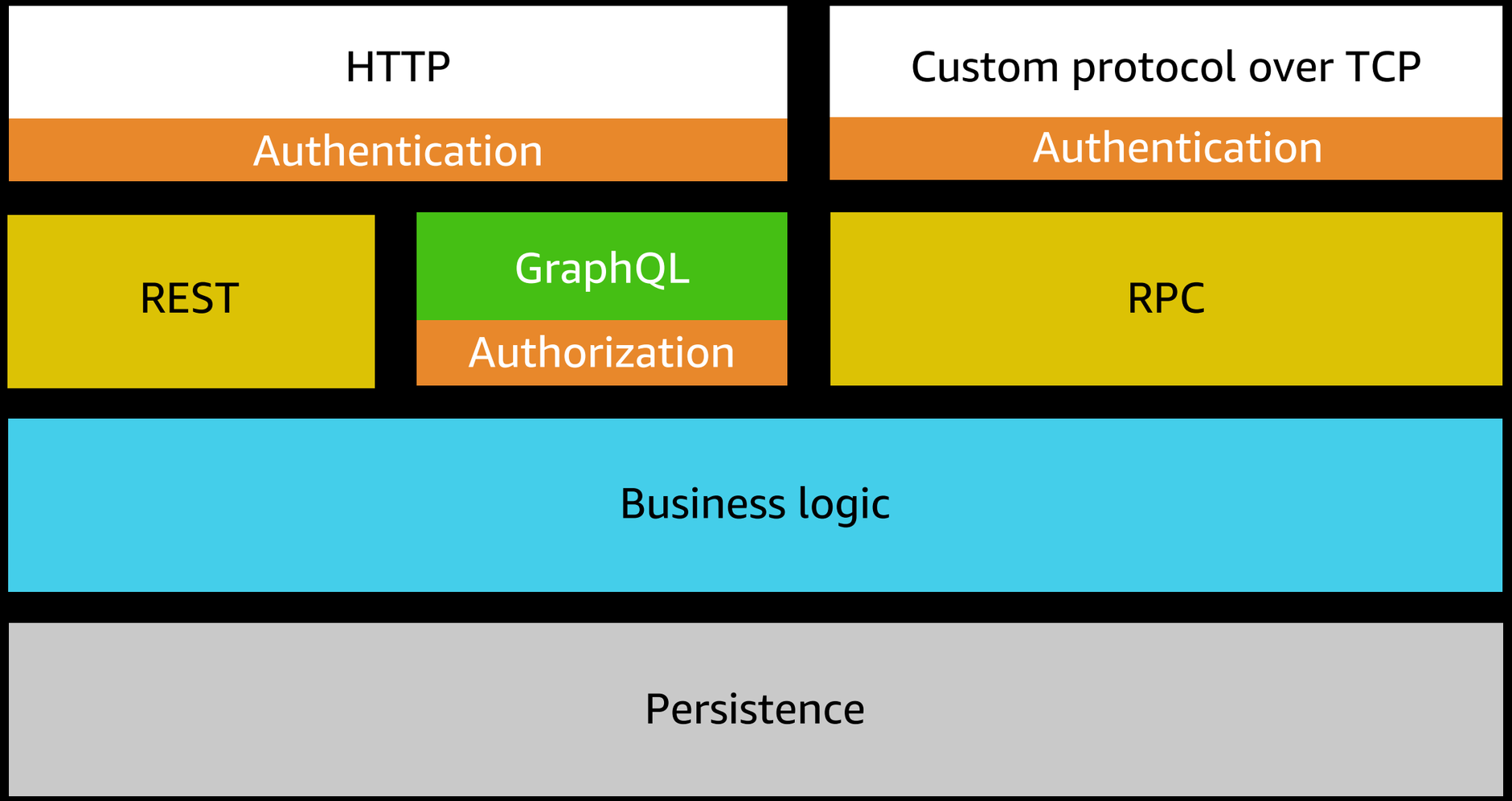
Domain modeling

- Domain-Driven Design
  - Eric Evans, 2004
- Shared language

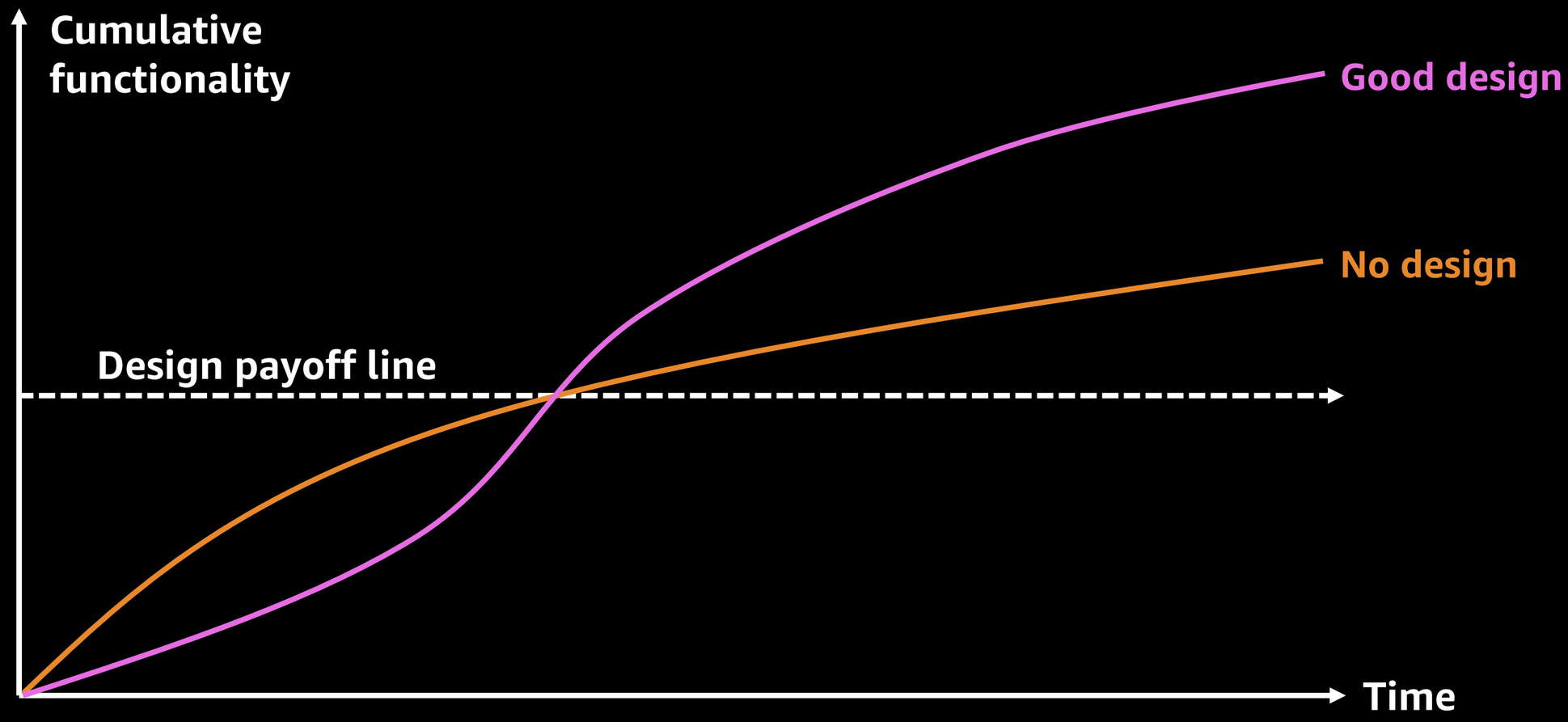
Authentication and authorization

- Authentication
  - Who are you?
- Authorization
  - Do you have permission?


HTTP
Custom protocol over TCP
Authentication
Authentication
REST
GraphQL
RPC
Authorization
Business logic
Persistence


HTTP
Authentication
Custom protocol over TCP
Authentication
REST
GraphQL
Authorization
RPC
Business logic
Persistence

When to use GraphQL



Getting GraphQL into production

DO NOT

- Build a complete schema upfront
- Try to replace your REST API in one push
- Place business logic in the GraphQL layer

DO

- Educate your team
- Choose a solution that fits with your stack
- Build a simple query first (read-only)
- Gradually layer in features like auth / mutations
- Gradually expand schema

Advanced use cases

- API Amazon Gateway
- 1st party vs. 3rd party
- Service-to-service
- Domain modeling – schema-first
- Persisted queries
- Subscriptions
- Live queries

Ecosystem

- AWS AppSync
  https://aws.amazon.com/appsync
- Apollo
  https://www.apollographql.com
- Prisma
  https://www.prisma.io
- Hasura
  https://hasura.io

Resources

- Start here: http://graphql.org
- Community: http://graphql.org/community
- Demo: https://github.com/robzhu/tsgraphql-starwars
- AppSync: https://aws.amazon.com/appsync

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.