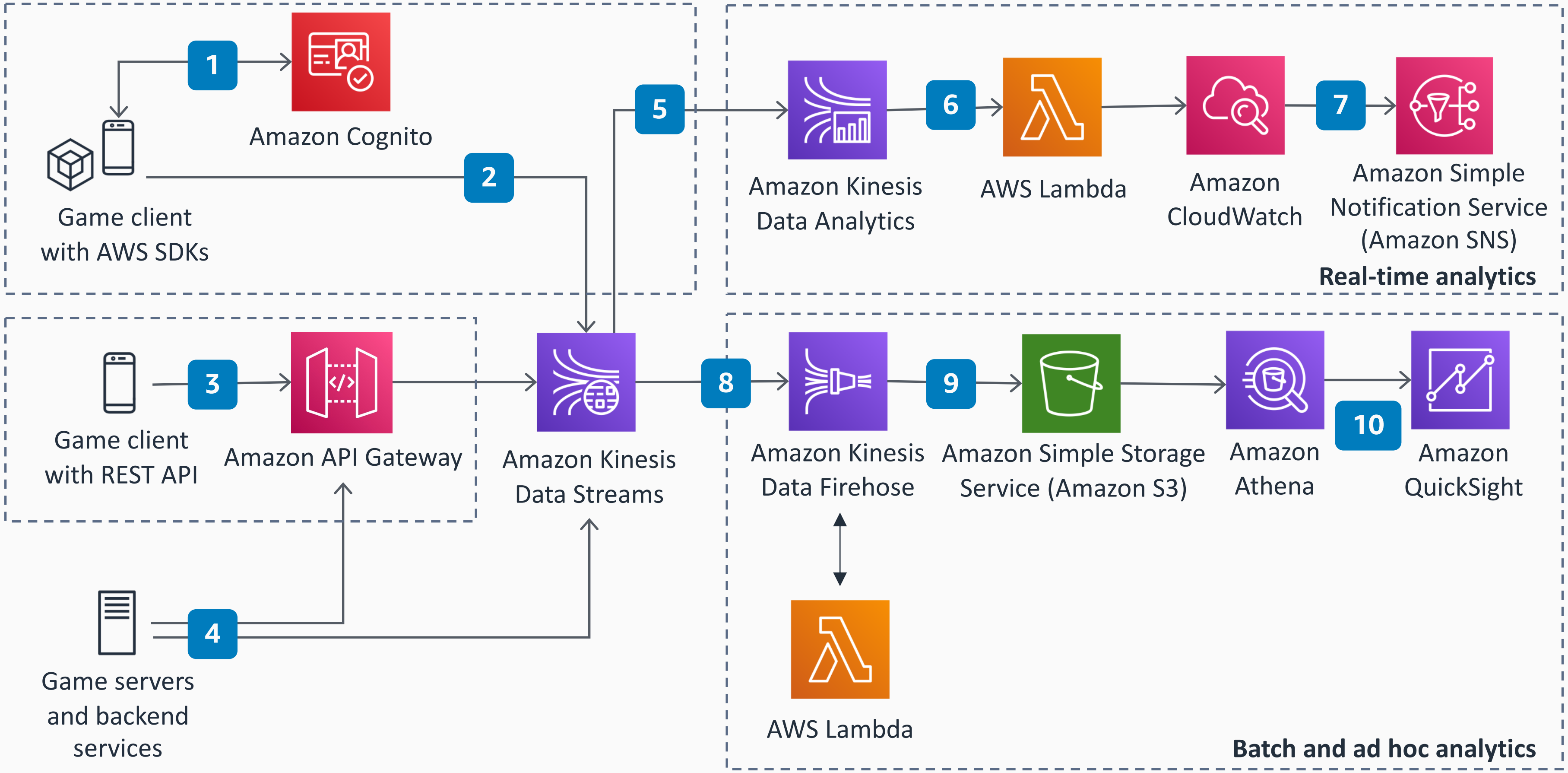
Serverless real-time analytics for games

Build serverless, real-time analytics pipelines for cross-platform games

This serverless architecture example collects events from games, analyzes them in real-time, and stores them for analysis. It can be leveraged to provide real-time flash offers, monitor user acquisition campaigns, detect abusive players, find deficiencies and anomalies during A/B testing, and build online dashboards for business metrics such as retention or operational metrics like crashes.



Collecting client events (option with AWS SDKs):

1. A game, for example, a mobile or PC game, recieves temporary AWS credentials from the **Amazon Cognito** identity pool to access **Amazon Kinesis Data Streams**.

2. The game uses an **AWS SDK** to submit events in JSON directly to **Kinesis Data Streams**. The SDK used depends on the game engine. If the game engine is Unity, then you can use the **AWS SDK for .NET**. If the game engine is Unreal Engine, then you can use the **AWS SDK for C++**.

Collecting client events (option without AWS SDK):

3. The game submits events via the REST API to the **Amazon API Gateway**, which has a native integration with **Kinesis**. This option comes at an added cost of having an **API Gateway**, but creates an additional layer of separation from the player and the **Kinesis Data Stream** via a REST API and does not require **Amazon Cognito**.

Submitting server events:

4. Multiplayer game servers, backend servers, and other services can submit events directly to **Kinesis** using the **AWS SDK**, or through the **API Gateway**. When possible, submit events from an authoritative server.

Processing and analyzing real-time events:

5. **Kinesis Data Analytics** consumes data from the **Kinesis Data Stream** instance and allows real-time SQL queries to run on the stream to analyze, filter, and process data.

6. The data is processed by a **Lambda** function, which sends custom metrics to **Amazon CloudWatch**.

7. The custom **CloudWatch** metrics are visualized in a real-time dashboard. You can create an operational dashboard that shows the health of your infrastructure, and a game events dashboard that shows real-time game KPIs such as concurrent users. You can create alerts and notifications through **CloudWatch** and **SNS**.

Storing events for batch analysis:

8. **Amazon Kinesis Data Firehose** consumes the data stream and pre-processes the data for storage using a built-in **Lambda** integration. For example, you can pre-process the data at this step by transforming it to Parquet, which is a compressed format optimized for performance and lower storage costs when doing analytics.

9. Processed data is batched and stored in **Amazon S3** to create a game analytics data lake. Leverage **S3 Intelligent-Tiering**, lifecycle policies, and different storage tiers for cost savings for historical data.

Visualizing batch data:

10. Query data on an ad hoc basis using **Amazon Athena**. Visualize the data to get business insights using **Amazon QuickSight**.

Reviewed for technical accuracy August 9th, 2021

AWS Reference Architecture