Laporan Resmi AWS

Solusi Data Pengaliran di AWS dengan Amazon Kinesis

Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Solusi Data Pengaliran di AWS dengan Amazon Kinesis: Laporan Resmi AWS

Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan produk Amazon tidak dapat digunakan sehubungan dengan produk atau layanan yang bukan milik Amazon, dengan segala cara yang mungkin menyebabkan kebingungan di antara pelanggan, atau dengan segala cara yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon adalah properti dari pemiliknya masing-masing, yang mungkin atau mungkin tidak berafiliasi dengan, berhubungan dengan, atau disponsori oleh Amazon.

Table of Contents

Abstrak ... 1
Abstrak ... 1
Pengantar ... 2
Skenario aplikasi waktu nyata dan hampir waktu nyata ... 2
Perbedaan antara pemrosesan batch dan aliran ... 3
Tantangan pemrosesan aliran ... 3
Solusi data pengaliran: contoh ... 4
Skenario 1: Penawaran internet berdasarkan lokasi ... 4
Amazon Kinesis Data Streams ... 4
Memproses aliran data dengan AWS Lambda ... 6
Ringkasan ... 7
Skenario 2: Data hampir waktu nyata untuk tim keamanan ... 7
Amazon Kinesis Data Firehose ... 8
Ringkasan ... 13
Skenario 3: Mempersiapkan data clickstream untuk proses wawasan data ... 14
Pengaliran AWS Glue dan AWS Glue ... 15
Amazon DynamoDB ... 16
Titik akhir layanan Amazon SageMaker dan Amazon SageMaker ... 17
Menginferensi wawasan data secara waktu nyata ... 17
Ringkasan ... 18
Skenario 4: Deteksi dan notifikasi anomali waktu nyata sensor perangkat ... 18
Amazon Kinesis Data Analytics ... 20
Aplikasi Amazon Kinesis Data Analytics for Apache Flink ... 20
Skenario 5: Pemantauan data telemetri waktu nyata dengan Apache Kafka ... 23
Amazon Managed Streaming for Apache Kafka (Amazon MSK) ... 24
Bermigrasi ke Amazon MSK ... 26
Kesimpulan dan kontributor ... 30
Kesimpulan ... 30
Kontributor ... 30
Revisi dokumen ... 31

Solusi Data Pengaliran di AWS

Tanggal publikasi: 1 September 2021 (Revisi dokumen)

Abstrak

Rekayasawan data, analis data, dan developer big data ingin mengembangkan analitik mereka dari batch ke waktu nyata agar perusahaan mereka dapat mempelajari tentang apa yang dilakukan pelanggan, aplikasi, dan produk mereka saat ini dan bereaksi dengan cepat. Laporan resmi ini membahas evolusi analitik dari batch ke waktu nyata. Laporan resmi ini menjelaskan bagaimana layanan seperti Amazon Kinesis Data Streams, Amazon Kinesis Data Firehose, Amazon EMR, Amazon Kinesis Data Analytics, Amazon Managed Streaming for Apache Kafka (Amazon MSK), dan layanan lainnya dapat digunakan untuk mengimplementasikan aplikasi waktu nyata, dan menyediakan pola desain umum menggunakan sejumlah layanan tersebut.

Pengantar

Bisnis saat ini menerima data dalam skala besar dan kecepatan tinggi karena pertumbuhan sumber data yang melonjak yang terus menghasilkan aliran data. Apakah itu data log dari server aplikasi, data clickstream dari situs web dan aplikasi seluler, atau data telemetri dari perangkat Internet untuk Segala (IoT), semuanya berisi informasi yang dapat membantu Anda mempelajari tentang apa yang dilakukan pelanggan, aplikasi, dan produk Anda saat ini.

Memiliki kemampuan memproses dan menganalisis data ini secara waktu nyata sangat penting untuk melakukan berbagai hal seperti terus memantau aplikasi Anda guna memastikan waktu aktif layanan yang tinggi serta mempersonalisasi penawaran promosi dan rekomendasi produk. Pemrosesan waktu nyata dan hampir waktu nyata juga dapat membuat kasus penggunaan umum lainnya, seperti analitik situs web dan machine learning, lebih akurat dan dapat ditindaklanjuti dengan membuat data tersedia untuk aplikasi ini dalam hitungan detik atau menit, bukan jam atau hari.

Skenario aplikasi waktu nyata dan hampir waktu nyata

Anda dapat menggunakan layanan data pengaliran untuk aplikasi waktu nyata dan hampir waktu nyata seperti pemantauan aplikasi, deteksi penipuan, dan papan peringkat langsung. Kasus penggunaan waktu nyata memerlukan latensi ujung ke ujung milidetik – dari penyerapan, pemrosesan, hingga mengirimkan hasil ke penyimpanan data target dan sistem lainnya. Misalnya, Netflix menggunakan Amazon Kinesis Data Streams untuk memantau komunikasi di antara semua aplikasinya sehingga dapat mendeteksi dan memperbaiki permasalahan dengan cepat, sehingga memastikan waktu aktif layanan dan ketersediaan yang tinggi bagi pelanggannya. Meskipun kasus penggunaan yang paling umum berlaku adalah pemantauan performa aplikasi, ada peningkatan jumlah aplikasi waktu nyata dalam teknologi iklan, game, dan IoT yang termasuk dalam kategori ini.

Kasus penggunaan hampir waktu nyata yang umum mencakup analitik pada penyimpanan data untuk ilmu data dan machine learning (ML). Anda dapat menggunakan solusi data pengaliran untuk terus memuat data waktu nyata ke danau data Anda. Anda kemudian dapat memperbarui model ML secara lebih sering seiring data baru tersedia, sehingga memastikan keakuratan dan keandalan output. Misalnya, Zillow menggunakan Kinesis Data Streams untuk mengumpulkan catatan data publik dan listingan Multiple Listing Service (MLS), lalu memberikan perkiraan nilai rumah yang paling terbaru secara hampir waktu nyata kepada pembeli dan penjual rumah. ZipRecruiter menggunakan Amazon MSK untuk alur pencatatan log peristiwa mereka, yang merupakan komponen infrastruktur penting yang mengumpulkan, menyimpan, dan terus memproses lebih dari enam miliar peristiwa per hari dari marketplace lowongan kerja ZipRecruiter.

Perbedaan antara pemrosesan batch dan aliran

Anda memerlukan serangkaian alat yang berbeda untuk mengumpulkan, menyiapkan, dan memproses data pengaliran waktu nyata daripada alat yang telah Anda gunakan secara tradisional untuk analitik batch. Dengan analitik tradisional, Anda mengumpulkan data, memuatnya secara berkala ke dalam basis data, dan menganalisisnya dalam hitungan jam, hari, atau minggu. Menganalisis data waktu nyata membutuhkan pendekatan yang berbeda. Aplikasi pemrosesan aliran terus memproses data secara waktu nyata, bahkan sebelum data disimpan. Data pengaliran dapat masuk dengan kecepatan yang tinggi dan volume data dapat bervariasi naik dan turun kapan saja. Platform pemrosesan data aliran harus mampu menangani kecepatan dan variabilitas data yang masuk dan memprosesnya saat data tiba, sering kali jutaan hingga ratusan juta peristiwa per jam.

Tantangan pemrosesan aliran

Memproses data waktu nyata saat data tiba dapat memungkinkan Anda membuat keputusan dengan lebih cepat daripada yang mungkin dilakukan dengan teknologi analitik data tradisional. Namun, membangun dan mengoperasikan alur data pengaliran kustom Anda sendiri itu rumit dan memerlukan banyak sumber daya:

- Anda harus membangun sebuah sistem yang dapat secara efektif mengumpulkan, mempersiapkan, dan mengirimkan data yang datang secara bersamaan dari ribuan sumber data.
- Anda perlu menyesuaikan sumber daya penyimpanan dan komputasi sehingga data di-batch dan dikirimkan secara efisien untuk throughput maksimum dan latensi rendah.
- Anda harus men-deploy dan mengelola armada server untuk menskalakan sistem agar Anda dapat menangani berbagai kecepatan data yang akan Anda kirimkan ke sistem tersebut.

Upgrade versi adalah proses yang kompleks dan mahal. Setelah Anda membangun platform ini, Anda harus memantau sistem dan memulihkan kegagalan server atau jaringan dengan melanjutkan pemrosesan data dari titik yang sesuai di aliran, tanpa membuat data duplikat. Anda juga memerlukan tim khusus untuk manajemen infrastruktur. Semua ini membutuhkan waktu yang berharga dan uang dan, pada akhirnya, sebagian besar perusahaan tidak pernah mampu mewujudkannya dan harus puas dengan status quo dan mengoperasikan bisnis mereka dengan informasi lama dari beberapa jam atau hari yang lalu.

Solusi data pengaliran: contoh

Skenario 1: Penawaran internet berdasarkan lokasi

Perusahaan InternetProvider menyediakan layanan internet dengan berbagai pilihan bandwidth untuk pengguna di seluruh dunia. Ketika pengguna mendaftar untuk internet, perusahaan InternetProvider memberi pengguna pilihan bandwidth yang berbeda-beda berdasarkan lokasi geografis pengguna. Mengingat persyaratan ini, perusahaan InternetProvider menerapkan Amazon Kinesis Data Streams untuk mengonsumsi detail dan lokasi pengguna. Detail pengguna dan lokasi diperkaya dengan opsi bandwidth yang berbeda-beda sebelum dipublikasikan balik ke aplikasi. AWS Lambda memungkinkan pengayaan waktu nyata ini.

Memproses aliran data dengan AWS Lambda

Amazon Kinesis Data Streams

Amazon Kinesis Data Streams memungkinkan Anda membangun aplikasi kustom dan waktu nyata menggunakan kerangka kerja pemrosesan aliran populer dan memuat data pengaliran ke banyak penyimpanan data yang berbeda. Sebuah aliran Kinesis dapat dikonfigurasi untuk terus menerima peristiwa dari ratusan ribu produsen data yang dikirim dari sumber seperti clickstream situs web, sensor IoT, umpan media sosial, dan log aplikasi. Dalam hitungan milidetik, data tersedia untuk dibaca dan diproses oleh aplikasi Anda.

Saat menerapkan solusi dengan Kinesis Data Streams, Anda membuat aplikasi pemrosesan data kustom yang dikenal sebagai aplikasi Kinesis Data Streams. Sebuah aplikasi Kinesis Data Streams standar akan membaca data dari aliran Kinesis sebagai catatan data.

Data yang dimasukkan ke dalam Kinesis Data Streams dipastikan akan sangat tersedia dan elastis, dan tersedia dalam milidetik. Anda dapat terus menambahkan berbagai jenis data seperti clickstream,

log aplikasi, dan media sosial ke aliran Kinesis dari ratusan ribu sumber. Dalam hitungan detik, data akan tersedia untuk Aplikasi Kinesis Anda yang dapat dibaca dan diproses dari aliran tersebut.

Amazon Kinesis Data Streams adalah layanan data pengaliran terkelola penuh. Layanan ini mengelola infrastruktur, penyimpanan, jaringan, dan konfigurasi yang diperlukan untuk melakukan pengaliran data pada tingkat throughput data.

Mengirim data ke Amazon Kinesis Data Streams

Ada beberapa cara untuk mengirim data ke Kinesis Data Streams, yang memberikan fleksibilitas dalam desain solusi Anda.

- Anda dapat menulis kode dengan menggunakan salah satu SDK AWS yang didukung oleh sejumlah bahasa populer.
- Anda dapat menggunakan Amazon Kinesis Agent, alat untuk mengirim data ke Kinesis Data Streams.

Amazon Kinesis Producer Library (KPL) menyederhanakan pengembangan aplikasi produsen dengan memungkinkan developer mencapai throughput penulisan yang tinggi ke satu atau beberapa aliran data Kinesis.

KPL adalah pustaka yang mudah digunakan dan sangat dapat dikonfigurasi yang Anda instal di host Anda. Layanan ini bertindak sebagai perantara antara kode aplikasi produsen Anda dan tindakan API Kinesis Streams. Untuk informasi lebih lanjut tentang KPL dan kemampuannya untuk menghasilkan peristiwa secara sinkron dan asinkron dengan contoh kode, lihat Menulis ke Kinesis Data Streams Anda Menggunakan KPL

Ada dua operasi yang berbeda dalam API Kinesis Data Streams yang menambahkan data ke sebuah aliran: `PutRecords` dan `PutRecord`. Operasi `PutRecords` mengirimkan sejumlah catatan ke aliran Anda per permintaan HTTP sementara `PutRecord` mengirimkan satu catatan per permintaan HTTP. Untuk mencapai throughput yang lebih tinggi untuk sebagian besar aplikasi, gunakan `PutRecords`.

Untuk informasi selengkapnya tentang API ini, lihat Menambahkan Data ke Stream. Detail untuk setiap operasi API dapat ditemukan di Referensi API Amazon Kinesis Data Streams.

Memproses data di Amazon Kinesis Data Streams

Untuk membaca dan memproses data dari aliran Kinesis, Anda perlu membuat aplikasi konsumen. Ada beragam cara untuk membuat konsumen untuk Kinesis Data Streams. Beberapa pendekatan ini termasuk menggunakan Amazon Kinesis Data Analytics untuk menganalisis data pengaliran dengan

KCL, menggunakan AWS Lambda, tugas ETL pengaliran AWS Glue, dan menggunakan API Kinesis Data Streams secara langsung.

Aplikasi konsumen untuk Kinesis Data Streams dapat dikembangkan menggunakan KCL, yang membantu Anda mengonsumsi dan memproses data dari Kinesis Data Streams. KCL menangani banyak tugas kompleks yang terkait dengan komputasi terdistribusi seperti penyeimbangan beban di sejumlah instans, merespons kegagalan instans, melakukan checkpointing catatan yang diproses, dan bereaksi terhadap resharding. KCL memungkinkan Anda fokus pada penulisan logika pemrosesan catatan. Untuk informasi lebih lanjut tentang cara membangun aplikasi KCL Anda sendiri, lihat Menggunakan Kinesis Client Library.

Anda dapat berlangganan fungsi Lambda untuk secara otomatis membaca batch catatan dari aliran Kinesis Anda dan memprosesnya jika catatan terdeteksi di aliran. AWS Lambda secara berkala memilih aliran (sekali per detik) untuk catatan baru dan ketika mendeteksi catatan baru, layanan ini memanggil fungsi Lambda yang meneruskan catatan baru sebagai parameter. Fungsi Lambda hanya dijalankan ketika catatan baru terdeteksi. Anda dapat memetakan fungsi Lambda ke konsumen throughput bersama (iterator standar)

Anda dapat membangun konsumen yang menggunakan fitur yang disebut enhanced fan-out ketika Anda memerlukan throughput khusus yang tidak boleh dibagi dengan konsumen lain yang menerima data dari aliran. Fitur ini memungkinkan konsumen menerima catatan dari aliran dengan throughput hingga dua MB data per detik per pecahan.

Untuk sebagian besar kasus, menggunakan Kinesis Data Analytics, KCL, AWS Glue, atau AWS Lambda harus digunakan untuk memproses data dari aliran. Namun, jika Anda mau, Anda dapat membuat aplikasi konsumen dari awal menggunakan API Kinesis Data Streams. API Kinesis Data Streams menyediakan metode `GetShardIterator` dan `GetRecords` untuk mengambil data dari aliran.

Dalam model pull ini, kode Anda mengekstrak data langsung dari pecahan aliran. Untuk informasi selengkapnya tentang menulis aplikasi konsumen Anda sendiri menggunakan API, lihat Mengembangkan Konsumen Kustom dengan Throughput Bersama Menggunakan AWS SDK for Java. Detail tentang API dapat ditemukan di Referensi API Amazon Kinesis Data Streams.

Memproses aliran data dengan AWS Lambda

AWS Lambda memungkinkan Anda menjalankan kode tanpa menyediakan atau mengelola server. Dengan Lambda, Anda dapat menjalankan kode hampir untuk semua jenis aplikasi atau layanan backend tanpa administrasi. Cukup unggah kode Anda dan Lambda akan menangani segala yang

diperlukan untuk menjalankan dan menskalakan kode Anda dengan ketersediaan yang tinggi. Anda dapat mengatur kode untuk secara otomatis dipicu dari layanan AWS lainnya atau memanggilnya secara langsung dari aplikasi web atau seluler.

AWS Lambda terintegrasi secara native dengan Amazon Kinesis Data Streams. Kompleksitas polling, checkpointing, dan penanganan kesalahan akan dikurangi saat Anda menggunakan integrasi native ini. Hal ini memungkinkan kode fungsi Lambda berfokus pada pemrosesan logika bisnis.

Anda dapat memetakan fungsi Lambda ke throughput bersama (iterator standar), atau ke konsumen yang berdedikasi dengan enhanced fan-out. Dengan iterator standar, Lambda melakukan polling terhadap setiap pecahan dalam aliran Kinesis Anda untuk catatan yang menggunakan protokol HTTP. Untuk meminimalkan latensi dan memaksimalkan throughput baca, Anda dapat membuat konsumen aliran data dengan enhanced fan-out. Konsumen aliran dalam arsitektur ini mendapatkan koneksi khusus untuk setiap pecahan tanpa bersaing dengan aplikasi lain yang membaca dari aliran yang sama. Amazon Kinesis Data Streams mendorong catatan ke Lambda melalui HTTP/2.

Secara default, AWS Lambda memanggil fungsi Anda segera setelah catatan tersedia dalam aliran. Untuk melakukan buffer terhadap catatan untuk skenario batch, Anda dapat menerapkan periode batch hingga lima menit di sumber peristiwa. Jika fungsi Anda mengembalikan kesalahan, Lambda akan mencoba ulang batch tersebut sampai pemrosesannya berhasil atau datanya kedaluwarsa.

Ringkasan

Perusahaan InternetProvider memanfaatkan Amazon Kinesis Data Streams untuk melakukan pengaliran detail dan lokasi pengguna. Aliran catatan dikonsumsi oleh AWS Lambda untuk memperkaya data dengan opsi bandwidth yang disimpan di pustaka fungsi. Setelah pengayaan, AWS Lambda memublikasikan opsi bandwidth kembali ke aplikasi. Amazon Kinesis Data Streams dan AWS Lambda menangani penyediaan dan pengelolaan server, sehingga memungkinkan perusahaan InternetProvider lebih fokus pada pengembangan aplikasi bisnis.

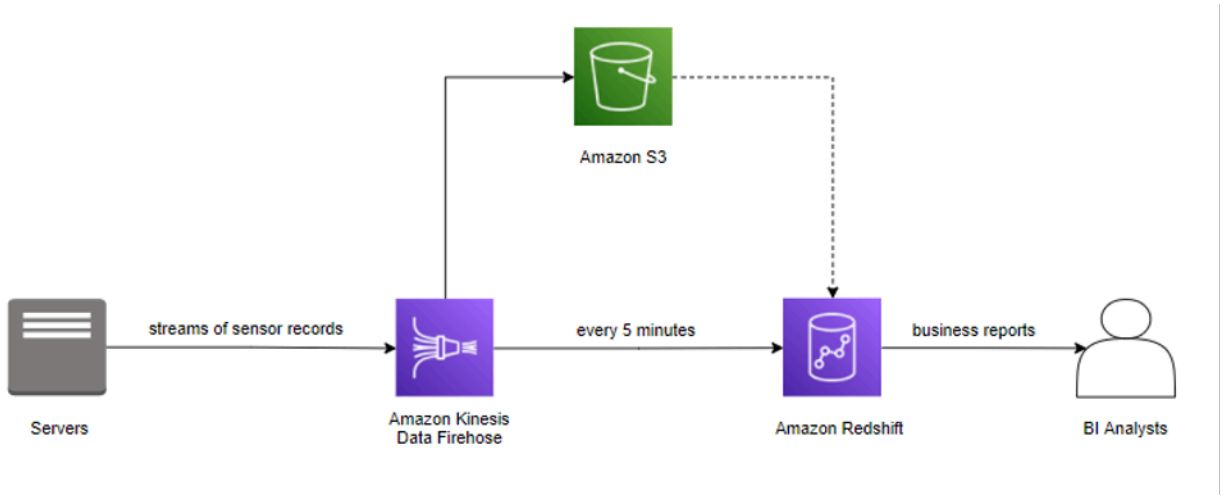
Skenario 2: Data hampir waktu nyata untuk tim keamanan

Perusahaan ABC2Badge menyediakan sensor dan lencana untuk acara korporasi atau acara berskala besar seperti AWS re:Invent. Pengguna akan mendaftar ke acara tersebut dan menerima lencana unik yang akan dideteksi oleh sensor di seluruh kampus. Ketika pengguna melewati sebuah sensor, informasi anonim mereka akan dicatat ke dalam basis data relasional.

Dalam acara mendatang, karena tingginya volume peserta, ABC2Badge telah diminta oleh tim keamanan acara ini untuk mengumpulkan data untuk area kampus yang paling terkonsentrasi setiap

15 menit. Data ini akan memberi tim keamanan cukup waktu untuk bereaksi dan mengerahkan personel keamanan secara proporsional ke area yang terkonsentrasi. Mengingat kebutuhan baru dari tim keamanan ini dan kurangnya pengalaman membangun solusi pengaliran, untuk memproses data secara hampir waktu nyata, ABC2Badge mencari solusi yang sederhana namun terukur dan dapat diandalkan.

Solusi gudang data mereka saat ini adalah Amazon Redshift. Saat meninjau fitur layanan Amazon Kinesis, mereka mendapati bahwa Amazon Kinesis Data Firehose dapat menerima aliran catatan data, melakukan batch terhadap catatan berdasarkan ukuran buffer dan/atau interval waktu, dan memasukkannya ke dalam Amazon Redshift. Mereka membuat aliran pengiriman Kinesis Data Firehose dan mengonfigurasinya sehingga layanan ini akan menyalin data ke tabel Amazon Redshift mereka setiap lima menit. Sebagai bagian dari solusi baru ini, mereka menggunakan Amazon Kinesis Agent di server mereka. Setiap lima menit, Kinesis Data Firehose memuat data ke Amazon Redshift, sehingga mendukung tim Kecerdasan Bisnis (BI) untuk melakukan analisisnya dan mengirim data ke tim keamanan setiap 15 menit.



Solusi baru menggunakan Amazon Kinesis Data Firehose

Amazon Kinesis Data Firehose

Amazon Kinesis Data Firehose adalah cara paling mudah untuk memuat data pengaliran ke AWS. Layanan ini dapat menangkap, mentransformasikan, dan memuat data pengaliran ke Amazon Kinesis Data Analytics, Amazon Simple Storage Service (Amazon S3), Amazon Redshift, Amazon OpenSearch Service (OpenSearch Service), dan Splunk. Selain itu, Kinesis Data Firehose dapat memuat data pengaliran ke titik akhir HTTP atau titik akhir HTTP kustom yang dimiliki oleh penyedia layanan pihak ketiga yang didukung.

Kinesis Data Firehose memungkinkan analitik hampir waktu nyata dengan alat Kecerdasan Bisnis (BI) yang ada dan dasbor yang Anda gunakan saat ini. Ini adalah layanan nirserver terkelola penuh yang secara otomatis diskalakan agar sesuai dengan throughput data Anda dan tidak memerlukan administrasi terus-menerus. Kinesis Data Firehose dapat melakukan batch, kompresi, dan enkripsi data sebelum memuatnya, sehingga meminimalkan jumlah penyimpanan yang digunakan di tujuan dan meningkatkan keamanan. Layanan ini juga dapat mentransformasikan data sumber menggunakan AWS Lambda dan mengirimkan data yang ditransformasi ke tujuan. Anda mengonfigurasi produsen data Anda untuk mengirim data ke Kinesis Data Firehose, yang akan mengirimkan data secara otomatis ke tujuan yang telah Anda tetapkan.

Mengirim data ke aliran pengiriman Firehose

Untuk mengirim data ke aliran pengiriman Anda, ada beberapa opsi. AWS menawarkan SDK untuk banyak bahasa pemrograman populer, yang masing-masing menyediakan API untuk Amazon Kinesis Data Firehose. AWS memiliki utilitas untuk membantu mengirim data ke aliran pengiriman Anda. Kinesis Data Firehose telah terintegrasi dengan layanan AWS lain untuk mengirim data langsung dari layanan tersebut ke dalam aliran pengiriman Anda.

Menggunakan agen Amazon Kinesis

Agen Amazon Kinesis adalah aplikasi perangkat lunak mandiri yang terus memantau serangkaian file log agar data baru dikirim ke aliran pengiriman. Agen ini secara otomatis menangani rotasi file, melakukan checkpointing, mencoba kembali jika ada kegagalan, dan mengirimkan metrik Amazon CloudWatch untuk memantau dan mengatasi masalah aliran pengiriman. Konfigurasi tambahan, seperti pra-pemrosesan data, pemantauan sejumlah direktori file, dan penulisan ke sejumlah aliran pengiriman, dapat diterapkan pada agen ini.

Agen ini dapat diinstal pada server berbasis Linux atau Windows seperti server web, server log, dan server basis data. Setelah agen ini terinstal, Anda cukup menentukan file log yang akan dipantau dan aliran pengiriman yang akan dikirimkan. Agen ini akan mengirim data baru ke aliran pengiriman secara tahan lama dan andal.

Menggunakan API dengan SDK AWS dan layanan AWS sebagai sumber

API Kinesis Data Firehose menawarkan dua operasi untuk mengirim data ke aliran pengiriman Anda. `PutRecord` mengirimkan satu catatan data dalam satu panggilan. `PutRecordBatch` dapat mengirim sejumlah catatan data dalam satu panggilan, dan dapat mencapai throughput yang lebih tinggi per produsen. Dalam setiap metode, Anda harus menentukan nama aliran pengiriman dan catatan data, atau array catatan data, ketika menggunakan metode ini. Untuk informasi selengkapnya

dan kode sampel untuk operasi API Kinesis Data Firehose, lihat Menulis ke Aliran Pengiriman Firehose Menggunakan SDK AWS.

Kinesis Data Firehose juga berjalan dengan Kinesis Data Streams, CloudWatch Logs, CloudWatch Events, Amazon Simple Notification Service (Amazon SNS), Amazon API Gateway, dan AWS IoT. Anda dapat mengirim aliran data, log, peristiwa, dan data IoT Anda secara terskala dan andal ke tujuan Kinesis Data Firehose.

Memproses data sebelum pengiriman ke tujuan

Dalam beberapa skenario, Anda sebaiknya mentransformasikan atau meningkatkan data pengaliran Anda sebelum dikirim ke tujuannya. Misalnya, produsen data mungkin mengirim teks yang tidak terstruktur di setiap catatan data, dan Anda perlu mentransformasikannya menjadi JSON sebelum mengirimkannya ke OpenSearch Service. Atau, Anda sebaiknya mengonversi data JSON menjadi format file kolom seperti Apache Parquet atau Apache ORC sebelum menyimpan data di Amazon S3.

Kinesis Data Firehose memiliki kemampuan konversi format data bawaan. Menggunakan kemampuan ini, Anda dapat dengan mudah mengonversi aliran data JSON Anda ke format file Apache Parquet atau Apache ORC.

Aliran transformasi data

Untuk memungkinkantransformasi data pengaliran, Kinesis Data Firehose menggunakan fungsi Lambda yang Anda buat untuk mentransformasikan data Anda. Kinesis Data Firehose melakukan buffer terhadap data yang masuk menjadi ukuran buffer tertentu untuk fungsi, lalu memanggil fungsi Lambda yang ditentukan secara asinkron. Data yang ditransformasi dikirim dari Lambda ke Kinesis Data Firehose, dan Kinesis Data Firehose mengirimkan data ke tujuan.

Konversi format data

Anda juga dapat mengaktifkan konversi format data Kinesis Data Firehose, yang akan mengonversi aliran data JSON Anda ke Apache Parquet atau Apache ORC. Fitur ini hanya dapat mengonversi JSON ke Apache Parquet atau Apache ORC. Jika Anda memiliki data yang ada di CSV, Anda dapat mentransformasikan data tersebut melalui fungsi Lambda ke JSON, lalu menerapkan konversi format data.

Pengiriman data

Sebagai aliran pengiriman hampir waktu nyata, Kinesis Data Firehose melakukan buffer terhadap data yang masuk. Setelah ambang batas buffering aliran pengiriman tercapai, data Anda dikirimkan

ke tujuan yang telah Anda konfigurasi. Ada beberapa perbedaan dalam bagaimana Kinesis Data Firehose memberikan data ke setiap tujuan, yang ditinjau dalam laporan ini di bagian berikut.

Amazon S3

Amazon S3 adalah penyimpanan objek dengan antarmuka layanan web sederhana untuk menyimpan dan mengambil jumlah data dari mana pun di web. Produk ini dirancang untuk menyediakan ketahanan 99,999999999% dan menskalakan melebihi triliunan objek di seluruh dunia.

Pengiriman data ke Amazon S3

Untuk pengiriman data ke Amazon S3, Kinesis Data Firehose menggabungkan sejumlah catatan masuk berdasarkan konfigurasi buffering aliran pengiriman Anda, lalu mengirimkannya ke Amazon S3 sebagai objek S3. Frekuensi pengiriman data ke S3 ditentukan oleh ukuran buffer (1 MB sampai 128 MB) atau interval buffer (60 detik sampai 900 detik) di S3, mana yang tercapai lebih dulu.

Pengiriman data ke bucket S3 Anda mungkin gagal karena berbagai alasan. Misalnya, bucket mungkin sudah tidak ada lagi, atau AWS Identity and Access Management (IAM) role yang diambil Kinesis Data Firehose mungkin tidak memiliki akses ke bucket. Dalam kondisi ini, Kinesis Data Firehose terus mencoba kembali hingga 24 jam sampai pengiriman berhasil. Waktu penyimpanan data maksimum Kinesis Data Firehose adalah 24 jam. Jika pengiriman data gagal selama lebih dari 24 jam, data Anda hilang.

Amazon Redshift

Amazon Redshift adalah gudang data cepat dan terkelola penuh yang memudahkan dan menghemat biaya dalam menganalisis semua data Anda menggunakan SQL standar dan alat-alat BI Anda yang sudah ada. Layanan ini memungkinkan Anda menjalankan kueri analitik kompleks pada data terstruktur berskala petabita menggunakan optimasi kueri canggih, penyimpanan kolom pada disk lokal berperforma tinggi, dan pemrosesan kueri paralel secara masif.

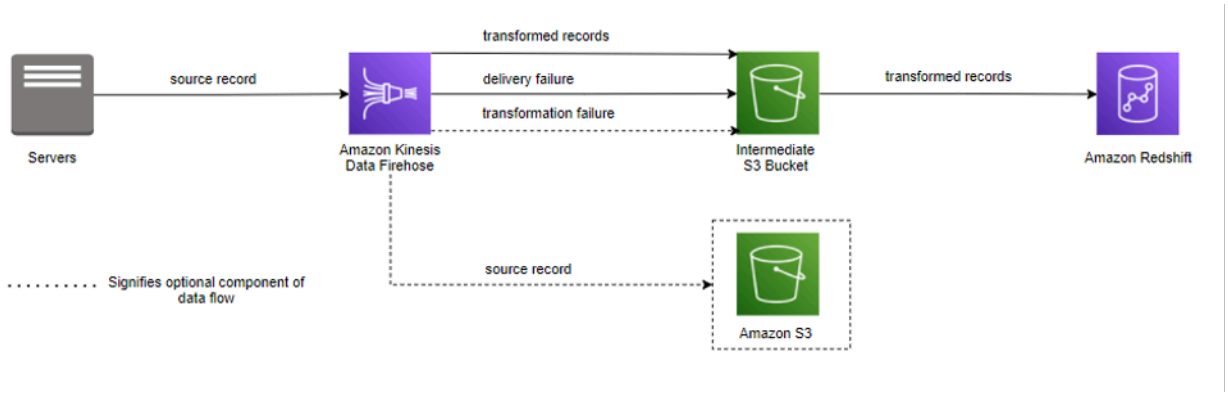
Pengiriman data ke Amazon Redshift

Untuk pengiriman data ke Amazon Redshift, Kinesis Data Firehose pertama kali mengirimkan data masuk ke bucket S3 Anda dalam format yang dijelaskan sebelumnya. Kinesis Data Firehose kemudian mengeluarkan perintah COPY Amazon Redshift untuk memuat data dari bucket S3 ke klaster Amazon Redshift Anda.

Frekuensi operasi `COPY` data dari S3 ke Amazon Redshift ditentukan berdasarkan seberapa cepat klaster Amazon Redshift Anda dapat menyelesaikan perintah `COPY`. Untuk tujuan Amazon Redshift,

Anda dapat menentukan durasi percobaan ulang (0-7200 detik) saat membuat aliran pengiriman untuk menangani kegagalan pengiriman data. Kinesis Data Firehose mencoba ulang selama durasi waktu yang ditentukan dan melewati batch objek S3 tersebut jika tidak berhasil. Informasi objek yang dilewati dikirimkan ke bucket S3 Anda sebagai file manifes dalam folder “errors/”, yang dapat Anda gunakan untuk backfill manual.

Berikut ini adalah diagram arsitektur aliran data Kinesis Data Firehose ke Amazon Redshift. Meskipun aliran data ini unik untuk Amazon Redshift, Kinesis Data Firehose mengikuti pola serupa untuk target tujuan lainnya.



Aliran data dari Kinesis Data Firehose ke Amazon Redshift

Amazon OpenSearch Service (OpenSearch Service)

OpenSearch Service adalah layanan terkelola penuh yang memberikan API OpenSearch yang mudah digunakan dan kemampuan waktu nyata bersama dengan ketersediaan, skalabilitas, dan keamanan yang diperlukan oleh beban kerja produksi. OpenSearch Service memudahkan untuk men-deploy, mengoperasikan, dan menskalakan OpenSearch untuk analitik log, pencarian teks lengkap, dan pemantauan aplikasi.

Pengiriman data ke OpenSearch Service

Untuk pengiriman data ke OpenSearch Service, Kinesis Data Firehose melakukan buffer terhadap data masuk berdasarkan konfigurasi buffering dari aliran pengiriman Anda, lalu menghasilkan permintaan OpenSearch massal untuk mengindeks sejumlah catatan ke klaster OpenSearch Anda. Frekuensi pengiriman data ke OpenSearch Service ditentukan oleh nilai ukuran buffer (1 MB sampai 100 MB) dan interval buffer (60 detik sampai 900 detik) di OpenSearch, mana yang tercapai lebih dulu.

Untuk tujuan OpenSearch Service, Anda dapat menentukan durasi percobaan ulang (0-7200 detik) saat membuat aliran pengiriman. Kinesis Data Firehose mencoba ulang selama durasi waktu yang

ditentukan, lalu melewati permintaan indeks tersebut. Dokumen yang dilewati dikirim ke bucket S3 Anda di folder `elasticsearch_failed/`, yang dapat Anda gunakan untuk backfill manual.

Amazon Kinesis Data Firehose dapat merotasi indeks OpenSearch Service Anda berdasarkan durasi waktu. Tergantung pada opsi rotasi yang Anda pilih (`NoRotation`, `OneHour`, `OneDay`, `OneWeek`, atau `OneMonth`), Kinesis Data Firehose menambahkan sebagian dari stempel waktu kedatangan Waktu Universal Terkoordinasi (UTC) ke nama indeks yang Anda tentukan.

Titik akhir HTTP kustom atau penyedia layanan pihak ketiga yang didukung

Kinesis Data Firehose dapat mengirim data baik ke titik akhir HTTP Kustom atau penyedia pihak ketiga yang didukung seperti Datadog, Dynatrace, LogicMonitor, MongoDB, New Relic, Splunk, dan Sumo Logic.

Titik akhir HTTP kustom atau penyedia layanan pihak ketiga yang didukung

Agar Kinesis Data Firehose berhasil mengirimkan data ke titik akhir HTTP kustom, titik akhir ini harus menerima permintaan dan mengirim respons menggunakan format permintaan dan respons Kinesis Data Firehose tertentu.

Saat mengirimkan data ke titik akhir HTTP yang dimiliki oleh penyedia layanan pihak ketiga yang didukung, Anda dapat menggunakan layanan AWS Lambda terintegrasi untuk membuat fungsi guna mentransformasikan catatan masuk ke format yang sesuai dengan format yang diharapkan oleh integrasi penyedia layanan.

Untuk frekuensi pengiriman data, setiap penyedia layanan memiliki ukuran buffer yang direkomendasikan. Hubungi penyedia layanan Anda untuk informasi lebih lanjut tentang ukuran buffer yang direkomendasikan. Untuk penanganan kegagalan pengiriman data, Kinesis Data Firehose membuat koneksi dengan titik akhir HTTP terlebih dahulu dengan menunggu respons dari tujuan. Kinesis Data Firehose terus membuat koneksi, sampai durasi percobaan ulang berakhir. Setelah itu, Kinesis Data Firehose menganggapnya sebagai kegagalan pengiriman data dan mencadangkan data ke bucket S3 Anda.

Ringkasan

Kinesis Data Firehose dapat terus-menerus mengirimkan data pengaliran Anda ke tujuan yang didukung. Layanan ini adalah solusi terkelola penuh, sehingga membutuhkan sedikit atau tidak ada pengembangan. Untuk Perusahaan ABC2Badge, menggunakan Kinesis Data Firehose adalah pilihan alami. Mereka sudah menggunakan Amazon Redshift sebagai solusi gudang data mereka. Karena

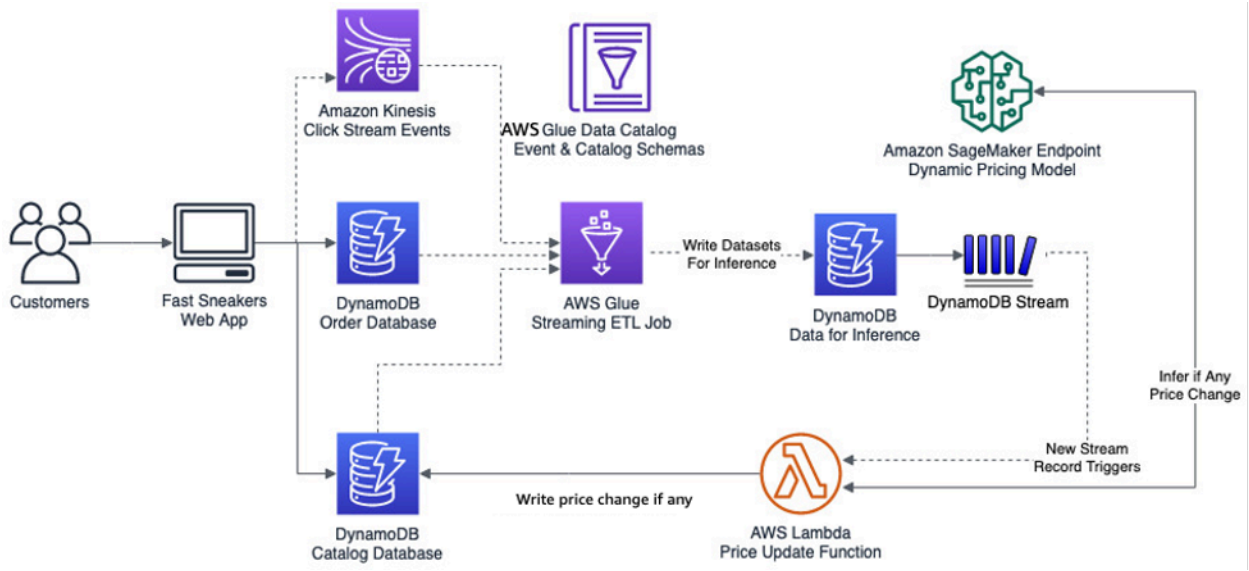
sumber data mereka terus menulis ke log transaksi, mereka dapat memanfaatkan Amazon Kinesis Agent untuk melakukan pengaliran data tersebut tanpa menulis kode tambahan apa pun. Sekarang perusahaan ABC2Badge telah membuat aliran catatan sensor dan menerima catatan ini melalui Kinesis Data Firehose. Mereka dapat menggunakan hal ini sebagai dasar untuk kasus penggunaan tim keamanan.

Skenario 3: Mempersiapkan data clickstream untuk proses wawasan data

Fast Sneakers adalah butik fesyen dengan fokus pada sepatu trendi. Harga untuk sepasang sepatu dapat naik atau turun tergantung pada persediaan dan tren, misalnya siapa selebriti atau bintang olahraga yang terlihat mengenakan sepatu bermerek di TV tadi malam. Penting bagi Fast Sneakers untuk melacak dan menganalisis tren tersebut guna memaksimalkan pendapatan mereka.

Fast Sneakers tidak ingin menimbulkan overhead tambahan ke dalam proyek dengan infrastruktur baru yang perlu dipelihara. Mereka ingin dapat membagi pengembangan ke pihak-pihak yang sesuai, sehingga rekayasawan data dapat fokus pada transformasi data dan ilmuwan data mereka dapat mengerjakan fungsi ML-nya secara independen.

Untuk bereaksi dengan cepat dan menyesuaikan harga sesuai permintaan secara otomatis, Fast Sneakers mengalirkan peristiwa penting (seperti data klik minat dan pembelian), mentransformasikan dan melengkapi data peristiwa ini, lalu mengumpankan data peristiwa ini ke sebuah model ML. Model ML-nya mampu menentukan apakah penyesuaian harga diperlukan. Hal ini memungkinkan Fast Sneakers mengubah harga mereka secara otomatis untuk memaksimalkan keuntungan pada produk mereka.



Penyesuaian harga waktu nyata Fast Sneakers

Diagram arsitektur ini menunjukkan solusi pengaliran waktu nyata yang dibuat Fast Sneakers menggunakan Kinesis Data Streams, AWS Glue, dan DynamoDB Streams. Dengan memanfaatkan layanan ini, mereka memiliki solusi yang elastis dan dapat diandalkan tanpa perlu menghabiskan waktu untuk menyiapkan dan memelihara infrastruktur pendukung. Mereka dapat menghabiskan waktu mereka pada berbagai hal yang menghasilkan nilai bagi perusahaan mereka dengan berfokus pada tugas Extract, Transform, Load (ETL) pengaliran dan model machine learning mereka.

Untuk lebih memahami arsitektur dan teknologi yang digunakan dalam beban kerja mereka, berikut ini adalah beberapa detail layanan yang digunakan.

Pengaliran AWS Glue dan AWS Glue

AWS Glue adalah layanan ETL terkelola penuh yang dapat Anda gunakan untuk mengatalogkan data Anda, membersihkannya, memperkaya, dan memindahkannya dengan andal di antara penyimpanan data. Dengan AWS Glue, Anda dapat secara signifikan mengurangi biaya, kompleksitas, dan waktu yang dihabiskan untuk membuat tugas ETL. AWS Glue bersifat nirserver, jadi tidak ada infrastruktur yang perlu disiapkan atau dikelola. Anda hanya membayar sumber daya yang digunakan saat tugas Anda berjalan.

Dengan memanfaatkan AWS Glue, Anda dapat membuat aplikasi konsumen dengan tugas ETL pengaliran AWS Glue. Hal ini memungkinkan Anda menggunakan Apache Spark dan penulisan modul berbasis Spark lainnya untuk mengonsumsi dan memproses data peristiwa Anda. Bagian selanjutnya dari dokumen ini lebih mendalami skenario ini.

AWS Glue Data Catalog

AWS Glue Data Catalog berisi referensi ke data yang digunakan sebagai sumber dan target tugas ETL Anda di AWS Glue. AWS Glue Data Catalog adalah indeks ke lokasi, skema, dan metrik runtime data Anda. Anda dapat menggunakan informasi dalam Data Catalog untuk membuat dan memantau tugas ETL Anda. Informasi dalam Data Catalog disimpan sebagai tabel metadata, yang masing-masing menentukan penyimpanan data tunggal. Dengan menyiapkan crawler, Anda dapat secara otomatis menilai berbagai jenis penyimpanan data, termasuk DynamoDB, S3, dan Java Database Connectivity (JDBC) yang terhubung penyimpanan, mengekstrak metadata dan skema, lalu membuat definisi tabel di AWS Glue Data Catalog.

Untuk bekerja dengan Amazon Kinesis Data Streams dalam tugas ETL pengaliran AWS Glue, praktik terbaiknya adalah menentukan aliran Anda dalam tabel di basis data AWS Glue Data Catalog. Anda menentukan tabel bersumber aliran dengan aliran Kinesis, yang merupakan salah satu dari banyak

format yang didukung (CSV, JSON, ORC, Parquet, Avro, atau format pelanggan dengan Grok). Anda dapat secara manual memasukkan skema, atau Anda dapat membuat langkah ini ditentukan oleh tugas AWS Glue Anda selama runtime tugas.

Tugas ETL pengaliran AWS Glue

AWS Glue menjalankan tugas ETL Anda di lingkungan nirserver Apache Spark. AWS Glue menjalankan tugas ini pada sumber daya virtual yang disediakan dan dikelolanya dalam akun layanannya sendiri. Selain mampu menjalankan tugas berbasis Apache Spark, AWS Glue menyediakan tingkat fungsionalitas tambahan selain Spark dengan DynamicFrames.

DynamicFrames adalah tabel terdistribusi yang mendukung data nested seperti struktur dan array. Setiap catatan mendeskripsikan diri sendiri, dan dirancang untuk fleksibilitas skema dengan data semi-terstruktur. Sebuah catatan dalam `DynamicFrame` berisi data dan skema yang mendeskripsikan data. Apache Spark `DataFrames` dan `DynamicFrames` didukung dalam skrip ETL Anda, dan Anda dapat mengonversinya secara bolak-balik. `DynamicFrames` menyediakan satu set transformasi canggih untuk pembersihan data dan ETL.

Dengan menggunakan Spark Streaming di Tugas AWS Glue Anda, Anda dapat membuat tugas ETL pengaliran yang berjalan terus-menerus, dan mengonsumsi data dari sumber pengaliran seperti Amazon Kinesis Data Streams, Apache Kafka, dan Amazon MSK. Tugas ini dapat membersihkan, menggabungkan, dan mentransformasikan data, lalu memuat hasilnya ke penyimpanan termasuk penyimpanan data Amazon S3, Amazon DynamoDB, atau JDBC.

AWS Glue memproses dan menulis data dalam periode 100 detik, secara default. Hal ini memungkinkan data diproses secara efisien, dan memungkinkan agregasi dilakukan pada data yang tiba lebih lambat dari yang diharapkan. Anda dapat mengonfigurasi rentang periode ini dengan menyesuaikannya untuk mengakomodasi kecepatan dalam respons vs. keakuratan agregasi Anda. Tugas pengaliran AWS Glue menggunakan titik pemeriksaan untuk melacak data yang telah dibaca dari Kinesis Data Stream. Untuk panduan tentang membuat tugas ETL pengaliran di AWS Glue, Anda dapat merujuk ke Menambahkan Tugas ETL Pengaliran di AWS Glue

Amazon DynamoDB

Amazon DynamoDB adalah basis data kunci-nilai dan dokumen yang memberikan performa milidetik satu digit dalam segala skala. Basis data ini terkelola penuh, multi-aktif, multi-Wilayah dan tahan lama dengan keamanan, pencadangan, dan pemulihan bawaan, serta caching dalam memori untuk aplikasi skala internet. DynamoDB dapat menangani lebih dari sepuluh triliun permintaan per hari dan mampu mendukung beban puncak sebesar lebih dari 20 juta permintaan per detik.

Ubah tangkapan data untuk aliran DynamoDB

Aliran DynamoDB adalah alur informasi yang diurutkan tentang perubahan pada item dalam tabel DynamoDB. Ketika Anda mengaktifkan aliran pada sebuah tabel, DynamoDB menangkap informasi tentang setiap modifikasi pada item data dalam tabel ini. DynamoDB berjalan di AWS Lambda agar Anda dapat membuat pemicu—potongan kode yang secara otomatis merespons peristiwa di aliran DynamoDB. Dengan pemicu, Anda dapat membangun aplikasi yang bereaksi terhadap modifikasi data dalam tabel DynamoDB.

Saat aliran diaktifkan pada sebuah tabel, Anda dapat mengaitkan Amazon Resource Name (ARN) aliran dengan fungsi Lambda yang Anda tulis. Segera setelah item dalam tabel dimodifikasi, catatan baru muncul di aliran tabel. AWS Lambda akan melakukan polling terhadap aliran ini dan memanggil fungsi Lambda Anda secara sinkron ketika mendeteksi catatan aliran baru.

Titik akhir layanan Amazon SageMaker dan Amazon SageMaker

Amazon SageMaker adalah platform terkelola penuh yang mendukung developer dan ilmuwan data dengan kemampuan untuk membangun, melatih, dan men-deploy model ML dengan cepat dan pada skala apa pun. SageMaker berisi modul yang dapat digunakan bersama-sama atau sendiri untuk membangun, melatih, dan men-deploy model ML Anda. Dengan titik akhir layanan Amazon SageMaker, Anda dapat membuat titik akhir host terkelola untuk inferensi waktu nyata dengan model yang di-deploy yang Anda kembangkan di dalam atau di luar Amazon SageMaker.

Dengan memanfaatkan SDK AWS, Anda dapat memanggil titik akhir SageMaker yang meneruskan informasi jenis konten beserta konten lalu menerima prediksi waktu nyata berdasarkan data yang diteruskan. Hal ini memungkinkan Anda menjaga desain dan pengembangan model ML secara terpisah dari kode Anda yang melakukan tindakan pada hasil yang diinferensi.

Hal ini memungkinkan ilmuwan data Anda fokus pada ML, dan developer yang menggunakan model ML ini dapat fokus pada cara menggunakannya dalam kode mereka. Untuk informasi selengkapnya tentang cara memanggil titik akhir di SageMaker, lihat InvokeEnpoint dalam Referensi API Amazon SageMaker.

Menginferensi wawasan data secara waktu nyata

Diagram arsitektur sebelumnya menunjukkan bahwa aplikasi web Fast Sneakers yang ada menambahkan Kinesis Data Stream yang berisi peristiwa clickstream, yang menyediakan data lalu lintas dan peristiwa dari situs web. Katalog produk, yang berisi informasi seperti kategorisasi, atribut produk, dan harga, serta tabel pesanan, yang memiliki data seperti item yang dipesan, penagihan,

pengiriman, dan sebagainya, adalah tabel DynamoDB terpisah. Sumber aliran data dan tabel DynamoDB yang sesuai akan memiliki metadata dan skema yang didefinisikan dalam AWS Glue Data Catalog yang akan digunakan oleh tugas ETL pengaliran AWS Glue.

Dengan memanfaatkan Apache Spark, Spark Streaming, dan `DynamicFrames` dalam tugas ETL pengaliran AWS Glue mereka, Fast Sneakers dapat mengekstrak data dari aliran data dan mentransformasikannya, dengan menggabungkan data dari tabel produk dan pesanan. Dengan data terhidrasi dari transformasi ini, set data yang digunakan untuk mendapatkan hasil inferensi akan dikirim ke tabel DynamoDB.

DynamoDB Stream untuk tabel ini akan memicu fungsi Lambda untuk setiap catatan baru yang ditulis. Fungsi Lambda mengirimkan catatan yang ditransformasi sebelumnya ke SageMaker Endpoint dengan SDK AWS untuk menginferensi, jika ada, penyesuaian harga apa yang diperlukan untuk suatu produk. Jika model ML mengidentifikasi bahwa penyesuaian terhadap harga diperlukan, fungsi Lambda menulis perubahan harga pada produk dalam tabel DynamoDB katalog.

Ringkasan

Amazon Kinesis Data Streams memudahkan pengumpulan, pemrosesan, dan analisis data pengaliran waktu nyata agar Anda dapat memperoleh wawasan secara tepat waktu dan bereaksi cepat terhadap informasi baru. Dikombinasikan dengan layanan integrasi data nirserver AWS Glue, Anda dapat membuat aplikasi pengaliran peristiwa waktu nyata yang mempersiapkan dan menggabungkan data untuk ML.

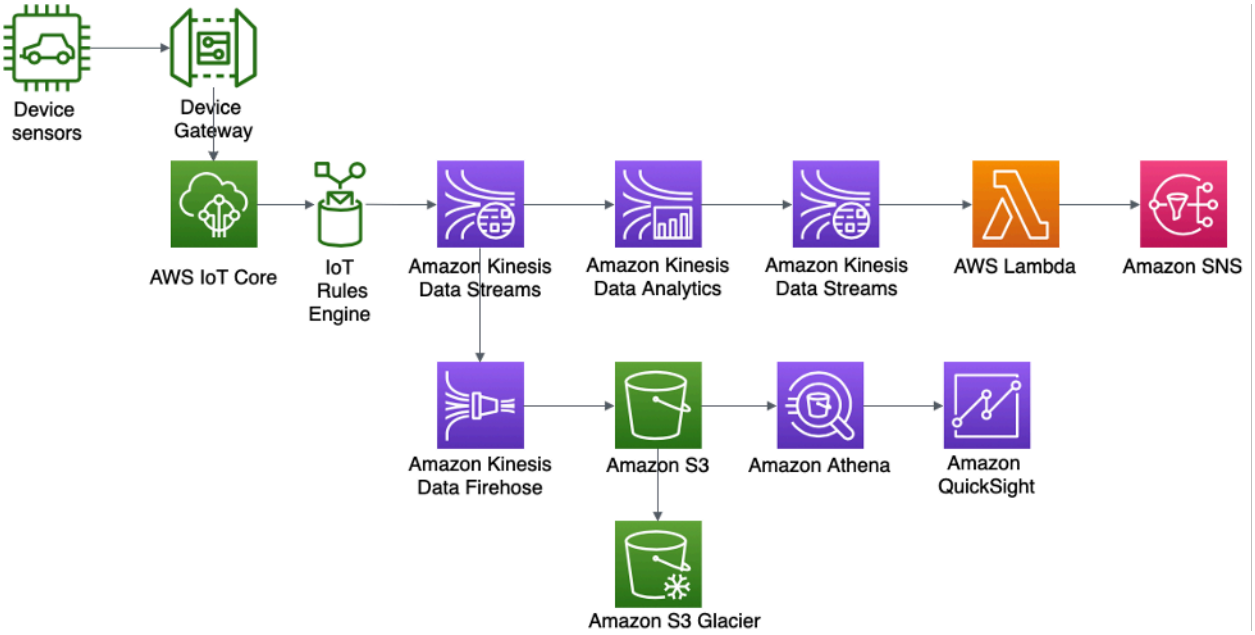
Karena Kinesis Data Streams dan layanan AWS Glue terkelola penuh, AWS menghilangkan pekerjaan berat yang tidak terdiferensiasi dalam mengelola infrastruktur untuk platform big data Anda, sehingga memungkinkan Anda fokus untuk menghasilkan wawasan data berdasarkan data Anda.

Fast Sneakers dapat memanfaatkan pemrosesan peristiwa waktu nyata dan ML untuk memungkinkan situs web mereka membuat penyesuaian harga waktu nyata yang sepenuhnya otomatis, untuk memaksimalkan stok produk mereka. Hal ini menghasilkan nilai terbanyak bagi bisnis mereka sambil menghindari kebutuhan untuk membuat dan memelihara platform big data.

Skenario 4: Deteksi dan notifikasi anomali waktu nyata sensor perangkat

Perusahaan ABC4Logistics mengangkut produk minyak bumi yang sangat mudah terbakar seperti bensin, gas minyak cair (LPG), dan nafta dari pelabuhan ke berbagai kota. Ada ratusan kendaraan

yang memiliki sejumlah sensor yang terpasang untuk memantau hal-hal seperti lokasi, suhu mesin, suhu di dalam kontainer, kecepatan mengemudi, lokasi parkir, kondisi jalan, dan sebagainya. Salah satu kebutuhan ABC4Logistics adalah memantau suhu mesin dan kontainer secara waktu nyata serta memberi tahu pengemudi dan tim pemantauan armada jika terjadi anomali. Untuk mendeteksi kondisi tersebut dan menghasilkan pemberitahuan secara waktu nyata, ABC4Logistics menerapkan arsitektur berikut di AWS.



Arsitektur deteksi dan notifikasi anomali waktu nyata sensor perangkat ABC4Logistics

Data dari sensor perangkat diserap oleh AWS IoT Gateway, tempat mesin aturan AWS IoT akan membuat data pengaliran tersedia di Amazon Kinesis Data Streams. Menggunakan Kinesis Data Analytics, ABC4Logistics dapat melakukan analitik waktu nyata pada data pengaliran di Kinesis Data Streams.

Menggunakan Kinesis Data Analytics, ABC4Logistics dapat mendeteksi jika pembacaan suhu dari sensor menyimpang dari pembacaan normal selama sepuluh detik, dan menyerap catatannya ke instans Kinesis Data Streams lain, dengan mengidentifikasi catatan yang beranomali. Amazon Kinesis Data Streams kemudian memanggil fungsi Lambda, yang dapat mengirim pemberitahuan ke pengemudi dan tim pemantauan armada melalui Amazon SNS.

Data dalam Kinesis Data Streams juga didorong ke Amazon Kinesis Data Firehose. Amazon Kinesis Data Firehose mempersistensi data ini di Amazon S3, sehingga memungkinkan ABC4Logistics melakukan analitik batch atau hampir waktu nyata terhadap data sensor. ABC4Logistics menggunakan Amazon Athena untuk mengkueri data di S3, dan Amazon QuickSight untuk

visualisasi. Untuk retensi data jangka panjang, kebijakan S3 Lifecycle digunakan untuk mengarsipkan data ke Amazon S3 Glacier.

Komponen penting dari arsitektur ini diuraikan selanjutnya.

Amazon Kinesis Data Analytics

Amazon Kinesis Data Analytics memungkinkan Anda mentransformasikan dan menganalisis data pengaliran dan merespons anomali secara waktu nyata. Ini adalah layanan nirserver di AWS, yang berarti Kinesis Data Analytics menangani penyediaan, dan secara elastis menskalakan infrastruktur untuk menangani throughput data apa pun. Layanan ini menghilangkan semua pekerjaan berat yang tidak terdiferensiasi dalam menyiapkan dan mengelola infrastruktur pengaliran, dan memungkinkan Anda meluangkan lebih banyak waktu untuk menulis aplikasi pengaliran.

Dengan Amazon Kinesis Data Analytics, Anda dapat secara interaktif mengkueri data pengaliran menggunakan sejumlah opsi, termasuk Standard SQL, aplikasi Apache Flink di Java, Python, dan Scala, serta membangun aplikasi Apache Beam menggunakan Java untuk menganalisis aliran data.

Sejumlah opsi ini memberi Anda fleksibilitas dalam menggunakan pendekatan tertentu tergantung pada tingkat kompleksitas aplikasi pengaliran dan dukungan sumber/target. Bagian berikut membahas opsi Aplikasi Kinesis Data Analytics for Flink.

Aplikasi Amazon Kinesis Data Analytics for Apache Flink

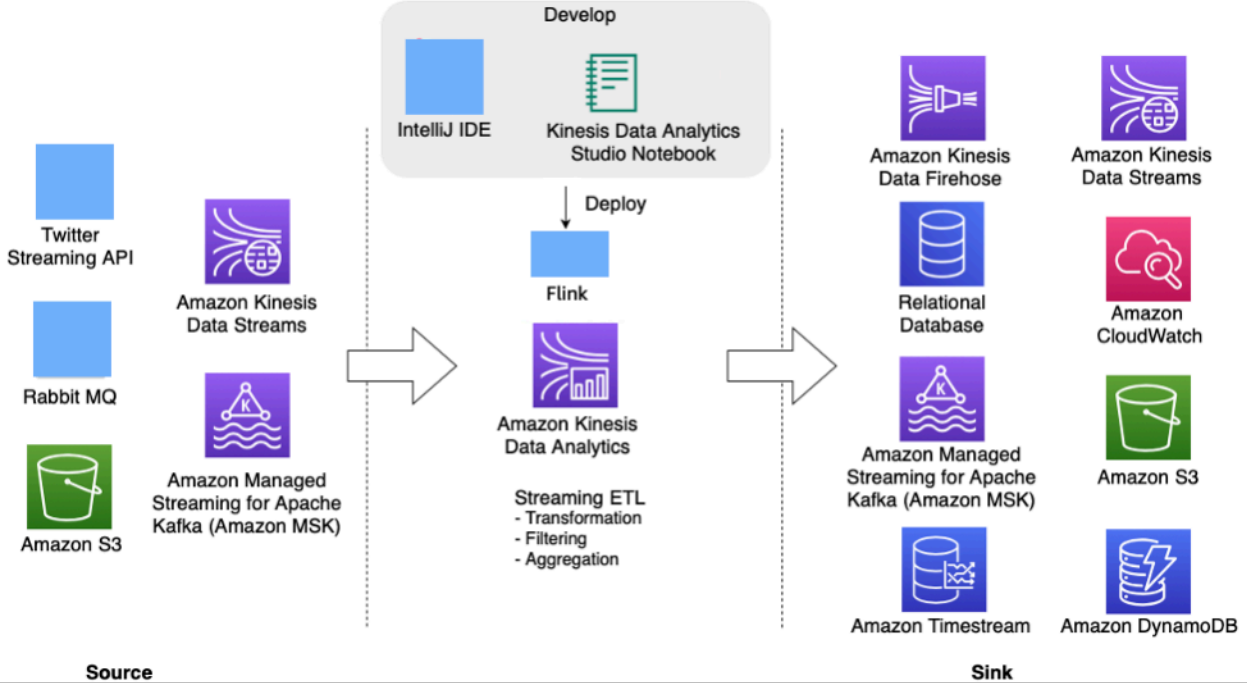
Apache Flink adalah kerangka kerja sumber terbuka dan mesin pemrosesan terdistribusi populer untuk komputasi stateful terhadap aliran data yang tak terbatas dan terbatasi. Apache Flink dirancang untuk melakukan komputasi pada kecepatan dalam memori dan dalam skala besar dengan dukungan untuk semantik persis sekali. Aplikasi berbasis Flink Apache membantu mencapai latensi rendah bersama throughput tinggi dengan cara yang toleran terhadap kesalahan.

Dengan Amazon Kinesis Data Analytics for Apache Flink, Anda dapat menulis dan menjalankan kode terhadap sumber pengaliran untuk melakukan analitik deret waktu, mengirim umpan ke dasbor waktu nyata, dan membuat metrik waktu nyata tanpa mengelola lingkungan Apache Flink terdistribusi yang kompleks. Anda dapat menggunakan fitur pemrograman Flink tingkat tinggi dengan cara yang sama seperti saat meng-host sendiri infrastruktur Flink.

Kinesis Data Analytics for Apache Flink memungkinkan Anda membuat aplikasi di Java, Scala, Python, atau SQL untuk memproses dan menganalisis data pengaliran. Sebuah aplikasi Flink standar akan membaca data dari aliran atau lokasi data input atau sumber, mentransformasikan/memfilter

atau menggabungkan data menggunakan operator atau fungsi, serta menyimpan data pada aliran atau lokasi data output, atau sink.

Diagram arsitektur berikut menunjukkan beberapa sumber dan sink yang didukung untuk aplikasi Kinesis Data Analytics Flink. Selain konektor bawaan untuk sumber/sink, Anda juga dapat membawa konektor kustom ke berbagai sumber/sink lainnya untuk Aplikasi Flink di Kinesis Data Analytics.



Aplikasi Apache Flink di Kinesis Data Analytics untuk pemrosesan aliran waktu nyata

Developer dapat menggunakan IDE pilihan mereka untuk mengembangkan aplikasi Flink dan men-deploy-nya pada Kinesis Data Analytics dari AWS Management Console atau alat DevOps.

Amazon Kinesis Data Analytics Studio

Sebagai bagian dari layanan Kinesis Data Analytics, Kinesis Data Analytics Studio tersedia bagi pelanggan untuk mengkueri aliran data dalam waktu nyata secara interaktif, dan membangun dan menjalankan aplikasi pemrosesan aliran dengan mudah menggunakan SQL, Python, dan Scala. Notebook Studio didukung oleh Apache Zeppelin.

Menggunakan notebook Studio, Anda memiliki kemampuan untuk mengembangkan kode Aplikasi Flink Anda di lingkungan notebook, melihat hasil kode Anda secara waktu nyata, dan memvisualisasikannya dalam notebook Anda. Anda dapat membuat Notebook Studio yang didukung oleh Apache Zeppelin dan Apache Flink dengan satu klik dari Kinesis Data Streams dan konsol Amazon MSK, atau meluncurkannya dari Kinesis Data Analytics Console.

Setelah Anda mengembangkan kode secara iteratif sebagai bagian dari Kinesis Data Analytics Studio, Anda dapat men-deploy notebook sebagai aplikasi analitik data Kinesis, untuk berjalan dalam mode pengaliran secara terus-menerus, membaca data dari sumber Anda, menulis ke tujuan Anda, mempertahankan status aplikasi yang berjalan lama, dan menskalakan secara otomatis berdasarkan throughput aliran sumber Anda. Sebelumnya, pelanggan menggunakan Kinesis Data Analytics for SQL Applications untuk analitik interaktif terhadap data pengaliran waktu nyata di AWS.

Kinesis Data Analytics for SQL Applications masih tersedia, tetapi untuk proyek baru, AWS merekomendasikan agar Anda menggunakan Kinesis Data Analytics Studio yang baru. Kinesis Data Analytics Studio menggabungkan kemudahan penggunaan dengan kemampuan analitik tingkat lanjut, yang memungkinkan pembangunan aplikasi pemrosesan aliran yang canggih dalam hitungan menit.

Untuk membuat aplikasi Kinesis Data Analytics Flink yang toleran terhadap kesalahan, Anda dapat menggunakan checkpointing dan snapshot, seperti yang dijelaskan dalam Menerapkan Toleransi Kesalahan di Kinesis Data Analytics for Apache Flink.

Aplikasi Kinesis Data Analytics Flink berguna untuk menulis aplikasi analitik pengaliran yang kompleks seperti aplikasi dengan semantik persis sekali pada pemrosesan data, kemampuan checkpointing, dan memproses data dari sumber data seperti Kinesis Data Streams, Kinesis Data Firehose, Amazon MSK, Rabbit MQ, dan Apache Cassandra yang mencakup Konektor Kustom.

Setelah memproses data pengaliran di aplikasi Flink, Anda dapat menyimpan data ke berbagai sink atau tujuan seperti Amazon Kinesis Data Streams, Amazon Kinesis Data Firehose, Amazon DynamoDB, Amazon OpenSearch Service, Amazon Timestream, Amazon S3, dan sebagainya. Aplikasi Kinesis Data Analytics Flink juga memberikan jaminan performa sub-detik.

Apache Beam applications for Kinesis Data Analytics

Apache Beam adalah model pemrograman untuk memproses data pengaliran. Apache Beam menyediakan lapisan API portabel untuk membangun alur pemrosesan paralel data canggih yang dapat dijalankan di berbagai mesin, atau runner seperti Flink, Spark Streaming, Apache Samza, dan sebagainya.

Anda dapat menggunakan kerangka kerja Apache Beam dengan aplikasi analitik data Kinesis Anda untuk memproses data pengaliran. Aplikasi analitik data Kinesis yang menggunakan Apache Beam memanfaatkan runner Apache Flink untuk menjalankan alur Beam.

Ringkasan

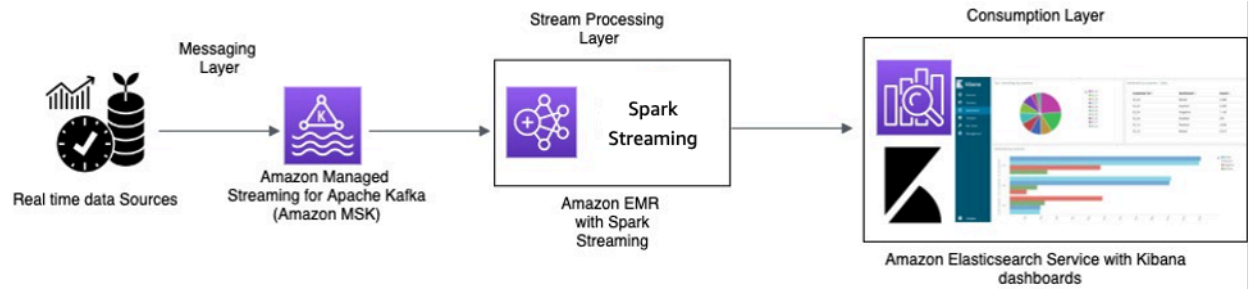
Dengan memanfaatkan layanan pengaliran AWS Amazon Kinesis Data Streams, Amazon Kinesis Data Analytics, dan Amazon Kinesis Data Firehose,

ABC4Logistics dapat mendeteksi pola anomali dalam pembacaan suhu dan memberi tahu pengemudi dan tim manajemen armada secara waktu nyata, sehingga mencegah kecelakaan besar seperti kerusakan kendaraan yang parah atau kebakaran.

Skenario 5: Pemantauan data telemetri waktu nyata dengan Apache Kafka

ABC1Cabs adalah perusahaan layanan pemesanan taksi online. Semua taksi memiliki perangkat IoT yang mengumpulkan data telemetri dari kendaraan. Saat ini, ABC1Cabs menjalankan klaster Apache Kafka yang dirancang untuk konsumsi peristiwa waktu nyata, mengumpulkan metrik kondisi sistem, pelacakan aktivitas, dan mengumpankan data ke platform Apache Spark Streaming yang dibangun di klaster Hadoop secara on-premise.

ABC1Cabs menggunakan OpenSearch Dashboards untuk metrik bisnis, debugging, pemberitahuan, dan membuat dasbor lainnya. Mereka tertarik dengan Amazon MSK, Amazon EMR with Spark Streaming, dan OpenSearch Service with OpenSearch Dashboards. Kebutuhan mereka adalah mengurangi overhead admin dalam memelihara klaster Apache Kafka dan Hadoop, sambil menggunakan perangkat lunak sumber terbuka dan API yang sudah dikenal untuk mengorkestrasi alur data mereka. Diagram arsitektur berikut menampilkan solusi mereka di AWS.



Pemrosesan waktu nyata dengan Amazon MSK dan pemrosesan Aliran menggunakan Apache Spark Streaming di Amazon EMR dan Amazon OpenSearch Service with OpenSearch Dashboards

Perangkat IoT taksi mengumpulkan data telemetri dan mengirimnya ke hub sumber. Hub sumber dikonfigurasi untuk mengirim data secara waktu nyata ke Amazon MSK. Menggunakan API pustaka produsen Apache Kafka, Amazon MSK dikonfigurasi untuk mengalirkan data ke dalam klaster

Amazon EMR. Klaster Amazon EMR memiliki klien Kafka dan Spark Streaming yang telah diinstal untuk dapat mengonsumsi dan memproses aliran data.

Spark Streaming memiliki konektor sink yang dapat menulis data langsung ke indeks yang didefinisikan di Elasticsearch. Klaster Elasticsearch dengan OpenSearch Dashboards dapat digunakan untuk metrik dan dasbor. Amazon MSK, Amazon EMR with Spark Streaming, dan OpenSearch Service with OpenSearch Dashboards adalah layanan terkelola, tempat AWS mengelola pekerjaan berat yang tidak terdiferensiasi dalam manajemen infrastruktur terhadap berbagai klaster, yang memungkinkan Anda membangun aplikasi menggunakan perangkat lunak sumber terbuka yang sudah dikenal dengan beberapa klik . Bagian selanjutnya akan mengamati lebih dekat layanan tersebut.

Amazon Managed Streaming for Apache Kafka (Amazon MSK)

Apache Kafka adalah platform sumber terbuka yang memungkinkan pelanggan menangkap data pengaliran seperti peristiwa clickstream, transaksi, peristiwa IoT, serta log aplikasi dan mesin. Dengan informasi ini, Anda dapat mengembangkan aplikasi yang melakukan analitik waktu nyata, menjalankan transformasi berkelanjutan, serta mendistribusikan data ini ke danau data dan basis data secara waktu nyata.

Anda dapat menggunakan Kafka sebagai penyimpanan data pengaliran untuk memisahkan aplikasi dari produsen dan konsumen serta memungkinkan transfer data yang dapat diandalkan di antara kedua komponen. Meskipun Kafka adalah platform pengaliran dan olahpesan data korporasi yang populer, layanan ini dapat sulit untuk disiapkan, diskalakan, dan dikelola dalam produksi.

Amazon MSK menangani tugas-tugas pengelolaan ini dan memudahkan untuk menyiapkan, mengonfigurasi, dan menjalankan Kafka, bersama dengan Apache Zookeeper, dalam lingkungan yang mengikuti praktik terbaik untuk ketersediaan dan keamanan yang tinggi. Anda masih dapat menggunakan operasi bidang kontrol Kafka dan operasi bidang data untuk mengelola produksi dan konsumsi data.

Karena Amazon MSK menjalankan dan mengelola Apache Kafka sumber terbuka, layanan ini memudahkan pelanggan untuk memigrasikan dan menjalankan aplikasi Apache Kafka yang ada di AWS tanpa perlu melakukan perubahan pada kode aplikasi mereka.

Penskalaan

Amazon MSK menawarkan operasi penskalaan sehingga pengguna dapat menskalakan klaster secara aktif saat klaster berjalan. Saat membuat klaster Amazon MSK, Anda dapat menentukan jenis instans broker pada peluncuran klaster. Anda dapat memulai dengan beberapa broker dalam klaster

Amazon MSK. Kemudian, dengan menggunakan AWS Management Console atau AWS CLI, Anda dapat menskalakan hingga ratusan broker per klaster.

Atau, Anda dapat menskalakan klaster Anda dengan mengubah ukuran atau kelompok broker Apache Kafka Anda. Dengan mengubah ukuran atau kelompok broker, Anda memiliki fleksibilitas untuk menyesuaikan kapasitas komputasi klaster Amazon MSK Anda sesuai dengan perubahan beban kerja. Gunakan spreadsheet Amazon MSK Sizing and Pricing (unduhan file) untuk menentukan jumlah broker yang benar untuk klaster Amazon MSK Anda. Spreadsheet ini memberikan perkiraan untuk ukuran klaster Amazon MSK dan biaya terkait Amazon MSK dibandingkan dengan klaster Apache Kafka berbasis EC2 yang serupa dan dikelola sendiri.

Setelah membuat klaster Amazon MSK, Anda dapat meningkatkan jumlah penyimpanan EBS per broker, kecuali mengurangi penyimpanan. Volume penyimpanan tetap tersedia selama operasi penaikan skala ini. Layanan ini menawarkan dua jenis operasi penskalaan: Penskalaan Otomatis dan Penskalaan Manual.

Amazon MSK mendukung ekspansi otomatis penyimpanan klaster Anda sebagai respons terhadap peningkatan penggunaan sesuai dengan kebijakan Penskalaan Otomatis Aplikasi. Kebijakan penskalaan otomatis Anda menetapkan pemanfaatan disk target dan kapasitas penskalaan maksimum.

Ambang batas pemanfaatan penyimpanan membantu Amazon MSK memicu operasi penskalaan otomatis. Untuk meningkatkan penyimpanan menggunakan penskalaan manual, tunggu sampai klaster berada dalam status `ACTIVE`. Penskalaan penyimpanan memiliki periode pendinginan selama setidaknya enam jam di antara peristiwa. Meskipun operasi ini menyediakan penyimpanan tambahan dengan segera, layanan ini melakukan optimasi pada klaster Anda yang dapat memakan waktu hingga 24 jam atau lebih.

Durasi optimasi ini sebanding dengan ukuran penyimpanan Anda. Selain itu, layanan ini juga menawarkan replikasi Multi-Zona Ketersediaan dalam Wilayah AWS untuk menyediakan Ketersediaan Tinggi.

Konfigurasi

Amazon MSK menyediakan konfigurasi default untuk broker, topik, dan node Apache Zookeeper. Anda juga dapat membuat konfigurasi kustom dan menggunakannya untuk membuat klaster Amazon MSK baru atau memperbarui klaster yang ada. Saat Anda membuat klaster MSK tanpa menentukan konfigurasi Amazon MSK kustom, Amazon MSK membuat dan menggunakan konfigurasi default. Untuk daftar nilai default, lihat Konfigurasi Apache Kafka ini.

Untuk tujuan pemantauan, Amazon MSK mengumpulkan metrik Apache Kafka dan mengirimkannya ke Amazon CloudWatch, tempat Anda dapat melihatnya. Metrik yang Anda konfigurasikan untuk klaster MSK secara otomatis dikumpulkan dan didorong ke CloudWatch. Memantau lag konsumen memungkinkan Anda mengidentifikasi konsumen yang lambat atau macet serta tidak mengikuti data terbaru yang tersedia dalam suatu topik. Jika perlu, Anda kemudian dapat melakukan tindakan perbaikan, seperti penskalaan atau reboot konsumen tersebut.

Bermigrasi ke Amazon MSK

Migrasi dari on-premise ke Amazon MSK dapat dicapai dengan salah satu metode berikut.

- MirrorMaker2.0 — MirrorMaker2.0 (MM2) MM2 adalah mesin replikasi data multi-klaster berdasarkan kerangka kerja Apache Kafka Connect. MM2 adalah kombinasi dari konektor sumber dan konektor sink Apache Kafka. Anda dapat menggunakan klaster MM2 tunggal untuk memigrasikan data di antara sejumlah klaster. MM2 secara otomatis mendeteksi topik dan partisi baru, sambil memastikan konfigurasi topik disinkronkan di antara klaster. MM2 mendukung ACL migrasi, konfigurasi topik, dan terjemahan offset. Untuk detail selengkapnya terkait migrasi, lihat Memigrasi Klaster Menggunakan MirrorMaker Apache Kafka. MM2 digunakan untuk kasus penggunaan yang terkait dengan replikasi konfigurasi topik dan terjemahan offset secara otomatis.
- Apache Flink — MM2 mendukung semantik setidaknya sekali. Catatan dapat diduplikasi ke tujuan dan konsumen diharapkan akan bersifat idempoten untuk menangani catatan duplikat. Dalam skenario yang memerlukan semantik persis sekali, pelanggan dapat menggunakan Apache Flink. Layanan ini memberikan alternatif untuk mencapai semantik persis sekali.

  Apache Flink juga dapat digunakan untuk skenario saat data memerlukan tindakan pemetaan atau transformasi sebelum dikirim ke klaster tujuan. Apache Flink menyediakan konektor untuk Apache Kafka dengan sumber dan sink yang dapat membaca data dari satu klaster Apache Kafka dan menulis ke yang lain. Apache Flink dapat dijalankan di AWS dengan meluncurkan klaster Amazon EMR atau dengan menjalankan Apache Flink sebagai aplikasi menggunakan Amazon Kinesis Data Analytics.
- AWS Lambda — Dengan dukungan untuk Apache Kafka sebagai sumber peristiwa untuk AWS Lambda, pelanggan sekarang dapat mengonsumsi pesan dari topik melalui fungsi Lambda. Layanan AWS Lambda secara internal melakukan polling untuk catatan atau pesan baru dari sumber peristiwa, lalu secara sinkron memanggil fungsi Lambda target untuk mengonsumsi pesan-pesan ini. Lambda membaca pesan dalam batch dan menyediakan batch pesan ke fungsi Anda dalam muatan peristiwa untuk diproses. Pesan yang dikonsumsi kemudian dapat ditransformasi dan/atau ditulis langsung ke klaster Amazon MSK tujuan Anda.

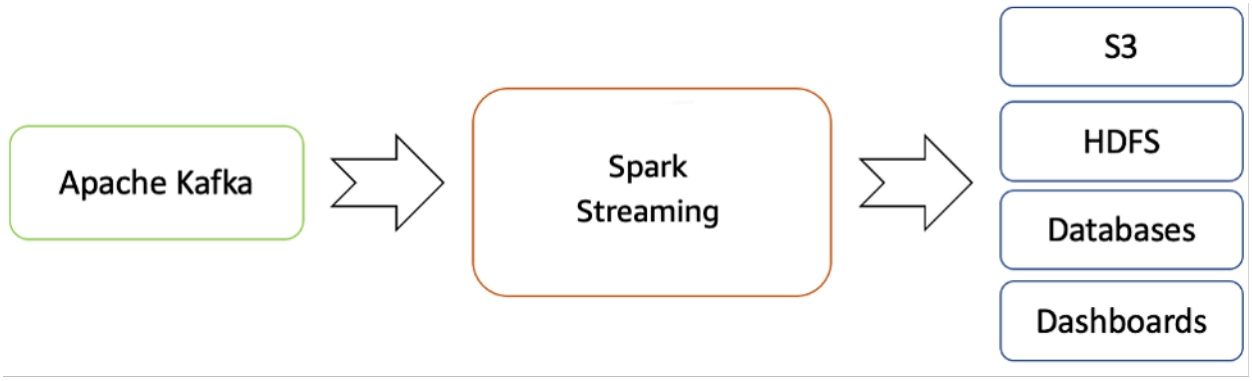
Amazon EMR with Spark Streaming

Amazon EMR adalah platform klaster terkelola yang memudahkan dalam menjalankan kerangka kerja big data, seperti Apache Hadoop dan Apache Spark di AWS, untuk memproses dan menganalisis data dalam jumlah besar.

Amazon EMR menyediakan kemampuan Spark dan dapat digunakan untuk memulai pengaliran Spark untuk mengonsumsi data dari Kafka. Spark Streaming adalah ekstensi dari API Spark inti yang memungkinkan pemrosesan aliran data langsung aliran yang dapat diskalakan, memiliki throughput tinggi, dan toleran terhadap kesalahan.

Anda dapat membuat klaster Amazon EMR menggunakan AWS Command Line Interface (AWS CLI) atau di AWS Management Console lalu memilih Spark dan Zeppelin dalam konfigurasi lanjutan saat membuat klaster. Seperti yang ditunjukkan dalam diagram arsitektur berikut, data dapat diserap dari banyak sumber seperti Apache Kafka dan Kinesis Data Streams, dan dapat diproses menggunakan algoritma kompleks yang diekspresikan dengan fungsi tingkat tinggi seperti map, reduce, join, dan window. Untuk informasi selengkapnya, lihat Transformasi di DStream.

Data yang diproses dapat didorong keluar ke sistem file, basis data, dan dasbor langsung.



Alur pengaliran waktu nyata dari Apache Kafka ke ekosistem Hadoop

Secara default, Apache Spark Streaming memiliki model proses mikro-batch. Namun, sejak Spark 2.3 dirilis, Apache telah memperkenalkan mode pemrosesan latensi rendah baru yang disebut Continuous Processing, yang dapat mencapai latensi ujung ke ujung satu milidetik dengan jaminan paling sedikit sekali.

Tanpa mengubah operasi Dataset/DataFrames dalam kueri Anda, Anda dapat memilih mode berdasarkan persyaratan aplikasi Anda. Beberapa manfaat dari Spark Streaming adalah:

- Ekstensi ini menghadirkan API terintegrasi bahasa Apache Spark ke pemrosesan aliran, sehingga memungkinkan Anda menulis tugas pengaliran dengan cara yang sama seperti menulis tugas batch.
- Ekstensi ini mendukung Java, Scala, dan Python.
- Ekstensi ini dapat memulihkan pekerjaan dan status operator yang hilang (seperti sliding window) tanpa konfigurasi dan kode tambahan dari Anda.
- Dengan menjalankan di Spark, Spark Streaming memungkinkan Anda menggunakan kembali kode yang sama untuk pemrosesan batch, menggabungkan aliran berdasarkan data historis, atau menjalankan kueri ad hoc pada status aliran dan membangun aplikasi interaktif yang canggih, bukan hanya analitik.
- Setelah aliran data diproses dengan Spark Streaming, OpenSearch Sink Connector dapat digunakan untuk menulis data ke klaster OpenSearch Service, lalu OpenSearch Service with OpenSearch Dashboards dapat digunakan sebagai lapisan konsumsi.

Amazon OpenSearch Service with OpenSearch Dashboards

OpenSearch Service adalah layanan terkelola yang memudahkan deployment, pengoperasian, dan penskalaan klaster OpenSearch di AWS Cloud. OpenSearch adalah mesin pencarian dan analitik sumber terbuka yang populer untuk kasus penggunaan seperti analitik log, pemantauan aplikasi waktu nyata, dan analitik clickstream.

OpenSearch Dashboards adalah alat visualisasi dan penjelajahan data sumber terbuka yang digunakan untuk kasus penggunaan analitik log dan deret waktu, pemantauan aplikasi, serta intelijen operasional. Layanan ini menghadirkan fitur yang canggih dan mudah digunakan seperti histogram, grafik garis, diagram lingkaran, peta panas, dan dukungan geospasial bawaan.

OpenSearch Dashboards menyediakan integrasi yang kuat dengan OpenSearch, mesin analitik dan pencarian populer, yang membuat OpenSearch Dashboards sebagai pilihan default untuk memvisualisasikan data yang disimpan dalam OpenSearch. OpenSearch Service menyediakan instalasi OpenSearch Dashboards dengan setiap domain OpenSearch Service. Anda dapat menemukan tautan ke OpenSearch Dashboards di dasbor domain Anda di konsol OpenSearch Service.

Ringkasan

Dengan Apache Kafka yang ditawarkan sebagai layanan terkelola di AWS, Anda dapat fokus pada konsumsi daripada mengelola koordinasi di antara broker, yang biasanya memerlukan pemahaman

mendetail tentang Apache Kafka. Fitur seperti ketersediaan tinggi, skalabilitas broker, dan kontrol akses granular dikelola oleh platform Amazon MSK.

ABC1Cabs memanfaatkan sejumlah layanan ini untuk membangun aplikasi produksi tanpa memerlukan keahlian manajemen infrastruktur. Mereka dapat fokus pada lapisan pemrosesan untuk mengonsumsi data dari Amazon MSK dan menyebarkan secara lebih lanjut ke lapisan visualisasi.

Spark Streaming di Amazon EMR dapat membantu analitik data pengaliran secara waktu nyata, dan publikasi di OpenSearch Dashboards di Amazon OpenSearch Service untuk lapisan visualisasi.

Kesimpulan dan kontributor

Kesimpulan

Dokumen ini meninjau beberapa skenario untuk alur kerja pengaliran. Dalam skenario ini, pemrosesan data pengaliran memberikan contoh kemampuan bagi perusahaan untuk menambahkan fitur dan fungsionalitas baru.

Dengan menganalisis data saat data dibuat, Anda akan mendapatkan wawasan tentang apa yang bisnis Anda lakukan sekarang. Layanan pengaliran AWS memungkinkan Anda fokus pada aplikasi Anda untuk membuat keputusan bisnis yang sensitif terhadap waktu, daripada men-deploy dan mengelola infrastruktur

Kontributor

- Amalia Rabinovitch, Arsitek Solusi Senior (Sr. Solutions Architect), AWS
- Priyanka Chaudhary, Arsitek Data (Data Architect), Danau Data, AWS
- Zohair Nasimi, Arsitek Solusi (Solutions Architect), AWS
- Rob Kuhr, Arsitek Solusi (Solutions Architect), AWS
- Ejaz Sayyed, Arsitek Solusi Partner Senior (Sr. Partner Solutions Architect), AWS
- Allan MacInnis, Arsitek Solusi (Solutions Architect), AWS
- Chander Matrubhutam, Manajer Pemasaran Produk (Product Marketing Manager), AWS

Revisi dokumen

Untuk mendapatkan notifikasi tentang pembaruan laporan resmi ini, sebaiknya berlangganan umpan RSS.

perubahan-riwayat-pembaruan	deskripsi-riwayat-pembaruan	tanggal-riwayat-pembaruan
Diperbarui	Diperbarui untuk akurasi teknis	1 September 2021
Publikasi awal	Laporan resmi pertama kali dipublikasikan	1 Juli 2017