Summary of the Amazon DynamoDB Service Disruption and Related Impacts in the US-East Region

Early Sunday morning, September 20, we had a DynamoDB service event in the US-East Region that impacted DynamoDB customers in US-East, as well as some other services in the region. The following are some additional details on the root cause, subsequent impact to other AWS services that depend on DynamoDB, and corrective actions we're taking.

Some DynamoDB Context

Among its many functions, DynamoDB stores and maintain tables for customers. A single DynamoDB table is separated into partitions, each containing a portion of the table's data. These partitions are spread onto many servers, both to provide consistent low-latency access and to replicate the data for durability.

The specific assignment of a group of partitions to a given server is called a "membership." The membership of a set of table/partitions within a server is managed by DynamoDB's internal metadata service. The metadata service is internally replicated and runs across multiple datacenters. Storage servers hold the actual table data within a partition and need to periodically confirm that they have the correct membership. They do this by checking in with the metadata service and asking for their current membership assignment. In response, the metadata service retrieves the list of partitions and all related information from its own store, bundles this up into a message, and transmits back to the requesting storage server. A storage server will also get its membership assignment after a network disruption or on startup. Once a storage server has completed processing its membership assignment, it verifies that it has the table/partition data locally stored, creates any new table/partitions assigned, and retrieves data from other storage servers to replicate existing partitions assigned.

The DynamoDB Event

On Sunday, at 2:19am PDT, there was a brief network disruption that impacted a portion of DynamoDB's storage servers. Normally, this type of networking disruption is handled seamlessly and without change to the performance of DynamoDB, as affected storage servers query the metadata service for their membership, process any updates, and reconfirm their availability to accept requests. If the storage servers aren't able to retrieve this membership data back within a specific time period, they will retry the membership request and temporarily disqualify themselves from accepting requests.

But, on Sunday morning, a portion of the metadata service responses exceeded the retrieval and transmission time allowed by storage servers. As a result, some of the storage servers were unable to obtain their membership data, and removed themselves from taking requests. The reason these metadata service requests were taking too long relates to a recent development in DynamoDB. Over the last few months, customers have rapidly adopted a new DynamoDB feature called Global Secondary Indexes ("GSIs"). GSIs allow customers to access their table data using alternate keys. Because GSIs are global, they have their own set of partitions on storage servers and therefore increase the overall size of a storage server's membership data. Customers can add multiple GSIs for a given table, so a table with large numbers of partitions could have its contribution of partition data to the membership lists quickly double or triple. With rapid adoption of GSIs by a number of customers with very large tables, the partitions-per-table ratio increased significantly. This, in turn, increased the size of some storage servers' membership lists significantly. With a larger size, the processing time inside the metadata service for some membership requests began to approach the retrieval time allowance by storage servers. We did not have detailed enough monitoring for this dimension (membership size), and didn't have enough capacity allocated to the metadata service to handle these much heavier requests.

So, when the network disruption occurred on Sunday morning, and a number of storage servers simultaneously requested their membership data, the metadata service was processing some membership lists that were now large enough that their processing time was near the time limit for retrieval. Multiple, simultaneous requests for these large memberships caused processing to slow further and eventually exceed the allotted time limit. This resulted in the disrupted storage servers failing to complete their membership renewal, becoming unavailable for requests, and retrying these requests. With the metadata service now under heavy load, it also no longer responded as quickly to storage servers uninvolved in the original network disruption, who were checking their membership data in the normal cadence of when they retrieve this information. Many of those storage servers also became unavailable for handling customer requests. Unavailable servers continued to retry requests for membership data, maintaining high load on the metadata service. Though many storage servers' renewal requests were succeeding, healthy storage servers that had successfully processed a membership request previously were having subsequent renewals fail and were transitioning back to an unavailable state. By 2:37am PDT, the error rate in customer requests to DynamoDB had risen far beyond any level experienced in the last 3 years, finally stabilizing at approximately 55%.

Initially, we were unable to add capacity to the metadata service because it was under such high load, preventing us from successfully making the requisite administrative requests. After several failed attempts at adding capacity, at 5:06am PDT, we decided to pause requests to the metadata service. This action decreased retry activity, which relieved much of the load on the metadata service. With the metadata service now able to respond to administrative requests, we were able to add significant capacity. Once these adjustments were made, we were able to reactivate requests to the metadata service, put storage servers back into the customer request path, and allow normal load back on the metadata service. At 7:10am PDT, DynamoDB was restored to error rates low enough for most customers and AWS services dependent on DynamoDB to resume normal operations.

There's one other bit worth mentioning. After we resolved the key issue on Sunday, we were left with a low error rate, hovering between 0.15%-0.25%. We knew there would be some cleanup to do after the event, and while this rate was higher than normal, it wasn't a rate that usually precipitates a dashboard post or creates issues for customers. As Monday progressed, we started to get more customers opening support cases about being impacted by tables being stuck in the updating or deleting stage or higher than normal error rates. We did not realize soon enough that this low overall error rate was giving some customers disproportionately high error rates. It was impacting a relatively small number of customers, but we should have posted the green-i to the dashboard sooner than we did on Monday. The issue turned out to be a metadata partition that was still not taking the amount of traffic that it should have been taking. The team worked carefully and diligently to restore that metadata partition to its full traffic volume, and closed this out on Monday.

There are several actions we'll take immediately to avoid a recurrence of Sunday's DynamoDB event. First, we have already significantly increased the capacity of the metadata service. Second, we are instrumenting stricter monitoring on performance dimensions, such as the membership size, to allow us to thoroughly understand their state and proactively plan for the right capacity. Third, we are reducing the rate at which storage nodes request membership data and lengthening the time allowed to process queries. Finally and longer term, we are segmenting the DynamoDB service so that it will have many instances of the metadata service each serving only portions of the storage server fleet. This will further contain the impact of software, performance/capacity, or infrastructure failures.

Impact on Other Services

There are several other AWS services that use DynamoDB that experienced problems during the event. Rather than list them all, which had similar explanations for their status, we'll list a few that customers most asked us about or where the actions are more independent from DynamoDB's Correction of Errors ("COE").

Simple Queue Service (SQS)

In the early stages of the DynamoDB event, the Amazon Simple Queue Service was delivering slightly elevated errors and latencies. Amazon SQS uses an internal table stored in DynamoDB to store information describing its queues. While the queue information is cached within SQS, and is not in the direct path for "send-message" and "receive-message" APIs, the caches are refreshed frequently to accommodate creation, deletion, and reassignment across infrastructure. When DynamoDB finished disabling traffic at 5:45am PDT (to enable the metadata service to recover), the Simple Queue Service was unable to read this data to refresh caches, resulting in significantly elevated error rates. Once DynamoDB began re-enabling customer traffic at 7:10am PDT, the Simple Queue Service recovered. No data in queues, or information describing queues was lost as a result of the event.

In addition to the actions being taken by the DynamoDB service, we will be adjusting our SQS metadata caching to ensure that send and receive operations continue even without prolonged access to the metadata table.

EC2 Auto Scaling

Between 2:15am PDT and 7:10am PDT, the EC2 Auto Scaling Service delivered significantly increased API faults. From 7:10am PDT to 10:52am PDT, the Auto Scaling service was substantially delayed in bringing new instances into service, or terminating existing unhealthy instances. Existing instances continued to operate properly throughout the event.

Auto Scaling stores information about its groups and launch configurations in an internal table in DynamoDB. When DynamoDB began to experience elevated error rates starting at 2:19am PDT, Auto Scaling could not update this internal table when APIs were called. Once DynamoDB began recovery at 7:10am PDT, the Auto Scaling APIs recovered. Recovery was incomplete at this time, as a significant backlog of scaling activities had built up throughout the event. The Auto Scaling service executes its launch and termination activities in a background scheduling service. Throughout the event, a very large amount of pending activities built up in this job scheduler and it took until 10:52am PDT to complete all of these tasks.

In addition to the actions taken by the DynamoDB team, to ensure we can recover quickly when a large backlog of scaling activities accumulate, we will adjust the way we partition work on the fleet of Auto Scaling servers to allow for more parallelism in processing these jobs, integrate mechanisms to prune older scaling activities that have been superseded, and increase the capacity available to process scaling activities.

CloudWatch

Starting at 2:35am PDT, the Amazon CloudWatch Metrics Service began experiencing delayed and missing EC2 Metrics along with slightly elevated errors. CloudWatch uses an internal table stored in DynamoDB to add information regarding Auto Scaling group membership to incoming EC2 metrics. From 2:35am PDT to 5:45am PDT, the elevated DynamoDB failure rates caused intermittent availability of EC2 metrics in CloudWatch. CloudWatch also observed an abnormally low rate of metrics publication from other services that were experiencing issues over this time period, further contributing to missing or delayed metrics.

Then, from approximately 5:51am PDT to 7:10am PDT CloudWatch delivered significantly elevated error rates for PutMetricData calls affecting all AWS Service metrics and custom metrics. The impact was due to the significantly elevated error rates in DynamoDB for the group membership additions mentioned above. The CloudWatch Metrics Service was fully recovered at 7:29am PDT.

We understand how important metrics are, especially during an event. To further increase the resilience of CloudWatch, we will adjust our caching strategy for the DynamoDB group membership data and only require refresh for the smallest possible set of metrics. We also have been developing faster metrics delivery through write-through caching. This cache will provide the ability to present metrics directly before persisting them and will, as a side benefit, provide additional protection during an event.

Console

The AWS Console was impacted for some customers from 5:45am PDT to 7:10am PDT. Customers who were already logged into the Console would have continued to remain connected. Customers attempting to log into the Console during this period saw much higher latency in the login process. This was due to a very long timeout being set on an API call that relied on DynamoDB. The API call did not have to complete successfully to allow login to proceed but, with the long timeout, it blocked progress for tens of seconds while it waited to finish. It should have simply failed quickly and allowed progress on login to continue.

The timeout had already been changed in a version of the login code that has entered our test process. Unfortunately it wasn't yet rolled out when the event happened. We will make this change in the coming days. The reduced timeout will mitigate any impact of latency in the API call on the Console.

Final Words

We apologize for the impact to affected customers. While we are proud of the last three years of availability on DynamoDB (it's effectively been 100%), we know how critical this service is to customers, both because many use it for mission-critical operations and because AWS services also rely on it. For us, availability is the most important feature of DynamoDB, and we will do everything we can to learn from the event and to avoid a recurrence in the future.