Upcoming Webinar and Q&A: Early-Days Microservices Observability at Google (JUL 30); Sponsored by Lightstep

Amazon EC2 Outage Explained and Lessons Learned

LIKE 1

APR 29, 2011 • 8 MIN READ

by Abel Avram FOLLOW

Amazon has published a detailed report on the service failure plaguing one availability zone in the US East Region. The online media is full with analysis, commentaries and lessons to be learned from the event.

In a timeline of the Amazon EC2 outage, Eric Kidd outlined a series of major events related to the AWS service disruption as seen from the outside world. It all started on April 21st, 2011, at around 1 AM PDT when Heroku began investigating the large number of errors reported in service functionality. The outage lasted almost 4 days until April 24th at around 7:30 PM PDT when Amazon reported that all RDS databases were back online, and during this time the services did not run on ran intermittently for some customers.

What happened actually? Amazon wrote a detailed report on the service disruption, eight days after it started. It all began at 12:47 AM PDT on April 21st when “a network change was performed as part of our normal AWS scaling activities in a single Availability Zone in the US East Region” with the intent “to upgrade the capacity of the primary network”. This procedure required that the network traffic to be shifted to another router in the primary network, but instead was diverged to a router belonging to a lower level traffic network by mistake. This router could not handle the traffic, so “many EBS nodes in the affected Availability Zone were completely isolated from other EBS nodes in its cluster”, “leaving the affected nodes completely isolated from one another.”

The traffic was quickly restored to a proper router, but when that happened many EBS nodes tried to get available server space in order to mirror their data. The large number of nodes concurrently looking for free space quickly exhausted the EBS cluster’s available space, leaving about 13% of the volumes in the affected Availability Zone in a “stuck” state. Because of that, the cluster could not longer service new “Create Volume” API requests coming to the EBS control plane – responsible for managing the EBS clusters -, and

because these calls have a long time-out period, the result was thread starvation in the thread pool allocated to service external volume-related requests. Since the control plane is distributed across multiple availability zones in the same region, the thread starvation propagated to other zones. As a result, more nodes could not find free space to replicate their content and became “stuck” escalating the race condition during the first hours of the morning. The AWS team cut the communication between the affected EBS cluster and the control plane in order to protect other clusters in the region from becoming stuck and they deployed a fix that made the affected volumes less “hungry” for a mirroring space.

Amazon reports that the outage was contained in one availability zone and the degraded EBS cluster was stabilized by noon the same day, and customers could launch new EBS-backed EC2 instances, but they still encountered a large number of error, and the priority was now “bringing additional storage capacity online to allow the “stuck” volumes to find enough space to create new replicas.” The AWS team started slowly adding mirroring capacity to prevent a new race condition, and 97.8% of the volumes in the affected EBS cluster were fully replicated over the next 24 hours, leaving 2.2% volumes still stuck. The communication between the cluster and the control plane had to be restored, but a large number of operations had been accumulated in the backlog, so it took another 24 hours to unwind those operations. After another 24 hours the team managed to reestablish the proper functionality of the last 2.2% volumes that had been stuck, leaving “0.07% of the volumes in the affected Availability Zone [that] could not be restored for customers in a consistent state.” So, there was some data lost. The report does not specify how much 0.07% means in GBs. Some RDS services were impacted by the troubled EBS cluster, and they were restored throughout the weekend, leaving 0.4% of single-AZ database instances unrecovered because they depended on EBS data that was lost.

Amazon said they will incorporate lessons learned into their systems to prevent such happenings in the future, including adding more storage capacity:

> “
>
> We will be making a number of changes to prevent a cluster from getting into a re-mirroring storm in the future. With additional excess capacity, the degraded EBS cluster would have more quickly absorbed the large number of re-mirroring requests and avoided the re-mirroring storm. We now understand the amount of capacity needed for large recovery events and will be

> modifying our capacity planning and alarming so that we carry the additional safety capacity that is needed for large scale failures. We have already increased our capacity buffer significantly, and expect to have the requisite new capacity in place in a few weeks.

Amazon also said they will make it easier for customers to take advantage of different availability zones in order to prevent failure, but they also recommend using different regions simultaneously which is the best way to provide uninterrupted service because regions are completely separated and independent from each other.

Amazon will provide a "10 day credit equal to 100% of their usage of EBS Volumes, EC2 Instances and RDS database instances that were running in the affected Availability Zone" to all customers affected. The report ends with apologies to AWS customers.

The media has been inundated with reports related to the first cloud outage. Jason Bloomberg draw a number of lessons:

> "
>
> - **There is no such thing as 100% reliability.** In fact, there's nothing 100% about any of IT—no code is 100% bug free, no system is 100% crash-proof, and no security is 100% impenetrable. Just because Amazon came up snake eyes on this throw of the dice doesn't mean that public Clouds are any less reliable than they were before the crisis. Whether investing in the stock market or building a high availability IT infrastructure, the best way to lower risk is to diversify. You got eggs? The more baskets the better.
> - **This particular crisis is unlikely to happen ever again.** We can safely assume that Amazon has some wicked smart Cloud experts, and that they had already built a Cloud architecture that could withstand most challenges. Suffice it to say, therefore, that the latest crisis had an unusual and complex set of causes. It also goes without saying that those experts are working feverishly to root out those causes, so that this particular set of circumstances won't happen again.
> - **The unknown unknowns are by definition inherently unpredictable**. Even though the particular sequence of events that led to the current crisis is unlikely to happen again, the chance that other entirely unpredictable issues will arise in the future is relatively likely. But such issues might very well

> apply to private, hybrid, or community Clouds just as much as they might impact the public Cloud again. In other words, bailing on public Clouds to take refuge in the supposedly safer private Cloud arena is an exercise in futility.
>
> - **The most important lesson for Amazon to learn is more about *visibility* than *reliability*.** The weakest part of Amazon's cloud offerings is the lack of visibility they provide their customers. This "never mind the man behind the curtain" attitude is part of how Amazon supports the Cloud abstraction I discussed in the previous ZapFlash. But now it's working against them and their customers. For Amazon to build on its success, it must open the kimono a bit and provide its customers a level of management visibility into its internal infrastructure that it's been uncomfortable delivering to this point.

RightScale also had a post with an analysis of the Amazon EC2 outage, with lessons learned and some strong remarks on Amazon's failure to communicate properly during the service failure:

> "
>
> - Do not wait 40 minutes to post the first status message!
> - Do not talk about "a small percentage of instances/volumes/…", give actual percentages! Those of us with many servers/volumes care whether it's 1% or 25%, we will take different actions.
> - Do not talk about "the impacted availability zone" or "multiple availability zones", give each zone a name and refer to them by name (I know that zone 1a in each account refers to a different physical zone, so give each zone a second name so I can look it up).
> - Provide individualized status information: use email (or other means) to tell us what the status of our instances and volumes is. I don't mean things I can get myself like cpu load or such, but information like "the following volumes (by id) are currently recovering and should be available within the next hour, the following volumes will require manual intervention at a later time, …". That allows users to plan and choose where to put their efforts.
> - Make predictions! We saw volumes in the "impacted availability zone" getting taken out many hours after the initial event. I'm sure you knew that the problem was still spreading and could have warned

everyone. Something like: “we recommend you move all servers and volumes that are still operating in the impacted availability zone [sic] to a different zone or region as the problem is still spreading.”

- Provide an overview! Each status update should list which functions are still affected and which have been repaired, don’t make everyone scan back through the messages and try to infer what the status of each function is.
- Is it so hard to write a blog post with an apology and some background information, even if it’s preliminary? AWS tweeters that usually send multiple tweets per day remained silent. I’m sure there’s *something* to talk about 24 hours after the event! Don’t you want to tell everyone what they should be thinking instead of having them make it up???

High Scalability has compiled a long list of most important articles related the Amazon outage, what some companies experienced, links to related forums, who’s fault was – Amazon or customers’-, and lessons learned.

This unfortunate event affecting the largest cloud provider and its customers will certainly make people think twice before choosing to deploy to the cloud, and will send a strong message throughout the industry on how fragile our well engineered systems can be, showing one more time, if it was necessary, that the fight for reliability and resiliency in not over, and will never be.

This content is in the Agile topic

FOLLOW TOPIC

Related Topics:

ARCHITECTURE & DESIGN | DEVOPS | IAAS | AWS | AMAZON | AMAZON RDS | DATABASE | FAILURE | ARCHITECTURE | AGILE | INFRASTRUCTURE | RELATIONAL DATABASES | STORIES & CASE STUDIES | CLOUD COMPUTING

Tell us what you think

Please enter a subject

Message

Allowed html: a,b,br,blockquote,i,li,pre,u,ul,p

Email me replies to any of my messages in this thread

POST MESSAGE

Community comments

+ WATCH THREAD

- **Its the network stupid!**
  by Nati Shalom / May 08, 2011 05:32

Its the network stupid!

by Nati Shalom / May 08, 2011 05:32

> “
>
> It all began at 12:47 AM PDT on April 21st when “a network change was performed as part of our normal AWS scaling activities in a single Availability Zone in the US East Region” with the intent “to upgrade the capacity of the primary network”

One of the areas of vulnerability in most cloud computing infrastructure is the network. The network in the cloud is flat and shared across all tenants (of a specific Availability Zone in the case of Amazon and thus creates central dependency.
Failure in any central component leads to a ripple effect as in this specific case.

Amazon suggested lesson for reducing the changes for storage replica for allocating replica at the same time is therefore only a "bandage" to the real issue that is not even specific to Amazon. We need to be able to create multi-tenant network in the same way that we create multi-tenant machines and control the isolation at finer grain level than we do today.

On top of that we need to assume that in a cloud environment the chances for a machine or network failure is higher (at least at this point in time) mostly due to the lack of maturity in the industry of running this sort of workload - so application that runs on the cloud need to be designed to cope with this sort of failure at the application level as noted in Amazon note above.

> “
>
> Amazon also said they will make it easier for customers to take advantage of different availability zones in order to prevent failure, but they also recommend using different regions simultaneously which is the best way to provide uninterrupted service because regions are completely separated and

independent from each other.

LIKE REPLY BACK TO TOP

DEVELOPMENT

- New H.266 Video Coding Standard Claims to Be 50% More Efficient Than H.265
- Designing Composable Functional Libraries, Not Just for Data Visualization
- PHP 7 — New Features for Types

ARCHITECTURE & DESIGN

- Rancher on Hybrid Cloud, Kubernetes at the Edge, and Open Standards
- QCon San Francisco Announces 2020 Tracks
- Responsible Microservices

CULTURE & METHODS

- What to Build First: Goal-Oriented MVP
- Optimizing for Speed with Continuous Organizational Transformation
- Agile Initiative Planning with Roadmaps

AI, ML & DATA ENGINEERING

- Everything You Wanted to Know about Apache Kafka but You Were Too Afraid to Ask!
- Microsoft's ZeRO-2 Speeds up AI Training 10x
- Databricks Contributes MLflow Machine Learning Platform to The Linux Foundation

DEVOPS

- Google Donates Trademarks to New Foundation
- Distributed Tracing in the Wild
- Chaos and Resilience Engineering: Mental Models, Tools and Experiments

The InfoQ Newsletter

A round-up of last week's content on InfoQ sent out every Tuesday. Join a community of over 250,000 senior developers. View an example

- Get a quick overview of content published on a variety of innovator and early adopter technologies
- Learn what you don't know that you don't know
- Stay up to date with the latest information from the topics you are interested in

Enter your e-mail address | Select a country | SUBSCRIBE

We protect your privacy.

Media Kit

General Feedback
feedback@infoq.com

Advertising
sales@infoq.com

Editorial
editors@infoq.com
Marketing
marketing@infoq.com

In

Pr