

Cloud Management

Actionable advice to guide your enterprise cloud strategy

Amazon EC2 Outage: Summary and Lessons Learned

📅 April 25, 2011 👤 Flexera 🐦 [@flexera](#) ✉ [Subscribe](#)

You've reached an archived blog post that may be out of date. Please visit the [blog homepage](#) for the most current posts.

Last Thursday's [Amazon EC2](#) outage was the worst in cloud computing's history. It made the front page of many news pages, including the New York Times, probably because many people were shocked by how many web sites and services rely on EC2. Seeing so much affected was a very graphical illustration of how pervasive cloud computing has become.

I will try to summarize what happened, what worked and didn't work, and what to learn from it. I'll do my best to add signal to all the noise out there, in that respect I liked [a tweet by Beaker](#) (Christofer Hoff): "Happy with my decision NOT to have written a blog about the misfortune of AWS, stating nothing but the obvious & sounding like a muppet."

Executive summary

- The Amazon cloud proved itself in that sufficient resources were available world-wide such that many well-prepared users could continue operating with relatively little downtime. But because Amazon's reliability has been incredible, many users were not well-prepared leading to widespread outages. Additionally, some users got caught by unforeseen failure modes rendering their failure plans ineffective.
- Some ripple effects within EC2 and in particular EBS caused by the initial failure should not have happened. There's important work Amazon needs to do to prevent such occurrences.
- Amazon's communication, while better than during previous outages still earns an F. This is probably the #1 threat to AWS's business.
- The cloud architecture provides ample opportunities to design systems to withstand failures. The material cost of such designs is a fraction of what comparative measures would cost using traditional hosting means. However, designing, building, and testing everything is not cheap. Many of our customers who used our [best practices](#) fared well (I'm not claiming we're perfect or that everything is automatic!) and we got numerous calls from other companies that were wholly unprepared.
- Overall this is just one of many bumps in the cloud computing road. It reminds us that this is still "day one" of the cloud and that we all have much to learn about building and operating robust systems on a large scale. We are

receiving a stream of calls from EC2 users that realize they need help in setting up a more robust architecture for their systems.

Outage analysis

At the time of writing Amazon has not yet posted a root cause analysis. I will update this section when they do. Until then, have to make some educated guesses.

We got the first alerts at 1:01am on Thursday, the proverbial Christmas lights lit up indicating I/O issues on a large number of our servers. We started failing servers over and opened a ticket with Amazon. They finally posted a status message at 1:41am containing no useful details, sadly this is a typical sequence of events.

It appears that a major network failure was the initial cause of problems but that the real damage happened when EBS (Elastic Block Store) volume replication was disrupted. We did some extrapolations and concluded that there must have been on the order of 500k EBS storage volumes in the affected availability zone. It appears that a significant fraction of the volumes concluded that the replication mirroring was out-of-sync and started re-replicating causing further havoc, including an overload of the EBS control plane. It is also possible that the EBS replication problem was the root cause and that the network issues were a consequence, hopefully Amazon's root cause analysis will shed light on this.

The biggest problem, from my point of view, was that more than one availability zone was affected. We didn't see servers or volumes fail in other zones but we were unable to create fresh volumes elsewhere, which of course makes it difficult to move services. This is "not supposed to happen" and is an indication that the EBS control plane has dependencies across zones. Amazon did manage to contain the problem to one zone approx 3 hours after the onset.

After Amazon managed to contain the problems to one zone, it took a very long time to get the EBS machinery under control and to recover all the volumes. Given the extrapolated number of volumes it would not be surprising that an event of this scale exceeded the design parameters and was never tested (or able to be tested). I'm not sure there is any system of comparable scale in operation anywhere.

I do want to state that while "something large" clearly failed, namely the EBS system as a whole, the real big failure is that multiple availability zones were affected for ~3 hours. I also want to mention two important things that didn't fail: we didn't see capacity constraints in relaunching servers in other zones after the initial cross-zone issues and we didn't see other regions affected at all. This is clearly good news!

Amazon communication failure

In my opinion the biggest failure in this event was Amazon's communication, or rather lack thereof. The status updates were far too vague to be of much use and there was no background information whatsoever. Neither the official AWS blog nor Werner Vogels' blog had any post whatsoever 4 days after the outage! Here is a list of improvements for Amazon:

- 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???

Coverage from around the web

Since Amazon did not communicate much of substance beyond the rather sparse and obscure status updates everyone else was left to speculate. Most of the blog posts or news articles contained little information. Here's a list of blog posts that I found interesting:

- [Amazon outage and the auto-immune vulnerabilities of resiliency](#) by Lydia Leong at Gartner. An early post during the outage that has a good overall analysis.
- [Amazon.com's real problem isn't the outage, it's the communication](#) by Keith Smith from BigDoor.
- [AWS is down: Why the sky is falling](#) by Justin Santa Barbara. One of the early posts with technical information.
- [Standing on the shoulders of giants and stumbling with them – the Amazon AWS outage's “pain” statistics](#) by PagerDuty (a service we happily use) with nice stats about alerts going out during the outage.
- [Cloud crash has a silver lining](#) by Leon Katsnelson from IBM about DB2 and replication.
- [nostromo comment on news.ycombinator.com](#) (look for nostromo's comment if it's not at the top anymore), a brief description of the pain with RDS.
- [Today's EC2 / EBS Outage: Lessons learned](#) by Stephen Nelson-Smith. One of the first good lessons learned posts I saw.
- [Why Twilio Wasn't Affected by Today's AWS Issues](#) has some interesting recommendations on how to architect for failure.
- [The AWS Outage: The Cloud's Shining Moment](#) by George Reese of Enstratus has a very nice analysis of what designing for failure means and how it contrasts with more traditional approaches.
- [Amazon's Trouble Raises Cloud Computing Doubts](#) on the front-page of the New York Times business section.
- [Working around the EC2 outage](#) by Jérôme Petazzoni of dotcloud with an interesting account of the issues they faced.

Lessons learned

Our services team handled 4x the incident volume last Thursday compared to a normal Thursday. A large number of callers needed help in assessing the situation or in bringing their servers back up. A typical request was: “It looks like my db server is down due to the outage, can you help confirm and assist with a migration?” Unfortunately we also heard from a good number of users who were using a single availability zone or didn't set up redundancy properly. Hindsight is always 20-20.

A clear lesson for everyone is obviously that backup and replication have to be taken seriously (duh). In EC2 this means live replication across multiple availability zones and backups to S3 (and ideally elsewhere also). It has also become clear that a minimum of replicas must be running and a certain degree of over-provisioning is necessary to handle the load spike after a massive failure. Adrian Cockcroft from Netflix summarized their strategy in [a tweet](#) a while ago: “Deploy in three AZ with no extra instances – target autoscale 30-60% util. You have 50% headroom for load spikes. Lose an AZ -> 90% util.” (Also see the [discussion around the tweet](#).) Users that relied on launching fresh servers or on creating fresh volumes from snapshots were not able to do so for several hours. The only previous event that I remember where multiple availability zones were affected was the [July 20th 2008 S3 outage](#) that took down S3 in the US and EU (multiple *regions*!).

A number of blogs mention NoSQL databases as a solution to the replication and failure difficulties with traditional relational databases. While we've started to use [Cassandra](#) ourselves it has become pretty clear to me that this is not a silver bullet by a long shot. When a single node fails the built-in replication and recovery functions well, although the extra load on remaining nodes is high when the failing node is repaired and resynchronizes. But when large numbers of nodes in the cluster lock-up one-by-one over the course of an hour, I'd be hesitant to make a prediction about the outcome both in terms of the cluster's availability and its consistency. We have two applications that make very different use of Cassandra and the behavior of the database is very different in both cases. My conclusion from what I have observed thus far is that clusters of replicated eventually-consistent NoSQL stores have pretty complex dynamics that can easily lead to unpleasant surprises. Sometimes it's nice to have a comparatively simple MySQL master-slave set-up that experiences some downtime during the fail-over but acts very predictably.

I can't help but feel uncomfortable about the performance of Amazon's RDS "database-as-a-service" in that some databases that were replicated across multiple availability zones did not fail-over properly. It evidently took more than 12 hours to recover a number of the multi-az databases. The obvious failure here is compounded by the fact that Amazon has made it difficult for users to backup their databases outside of RDS, leaving them no choice but to wait for someone at Amazon to work on their database. This lock-in is one reason many of our customers prefer to use our MySQL master-slave setup or to architect their own.

The biggest lesson we learned about operating RightScale itself is that we have to continue pushing hard on reducing the load on our central MySQL database and distributing our service. The database has grown too big and failover consequently takes too long because it takes forever to load the working set (over 30GB) into memory. We have some short-term measures we will be implementing to reduce the failover time, but more is needed. We also need to provide our users a choice of RightScale systems located in [different regions and clouds](#): users operating primarily out of one region need to be able to use RightScale in an independent region or cloud. Ironically the first thing every public cloud operator and every company with a private cloud asks us is whether we can run RightScale inside their cloud: that seems pretty misguided to me!

We also were confused by Amazon's status messages. In hindsight we should have intentionally failed-over our master database which was operating in the "impacted availability zone" early on at a time where we could minimize downtime. We were lucky that it didn't get affected until about 12 hours after the start of the outage but we didn't connect one and one. A clear message from Amazon that more and more volumes were continuing to fail in the zone would have been really helpful.

What's next?

With Amazon's overall stellar operating reliability it is easy to become complacent. This outage was a wake-up call for many of us. What remains to be seen is whether Amazon decides to take a lead and provide more granular descriptions of failure modes and recommended actions or whether they will leave it to everyone else to guess and figure it out. I see this as being one of the main long-term problems of cloud computing, namely that it is extremely difficult for users to list the possible failure modes and even more difficult to actually test any of them.

In the big picture I find [Lew Moorman's](#) analogy in the [NYT article](#) very appropriate: "The Amazon interruption was the computing equivalent of an airplane crash. It is a major episode with widespread damage. But airline travel is still safer than traveling in a car — analogous to cloud computing being safer than data centers run by individual companies. Every day, inside companies all over the world, there are technology outages, each episode is smaller, but they add up to far more lost time, money and business." Most of the articles that predict a run away from cloud computing fail to explain where to run to. Unless you can hire superman to run your private datacenters my experience tells me that you'll be worse off.



Tags: [Cloud Industry Insights](#)

← PREVIOUS POST

[Launch VMware's Cloud Foundry PaaS Using RightScale](#)

APRIL 12, 2011

NEXT POST →

[Commercial Support for OpenStack on the Horizon](#)

MAY 25, 2011

Recent Posts Popular

- 1 [Scale with the Cloud to Meet Increased Business Demand](#)
- 2 [Benefits of the Cloud for Business: Navigating Your Cloud Journey](#)
- 3 [Cloud migration best practices: Don't let efforts get stranded while working remotely](#)
- 4 [Driving Digital Transformation with Flexera: A Leader in the 2020 Gartner Magic Quadrant for Cloud Management Platforms—Again!](#)
- 5 [5 Cloud Cost Savings Strategies for Now \(and in the Future\)](#)

Topics

[Application Readiness](#)

[Cloud Management](#)

[Data Platform](#)

[Software License Optimization](#)

[Software Vulnerability Management](#)

[IT Industry Trends](#)



Tags

[Apache Struts](#) [Application Compatibility](#) [Application Readiness](#) [Application Virtualization](#) [Archive](#) [Cloud Computing](#) [Cloud Cost](#) [Cloud Cost Analysis](#) [Cloud Industry Insights](#) [Cloud Management](#) [Best Practices](#) [CMDB](#) [Data](#) [Data Platform](#) [Data Quality](#)

News [IBM software licensing](#) [IT Asset Management](#) [IT Industry Trends](#) [Microsoft software licensing](#) [Mobile Applications](#) [Oracle software licensing](#)

SaaS [SaaS IT Spend](#) [SAP software licensing](#) [Security](#) [Software Asset Management](#) [Software budget planning](#) [Software license agreements](#) [Software license compliance](#) [Software License Optimization](#) [Software spend management](#) [Software Vulnerabilities](#) [Technology Spend](#)

[Technopedia](#) [Vulnerability](#) [Windows 7](#) [Windows 8](#) [Windows 10](#)


About the Cloud Management blog

Flexera's Cloud Management Blog provides actionable advice to guide your enterprise cloud strategy and keep you informed on the latest industry trends

Updates in your inbox

Give us your email and we'll keep you in the loop

Get In Touch

 +1 (800) 374-4353

[Contact Us](#)

[Careers](#)

[Support](#)

Stay Connected



Other Sites

[Flexera.com](#)

[Customer Community](#)