Summary of the Amazon EC2, Amazon EBS, and Amazon RDS Service Event in the EU West Region

We would like to share more details with our customers about the events that occurred with Amazon Elastic Compute Cloud ("EC2"), Amazon Elastic Block Store ("EBS"), and Amazon Relational Database Service ("RDS") earlier this week, and what we are doing to prevent these sorts of issues from happening again. The service disruption primarily affected EC2 instances, RDS instances, and a subset of EBS volumes in a single Availability Zone in the EU West Region.

Primary Outage

The service disruption began at 10:41 AM PDT on August 7th when our utility provider suffered a failure of a 110kV 10 megawatt transformer. This failure resulted in a total loss of electricity supply to all of their customers connected to this transformer, including a significant portion of the affected AWS Availability Zone. The initial fault diagnosis from our utility provider indicated that a lightning strike caused the transformer to fail. The utility provider now believes it was not a lightning strike, and is continuing to investigate root cause.

Normally, when utility power fails, electrical load is seamlessly picked up by backup generators. Programmable Logic Controllers (PLCs) assure that the electrical phase is synchronized between generators before their power is brought online. In this case, one of the PLCs did not complete the connection of a portion of the generators to bring them online. We currently believe (supported by all observations of the state and behavior of this PLC) that a large ground fault detected by the PLC caused it to fail to complete its task. We are working with our supplier and performing further analysis of the device involved to confirm. With no utility power, and backup generators for a large portion of this Availability Zone disabled, there was insufficient power for all of the servers in the Availability Zone to continue operating. Uninterruptable Power Supplies (UPSs) that provide a short period of battery power quickly drained and we lost power to almost all of the EC2 instances and 58% of the EBS volumes in that Availability Zone. We also lost power to the EC2 networking gear that connects this Availability Zone to the Internet and connects this Availability Zone to the other Availability Zones in the Region. This caused connectivity issues to the affected Availability Zone and resulted in API errors when customers targeted API requests (RunInstance, CreateVolume, etc.) to the impacted Availability Zone.

At 11:05 AM PDT, we were seeing launch delays and API errors in all EU West Availability Zones. There were two primary factors that contributed to this. First, our EC2 management service (which handles API requests to RunInstance, CreateVolume, etc.), has servers in each Availability Zone. The management servers which receive requests continued to route requests to management servers in the affected Availability Zone. Because the management servers in the affected Availability Zone were inaccessible, requests routed to those servers failed. Second, the EC2 management servers receiving requests were continuing to accept RunInstances requests targeted at the impacted Availability Zone. Rather than failing these requests immediately, they were queued and our management servers attempted to process them. Fairly quickly, a large number of these requests began to queue up and we overloaded the management servers receiving requests, which were waiting for these queued requests to complete. The combination of these two factors caused long delays in launching instances and higher error rates for the EU West EC2 APIs. At 12:00 PM PDT, when we disabled EC2 launches in the affected Availability Zone and removed the failed management servers from service, EC2 API launch times for other Availability Zones recovered.

At 11:54 AM PDT, we had been able to bring some of the backup generators online by manually phase-synchronizing the power sources. This restored power to many of the EC2 instances and EBS volumes, but a majority of the networking gear was still without power, so these restored instances were still inaccessible. By 1:49 PM PDT, power had been restored to enough of our network devices that we were able to re-establish connectivity to the Availability Zone. Many of the instances and volumes in the Availability Zone became accessible at this time.

Recovering EBS in the Affected Availability Zone

To understand why restoration of EBS took longer, it's helpful to understand a little about the EBS architecture. EBS volume data is replicated across a set of EBS nodes for durability and availability. These nodes serve read and write requests to EC2 instances. If one node loses connectivity to another node that it is replicating data to, it must find and replicate its data to a new node (this is called re-mirroring)-- and it will block writes until it has found that new node. From the perspective of an EC2 instance trying to do I/O on an EBS volume that is blocking writes, the volume will appear "stuck."

On Sunday, when a large portion of the EBS servers lost power and shut down, EBS volumes in the affected Availability Zone entered one of three states: (1) online – none of the nodes holding a volume's data lost power, (2) re-mirroring – a subset of the nodes storing the volume were offline due to power loss and the remaining nodes were re-replicating their data, and (3) offline – all nodes lost power.

In the first case, EBS volumes continued to function normally.

In the second case, the majority of nodes were able to leverage the significant amount of spare capacity in the affected Availability Zone, successfully re-mirror, and enable the volume to recover. But, because we had such an unusually large number of EBS volumes lose power, the spare capacity we had on hand to support re-mirroring wasn't enough. We ran out of spare capacity before all of the volumes were able to successfully re-mirror. As a result, a number of customers' volumes became "stuck" as they attempted to write to their volume, but their volume had not yet found a new node to receive a replica. In order to get the "stuck" volumes back online, we had to add more capacity. We brought in additional labor to get more onsite capacity online and trucked in servers from another Availability Zone in the Region. There were delays as this was nighttime in Dublin and the logistics of trucking required mobilizing transportation some distance from the datacenter. Once the additional capacity was available, we were able to recover the remaining volumes waiting for space to complete a successful re-mirror.

In the third case, when an EC2 instance and all nodes containing EBS volume replicas concurrently lose power, we cannot verify that all of the writes to all of the nodes are completely consistent. If we cannot confirm that all writes have been persisted to disk, then we cautiously assume that the volume is in an inconsistent state (even though in many cases the volume is actually consistent). Bringing a volume back in an inconsistent state without the customer being aware could cause undetectable, latent data corruption issues which could trigger a serious impact later. For the volumes we assumed were inconsistent, we produced a recovery snapshot to enable customers to create a new volume and check its consistency before trying to use it. The process of producing recovery snapshots was time-consuming because we had to first copy all of the data from each node to Amazon Simple Storage Service (Amazon S3), process that data to turn it into the snapshot storage format, and re-copy the data to make it accessible from a customer's account. Many of the volumes contained a lot of data (EBS volumes can hold as much as 1 TB per volume). By 6:04 AM PDT on August 9th, we had delivered approximately 38% of the recovery snapshots for these potentially inconsistent volumes to customers. By 2:37 AM PDT on August 10th, 85% of the recovery snapshots had been delivered. By 8:25 PM PDT on August 10th, we were 98% complete, with the remaining few snapshots requiring manual attention.

Impact on Amazon RDS

RDS Instances were also affected by the disruption. RDS database instances utilize EBS volumes for database and log storage. As a result, the power outage in the affected Availability Zone had significant impact on RDS as well. Single Availability Zone ("Single-AZ") database instances in the affected Availability Zone were almost all initially unavailable. They recovered when their corresponding EBS volumes were restored or their databases were restored to new volumes. All Amazon RDS Single-AZ database instances have automated backups turned on by default. The majority of customers whose databases did not recover when the first tranche of EBS volumes came back online, or could not be recovered due to inconsistency of their volumes, used this backup functionality to initiate Point-in-Time-Restore operations, per our Service Health Dashboard instructions. Customers with automated backups turned off, could not initiate Point-in-Time-Restores.

In addition to impacting Single-AZ database instances, the severity of the event and nature of failure also caused a portion of Multiple Availability Zone ("Multi-AZ") database instances to be impacted. Rapid failover occurred for the vast majority of Multi-AZ databases, and all affected Multi-AZ databases in the EU-West Region failed over without data loss. However, a portion of Multi-AZ database instances experienced prolonged failover times.

To understand why some Multi-AZ database instances did not promptly failover, it is useful to understand how Multi-AZ databases work. RDS Multi-AZ database instances consist of a "primary" database instance and a synchronously replicated "secondary" database instance in another Availability Zone. When the system detects that a primary database instance might be failing, upon verification via a health check that the primary is no longer accepting traffic, the secondary is promoted to primary. This verification is important to avoid a "split brain" situation, one where both the primary and the secondary database instances are accepting writes and some writes exist on one database while some exist on another. Similarly, when the system detects that a secondary database instance is failing, upon performing the health check and verifying that the secondary hasn't assumed the role of primary, the primary will allow itself to continue as a Single-AZ database instance until a new secondary is established and connected to the primary, bringing the pair back into Multi-AZ status.

During the event, there were failures of Multi-AZ primary database instances in the affected Availability Zone. However, for a portion of these Multi-AZ primary-secondary pairs, a DNS connectivity issue related to the power loss prevented the health check from finding the IP address it needed to contact and kept the secondary from immediately assuming the role of the primary. DNS connectivity was restored within 4 minutes, and the majority of Multi-AZ deployments then completed failover within an additional 10 minutes. However, the DNS connectivity issues triggered a software bug that caused failover times to the secondary database instance to extend significantly for a small subset of Multi-AZ deployments.

This DNS connectivity issue also triggered extended failover times for a small portion of Multi-AZ deployments with secondary replicas in the affected Availability Zone. For these deployments, DNS connectivity prevented the primary replicas from confirming their secondary replica's status. In the rare case where the status of the secondary cannot be determined, the primary does not make itself a Single AZ-mode database instance and instead immediately involves the RDS team. This cautious approach is taken to help prevent the "split brain" scenario described above. Instead, an RDS engineer makes the decision to either promote the secondary to primary (if the old primary is not functioning), or to move the primary to Single-AZ mode (if the secondary is not functioning). This approach minimizes the risk of data loss in edge cases, but extends the period of time the Multi-AZ instance is unavailable.

EBS Software Bug Impacting Snapshots

Separately, and independent from issues emanating from the power disruption, we discovered an error in the EBS software that cleans up unused storage for snapshots after customers have deleted an EBS snapshot. An EBS snapshot contains a set of pointers to blocks of data, including the blocks shared between multiple snapshots. Each time a new snapshot is taken of an EBS volume, only the data that has been modified since the last snapshot is pushed to S3. When a snapshot is deleted, only the blocks not referenced by later snapshots should be deleted. A cleanup process runs periodically to identify all blocks that are no longer included in any snapshots. This snapshot cleanup identification process builds a list of the blocks included in the deleted customer snapshots, a list of blocks referenced by active EBS volumes, and a list of blocks referenced by other snapshots. Blocks that are referenced by active volumes or snapshots are removed from the list of blocks to cleanup.

The resulting cleanup list is saved, but not acted upon. At least one week passes from the time the snapshot cleanup identification process runs before any blocks it has flagged for deletion are allowed to be removed. Each day, it updates the lists of blocks to delete, blocks referenced by active volumes, and blocks referenced by other snapshots. It also compares its updated lists to the prior day's and if any block eligible for deletion the day before now shows up in the most recent list of blocks referenced by active EBS volumes or snapshots, the process flags those blocks for analysis. Typically, there are very few, if any, items that get flagged for analysis. But, this part of the process was introduced to protect against system or software errors that could result in blocks falsely flagged for deletion. Actual deletion is executed by an engineer who first, before running the actual deletion process, evaluates the blocks flagged for analysis and verifies that there are no blocks in the list scheduled to be deleted that have been flagged for analysis. The engineer must present their verification step to another engineer who approves the deletion.

In one of the days leading up to the Friday, August 5th deletion run, there was a hardware failure that the snapshot cleanup identification software did not correctly detect and handle. The result was that the list of snapshot references used as input to the cleanup process was incomplete. Because the list of snapshot references was incomplete, the snapshot cleanup identification process incorrectly believed a number of blocks were no longer referenced and had flagged those blocks for deletion even though they were still referenced by customer snapshots. A subsequent run of the snapshot cleanup identification process detected the error and flagged blocks for further analysis that had been incorrectly scheduled for deletion. On August 5th, the engineer running the snapshot deletion process checked the blocks flagged for analysis before running the actual deletion process in the EU West Region. The human checks in this process failed to detect the error and the deletion process was executed. On Friday evening, an error accessing one of the affected snapshots triggered us to investigate.

By Sunday morning, August 7th, we had completed the work to fully understand root cause, prevent the problem from recurring, and build a tool that could create recovery snapshots for affected snapshots. We then started to do the work necessary to map these affected snapshots to customers and build the recovery snapshots, with the aim to communicate this information to customers by Sunday night. However, before we got very far in this endeavor, the power event began. We had to temporarily stop work on the snapshot issue to respond to the power event. Once we'd been able to restore the majority of the EBS volumes affected by the power event, we returned to working on the snapshot issue in parallel with restoring the remainder of the EBS volumes that were recovering from the power event. By 4:19 PM PDT on August 8th, we'd completed creating recovery snapshots for all affected snapshots, delivered them to customers' accounts, and communicated about the issue on the Service Health Dashboard.

Actions to Prevent Recurrence

There are several actions we intend to take to protect against a similar occurrence. The following are some of the key ones.

To further prevent the loss of power, we will add redundancy and more isolation for our PLCs so they are insulated from other failures. Specifically, in addition to correcting the isolation of the primary PLC, a cold, environmentally isolated backup PLC is being worked with our vendors. We will deploy this as rapidly as possible.

For EC2, we are going to address the resource saturation that affected API calls at the beginning of the disruption. We will implement better load balancing to quickly take failed API management service hosts out of production. Over the last few months, we have been developing further isolation of EC2 control plane components (i.e. the APIs) to eliminate possible latency or failure in one Availability Zone from impacting our ability to process calls to other Availability Zones. While some of those mitigations significantly reduced the impact of this disruption and helped us recover the APIs quickly, we realize how important those APIs are to customers, especially during an event. It will take us several more months to complete some of the changes we're making, and we will test and roll out these changes carefully. At the time of the disruption, customers who had EC2 instances and EBS volumes independently operating in multiple EU West Region Availability Zones did not experience service interruption. We will continue to create additional capabilities that make it easier to develop and deploy applications in multiple Availability Zones.

For EBS, our primary action will be to drastically reduce the long recovery time required to recover stuck or inconsistent EBS volumes when there is a substantial infrastructure disruption. While some volumes were recoverable immediately once we had power back, there was an extended period of time for many volumes to recover due to the need to create EBS snapshots within S3. As we described above, this long period of delay was caused by the time required to move a very large amount of data into S3 and then transfer that data to EBS recovery snapshots. To significantly reduce the time required to restore these volumes, we will create the capability to recover volumes directly on the EBS servers upon restoration of power, without having to move the data off of those servers. This will require providing a way for customers to know that a volume has been shut down and restored, but will avoid the need for restoration via snapshot. This will also substantially diminish any risk associated with lack of capacity, regardless of how many volumes fail.

We've made changes to our deletion process to prevent recurrence of the EBS software bug impacting snapshots. We are instrumenting an alarm that will alert us if there are any unusual situations discovered by the snapshot cleanup identification process, including blocks falsely flagged as being unreferenced. We're also adding another holding state for blocks flagged for deletion where they are logically unavailable but retrievable for an additional, longer period of time. This will provide additional time to discover and correct any problem without loss of data.

We learned a number of lessons from this event that we will use to continually improve the reliability of RDS Multi-AZ deployments. First, we will implement changes to our health checks to avoid customer impact in the event of a unique DNS connectivity issue like we experienced here. Second, we will promptly fix the software bug that extended failover times for a portion of Multi-AZ customers with primaries in the affected Availability Zone. Third, we will implement an improved handling of the edge case where either primary or secondary is down and the health check cannot complete. In such a case, the successfully running member of the Multi-AZ pair will initiate connection retries to confirm it is no longer in a "split brain" mode, such that involving an engineer might not be necessary.

Improving Communication

Communication in situations like this is difficult. Customers are understandably anxious about the timing for recovery and what they should do in the interim. We always prioritize getting affected customers back to health as soon as possible, and that was our top priority in this event, too. But, we know how important it is to communicate on the Service Health Dashboard and AWS Support mechanisms. Based on prior customer feedback, we communicated more frequently during this event on our Service Health Dashboard than we had in other prior events, we had evangelists tweet links to key early dashboard updates, we staffed up our AWS Support team to handle much higher forum and Premium Support contacts, and we tried to give an approximate time-frame early on for when the people with extra-long delays could expect to start seeing recovery. Still, we know what matters most to customers in circumstances like this is knowing the status of their resources, when the impacted ones will be healthy, and what they should do now. While we provided best estimates for the long-lead recovery snapshots, we truly didn't know how long that process was going to take or we would have shared it. For those waiting for recovery snapshots, we tried to communicate what was possible. If customers were architected to operate across multiple Availability Zones, they could flip over to and/or deploy resources in other Availability Zones. If customers were architected such that spinning up new instances or volumes in the same Availability Zone worked, they could do that. But, for those single Availability Zone customers who needed a specific EBS volume to recover, and whose EBS volume was in the group waiting for recovery snapshots, there were really no short term actions possible.

There are several places we can improve on the communication front. First, we can accelerate the pace with which we staff up our Support team to be even more responsive in the early hours of an event. Second, we will do a better job of making it easier for customers (and AWS) to tell if their resources have been impacted. This will give customers (and AWS) important shared telemetry on what's happening to specific resources in the heat of the moment. We've been hard at work on developing tools to allow you to see via the APIs if your instances/volumes are impaired, and hope to have this to customers in the next few months. Third, as we were sending customers recovery snapshots, we could have been clearer and more instructive on how to run the recovery tools, and provided better detail on the recovery actions customers could have taken. We sometimes assume a certain familiarity with these tools that we should not.

Service Credit for Affected Customers

For customers with an attached EBS volume or a running RDS database instance in the affected Availability Zone in the EU West Region at the time of the disruption, regardless of whether their resources and application were impacted or not, we are going to 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 in the EU West region. Additionally, any customers impacted by the EBS software bug that accidentally deleted blocks in their snapshots will receive a 30 day credit for 100% of their EBS usage in the entire EU West Region (inclusive of snapshot storage and requests as well as volume storage and I/O). These customers will also have access to our Premium Support Engineers (via the AWS Support Center) if these customers need any additional technical assistance in recovering from this issue.

These customers will not have to do anything in order to receive the credits, as they will be automatically applied to customers' next AWS bill. The credits can also be viewed as they become available over the next few weeks by logging into the AWS Account Activity page.

Summary

Last, but certainly not least, we want to apologize. We know how critical our services are to our customers' businesses. We will do everything we can to learn from this event and use it to drive improvement across our services. As with any significant operational issue, we will spend many hours over the coming days and weeks improving our understanding of the details of the various parts of this event and determining how to make changes to improve our services and processes.

Sincerely,
The AWS Team