The Best Plan to Avoid the Worst Outcome

To establish an effective AWS backup and disaster recovery plan, it's essential to account for how DR in the cloud differs from an on-premises infrastructure. Here's what you'll need to know about secondary sites, availability zones and load balancing.
Plan for the Worst

**IT professionals rightfully** take pride in designing and maintaining systems that reliably perform the tasks assigned them. Yet, there’s the awareness that at any time something could—and probably will—go awry. It’s an accepted fact of life.

What matters then is how well an IT team can react and recover. Doing so in AWS is no less important, but the best practices and responses are different than they would be in other environments. This handbook looks specifically at how to be sure AWS workloads are available when you need them and, in the event of disaster, are easily recoverable.

Tech advisor Ofir Nachmani details the essential elements in an AWS disaster recovery strategy. He explains the key differences between the hot standby and pilot light approaches to DR. And he discusses how an organization will want to look at the building blocks for DR and backup, incorporating things such as resource tagging, availability zones and snapshots.

In his article, cloud expert Dan Sullivan writes about the critical role load balancing can play in ensuring application availability—even when a server fails.

IT writer Alan Earls, meanwhile, describes how testing your systems’ reliability is just as crucial in the cloud as it is with traditional data centers. The good news is that backup and DR testing is much easier in AWS than in those on-premises environments.

**Phil Sweeney**
Senior Managing Editor
Data Center and Virtualization Group
TechTarget
The Building Blocks for AWS Disaster Recovery

**Traditional backup methods** are carried out over a local area network by agents installed on machines. If components fail, replacing them entails time-consuming tasks that consequently create complicated backup environments. These difficult-to-manage environments require plentiful resources and technologies to operate, including virtual tape libraries and methods such as data deduplication to handle the ever-growing workloads.

These complications can be untangled now that businesses function more in the cloud. In the world of AWS, for instance, organizations can simply provision resources for backup purposes at a specific time or in a specific place, using the cloud giant’s flexible infrastructure and global presence. With the AWS console or command-line interface tools, IT teams can easily automate snapshots and recovery for their entire application stack.

Let’s examine two different types of use cases: on-premises deployments that use AWS as a secondary site for purposes of disaster recovery (DR) and backup, and applications that are run and backed up in AWS. We’ll discuss cloud methods and AWS building blocks that can help streamline your cloud-based backup and DR.

**UNDERSTAND YOUR BUSINESS NEEDS**

Outages are an inevitable occurrence. Whether your environment is in the public cloud, on premises, or both, it is a basic assumption that something at some point will go wrong. To prepare for such events, you’ll need to have an effective DR strategy in place. While the basics of what is needed from backup and how DR should be carried out remain the same, the methods and tools involved have changed.

In both of the deployment cases discussed here, it is important to know how to synchronize backup and disaster recovery with your
business requirements, such as recovery time objective (RTO) and recovery point objective (RPO).

You can plan your implementation only after you understand your requirements, taking potential costs into account. For example, in a mission-critical service, deploying an active DR site is ideal. Non-critical environments, such as ones used for development and testing, may require only a lightweight backup that can be achieved by spinning up machines on demand.

**TWO TYPES OF SECONDARY FAILOVER**

Whether you use AWS as a secondary DR site for your on-premises environment or are running all of your operations on AWS, there are two types of configurations you can implement: hot standby or pilot light.

Hot standby is an AWS secondary site that is continuously up and running. It is kept updated and ready for user requests in case the main environment fails. The two sites are automatically synchronized. With hot standby, an entire configuration is replicated “as is” in the cloud, across AWS availability zones (AZs) and regions. By enabling seamless failover, the user experience is maintained.

While hot standby is the ultimate answer, a constantly running replica means that you will nearly double the costs of running your application in the cloud.

In the pilot light scenario, a secondary site holds the minimum amount of resources in an ideal state, effectively serving as a spare tire for your service. The infrastructural elements of your application are saved as images and snapshots, and are ready to spin off another environment if disaster strikes. For example, instances can be in a stopped state or spun from ready-made Amazon Machine Images (AMIs).

Pilot light results in minimal costs. Since nothing is really running in your secondary stack, you’re paying to cover only its maintenance and storage.

Typically, AWS users choose a combination of the two approaches. For example, you may want to autoscale your Web servers to keep up with demand, and keep AMI replicas in a separate AZ and active database instances in the
same zone. Ultimately, your decision will be based on how much you are willing to invest to fulfill your business needs and compliance requirements.

AWS BUILDING BLOCKS AND CAPABILITIES
AWS provides five key features to help automate backup and DR processes. Let’s look at how they can help.

1. EBS Volume Snapshots. Amazon Elastic Block Store (EBS) volumes can be stored in Amazon Simple Storage Service (S3) by taking snapshots. EBS snapshots are considered to be the base object when it comes to keeping a backup on AWS. Snapshot backups are incremental, meaning they save only blocks that have been altered since the last backup.

2. Back up and Recover an Elastic Cloud Compute (EC2) Instance. AWS provides the choice to keep an AMI of your instance or to take a snapshot of its root EBS volume. These options allow you to recover an instance. Recovering from an EBS volume is better because it can ensure system consistency, though it is a bit trickier with Windows instances.

3. Availability Zones and Regions. Amazon EC2 allows you to place resources in multiple locations across separate AZs and global regions. That way, if there is a failure in one zone, the performance of your resource in another will not be affected.

4. Resource Tagging. Resources, whether they are instances, EBS volumes or snapshots, can be tagged to leverage backup processes for a higher level of automation. If you tag your application’s resources and a new resource is added to the pool by the autoscaling mechanism, for example, the instance will still be automatically added to an existing backup policy.

5. CloudFormation. CloudFormation lets you treat and provision a stack of resources as one logical unit. With the CloudFormation template, you’re able to restore your application’s most current infrastructure stack in pilot light mode.
While these are the key features, AWS offers plenty of others to assist with backup and DR. Direct Connect, for example, allows you to securely streamline data and resources from your on-premises environment to your AWS secondary site.

**AUTOMATE YOUR BACKUP TESTS**

The purpose of your backup and DR site is, of course, to ensure that you can handle trouble if it arises. It is essential, therefore, to have automated routines in place to test recovery and to identify where problems might occur.

IT teams must also test consistency to ensure that policies, including RTO and RPO, are aligned. Make sure to document the guidelines for how recovery works and which actions would need to be performed manually. It’s all about finding the balance between bringing as much automation as possible into the process while knowing when things need to be carried out manually.

DR and backup are prime motivations for AWS adoption. Being able to recover your service at any point in time in almost any place in the world makes AWS a near-perfect solution. Still, to build and automate backup and DR, you’ll want to see which third-party tools available through the AWS Marketplace can assist and accelerate implementation.

—Ofir Nachmani
Load Balancing’s Role in App Availability

**Load balancing is** an effective way to improve the availability and scalability of certain applications. AWS’ Elastic Load Balancing (ELB) reroutes traffic to ensure that systems function despite server failures or outages.

Let’s look at some basic guidelines for using AWS load balancing to maintain availability, as well as information on how to combine it with the AWS Auto Scaling feature to further stabilize Amazon workloads.

Load balancers help maintain availability by routing traffic away from failed devices. They also create scalability by distributing work to servers that can process additional tasks. Load balancing places an extra server between client devices and back-end servers to distribute workloads among back-end servers. The load balancer then routes processing results back to clients.

**How load balancing works**

In a simple case, a load balancer directs 20% of all incoming traffic to five back-end servers. If one of the five servers fails, the load balancer detects this and stops routing traffic to that failed server. The remaining functioning servers then each get 25% of the workload.

A more sophisticated load balancer might monitor the load on each server and distribute incoming work away from heavily loaded servers to those with idle capacity.

Load balancing supports different sets of

*A sophisticated load balancer might monitor the load on each server and distribute incoming work away from heavily loaded servers.*
protocols to meet different needs. Basic load balancing can use HTTP from the client to the load balancer and from the load balancer to the back-end client. Secure communications between client devices and a website can use HTTPS between the client and the load balancer and HTTP from the load balancer to the back-end server.

The load balancer handles encryption and decryption; if end-to-end encryption is required, use HTTPS from client to load balancer and from load balancer to back-end server.

**AWS ELASTIC LOAD BALANCING FEATURES**

For high availability in a cloud environment, services must continue to run—even if there is a server failure or if connectivity to an availability zone (AZ) is lost. Admins can register AWS Elastic Compute Cloud (EC2) instances with a load balancer that’s configured to perform health checks, typically as pings to HTTP port 80. When a health check fails, AWS ELB reroutes traffic to other instances registered to the same load balancer.

EC2 instances can reside in multiple availability zones; if a single AZ is unavailable, traffic can be routed to registered instances in another AZ.

AWS recommends using two subnets in different zones to ensure high availability. As with other AWS resources, it’s important to configure security controls to meet your needs. With AWS load balancing, you likely will need to create a security group that contains rules to govern the flow of traffic to the load balancer.

On its own, AWS ELB can improve the availability of applications. Admins can also use AWS load balancing in conjunction with AWS Auto Scaling not only to distribute workloads but also to monitor and adjust compute resources as needed.

Auto Scaling adjusts the number of servers according to the workload and uses templates, known as launch configurations, to create EC2 instances. It also uses a policy, known as a scaling plan, to determine conditions that trigger a change in EC2 capacity. Auto Scaling manages groups of instances, which could also be registered to a load balancer. Combining load balancing and autoscaling can help improve
availability and scalability.

This approach is not suitable for all applications, however. If your back end depends on an application that is not easily distributed among servers, such as certain relational databases, then load balancing may not help improve availability. In such a case, a failover server may be a better option.

Another option is to use an AWS-managed database service such as Relational Database Service to avoid some database management overhead. —Dan Sullivan
Don’t Fail the Testing Test

**Some cloud consumers** might imagine that AWS is completely turnkey—everything works without having to be tested. In reality, it’s not that simple.

While it takes some effort to keep an AWS cloud fully operational, testing is a predictable and generally straightforward process. Many aspects of AWS make it far easier than testing in conventional environments. And the scale and diversity of AWS makes it easier to achieve high reliability.

Shlomo Swidler, CEO of New York-based consultancy Orchestratus, advises companies to think about reliability in a different way within AWS than they would in other environments. You can’t pay for a specific service level within AWS, but you can take advantage of options such as load balancing and autoscaling, which improve reliability.

“Those services let you take advantage of the scale of AWS and let you set up self-healing copies of machines that will be there when a given machine becomes unhealthy,” Swidler said.

Similarly, regardless of whether a business is running on traditional **big iron**, in a virtual environment or in a cloud, it has to understand what can fail and how much uptime its system requires. For instance, to ensure resilience on AWS, users need at least one instance in a different availability zone (AZ). That, in turn, means users need to have an AutoScale group stand up another instance so that two instances are always running.

“That way, if a zone goes out you have another instance in another zone,” said Andrew Siemer, chief architect at Clear Measure, a software engineering firm.

The more testing a user has performed on an existing system, Siemer said, the easier it is to validate a successful AWS migration. However, backup and disaster recovery (DR) are
different in the cloud than in an on-premises data center.

While AWS manages many activities, Siemer said, you still need to understand how a zone failing will affect your infrastructure design and implementation.

Not every user will be able to test reliability and DR at an elaborate scale, as Netflix does with Chaos Monkey, but every bit of effort is worthwhile. “Every company should constantly test what they have built to ensure it works as expected,” Siemer said.

Taking a similar approach, data management company OneVizion enlists a range of products and services to ensure its AWS-based operations function smoothly. For instance, it uses a third-party tool to ping its servers and monitor for high usage.

OneVizion has also adopted Connectria, a provider of managed AWS hosting services, to design its AWS infrastructure. The companies work together to build monitoring applications that look inside Elastic Compute Cloud to find problems before they manifest.

“We automatically log errors in our application and screen them daily to make sure that all components work properly,” said John Patton, OneVizion’s CEO.

OneVizion enlists a range of products to ensure its AWS operations run smoothly, including a tool to ping servers and monitor for high usage.

Another powerful option for AWS testing is simply spinning up new servers and replicating your environment. This is something that’s all but impossible in a traditional data center, but is easy in the cloud.

“This is one of the big benefits of moving to AWS. You can test new environments dynamically, without affecting operations,” Swidler said. —Alan R. Earls
ALAN R. EARLS is a Boston-based freelance writer focused on business and technology. He writes for several TechTarget sites, and his work has appeared in numerous trade publications and newspapers.

OFIR NACHMANI is a business technology advisor, blogger and lecturer. In addition to operating the critically acclaimed IamOnDemand.com, he advises organizations on executing a modern go-to-market strategy.

DAN SULLIVAN is an author, systems architect and consultant with more than 20 years of IT experience. He has had engagements in advanced analytics, systems architecture, database design, enterprise security and business intelligence across a broad range of industries.