Different AWS Pricing Models

To help customers choose an optimal pricing strategy, AWS offers four pricing models for its resources, particularly for compute and database resources. These pricing models apply to:

- EC2 instances
- RDS database instances
- Redshift cluster nodes
- EMR nodes
- Lambda functions

On-Demand Instances

This is the default pricing model for AWS instances. With this model, customers pay by the minute or by the hour and there is no upfront payment for resource usage. The advantage of using On-Demand is that it's flexible, and there's no long term commitment, nor pre-payment of any kind.

The drawback here is that due to this flexibility, this is the most expensive option. Without proper planning and budgeting, scaling up applications with only On-Demand instances will have a huge impact on the monthly bill.

For example, consider running an m5.xlarge general purpose Linux EC2 instance with 4 vCPUs and 16GB RAM in the us-east-1 region 24×7. Let's say, the instance will be using a 200 GB general-purpose SSD volume, with two daily snapshots, each snapshot having 3GB of data change. There will be no incoming data from the Internet, but there will be data coming from other instances in other or the same region. The instance will send an estimated 4 GB data to another instance in the same region and 4 GB data to other instances in different regions.

Using the AWS pricing calculator, the monthly price (as of July 2021) comes up to $174.81 American dollars.

Keeping the same configuration and workload requirements, purchasing a Reserved Instance for this node with one-year commitment and no upfront payment will lower the monthly bill to $122.98.

With a difference of $51.83 per month, it's easy to see why On-Demand instances are not good for scalability, ROI, or even for long term, predictable usage.

However, it's actually good to have a small percentage of On-Demand instances ready. Here are some use cases:

- Pre-production workloads—i.e., those used in development, testing or Proof-of-Concepts (PoC)
- When trying out AWS for the first time

It's also useful because not every use case may require long-term usage. Some workloads may only be seasonal—like end-of-year sales, or Christmas. It's not worthwhile to purchase long-term commitment instances for these seasonal workloads only to over provision them when the demand recedes. In this instance using On-Demand instances for the duration of the peak would save costs later down the track.

Spot Instances

Due to the enormity and scale of its infrastructure, AWS will always have some unutilized resources. To lessen the upkeep cost for these unused resources, AWS introduced Spot Instances.

Think of Spot Instances like an airline offering discounted prices for unoccupied seats on a flight. Since this is leftover capacity, customers can purchase these capacities for savings of up to 90% off the On-Demand price. However, there is a caveat. When another paying AWS customer needs the resource capacity and is willing to pay On-Demand, the Spot Instance is terminated and the capacity reassigned to the other customer. There is however, a hibernation option (so any processing can pick up where it was left off when a resource becomes available again) and a two-minute warning.

Spot Instances are mostly limited to short bursts of performance, and as such, not suitable for long running workloads. On top of that, these instances are not covered by AWS SLA—Amazon's commitment to ensure availability of services for each AWS region with a Monthly Uptime Percentage of at least 99.99%. Thus even if you hold a Spot Instance, AWS doesn't have any responsibility to ensure its uptime.

Considering the risks, Spot Instances are not seen as a viable option for mission-critical production applications. Spot's fault-intolerant and inflexible nature means using it for production purposes would need a solid, bullet-proof launch configuration that includes:

- Accurate predictions of instance outage and timely bidding in the Spot market.
- Automated handling of instance terminations to address critical data loss and capturing any stateful information.
- Time-critical re-launch of the correct number of instances to maintain the same level of performance and SLA.
- Automated load balancing configuration.

However, provided users are willing to accept the risks, there are few workloads which can be considered as "Spot Candidates":

- CI/CD pipelines. Here, losing an instance means running any unfinished workflow again once another instance becomes available

- Non-production workloads that don't need continuous uptime
- Workloads where lost data can be quickly regenerated
- Containerized workloads where container orchestration may need extra nodes from time to time.

Another way Spot Instances can save costs is when they are used in conjunction with other commitment-specific pricing models. Consider this scenario, for example:

Let's say a travel booking application runs on two EC2 instances. During peak seasons of the year, the auto-scaling group brings up two more instances to cope with the load. The customer has purchased four Reserved Instances, two of which are attached to the current instances. The rest are attached to other nodes. During peak seasons, these two RIs are detached from the other nodes and attached to the two extra nodes of the travel application.

As the company grows, sometimes even the four nodes can't keep up with the load at peak periods. If there's no definite pattern of these peaks, the company may not want to invest in more RIs. Instead, they can opt to spin up a Spot instance when the unpredictable spike happens. That way, it's not only allowing the application to operate with the same level of performance, but also saving costs.

Reserved Instances

Reserved Instances (RIs) are similar to the default On-Demand pricing model, except customers "reserve" a certain resource for a 1 to 3-year "term." As part of the reservation, the customer pays an upfront fee. This upfront payment can be the:

- Total cost of running the instance over the chosen period (full upfront payment).
- Partial cost of running the instance over the chosen period (partial upfront payment).
- No cost of running the instance over the chosen period (no upfront payment).

Customers can save up to 72% of AWS instance costs, depending on how much they pay upfront, the payment term (1 or 3-year), and which type of RIs are available. In simplest terms, the more you pay upfront and the longer the commitment, the more you save.

There are three different types of RIs: Standard Reserved Instances, Convertible Reserved Instances (which have a smaller discount, but are more flexible) and Scheduled Reserved Instances. Each type has a best use case to consider:

- **Standard**: Steady-state production systems that will run for long periods of time.

- **Scheduled**: Production systems that will run for long periods of time with definite and predictable peaks and workloads.
- **Convertible**: Production systems that will run for long periods of time, but may have slightly different needs and allocations than normal.

Now, one thing to be aware of is that RIs are not tied to any particular instance. They work more like a purchase license. In other words, you can apply your RIs on a number of instances, and then decide to take those instances off of the RI and apply the RI to another group of eligible instances. While RIs save a lot of money for users, they have pitfalls. The flexibility of being able to convert to a different instance type can be confusing. Moreover, once purchased, customers will have to pay for the reserved capacity regardless of its usage (or lack thereof). Many organizations switch to using RIs to save money, but it's not always easy to manage them manually to achieve the best possible pricing benefits.

Realizing that many customers were facing difficulties with their unused RIs, AWS introduced the AWS Reserved Instance Marketplace. This is where organizations can list their unused RIs for other companies to purchase. With the RI Marketplace, companies don't have to buy a full 1 or 3-year term commitment—RIs with up to 1 month's remaining term are eligible for sale. Our blog post on AWS Reserved Instance Marketplace explains this in detail.

Savings Plans

Introduced in 2019, AWS Savings Plans are an alternative to Reserved Instances. With Savings Plans, instead of committing to a payment term and instance type, customers commit to spending a minimum amount of money per hour for the next one or three years. In exchange, AWS charges a significantly reduced hourly rate during the term. Also, Savings Plans are applicable to computing resources like EC2, Fargate, or Lambda. This allows the flexibility of choosing the best instance type for getting pricing discounts.

There are currently two types of Savings Plans offered:

- **Compute**: These are best suited for systems that evolve quickly and may need a different set of services other than EC2, and different instance families in the future. It costs more but is also more flexible. Compute Savings Plans can be applied to EC2 instances, Fargate, or Lambda services. This extends over any instance family, size, AWS region, operating system, or tenancy. The savings can be up to 66%.
- **EC2**: As the name suggests, this is applicable to EC2 only, and applicable only for a specific instance family in a particular region. It should be mentioned that this is regardless of availability zone, across any instance size, operating system, or tenancy. This is best suited for large but predictable workloads. EC2 SPs are more restricted, but offer greater savings (up to 72%).

The disadvantage of SPs is that it's currently available for EC2 Compute, Fargate, and Lambda usage only. Meaning you'll still need a different approach to save on your RDS, EMR, or Redshift expenses. Another shortcoming is that RIs have a marketplace for selling unused instances, while SPs don't. That means customers are locked into spending the minimum amount for the length of time chosen. Overcommitting to SPs can therefore cause a huge spike in expenses. Lastly, purchasing Savings Plans takes more careful planning when compared with alternative pricing models.

There are many reasons for embracing hybrid including reducing costs, improving efficiency, etc. But just because hybrid comes with an array of benefits does not mean it is without challenges.

For starters, hybrid cloud security is a complicated animal. By adding the public cloud to the mix, your security is now directed by APIs, and as a result, developers rather than operations will be responsible for its implementation. This can be a tremendous skills gap for many who are not used to these types of tasks. Furthermore, when data is transferred between various clouds, it is extremely vulnerable to malicious actors. Robust encryption can help ensure your data is protected as it travels between various clouds.

Governance and compliance are also challenges for those getting started with hybrid clouds because they are handled differently and managed by different teams depending on if you are working on premises, on private, or on public clouds. This may mean your team lacks experience handling these initiatives, and as a result, they'll require some training to make it successful.

Final words

Cloud computing with AWS requires a certain level of preparation and planning. Knowing what type of resources you need to run your workload and estimating the usage pattern as best as you can will go a long way towards optimizing your cloud expenditures. Also, the simpler the needs of your application, the cheaper it will be to host it on AWS—so you may want to re-architect large monolithic applications to smaller, independent services that work together.

That being said, there's no one-size-fits all solution to every scenario. Each use case will have its own set of requirements. A good combination of pricing plans can yield a better result in most cases.

For example, you can use a combination of Reserved Instances and Savings Plans to cover most of your uninterruptible production workloads and then fill the rest of the gap using On-Demand and Spot instances using automated observations and pricing calculations.

It's easier said than done though. Developing such a system to effectively manage your instances, plans and billing is quite complex. And it will definitely involve your engineers' time which can be better spent on developing new features for your application.

Fortunately, Zesty offers an automated platform to manage your RIs with zero engineering effort. Using real-time data, Zesty uses AI to automatically purchase and sell the most lucrative deals in the RI marketplace, resulting in increased savings and better budgeting predictions. With Zesty, you can leverage the strengths of AWS cloud without worrying about the price-related challenges.
Interested in learning more? Chat with one of our cloud experts to find out how you can automate your cloud experience to cut costs by over 50%.