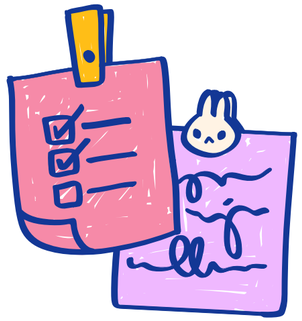
DevKTOps

ARCHITECTING ON aws

Module - 8 High Availability/Elasticity

Module Overeiw

- Understanding High Availability
- Understanding Elasticity
- Monitoring
- Scaling

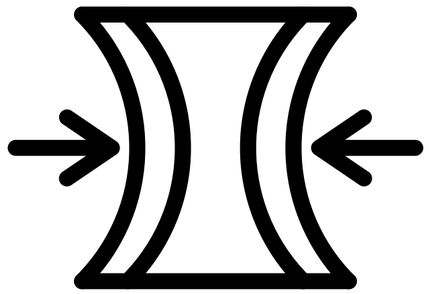


High Availability

Three factors that determine the overall availability of your application

- Fault tolerance : The built-in redundancy of an application's component
- Scalability: The ability of an application to accommodate growth without changing design
- Recoverability: The process, policies, and procedures related to restoring service after a catastrophic event

Elasticity



An elastic infrastructure can intelligently expand and contract as its capacity needs change.

Example :

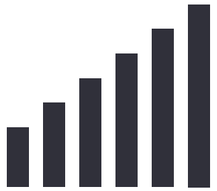
- Increasing the number of web servers when traffic spikes
- Lowering write capacity on your database when that traffic goes down
- Handling the day-to-day fluctuation of demand throughout your architecture

Three Types Elasticity

Time-Based : Turning off resources when they are not being used (Dev and Test environments)

Three Types Elasticity

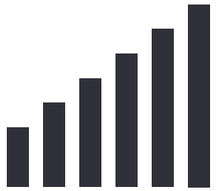
Time-Based : Turning off resources when they are not being used (Dev and Test environments)



Volume-Based : Matching scale to the intensity of your demand (making sure you have enough computer power)

Three Types Elasticity

Time-Based : Turning off resources when they are not being used (Dev and Test environments)

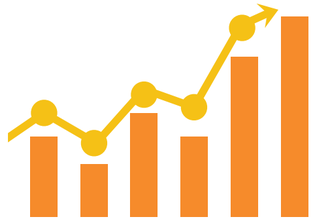


Volume-Based : Matching scale to the intensity of your demand (making sure you have enough computer power)

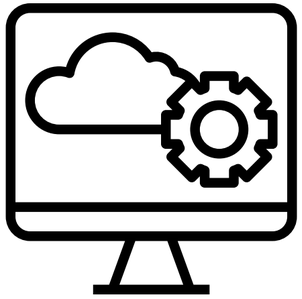
Predictive-Based : Predict future traffic based on daily and weekly trends (including regular-occurring spikes)

Monitoring

Operational Health



Resource Utilization



Application Performance

Security Auditing

Monitoring to Understand Cost

- Cost Optimization Monitor – Can generate reports that provide insight into service usage and cost. Provides estimates costs that can break down by period, account, resource or tags.

- AWS Cost Explorer – Can view data up to the last 13 months, allowing you to see patterns in how you spend on AWS resources over time.

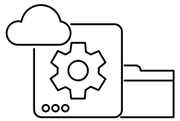
- Forecasting with AWS Cost Explorer - A forecast is a prediction of how much you will use AWS services over the forecast time period you selected, based on your past usage. You create a forecast by selecting a future time range for your report. Forecasting provides an estimate of what your AWS bill will be and enables you to use alarms and budgets for amounts that you're predicted to use. Because forecasts are predictions, the forecasted billing amounts are estimated and might differ from your actual charges for each statement period.

CloudWatch

- Collects and tracks metrics for your resources
- Enables you to create alarms and send notifications
- Can trigger changes in capacity in a resource, based on rules that you set

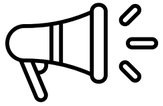
The Ways of CloudWatch

Metrics



Logs

Alarms



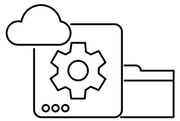
Events

Rules

Targets

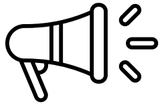
CloudWatch Metrics

Metrics



Logs

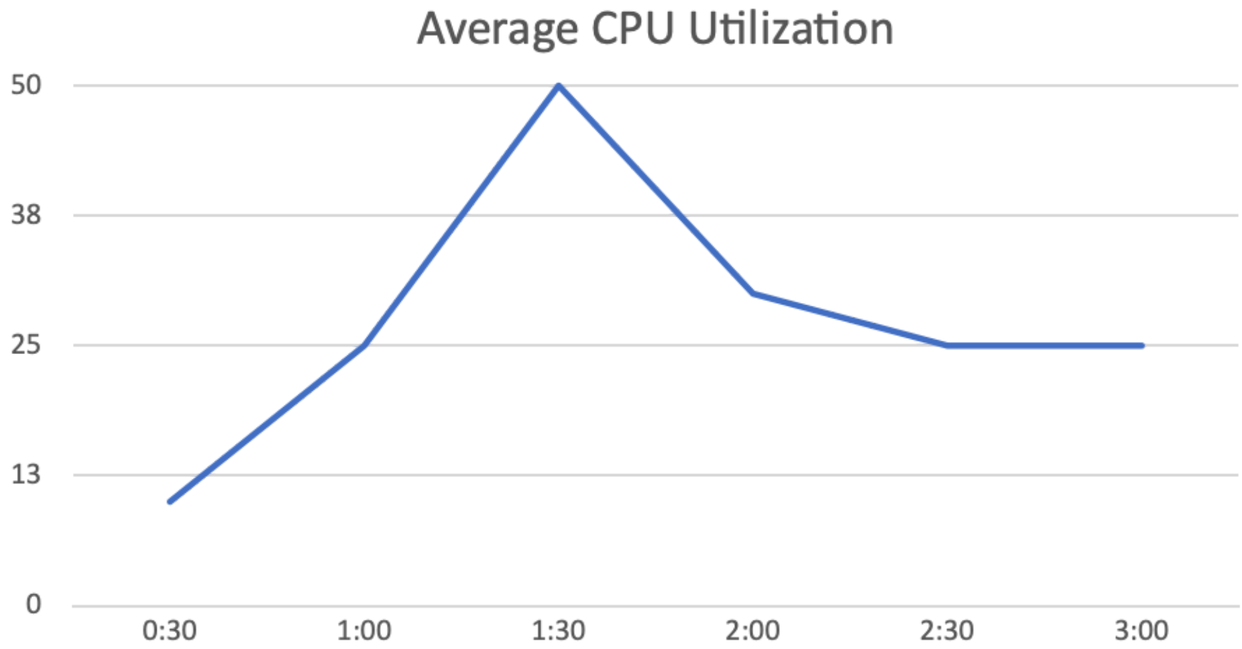
Alarms



Events

Rules

Targets

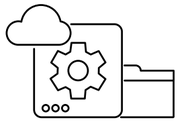


Metric data is kept for a period of 15 months

CloudWatch Logs

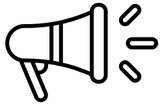
DevKTOps

Metrics



Logs

Alarms



Events

Rules

Targets

Web Application

Agent

Errors: 4
Warnings: 10
....
....

Log_file.txt

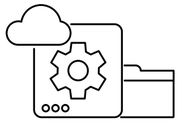
Amazon CloudWatch

Amazon S3

Sources Examples
AWS CloudTrail
Route 53
Amazon EC2

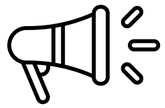
CloudWatch Alarms

Metrics



Logs

Alarms



Events

Rules

Targets

Application

CPU Utilization

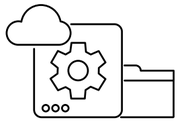
80%, 60%, 50%, 25%, ...

If CPU Utilization is > 50% for 3 minutes

Do an action
. Send a notification to the DevOps Team
. Create another instance

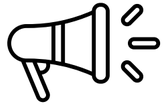
Amazon EventBridge Events

Metrics



Logs

Alarms



Events

Rules

Targets

Instance Down Event

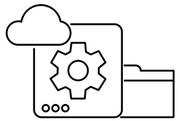
Amazon EventBridge

Event Examples

EC2 Instance Change
Auto Scaling state change
Console sign-in
...

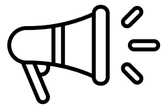
CloudWatch Rules

Metrics



Logs

Alarms



Events

Rules

Targets

Stop Event

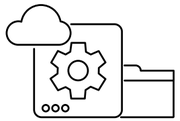
```json
{
  "source": ["aws.ec2"],
  "detail-type": ["EC2 Instance State-change Notification"],
  "detail": {
    "state": ["stopped"]
  }
}
```

Amazon EventBridge

CloudWatch Targets

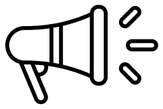
DevKTOps

Metrics



Logs

Alarms



Events

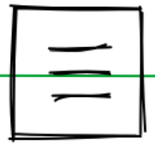
Rules

Targets

Stop Event

```json
{
  "source": ["aws.ec2"],
  "detail-type": ["EC2 Instance State-change Notification"],
  "detail": {
    "state": ["stopped"]
  }
}
```

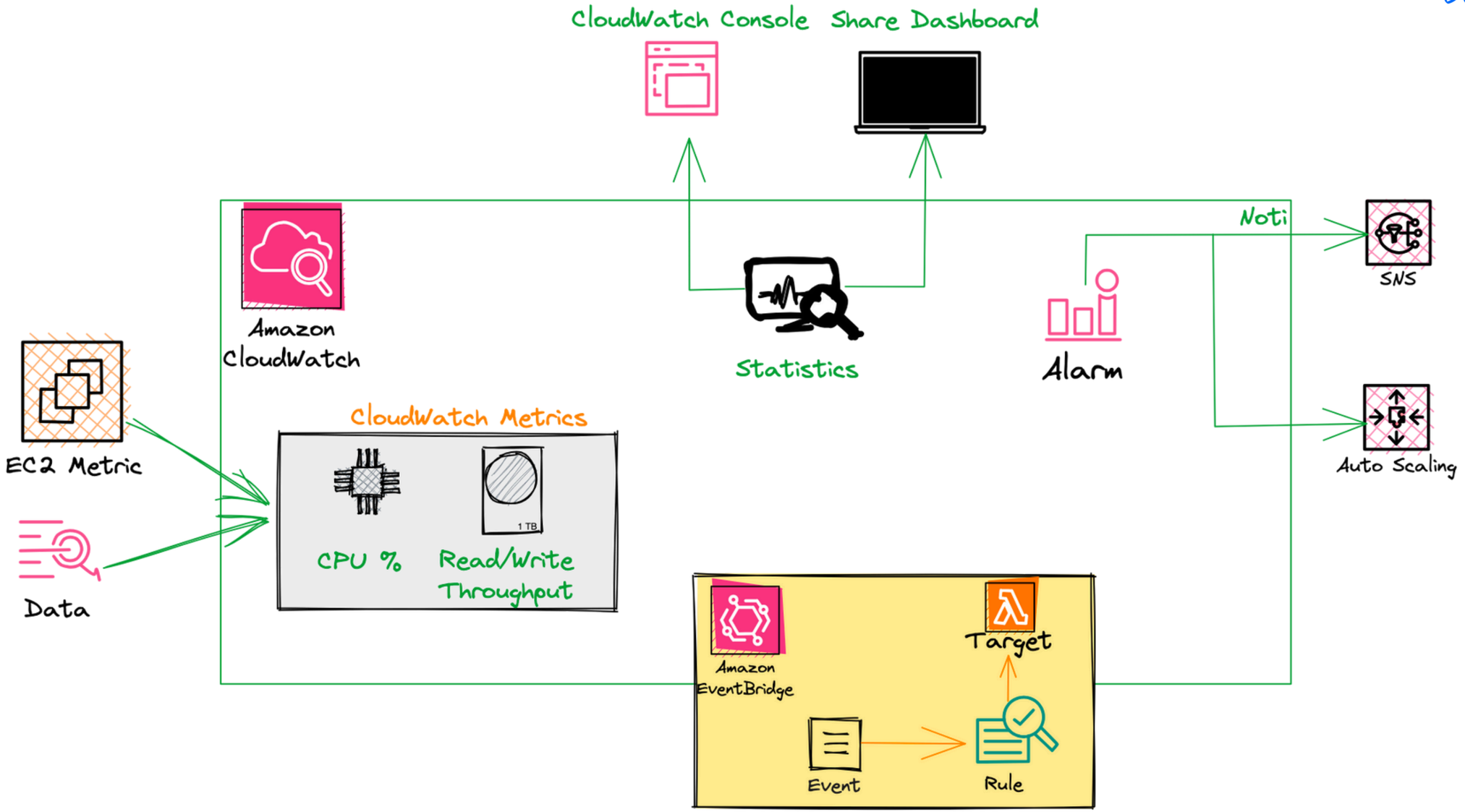
Amazon EventBridge



EC2 Instances
AWS Lambda
Amazon ECS
Amazon SNS
Amazon SQS

Target Examples

CloudWatch Visualized



CloudTrail

AWS CloudTrail is a web service that records AWS API calls for your account and delivers log files to you with Amazon S3 bucket.

With CloudTrail, you can get a history of AWS API calls for your account, including API calls made via the AWS Management Console, AWS SDKs, command line tools, and higher-level AWS services (such as AWS CloudFormation).
You turn on CloudTrail on a per-region basis. If you use multiple regions, you can choose where log files are delivered for each region.

Monitoring Your Network

VPC Flow Logs

- Captures traffic flow details in your VPC
- Accepted, rejected, or all traffic
- Can be enabled for VPCs, subnets, and ENIs
- Logs published to CloudWatch Logs
- Logs published to Amazon S3

Using Auto Scaling for Elasticity

Amazon EC2 Auto Scaling

- Launches or terminates instances based on specified conditions
- Automatically registers new instances with load balancers when specified
- Can launch across Availability Zones

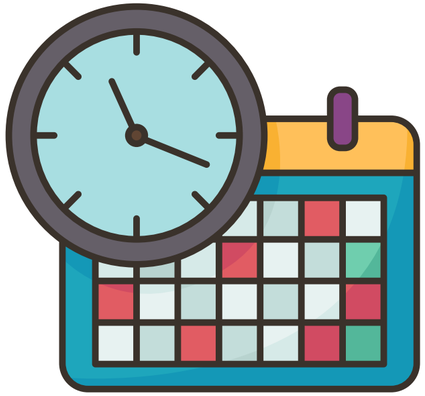
Auto Scaling group defines:

Desired capacity

Minimum capacity

Maximum capacity

Auto Scaling Ways

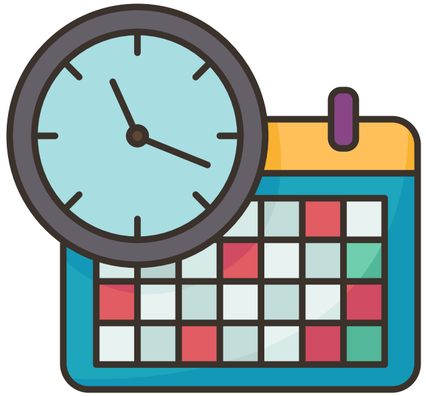


Scheduled

Good for predictable workloads

Use Case : Turning off your Dev and Test Instances at night

Auto Scaling Ways



Scheduled

Good for predictable workloads

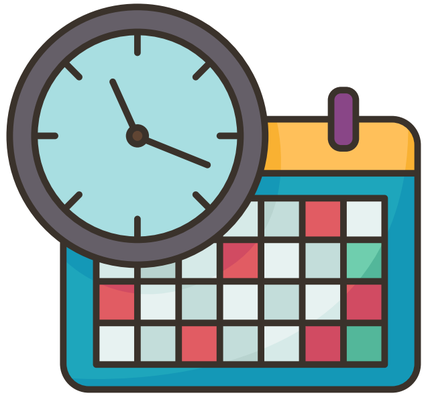
Use Case : Turning off your Dev and Test Instances at night

Dynamic

Excellent for general scaling

Use Case : Scaling based on CPU utilization

Auto Scaling Ways



Scheduled

Good for predictable workloads

Use Case : Turning off your Dev and Test Instances at night

Dynamic

Excellent for general scaling

Use Case : Scaling based on CPU utilization

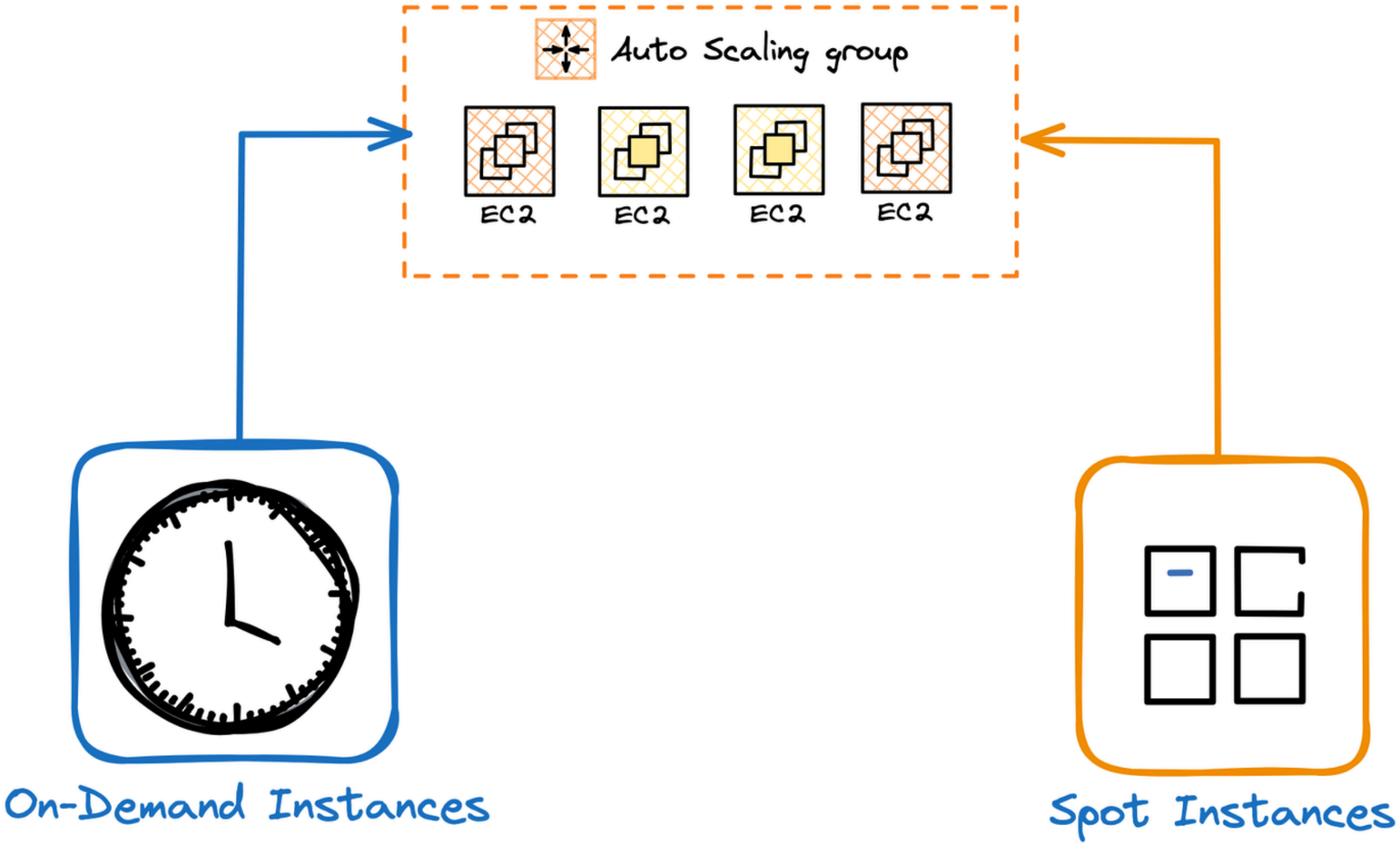
* simple scaling, step scaling, target tracking scaling

Predictive

Easiest to use

Use Case : No longer need to manually adjust rules

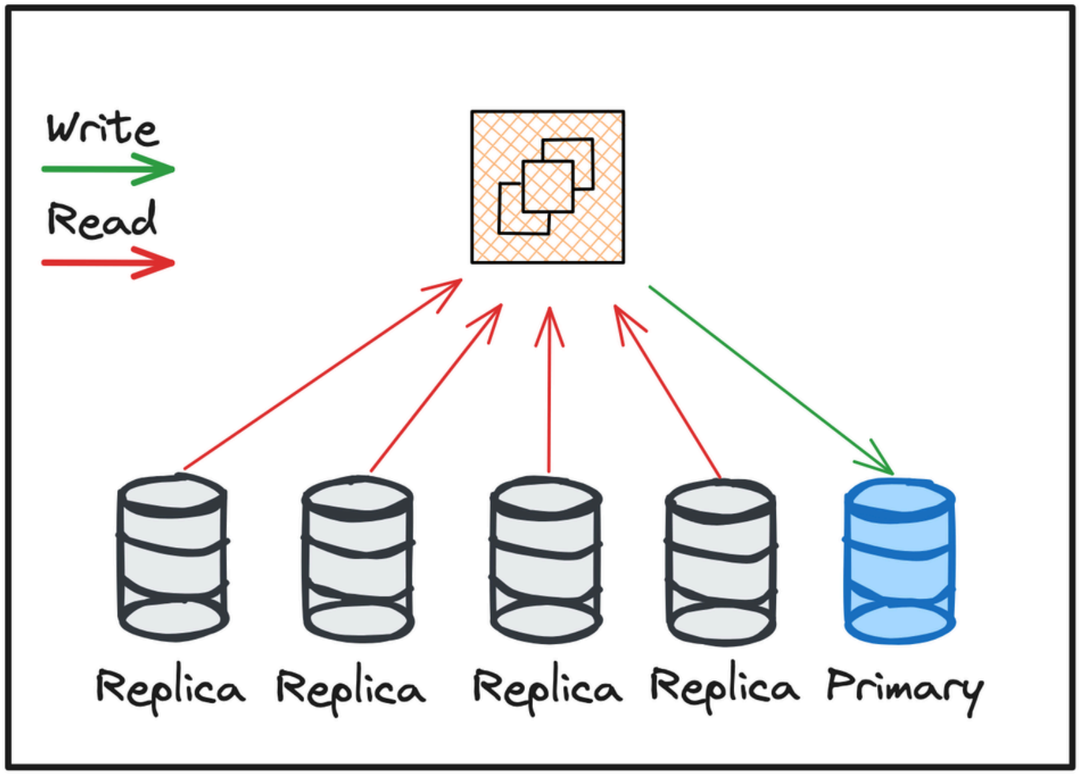
Auto Scaling Instance Options



Auto Scaling Considerations

- You might need to combine multiple types of autoscaling
- Your architecture might require more hands scaling using: Step Scaling
- Some architectures need to scale on two or more metrics (e.g. not just CPU)
- Try to scale out early and fast, while scaling in slowly over time
- Use lifecycle hooks
- Perform custom actions as Auto Scaling launches or terminates instances

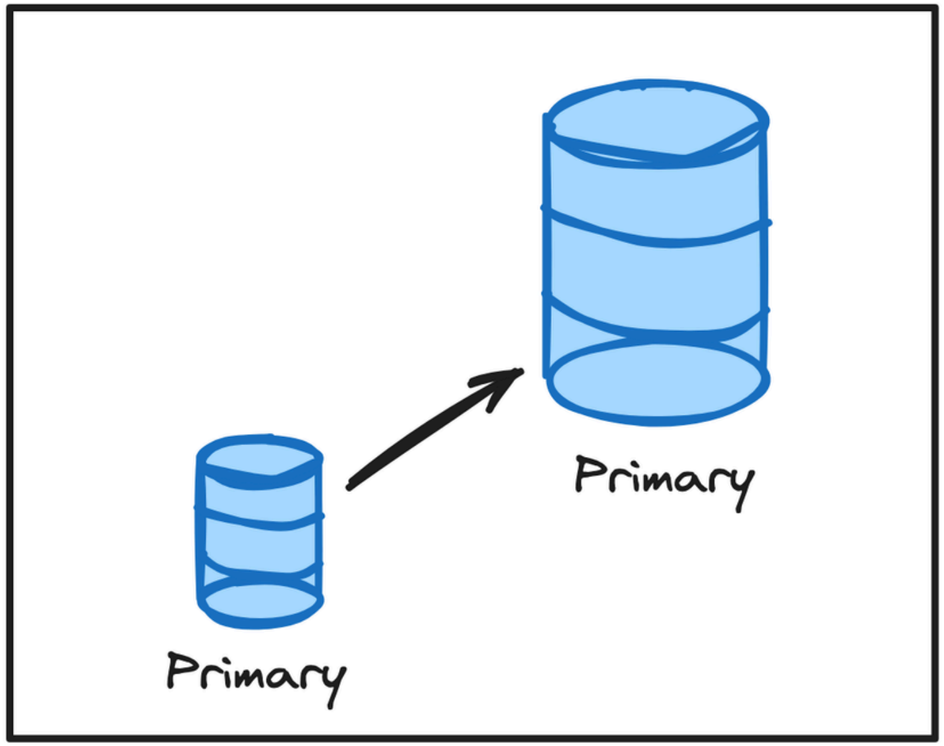
Scaling Your Databases



Horizontal Scaling

- Horizontally scale for read-heavy workloads
- Offload reporting
- Keep in mind:
  - Replication is asynchronous
  - Currently available for: Amazon Aurora, MySQL, MariaDB, PostgreSQL, and Oracle.

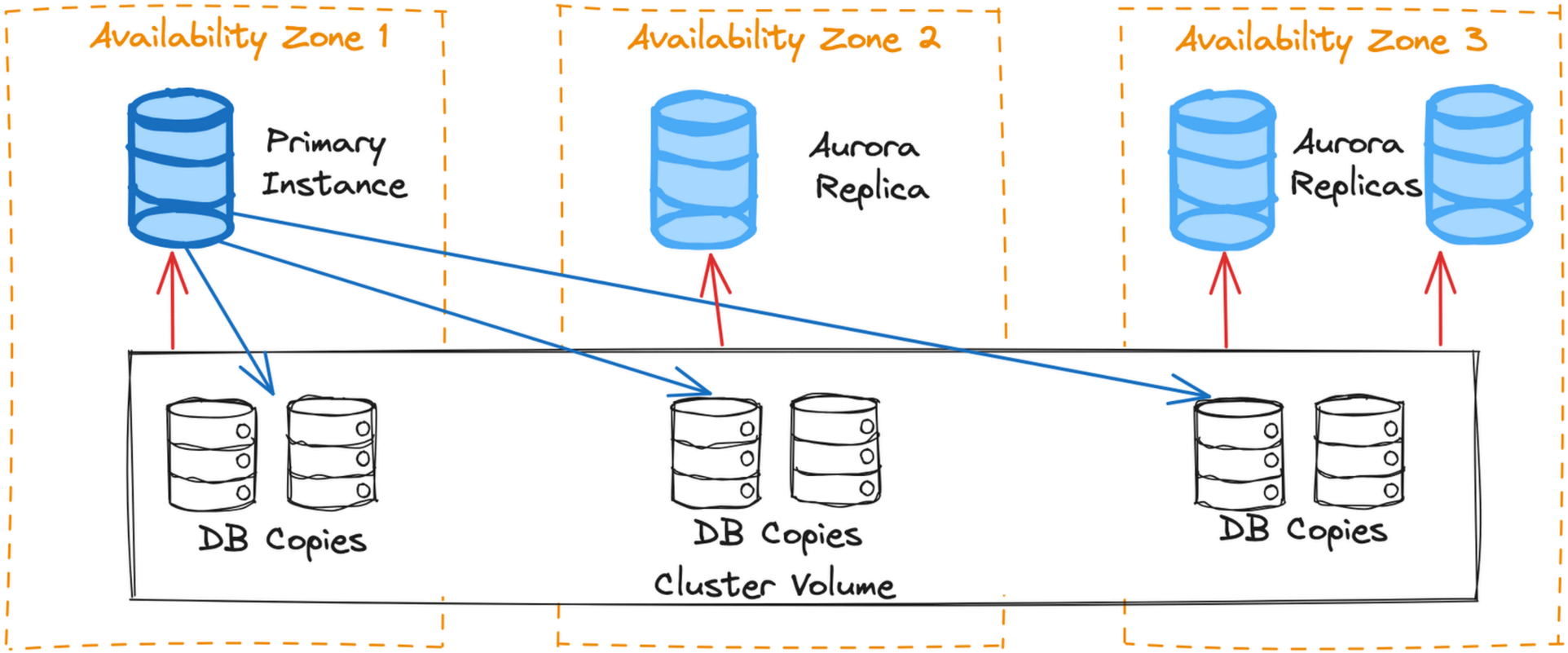
Scaling Your Databases



Vertical Scaling (Push-Buttom Scaling)

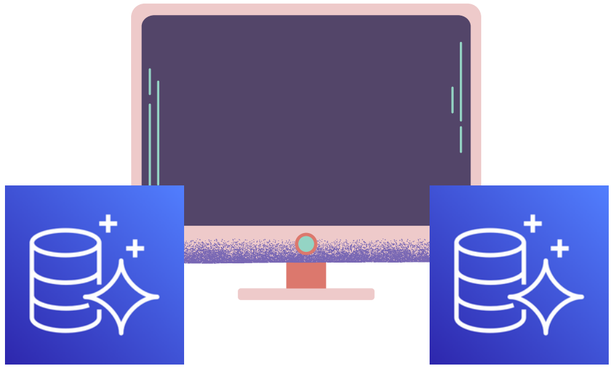
- Scale nodes vertically up or down
- From micro to 24xlarge and everything in-between
- Scale vertical often with no downtime*
- Monitor with autoscaling

Aurora DB Cluster



Each Aurora DB cluster can have up to 15 Aurora Replicas

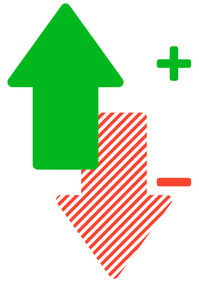
Aurora Serverless



Responds to your application automatically :

- Scale capacity
- Shut down
- Start up

Pay for number of ACUs used

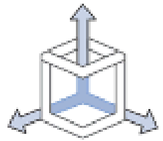


Good for spiky, unpredictable workloads

DynamoDB Scaling

Auto Scaling

Default for all new tables



Specify upper and lower bounds

Use case: General scaling, great solution for most applications.

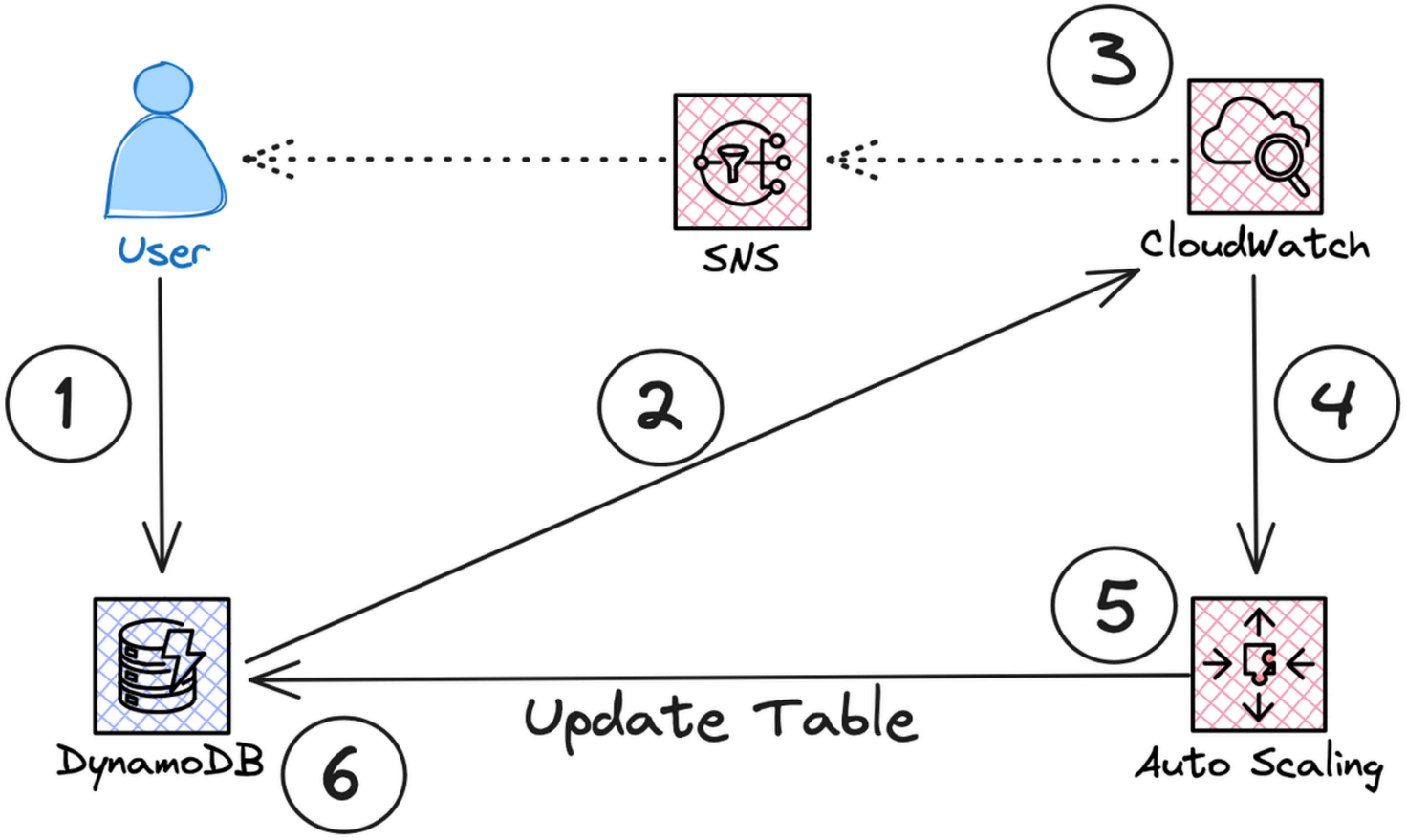
On-Demand

Pay per request

No more provisioning

Use case: Spiky, unpredictable workloads. Rapidly accommodates to need.

DynamoDB Auto Scaling in Action



DevKTOps

Thank you!

Architecting on AWS : Module 8