

Deploy Rapydo Scout (Observability) in your AWS environment

For any question or issues please contact us at: info@rapydo.io

Introduction

This documentation will guide you through the deployment of Rapydo Observability using a single EC2 instance running in your AWS VPC.

What We Do

Rapydo is all about helping you get the most out of your Amazon RDS for MySQL and Amazon Aurora databases hosted in your AWS account (referred to as MySQL from onwards).

Whether you're trying to fix performance bottlenecks, scale your infrastructure, or simply make sure everything is running as efficiently as possible, we've got the tools to make it happen. We provide real-time monitoring, actionable insights, and recommendations to keep your databases in top shape—without the headaches.

[Why Rapydo?](#)

We know that no two databases are the same. That's why Rapydo is designed to adapt to your unique setup, whether you're running a single MySQL instance or managing multiple clusters across the globe. We don't believe in cookie-cutter solutions. Instead, we give you the flexibility to monitor, tweak, and optimize your databases exactly how you want.

Plus, we're all about transparency. With Rapydo, you get a clear view of what's happening under the hood of your MySQL instances. No more guesswork or reactive troubleshooting—our platform gives you the information you need, when you need it, so you can stay ahead of potential issues.

[Who Is It For?](#)

Whether you're a database administrator (DBA), a part of a development team, or DevOPS, Rapydo is built with you in mind. Our goal is to make database monitoring and optimization as seamless as possible, so you can focus on building great products rather than putting out database fires. No matter your level of expertise, our platform is designed to guide you through best practices, optimizations, and performance enhancements.

[How We Help You?](#)

- **Real-Time Monitoring:** Stay updated on your MySQL database health with instant insights into performance metrics.
- **Performance Tuning**:** Our platform helps you fine-tune your databases by offering suggestions based on best practices.
- **Scalability:** Whether you're dealing with a single database or an entire fleet, Rapydo scales with you, offering support for both horizontal and vertical scaling.
- **Custom Alerts**:** Set up alerts tailored to your unique environment, so you know about issues before they become problems.
- **Easy Setup**:** We don't believe in complicated onboarding processes. Getting started with Rapydo is quick and easy, so you can begin optimizing in no time.

Pre-requisites

- For deployment skills prerequisite:
you need a DevOps with basic AWS skills equal to minimum Cloud Practitioner
- ideally Solution Architect, DBA or System Administrator or Cloud Operations certification knowledge level
- For operations/usage of Rapydo skills prerequisite: Data Engineer certification or equal
- You must have an existing AWS account, VPC, AZ and subnet to deploy our solution.
- you must have existing RDS instances of Amazon RDS for MySQL and Amazon Aurora database, 5.7 and 5.8
- Performance Insights must be enabled, otherwise Rapydo functionality will be limited
- RDS instance sizes not compatible with Performance Insights - please do not use them with Rapydo:
 - Db.t2.micro
 - Db.t2.small
 - Db.t3.micro
 - Db.t3.small
 - Db.t4g.micro
 - Db.t4g.small
- AWS regions supported for deployment: all AWS commercial regions are supported. No support for AWS China and GovCloud.

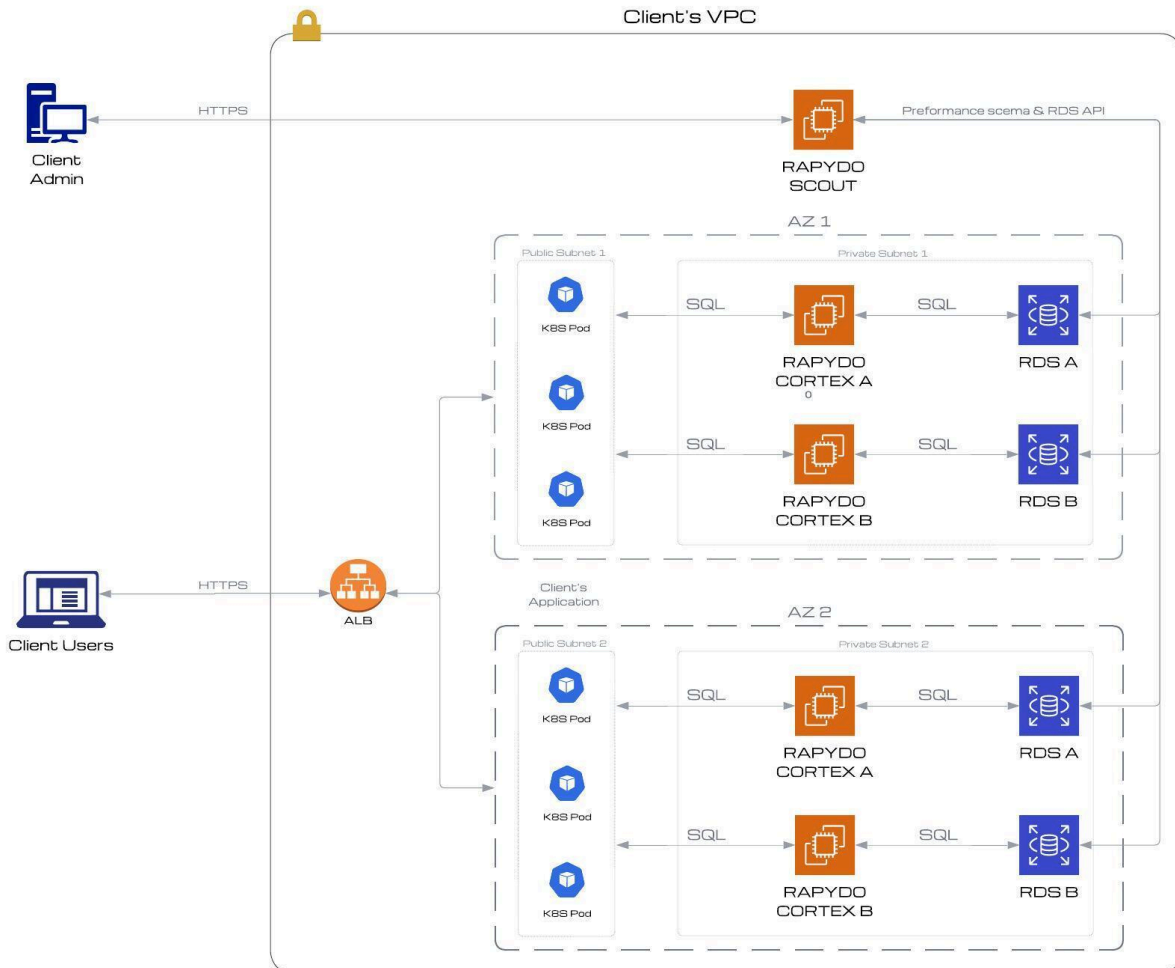
Overview

Description of a typical deployment of Rapydo in AWS Resources:

- your existing RDS instances (s) that Rapydo will be monitoring and optimizing
- New resources created by Rapydo deployment Cloud Formation Template (see below)
- one EC2 instance - default t3a.xlarge - image running Ubuntu 24.04 currently
 - EBS 6000 IOPS gp3 storage
 - ENI
- Security group + egress rule to reach S3
- New IAM role for EC2
- New IAM policy for accessing RDS, accessing docker images from Rapydo's ECR
- Optional: frontend (web interface of Rapydo) TLS certificate created in Ubuntu during deployment or generate and deploy your own TLS certificate

Expected time to deploy solution: ~30 minutes

Architecture diagram of typical deployment



Security

Please always follow AWS Well Architected and Security Shared Responsibilities policies when managing your AWS environment!

Warning – DO NOT USE AWS root account for Rapydo deployment or usage.

There are no resources with public access deployed by Rapydo.

Rapydo does NOT store customer sensitive data – we only handle RDS metadata.

You are responsible for protecting your RDS instances with encryption.

Rapydo adheres to least privilege principles within AWS deployments as well.

- IAM role created for EC2 instance deployed

Unset

```
RoleName: !Sub "rapydo-scout-ec2-role-${AWS::StackName}"
Description: A role assigned to Rapydo EC2 instances
AssumeRolePolicyDocument:
  Statement:
    - Action: sts:AssumeRole
      Effect: Allow
      Principal:
        Service: ec2.amazonaws.com
  Version: "2012-10-17"
ManagedPolicyArns:
  - arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
  - !Ref RapydoDockerRegistryAccess
```

- IAM policy created to provide access to RDS configuration and operational data

Unset

```
DBScoutPolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyName: !Sub 'RapydoDBScoutPolicy-${AWS::StackName}'
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Action:
            - rds:DescribeDBClusters
            - rds:DescribeDBInstances
            - rds:DescribeDBParameters
          Effect: Allow
          Resource:
            - !Sub "arn:aws:rds:${AWS::Region}:${AWS::AccountId}:cluster:*"
            - !Sub "arn:aws:rds:${AWS::Region}:${AWS::AccountId}:db:*"
            - !Sub "arn:aws:rds:${AWS::Region}:${AWS::AccountId}:pg:*"
        - Action: cloudwatch:GetMetricData
          Effect: Allow
          Resource: "*"
    Roles:
      - !Ref RapydoScoutEC2Role
```

- IAM policy to provide access to the Rapydo docker image in Amazon ECR hosted in Rapydo's AWS account (jsx)

Unset

```
RapydoDockerRegistryAccess:
  Type: AWS::IAM::ManagedPolicy
  Properties:
    PolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: Allow
          Action:
            - ecr:GetDownloadUrlForLayer
            - ecr:BatchGetImage
          Resource:
            - !Sub "arn:aws:ecr:${AWS::Region}:941803196984:repository/*"
        - Effect: Allow
          Action:
            - ecr:GetAuthorizationToken
          Resource:
            - "*"
    ManagedPolicyName: !Sub
      "rapydo-docker-registry-access-policy-${AWS::StackName}"
```

- Before deployment, customer is required to create a key pair for accessing Rapydo EC2 instance over SSH, and select it during the CFT deployment
- Rapydo advises customers to store related secrets in AWS Secrets Manager or Hashicorp Vault
- Encryption
 - You are responsible for protecting your RDS instances with encryption.
 - You can disable IMDSv1 for the EC2 instance Rapydo creates through the CFT.

Infrastructure costs of Rapydo deployment in your account

Please check the resources created in the Overview chapter.

Your main additional AWS infrastructure cost will impact from the

- EC2 resource created for Rapydo
- CloudWatch metrics consumed (GetMetricData)

For your current AWS pricing consult with your FinOps expert, check AWS calculator or contact your AWS Account manager.

Sizing

- For EC2 instance, we suggest that you pick **t3a.xlarge** size by default. Only x64 instance types are supported (no Graviton support currently).
- For customers running a large number of RDS systems (100+), we suggest picking EC2 instance size **r7a.xlarge**
- For EBS storage we define 6000 IOPS gp3 storage for our EC2 instance by default, please change this to 12 000 IOPS if you are running a large number of RDS systems (100+)

Health Check

In order to validate if your Rapydo instance is running in a healthy (green) status, please check the following in Rapydo and in your AWS console:

1. Try logging in to Rapydo web console
2. list of RDS instances in the console is initially empty, once you have onboarded RDS instances into Rapydo, you can see in the Database Details View in our console if RDS instance metadata/config information is loaded

Backup, Recovery and Maintenance

- Backup and Recovery
 - You should protect the operational data in Rapydo in the following way:
 - EBS snapshots can be used
 - AWS Backup can be used to back up the EBS of the Rapydo EC2 instance
- Routine Maintenance
 - Recurring actions or monitoring you should run on Rapydo EC2 instance (updating etc)
 - Cloudwatch can be used to follow Rapydo EC2 instance metrics
 - TLS certificate rotation for the web console
 - EC2 OS Patching is customer responsibility using Ubuntu default or other centralized patching solutions.
 - Rapydo itself is updated through container update through ECR with the following commands (jsx):

Unset

```
rapydo-update  
rapydo-restart
```

- AWS service limits: Rapydo itself is not going to hit any AWS service limits, if your RDS instances need to be adjusted, please follow standard AWS procedures for account service limit changes
- Emergency Maintenance
 - If your Rapydo EC2 instance crashes and is not restarting:
 - Redeploy Rapydo CFT, and restore your connections to RDS, or restore your AWS Backup or EBS Snapshot

Support

- Rapydo offers a standard support offering (single tier) to our customers, with a guaranteed response time specified in your own contract with Rapydo.
- Support is available through email: support@rapydo.io and Slack
- If any problems in your RDS instances need AWS support, you need to use your own AWS support subscription to open ticket with AWS

Deploy using the CloudFormation template

1. Deploy the required AWS resources using the CloudFormation template through [this link](#). Fill out the parameters to match your environment.
Make sure you're deploying to your preferred region.
2. When the deployment completes a new EC2 instance will be running. Make sure it is accessible using SSH and log into it as the `ubuntu` user.
3. Make sure there's access from the instance to the target MySQL DBs you'd like to monitor on port 3306 (bash):

Unset

```
mysql -h <rds-endpoint> -u<user> -p
```

4. Create an admin user by running (bash):

Unset

```
create-admin-user
```

5. Open your browser with the address of the EC2 instance running Rapydo and login using the credentials created in the previous step.

Configuring Rapydo Scout

1. Adding databases to Rapydo Scout: In the `Databases Details` screen, select `Configure Databases`. You should see all the MySQL instances existing in the same region:

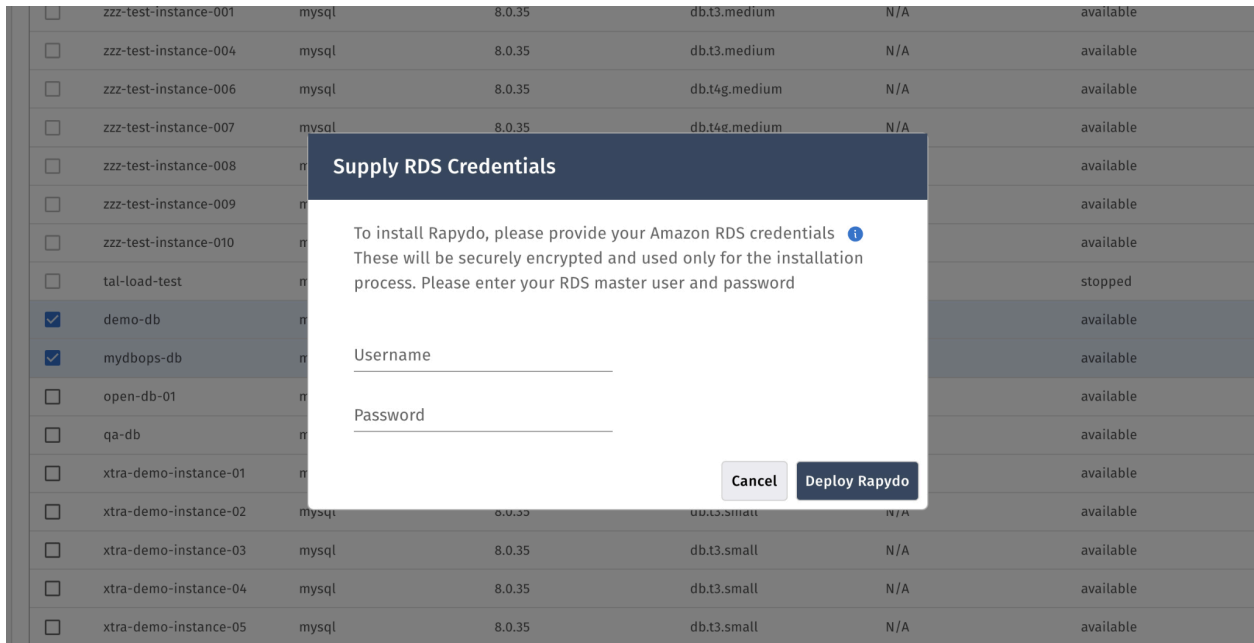
Configure Databases

DB NAME ↑	ENGINE	ENGINE VERSI...	SIZE	REGION & AZ	CPU	MEMORY	CONNECTIONS	MAX DURATI...
staging-db	mysql	5.7.44	db.t3.medium	us-east-1a	30%	217 MB	60 / 13 / 3	00:00:03
tal-load-test	mysql	8.0.35	db.m6gd.large	us-east-1b	3%	5 GB	628 / 6 / 1	00:00:00
zzz-test-instance-001	mysql	8.0.35	db.t3.medium	us-east-1b	16%	2 GB	1500 / 7 / 1	00:00:00
zzz-test-instance-004	mysql	8.0.35	db.t3.medium	us-east-1a	30%	2 GB	1500 / 8 / 1	00:00:00
zzz-test-instance-006	mysql	8.0.35	db.t4g.medium	us-east-1c	18%	2 GB	1500 / 6 / 1	00:00:00
zzz-test-instance-007	mysql	8.0.35	db.t4g.medium	us-east-1f	21%	2 GB	1500 / 8 / 1	00:00:00
zzz-test-instance-008	mysql	8.0.35	db.t4g.medium	us-east-1c	19%	2 GB	1500 / 8 / 1	00:00:00
zzz-test-instance-009	mysql	8.0.35	db.t4g.medium	us-east-1a	20%	2 GB	1500 / 6 / 1	00:00:00
zzz-test-instance-010	mysql	8.0.35	db.t4g.medium	us-east-1c	5%	2 GB	1500 / 6 / 1	00:00:00

<input type="checkbox"/>	DB IDENTIFIER	ENGINE	ENGINE VERSION	SIZE	CLUSTER IDENTIFIER	STATUS	HAS RAPYDO
<input type="checkbox"/>	staging-db	mysql	5.7.44	db.t3.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-001	mysql	8.0.35	db.t3.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-004	mysql	8.0.35	db.t3.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-006	mysql	8.0.35	db.t4g.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-007	mysql	8.0.35	db.t4g.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-008	mysql	8.0.35	db.t4g.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-009	mysql	8.0.35	db.t4g.medium	N/A	available	Yes
<input type="checkbox"/>	zzz-test-instance-010	mysql	8.0.35	db.t4g.medium	N/A	available	Yes
<input type="checkbox"/>	tal-load-test	mysql	8.0.35	db.m6gd.large	N/A	stopped	Yes
<input type="checkbox"/>	demo-db	mysql	8.0.33	db.m6g.large	N/A	available	No
<input type="checkbox"/>	mydbops-db	mysql	8.0.33	db.t3.large	N/A	available	No
<input type="checkbox"/>	open-db-01	mysql	8.0.35	db.t4g.medium	N/A	available	No
<input type="checkbox"/>	qa-db	mysql	8.0.35	db.t3.small	N/A	available	No
<input type="checkbox"/>	xtra-demo-instance-01	mysql	8.0.35	db.t3.small	N/A	available	No
<input type="checkbox"/>	xtra-demo-instance-02	mysql	8.0.35	db.t3.small	N/A	available	No
<input type="checkbox"/>	xtra-demo-instance-03	mysql	8.0.35	db.t3.small	N/A	available	No
<input type="checkbox"/>	xtra-demo-instance-04	mysql	8.0.35	db.t3.small	N/A	available	No
<input type="checkbox"/>	xtra-demo-instance-05	mysql	8.0.35	db.t3.small	N/A	available	No

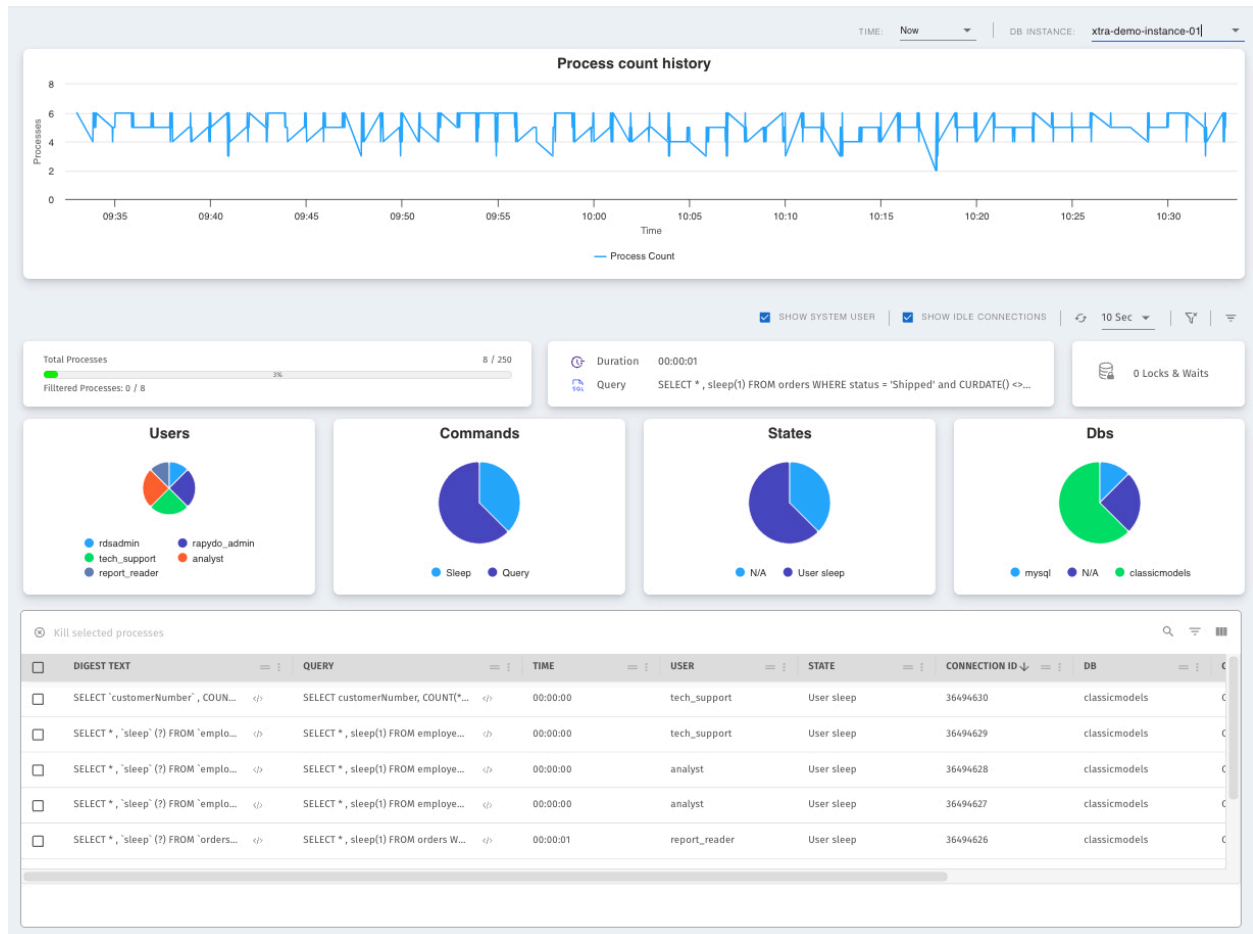
Rows per page 25 1-25 of 33

2. Select the databases to start monitoring and click on the setup button in the right top corner
3. Provide the master user name and password of the RDS instance - Those credentials are not being stored anywhere and used only for a one-time connection to create a Rapydo user in the selected databases.



Note: If selected multiple databases we assume that all of them have the same credentials. If having different credentials those databases will have to be configured one by one.

4. After adding one or more databases to the system we'll be able to monitor them in the `Queries` screen. Change the selected database using the `DB HOST` filter on the right side.



- Click on the layout toggle button in the right top corner in order to move to a table display of all current queries running in the database.