Running Jira on AWS EC2 and Using RDS as the Database

Author: Zayan Ahmed | **Estimated Reading time:** 6 min

This document provides a comprehensive guide to deploying Jira on an AWS EC2 instance and using Amazon RDS for database management. It covers prerequisites, deployment steps, configuration, and best practices.

Prerequisites

Before starting, ensure you have:

- An AWS account.
- Basic knowledge of AWS services such as EC2, RDS, and networking.
- A registered Jira license or access to a trial license.
- SSH client installed for connecting to the EC2 instance.
- Proper IAM permissions to create and manage EC2 and RDS resources.

Step 1: Launch an EC2 Instance for Jira

1. **Choose an AMI**:
   - Use a Linux-based AMI such as Amazon Linux 2, Ubuntu, or RHEL.
   - Ensure the instance has adequate resources (at least t3.medium for small teams).
2. **Configure Instance Details**:

   - Set up the instance in a public or private subnet as per your requirements.
   - Attach a security group allowing:
     - SSH (port 22) for administration.
     - HTTP (port 80) and/or HTTPS (port 443) for Jira access.
3. **Add Storage**:
   - Allocate sufficient storage (e.g., 20 GB) for Jira application files.
4. **Launch and Connect**:
   - Launch the instance and connect using SSH.

Step 2: Install Jira on the EC2 Instance

1. **Update the System**:

```
sudo yum update -y   # For Amazon Linux 2
sudo apt update && sudo apt upgrade -y  # For Ubuntu
```

2. Download Jira:

- Obtain the latest Jira installer from Atlassian's website.

```
wget
https://product-downloads.atlassian.com/software/jira/downloads/atlassian-jira-software-X.Y.Z-x64.bin
```

3. **Make the Installer Executable**:

```
chmod +x atlassian-jira-software-X.Y.Z-x64.bin
```

4. **Run the Installer**:

```
sudo ./atlassian-jira-software-X.Y.Z-x64.bin
```

- Follow the interactive prompts to complete the installation.

5. **Start Jira**:

```
sudo /opt/atlassian/jira/bin/start-jira.sh
```

6. Access Jira:

- Navigate to `http://<EC2-Public-IP>:8080` to access the Jira setup wizard.

Step 3: Launch an RDS Instance for Jira's Database

1. **Choose a Database Engine**:
    - Supported options: PostgreSQL, MySQL, or Oracle. PostgreSQL is recommended.
2. **Configure RDS Instance**:
    - Select the appropriate instance class (e.g., db.t3.medium).
    - Allocate sufficient storage (e.g., 20 GB).
    - Enable Multi-AZ for high availability (optional).
3. **Configure Security**:
    - Set up an RDS security group to allow connections from the EC2 instance (default PostgreSQL port: 5432).
4. **Database Initialization**:
    - Create a database for Jira (e.g., `jira_db`) and a user with necessary privileges.
    - Save the database connection string, username, and password.

Step 4: Configure Jira to Use the RDS Database

1. **Stop Jira Service**:

```
sudo /opt/atlassian/jira/bin/stop-jira.sh
```

2. **Configure `dbconfig.xml`**:

- Edit or create `/opt/atlassian/jira/atlassian-jira/WEB-INF/classes/dbconfig.xml`.

```
<jira-database-config>

    <name>defaultDS</name>

    <type>postgres72</type>

    <database-type>postgresql</database-type>

    <schema-name>public</schema-name>

    <jdbc-datasource>

        <url>jdbc:postgresql://<RDS-Endpoint>:5432/jira_db</url>

        <driver-class>org.postgresql.Driver</driver-class>

        <username>jira_user</username>
        <password>your_password</password>

        <pool-min-size>20</pool-min-size>

        <pool-max-size>20</pool-max-size>

        <pool-max-wait>30000</pool-max-wait>

        <validation-query>select 1</validation-query>

        <validation-query-timeout>3</validation-query-timeout>

    </jdbc-datasource>

</jira-database-config>
```

3. **Start Jira Service**:

```
sudo /opt/atlassian/jira/bin/start-jira.sh
```

4. **Verify Database Connection**:

   - Open Jira in a browser and verify that it connects to the RDS database during the setup wizard.

Step 5: Secure the Setup

1. **Enable HTTPS**:
   - Use a reverse proxy (e.g., Nginx or Apache) to handle HTTPS.
   - Obtain an SSL certificate from AWS Certificate Manager or Let’s Encrypt.
2. **Harden EC2 and RDS Security**:
   - Restrict access to the EC2 instance and RDS using security groups.
   - Enable VPC Flow Logs and CloudTrail for monitoring.
3. **Enable Backups**:
   - Configure automated backups for the RDS instance.
   - Set up periodic snapshots for the EC2 instance.
4. **Monitoring**:
   - Use AWS CloudWatch for EC2 and RDS monitoring.
   - Install Jira monitoring plugins for in-application metrics.

Best Practices

- Use a dedicated IAM role for EC2 with least-privilege permissions.
- Enable Multi-AZ for RDS to ensure high availability.
- Regularly update Jira to the latest version to patch security vulnerabilities.
- Use AWS Systems Manager Session Manager instead of SSH for accessing the EC2 instance.
- Monitor costs and set up AWS Budgets to avoid unexpected expenses.

Troubleshooting

- **Database Connection Errors**:
  - Verify RDS endpoint and security group rules.
  - Check the `dbconfig.xml` file for typos.
- **Jira Performance Issues**:
  - Allocate more CPU and memory to the EC2 instance.
  - Optimize the RDS database by tuning parameters.
- **Access Issues**:
  - Ensure the correct inbound rules in the EC2 security group.
  - Use the public IP or DNS name for accessing Jira.

Conclusion

Deploying Jira on AWS EC2 with RDS as the database provides a scalable, secure, and highly available environment for managing projects and teams effectively. By leveraging the power of EC2 for computing and RDS for database management, you can ensure optimal performance and reliability for your Jira instance. This setup not only separates application and database layers but also enhances flexibility and scalability, making it easier to adapt to growing organizational needs.

Want more ? 🤔
Follow me on LinkedIn 😊