

# 1.0.0 - ami-0e08dcc49a799676a

## GitLab CE on AWS

Self-hosted GitLab Community Edition with automated deployment on AWS.

### Requirements

#### Instance Size

| Instance Type | vCPUs | RAM   | Recommendation               |
|---------------|-------|-------|------------------------------|
| t3.medium     | 2     | 4 GB  | Minimum (up to 10 users)     |
| t3.large      | 2     | 8 GB  | Recommended (up to 50 users) |
| t3.xlarge     | 4     | 16 GB | Production (up to 200 users) |
| m5.xlarge     | 4     | 16 GB | High availability            |

**Note:** GitLab requires a minimum of 4GB RAM. Instances with less memory will experience poor performance or crashes.

### Storage

Attach an EBS volume for persistent data storage:

| Use Case                  | Recommended Size |
|---------------------------|------------------|
| Small team (< 10 users)   | 50 GB            |
| Medium team (10-50 users) | 100 GB           |
| Large team (50-200 users) | 250 GB+          |

Use **gp3** volume type for best price/performance ratio.

### Security Groups

Configure the following inbound rules:

| Port | Protocol | Source    | Description                 |
|------|----------|-----------|-----------------------------|
| 22   | TCP      | Your IP   | SSH access                  |
| 80   | TCP      | 0.0.0.0/0 | HTTP                        |
| 443  | TCP      | 0.0.0.0/0 | HTTPS                       |
| 2222 | TCP      | 0.0.0.0/0 | GitLab SSH (Git operations) |

# Quick Start

## Option 1: Interactive Setup

1. Launch the AMI with an attached EBS volume
2. SSH into your instance:

```
ssh -i your-key.pem ubuntu@your-instance-ip
```

3. Run the configuration wizard:

```
sudo /opt/gitlab/configure-gitlab.sh
```

4. Follow the prompts to set your domain, password, and HTTPS preferences
5. Wait 3-5 minutes for GitLab to initialize
6. Access GitLab at your configured URL

## Option 2: Automated Setup (User Data)

Launch with the following JSON in the EC2 user-data field:

```
{
  "host": "gitlab.example.com",
  "password": "YourSecurePassword123",
  "https": true,
  "email": "admin@example.com",
  "ssh_port": 2222
}
```

### Parameters:

| Parameter | Required | Default | Description |
|-----------|----------|---------|-------------|
|-----------|----------|---------|-------------|

|          |               |                |                                     |
|----------|---------------|----------------|-------------------------------------|
| host     | No            | Auto-detect IP | Domain name or IP address           |
| password | Yes           | -              | Root account password (min 8 chars) |
| https    | No            | false          | Enable HTTPS with Let's Encrypt     |
| email    | If https=true | -              | Email for Let's Encrypt             |
| ssh_port | No            | 2222           | Port for Git SSH operations         |

### Example for HTTP-only setup:

```
{  
  "password": "MySecurePassword123"  
}
```

# Accessing GitLab

## Web Interface

After configuration completes:

1. Open your browser to `http://your-host` or `https://your-host`
2. Log in with:
  - Username: `root`
  - Password: The password you configured

**Note:** GitLab takes 3-5 minutes to fully initialize on first boot. If you see a 502 error, wait and refresh.

## Git Operations via SSH

Configure your SSH to use the GitLab SSH port:

Add to `~/.ssh/config`:

```
Host gitlab.example.com  
  Port 2222  
  User git  
  IdentityFile ~/.ssh/your-key
```

Then clone repositories:

```
git clone git@gitlab.example.com:username/repo.git
```

Or use the full URL format:

```
git clone ssh://git@gitlab.example.com:2222/username/repo.git
```

# Management Commands

The `gitlab-cli` utility provides easy management:

```
# Check status
gitlab-cli status

# View logs
gitlab-cli logs
gitlab-cli logs gitlab

# Start/stop/restart
gitlab-cli start
gitlab-cli stop
gitlab-cli restart

# View configuration info
gitlab-cli info

# Update to latest version
gitlab-cli update

# Create backup
gitlab-cli backup

# Restore from backup
gitlab-cli restore /path/to/backup.tar

# Reset root password
gitlab-cli reset-password

# Run health checks
```

```
gitlab-cli health

# Open Rails console
gitlab-cli shell

# Run Rake tasks
gitlab-cli rake gitlab:check
```

# HTTPS Configuration

## With Let's Encrypt (Recommended)

During interactive setup, select "Yes" when asked about HTTPS. Requirements:

1. A valid domain name (not IP address)
2. DNS pointing to your instance's public IP
3. Ports 80 and 443 open in security group

Certificates are automatically obtained and renewed.

## Manual SSL Certificate

To use your own certificate:

1. Place certificate files in `/mnt/gitlab-data/certs/`
2. Edit `/opt/gitlab/docker-compose.yml` to mount certificates
3. Update GitLab configuration for SSL
4. Run `gitlab-cli restart`

# Backup and Recovery

## Creating Backups

```
# Create a backup
sudo gitlab-cli backup
```

Backups are stored in `/mnt/gitlab-data/data/backups/`

# Automated Backups

Add a cron job for daily backups:

```
sudo crontab -e
```

Add:

```
0 2 * * * /usr/local/bin/gitlab-cli backup >> /var/log/gitlab-backup.log 2>&1
```

# Offsite Backup to S3

```
# Install AWS CLI (already installed)
aws s3 cp /mnt/gitlab-data/data/backups/ s3://your-bucket/gitlab-backups/ --recursive
```

# Restoring from Backup

```
# List available backups
ls /mnt/gitlab-data/data/backups/

# Restore
sudo gitlab-cli restore /mnt/gitlab-
data/data/backups/gitlab_backup_TIMESTAMP_gitlab_backup.tar
```

# Data Storage

All GitLab data is stored on your EBS volume:

| Path                    | Contents                         |
|-------------------------|----------------------------------|
| /mnt/gitlab-data/config | GitLab configuration             |
| /mnt/gitlab-data/data   | Repositories, uploads, artifacts |
| /mnt/gitlab-data/logs   | GitLab logs                      |
| /mnt/gitlab-data/certs  | SSL certificates                 |

# Performance Tuning

For larger instances, edit `/mnt/gitlab-data/config/gitlab.rb`:

```
# Increase workers for more concurrent requests
puma['worker_processes'] = 4

# Increase Sidekiq for background jobs
sidekiq['concurrency'] = 20

# Enable Prometheus monitoring
prometheus_monitoring['enable'] = true
```

Then apply changes:

```
docker exec gitlab gitlab-ctl reconfigure
```

# Troubleshooting

## GitLab Shows 502 Error

This is normal during startup. GitLab takes 3-5 minutes to fully initialize.

Check startup progress:

```
gitlab-cli logs
```

## Container Won't Start

Check Docker status:

```
sudo systemctl status docker
sudo docker ps -a
```

Check logs:

```
gitlab-cli logs
```

## Out of Memory

GitLab requires minimum 4GB RAM. Check memory:

```
free -h
```

If memory is low, consider upgrading instance type or reducing workers in gitlab.rb.

## EBS Volume Not Detected

Verify the volume is attached:

```
lsblk
```

Manually mount if needed:

```
sudo mount /dev/nvme1n1 /mnt/gitlab-data
```

## SSL Certificate Issues

Check certificate status:

```
docker logs acme-companion
```

Ensure DNS is properly configured and propagated:

```
dig +short your-domain.com
```

## Reset Root Password

```
sudo gitlab-cli reset-password
```

Or via Rails console:

```
docker exec -it gitlab gitlab-rails console
```

Then in console:

```
user = User.find_by(username: 'root')
user.password = 'NewPassword123'
user.password_confirmation = 'NewPassword123'
user.save!
exit
```

# Updating GitLab

```
# Check current version
docker exec gitlab cat /opt/gitlab/embedded/service/gitlab-rails/VERSION

# Update to latest
sudo gitlab-cli update
```

**Important:** Always backup before updating!

## Support

For GitLab documentation, visit: <https://docs.gitlab.com>

For issues with this AMI, contact the publisher through AWS Marketplace.

---

Revision #2

Created 2026-01-19 22:51:15 UTC by Admin

Updated 2026-01-19 22:55:20 UTC by Admin