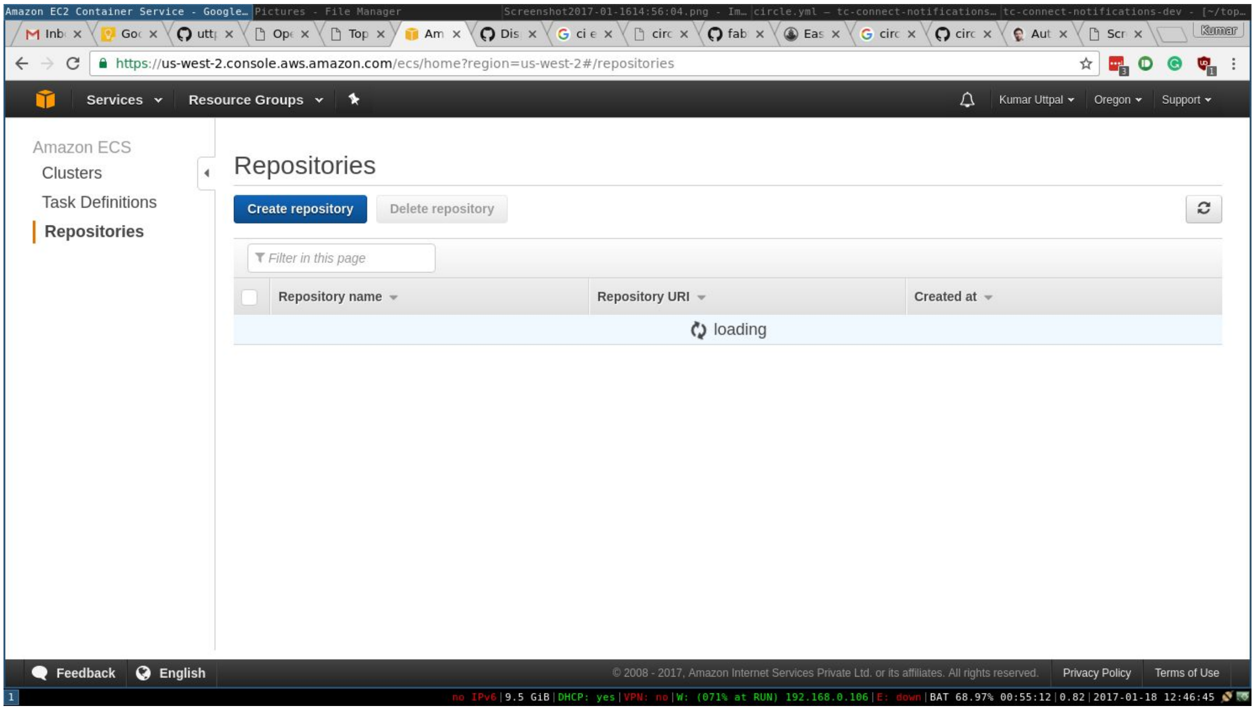
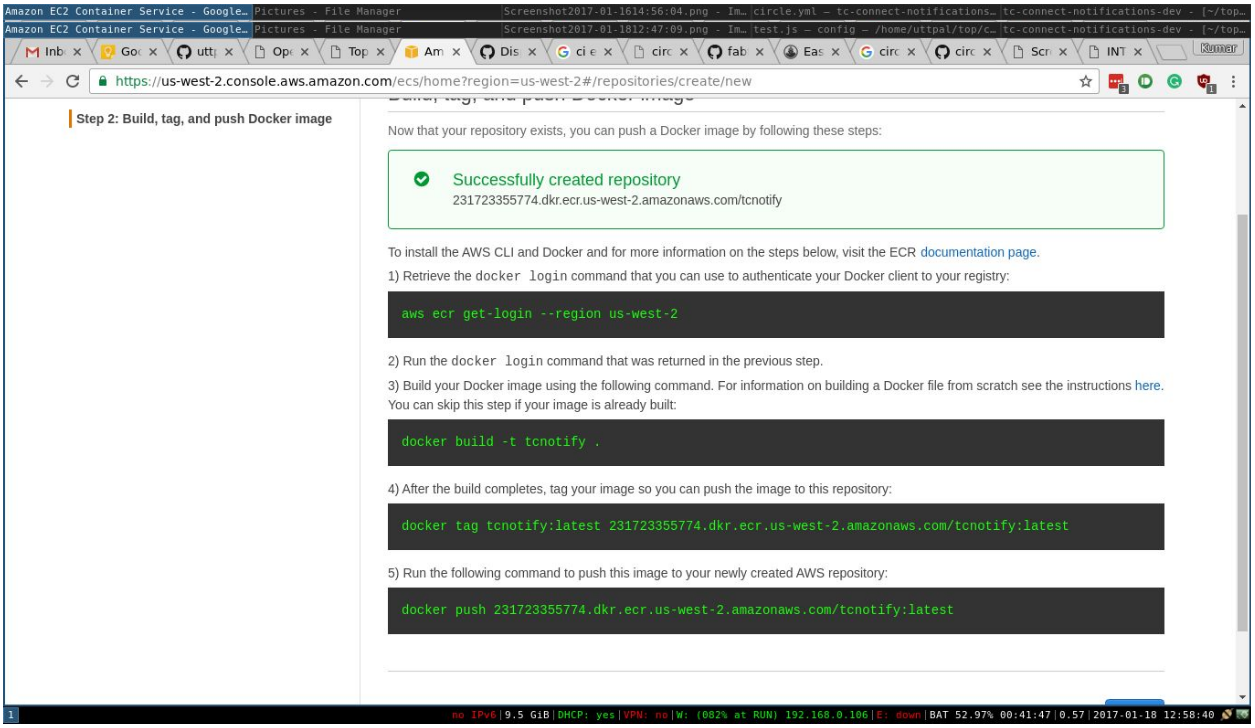
1> Login to your aws console > ecs > create repository



2> enter repository name


https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/repositories/create/new
Step 2: Build, tag, and push Docker image

Now that your repository exists, you can push a Docker image by following these steps:

Successfully created repository
231723355774.dkr.ecr.us-west-2.amazonaws.com/tcnotify

To install the AWS CLI and Docker and for more information on the steps below, visit the ECR documentation page.

1) Retrieve the `docker login` command that you can use to authenticate your Docker client to your registry:

```
aws ecr get-login --region us-west-2
```

2) Run the `docker login` command that was returned in the previous step.

3) Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:

```
docker build -t tcnotify .
```

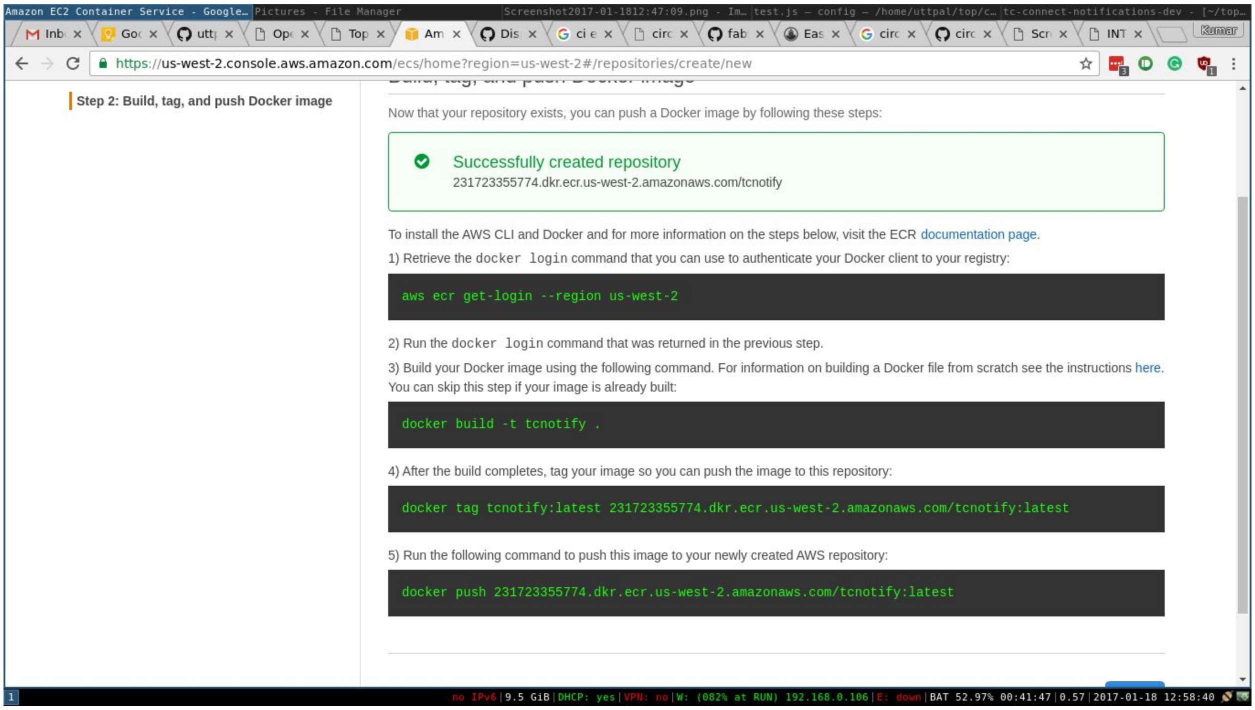
4) After the build completes, tag your image so you can push the image to this repository:

```
docker tag tcnotify:latest 231723355774.dkr.ecr.us-west-2.amazonaws.com/tcnotify:latest
```

5) Run the following command to push this image to your newly created AWS repository:

```
docker push 231723355774.dkr.ecr.us-west-2.amazonaws.com/tcnotify:latest
```

3 Repository is created successfully , note account no. (2317233…) aws region



https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/repositories/create/new

Step 2: Build, tag, and push Docker image

Now that your repository exists, you can push a Docker image by following these steps:

Successfully created repository
231723355774.dkr.ecr.us-west-2.amazonaws.com/tcnotify

To install the AWS CLI and Docker and for more information on the steps below, visit the ECR documentation page.

1) Retrieve the `docker login` command that you can use to authenticate your Docker client to your registry:

```
aws ecr get-login --region us-west-2
```

2) Run the `docker login` command that was returned in the previous step.

3) Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:

```
docker build -t tcnotify .
```

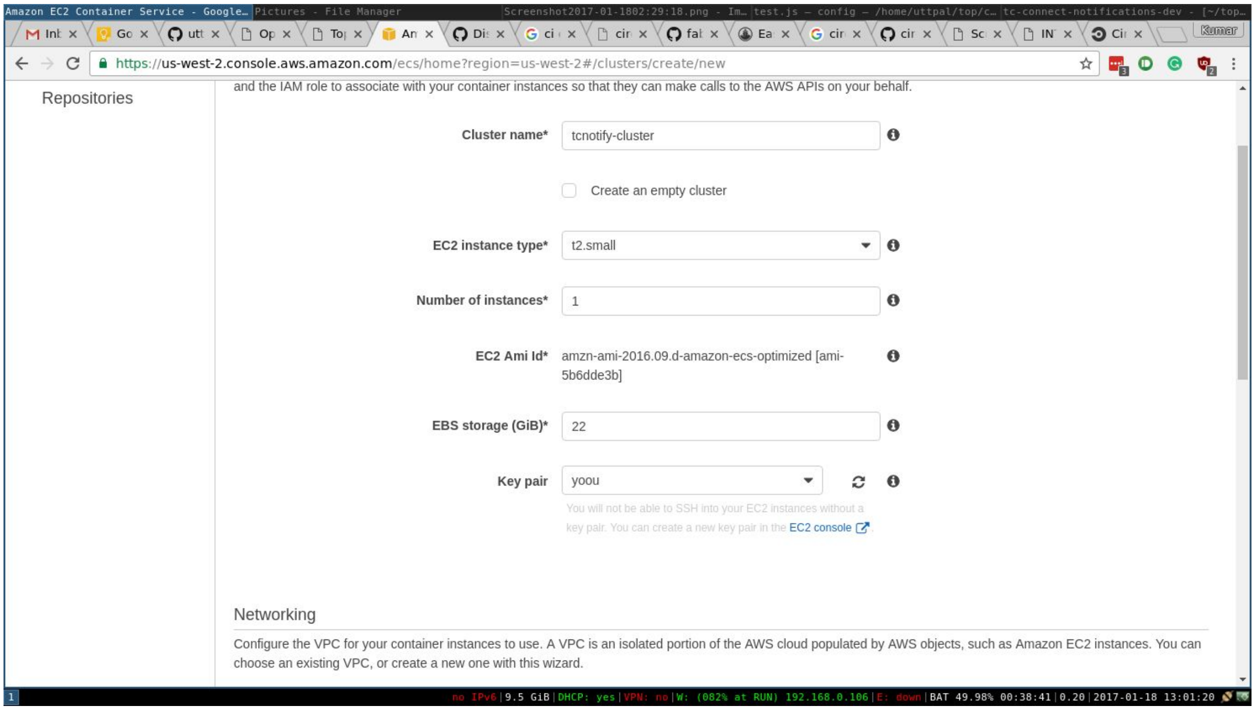
4) After the build completes, tag your image so you can push the image to this repository:

```
docker tag tcnotify:latest 231723355774.dkr.ecr.us-west-2.amazonaws.com/tcnotify:latest
```

5) Run the following command to push this image to your newly created AWS repository:

```
docker push 231723355774.dkr.ecr.us-west-2.amazonaws.com/tcnotify:latest
```

4> goto ecs> cluster> new cluster> enter cluster name, no. of instance and ssh key….



https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/clusters/create/new

Repositories

and the IAM role to associate with your container instances so that they can make calls to the AWS APIs on your behalf.

Cluster name* tcnotify-cluster

□ Create an empty cluster

EC2 instance type* t2.small

Number of instances* 1

EC2 Ami Id* amzn-ami-2016.09.d-amazon-ecs-optimized [ami-5b6dde3b]

EBS storage (GiB)* 22

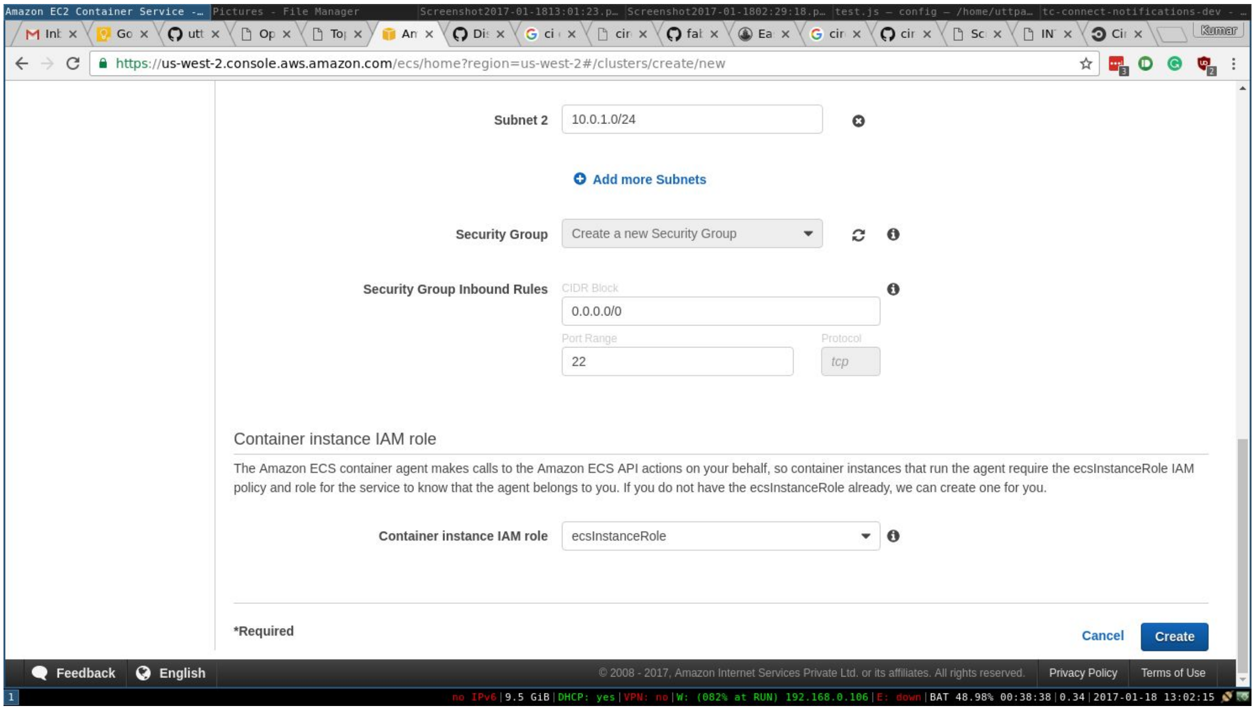
Key pair yoou

You will not be able to SSH into your EC2 instances without a key pair. You can create a new key pair in the EC2 console.

Networking

Configure the VPC for your container instances to use. A VPC is an isolated portion of the AWS cloud populated by AWS objects, such as Amazon EC2 instances. You can choose an existing VPC, or create a new one with this wizard.

5> now create.


https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/clusters/create/new

Subnet 2: 10.0.1.0/24

Add more Subnets

Security Group: Create a new Security Group

Security Group Inbound Rules: CIDR Block 0.0.0.0/0, Port Range 22, Protocol tcp

Container instance IAM role

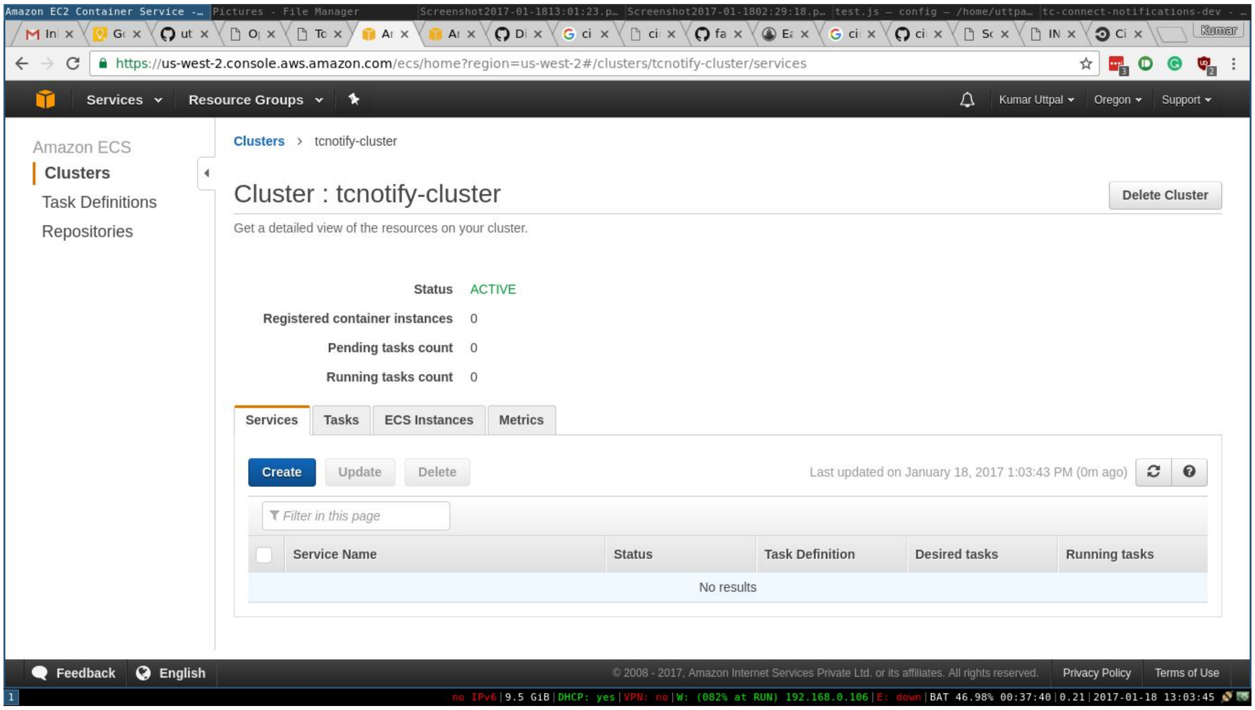
The Amazon ECS container agent makes calls to the Amazon ECS API actions on your behalf, so container instances that run the agent require the ecsInstanceRole IAM policy and role for the service to know that the agent belongs to you. If you do not have the ecsInstanceRole already, we can create one for you.

Container instance IAM role: ecsInstanceRole

*Required

Cancel
Create
Feedback
English
© 2008 - 2017, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

6> After creation view cluster and add new service


https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/clusters/tcnotify-cluster/services

Cluster : tcnotify-cluster

Get a detailed view of the resources on your cluster.

Status	ACTIVE
Registered container instances	0
Pending tasks count	0
Running tasks count	0

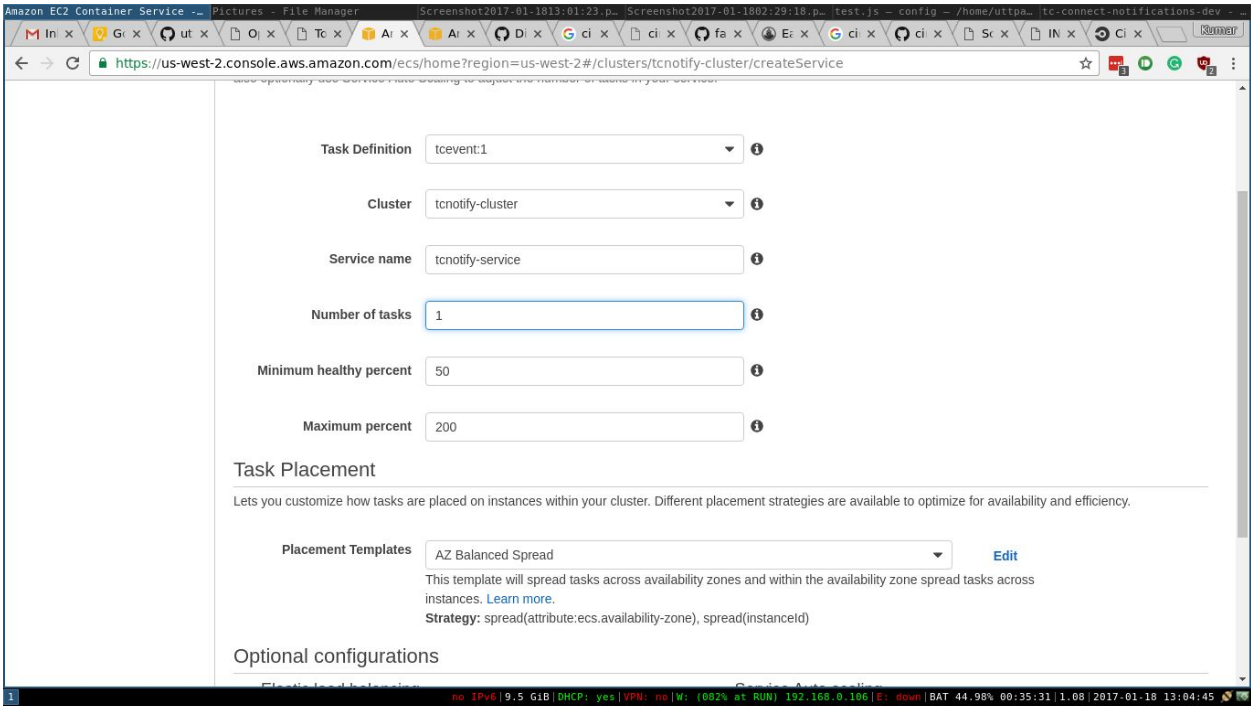
Last updated on January 18, 2017 1:03:43 PM (0m ago)

Filter in this page

Service Name	Status	Task Definition	Desired tasks	Running tasks
No results				

Feedback
English
© 2008 - 2017, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

7> Enter service name and create


Amazon EC2 Container Service -…
Pictures - File Manager
Screenshot2017-01-1813:01:23.p…
Screenshot2017-01-1802:29:18.p…
test.js — config — /home/uttpa…
tc-connect-notifications-dev - …
Kumar
https://us-west-2.console.aws.amazon.com/ecs/home?region=us-west-2#/clusters/tcnotify-cluster/createService
no IPv6 | 9.5 GiB | DHCP: yes | VPN: no | W: (082% at RUN) 192.168.0.106 | E: down | BAT 44.98% 00:35:31 | 1.08 | 2017-01-18 13:04:45

Task Definition: tcevent:1

Cluster: tcnotify-cluster

Service name: tcnotify-service

Number of tasks: 1

Minimum healthy percent: 50

Maximum percent: 200

Task Placement

Lets you customize how tasks are placed on instances within your cluster. Different placement strategies are available to optimize for availability and efficiency.

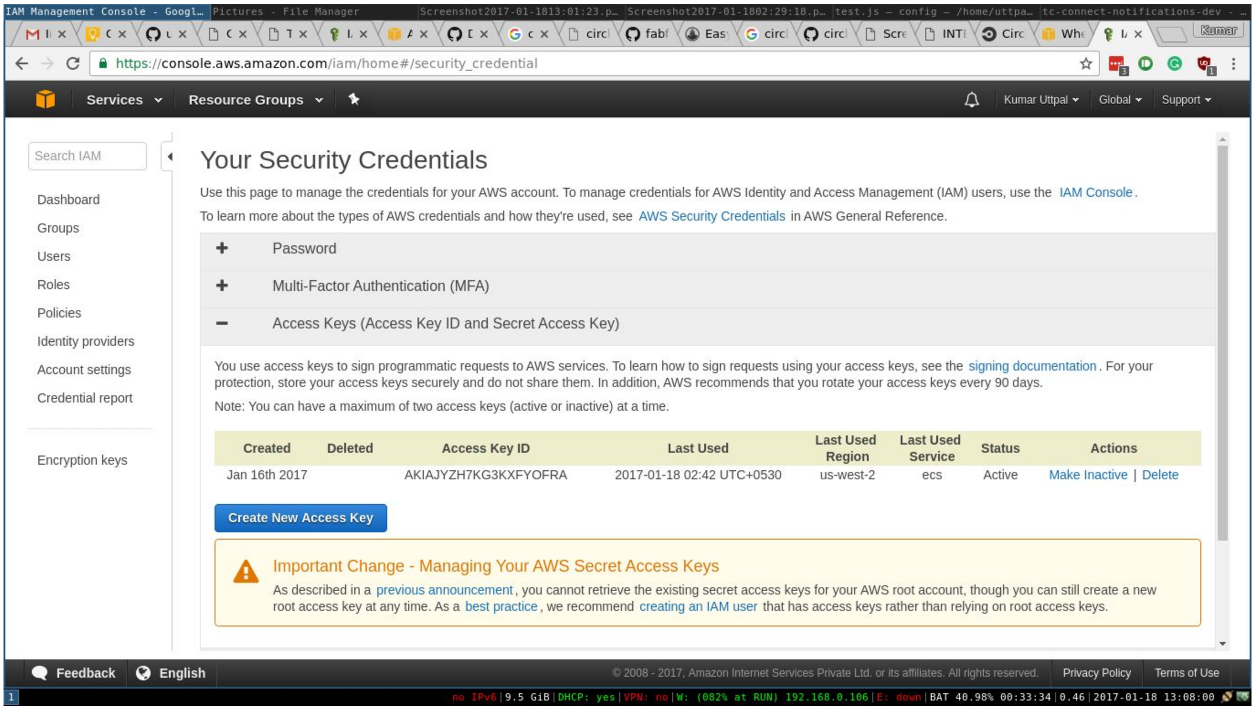
Placement Templates: AZ Balanced Spread — Edit

This template will spread tasks across availability zones and within the availability zone spread tasks across instances. Learn more.

Strategy: spread(attribute:ecs.availability-zone), spread(instanceId)

Optional configurations

8> Create aws credentials iam> manage security credential > add new


IAM Management Console - Googl…
Pictures - File Manager
Screenshot2017-01-1813:01:23.p…
Screenshot2017-01-1802:29:18.p…
test.js — config — /home/uttpa…
tc-connect-notifications-dev - …
Kumar
https://console.aws.amazon.com/iam/home#/security_credential
Services
Resource Groups
Kumar Uttpal
Global
Support
Search IAM
Dashboard
Groups
Users
Roles
Policies
Identity providers
Account settings
Credential report
Encryption keys
Feedback
English
© 2008 - 2017, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
no IPv6 | 9.5 GiB | DHCP: yes | VPN: no | W: (082% at RUN) 192.168.0.106 | E: down | BAT 40.98% 00:33:34 | 0.46 | 2017-01-18 13:08:00

Your Security Credentials

Use this page to manage the credentials for your AWS account. To manage credentials for AWS Identity and Access Management (IAM) users, use the IAM Console.

To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials in AWS General Reference.

+ Password

+ Multi-Factor Authentication (MFA)

− Access Keys (Access Key ID and Secret Access Key)

You use access keys to sign programmatic requests to AWS services. To learn how to sign requests using your access keys, see the signing documentation. For your protection, store your access keys securely and do not share them. In addition, AWS recommends that you rotate your access keys every 90 days.

Note: You can have a maximum of two access keys (active or inactive) at a time.

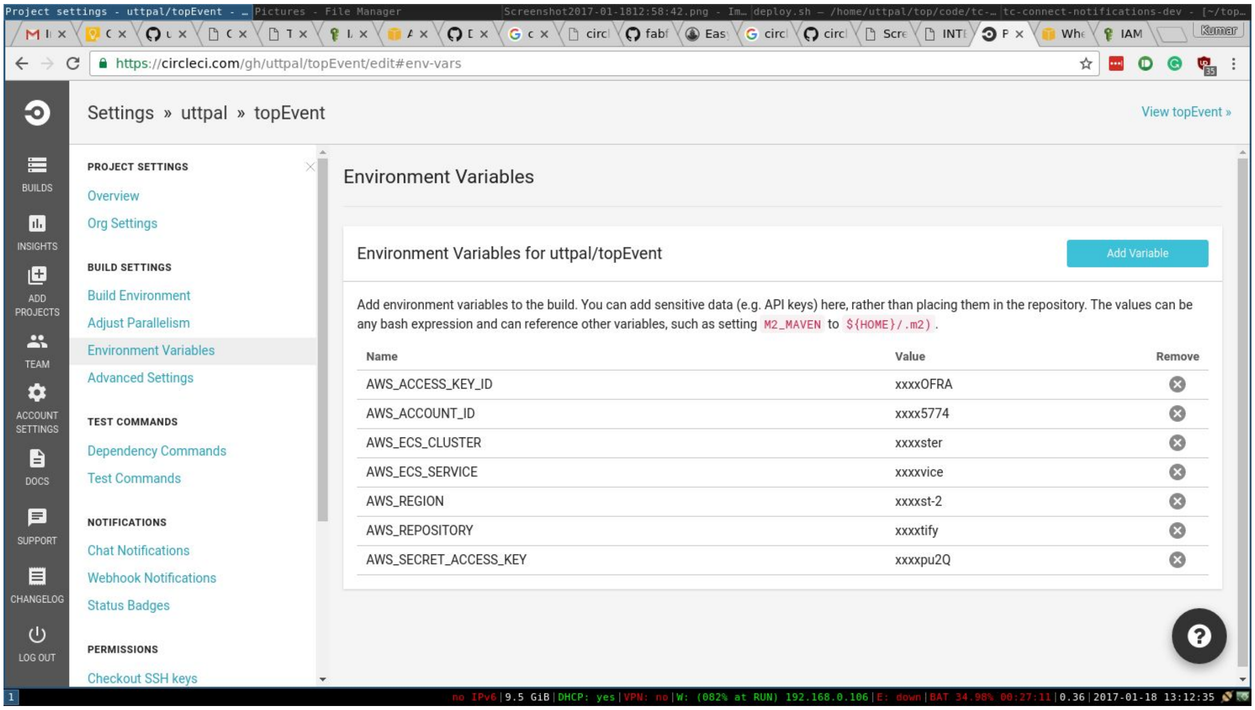
Created	Deleted	Access Key ID	Last Used	Last Used Region	Last Used Service	Status	Actions
Jan 16th 2017		AKIAJYZH7KG3KXFYOFRA	2017-01-18 02:42 UTC+0530	us-west-2	ecs	Active	Make Inactive | Delete

Create New Access Key

Important Change - Managing Your AWS Secret Access Keys

As described in a previous announcement, you cannot retrieve the existing secret access keys for your AWS root account, though you can still create a new root access key at any time. As a best practice, we recommend creating an IAM user that has access keys rather than relying on root access keys.

9> Go to circleci dashboard > projectsettings > environment variable> add all parameters as environment variable



https://circleci.com/gh/uttpal/topEvent/edit#env-vars

Settings » uttpal » topEvent

View topEvent »

PROJECT SETTINGS

- Overview
- Org Settings

BUILD SETTINGS

- Build Environment
- Adjust Parallelism
- Environment Variables
- Advanced Settings

TEST COMMANDS

- Dependency Commands
- Test Commands

NOTIFICATIONS

- Chat Notifications
- Webhook Notifications
- Status Badges

PERMISSIONS

- Checkout SSH keys

Environment Variables

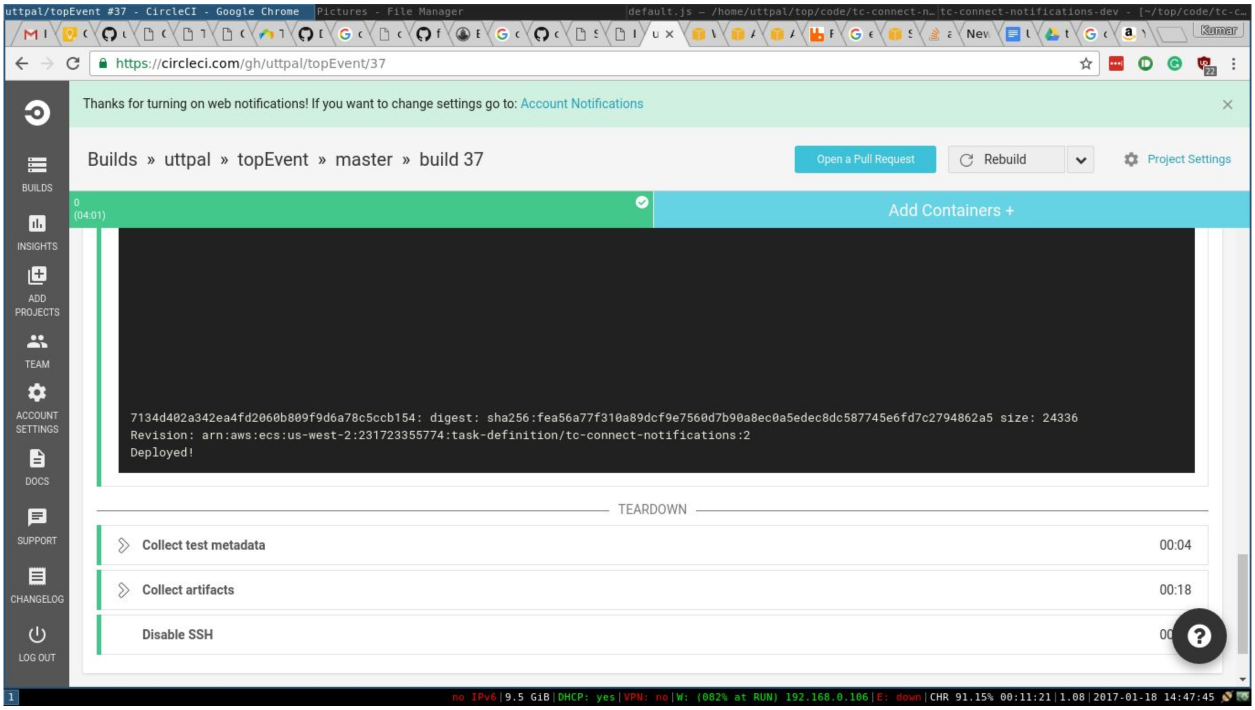
Environment Variables for uttpal/topEvent

Add environment variables to the build. You can add sensitive data (e.g. API keys) here, rather than placing them in the repository. The values can be any bash expression and can reference other variables, such as setting `M2_MAVEN` to `${HOME}/.m2)`.

Name	Value	Remove
AWS_ACCESS_KEY_ID	xxxxOFRA	⊗
AWS_ACCOUNT_ID	xxxx5774	⊗
AWS_ECS_CLUSTER	xxxxster	⊗
AWS_ECS_SERVICE	xxxxvice	⊗
AWS_REGION	xxxxst-2	⊗
AWS_REPOSITORY	xxxxtify	⊗
AWS_SECRET_ACCESS_KEY	xxxxpu2Q	⊗

```
no IPv6|9.5 GiB|DHCP: yes|VPN: no|W: (082% at RUN) 192.168.0.106|E: down|BAT 34.98% 00:27:11|0.36|2017-01-18 13:12:35
```

10> push your commit to master branch > bulid will pass(if npm test fail due to timeout then retry).



https://circleci.com/gh/uttpal/topEvent/37

Thanks for turning on web notifications! If you want to change settings go to: Account Notifications

Builds » uttpal » topEvent » master » build 37

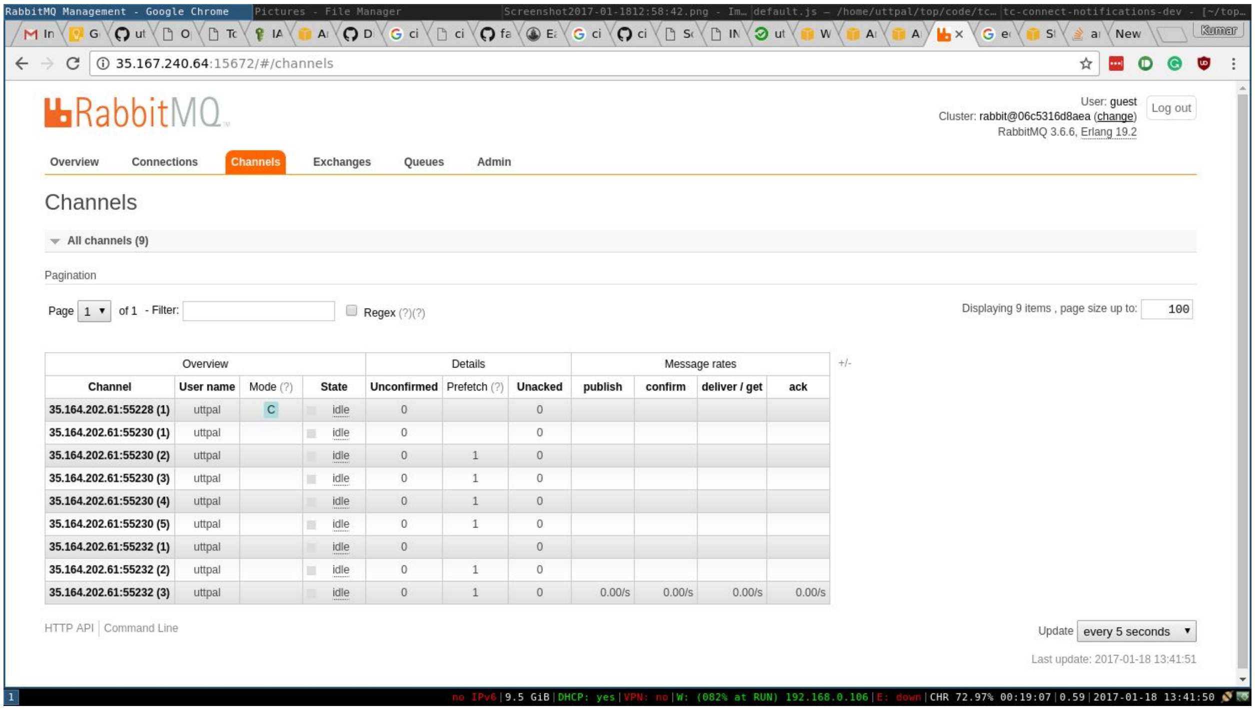
```
7134d402a342ea4fd2060b809f9d6a78c5ccb154: digest: sha256:fea56a77f310a89dcf9e7560d7b90a8ec0a5edec8dc587745e6fd7c2794862a5 size: 24336
Revision: arn:aws:ecs:us-west-2:231723355774:task-definition/tc-connect-notifications:2
Deployed!
```

TEARDOWN

Collect test metadata	00:04
Collect artifacts	00:18
Disable SSH	00

```
no IPv6|9.5 GiB|DHCP: yes|VPN: no|W: (082% at RUN) 192.168.0.106|E: down|CHR 91.15% 00:11:21|1.08|2017-01-18 14:47:45
```

11> go to aws console and see task running as well as connections created in rabbitmq from ec2 instance



RabbitMQ Management - Google Chrome
Pictures - File Manager
Screenshot2017-01-1812:58:42.png - Im…
default.js — /home/uttpal/top/code/tc…
tc-connect-notifications-dev - [~/top…
35.167.240.64:15672/#/channels
RabbitMQ
Log out
Overview
Connections
Channels
Exchanges
Queues
Admin

User: guest
Cluster: rabbit@06c5316d8aea (change)
RabbitMQ 3.6.6, Erlang 19.2

Channels

All channels (9)

Pagination

Page 1 ▾ of 1 - Filter: □ Regex (?)(?)

Displaying 9 items , page size up to: 100

Overview				Details			Message rates			
Channel	User name	Mode (?)	State	Unconfirmed	Prefetch (?)	Unacked	publish	confirm	deliver / get	ack
35.164.202.61:55228 (1)	uttpal	C	idle	0		0				
35.164.202.61:55230 (1)	uttpal		idle	0		0				
35.164.202.61:55230 (2)	uttpal		idle	0	1	0				
35.164.202.61:55230 (3)	uttpal		idle	0	1	0				
35.164.202.61:55230 (4)	uttpal		idle	0	1	0				
35.164.202.61:55230 (5)	uttpal		idle	0	1	0				
35.164.202.61:55232 (1)	uttpal		idle	0		0				
35.164.202.61:55232 (2)	uttpal		idle	0	1	0				
35.164.202.61:55232 (3)	uttpal		idle	0	1	0	0.00/s	0.00/s	0.00/s	0.00/s

+/-

HTTP API | Command Line

Update every 5 seconds ▾

Last update: 2017-01-18 13:41:51

```
no IPv6|9.5 GiB|DHCP: yes|VPN: no|W: (082% at RUN) 192.168.0.106|E: down|CHR 72.97% 00:19:07|0.59|2017-01-18 13:41:50
```