# Case Study Test

**1.** Write a Dockerfile to run Bitcoin 0.21.0 in a container. It should somehow verify the checksum of the downloaded release (there's no need to build the project), run as a normal user, and when run without any modifiers (i.e. docker run something/bitcoin:0.21.0) should run the daemon, and print its output to the console. The build should be security conscious (you can use Prisma, grype, etc.)

**2.** Using terraform write an ECS task definition to run the above. Remember to use persistent volumes, ideally EFS.

**3.** Write a simple pipeline to build and deploy the above. If you like groovy you can write a Jenkinsfile, if not you can use Github Actions or Circle CI

**4.** According to AWS well-architected framework, please provide feedback about the following code:

```
resource "aws_vpc" "main" {

cidr_block = "10.88.40.0/15 ( http://10.88.40.0/15 )"

instance_tenancy = "dedicated"

tags = {

Name = "main"

}

}
```

**5.** Write a terraform module that creates the following resources on IAM:

- A role, with no permissions, which can be assumed by users within the same account

- A policy, allowing users / entities to assume the above role

- A group, with the above policy attached

- A user, belonging to the above group

Entities should have the same name or be similarly named in some meaningful way such as: btc-prod-role, btc-prod-policy, btc-prod-group, btc-prod-user. Extra points if you make the suffixes toggleable.

**6.** How would you provision runtime parameters such as database connection strings, credentials into Docker containers so they can be used in different environments (development, staging, production)? Please take into account that parameters might change often as the development team adds new options or changes existing ones.

**7.** Imagine you have an EC2 instance running in production and for some reason you and your team don't have access to the PEM key configured on it. What other options do you have to connect to that EC2 instance over SSH?