1. Launching ec2 instances for elasticsearch cluster using ansible playbook with bootstrap script to install elasticsearch on all nodes.
I have uploaded cloudformation template to launch an ec2 instance. I have tried the same through ansible playbook because of instructions given.

```
[ansible@ip-172-31-15-10 ~]$ cat inventory.yaml
[localhost]
local
[ansible@ip-172-31-15-10 ~]$
[ansible@ip-172-31-15-10 ~]$ ansible-playbook -i inventory.yaml launch-ec2-ansible.ya
ml
```

```
[ansible@ip-172-31-15-10 ~]$ ansible-playbook -i inventory.yaml launch-ec2-ansible.ya
ml

PLAY [provisioning EC2 instances using Ansible] ************************************

TASK [Task1 - Create security group] ************************************************
changed: [local]

TASK [Task2 Launch EC2 Instance for elasticsearch cluster] **************************
changed: [local]

PLAY RECAP **************************************************************************
local                      : ok=2    changed=2    unreachable=0    failed=0    skippe
d=0    rescued=0    ignored=0

[ansible@ip-172-31-15-10 ~]$
```

2. created elasticsearch role to deploy elasticsearch cluster

```
[ansible@ip-172-31-15-10 ~]$ cd /etc/ansible/roles/elasticsearch/
[ansible@ip-172-31-15-10 elasticsearch]$ ls -lrt
total 4
-rw-rw-r-- 1 ansible ansible 1328 Nov 12 09:29 README.md
drwxrwxr-x 2 ansible ansible   22 Nov 12 09:29 handlers
drwxrwxr-x 2 ansible ansible   39 Nov 12 09:29 tests
drwxrwxr-x 2 ansible ansible   22 Nov 12 09:29 meta
drwxrwxr-x 2 ansible ansible   22 Nov 12 09:29 vars
drwxrwxr-x 2 ansible ansible   31 Nov 13 16:12 templates
drwxrwxr-x 2 ansible ansible   68 Nov 13 16:23 files
drwxrwxr-x 2 ansible ansible   22 Nov 13 16:39 defaults
drwxrwxr-x 2 ansible ansible   22 Nov 13 16:50 tasks
[ansible@ip-172-31-15-10 elasticsearch]$
```

3. implement passwordless authentication for newly created instances in step1 or configure ansible dynamic inventory to access aws ec2 instances.
Note: I am using 1st option in this test.

```
[ansible@ip-172-31-15-10 ~]$ cat inventory.yaml
#[localhost]
#local
[masternode]
15.207.110.214
[datanode1]
3.108.215.194
[datanode2]
13.233.117.112
```

4. configure elasticsearch cluster using ansible elasticsearch roles.

```
[ansible@ip-172-31-15-10 ~]$ ansible-playbook -i inventory.yaml elasticsearch-deply.y
ml

PLAY RECAP *********************************************************************
13.233.117.112             : ok=4    changed=3    unreachable=0    failed=0    skippe
d=0    rescued=0    ignored=0
15.207.110.214             : ok=4    changed=2    unreachable=0    failed=0    skippe
d=0    rescued=0    ignored=0
3.108.215.194              : ok=4    changed=3    unreachable=0    failed=0    skippe
d=0    rescued=0    ignored=0

[ansible@ip-172-31-15-10 ~]$
```

5. demonstrating that elasticsearch is functioning,

```
[root@ip-172-31-13-192 ~]# curl localhost:9200/_cat/nodes?v
ip           heap.percent ram.percent cpu load_1m load_5m load_15m node.role master
name
172.31.13.192           22          93   6    0.02    0.11     0.06 im        *
masternode
[root@ip-172-31-13-192 ~]# curl -XGET '15.207.110.214:9200/_cluster/health?pretty'
{
  "cluster_name" : "elasticsearch-cluster",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 1,
  "number_of_data_nodes" : 0,
  "active_primary_shards" : 0,
  "active_shards" : 0,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}
[root@ip-172-31-13-192 ~]#
```

6. Implementing basic security for elasticsearch manually,
encrypt the network of elasticsearch cluster
Enabled user authentication in elasticsearch
set build-in-elasticsearch user password using following command,
/usr/share/elasticsearch/bin/elasticsearch-setup-passwords auto –verbose

above manual tasks are automated by adding a task in ansible playbook. You can find the same in ansible playbook uploaded on github.

Note: Have referred
https://www.elastic.co/guide/en/elasticsearch/reference/current/security-basic-setup.html

I have invested my weekend to solve this test which can be quantified as approx 36 hrs.

1. **What did you choose to automate the provisioning and bootstrapping of the instance? Why?**
I have automated ec2 instance provisioning and bootstrapping of the instance using cloudformation templates due to being its aws native service and easy to configure.
I have also achieved the same using ansible playbooks because Ansible or Terraform was preferred language for this test.

2. **How did you choose to secure ElasticSearch? Why?**
I have used build-in elasticsearch xpack functionality to configure basic minimum security for Elasticsearch.
As per my knowledge, No matter what technology we are working with, we always need to be mindful of security. Big data platforms are certainly no exception, as they can contain massive amounts of sensitive data that must be protected. Elasticsearch has made securing your cluster very easy with native security configurations and tools to ensure that your data is only accessible to authorized users.

3. **How would you monitor this instance? What metrics would you monitor?**
I can use cloutwatch aws service to monitor elasticsearch instance. CPUUtilization, ClusterStatus, JVMMemoryPressure metrics can be used for monitoring.
Adding to it, we can deploy kibana to monitor elasticsearch instance.

4. **Could you extend your solution to launch a secure cluster of ElasticSearch nodes? What would need to change to support this use case?**
Have already prepared a solution with security configuration. Just need to find a way to automate “setup build-in user password” task.

5. **Could you extend your solution to replace a running ElasticSearch instance with little or no downtime? How?**
Yes Its possible. I can configure ASG and prepare a lunch template for elasticsearch instance creation.

6. **Was it a priority to make your code well structured, extensible, and reusable?**
Of course yes. Its devOps practice to automate as much as possible with well structured, extensible and reusable code.

7. **What sacrifices did you make due to time?**
I don’t call it as sacrifice rather it was a mutual requirement for me and organization to hire best candidate. Adding to it, I could push my limits to provision ec2 using ansible. Prior this I was preferring cloudformationm for the same.