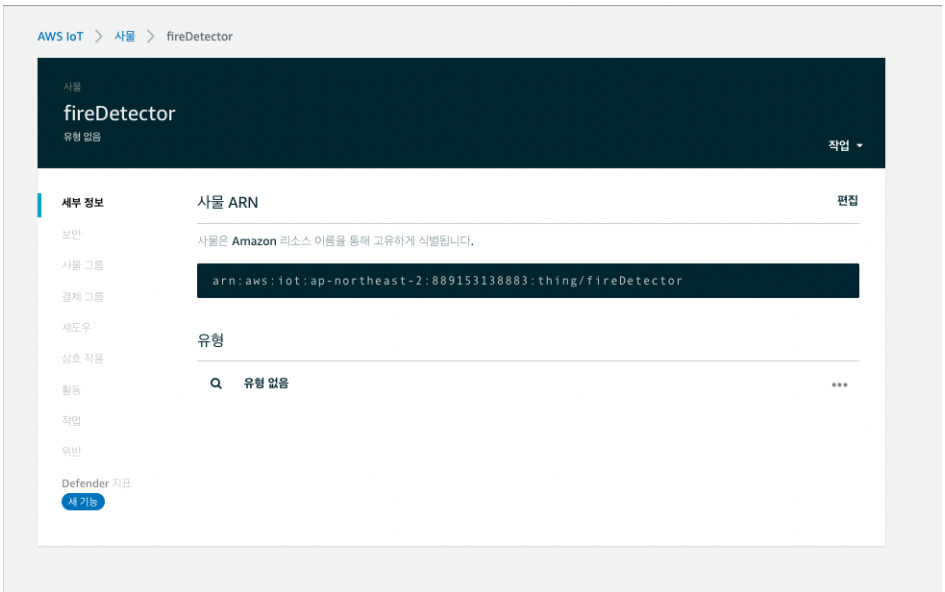
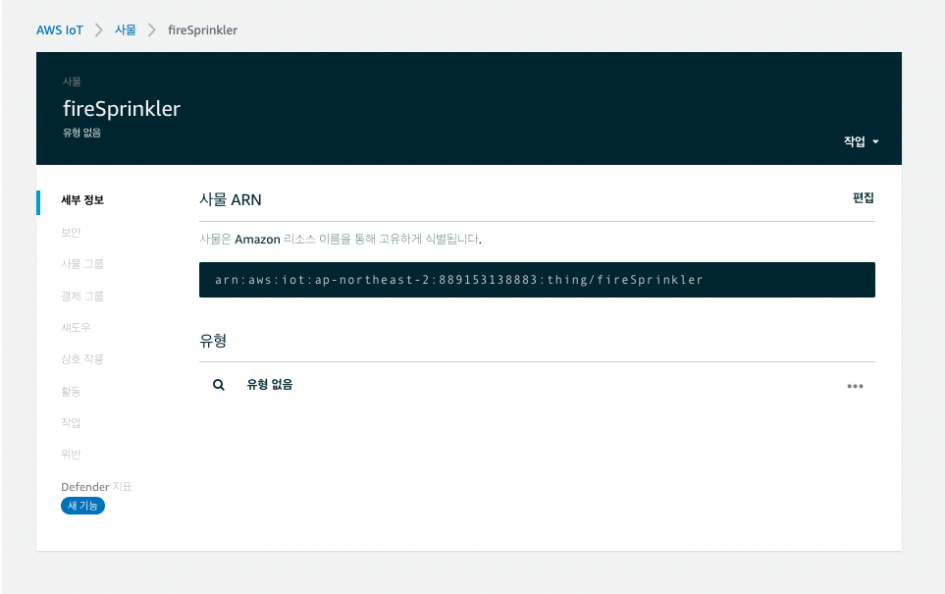
HW4보고서

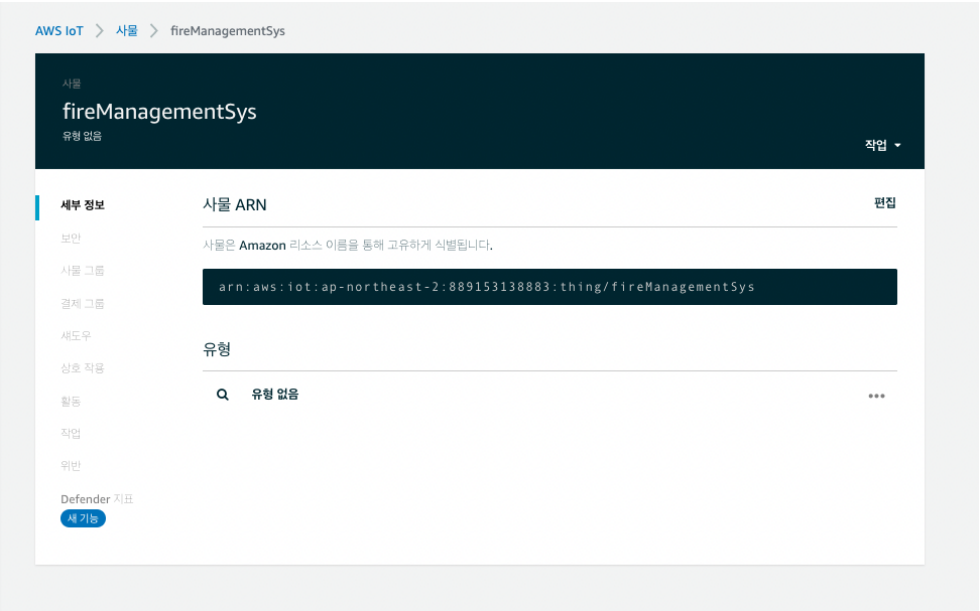
컴퓨터공학부 201814121 이연희

1. AWS IoT Core Applications

1) Fire Detector, Fire Sprinkler와 Fire Management System 사물을 생성했다.







2) 각 사물에 해당하는 정책을 생성하고 연결시켰다.

- policy4fireDetector

버전1 정책 문서 편집

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:client/fire_detector"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Publish",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/alarm"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topicfilter/fire/alert"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/alert"
    }
  ]
}
```

- policy4fireSprinkler

버전1 정책 문서 편집

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:client/fire_sprinkler"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topicfilter/fire/sprinkler"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/sprinkler"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topicfilter/fire/alert"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/alert"
    }
  ]
}
```

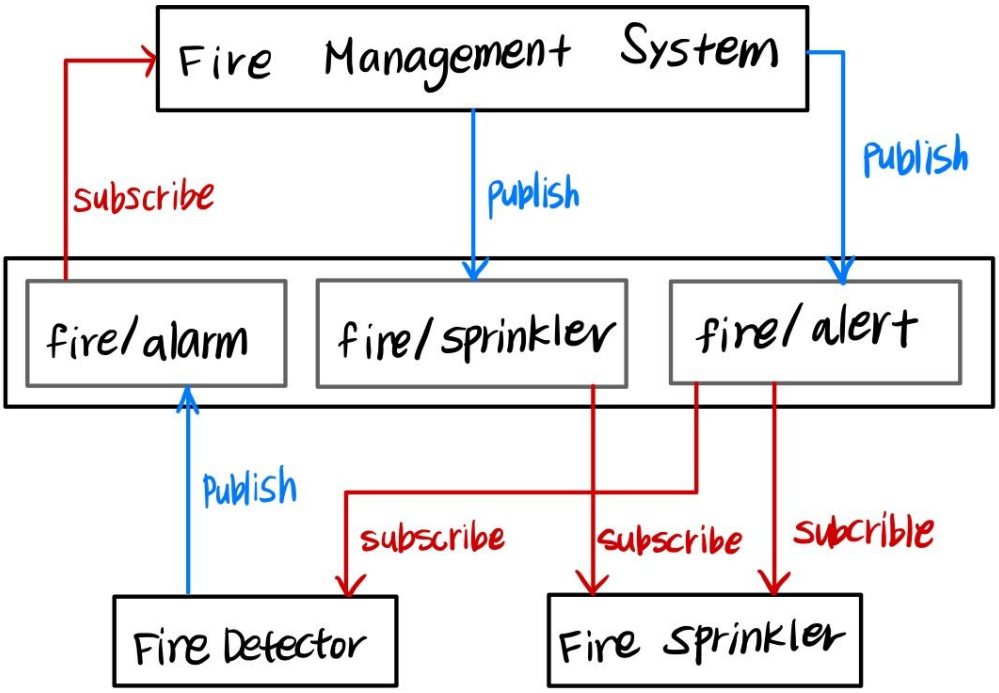
-policy4fireManagementSystem

버전1 정책 문서 편집

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:client/fire_management_syst
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topicfilter/fire/alarm"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/alarm"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Publish",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/alert"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Publish",
      "Resource": "arn:aws:iot:ap-northeast-2:889153138883:topic/fire/sprinkler"
    }
  ]
}
```

2. Three Device Gateway Topics

1) fire detector, fire sprinkler와 fire management system의 관계를 정리했다.



2) fireDetector.js

- Clientid: fire_detector
- publish: fire/alarm
- subscribe: fire/alert
- 3초 간격으로 'fire' 또는 'notfire' 메시지를 fire/alarm에 퍼블리시한다.

```
//fireDetector.js
var awsIot = require('aws-iot-device-sdk');

var fireDetector = awsIot.device({
  keyPath: "./credentials/detector/c0752c99f9-private.pem.key",
  certPath: "./credentials/detector/c0752c99f9-certificate.pem.crt",
  caPath: "./credentials/detector/AmazonRootCA1.pem",
  clientId:"fire_detector",
  host:"a19tksa9tfqz8n-ats.iot.ap-northeast-2.amazonaws.com"//MQTT DN for
Device Gateway//msg broker server의 domain name
});
```

```
fireDetector.on('connect',function(){
  console.log('fire detector connected');

  fireDetector.subscribe('fire/alert',function(){
    console.log('subscribing to the topic fire/alert!');
  });

  fireDetector.on('message',function(topic,message){
    if(topic=='fire/alert'){
      console.log('receive message from fire/alert: '+ message.toString());
    }
  });

  var alarmMsg = ['fire','notfire'];
  setInterval(function(){
    var idx = Math.floor(Math.random()*2);
    var message={'notify':'fire/sprinkler', 'fmsg':alarmMsg[idx]};
    fireDetector.publish('fire/alarm',JSON.stringify(message));
    console.log('publish to fire/alarm '+JSON.stringify(message));
  },3000);
});
```

3) fireSprinkler.js

- Clientid: fire_sprinkler

- subscribe: fire/sprinkler, fire/alert

- 받은 메시지의 토픽이 fire/sprinkler인 경우 메시지의 내용이 ‘fire’라면 sprinkler on 한다.

```
//fireSprinkler.js
var awsIot = require('aws-iot-device-sdk');

var fireSprinkler = awsIot.device({
  keyPath: "./credentials/sprinkler/87ce260da4-private.pem.key",
  certPath: "./credentials/sprinkler/87ce260da4-certificate.pem.crt",
  caPath: "./credentials/sprinkler/AmazonRootCA1.pem",
  clientId:"fire_sprinkler",
  host:"a19tksa9tfqz8n-ats.iot.ap-northeast-2.amazonaws.com"//MQTT DN for
Device Gateway//msg broker server의 domain name
});

fireSprinkler.on('connect',function(){
  console.log('fire sprinkler connected');
  fireSprinkler.subscribe('fire/sprinkler',function(){
```

```
    console.log('subscribing to the topic fire/sprinkler!');
  });

  fireSprinkler.subscribe('fire/alert',function(){
    console.log('subscribing to the topic fire/alert!');
  });

  fireSprinkler.on('message',function(topic,message){
    if(topic=='fire/sprinkler'){
      var noti=JSON.parse(message.toString());
      if(noti.command=='fire') console.log(noti.command+': sprinkler on!');
      else console.log(noti.command+': sprinkler off!');
    }
    else if(topic=='fire/alert'){
      console.log('receive message from fire/alert: '+message.toString());
    }
  });
});
```

4) fireManagementSystem.js

- Clientid: fire_management_system

- publish: fire/alert

- subscribe: fire/alarm, fire/sprinkler

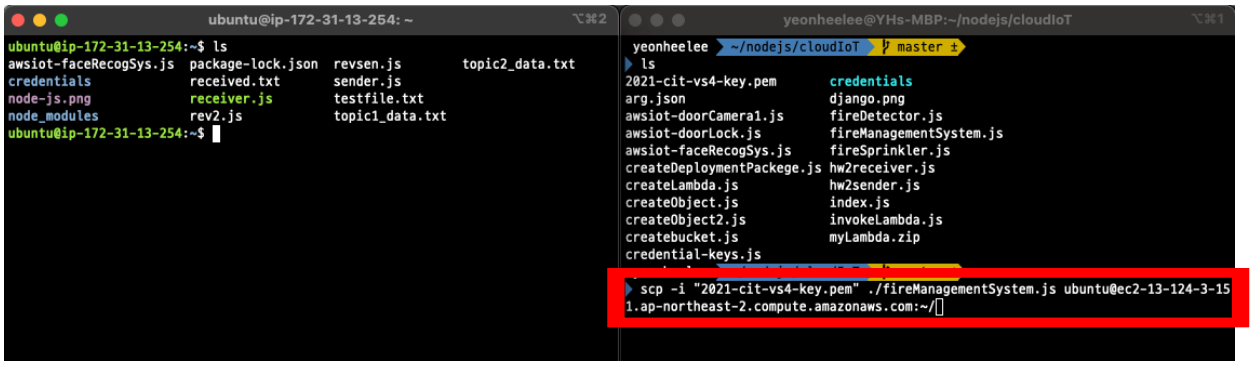
- 받은 메시지의 토픽이 fire/alarm이고 메시지 내용이 'fire'인 경우 fire/sprinkler에 {'command': 'fire'}를 보낸다.

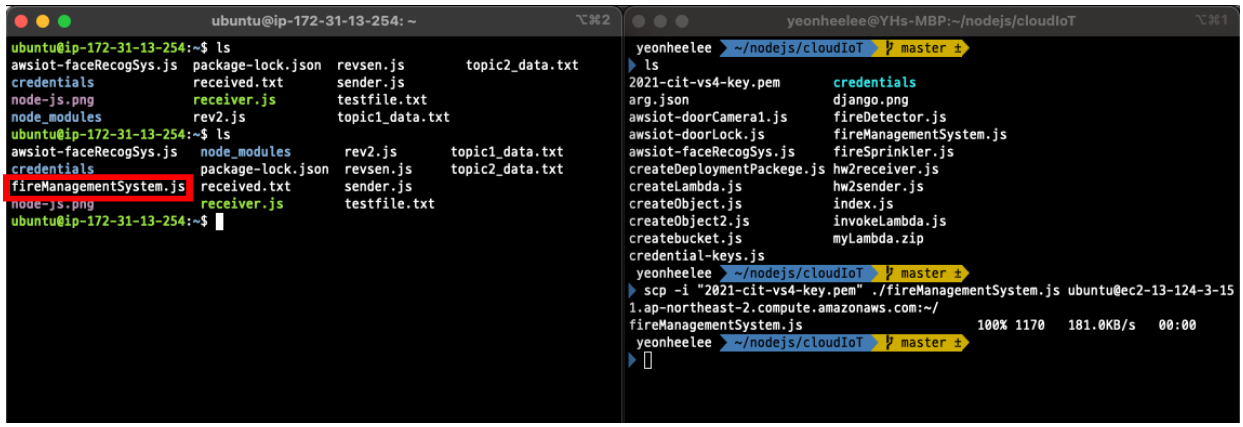
```
//fireManagementSystem.js
var awsIot = require('aws-iot-device-sdk');
var fireMngSys = awsIot.device({
  keyPath:"./credentials/management/bad3c73ae6-private.pem.key",
  certPath:"./credentials/management/bad3c73ae6-certificate.pem.crt",
  caPath:"./credentials/management/AmazonRootCA1.pem",
  clientId:"fire_management_system",
  host:"a19tksa9tfqz8n-ats.iot.ap-northeast-2.amazonaws.com"
});

fireMngSys.on('connect',function(){
  console.log('fire_management_system connected');
  fireMngSys.subscribe('fire/alarm',function(){
    console.log('subscribing to the topic fire/alarm');
  });
```

```
  fireMngSys.on('message',function(topic,message){
    if(topic!='fire/alarm') return;
    var req = JSON.parse(message.toString());
    if(req.fmsg=='fire'){
      fireMngSys.publish(req.notify,JSON.stringify({'command':'fire'}));
      console.log('publish sprinkler fire');
    }else if(req.fmsg=='notfire'){
      fireMngSys.publish(req.notify,JSON.stringify({'command':'notfire'}));
      console.log('publish sprinkler notfire');
    }
    else{

    }
  });

});
```

5) 실행결과

- scp 명령어를 사용하여 fireManagementSystem.js와 credential 파일들을 ec2 서버로 전송했다.





```
ubuntu@ip-172-31-13-254: ~
ubuntu@ip-172-31-13-254:~$ ls
awsiot-faceRecogSys.js  package-lock.json  revsen.js        topic2_data.txt
credentials             received.txt       sender.js
node-js.png             receiver.js        testfile.txt
node_modules            rev2.js            topic1_data.txt
ubuntu@ip-172-31-13-254:~$ ls
awsiot-faceRecogSys.js   node_modules       rev2.js       topic1_data.txt
credentials              package-lock.json  revsen.js     topic2_data.txt
fireManagementSystem.js  received.txt       sender.js
node-js.png              receiver.js        testfile.txt
ubuntu@ip-172-31-13-254:~$ ls
AmazonRootCA1.pem               fireManagementSystem.js  rev2.js
awsiot-faceRecogSys.js          node-js.png              revsen.js
bad3c73ae6-certificate.pem.crt  node_modules             sender.js
bad3c73ae6-private.pem.key      package-lock.json        testfile.txt
bad3c73ae6-public.pem.key       received.txt             topic1_data.txt
credentials                     receiver.js              topic2_data.txt
ubuntu@ip-172-31-13-254:~$
```

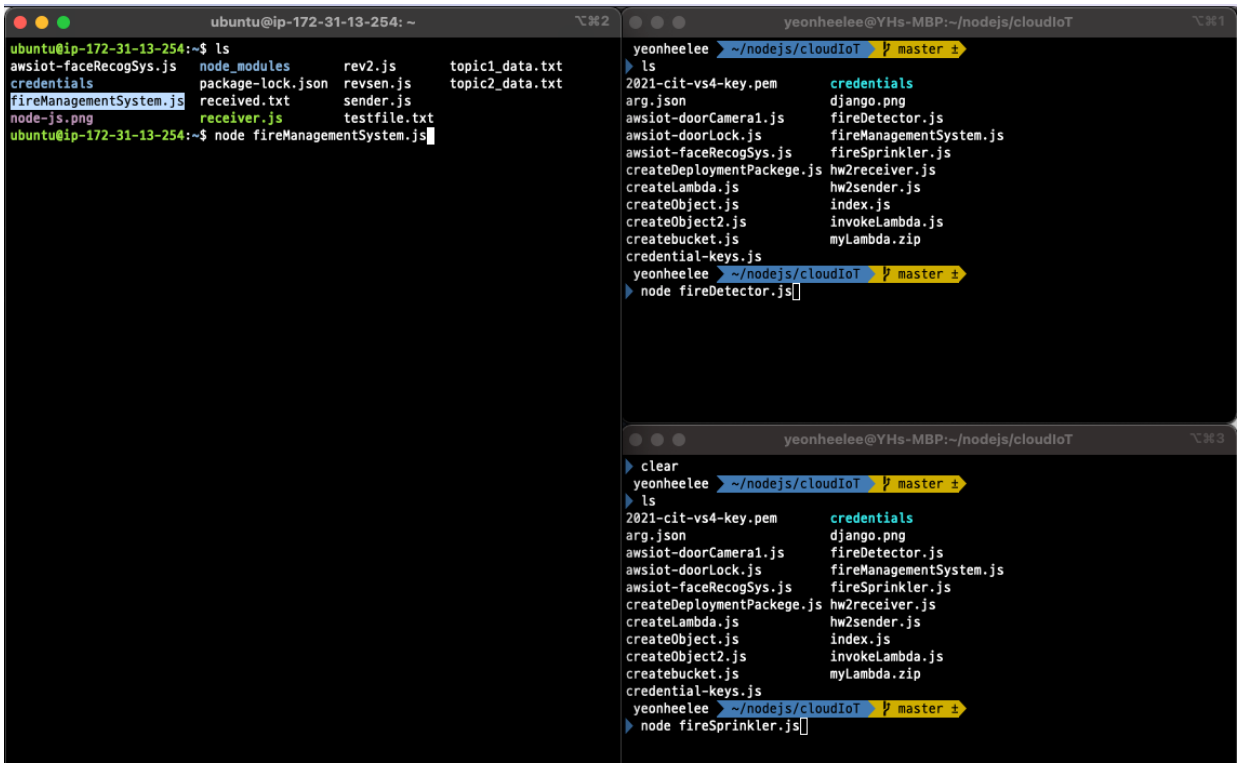
```
yeonheelee@YHs-MBP:~/nodejs/cloudIoT
yeonheelee  ~/nodejs/cloudIoT  master ±
❯ ls
2021-cit-vs4-key.pem       credentials
arg.json                   django.png
awsiot-doorCamera1.js      fireDetector.js
awsiot-doorLock.js         fireManagementSystem.js
awsiot-faceRecogSys.js     fireSprinkler.js
createDeploymentPackege.js hw2receiver.js
createLambda.js            hw2sender.js
createObject.js            index.js
createObject2.js           invokeLambda.js
createbucket.js            myLambda.zip
credential-keys.js
yeonheelee  ~/nodejs/cloudIoT  master ±
❯ scp -i "2021-cit-vs4-key.pem" ./fireManagementSystem.js ubuntu@ec2-13-124-3-15
1.ap-northeast-2.compute.amazonaws.com:~/
fireManagementSystem.js                     100% 1170   181.0KB/s   00:00
yeonheelee  ~/nodejs/cloudIoT  master ±
❯ scp -i "2021-cit-vs4-key.pem" ./credentials/management/* ubuntu@ec2-13-124-3-1
51.ap-northeast-2.compute.amazonaws.com:~/
bad3c73ae6-certificate.pem.crt              100% 1224   121.4KB/s   00:00
bad3c73ae6-private.pem.key                  100% 1675   138.1KB/s   00:00
bad3c73ae6-public.pem.key                   100%  451    48.1KB/s   00:00
yeonheelee  ~/nodejs/cloudIoT  master ±
❯
```

```
ubuntu@ip-172-31-13-254: ~/credentials/management
ubuntu@ip-172-31-13-254:~/credentials$ ls
recogsys
ubuntu@ip-172-31-13-254:~/credentials$ mkdir management
ubuntu@ip-172-31-13-254:~/credentials$ ls
management  recogsys
ubuntu@ip-172-31-13-254:~/credentials$ cd ..
ubuntu@ip-172-31-13-254:~$ ls
AmazonRootCA1.pem               fireManagementSystem.js  rev2.js
awsiot-faceRecogSys.js          node-js.png              revsen.js
bad3c73ae6-certificate.pem.crt  node_modules             sender.js
bad3c73ae6-private.pem.key      package-lock.json        testfile.txt
bad3c73ae6-public.pem.key       received.txt             topic1_data.txt
credentials                     receiver.js              topic2_data.txt
ubuntu@ip-172-31-13-254:~$ mv *.key *.crt *.pem credentials/management/
ubuntu@ip-172-31-13-254:~$ cd credentials/management/
ubuntu@ip-172-31-13-254:~/credentials/management$ ls
AmazonRootCA1.pem               bad3c73ae6-private.pem.key
bad3c73ae6-certificate.pem.crt  bad3c73ae6-public.pem.key
ubuntu@ip-172-31-13-254:~/credentials/management$
```

-cat fireManagementSystem.js 명령어를 사용하여 ec2로 옮겨진 코드 파일을 확인했다.

```
ubuntu@ip-172-31-13-254: ~
ubuntu@ip-172-31-13-254:~$ ls
awsiot-faceRecogSys.js   node_modules       rev2.js       topic1_data.txt
credentials              package-lock.json  revsen.js     topic2_data.txt
fireManagementSystem.js  received.txt       sender.js
node-js.png              receiver.js        testfile.txt
ubuntu@ip-172-31-13-254:~$ cat fireManagementSystem.js
//EC2-based application
//subscribe 'fire/alarm'
//publish 'fire/sprinkler' activation command
//publish 'fire/alert' broadcasting
var awsIot = require('aws-iot-device-sdk');
var fireMngSys = awsIot.device({
  keyPath:"./credentials/management/bad3c73ae6-private.pem.key",
  certPath:"./credentials/management/bad3c73ae6-certificate.pem.crt",
  caPath:"./credentials/management/AmazonRootCA1.pem",
  clientId:"fire_management_system",
  host:"a19tksa9tfqz8n-ats.iot.ap-northeast-2.amazonaws.com"
});

fireMngSys.on('connect',function(){
  console.log('fire_management_system connected');
  fireMngSys.subscribe('fire/alarm',function(){
    console.log('subscribing to the topic fire/alarm');
  });


  fireMngSys.on('message',function(topic,message){
    if(topic!='fire/alarm') return;
    var req = JSON.parse(message.toString());
    if(req.fmsg=='fire'){
      fireMngSys.publish(req.notify,JSON.stringify({'command':'fire'}));
      console.log('publish sprinkler fire');
    }else if(req.fmsg=='notfire'){
      fireMngSys.publish(req.notify,JSON.stringify({'command':'notfire'}));
      console.log('publish sprinkler notfire');
    }
    else{

    }
  });

});
ubuntu@ip-172-31-13-254:~$
```

- 3개의 터미널 창을 열고 fireDetector.js, fireSprinkler.js을 실행시키고, ec2에서 fireManagementSystem.js를 실행시켰다.



- detector에서 보낸 fire 메시지가 management에게 전달된다. management에서 인지하고 sprinkler에게 메시지를 보내면 sprinkler on이라는 결과를 확인할 수 있다.