Event dashboard > Write and deploy chaincode > **Create member identities**

Create member identities

One additional step will be to create certificates for each member's staff. These certificates will have custom attributes on them containing a list of operations that the staff member is allowed to perform. In this section, each member will create certificates used to verify the identities of each of its personnel and track the operations they perform.

Supplier

worker - manufactures and ships products to retailer
inspector - inspects products after fabrication

Retailer

worker - receives and labels products from supplier
seller - sells labeled products to customers

> Both the **Retailer** and **Supplier** should copy the following commands into the Cloud9 terminal *before* moving on to subsequent steps.

The following commands will create a unique randomly-generated password for each user and store that password in AWS Secrets Manager for future use:

```
WORKER1_PASSWORD=$(aws secretsmanager get-random-password --exclude-punctuation | jq -r ".RandomPassword")

WORKER2_PASSWORD=$(aws secretsmanager get-random-password --exclude-punctuation | jq -r ".RandomPassword")


aws secretsmanager create-secret --name="HLF-MEMBER-PW-NETWORK-${NETWORKID}-ACCOUNT-${WORKER1_NAME}" --secret-string=$WORKER1_PASSWORD


```

Blueventure: Blockchain Lab ✕

Track-and-Trace Blockchain Workshop for Hyperledger Fabric 2.2 (BETA)

▼ Create a Hyperledger Fabric Network

▶ Create Network & Member

▶ Accept invite and create Supplier member

Congratulations

▶ Setup Development Environment

▶ Set up a Fabric client

▼ Write and deploy chaincode

Chaincode development environment

Write chaincode

Create sharing policy

▼ **AWS account access**

Open AWS console (us-east-1)

Get AWS CLI credentials

Exit event

```
aws secretsmanager create-secret --name="HLF-MEMBER-PW-NETWORK-${NETWORKID}-ACCOUNT-${WORKER2_NAME}" --secret-string=$WORKER2_PASSWORD
```

The following commands create certificates for various personnel / roles, and then copies the certificate public keys for these identities to an S3 bucket where the other members can download them.

Create the first worker role:

```
# create worker 1 cert
cd
fabric-ca-client register -u https://$CASERVICEENDPOINT --id.name $WORKER1_NAME --id.affiliation $MEMBER_NAME --tls.certfiles $HOME/mana
fabric-ca-client enroll -u https://$WORKER1_NAME:$WORKER1_PASSWORD@$CASERVICEENDPOINT --tls.certfiles $HOME/managedblockchain-tls-chain.
cp -r admin-msp/admincerts/ $WORKER1_NAME-msp
```

Create the second worker role:

```
# create worker 2 cert
fabric-ca-client register -u https://$CASERVICEENDPOINT --id.name $WORKER2_NAME --id.affiliation $MEMBER_NAME --tls.certfiles $HOME/mana
fabric-ca-client enroll -u https://$WORKER2_NAME:$WORKER2_PASSWORD@$CASERVICEENDPOINT --tls.certfiles $HOME/managedblockchain-tls-chain.
cp -r admin-msp/admincerts/ $WORKER2_NAME-msp
```

Upload the certificates to S3 where they will be accessible via the sharing policy:

```
# upload admin certs to S3 bucket
export cacert=$(ls $HOME/admin-msp/cacerts/ca-*.pem)
aws s3api put-object --bucket $BUCKET_NAME --key ${MEMBER_ABBREVIATION}cacert.pem --body $cacert --acl bucket-owner-full-control
aws s3api put-object --bucket $BUCKET_NAME --key ${MEMBER_ABBREVIATION}admincert.pem --body $HOME/admin-msp/admincerts/cert.pem --acl bu
```

Previous Next