Amazon Managed Blockchain (Hyper Ledger Fabric) ワークショップ

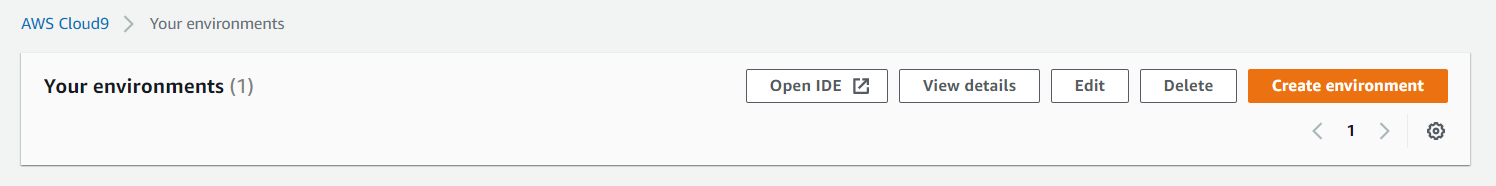
2021/04/07

シニアエバンジェリスト

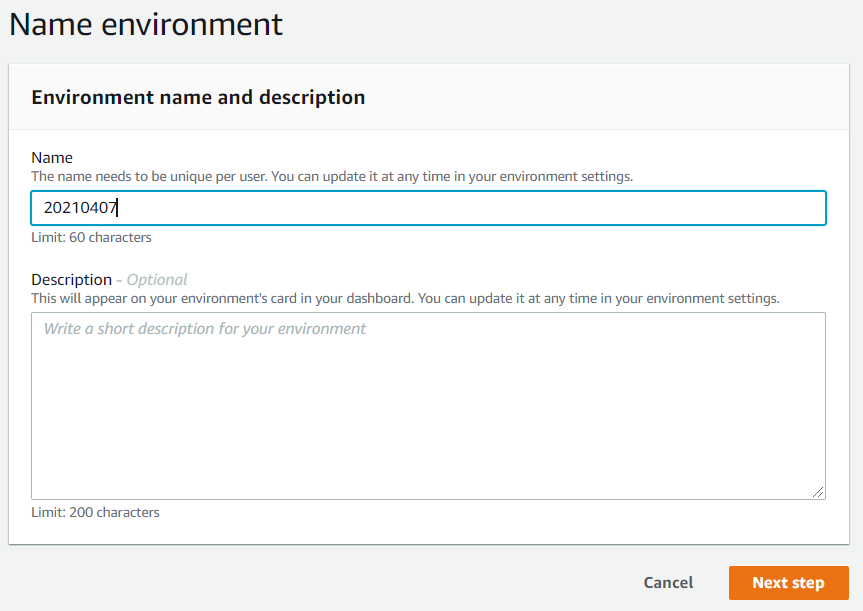
亀田 治伸

【環境構築】

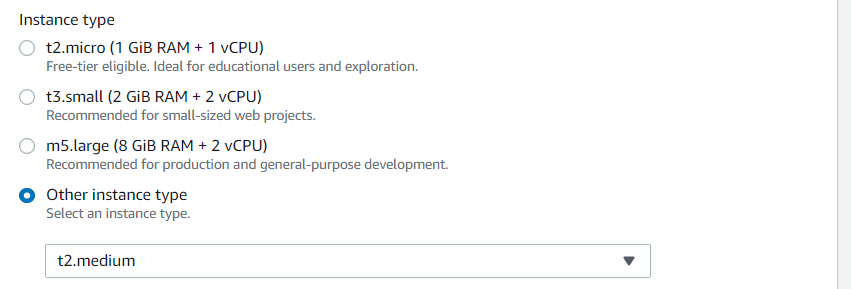
1. マネージメントコンソールでCloud9にアクセスします。



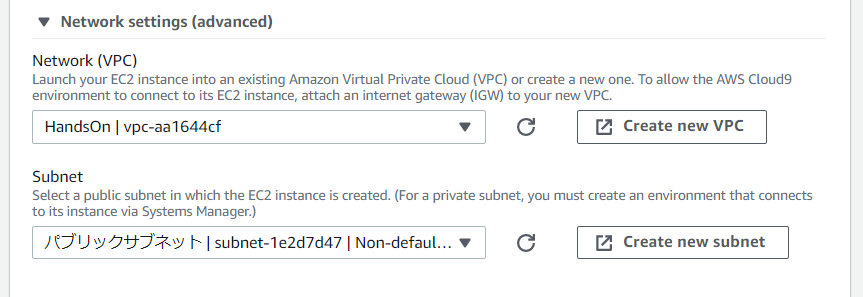
1. [Create environment]を押します
2. 適当な名前を付け、[Next step]を押します



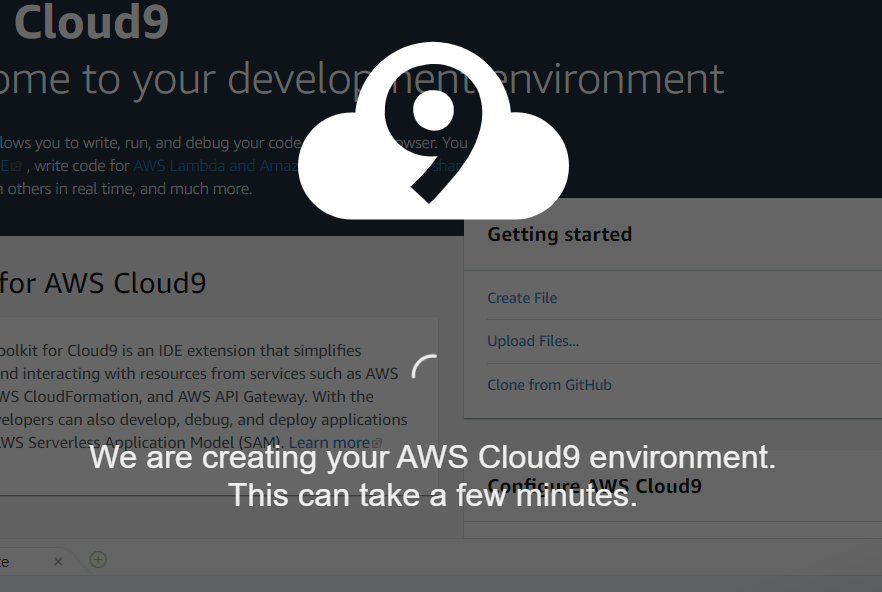
1. Instance Typeを[Other…]から[t2.medium]にします



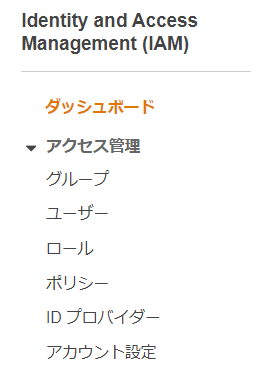
1. [Network settings (advanced)]を開き、VPCとPublic subnetを選びます。（Cloud9は通常開発環境なので、Privateに構築しますが、今日の手順ではCloud9のブロックチェーンのメンバークライアントとして設定するため、Publicに設定します）（VPCはなんでもいいですが、Public Subnetがあること。インターネットに出れる環境であること、が必要です）



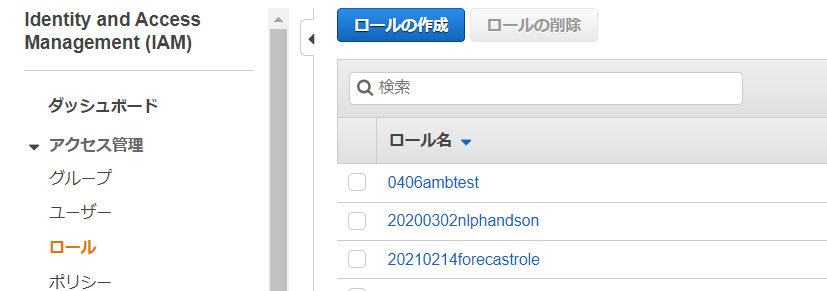
1. [Next step]を押します。次の画面ではそのまま[Create environment]を押します。
2. Cloud9が構築中になりますので、コンソールにアクセスできるまでしばらく待ちます



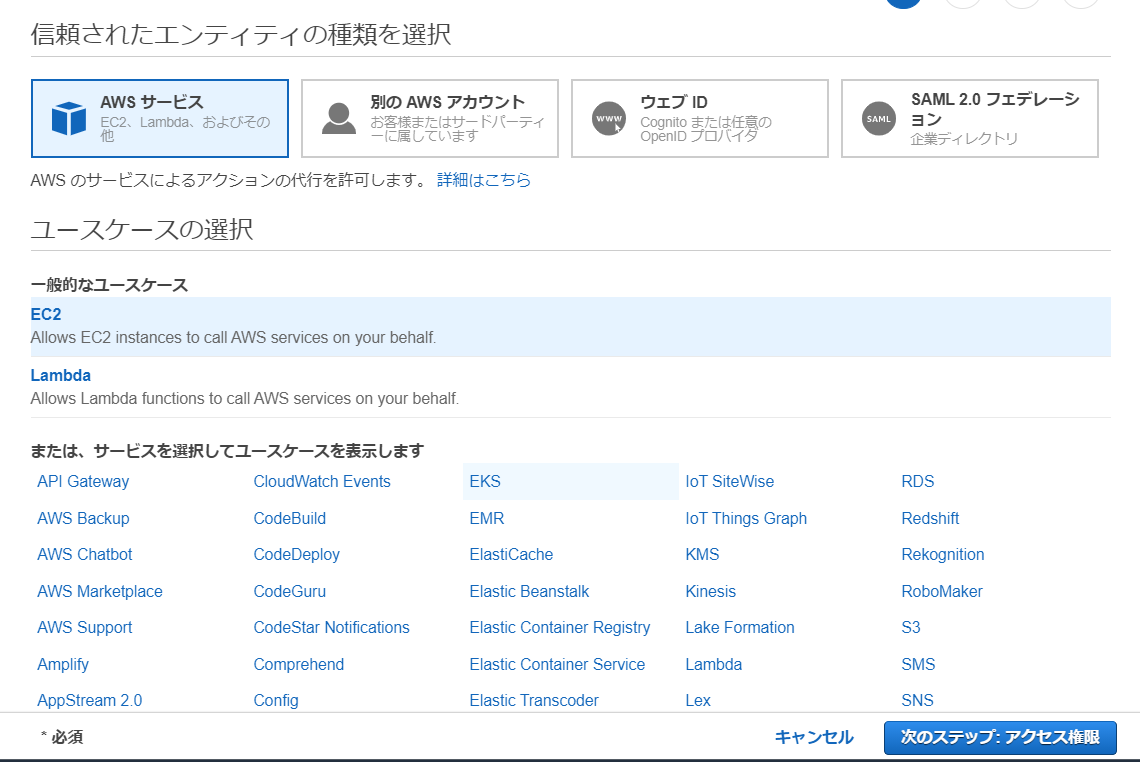
1. ブラウザ別タブでIAMロールの画面にいきます



1. [ロール]をクリックし、[ロールの作成]を押します

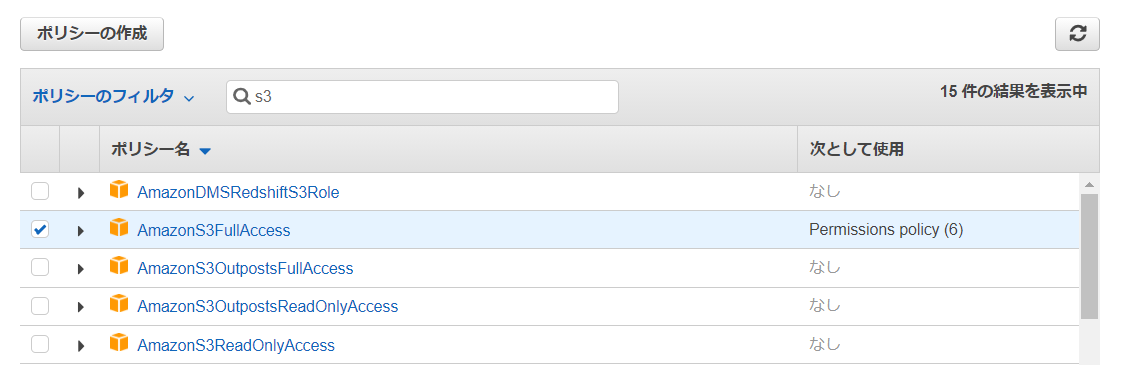


1. [一般的なユースケース]からEC2を選び、[次のステップ:アクセス権限]を押します



1. 以下2つのロールをアタッチします





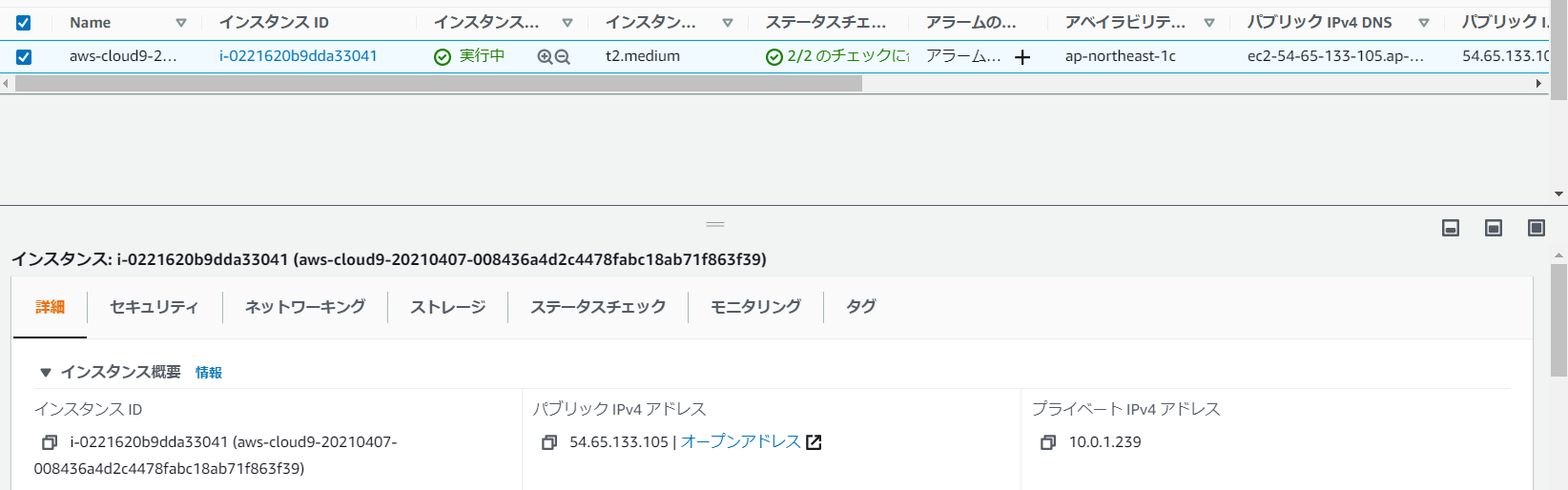
1. [次のステップ：タグ]を押します。その次のページではそのまま[次のステップ：確認]を押します
2. 適当な名前を付け、[ロールの作成]を押します。ポリシーが正しく2つ設定されていることを確認してください



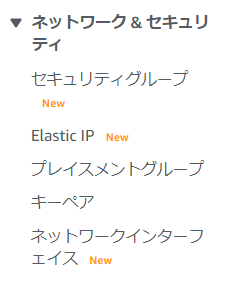
1. 以下が表示されれば完了です



1. ブラウザの別タブを開きEC2の画面にいきます。画面左ペインのインスタンスをクリックし、Cloud9用EC2インスタンスを特定します。Nameに[cloud9]の文字列が自動で設定されているものがそれです。
2. インスタンスのチェックボックスを押し、パブリックIPが正しく設定されているか確認します



（設定されていない場合、左ペインのElastic IPから設定してください）



1. [アクション]→[セキュリティ]→[IAMロールの変更]を選びます



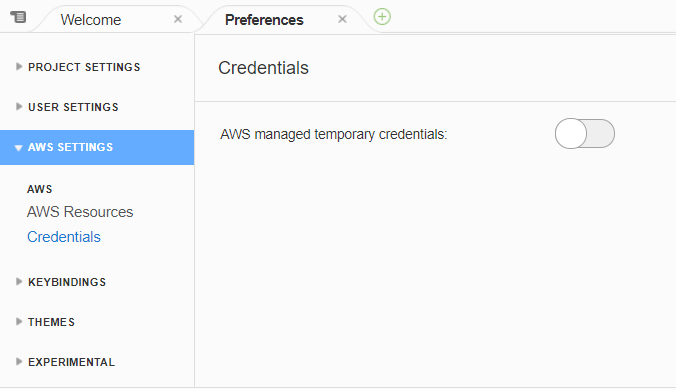
1. 今日作成したロールを選択し、[保存]を押します



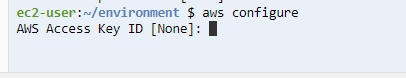
1. Cloud9画面右上の歯車マークを押します



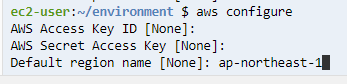
1. 左ペインから[AWS SETTINGS]→[Credentials]を選び、[AWS managed temporary credentials]をオフにします



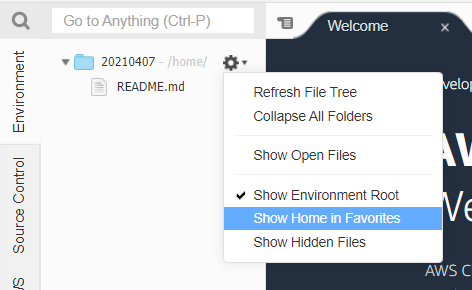
1. タブ右上のxマークでタブを閉じます
2. aws configure をターミナルで入力し実行します



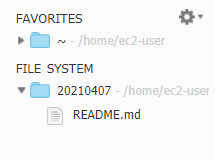
1. [Default region name]に[ap-noratheast-1]と設定し、それ以外は空欄のまま、エンターをおします



1. 左ペイン、[Environment]から歯車を押し、[Show Home in Favorites]を押します



1. 以下のようにEC2のホームディレクトリが表示されます



**[Managed Blockchain]の初期設定**

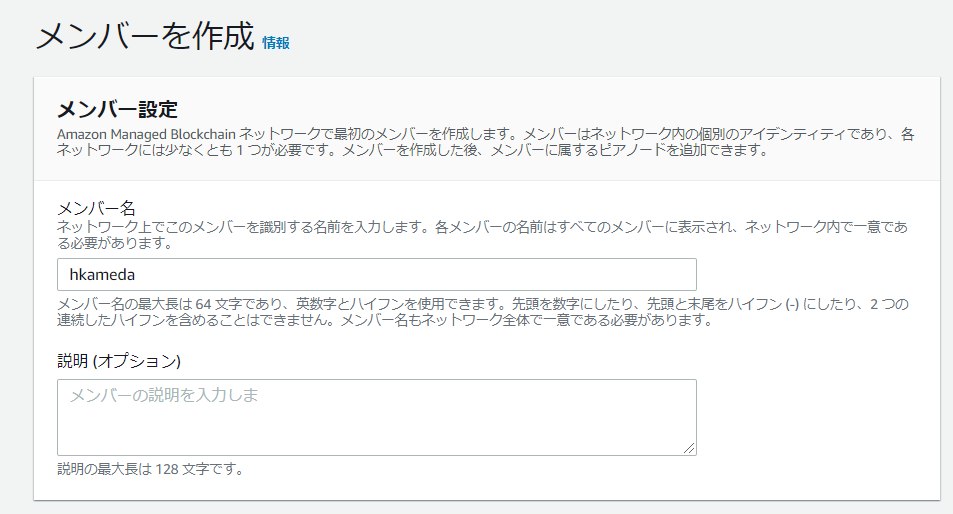
1. マネージメントコンソールでmanaged Blockchainを開きます



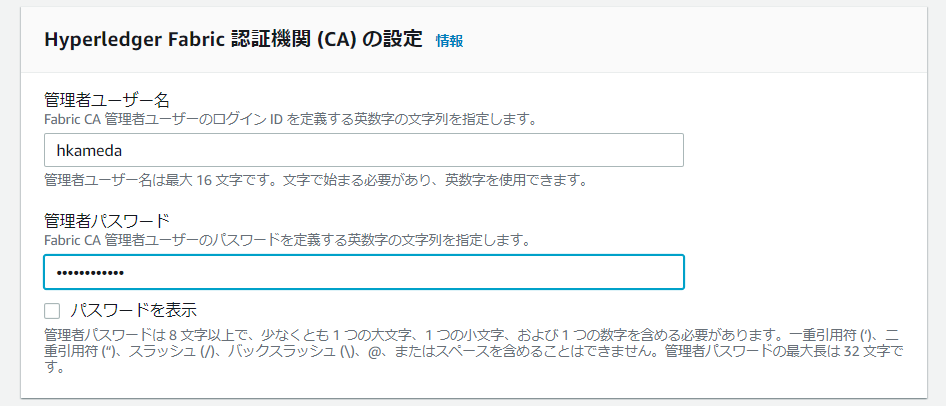
1. [ネットワークを作成]をおし、すべての設定はデフォルトのまま、適当な名前をつけ、[次へ]をおします



1. メンバー名に適当な名前をつけます



1. 認証機関の名前とパスワードを付けます。この2つは後で使うので必ずメモをお願いします。パスワードは大文字小文字数字の組み合わせが必要です



1. [次へ]をおし、次の画面で[ネットワークとメンバーの作成]を押します



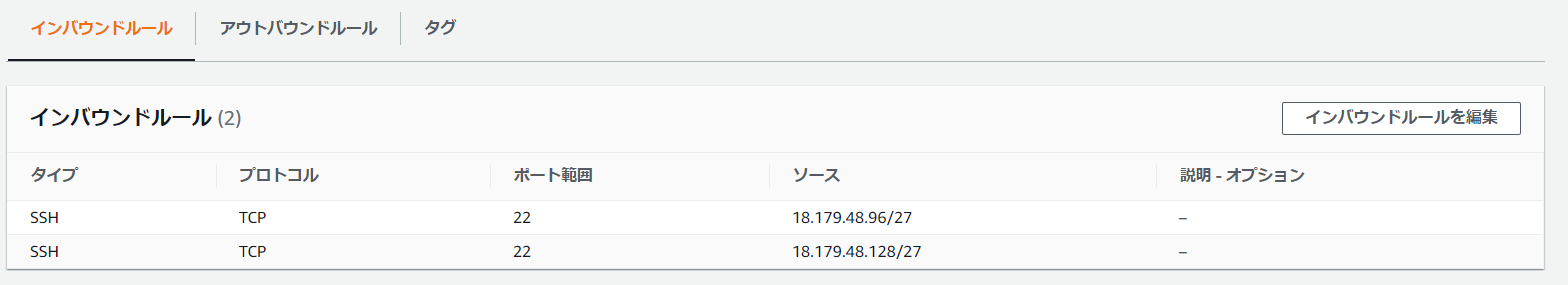
30分ほど待ち時間が発生しますので、その間Cloud9に戻り作業を続行します

**[Cloud9でHyperLedger Fabricクライアントの設置]**

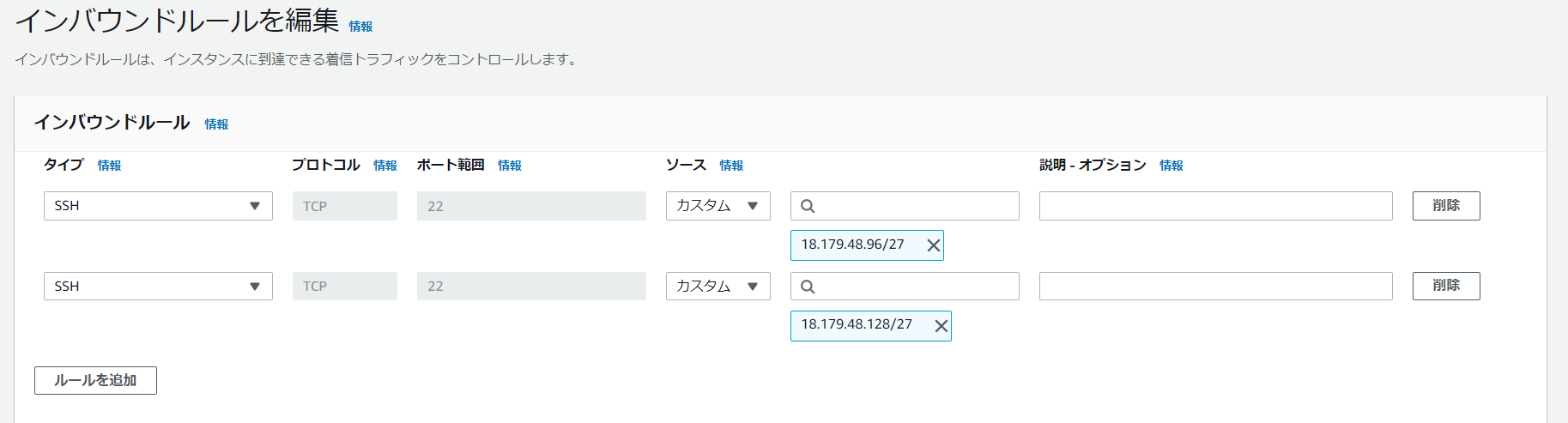
1. EC2マネージメントコンソールでCloud9インスタンスを特定し、チェックボックスにチェックを押し、画面下のセキュリティグループをクリックします。（セキュティグループIDは後で使うのでメモをとっておきます）



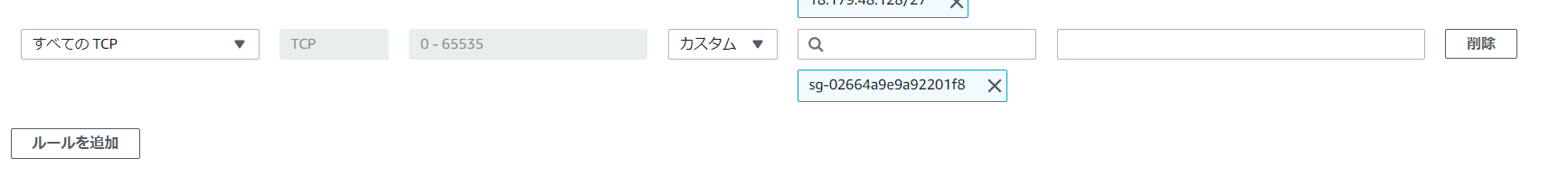
1. [インバウンドルールの編集]ボタンをおします。（設定済ルールは人により異なりますので、気にする必要はありません）



1. [ルールの追加]ボタンをおします

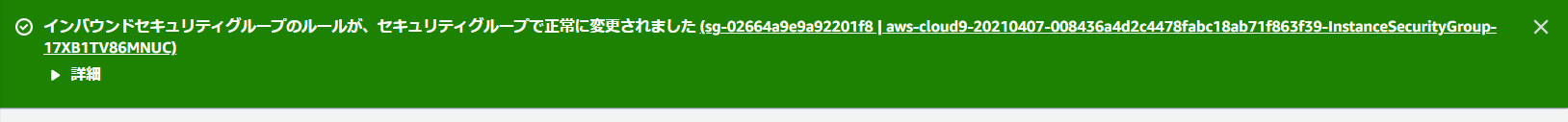


1. タイプにすべてのTCP、ソースに先ほどメモをしたセキュリティグループID（つまり自分自身です）を選びます



これにより、同じセキュリティグループを持つインターフェースからの通信が許可されるようになります

1. [ルールを保存]をおします



1. Cloud9コンソールで以下を実行します

sudo yum update –y

1. 以下を実行します

sudo yum install jq telnet emacs docker libtool libtool-ltdl-devel git –y

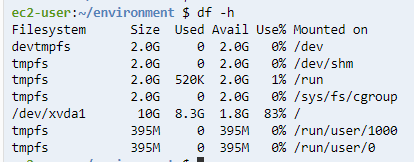
1. 以下を実行します

sudo service docker start

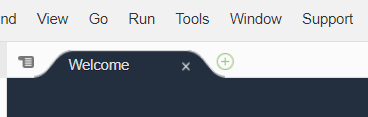
1. 以下を実行します

sudo usermod -a -G docker ec2-user

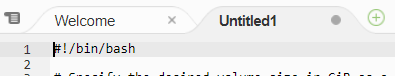
1. df –h を実行すると、ディスク残り容量が少ないことがわかります。



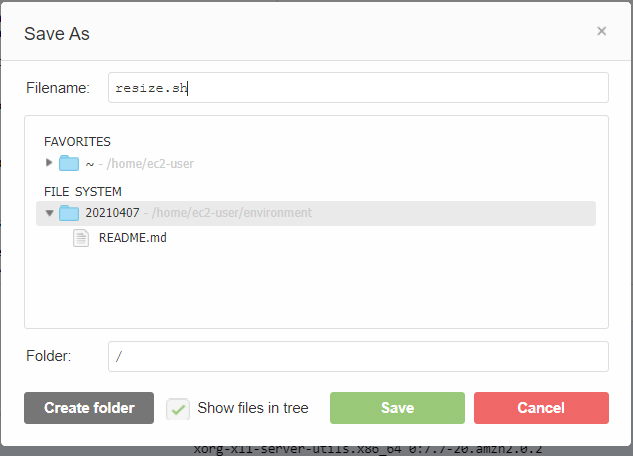
1. [resize.sh]を今日の資料フォルダやgitから開き内容をコピーします。
2. Cloud9上部画面のプラスボタンをおして[New File]を選びます



1. 先ほどの内容を貼り付け、タブの黒丸を押すと保存画面がでてきます。



1. [resize.sh]と名前を付けて[Save]をおします



1. 先ほど作成したIAMロールに以下2つの権限を追加で付与します

[AmazonEC2FullAccess](https://console.aws.amazon.com/iam/home?region=ap-northeast-1#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonEC2FullAccess)

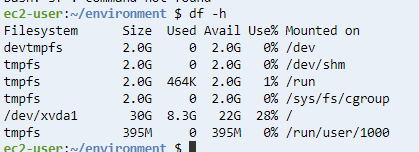
[AWSCloud9Administrator](https://console.aws.amazon.com/iam/home?region=ap-northeast-1#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSCloud9Administrator)



1. 2-3分待って、以下を実行します

sh resize.sh 30

1. df –hの出力が以下のようになれば成功です



1. EC2の画面から、一度Cloud9インスタンスを再起動し、再度ターミナル画面に繋げます
2. 以下を実行します

sudo curl -L \

https://github.com/docker/compose/releases/download/1.20.0/docker-compose-`uname \

-s`-`uname -m` -o /usr/local/bin/docker-compose

1. 以下を実行します

sudo chmod a+x /usr/local/bin/docker-compose

1. 以下を実行します

wget <https://dl.google.com/go/go1.14.4.linux-amd64.tar.gz>

1. 以下を実行します

tar -xzf go1.14.4.linux-amd64.tar.gz

1. 以下を実行します

sudo mv go /usr/local

1. 以下を実行します

sudo yum install git –y

（Nothing to doと表示される場合がありますが、気にせずすすめます）

1. シナリオアセットの[bash\_profile]を開きます。
2. そろそろblockchainの構築が終わっているので、ブラウザでステータスを確認します。利用可能となっていれば構築完了です。まだの方は利用可能となるまで待ちます。定期的にブラウザをリロードしてみてください



1. [bash\_profile]の[MyMemberCaEndpoint]をメンバー詳細画面で表示される[Fabric 認証機関エンドポイント]の値に置換します。同様に[MyNetworkOrdererEndpoint]を[サービスエンドポイントの注文]の値に置換します
2. Cloud9で以下を実行します

vi ~/.bash\_profile

1. 小文字の[d]を押して、全部消します。その後カーソルが一番左上にあることを確認し、[a]をおし（画面下にINSERTと表示されます）、修正したbash\_profileの中身をコピペます。
2. escキーを押した後、[:wq]と入力します
3. 以下を実行します

source ~/.bash\_profile

1. 以下を実行します

sudo docker version

1. 以下を実行します

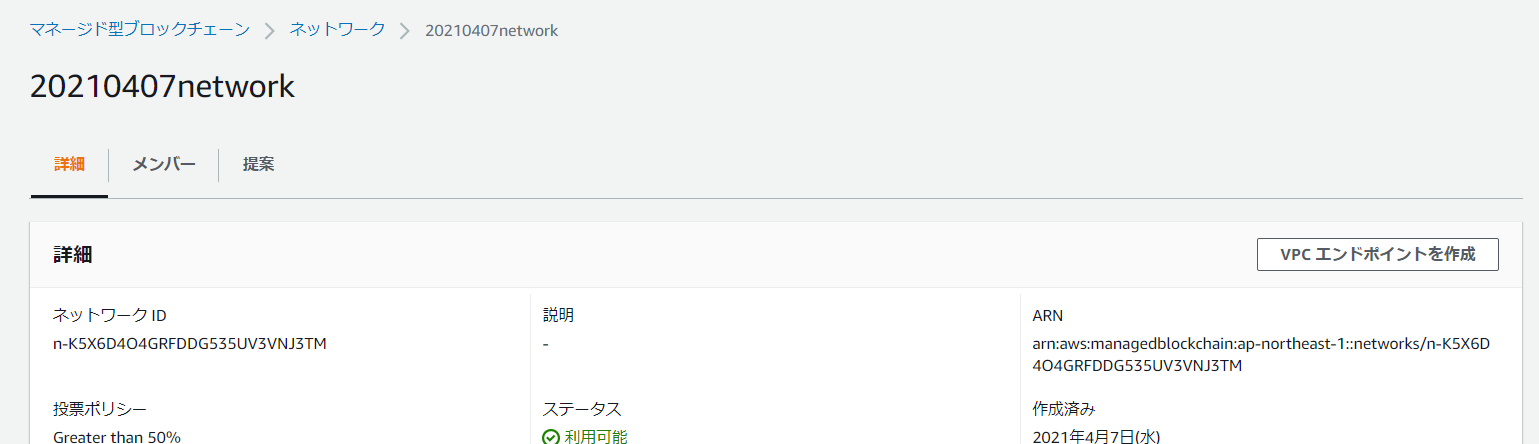
sudo /usr/local/bin/docker-compose version

1. 以下を実行します

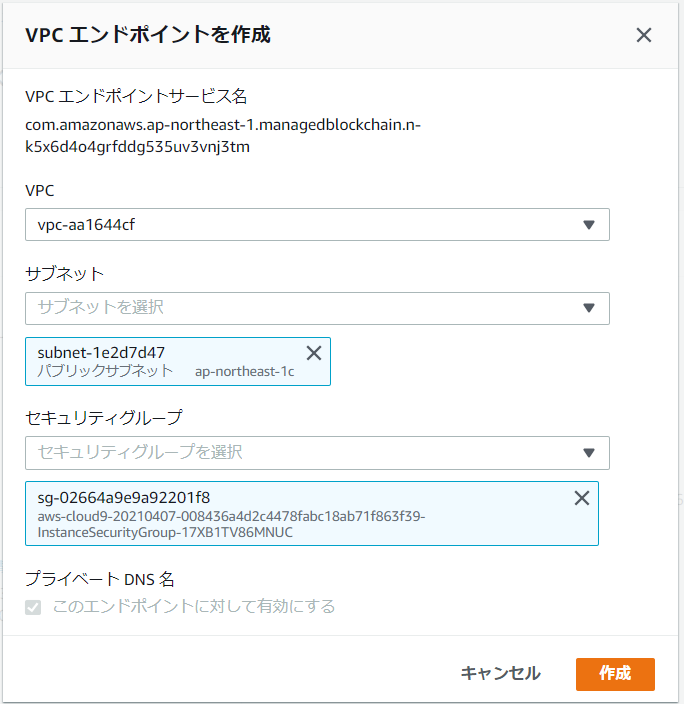
go version

[Managed BlockchainのVPCエンドポイント作成]

1. Managed blockchainのマネージメントコンソールから、ネットワークの詳細画面を開き、[VPCエンドポイントを作成]ボタンをおします

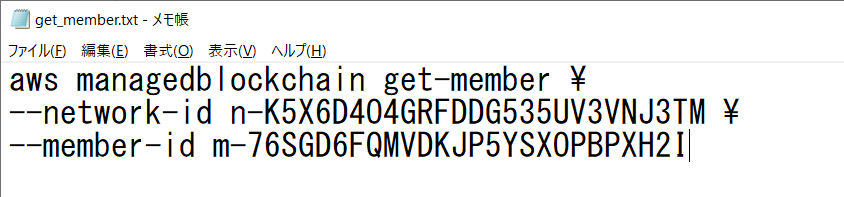


1. Cloud9を構築した、VPC,サブネットを指定し、先ほどメモした同じセキュリティグループを指定し、[作成]を押します。



これにより、Cloud9とVPC内部で通信が行えるようになりました

1. アセットから[get\_member]を開き、[network-id],[member-id]をぞれぞれ、マネージメントコンソールで表示されているIDに置き換えます。以下のようになるはずです。なお、この2つのIDもどこかにメモしておいた方が便利です



1. 全体をコピーし、Cloud9で実行します。以下のような表示になれば成功です



1. 以下を実行します

curl https://$CASERVICEENDPOINT/cainfo -k

出力の最後に[true]と表示されれば成功です

1. 以下を実行します

mkdir -p /home/ec2-user/go/src/github.com/hyperledger/fabric-ca

1. 以下を実行します

cd /home/ec2-user/go/src/github.com/hyperledger/fabric-ca

1. 以下を実行します

wget <https://github.com/hyperledger/fabric-ca/releases/download/v1.4.7/hyperledger-fabric-ca-linux-amd64-1.4.7.tar.gz>

1. 以下を実行します

tar -xzf hyperledger-fabric-ca-linux-amd64-1.4.7.tar.gz

1. 以下を実行します

cd /home/ec2-user

1. 以下を実行します

git clone --branch v1.4.7 <https://github.com/hyperledger/fabric-samples.git>

**[Managed BlockchainでPeer Noteを作成]**

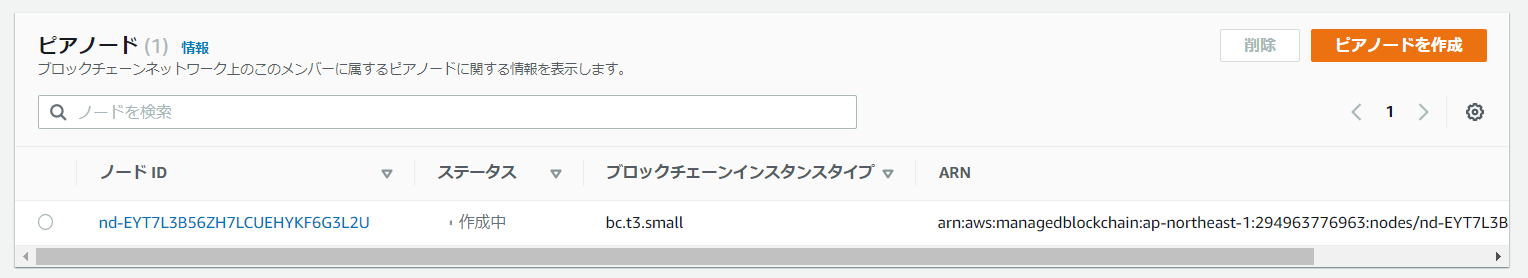
1. マネージメントコンソールのBlockchainで、メンバーの詳細画面にいき[ピアノードを作成]ボタンをおします



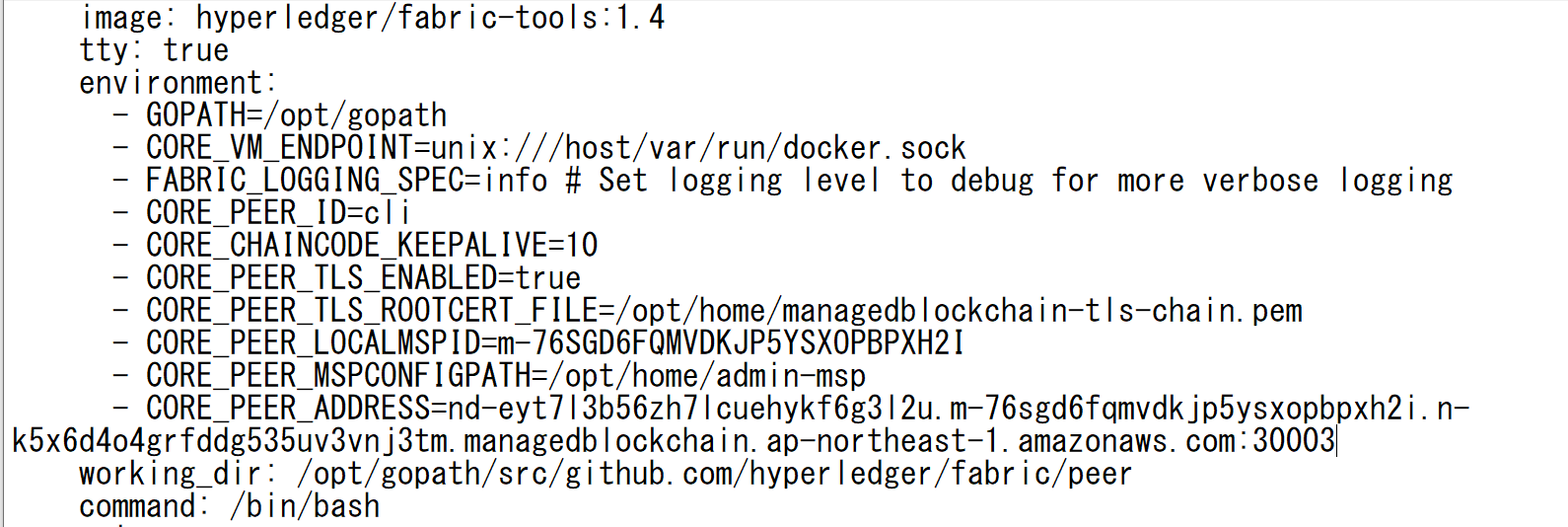
1. 全てデフォルトのまま[ピアノードを作成]ボタンをおします



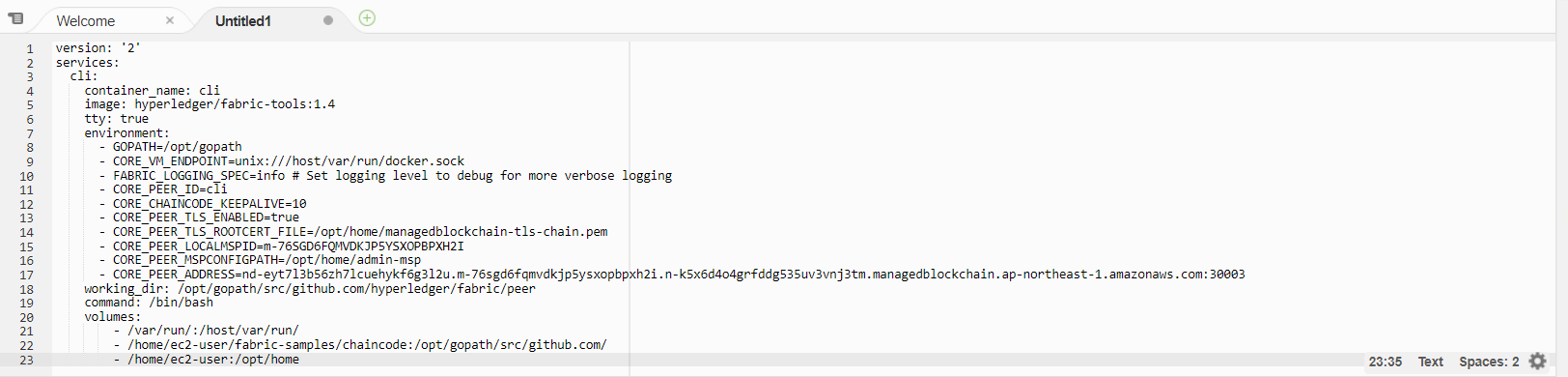
1. 作成中となりますの。作成中でもノードIDをクリックすると詳細が出てきますので[ピアエンドポイント]の値をメモします



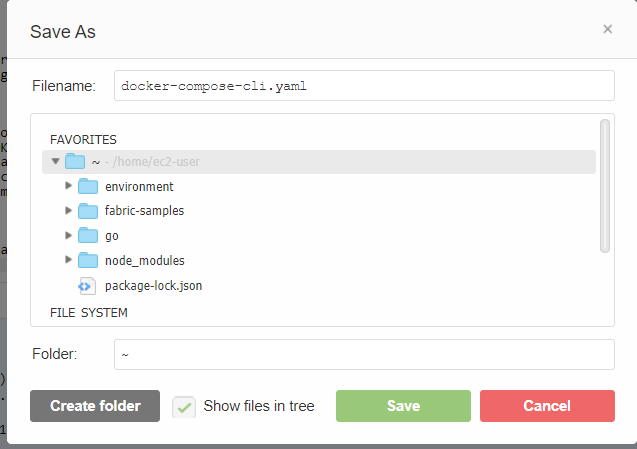
1. アセットから[docker-compose-cli.yaml]を開きます
2. [MyMemberID],[MyPeerNodeEndpoint]の値をそれぞれ置き換えます。以下のようになります

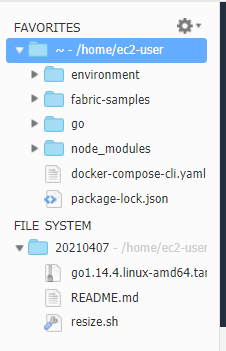


1. 今日すでに作業をしたように、Cloud9でNew Fileを作成し、上記で修正した内容をペーストします。



1. [docker-compose-cli.yaml]のファイル名で保存します。保存先は[ec2-user]の直下であることに注意してください

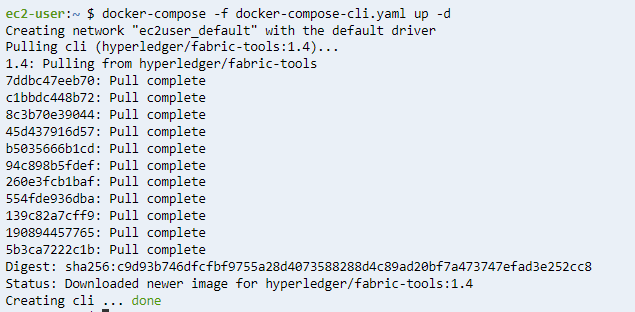




1. 以下を実行します

docker-compose -f docker-compose-cli.yaml up –d

正しく設定されていると以下のような出力となります



1. 以下を実行します

sudo /usr/local/bin/docker-compose -f docker-compose-cli.yaml up –d

**[証明書の設定]**

1. 今までの手順でCloud9がネットワークに参加しているメンバーのクライアントとして設定され、ブロックチェーン処理を行うためのコンテナが作成されました。ただしネットワークに処理を流すためには、証明書が正しく設定されなければ認証されないため、設定を行う必要があります。以下を実行します

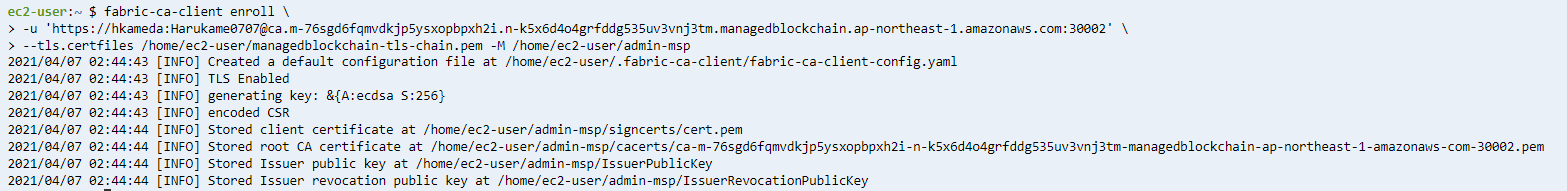
aws s3 cp s3://ap-northeast-1.managedblockchain/etc/managedblockchain-tls-chain.pem /home/ec2-user/managedblockchain-tls-chain.pem

（別リージョンで作業している場合は、ap-northeast-1を書き換えてください）

1. 以下を実行します

openssl x509 -noout -text -in /home/ec2-user/managedblockchain-tls-chain.pem

1. アセットから[fabric-ca-client]を開き、[AdminUsername],[AdminPassword]を今日設定したIDとパスワードに置換し、さらに、[$CASERVICEENDPOINT]を[Fabric 認証機関エンドポイント]の値に置き換えます。（ドルマーク毎置き換えてください）
2. 置き換えが完了したら全体をコピペし、Cloud9で実行します



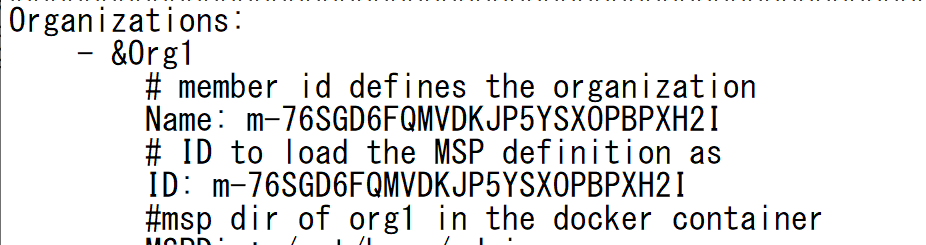
1. 以下を実行します

cp -r /home/ec2-user/admin-msp/signcerts admin-msp/admincerts

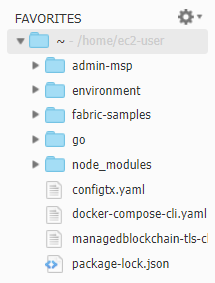
**[チャネルの作成]**

**Hyperledger Fabricでは、元帳はチャネルで管理されます。すべてのメンバーが共通のチャネルで操作する場合、元帳はネットワーク全体で共有できます。**

1. アセットから[configtx.yaml]を開きます
2. 2か所の[MemberID]を置換します



1. Cloud9でNew Fileを作成し中身を全部コピーし以下のようにファイルを作成します



1. 以下のコマンドを実行します

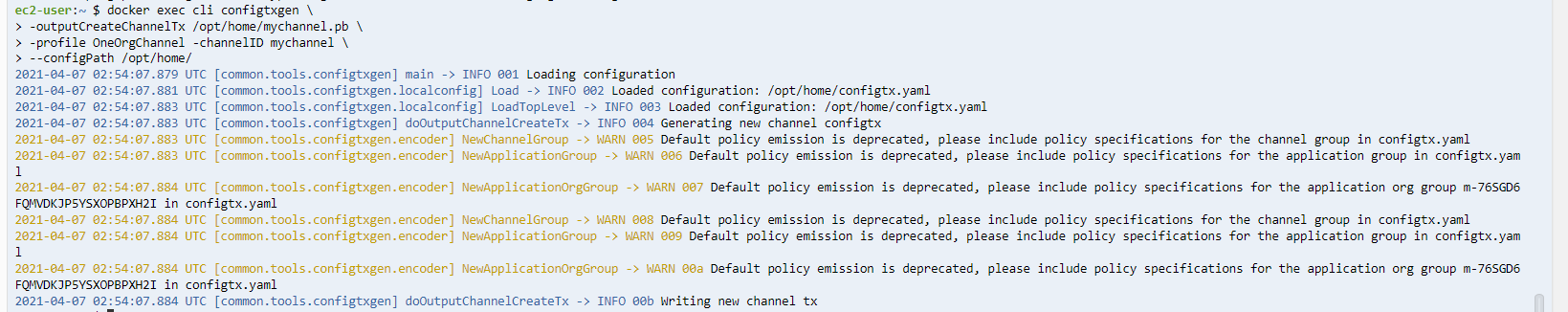
docker exec cli configtxgen \

-outputCreateChannelTx /opt/home/mychannel.pb \

-profile OneOrgChannel -channelID mychannel \

--configPath /opt/home/

以下のように表示がされれば成功です。（Warningは新しいチャネル作成をしていない、というただの表示なので気にせず進めます）



1. 以下の部分の置き換え、Cloud9で実行します

export ORDERER=<サービスエンドポイントの注文、の値>

1. 以下を実行します

docker exec cli peer channel create -c mychannel \

-f /opt/home/mychannel.pb -o $ORDERER \

--cafile /opt/home/managedblockchain-tls-chain.pem –tls

1. 以下を実行します

docker exec cli peer channel join -b mychannel.block \

-o $ORDERER --cafile /opt/home/managedblockchain-tls-chain.pem –tls

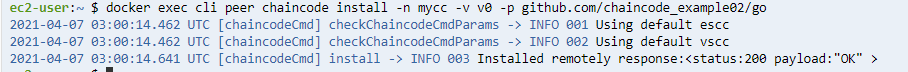
**[Chaincode]の実行**

chaincodeは、Hyperledger Fabric上におけるいわゆるSmartContractと言われる部分のことで、台帳への読み書き操作をつかさどるモジュールです

1. 以下を実行します

docker exec cli peer chaincode install -n mycc -v v0 -p github.com/chaincode\_example02/go

ここまでの手順が正しければ、以下のような表示になります。



1. 以下を実行します

docker exec cli peer chaincode instantiate \

-o $ORDERER -C mychannel -n mycc -v v0 \

-c '{"Args":["init","a","100","b","200"]}' \

--cafile /opt/home/managedblockchain-tls-chain.pem –tls

（実行完了には数分待ちます）

1. 以下を実行します

docker exec cli peer chaincode list --instantiated \

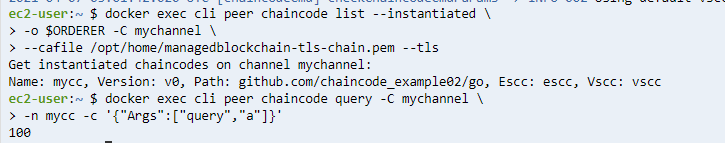
-o $ORDERER -C mychannel \

--cafile /opt/home/managedblockchain-tls-chain.pem –tls

1. いよいよトランザクションをテストで流してみます。以下を実行します

docker exec cli peer chaincode query -C mychannel \

-n mycc -c '{"Args":["query","a"]}'



1. 今度は100という値から10を引くコマンドを実行します

docker exec cli peer chaincode invoke -C mychannel \

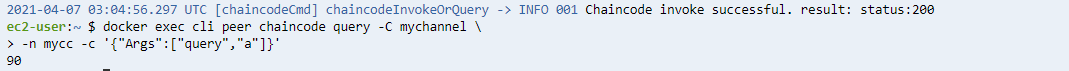
-n mycc -c '{"Args":["invoke","a","b","10"]}' \

-o $ORDERER --cafile /opt/home/managedblockchain-tls-chain.pem –tls

1. 以下を実行します

docker exec cli peer chaincode query -C mychannel \

-n mycc -c '{"Args":["query","a"]}'



**[Next Step]**

現在はメンバーが1人しかいませんが、異なるAWSアカウントで同じ手順を行うことで、メンバーを増やすことが可能です。環境の準備が可能な方は、こちらを参考に挑戦してみてください。英語ですが、実際の物流を想定したシナリオになっています。

<https://track-and-trace-blockchain.workshop.aws/>

おつかれさまでした！

後片付けですが、以下の削除をお願いします

* Cloud9の画面からCloud9インスタンス
* Managed Blockchainのピアノード削除
* Managed Blockchainのメンバー削除
* Managed Blockchainのネットワーク削除