Installing ICP CloudFoundry

vCenter Preparations

1. (optional) Create a new Cluster for CloudFoundry

   ```
   Cluster Name: CloudFoundry
   ```

2. (optional) Create a new Datastore for CloudFoundry

   ```
   Datstore Name: CF_DS
   ```

3. Browse the CloudFoundry datastore and create a /Disks folder off the root

4. Create a network to be used by Cloud Foundry

   ```
   Network Name: CF_Net
   ```

5. Create a vCenter user with sufficient privileges

   - Create and manage VMs in 'CloudFoundry' Cluster
   - Create and manage files and VMs on 'CF_DS' Datastore
   - Use the 'CF_Net' network

6. Create a new ResourcePool for CloudFoundry

   `ICP_CF_210`

Prepare Installation Virtual Machine

1. Create an installation VM in your 'CloudFoundry' cluster

   - 2 vCPU, 8GB Memory, 200GB Disk (thin provisioned)

- Ubuntu 16.04, basic server with ssh

- *NOTE:* At this time the CF installation is only supported for Ubuntu.

2. Install docker-ce on the installation VM

   - If you don't have access to the public Internet, you need to download the `.deb` package for your Ubuntu distribution.
   - See the Docker documentation, Install from a package

3. Copy tarball to /opt on the newly provisioned Virtual machine
   From the /opt directory on the installation VM

```
cd /opt
scp myuser@filestore.local:/storage/icp/ibm-cloud-private-cloud-foundry-x86_64-
```

4. Copy CF tarball to /opt and untar to /opt/cf

```
mkdir -p /opt/cf
cd /opt/cf
gunzip -c /opt/<tarball.tar.gz> |tar -xvf -
```

5. Import images into docker
   `cd /opt/icp/cf ./import_images.sh`

Generate Certificate keys for your CloudFoundry domains

1. Generate certificate keys for your new domain[s]

   Just like IBM Bluemix Public, ICP uses two domains for its CloudFoundry implementation, one for the infrastructure components (e.g. bluemix.net), and one for the applications (e.g. mybluemix.net). In our example we will use bluemix.csplab.local and mybluemix.csplab.local.

   Perform the following on your installation virtual machine as the root user:

   - Generate the root certificate (remember the password you choose, you will need it in

the next step):

```
openssl genrsa -des3 -out rootCA.key 2048
```

- Self-sign the Certificate (use the password from the previous step when asked):

```
openssl req -x509 -new -nodes -key rootCA.key -days 1024 -out rootCA.pem
```

- Store the rootCA.key file and its password in a secure location
- Generate a domain key for the self-signed certificates each domain (use the same values as specified in the previous step):

```
openssl req -new -newkey rsa:2048 -nodes -out star_<your_domain>.csr -keyout st
```

Example:

```
openssl req -new -newkey rsa:2048 -nodes -out star_bluemix.csplab.local.csr -ke

openssl req -new -newkey rsa:2048 -nodes -out star_mybluemix.csplab.local.csr -
```

- Generate the domain certificates

i. Create domain certificate extension files:

   - File name: v3_ext.bluemix

```
[ v3_req ]
subjectAltName=DNS:*.bluemix.csplab.local, DNS:csplab.local
```

   - File name: v3_ext.mybluemix

```
[ v3_req ]
subjectAltName=DNS:*.mybluemix.csplab.local,DNS:csplab.local
```

ii. Generate the domain certificate for each domain (use the password for your

rootCA.key file again here):

```
openssl x509 -req -in star_bluemix.csplab.local.csr -CA rootCA.pem -CAkey r

openssl x509 -req -in star_mybluemix.csplab.local.csr -CA rootCA.pem -CAkey
```

Prepare CloudFoundry for Installation

1. Launch the inception image

   This will launch the inception container and copy configuration files to the local filesytem. The container will continue to run and await further instructions.

   ```
   cd /opt/cf
   ./launch.sh -n myEnv -e LICENSE=accept -b ./BOM.yml -c /data |tee launch.log
   ```

   - **-n:** Name. This is the name of the environment you will deploy. It is expected that an enterprise could have more than one environment. This option keeps configuration information for separate environments separate.
   - **-b:** Location of the BOM.yml file.
   - **-c:** Local directory to hold persistent configuration data
   - **|tee launch.log 2>&1:** Copy everything from stdout and stderr to a file named launch.log in the current directory for later troubleshooting as needed.

2. Configure your deployment

   - **NOTE:** The JSON configuration file is no longer supported. Even though you see a uiconfig.json file in the data artifacts for the installation, do not use it. Only YAML is supported.
   - Copy the content below or the content from the KC into a file that will be used for defining the CF configuration. You can use uiconfig.yml for the file name, or some name that is more meaningful to the particular deployment.
   - The same YAML content is available in the ICP/CF Knowledge Center section,

Installing IBM Cloud Private Cloud Foundry

- NOTE: The YAML content in the KC is missing the `address_range` attribute.
- Update the file to match your environment

```
uiconfig:
bluemix_apps_domain: mybluemix.csplab.local
bluemix_apps_domain_cert: |+
 -----BEGIN CERTIFICATE-----
 -----END CERTIFICATE-----
bluemix_apps_domain_cert_rsa_key: |+
 -----BEGIN PRIVATE KEY-----
 -----END PRIVATE KEY-----
bluemix_env_domain: bluemix.csplab.local
bluemix_env_domain_cert: |+
 -----BEGIN CERTIFICATE-----
 -----END CERTIFICATE-----
bluemix_env_domain_cert_ca: |+
 -----BEGIN CERTIFICATE-----
 -----END CERTIFICATE-----
bluemix_env_domain_cert_rsa_key: |+
-----BEGIN PRIVATE KEY-----
-----END PRIVATE KEY-----
cluster_name: cf
datacenter_name: CSPLAB
datastore_pattern: CloudFoundry
diego_cell_instances: 1
director_ip: 172.16.60.0
disk_path: /Disks
external_dns: 172.16.0.11,172.16.0.17
gateway: 172.16.255.250
haproxy_ip: 172.16.60.1
main_user_name: cfadmin
main_user_password: Passw0rd!
ntp_servers: 172.16.0.9
persistent_datastore_pattern: CloudFoundry
portgroup: csplab
resource_pool: ICP-CF-210
subnet: 172.16.0.0/16
address_range: 172.16.60.0-172.16.60.255
template_folder: /icp-cf-210/templates
vm_folder: /icp-cf-210/vms
vmware_address: 10.1.212.26
vmware_password: Passw0rd!
vmware_username: cfadmin
```

- Under *bluemix_apps_domain_cert* paste the contents of star_mybluemix.csplab.local.crt

- Under *bluemix_apps_domain_cert_rsa_key* paste the contents of star_mybluemix.csplab.local.key

- Under *bluemix_env_domain_cert* paste the contents of star_mybluemix.csplab.local.crt

- Under *bluemix_env_domain_cert_ca* paste the contents of rootCA.pem

- Under *bluemix_env_domain_cert_rsa_key* paste the contents of star_mybluemix.csplab.local.key

Important: Any folders, resource pools, datacenters, etc. specified in the uiconfig.yml file must exist on the target vCenter server prior to installation and the user specified as the vmware username must have authority to create virtual machines and deploy vApps to these objects. The vmware username must have "read" access to the vCenter in order to monitor status of various tasks. That user must also have the authority to write to the specified datastores. (**NOTE:** The disk_path directory does not need to be created in the datastores. Whether or not the template_path and vm_path needs to be created is an open question.)

- **main_user_name:** The username that should be used to login to the CF UI as the administrator (will be created in the environment). e.g. "cfadmin"

- **main_user_password:** Password for the administrator user. e.g. "SuperS3cretPassw0rd"

- **bluemix_env_domain:** API and Management domain for the CF environment. Config for this domain in the DNS server will occur later. e.g. "cf.mydomain.local"

- **bluemix_app_domain:** The default shared domain to which applications are deployed. e.g. "cfapps.mydomain.local"

- **diego_cell_instances:** Number of Diego cells to deploy. Each Diego cell uses 4 vCPU, 32GB of RAM, and 300GB of disk space. e.g. "1"

- **external_dns:** Comma separated list of enterprise or lab domain name servers. e.g. "172.16.0.11,172.16.0.17"

- **ntp_servers:** Comma separated list of local ntp servers. e.g. "172.16.0.9"

- **vmware_address:** IP address of the VMware vCenter Server e.g. "10.1.212.26"

- **vmware_username:** Value username on the vCenter Server with permissions to create VMs on the Cluster, on the datastore, and using the network. e.g. "cfadmin". The vmware_username needs read access to the vCenter itself to be able to monitor the status of various deployment operations such as disk creation for the director VM.

- **vmware_password:** Password for vmware_username. e.g. "SuperS3cretPassw0rd"

- **datacenter_name:** The name of the datacenter on the vCenter server where the cluster was created e.g. "ICPLAB"

- **cluster_name:** Name of the cluster where CF VMs should be created. Cluster should have DRS enabled. e.g. "CloudFoundry".

- **resource_pool:** Name of resource pool to hold CF VMs. e.g. "ICP_CF_21"

- **vm_folder:** (optional) Name of folder on VMware Datastore where VMs should be created. e.g. "VMs"

- **template_folder:** (optional) Name of vSphere folder where stemcell VMs (templates) should be stored. e.g. "Templates"

- **disk_path:** The subdirectory on the datastore where VM persistent disks should be stored. e.g. "/icp-cf-210/Disks". This directory does not need to exist. It will be created if it does not exist.

- **datastore_pattern:** A pattern describing which datastore hosts the virtual machines. e.g. "CF_DS"

- **persistent_datastore_pattern:** A pattern describing which datastore hosts persistend disks (can be the same as datastore_pattern) e.g. "CF_DS"

- **subnet:** Subnet for VMs. e.g. "172.16.0.0/16"

- **address_range:** Range of IP addresses that are available for use in 'subnet'. e.g. "172.16.60.0-172.16.60.255"

- **portgroup:** The vmware network portgroup on which the VMs should communicate. e.g. "CF_net"

- **director_ip:** The IP address within the address_range which should be used for the director VM. e.g. "172.16.60.0"

Change configuration to install POC instance (or opposite to install production instance)

From the CF_HOME directory (e.g., `/opt/cf`), you should see a `cm` (config manager) executable.

NOTE: For ICP-CF v2.1.0.1 these state variables no long exist. Use the developer_mode: true attribute in the uiconfig file instead.

In ICP-CF v2.1.0.0, before each run of launch_deployment.sh you need to set the state variables as described below depending on whether you want a development (POC) or production deployment. (The development deployment is scaled back and does not support high availability.)

To set up a POC use the following `cm` commands:

```
./cm state -s cfpoc set --status READY # Install POC instance
```

```
./cm state -s cf set --status SKIP # Do not install production instance
```

```
./cm state -s diegopoc set --status READY # Install Diego POC instance
```

```
./cm state -s diego set --status SKIP # Skip the Diego Production instance
```

NOTE: To install a production instance, reverse the flags setting cf and diego to READY and cfpoc and diegopoc to SKIP. The default environment will do a production install.

Deploy the environment

NOTE: You may have to execute this command many times to get to completion. See troubleshooting section below.

```
cd /opt/cf ./launch_deployment.sh -c mycfdeployment.yml |tee deploy.log
```

- **-c:** Config file to use for this deployment
- **|tee deploy.log** Copy all output to deploy.log for later troubleshooting and redirect stderr to stdout to capture it as well.

An "error" may be reported that the gateway is inaccessible. If you can ping the gateway from the installation VM, then you can ignore that error message. Some gateways don't respond.

Configure the DNS

Create an `A` Record in the DNS pointing to the IP address of your ha_proxy (which was defined in your uiconfig-csplab.yml file) with a hostname of *.bluemix.csplab.local and another for *.bluemix.csplab.local. The DNS entries can be set to allow subdomains.

In Microsoft Active Directory:

- Highlight the domain (csplab.local in our example)
- Right click on the domain name and choose to add an "A" record
- In the hostname blank put `*.bluemix` and in the address blank use the IP address of the ha_proxy returned from the bosh command.
- Repeat and create an "A" record for `*.mybluemix` using the same IP address.

Deploy the buildpacks

Important: Deploying the buildpacks requires the access to the api server and should not be completed prior to making the DNS changes described above.

NOTE: It is recommended that the create_buildpacks.sh script be used for this step. The other option is to do:

```
./cm state -s buildpacks set --status READY relaunch the launch_deployment.sh
```

The recommended approach:

```
cd /opt/cf ./create_buildpacks.sh
```

2. You are now ready to use your CloudFoundy environment, the target for your cf commands is `http://api.<bluemix domain>`. e.g. `http://api.bluemix.csplab.local`.

Test your newly provisioned environment

1. Install the cloud foundry CLI to your local workstation

   Note: As of the time of this writing, setting roles for an org requires cf cli version 6.13. All other commands can use the latest version of the cf cli which is available.

   Go to https://github.com/cloudfoundry/cli/releases to find the release for your

workstation's architecture.

2. Install the certificate key for the api server

   - Linux

```
cat rootCA.pem >> /etc/ssl/certs/ca-certificates.crt
```

   - Mac

```
sudo security add-trusted-cert -d -r trustRoot -k /Library/Keychains/System.key
```

3. Configure your cf client's local

```
cf config --locale en-US
```

4. Login to CF using the credentials you provided in uiconfig.json. In our example we used `cfadmin/Passw0rd!`.
   `cf api http://api.bluemix.mydomain.local cf login -u cfadmin -p Passw0rd!`

Troubleshooting

If the installer fails it will output some information which is not very useful plus something like "bosh task 97 --debug" for more information (the 97 may be any number). bosh runs inside the inception container and so executing this command requires a bash shell on the container and not on the installer VM's local filesystem.

Your environment name is the value you used as the "-n" parameter to your launch.sh command (in this case "csplab").

```
connect.sh -name csplab
```

You are now running a bash shell on the inception vm. To get more information about the failed task execute:

```
bosh task <task number> --debug
```

Don't forget to type `exit` on the command line to exit the container shell when you are done.

To output debug information for the task and save the data to a file on the disk using a single command, try something like this:

```
docker exec -it $(docker ps -a |grep inception |cut -d' ' -f1) bosh task <97> --deb
```

Substitute the two <97>'s with the task ID output by the installer. This will write the debugging output to the screen and save it as a file named task97.log on your local filesystem for further scrutiny.

Checking the status of the install

View the file: /data//CloudFoundry/pie/states-to-deploy-cf.yaml

Each task shows a status which could be READY, FAILED, or SUCCESS.

If a task is in a SUCCESS state and you want to reinstall, edit the file and replace "SUCCESS" with "READY". Tasks in a FAILED state will automatically be reattempted on the next launch_deployment.sh execution.

Login to a specific bosh image

From the inception container execute "bosh ssh "
e.g. `bosh -d /data/CloudFoundry/cf-deploy-poc.yml ssh cc_core/0`
On the bosh container, data is in /var/vcap. Most calls will require running as root, so use `sudo <command>` or become root with `sudo su -`.

To find the credentials for various users

See `less /data/<env>/CloudFoundry/credentials.yml`

To execute bosh commands

```
bosh login
bosh target http://<director_ip>:25555
```

Show all bosh VMs and their IP addresses

From the inception container execute `bosh vms`

Trace cf calls: `export CF_TRACE=true`

Default vcap credentials: vcap/c1oudc0w

Troubleshoot director installation

connect to inception container

```
export BOSH_INIT_LOG_LEVEL=DEBUG
bosh-init deploy /data/gen-vmware_micro_boshinit.yml
```

It is not uncommon for the director VM disk creation to fail with a timeout. Normally disk creation happens in a relatively short time, e.g., about 10 seconds. If you see disk creation taking a long time as indicated by many dots in the stdout log, then there is a problem. The root cause of this problem is that the vmware_user specified for admin access the the CF cluster must also have read to vCenter in order to monitor the status of the disk deployment.

Using CloudFoundry

1. First, you much install the CloudFoundry client (cf) version 6.13
   https://github.com/cloudfoundry/cli/releases/tag/v6.13.0

2. Configure your client

   - Set your locale

   ```
   cf config --locale en-US --color true
   ```

   For Debugging, you can also set `-trace true`

3. Login to your environment

   - Specify your api target

   ```
   cf api https://api.cf.csplab.locale
   ```

- Login to your instance

```
cf login -u cfadmin -p Password!
```

important: to login with a self-signed certificate you should import the cert key (operation varies by operating system), or specify --skip-ssl-validation on the command line.

```
cf login -u cfadmin -p Password! --skip-ssl-validation
```

- Create a new org

```
cf create-org vhavard@us.ibm.com
```

- Create a new user

```
cf create-user vhavard@us.ibm.com Passw0rd!
```

- Make a user the manager of his/her org

```
cf set-org-role vhavard@us.ibm.com vhavard@us.ibm.com OrgManager
```

- Login as your new user to your new org

```
cf login -u vhavard@us.ibm.com -p Password! -o vhavard@us.ibm.com
```

- Create a new space in your new org and use it as your target

```
cf create-space dev
cf target -s dev
```

- Login to an api, org, and space with a single command

```
cf login -a https://api.cf.csplab.local -u vhavard@us.ibm.com -p Password! -o v
```

- See what buildpacks are available

```
cf buildpacks
```

- Get the node.js starter code from github

```
git clone https://github.com/IBM-Bluemix/get-started-node
```

- Get needed packages

```
cd get-started-node
npm install
```

- Start your getting started apps

```
node server.js
```

- Using a browser test your app at http://localhost:3000

- Push to bluemix public

```
cf login -a https://api.ng.bluemix.net -u vhavard@us.ibm.com -p myPassword -o v
cf push
```

- Push to bluemix private

```
cf login -a https://api.cf.csplab.local -u vhavard@us.ibm.com -p myPassword -o
cf push
```

- Set an envar in your environment

```
cf set-env LOCATION csplab-beta3
```