ICE Server Installation With Kubespray

Product guide for prerelease

Document version 1841, produced on Friday, September 06, 2024.

main 90adc8bf40040649230176bbdd465f6261a2d8e0

Contents

List of Tables

1 Document History

Publication Date	Product Release	Notes
May 29, 2024	3.5.1	Updated ICE Server version reference to 3.5.41629.
April 15, 2024	3.5.0	Updated ICE Server version to 3.5.41160. Updated supported Kubernetes version to 1.28.5. Updated 'https.json' section to include multiple FQDN. Updated processes to enable 'Recording Archiver' and 'telephony' by removing node labelling steps.
October 10, 2023	3.4.0	Updated X.509 certificate validity period to 397 days (13 months) or less. Added note to use VMware *EXSi*, instead of *vCenter Server*, to set VM latency sensitivity setting to 'Medium'.
September 20, 2023	3.4.0	Updated ICE Server version to 3.4.30527. Updated supported Kubernetes version to 1.27.x. Updated Kubespray version to 2.22.1. Added 'minio-multinode.json' section to 'Appendix A: Installation option files'.
July 27, 2023	3.3.0	Updated ICE Server version reference to 3.3.28975.
July 24, 2023	3.3.0	Added 'Kafka service scaling' section. Added `ice-archiver.json` section and updated `https.json` section in 'Appendix A: Installation option files'. Added explanation that `ldap.pem` file contains LDAP server `server.certificate` in 'Appendix B: Certificate management > LDAP Certificate' section. Added `8443/tcp` to 'Appendix C: Nginx load balancer example > Create Nginx configuration file' section.
February 13, 2023	3.2.0	Added 'Appendix D: Storage specification samples', added JSON sample to disable SMTP under the `ice-monitoring.json` section of 'Appendix A: Installation option files', updated the code block under the 'Install additional tools for ICE Server installation' section, updated VM latency sensitivity setting to 'Medium' (previously was 'High').

Publication Date	Product Release	Notes
January 31, 2023	3.2.0	Added 'Virtualization Guidance' section, updated virtualization guidance and VM creation information, added 'Arcus service scaling' section, updated multiple sections of 'Appendix A: Installation option files'.
January 19, 2023	3.2.0	Document created.

2 Introduction

This document is for installing ICE Server 3.5.1 on a three-node Kubernetes (K8s) cluster using Kubespray on on-premise Ubuntu 20 LTS servers with internet access.

If a single node Kubernetes cluster is desired, or if geo-redundancy is required, please refer to the ***ICE Server Installation Guide***.

3 Minimum requirements

This section discusses the installation of an Instant Connect Enterprise cluster in a single data center. To provide resiliency for node failures, install Kubernetes components on separate physical servers. If your cluster is deployed on virtual machines that are managed by a hypervisor, configure and distribute the virtual machines to ensure that a hardware failure does not pose a single point of failure for the entire cluster.

> **Note:** It is important to have proper time synchronization between all cluster nodes.

3.1 Nodes

A *node* is either a virtual machine or a physical computer running the Linux operating system that participates in your Kubernetes cluster.

The installation process described assumes that each node is running Ubuntu 20 LTS. Each of the three nodes must have identical computing resource (cpu/memory/disk) and configuration (filesystem layout, OS version, patch level, installed software packages, security setting, etc).

All nodes in your deployment must be accessible via SSH to the host computer being used to configure them, so SSH must be installed as an operating system component when installing Ubuntu 20 LTS.

3.2 Internet access

The process described in this guide requires all nodes be able to pull packages and Docker images from internet using https. In addition, all package repositories must be accessible.

For other available Kubespray installation types such as offine and bastion environment, please visit Kubespray Official Site.

3.3 Recommended configuration

For a three-node Kubernetes cluster, each VM/host in the cluster should have a minimum of:

- Ubuntu 20 LTS or higher
- 8 vcpus / 8 cores
- 28Gb memory
- 500Gb disk space
  - If `/var` is a separate filesystem, it should have at least 150Gb of storage
  - If `/opt` is a separate filesystem, it should have at least 150Gb of storage
  - Solid State Disk (SSD) is ***strongly*** recommended
  - If spinning disk (HDD) is used, the rating should be *at least* 10000rpm

3.4 Virtualization Guidance

Please consult the documentation for your virtualization platform and hardware for specific configuration recommendations.

- **Solid state drive (SSD):** Verify the host server's hard drives are sufficient. We *strongly* recommend using an SSD to host the ICE Server VM. ICE Server performance and stability are heavily dependent on (minimal) disk I/O latency, so an SSD is essential for successful deployment. Disk storage for the VM should be created by thick provisioning with highest shares allowed.

  Since traditional spinning drives (HDD) are not recommended, if a customer is using one, then we can only provide support if the ICE Server VM is the only VM running on it. The ICE Server VM cannot be competing with other VMs for disk access. Also, if any drive does not pass the disk performance (speed) test, which is conducted during installation, then the drive is not capable of running ICE Server and the installation will fail. This is likely to be an issue for non-SSD hard drives.

- **Memory:** Memory must not be oversubscribed. As a starting point, assume an amount equal to 25% of total VM memory usage should be allocated for memory overhead. For example, if there

are two VMs on a host, and each uses 32Gb of memory (64Gb in total), then allocating an additional 16Gb (at least) for overhead is prudent. Please consult VMware product documentation for guidance on appropriate memory overhead based on VM size and type.

- **Processing:** If hyperthreading, then the number of vcpus (cores) used must not be oversubscribed. As a starting point, assume an amount equal to 25% of total vcpus in use should be allocated for hyperthreading overhead. For example, if there are 2 VMs on a host, and each uses 8 vcpus (16 vcpus in total), then allocating an additional 4 vcpus (at least) for overhead is prudent. For hyperthreading, a minimum of 20 vcpus are recommended.
- **Hosting:** Avoid deploying the ICE Server VM on virtualization hardware which also hosts other I/O intensive VMs, e.g., database servers, otherwise the installation will fail.
- **VM snapshot:** Limit the ICE Server VM to two snapshots. Disk space equal to twice the size of the ICE deployment size should be available for the snapshots. For example, a medium sized ICE deployment of 750Gb, requires at least 1.5Tb of disk space for two snapshots. Delete snapshots after 48 hours. Follow this same guidance for other VMs sharing the same hardware. A build up of snapshots may result in adverse disk I/O performance due to lack of free disk space.
- **Virtual disk consolidation:** If the ESXi console displays the *VMware virtual machine disks consolidation is needed* error message, please address the situation.
- **Virtual disk write cache configuration:** When possible, always set the virtual disk write cache policy to `write-back`, rather than the default of `write-through`. Please consult VMware or your virtualization hardware vendor for additional information.
- **RAID controller firmware:** It is essential to keep the RAID controller firmware up-to-date. Please consult VMware or your virtualization hardware vendor for additional information.
- **Required VM settings:**
  - Compatibility = ESXi 6.5 (or later) virtual machine.

    > **Note:** Older versions are *not* recommended. Instant Connect does not support the use of obsolete hypervisors, such as VMware ESXi versions older than 6.5. Instant Connect cannot provide installation, performance, or server-related support to customers using these virtualization products.

  - Guest OS family = Linux
  - Guest OS version = Other Linux (64-bit)
  - CPU = Based on deployment size. More is better.
    * Cores per Socket = Select the appropriate value based on the virtualization hardware's CPU configuration. Please consult your VMware admin for more information.

    * Reservation = Select the highest available value.
    * Limit = Unlimited
    * Shares = High
  - Memory = Based on deployment size. More is better.
  - Hard disk = The storage cannot be resized once the VM is created, so verify the appropriate space is available before proceeding.
    * Disk Provisioning = Thick provisioned, eagerly zeroed
    * Shares = High. If 'Limit - IOPS' is configured, then set 'Shares' to the maximum allowed.
  - VM Options > Advanced > Latency Sensitivity = Medium

    > **Note:** If using VMware's *vCenter Server*, the 'Medium' option may not appear, so instead use their *ESXi* to access the VM and select 'Medium'.

4 Installing Kubernetes using Kubespray

4.1 Kubernetes server versions

ICE Server currently supports Kubernetes server version 1.28.5.

4.2 Pre-installation tasks

4.2.1 Administrator account

To configure each host in the cluster, we recommend creating a common, non-root administrator account on each target physical server (including the host system being used to administer the node of the cluster).

The administrator account must have `sudo` access on each target system you intend to add to your cluster. On each target system, log in as this superuser and then execute the following command to grant the user `sudo` access. For example, to allow the user `iceadmin` to be the administrator:

```
# Ubuntu
echo "iceadmin  ALL=(ALL) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/iceadmin
```

4.2.2 Designate a Network for Docker

`docker` is installed as part of the Kubespray process. By default, 172.17.0.0/16 is reserved for `docker`. If that conflicts with your local network, then designate another network for `docker`. One way to do this is to create a file called `/etc/docker/daemon.json` on each cluster node before beginning the Kubespray process. The following is an example file designating 107.0.0.0/8 for `docker`:

```
json
"live-restore": true,
"bip": "107.10.0.1/16",
"default-address-pools": [{
"base": "107.0.0.0/8",
"size": 16
}]
```

4.2.3 Management host for installing the cluster

A *management host* (referred to as `kubesprayhost` going forward) is an additional Ubuntu 20 LTS host to install your cluster. It is not a member node in the cluster.

> **Note:** We recommend that you use a host machine that is running macOS, Linux or WSL (Windows Subsystem for Linux) to perform the cluster installation.

> **Note:** The rest of the document is based on `kubesprayhost` running Ubuntu 20 LTS, using a service account.

4.2.4 Install required tools on `kubesprayhost`

4.2.4.1 Git `git` is a version control system that we use to clone the Kubespray repository. It is typically pre-installed on most Linux systems.

4.2.4.2 Docker Docker is an open platform that packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Install Docker with the following command:

```
sudo snap install docker
```

4.2.5 Define SSH trust to all cluster members on `kubesprayhost`

`kubesprayhost` must have SSH trust into all cluster member VMs. i.e., the SSH session from user `$ADMIN_USER` on `kubesprayhost` to all cluster VMs should be prompt-less. This could be accomplished by using `ssh-keygen` to establish the SSH trust as shown below.

There are variety of key types available in `ssh-keygen`, such as `rsa`. Hit `ENTER` to accept the default settings.

> **Note:** Do not specify a passphrase until after cluster is created.

```
ssh-keygen -t rsa
```

Then, use `ssh-copy-id` to establish the SSH trust from `kubesprayhost` to each cluster VM, by running the following command. `iceadmin` on all cluster VMs should only use a single, identical key type.

```
declare -a IPS=(${NODE_1_IP} ${NODE_2_IP} ${NODE_3_IP} ${KUBESPRAYHOST_IP
    })

for IP in ${IPS[@]}
do
        ssh-copy-id -o ConnectTimeout=10 ${IP}
done
```

Verify the connectivity by SSH into each host and get a host name, by running the following shell command:

```
for IP in ${IPS[@]}
do
        ssh -o ConnectTimeout=10 ${IP} hostname
done
```

Finally, `kubesprayhost` must be able to SSH to each cluster node as `root`. This is required during cluster creation only. On each cluster node:

```
sudo sh
su - root

scp iceadmin@KUBESPRAYHOST_IP:.ssh/id_rsa.pub ~/.ssh/authorized_keys
```

> **Note:** Once the Kubernetes cluster is created, `/root/.ssh/authorized_keys` should be removed on all cluster nodes.

4.2.6 Setting up Kubespray on `kubesprayhost`

The Kubespray tool is used install the Kubernetes cluster (see https://kubespray.io for basic instructions before proceeding).

> **Note:** The official Kubespray website updates the Kubespray process regularly for feature enhancements and product improvements. The process described in this document is current as of December 2022.

4.2.6.1 Cloning the Kubespray Repository The Kubespray software is distributed as a git repository.

To clone the Kubespray repository, follow these steps:

1. Establish a directory on `kubesprayhost` in which to store the Kubespray repository. This repository is used for cluster upgrades in the future, so we recommended that backing it up on a team drive or using version control. inventory/ice

```
mkdir -p /root/kube-install
```

2. Navigate to the directory and enter the following command to clone the current Kubespray repository:

```
cd /root/kube-install
git clone https://github.com/kubernetes-sigs/kubespray.git
```

3. Check out the v2.22.1 Kubespray version:

```
cd /root/kube-install/kubespray
git checkout 2cf23e3
```

4. Create a working directory for a cluster named `ice-cluster`:

```
/bin/cp -rfp inventory/sample inventory/ice-cluster
```

4.2.6.2 Generate the inventory's `hosts.yml` on `kubesprayhost` Create a softlink to the inventory file that will be generated later:

```
ln -s inventory/ice-cluster/hosts.yml hosts.yml
```

Create a Kubespray Docker container bash shell, then generate the inventory file:

```
docker run --rm -it \
        -v /root/kube-install/kubespray:/kubespray \
        -v /root/.ssh:/kubespray/.ssh \
```

```
        quay.io/kubespray/kubespray:v2.22.1 \
        bash

export CONFIG_FILE=inventory/ice-cluster/hosts.yml

python3 ./contrib/inventory_builder/inventory.py NODE_1_IP NODE_2_IP
   NODE_3_IP
```

Modified allowed Ansible version:

The quay.io docker image comes installed with Ansible 1.12.5, as of the time of this writing. However, the kubespray distribution requires the Ansible version to be between 1.14.0 and 1.16.0. Therefore, you must modify the `playbooks/ansible_version.yml` as follows:

- Change `minimal_ansible_version: 2.12.0` to `minimal_ansible_version: 2.14.0`
- Change `maximal_ansible_version: 2.13.0` to `maximal_ansible_version: 2.16.0`

Make sure the file is created with desired IP address:

```
cat inventory/ice-cluster/hosts.yml
```

A sample `hosts.yml` may look like this:

```
all:
  hosts:
    node1:
      ansible_host: 192.168.1.71
      ip: 192.168.1.71
      access_ip: 192.168.1.71
    node2:
      ansible_host: 192.168.1.72
      ip: 192.168.1.72
      access_ip: 192.168.1.72
    node3:
      ansible_host: 192.168.1.73
      ip: 192.168.1.73
      access_ip: 192.168.1.73
  children:
    kube_control_plane:
      hosts:
        node1:
        node2:
    kube_node:
      hosts:
        node1:
        node2:
        node3:
```

```
    etcd:
      hosts:
        node1:
        node2:
        node3:
    k8s_cluster:
      children:
        kube_control_plane:
        kube_node:
    calico_rr:
      hosts: {}
```

4.2.6.3 Review and adjust the inventory *Please review the IP address and hostnames carefully!*

You may change default hostnames/nodenames `node1`, `node2` and `node3` to match the desired hostname of each virtual machine, using a text editor.

> **Note:** Once the cluster is created, neither hostname nor IP address could be changed.

> **Note:** `hosts.yml` is white space sensitive. Do not alter any leading white space before each line.

4.2.6.4 Review and adjust cluster-specific configurations Optionally, review and customize cluster-specific configurations in the following directory:

`/root/kube-install/kubespray/inventory/ice-cluster/group_vars/`

These files contain the general configuration options:

`/root/kube-install/kubespray/inventory/ice-cluster/group_vars/all/all.yml`

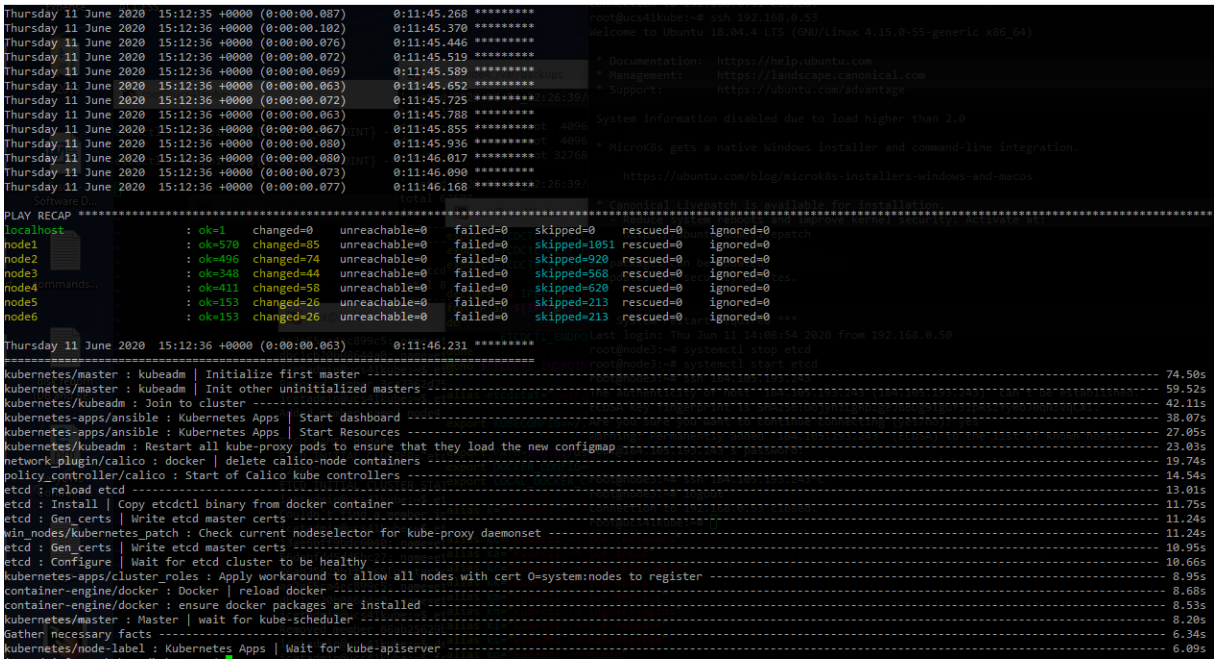
`/root/kube-install/kubespray/inventory/ice-cluster/group_vars/k8s_cluster/k8s-cluster.yml`

4.2.6.5 Run `ansible-playbook` on `kubesprayhost` to create the Kubernetes cluster

In the Kubespray Docker container's bash shell, create the cluster:

```
ansible-playbook -i inventory/ice-cluster/hosts.yml \
    --private-key=/kubespray/.ssh/id_rsa \
    -e kube_version=v1.28.5 \
    -vvv \
    -b \
    cluster.yml
```

Review the output under the 'PLAY RECAP' section. All status should be 'ok'. No red-colored text should be displayed.



If you need to start all over again, the cluster may be removed by using the `reset` playbook:

```
ansible-playbook -i inventory/ice-cluster/hosts.yml \
    --private-key=/kubespray/.ssh/id_rsa \
    -e kube_version=v1.28.5 \
    -vvv \
    -b \
    reset.yml
```

4.2.7 Post-installation tasks on `kubesprayhost`

4.2.7.1 KUBECONFIG To access the Kubernetes cluster using kubectl from `kubesprayhost`, the `KUBECONFIG` file is needed. Preferably, it should be using the default file name and location, `/root/.kube/config`, on `kubesprayhost`:

Create `/root/.kube` directory on `kubesprayhost`:

```
mkdir -p /root/.kube
```

Copy `KUBECONFIG` from any cluster node:

```
sudo scp \
  /etc/kubernetes/admin.conf \
  iceadmin@kubesprayhost:.kube/config
```

Make sure KUBECONFIG is secure on kubesprayhost:

```
chown -R iceadmin /root/.kube
chmod -R go-rwx /root/.kube
```

4.2.7.2 Verify Kubernetes cluster version Use kubectl to verify the Kubernetes version is 1.28.5:

```
$ kubectl version
Client Version: version.Info{Major:"1", Minor:"24", GitVersion:"v1.28.5",
    .....
Server Version: version.Info{Major:"1", Minor:"24", GitVersion:"v1.28.5",
    .....
```

4.2.7.3 Verify Kubernetes cluster state Use kubectl to ensure the Kubernetes cluster state is healthy. Pay particular attention to any pod that shows excessive number of restarts. Such typically indicates there is underlying connectivity and/or networking issue:

```
kubectl get nodes -o wide
kubectl get pods -o wide -A
```

4.2.7.4 Install additional tools for ICE Server installation

```
sudo snap install jq

sudo snap install helm --classic
```

4.2.7.5 Define the local storage class

4.2.7.5.1 Install Rancher local path provisioner StatefulSets require persistent volumes. Get a copy of the local storage class yaml from Rancher:

```
_TMP_URL=https://raw.githubusercontent.com/rancher/local-path-provisioner

wget ${_TMP_URL}/master/deploy/local-path-storage.yaml
```

local-path-storage.yaml, by default, uses /opt/local-path-provisioner directory as the dynamic storage path. You should review the directory setting, and change local-path-storage.yaml to match your installation.

If default directory location is acceptable, create a new filesystem or directory /opt/local-path-provisioner on all three cluster nodes.

```
sudo mkdir -p /opt/local-path-provisioner
```

> **Note:** *Indentation in a yaml file is significant.* Do *not* alter the indentation of the key/values in `local-path-storage.yaml`

After `local-path-storage.yaml` is reviewed, create the storage class. This command is to be executed only once:

```
kubectl create -f local-path-storage.yaml
```

4.2.7.5.2 Scale up local path provisioner

Since the cluster has three nodes, run the following command to scale the provisioner to match number of cluster nodes:

```
kubectl -n local-path-storage \
    patch deployment local-path-provisioner \
    -p '{"spec":{"replicas":3}}'
```

4.2.7.5.3 Set local path provisioner as default storage class

Typically, the Rancher local storage class should be the default local storage provisioner:

```
cat > patch_local_path_provisioner.json << EOF
{
  "metadata": {
    "annotations": {
      "storageclass.kubernetes.io/is-default-class": "true"
    }
  }
}
EOF

kubectl patch storageclass local-path -p "$(cat
   patch_local_path_provisioner.json)"
```

Verify the local storage class is ready and is the default:

```
kubectl get storageclass
```

5 ICE Server installation

> **Note:** The following ICE Server installation process applies to Azure AKS, Amazon EKS, or any other compatible cloud service.

5.1 ICE Server installation using Helm

On `kubesprayhost`, add ICE Server's Helm repository.

```
helm \
  repo \
  add \
  ice-release-helm \
  https://ic.repo.dillonkane.com:443/artifactory/ice-release-helm \
  --username instantconnect-customer \
  --password sazkax-jibzuc-5pEpgi
```

Then create the installation option files per *Appendix A: Installation option files* below, as needed.

Finally, run the `helm` command to install ICE Server:

```
helm -n ice-release \
    --create-namespace \
    upgrade -i ice-helm-operator \
    ice-release-helm/ice-helm-operator-release-3-5-1 \
    --version 3.5.41629 \
    -f ___INSTALL___OPTION___FILE___01 \
    -f ___INSTALL___OPTION___FILE___02 \
    -f ___INSTALL___OPTION___FILE___03 \
    ...
```

Please consult the ***ICE Server Administration Guide*** on how to monitor the installation progress.

> **Note:** Please contact ICE Technical Support for the latest valid list of release versions.

5.2 Post ICE Server installation task

5.2.1 Initial superuser password

The initial password for the ICE administration user account, `superuser@superuser.com`, may be obtained by the following command:

```
kubectl -n ice-arcus get secrets \
    init-superuser-pass \
    -o jsonpath \
    --template '{.data.pass}' \
    | base64 -d
```

5.2.2 Grafana admin password

The Grafana password for the `admin` user may be obtained by the following command:

```
kubectl -n ice-metrics get secrets \
    ice-release-ice-metrics-ice-monitoring-grafana \
    -o jsonpath \
    --template '{.data.admin-password}' \
    | base64 -d
```

5.2.3 Arcus service scaling

It is recommended to scale up the following deployments to match number of cluster worker nodes:

```
kubectl -n ice-arcus scale deployment client-bridge-dc1 --replicas=3
kubectl -n ice-arcus scale deployment platform-services-dc1 --replicas=3
```

5.2.4 Kafka service scaling

```
kubectl create job --from=cronjob/admin-kafka-scaler -n ice-arcus kafka-
    scaler
```

The `cronjob/admin-kafka-scaler` is already added to the `helm` chart. When either scaling up or down, the script behaves in the same way:

1. Checks the current active nodes in the Kubernetes cluster and then makes an appropriate number of Kafka brokers.
2. Distributes Kafka topics across the brokers. and creates replicas of every topic on every broker.
3. Execution takes about 6-7 minutes as the brokers are brought up or terminated, depending on is scaling up or down.

6 Appendix A: Installation option files

`___STRING___` values must be provided in each JSON file. Failure to supply valid values may cause fatal installation errors.

6.1 docker_repo.json

This file is used only when installing ICE Server using a local or air-gapped Docker repository. This file should be supplied to the `helm upgrade` command with `-f docker_repo.json`:

```
{
  "charts": {
```

```
    "iceInit": {
      "dockerConfigJsons": {
        "dockerHub": {
          ___CONTENT___FROM___DOCKER___CONFIG___JSON___
        }
      }
    }
  }
}
```

For example, use `kubesprayhost` to log into the local Docker repository. A file called `config.json` would be created. `___CONTENT___FROM___DOCKER___CONFIG___JSON___` may then be obtained by this command:

```
cat config.json | base64 -w 0
```

6.2 helm_repo.json

This file is used only when installing ICE Server using a local or air-gapped Helm repository. This file should be supplied to the `helm upgrade` command with `-f helm_repo.json`:

```
{
  "charts": {
    "iceInit": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "iceCassandra": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "strimziKafka": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "instantConnectEnterprise": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "rallypoint": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "iceIngress": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "iceMonitoring": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "iceMinio": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "iceElasticsearch": {
```

```
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    },
    "iceGateway": {
      "repository": "http://___HELM___REPO___SERVER___:___PORT___"
    }
  }
}
```

6.3 https.json

This file should be supplied to the `helm upgrade` command with `-f https.json`:

`___FQDN___` should be the FQDN to access the cluster and it *must* be a valid string in FQDN format, doing otherwise may cause an irreversible fatal installation error.

```
{
  "charts": {
    "iceDesktop": {
      "values": {
        "config": {
          "ingress": {
            "hosts": [
              "___FQDN1___",
              "___FQDN2___",
              "___FQDN3___" .....
            ]
          }
        }
      }
    },
    "iceIngress": {
      "values": {
        "hosts": [
          "___FQDN1___",
          "___FQDN2___",
          "___FQDN3___" .....
        ]
        "ingress-nginx": {
          "controller": {
            "config": {
              "use-proxy-protocol": false
            }
          }
        }
      }
    },
    "iceMinio": {
      "values": {
        "config": {
```

```
          "ingress": {
            "hosts": [
              "___FQDN1___",
              "___FQDN2___",
              "___FQDN3___" .....
            ]
          }
        }
      }
    },
    "iceMonitoring": {
      "values": {
        "grafana": {
          "grafana.ini": {
            "server": {
              "domain": "___FQDN1___"
            }
          },
          "ingress": {
            "enabled": true,
            "hosts": [
              "___FQDN1___",
              "___FQDN2___",
              "___FQDN3___" .....
            ]
          }
        }
      }
    },
    "instantConnectEnterprise": {
      "values": {
        "config": {
          "ingress": {
            "hosts": [
              "___FQDN1___",
              "___FQDN2___",
              "___FQDN3___" .....
            ]
          }
        }
      }
    },
    "kafkaUI": {
      "values": {
        "kafka-ui": {
          "ingress": {
            "enabled": true,
            "host": "___FQDN1___"
          }
        }
      }
    }
```

```
    },
    "rallypoint": {
      "values": {
        "ingress": {
          "hosts": [
            "___FQDN1___",
            "___FQDN2___",
            "___FQDN3___" .....
          ]
        }
      }
    }
  }
}
```

> **Note:** As seen in the full `https.json` above, some entries do NOT accept multiple FQDN values, e.g. `iceMonitoring`, `kafkaUI`.

Example 1: For a single FQDN…

```
{
  "charts": {
    "iceDesktop": {
      "values": {
        "config": {
          "ingress": {
            "hosts": [
              "my-ice-cluster.mydomain.com"
            ]
          }
        }
      }
    },
    ...
```

Example 2: For more than one FQDN, list them as comma-separated values…

```
{
  "charts": {
    "iceDesktop": {
      "values": {
        "config": {
          "ingress": {
            "hosts": [
              "my-ice-cluster.mydomain.com",
              "my-alt-ice-cluster.mydomain.com"
            ]
          }
        }
      }
```

```
    },
    ...
```

6.4 ice-archiver.json

To enable Recording Archiver, supply this file to the `helm upgrade` command with `-f ice-archiver.json`:

```
{
  "charts": {
    "rallypoint": {
      "values": {
        "archiver": {
          "enabled": true
        }
      }
    }
  }
}
```

If the cluster is not using https (i.e., `https.json` is not used), and the cluster is behind a firewall, then the public-facing IP address of the cluster must be defined:

```
{
  "charts": {
    "rallypoint": {
      "values": {
        "archiver": {
          "enabled": true,
          "config": {
            "recordingURL": "http://___PUBLIC___FACING___CLUSTER___IP___"
          }
        }
      }
    }
  }
}
```

> **Note:** The `___PUBLIC___FACING___CLUSTER___IP___` defaults to the cluster node's IP address, if https is not used and `___PUBLIC___FACING___CLUSTER___IP___` is not provided in `ice-archiver.json`.

6.5 ice-monitoring.json

This file should be supplied to the `helm upgrade` command with `-f ice-monitoring.json` :

```
{
  "charts": {
    "iceMonitoring": {
      "values": {
        "promtail": {
          "extraVolumes": [
            {
              "name": "varlog",
              "hostPath": {
                "path": "/var/log"
              }
            }
          ],
          "extraVolumeMounts": [
            {
              "name": "varlog",
              "mountPath": "/host/var/log",
              "readOnly": true
            }
          ],
          "config": {
            "snippets": {
              "extraScrapeConfigs": "- job_name: system\n  static_configs
                 :\n  - targets:\n      - localhost\n    labels:\n
                 job: varlogs\n      __path__: /host/var/log/*.log\n"
            }
          }
        },
        "grafana": {
          "grafana.ini": {
            "smtp": {
              "enabled": true,
              "from_address": "___CLUSTER___EMAIL___ADDRESS___",
              "from_name": "___EMAIL___USER___COMMON___NAME___",
              "host": "___SMTP___SERVER___: ___SMTP___PORT___",
              "skip_verify": true,
              "user": "apikey",
              "password": "___EMAIL___API___KEY___"
            }
          }
        }
      }
    }
  }
}
```

For cloud-native deployments, such as Amazon EKS or Azure AKS, then remove the `promtail` section, as in the following JSON file sample:

```
{
  "charts": {
    "iceMonitoring": {
      "values": {
        "grafana": {
          "grafana.ini": {
            "smtp": {
              "enabled": true,
              "from_address": "___CLUSTER___EMAIL___ADDRESS___",
              "from_name": "___EMAIL___USER___COMMON___NAME___",
              "host": "___SMTP___SERVER___: ___SMTP___PORT___",
              "skip_verify": true,
              "user": "apikey",
              "password": "___EMAIL___API___KEY___"
            }
          }
        }
      }
    }
  }
}
```

For cloud-native deployments, such as Amazon EKS or Azure AKS, if SMTP email notification is not being used, then remove the `smtp` section to disable SMTP, as in the following JSON file sample:

```
{
  "charts": {
    "iceMonitoring": {
      "values": {
        "grafana": {
          "grafana.ini": {
            "smtp": {
              "enabled": false
            }
          }
        }
      }
    }
  }
}
```

6.6 ice-reflector.json

This file should be supplied to the `helm upgrade` command with `-f ice-reflector.json`:

`___MULTICAST___INTERFACE___` should be the network interface to be used for static reflector

traffic.

```
{
  "charts": {
    "rallypoint": {
      "values": {
        "reflector": {
          "config": {
            "multicastInterfaceName": "___MULTICAST___INTERFACE___"
          }
        }
      }
    }
  }
}
```

It is typical to set `___MULTICAST___INTERFACE___` to the default interface. On Ubuntu, it is usually `ens160`. You may verify the available interface(s) by running `ip a`, or find the default interface using `route` command:

```
sudo apt install net-tools -y

route | grep default | awk '{print $8}'
```

For cloud-native deployment, static reflector should be turned off. Please use the following `ice-reflector.json` instead:

```
{
  "charts": {
    "rallypoint": {
      "values": {
        "reflector": {
          "enabled": false
        }
      }
    }
  }
}
```

6.7 ice-telephony.json

To enable telephony, supply this file to the `helm upgrade` command with `-f ice-telephony.json`:

```
{
  "charts": {
    "iceGateway": {
      "enabled": true,
```

```
      "values": {
        "telephony": {
          "pjsipTLS": {
            "enabled": false
          }
        }
      }
    }
  }
}
```

If TLS/SRTP is required, please review the 'Telephony Certificate' section of *Appendix B - Certificate management* below, and use the following `ice-telephony.json`:

```
{
  "charts": {
    "iceGateway": {
      "enabled": true,
      "values": {
        "telephony": {
          "pjsipTLS": {
            "enabled": true
          }
        }
      }
    }
  }
}
```

6.8 minio-multinode.json

This file should be supplied to the `helm upgrade` command with `-f minio-multinode.json` :

```
{
  "charts": {
    "iceMinio": {
      "values": {
        "multinode": {
          "enabled": true
        }
      }
    }
  }
}
```

6.9 minio_secret.json

This file should be supplied to the `helm upgrade` command with `-f minio_secret.json`. It is used only when installing ICE Server for the first time. It will be ignored afterward.

`___DEFAULT___MINIO___ACCESS___SECRET___` should be a random password for accessing Minio service during a fresh installation. This file should be deleted after installation is completed.

```
{
  "charts": {
    "iceInit": {
      "values": {
        "minio": {
          "rootUser": {
            "accesskey": "minio",
            "secretkey": "___DEFAULT___MINIO___ACCESS___SECRET___"
          }
        }
      }
    }
  }
}
```

If it is necessary to reset the minio service secret, you must first remove the existing secret:

```
kubectl -n ice-minio exec \
  $(kubectl -n ice-minio get pod -l app=minio -o=jsonpath='{.items..
     metadata.name}') \
  -c minio -- mv data/.minio.sys/config data/.minio.sys/config.old

kubectl -n ice-arcus delete secret minio-access-secret
kubectl -n ice-cassandra delete secret minio-access-secret
kubectl -n ice-kafka delete secret minio-access-secret
kubectl -n ice-metrics delete secret minio-access-secret
kubectl -n ice-minio delete secret minio-access-secret
kubectl -n ice-rallypoint delete secret minio-access-secret
```

Then re-run the full `helm upgrade` command with `-f minio_secret.json`.

6.10 multinode.json

This file should be supplied to the `helm upgrade` command with `-f multinode.json`:

```
{
  "charts": {
    "rallypoint": {
      "values": {
        "rallypoint": {
```

```
          "replicas": 3
        }
      }
    },
    "instantConnectEnterprise": {
      "values": {
        "config": {
          "cassandra": {
            "nCassandraNodes": 3,
            "replicationFactor": 3
          },
          "elasticsearch": {
            "replicas": 3
          },
          "kafka": {
            "kafka": {
              "replicas": 3
            }
          }
        }
      }
    }
  }
}
```

7 Appendix B: Certificate management

> **Note:** All X.509 certificates used for Instant Connect must expire in 397 days (13 months) or less. This includes server certificates, intermediate CA, root CA, etc. Certificates whose expiration dates exceed this validity period will not be accepted by Instant Connect clients, resulting in 'Cannot connect to server' error messages.

7.1 ICE Server Certificate

First, create a `mysite.pem` file by concatenating the server certificate, the intermediate CA certificate (if applicable), and the root CA certificate:

```
cat ___SERVER___.cer ___INTERMEDIATE___CA___.cer ___ROOT___CA___cer >
   mysite.pem
```

Then, create a **`private.key`** file by coping the server certificate's corresponding private key:

```
cp -p ___SERVER___.key private.key
chmod 0600 private.key
```

Finally, supply the ICE Server with both files:

```
kubectl -n ice-release \
  create secret tls \
  defaultcert \
  --cert=mysite.pem \
  --key=private.key

kubectl -n ice-arcus \
  create secret tls \
  arcus-tls \
  --cert=mysite.pem \
  --key=private.key
```

If you need to amend the certificates, or if you need to renew the certificates:

```
kubectl -n ice-release delete secret defaultcert \
    && kubectl -n ice-release \
          create secret tls \
          defaultcert \
          --cert=mysite.pem \
          --key=private.key

kubectl -n ice-arcus delete secret arcus-tls \
    && kubectl -n ice-arcus \
          create secret tls \
          arcus-tls \
          --cert=mysite.pem \
          --key=private.key
```

7.2 LDAP Certificate

If the LDAP server is using public/well-known CA, no further action is needed.

For connection to LDAP server `___LDAP___SERVER___` which uses self-signed or Enterprise CA, first create a .pem file locally:

```
openssl s_client -showcerts -connect ___LDAP___SERVER___:636 < /dev/null \
    | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' | tee ldap.pem
```

Then apply the `ldap.pem` file, which contains the LDAP server `server.certificate`, to the ICE Server:

```
kubectl -n ice-arcus \
    create configmap extra-cacerts \
    --from-file=ldap.pem \
    -o yaml \
    --dry-run=client \
```

```
    | kubectl -n ice-arcus replace -f -
```

7.3 Telephony Certificate

Please refer to Enable TLS/SRTP section of ICE Telephony Administration Guide on how to generate `icegwkey.pem` and `icegw.pem`.

Then, apply the .pem to the ICE Server:

```
kubectl -n ice-arcus \
    create configmap gateway-certs \
    --from-file=icegwkey.pem \
    --from-file=icegw.pem
```

If you need to amend the certificate, or if you need to renew the certificate, delete the existing certificate configuration first:

```
kubectl -n ice-arcus delete configmap gateway-certs

kubectl -n ice-arcus \
    create configmap gateway-certs \
    --from-file=icegwkey.pem \
    --from-file=icegw.pem
```

8 Appendix C: Nginx load balancer example

Each worker node handles all application requests including application authentication. If the cluster is behind a firewall, or for a more seamless application access experience, an extra load balancer can be deployed. There are many ways to do this, the following is a very simple example.

8.1 Create Nginx configuration file

Create a basic Nginx configuration file for a three workers Kubernetes cluster running ICE Server. Add your own customizations as needed per Nginx documentation.

```
error_log stderr notice;
worker_processes auto;
worker_rlimit_nofile 130048;
worker_shutdown_timeout 10s;
events {
  multi_accept on;
  use epoll;
  worker_connections 16384;
```

```
}
stream {
    upstream worker_80 {
        least_conn;
        server IP___OF___WORKER___ONE:80;
        server IP___OF___WORKER___TWO:80;
        server IP___OF___WORKER___THREE:80;
    }
    upstream worker_443 {
        least_conn;
        server IP___OF___WORKER___ONE:443;
        server IP___OF___WORKER___TWO:443;
        server IP___OF___WORKER___THREE:443;
    }
    upstream worker_7443 {
        least_conn;
        server IP___OF___WORKER___ONE:7443;
        server IP___OF___WORKER___TWO:7443;
        server IP___OF___WORKER___THREE:7443;
    }
    upstream worker_8443 {
        least_conn;
        server IP___OF___WORKER___ONE:8443;
        server IP___OF___WORKER___TWO:8443;
        server IP___OF___WORKER___THREE:8443;
    }
    server {
        listen 80;
        proxy_pass worker_80;
        proxy_protocol on;
    }
    server {
        listen 443;
        proxy_pass worker_443;
        proxy_protocol on;
    }
    server {
        listen 7443;
        proxy_pass worker_7443;
    }
}
```

8.2 Install Docker

Install Docker on the load balancer host. Consult the official Docker website for more information.

> **Note:** *For RHEL.* Setting SELinux to `Permissive` is recommended. Consult with your RHEL systems administrator for more information.

8.3 Run Nginx in Docker

Assuming the Nginx configuration file is saved in your home directory as `my_k8s_lb.conf`, start Nginx on the load balancer host by running the following Docker command:

```
docker run --ulimit nofile=130048:130048 \
        --net=host \
        -v ~/my_k8s_lb.conf:/etc/nginx/nginx.conf \
        --restart=always \
        --name=nginx \
        -d nginx:latest
```

Once ICE Server is installed, you can login using the load balancer's IP address (non-https) or FQDN name (https).

9 Appendix D: Storage specification samples

Specify an appropriate amount of storage space based on the size of your ICE deployment (more is *always* better). Refer to the JSON samples below and also consult the documentation for your virtualization platform and hardware for specific configuration recommendations.

Storage type	Small	Medium	Large
`instantConnectEnterprise`	10Gb	30Gb	50Gb
`iceElasticsearch`	30Gb	100Gb	200Gb
`iceMinio`	70Gb	150Gb	270Gb

9.1 Small

```
{
  "charts": {
    "instantConnectEnterprise": {
      "values": {
        "config": {
          "cassandra": {
            "dataCapacity": "10Gi"
          }
        }
      }
    },
    "iceElasticsearch": {
```

```
      "values": {
        "elasticsearch": {
          "storage": "30Gi"
        }
      }
    },
    "iceMinio": {
      "values": {
        "config": {
          "size": "70Gi"
        }
      }
    }
  }
}
```

9.2 Medium

```
{
  "charts": {
    "instantConnectEnterprise": {
      "values": {
        "config": {
          "cassandra": {
            "dataCapacity": "30Gi"
          }
        }
      }
    },
    "iceElasticsearch": {
      "values": {
        "elasticsearch": {
          "storage": "100Gi"
        }
      }
    },
    "iceMinio": {
      "values": {
        "config": {
          "size": "150Gi"
        }
      }
    }
  }
}
```

9.3 Large

```
{
  "charts": {
    "instantConnectEnterprise": {
      "values": {
        "config": {
          "cassandra": {
            "dataCapacity": "50Gi"
          }
        }
      }
    },
    "iceElasticsearch": {
      "values": {
        "elasticsearch": {
          "storage": "200Gi"
        }
      }
    },
    "iceMinio": {
      "values": {
        "config": {
          "size": "270Gi"
        }
      }
    }
  }
}
```