HCIBench User Guide

Table of Contents

Introduction

Evaluating performance is an important part of considering any storage solution. Higher performing solutions can support more workloads on a given configuration, better accommodate application, minimize potential performance problems, as well as be more cost-effective. There are strong motivations to prefer higher performing solutions to lesser alternatives.

Unfortunately, obtaining directly comparable performance results from publicly available information is difficult at best. There is an infinite variety of potential test scenarios—and many vendors discourage publishing for marketing and competitive reasons.

This leaves IT professionals in the position of having to run their own tests and interpreting the results. This has long been a standard practice in evaluating external storage arrays, but the newer generation of hyper-converged solutions – such as VMware vSAN™—presents new testing challenges.

In a hyper-converged architecture, each server is intended to support both many application VMs, as well as contribute to the pool of storage available to applications. This is best modeled by invoking many dozens of test VMs, each accessing multiple stored VMDKs. The goal is to simulate a very busy cluster.

Unfortunately, popular storage performance testing tools do not directly support this kind of model. To achieve a simulation of a busy production cluster, much effort is required to automate load generation, monitoring and data collection after the fact. These steps waste so much valuable time available to do actual testing, even worse may introduce errors into the process.

To address this situation, VMware released a storage performance testing automation tool—HCIBench—that automates the use of the popular Vdbench and Fio as testing tools in larger clusters. Users simply specify the testing parameters they would like to run, and HCIBench instructs these workload generators what to do on every node in the cluster.

HCIBench aims to simplify and accelerate customer Proof of Concept (POC) performance testing in a consistent and controlled way. This tool fully automates the end-to-end process of deploying test VMs, coordinating workload runs, aggregating test results, and collecting necessary data for troubleshooting purposes. Evaluators choose the profiles they are interested in; HCIBench does the rest quickly and easily.

This tool is provided free of charge and with no restrictions. Support will be provided solely on a best-effort basis as time and resources allow, can be found here on Github.

Per the VMware EULA, users who want to publicly share their testing results are requested to submit their hardware configuration, methodology, parameter file and test results for review before publication at hcibench@broadcom.com.

We will make every effort to get back to you quickly.

Overview

HCIBench Tool Architecture

HCIBench is specifically designed for running performance tests against shared or local datastore(s) in vSphere. It generates a test workload using either Vdbench or Fio. HCIBench is delivered in the form of an Open Virtualization Appliance (OVA).

The Controller VM contains the following components.

- Ruby vSphere Console (RVC)
- GOVC
- Docker
    - Graphite Container: 1.1.5-10
    - Grafana Container: 8.3.11
    - InfluxDB Container: 1.8.1-alpine
- vSAN Observer
- Automation bundle
- Configuration files
- Fio binary: 3.30
- Photon guest VM template: 3.0

The Controller VM has all the needed components installed. The core component is RVC (https://github.com/vmware/rvc) with some extended features enabled. RVC is the engine of this performance test tool, responsible for deploying guest VMs, conducting Vdbench or Fio runs, collecting results, and monitoring vSAN by using vSAN Observer.

Starting from HCIBench 2.6, GOVC(https://github.com/vmware/govmomi/tree/main/govc) was added as another component to invoke vCenter API, going forward, with more functions added to GOVC, the core engine of HCIBench will be moved from RVC to GOVC eventually.

During the installation process, you need to download the Vdbench binaries directly from the Oracle website one time only if you choose Vdbench as the workload generator. While the use of Vdbench is unrestricted, Oracle does not provide redistribution rights in their license. If you choose to use Fio, you don't need to do anything because we already have the Fio binary included

The automation bundle, consisting of Ruby and Bash scripts, is developed to modularize features such as test VM deployment, VMDK initialization, and Vdbench or Fio runs, as well as automate and simplify the entire testing process. The automation bundle reads user-defined configuration information about the test environment and the target workload profile, then interacts with RVC as necessary to carry out the following tasks:

- Connect to the vSphere environment to be tested. The tool itself can be deployed in a separate vSphere environment but must have access to the target cluster that will be tested.

- Deploy Photon guest VMs in the target cluster based on user input of the number of guest VMs and the number of virtual disks per VM.
- Optionally prepare each virtual disk to initialize storage, a similar way to “thick provisioning eager zero” or sequentially writing to storage before benchmarking to avoid first write penalty.
- Optionally drop read cache and write buffer if testing against vSAN OSA datastore.(vSAN configured with two tiers)
- Optionally configure multi-write virtual disks for the worker VMs.
- Transfer workload parameter file to each guest VM. The parameter file defines the target workload and runtime specification.
- Start vSAN Observer before testing and generate vSAN statistics upon test completion.
- Kick off Vdbench or Fio instances against each virtual disk on each guest VM and run for the defined duration.
- Collect vmkstats after 30 minutes of testing started for one minute and collect vm-support bundle from all ESXi hosts if **vSAN Debug Mode** is used.
- Collect and aggregate Vdbench or Fio performance data.

Figure 1 shows the architecture of the tool and its components.

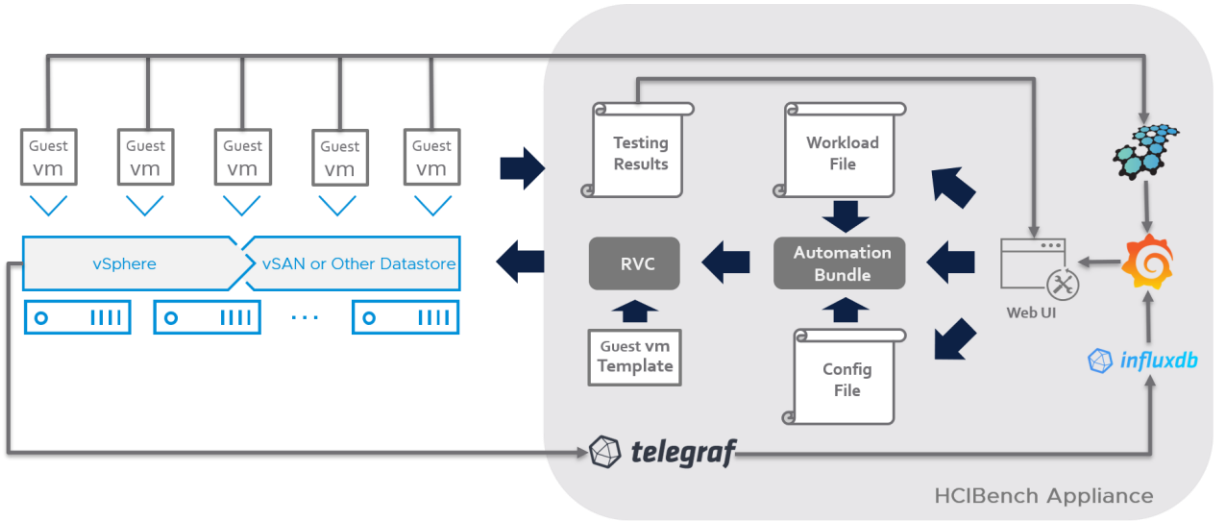


Figure 1. HCIBench VM Specification

Controller VM

- CPU: 8 vCPU
- RAM: 8GB
- OS VMDK: 16GB
- Data VMDK: 200GB
- Operating system: Photon OS 3.0
- OS Credential: user is responsible for creating the root password when deploying the VM.
- Software installed: Ruby 2.5.4, Rubygem 2.7.6.2, Rbvmomi 2.4.1, RVC 1.8.0, sshpass 1.06, GOVC 0.23.0, Apache 2.4.18, Tomcat 8.54, OpenJDK 1.8.0-internal, Fio 3.16, Graphite 1.15, Grafana 8.3.11, Python 3.7.4

Guest VM

- CPU: 4 vCPU as default

- RAM: 8 GB as default
- OS VMDK: 16GB
- OS: Photon OS 3.0
- OS Credential: root/vdbench
- Software installed: OpenJDK 1.8.0-internal, Python 3.7.4
- SCSI Controller Type: VMware Paravirtual
- Data VMDK: number and size to be defined by user

Installation and Configuration

Prerequisites

Before deploying HCIBench the environment must meet the following requirements:

- The cluster is created and configured properly
- The network that will be used by the Guest VM is defined on all the hosts in the cluster. If a DHCP service is available, the Guest VM can obtain their network configurations from the DHCP server. If the network does not have DHCP service or an insufficient number of IP addresses HCIBench can assign static IP address. To accomplish this, the HCIBench source network "VM Network" must be mapped to the same network as the guest VM (Fig. 2)
- The vSphere environment where the tool is deployed can access the vSAN Cluster environment to be tested

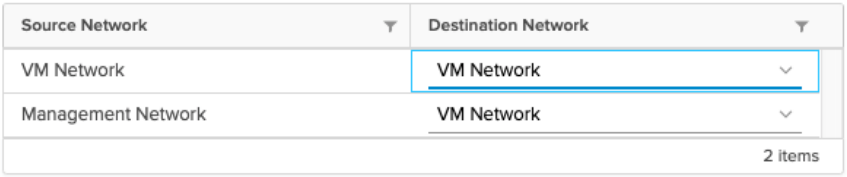


Figure 2. Source Networks

Tool Installation

1. In vCenter, select Deploy OVF Template then enter either the URL to the OVA or select a local copy of the **HCIBench 2.8.1.ova** from the client system.

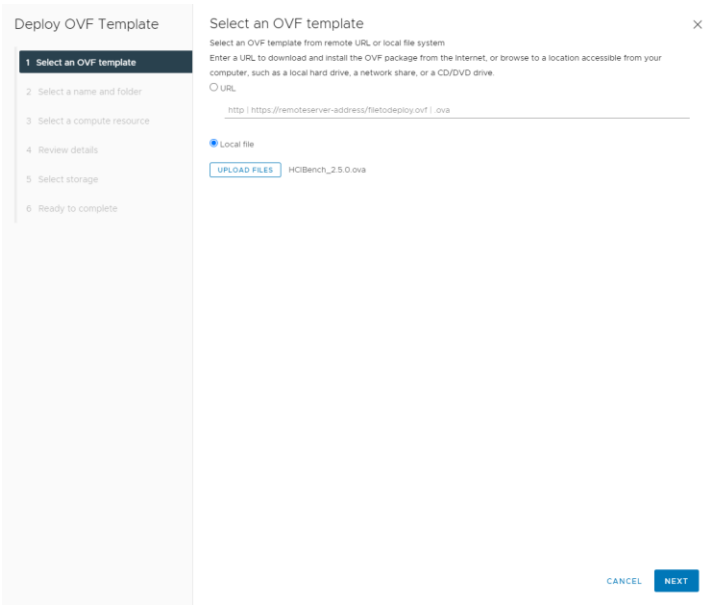


Figure 3. Select an OVF Template

2. When prompted for a name leave the default name or enter a new name, then select a location for the appliance.

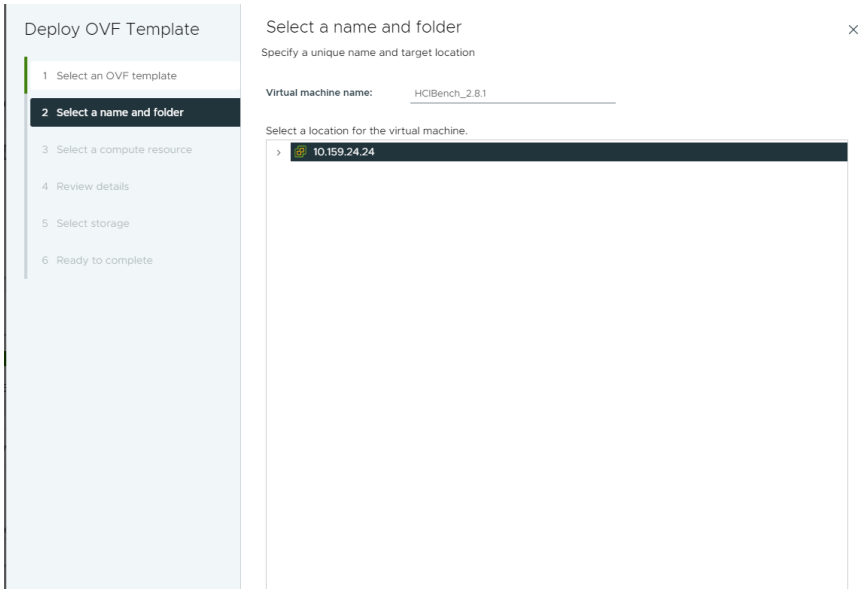


Figure 4. Select Name and Folder

3. Next, select the cluster where the HCIBench appliance should be deployed.

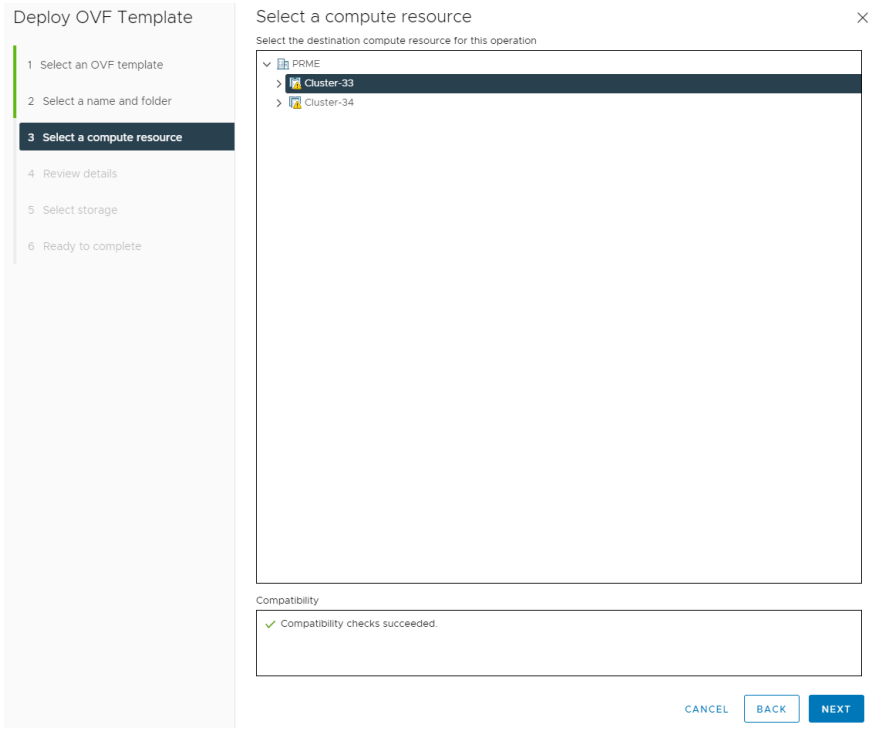


Figure 5. Select Compute Resource

4. Review the deployment details.

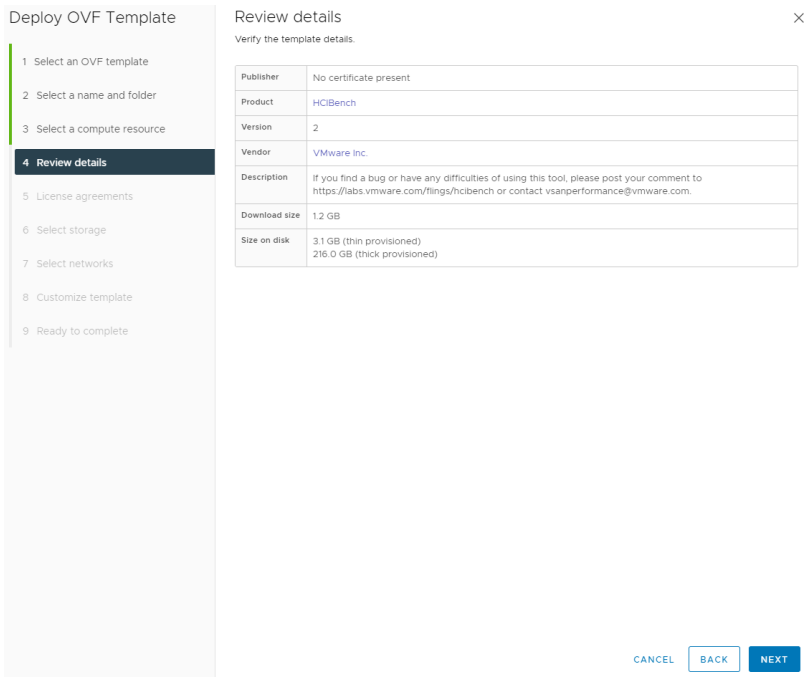


Figure 6. Review Details

5. Review and accept the license agreement.

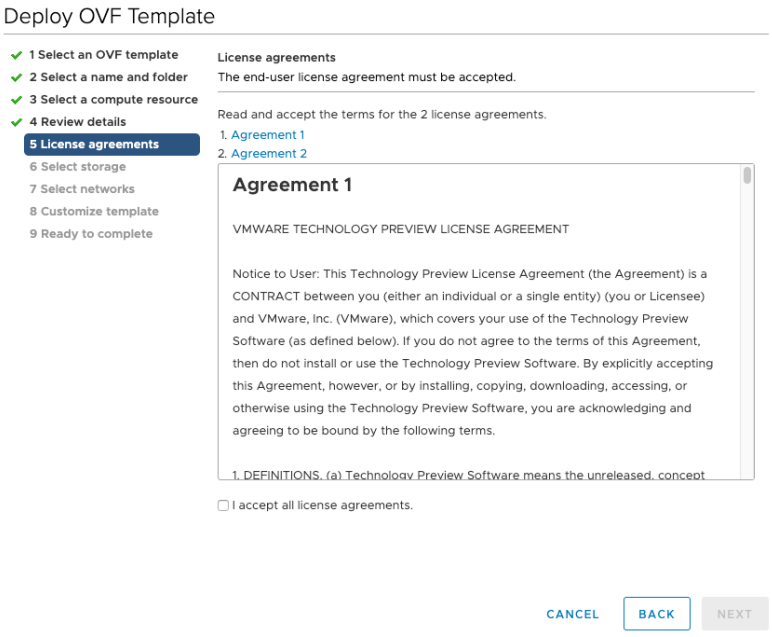


Figure 7. License Agreement

6. Select the storage and storage policy for the appliance. HCIBench does not generate a substantial amount of I/O during testing so it can reside on the datastore being tested.

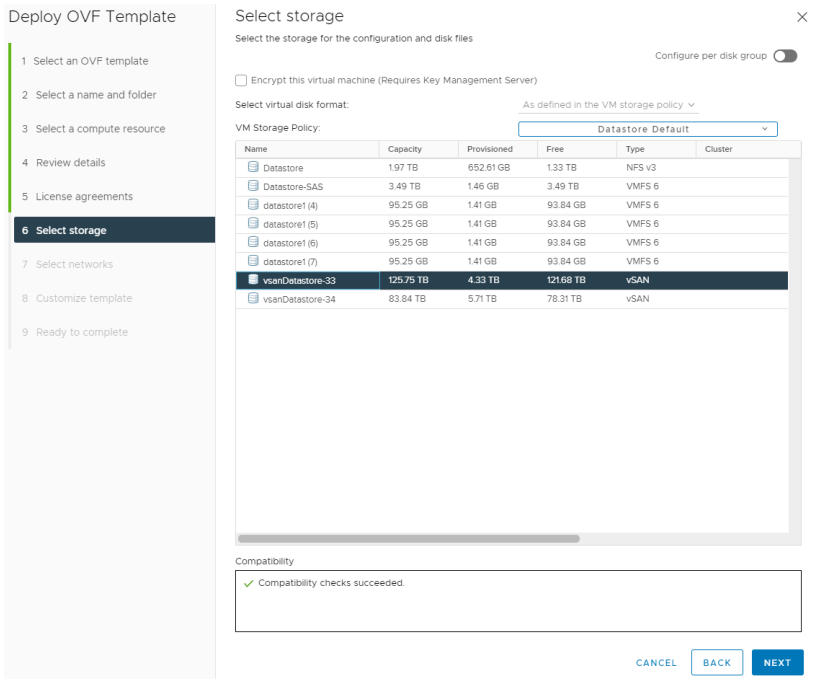


Figure 8. Select Storage and Policy

7. Map the “Management Network” to the network which the HCIBench will be accessed through; if the network prepared for Guest VM doesn’t have DHCP service, map the “VM Network” to the same network, otherwise ignore the “VM Network”.

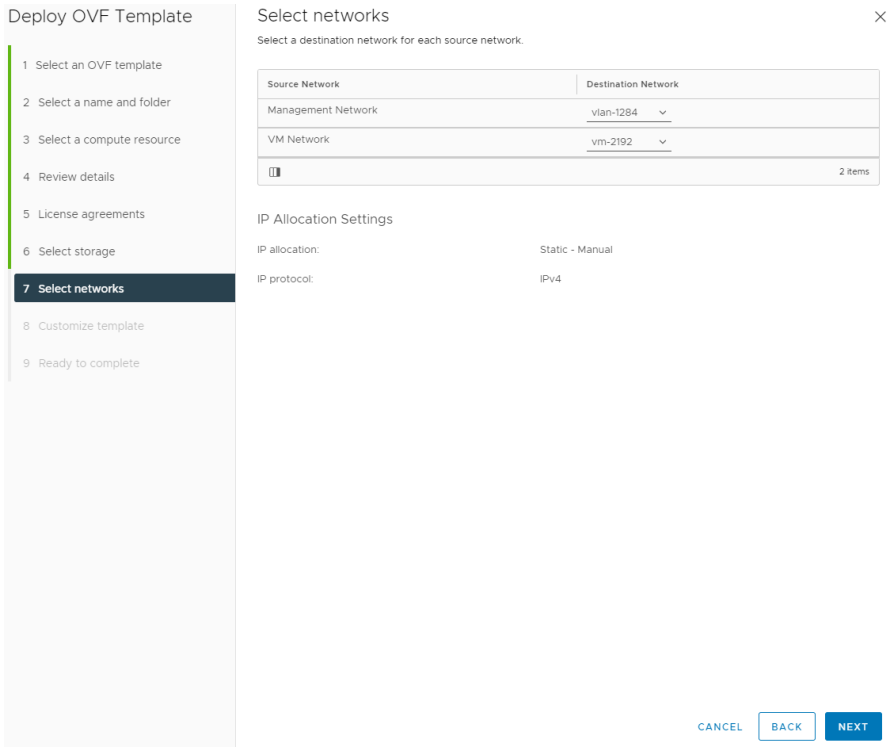


Figure 9. Map Networks

8. On the customize template enter a system password for HCIBench. If the HCIBench management interface should use DHCP the network information should be left blank. If HCIBench should use a specific address, select static on the management network then enter desired network configuration.

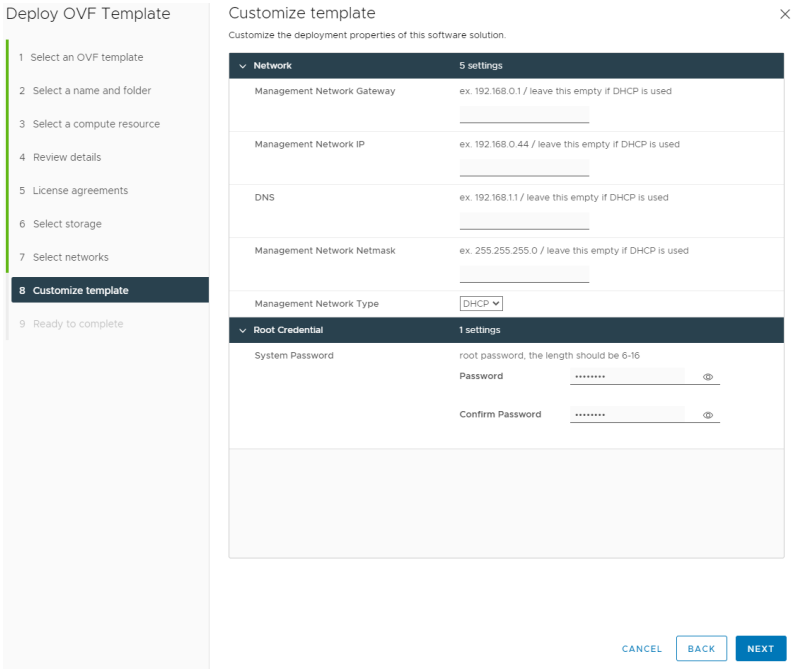


Figure 10. Configure Management Network and System Password

9. Review the configuration and click finish.

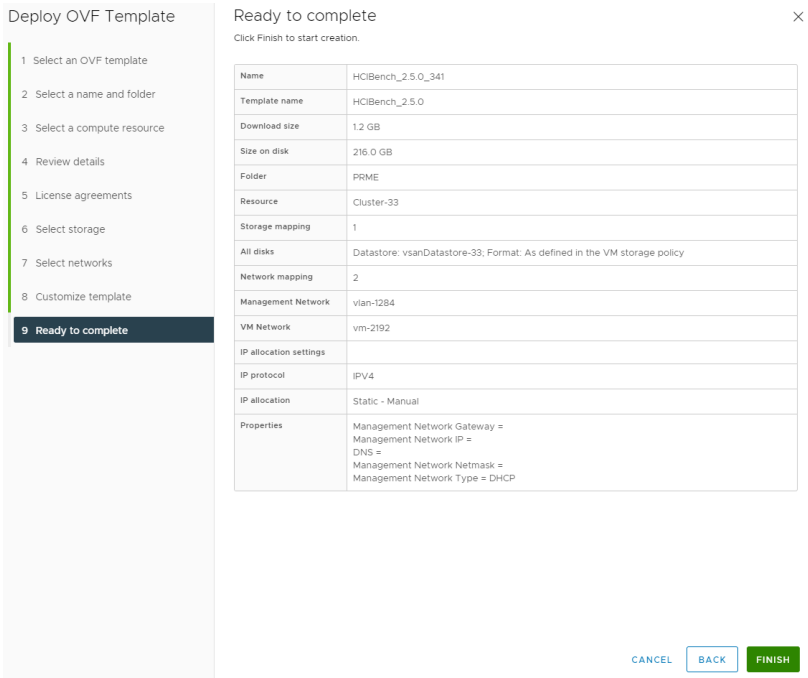


Figure 11. Review and Start Deployment

Test Configuration

After deployment, you can navigate to https://HCIBench_IP:8443/ to start configuration and kick off the test. Before accessing the configuration page, the root user ID and password must be used to authenticate to prevent unauthorized access to HCIBench.

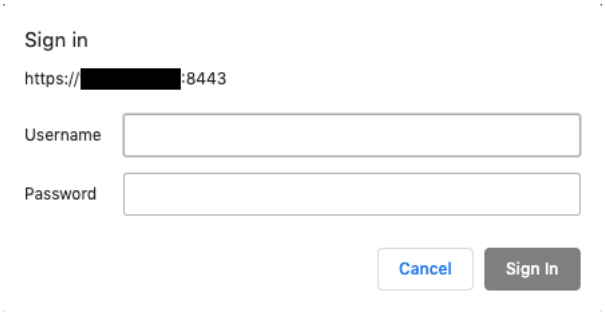


Figure 12. HCIBench Login

There are four main sections in this configuration Page:

vSphere Environment Information

In this section, all fields not marled "**OPTIONAL**" are required. You must provide the vSphere environment information where the target Cluster is configured, including vCenter IP address, vCenter credential, name of the datacenter, name of the target cluster, and name of the Datastore(s). If you are testing on VMC environment or want to specify the resource pool and/or vm folder to deploy guest VMs, you should fill those fields as well.

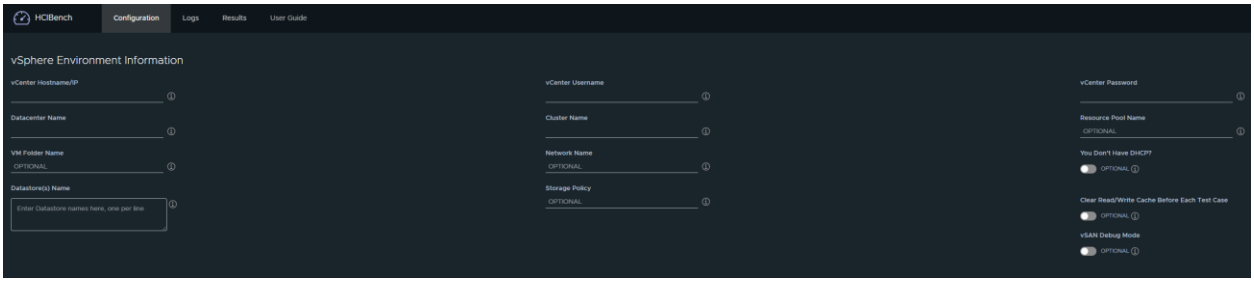


Figure 13. Specify vSphere Environment Information

- **Network Name** defines which network the guest VMs should use. If not specified, the default value is **VM Network**.
- **You Don't Have DHCP?** Instructs HCIBench to set static IPs for guest VMs and use the "VM Network" NIC to communitcate with the guest VMs. If it is checked, you can find a static IP prefix from the list on the right handside. Make sure the prefix you choose is **NOT** being used in the guest VM Network.

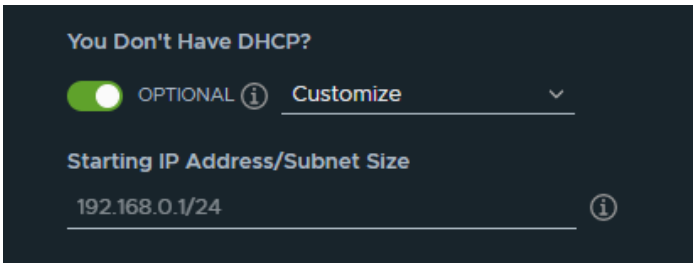


Figure 14. Specify the IP Range in CIDR

If your switch or network policy doesn't allow to specify arbitrary IP addresses in the network you specified, you should give specific IP addresses within an allowed range in the network. To do so, select **Customize** which is at the bottom of the drop-down list, then specify the starting IP address of the range as well as the network size in CIDR format. In the deployment, HCIBench will search for the available IPs within the range specified and assign to the guest VMs, it will error out of less available Ips found than the number of guest VMs needed.

- **Datastore Name** specifies the datastores that are tested against and all the guest VMs are deployed on. You need to enter the name of the datastore. Testing multiple datastores in parallel is also supported. You can enter the datastore names one per line. In this cases, the virtual machines are deployed evenly on the datastores. For example, if you enter two datastores and 100 virtual machines, 50 virtual machines will be deployed on each datastore. If local datastore is specified, HCIBench will find the host which has the access to the datastore to deploy guest VM(s) within the cluster.

Figure 15. Specify Hosts in the Cluster

- **Specify Hosts to Deploy** allows you to specify hosts to deploy guest VMs on, when this parameter checked, you will need to fill up the host(s) in the target cluster you want to have the VMs deployed on; if this is not checked, VMs will be deployed on all the hosts(or selected hosts if local datastore is defined) in the target cluster in round-robin manner. It will error out if the none of hosts specified here have access to any of the local datastores specified.
  In general, it's only needed when you want to deploy guest VMs onto part of the hosts within the cluster.

> ***Best Practice:***
>
> *We recommend **NOT** to specify Hosts to deploy unless you intend to have some hosts not having VMs.*

- **Storage Policy** is the option allow you to specify the name of a Storage Policy which will be applied to the clinet VMs and all the virtual disks. The policy will only be applied to the guest VMs being deployed to vSAN datastore.

Figure 16. Clear Read/Write Cache and vSAN Debug Mode

- **Clear Read/Write Cache Before Each Testing** is the option desgined for vSAN OSA(multi-tier architecture) user to flush the cache tier before each test case, **ESXi Username** and **ESXi Password** must be specified if this box is checked. Also, you will need SSH access from HCIBench to all the ESXi hosts in the vSAN Cluster. If you specified remote vSAN datastore(new HCI Mesh feature introduced in vSAN 7U1) to test against, the read/write cache clear activity will **ONLY** happen on the hosts in the remote cluster, in this case, the **ESXi Username** and **ESXi Password** should be specified for the hosts in the remote vSAN cluster.
- **vSAN Debug Mode** is designed for collecting vSAN vmkstats during the I/O testing and VM-Support Bundle after the I/O testing. You need to specify **ESXi Username** and **ESXi Password** of all the hosts in the cluster and the hosts in the remote cluster if remote vSAN datastore is specified.
- For specifying the **ESXi Username** and **ESXi Password**:
    - If each ESXi host has different **ESXi Username** and **ESXi Password**, the ESXi hosts crendential can be specified individually.
    - If all the ESXi hosts have the same **ESXi Username** and **ESXi Password**, you only need to put the **ESXi Username** and **ESXi Password** in the first line without specifying ESXi host name.
    - If there's ESXi host in the vSAN cluster but not specified here, it will automatically use the **ESXi Username** and **ESXi Password** specified in the first line.

Best Practice:

1. *We recommend having all ESXis username and password set to identical, if remote vSAN datastore is specified, we recommend having the ESXis **username** and **password** set to the same as the local cluster.*
2. *If **vSAN Debug Mode** is turned on, the vmkstats collection will be started after 30 minutes the I/O testing started, so we recommend setting the testing time of each test case for at least 1 hour.*
3. *Starting from version 2.8.3, when collecting vmkstats, HCIBench will create a NFS target on itself, and have all the ESXi hosts mounting to the target and collecting vmkstats right into the target. So please make sure NFS protocol is not block between HCIBench and ESXi hosts.*

- **Reuse VMs If Possible** allows user to reuse the guest VMs in the cluster if they are existing and compatible with the VM specification. If not compatible, existing guest VMs will be deleted and

new VMs will be deployed. **Compatible** means the existing VMs can be found and access from HCIBench; specified **VM Prefix** is same with existing VMs; **Number of VMs**, **Number of Disks** are not greater than existing VMs and **Size of Data Disk** is same with existing VMs, Vdbench or Fio binaries installed properly.

- **EASY RUN** was specifically designed for vSAN user, however, starting from version 2.8.4, EASY RUN is expanding to support non-vSAN datastore. By checking this, HCIBench is able to handle all the test configurations below by identifying the vSAN configuration(if testing against vSAN). Then it helps to decide how many guest VMs should be deployed, the number and size of virtual disks of each VM, the way of preparing virtual disks before testing etc. the **Guest VM Configuration** and **Testing Configuration** sections below will be hidden if this option is checked. Once **EASY RUN** is checked, only **one** datastore can be specified, and you can choose one of the four following profile to test:
    - **Max IOPS or Throughput**
      HCIBench will configure the reasonable I/O concurrency to try to achieve the best possible IOPS or Throughput
    - **Low Latency Threshold**
      HCIBench will try to limit the 95th percentile latency of small random I/O **under 1ms**, and large sequential I/O **under 3ms**. FIO is the only tool can be used for this profile.
    - **Medium Latency Threshold**
      HCIBench will try to limit the 95th percentile latency of small random I/O **under 3ms**, and large sequential I/O **under 5ms**. FIO is the only tool can be used for this profile.
    - **High Latency Threshold**
      HCIBench will try to limit the 95th percentile latency of small random I/O **under 5ms**, and large sequential I/O **under 10ms**. FIO is the only tool can be used for this profile.

  Then, you can select the following one to six workload profiles to run:

    - General purpose workloads:
        - 4K, 70% Read, 100% Random test to simulate the most common workloads.
        - 8K, 50% Read, 100% Random test to simulate the OLTP workloads.
    - 4 Corner workloads:
        - Random Read with Small Blocksize
          4KB, 100% random, 100% read
        - Random Write with Small Blocksize
          4KB, 100% random, 0% read
        - Sequential Read with Large Blocksize
          256KB for vSAN OSA and 512KB for vSAN ESA or non vSAN, 100% sequential, 100% read
        - Sequential Write with Large Blocksize
          256KB for vSAN OSA and 512KB for vSAN ESA or non vSAN, 100% sequential, 0% read

  All those workloads above are configured with 50% compressible and 50% dedupable data for write operations.

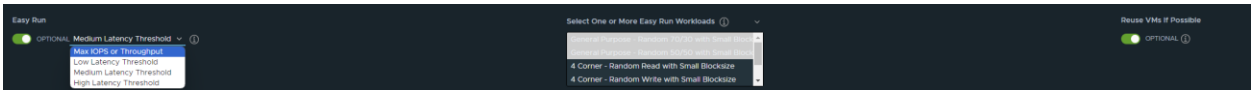


Figure 17. Easy Run and Reuse VMs

Benchmarking Tool

HCIBench can use Fio or Vdbench as the performance workload generator, if Vdbench is selected, you need to download it here and upload the Vdbench zip to HCIBench. To do so, click **Download Vdbench**. After the download is completed, you should upload the zip file. And the server will automatically put the Vdbench zip to /opt/output/vdbench-source. This step is a once-for-all action. The following screen disappears from the page after you upload the Vdbench file successfully.

Figure 18. Benchmarking Tool Selection

Guest VM Configuration

In this section, the only required parameter is **Number of VMs** that specifies the total number of guest VMs to be deployed for testing. If you enter multiple datastores, these VMs are deployed evenly across the datastores. The rest parameters are optional:

Figure 19. Specify Guest VM Information

- **VM Name Prefix** specified the prefix of the VM Name. The default value is depending on the benchmarking tool selection, if Fio is selected, the value here will be hci-fio; when Vdbench is selected, the value will be hci-vdb. Also, you can change the prefix as you want.
- The **Number of CPU** specifies the number of vCPU per guest VM, default value is four.
- The **Size of RAM in GB** specifies the memory size per guest VM, default value is eight.
- The **Number of Data Disk** parameter specifies how many virtual disks to be tested are added to each guest VM. Default number is eight.
- The **Size of Data Disk** parameter specifies the size (GB) of each VMDK to be tested. The total number of simulated workload instances is **Number of VM** * (times) **Number of Data Disk**. Default number is ten.
- The **Multi writer Disks** parameter will allow the worker VMs to be configured with multi-writer option. If this is enabled, the multi-writer data disks will be provisioned in a round-robin manner across all the worker VMs, the total amount of the multi-writer data disks is the **Number of Data Disk**. The figure below illustrates the multi-write Data Disks distribution with 4 **Number of VMs** and 6 **Number of Data Disk** specified.

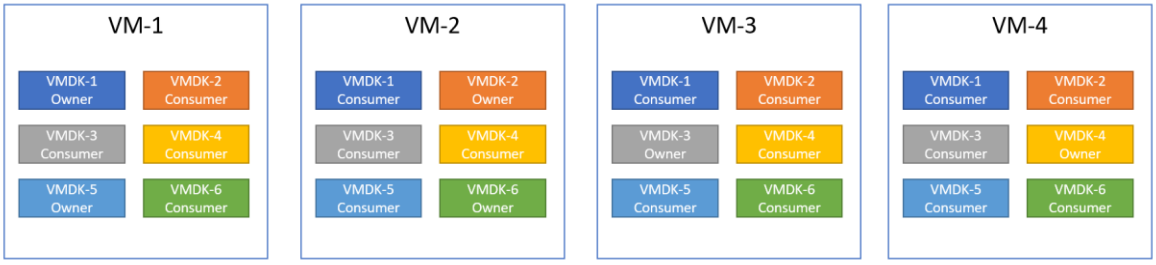


Figure 20 Multi-Writer Data Disks Distribution

Testing Configuration

- **Test Name** parameter is the name of the test, by specifying this parameter, for example "DemoTest", HCIBench will create a local directory with the same name in "/opt/output/results/" on the Controller VM for storing collected results from all guest VMs and statistics produced by vSAN Observer. If not specified, a name "resultsTIMESTAMP" will be generated and the same name directory will be created under "/opt/output/results".All the test cases results could be browsed at http://HCIBench_IP/results, or click the **Results** tab on the navigation bar.
- For the **Workload Parameter File**, If a parameter file is uploaded or generated to the controller before, it already exists in HCIBench. In this case, you can select the existing Vdbench or Fio parameter file from the drop-down list depending on which workload you selected. You can also refresh the drop-down list by clicking the **REFRESH** button. After you finish generating a parameter file or uploading a parameter file, click the **REFRESH** button and it makes the file displayed in the drop-down list without refreshing the entire page to avoid user-input loss. Delete the parameter file by clicking the **DELETE** button.You have two options to add parameter file into the drop-down list:

Generate it by yourself:

you can create parameter files by clicking **ADD**, and which will redirect you to the workload generation page, the title of this page is depending on the tool selection you made earlier, if you had Fio selectted, the title is Fio Parameter Generation. No matter which tool you selected, the input fields are the same. All the fields without "OPTIONAL" are required. After clicking submit, click **REFRESH** to update the drop-down list.

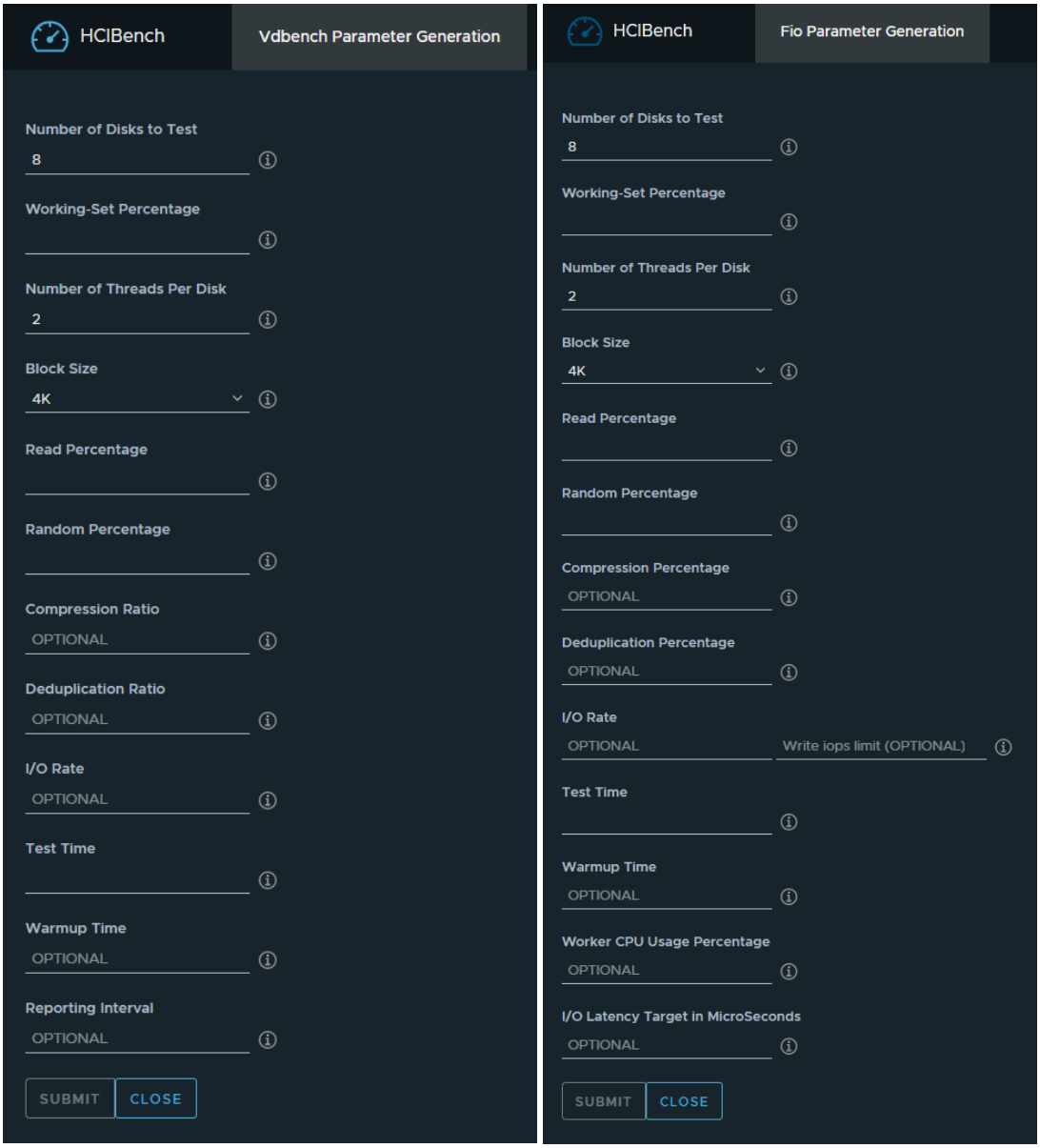


Figure 21. Specify Vdbench or Fio Workload Parameters

Upload it by yourself:

If the desired parameter file does not exist, you can create a self-defined parameter file and upload it to the controller by clicking the **Choose File** button in the **Upload a Parametner File** section. After uploading, click **REFRESH** button, the file you uploaded will be in the drop-down list. For Vdbench or Fio parameter file format, refer to the Vdbench User Guide or Fio User Guide.

Best Practice:

*We recommend having the number of VMs, number and size of Data Disks as well as the Number of thread per Disk reasonably configured. The total threads configured = Number of VMs * Number of Data Disks per VM * Number of Threads per Disk.*

If your goal is to achieve best possible IOPS or Throughput, you can start with 4 VMs per ESXi host, 8 Data Disks per VM and 4 Threads per Disk, run the testing, increase any of those variable for another run if the first run doesn't satisfy you.

if your goal is to achieve best possible latency, you can start with 2 VMs per ESXi host, 4 Data Disks per VM and 1 Threads per Disk, run the testing, reduce any of those variable for another run if the first run doesn't satisfy you.

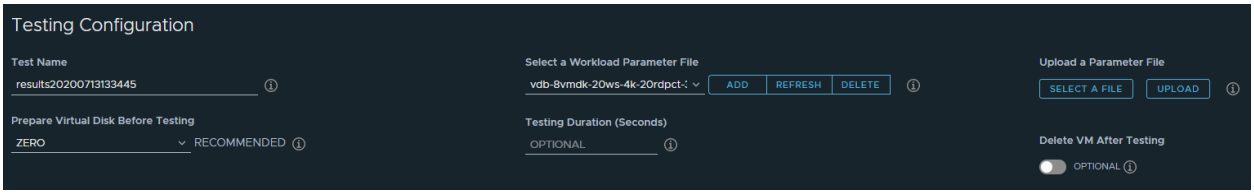


Figure 22. Specify Test Configuration

Note: The value of **Number of Data Disk** in the guest VM Specification section must match the value of **Number of Disks to Test** defined in the parameter files. For example, if you specify to create 10 data disks per guest VM, 10 raw disks are created. Therefore, in the parameter files, the same number or less of disks are expected. Since we are using Photon OS, beware the first data disk starts from /dev/sda, the last disk is the OS disk.

Users can choose whether to intialize the data VMDKs of guest VMs. There are two options of storage initialization, **ZERO** and **RANDOM**. **RANDOM** is particularly for storage that has de-duplication enabled, if the storage will be tested against doesn't have de-duplication enabled, use **ZERO** instead to initialize storage to avoid first-write penalty.

The **Testing Duration** parameter is for overriding the elapsed value in parameter files. This parameter defines the test duration for each run. If not specified, each test run uses its own elapsed value.

When the **Clean up VMs** parameter is checked, all the guest VMs are removed after all the testing is completed; otherwise, all the VMs are perserved.

Best Practice:

*1. We strongly recommend selecting to **Prepare Virtual Disk Before Testing** to allow blocks being allocated to avoid first write penalty as well as the read served by memory only, this operation is one-off and will be skipped if guest VMs are reused.*

2. We recommend set "Delete VM After Testing" to false, you can re-use the deployed VMs for more testing and to skip the Virtual Disk preparation.

Save Configuration

Press the **SAVE CONFIG** button to save the parameter configuration settings. If the configuration setting is not saved and the page is refreshed, the system will read the previous-saved parameter configuration. Until you successfully saved the config, the **VALIDATE CONFIG** and **START TEST** buttons are disabled to enforce you save your before validating or starting testing.

Configuration Validation

After completing the tool configuration, you can validate all settings by clicking the **VALIDATE CONFIG** button. This step checks if all the required information is correctly provided. Additionally, it validates basic environment sanity including whether vSAN is enabled in the cluster, whether the hosts specified belong to the cluster and can access the vSAN datastore. Furthermore, this function estimates the compute and storage usage by all guest VMs and alert if it exceeds 80 percent storage capacity usage or compute resoruce overprovisioning is predicted.

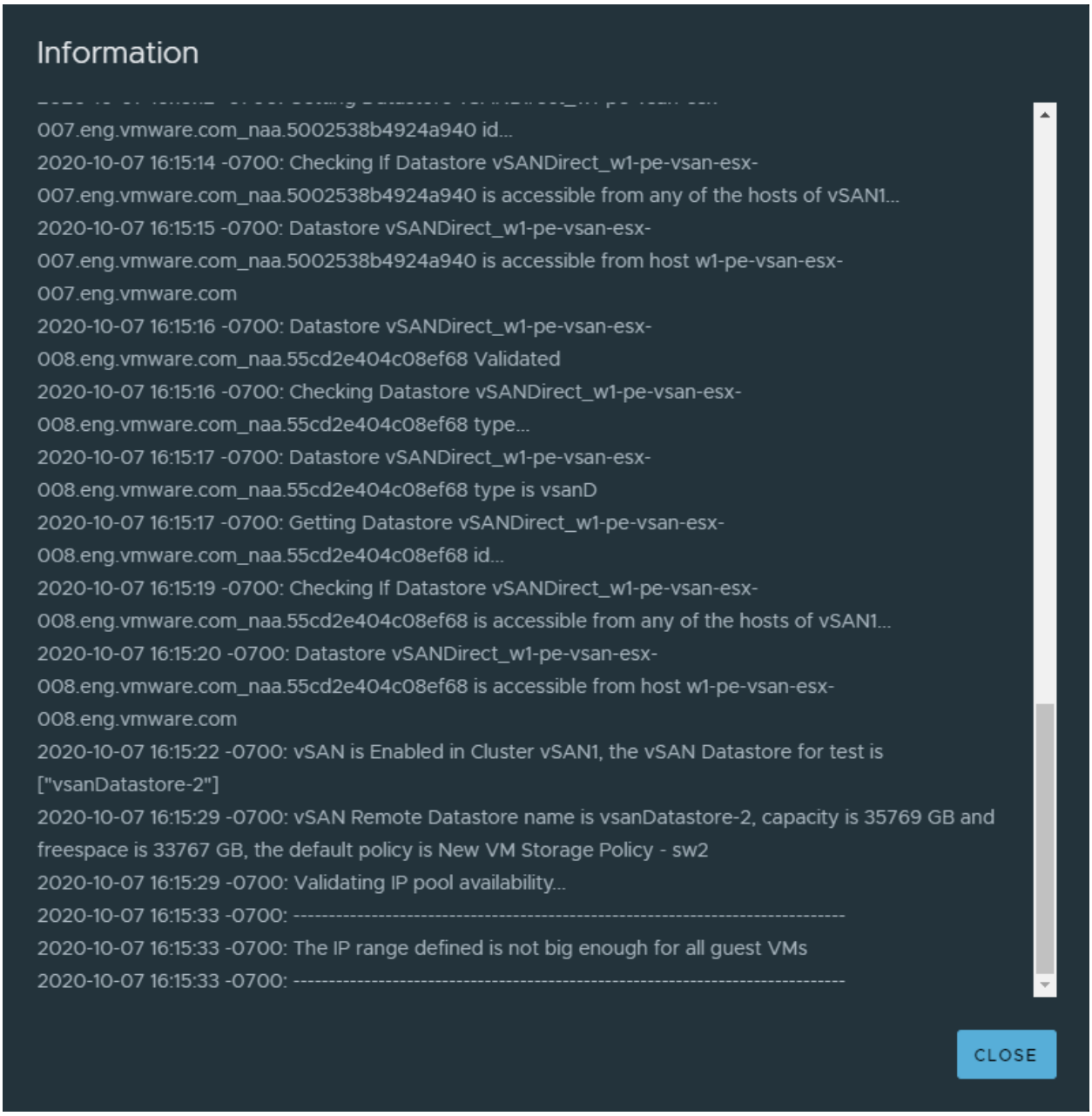


Figure 23. Validation Failure

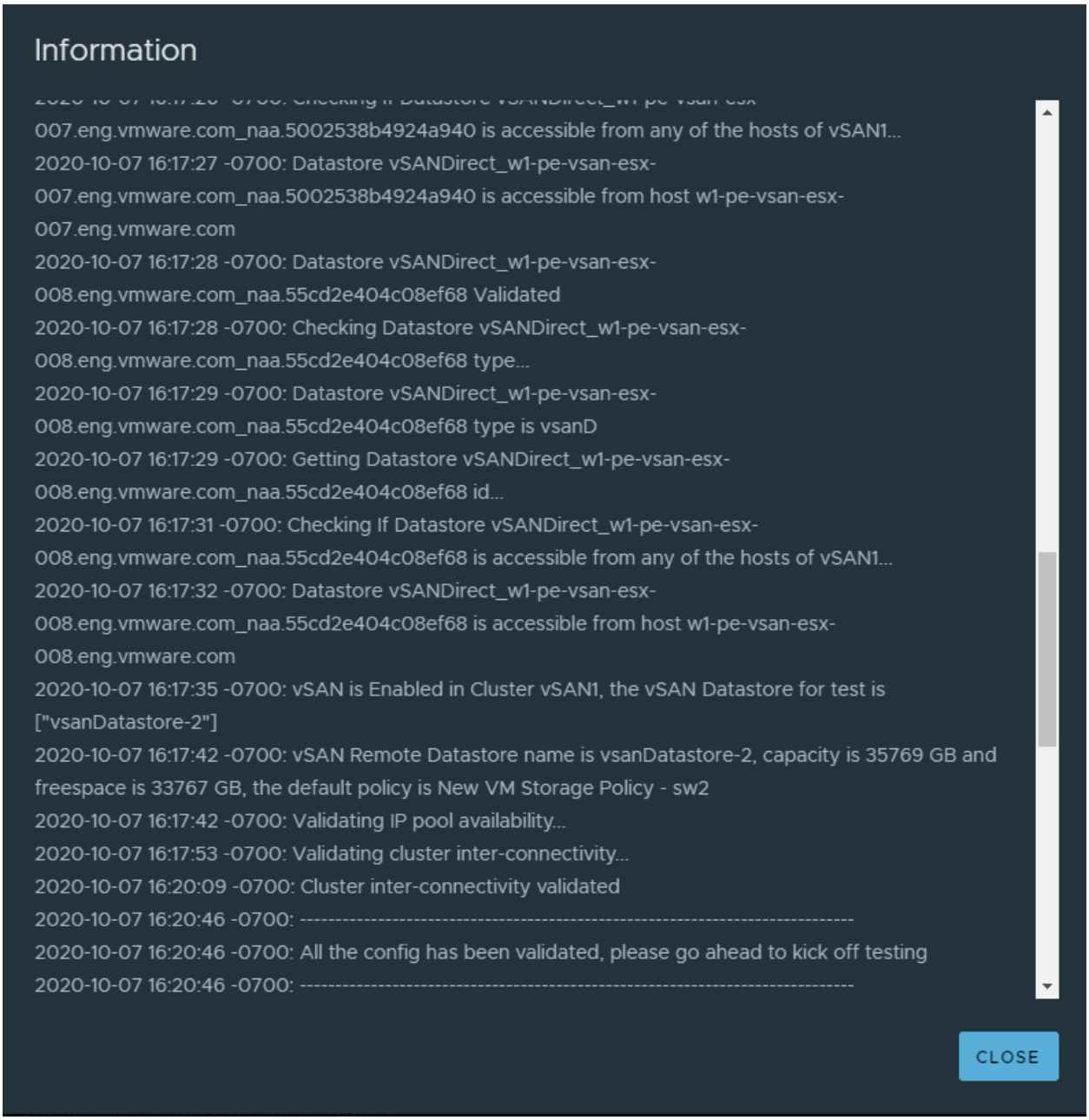


Figure 24. Configuration Validated

After the validation is successfully completed, a message is displayed to inform you that you can continue to the testing.

Tool Usage

How to Run Tests

You can click the **START TEST** button to start the program. The testing is a time-consuming operation with the test progress toolbar displayed on the web page.

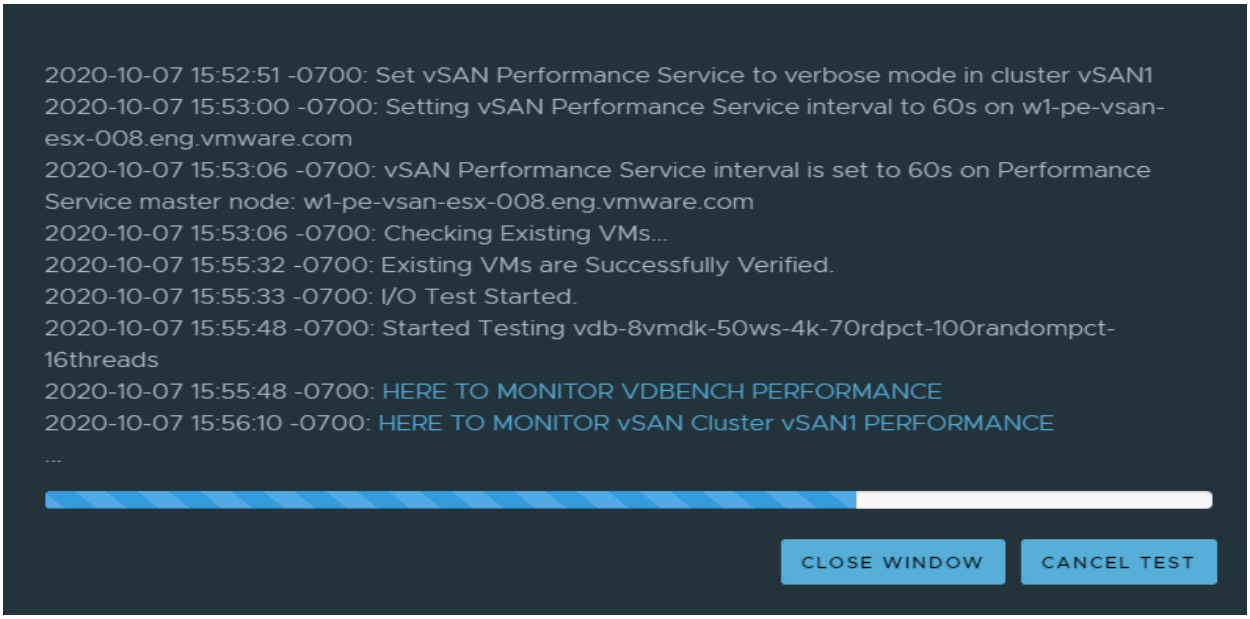


Figure 25. Test in Progress

During the testing, you can monitor the live performance from guest VMs showed up in Grafana by clicking **HERE TO MONITOR**, which will land you on Grafana page: http://HCIBench_IP:3000 to monitor the live performance.

Figure 26. Guest VM Performance Monitoring

If you are testing against vSAN OSA datastore and have vSAN performance service enabled, you can click **HERE TO MONITOR vSAN Cluster CLUSER_NAEM PERFORMANCE** to monitor the vSAN performance from Grafana too.

Figure 27. vSAN Performance Monitoring

Also, you can kill the test process by clicking the **CANCEL TEST** tab**.**

How to Consume Test Results

After the Vdbench or Fio testing finishes, the test results are collected from all the guest VMs. And you can view the results at http://HCIBench_IP/results in a web browser or click **Results** tab to review it.

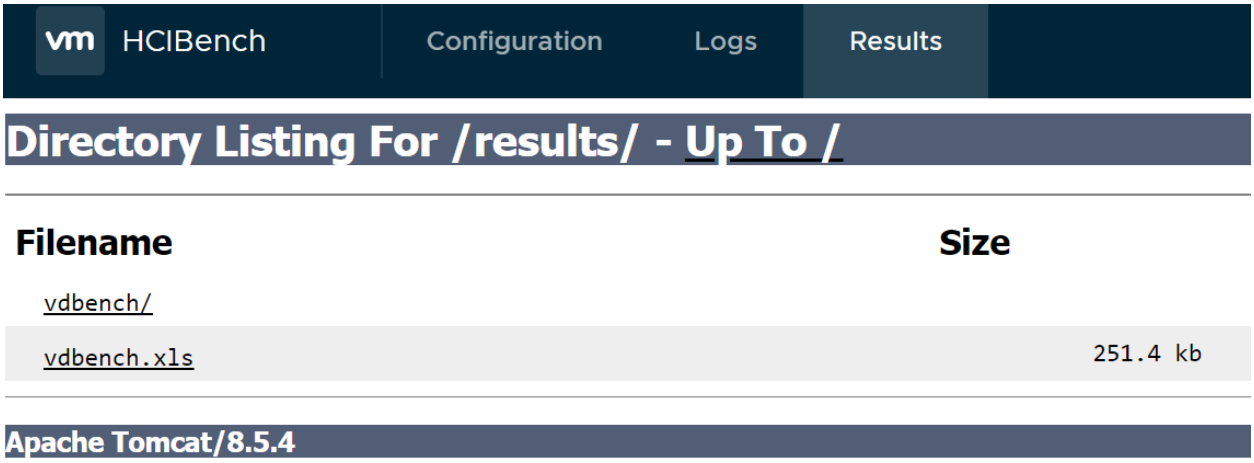


Figure 28. Test Results

The xls file is the spreadsheet which summarizes all the test cases inside a TestName folder, you can download to compare the performance of different test cases in one spreadsheet. Also, the spreadsheet

provides the detail information showing every single interval of each test case thus you can easily create charts to present the historical graph.

Each of the subdirectories in "/opt/output/results/TestName" directory uses the name of the user-defined parameter file, and contains all original results produced by each Vdbench or Fio instance and vSAN Observer data.

The pdf report of each test run shows the comprehensive information of the test case including testing time, HCIBench version, performance results, Grafana dashboards as well as detailed configuration including HCIBench, workload as well as vSAN if applicable. You can simply grab it and send to your team without having everything explained.

Index of /results/demo

Name	Last modified	Size	Description
Parent Directory		-	
vdb-8vmdk-50ws-4k-70rdpct-100randompct-16threads-1602113277-report.pdf	2020-10-07 16:38	403K	
vdb-8vmdk-50ws-4k-70rdpct-100randompct-16threads-1602113277/	2020-10-07 16:38	-	
vdb-8vmdk-50ws-4k-70rdpct-100randompct-16threads-1602113277-res.txt	2020-10-07 16:38	2.1K	

Figure 29. PDF report and results text file

The aggregated result of one test run is summarized in the text file with the name <*DIR_NAME*>-res.txt, containing the datastore's name and four statistics: number of VMs used for testing, IOPS, throughput, latency details and host resource consumption.

```
Datastore     = VMFS-Datastore
============================
Version:        vdbench50406
Run Def:        RD=run1; I/O rate: Uncontrolled MAX; elapsed=600; For loops: None
VMs             = 2
IOPS            = 1175.34 IO/S
THROUGHPUT      = 36.73 MB/s
LATENCY         = 54.04 ms
R_LATENCY       = 0.53 ms
W_LATENCY       = 178.19 ms
95%tile_LAT     = 294.99 ms
============================
Datastore     = vsanDatastore (OSA)
============================
Version:        vdbench50406
Run Def:        RD=run1; I/O rate: Uncontrolled MAX; elapsed=600; For loops: None
VMs             = 2
IOPS            = 74060.66 IO/S
THROUGHPUT      = 2314.4 MB/s
LATENCY         = 0.86 ms
R_LATENCY       = 0.66 ms
W_LATENCY       = 1.33 ms
95%tile_LAT     = 2.12 ms
============================
Resource Usage:
+-------------+-----------+-----------------+-----------+
|                  Resource Utilization                  |
+-------------+-----------+-----------------+-----------+
| OSA         | cpu.usage% | cpu.utilization% | mem.usage% |
+-------------+-----------+-----------------+-----------+
| 10.156.28.1  | 10.42     | 6.48            | 11.04     |
| 10.156.28.14 | 7.07      | 3.92            | 10.49     |
| 10.156.28.7  | 2.67      | 1.63            | 8.35      |
| 10.156.28.8  | 8.46      | 5.48            | 8.45      |
+-------------+-----------+-----------------+-----------+
| Average     | 7.16      | 4.38            | 9.58      |
+-------------+-----------+-----------------+-----------+


cpu.usage - Provides statistics for logical CPUs. This is based on CPU Hyperthreading.
cpu.utilization - Provides statistics for physical CPUs.


vSAN Performance Stats Graph: http://10.159.24.4:3000/d/vsan_client/dom-client?orgId=1&from=1672795260000&to=1672795860000&var-datasource=vdb_8vmdk_100ws_32k_70rdpct_0randompct_4threads_1672795228
```

Figure 30. Aggregated Performance Data, showing all datastores' performance

If the testing is against vSAN datastore and **vSAN Debug Mode** is enabled, there will be a hyperlink in the bottom of the result file which will land you on the Grafana dashboard illustrating the vSAN detail stats parsed from the vm support bundle collected after the test.

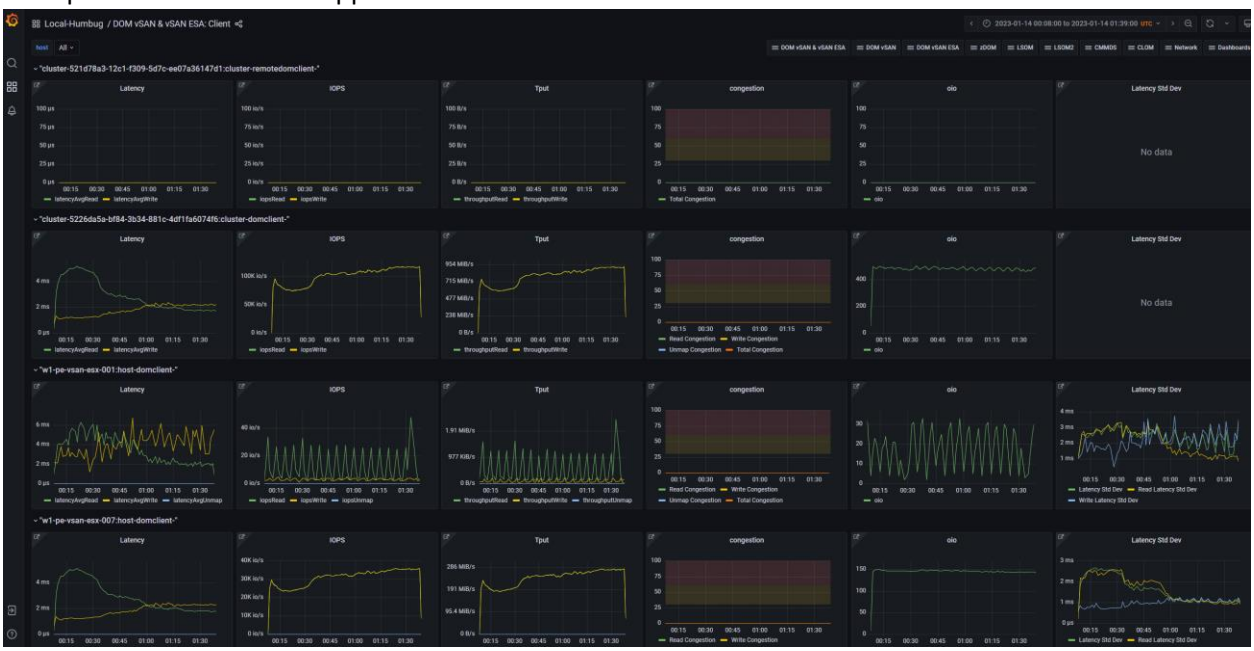


Figure 31 vSAN Performance Stats in Grafana

You can find all the original result files produced by Vdbench or Fio instances inside the subdirectory corresponding to a test run. In addition to the text files, there is another subdirectory named iotest-hcibench/fio-<*VM#*>vm inside, which is the statistics directory generated by vSAN Observer. Also, you should be able to find the following files:

HCIBench-VERSION-logs.tar.gz: HCIBench pre-validation and testing logs.

CLUSTER_NAME-health.info: vSAN health information.

resource_usage.info/resource_util.info: Average and Aggregated Resource Utilization information.

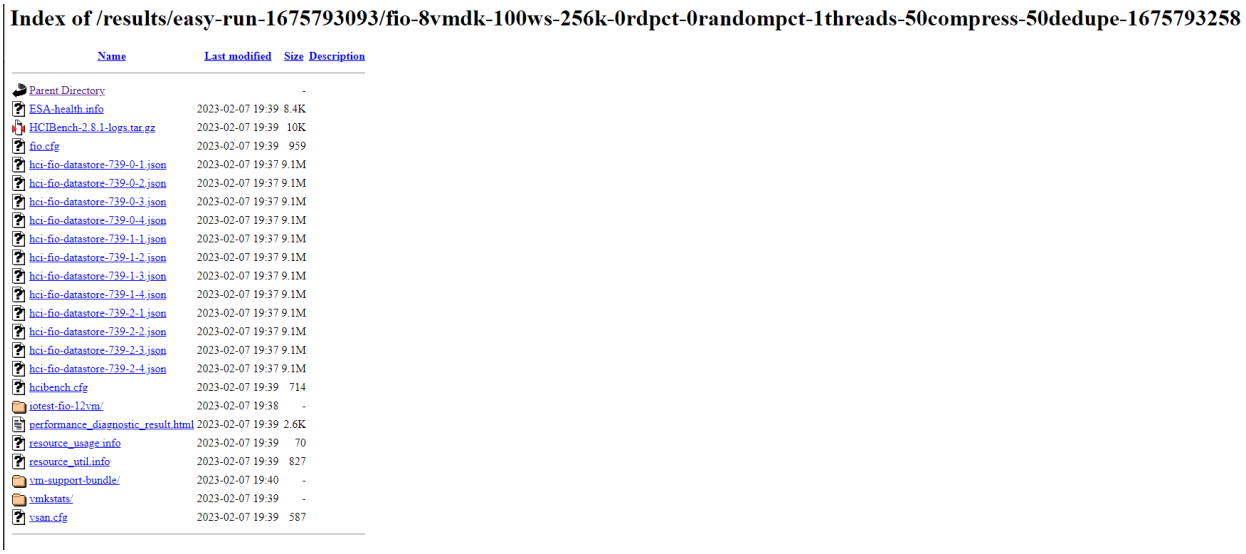
hcibench.cfg: HCIBench configuration parameters.

vsan.cfg: vSAN configuration.

vdbench.cfg/fio.cfg: Vdbench/Fio parameter profile.

vmkstats: vmkstats collected from all ESXi hosts if vSAN debug is enabled.

vm-support-bundle: vm-support bundle collected from all ESXi hosts if vSAN debug is enabled.

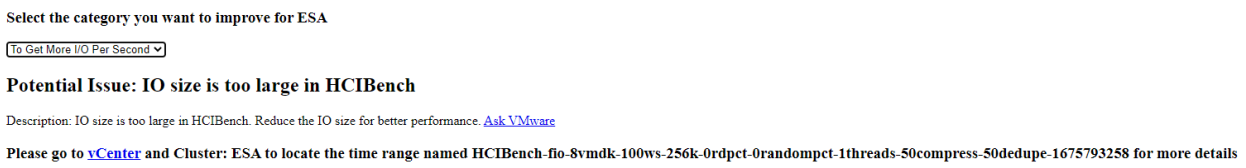


Index of /results/easy-run-1675793093/fio-8vmdk-100ws-256k-0rdpct-0randompct-1threads-50compress-50dedupe-1675793258

Name	Last modified	Size	Description
Parent Directory		-	
ESA-health.info	2023-02-07 19:39	8.4K	
HCIBench-2.8.1-logs.tar.gz	2023-02-07 19:39	10K	
fio.cfg	2023-02-07 19:39	959	
hci-fio-datastore-739-0-1.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-0-2.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-0-3.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-0-4.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-1-1.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-1-2.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-1-3.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-1-4.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-2-1.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-2-2.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-2-3.json	2023-02-07 19:37	9.1M	
hci-fio-datastore-739-2-4.json	2023-02-07 19:37	9.1M	
hcibench.cfg	2023-02-07 19:39	714	
iotest-fio-12vm/	2023-02-07 19:38	-	
performance_diagnostic_result.html	2023-02-07 19:39	2.6K	
resource_usage.info	2023-02-07 19:39	70	
resource_util.info	2023-02-07 19:39	827	
vm-support-bundle/	2023-02-07 19:40	-	
vmkstats/	2023-02-07 19:39	-	
vsan.cfg	2023-02-07 19:39	587	

Figure 32. Files in Test Subfolder

performance_diag_result.html: If testing against vSAN 6.6U1 or above and using HCIBench 1.6.6 or above, turning on **CEIP**(Customer Experience Improvement Program) and **vSAN Performance Service**, each HCIBench run will send the testing results as well as the testing configuration to VMware Cloud to help user to analyze the potential issue that blocks from achieving a certain goal(Max IOPS, Max Throughput or Minimum Latency). User can land to the specific vCenter page and the KB article of any potential issued detected from the hyperlink provided in this file. If HCI Mesh(remote vSAN datastore) is tested, we may see the perofrmance diagnostic information about two or more vSAN clusters which are involved.



Select the category you want to improve for ESA

To Get More I/O Per Second

Potential Issue: IO size is too large in HCIBench

Description: IO size is too large in HCIBench. Reduce the IO size for better performance. Ask VMware

Please go to vCenter and Cluster: ESA to locate the time range named HCIBench-fio-8vmdk-100ws-256k-0rdpct-0randompct-1threads-50compress-50dedupe-1675793258 for more details

Figure 33. vSAN Performance Diagnostic

Open the stats.html file inside the statistics directory, you can find the vSAN performance statistics for debugging or evaluating purposes.

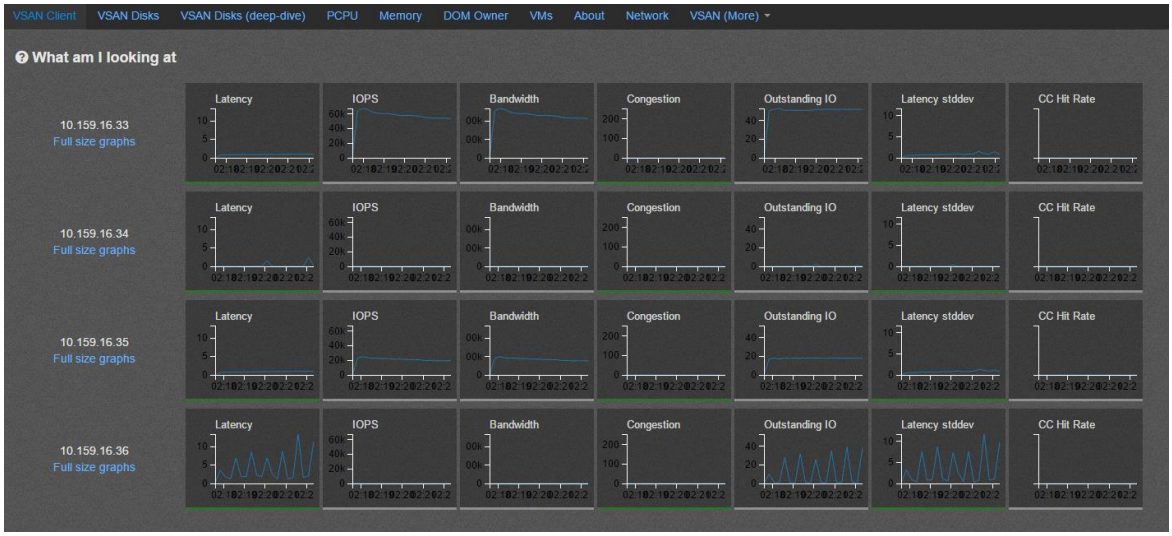


Figure 34. vSAN Observer Statistics

How to Download Test Results to Local Disk

Download the test results by clicking the **SAVE RESULT** button. The latest test result details are zipped to a file, and you can download the file to your local client.

Appendix A – Networking

Networking configuration is often the most challenging aspect when deploying and configuring HCIBench. While the appliance is easily deployed from a single OVA file, proper operation depends on the connectivity provided by the underlying network.

HCIBench Connectivity

For proper operation HCIBench communicate with four systems.

- The user desktop is the system used to connect to the HCIBench web interface to configure and monitor tests. This system should communicate with HCIBench on the vmnic0 (eth0) interface.
- The VMware vCenter controlling the environment being tested. Communication with the vCenter is required to create the worker VM an gather vSAN performance data. HCIBench should communicate with the vCenter using its vmnic0 (eth0) interface.
- The ESXi hosts part of the cluster being tested by HCIBench. Connectivity with the ESXi hosts is required when deploying directly to the hosts as well as the drop cache and/or vfeature. HCIBench should communicate with the ESXi hosts using its vmnic0 (eth0) interface.
- Workers are the virtual machines created by HCIBench to generate the test load. HCIBench communicates with the workers to configure them, launch tests, and collect metrics. HCIBench should communicate with the workers on its vmnic1 (eth1) interface.

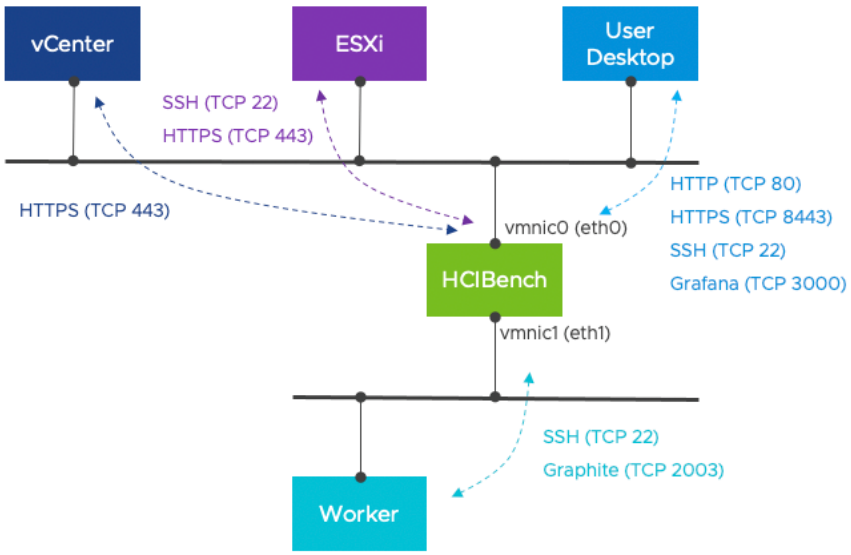


Figure 35. HCIBench Connectivity

HCIBench can use standard vswitches, distributed switches, or NSX-T logical segments. When using VLAN backed networks it is important to ensure the VLAN is properly trunked on all physical switch ports connected to the ESXi hosts. The presence of a portgroup does not imply a properly configured physical switch port. A common mistake is a missing VLAN on one or more trunk ports resulting in connectivity problems. In the following example all hosts have a portgroup PG2 configured, but the physical port for ESXi host 3 does not trunk VLAN 20 (Figure 36).

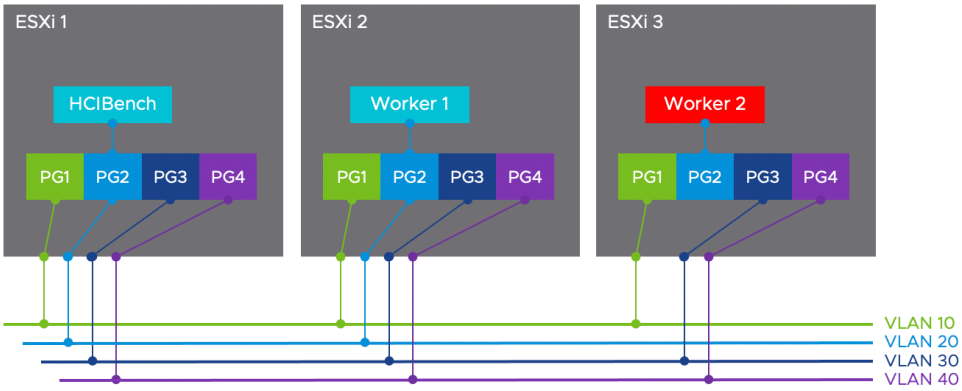


Figure 36. Missing VLAN on Trunk

- HCIBench has no special networking requirements beyond reliable and unrestricted communication on the required service ports (Table 1). HCIBench expects the standard TCP/IP features, such as ARP and DHCP, to operate as defined in their respective RFC[1]. It should be noted that some physical network fabric technologies make subtle tradeoffs in the operation of network services for better scalability or abstraction. In some instances, these subtle differences can result in HCIBench connectivity issues because it was designed with the assumption of a network operating with standard behavior and timers. For more information, see the

[1] https://www.ietf.org/download/rfc-index.txt

Troubleshooting section.

Table 1 HCIBench Required TCP/IP Ports

Name	Protocol	Port	Source	Destination
HTTPS	TCP	443	HCIBench	vCenter
SSH	TCP	22	HCIBench	ESXi
HTTPS	TCP	443	HCIBench	ESXi
SSH	TCP	22	User Desktop	HCIBench
HTTP	TCP	80	User Desktop	HCIBench
HTTPS	TCP	8443	User Desktop	HCIBench
Grafana	TCP	3000	User Desktop	HCIBench
SSH	TCP	22	HCIBench	Worker
Graphite	TCP	2003	Worker	HCIBench

HCIBench only supports IPv4 networking. To operate HCIBench expects every system to be configured and reachable with a valid IPv4 address. HCIBench is completely self-contained and only needs connectivity to the vCenter, ESXi, user desktop, and worker VM. For normal operation HCIBench does not require access to the internet or other VMware resources.

HCIBench communication with vCenter, ESXi hosts, and the user desktop can be either layer 2 or layer 3. Provided HCIBench is able to reach the systems, the systems are able to reach HCIBench, and none of the required ports are restricted, connectivity requirements are met.

For HCIBench communication with the Worker VM the preferred approach is to have both on the same broadcast domain (layer 2). Worker VM will required some form of dynamic IP address assignment. Dynamic IP assignment for the worker VM can be DHCP or static assignment by HCIBench. HCIBench static assignment is enabled by the “You Don't Have DHCP?” configuration option. An important detail of the HCIBench static IP assignment method is that it temporarily reconfigures the HCIBench second interface for the selected subnet. Additionally, the HCIBench static IP assignment does not configure the workers with a valid default gateway.

Validated Network Topologies

This section outlines the common and validated network topologies used when testing with HCIBench. The topologies presented are intended to provide users with reference when planning or deploying HCIBench in their environments.

Single Network without DHCP

Simplest topology with no dependance on network services or routing. Management and worker traffic use different subnets but run on the same network segment.

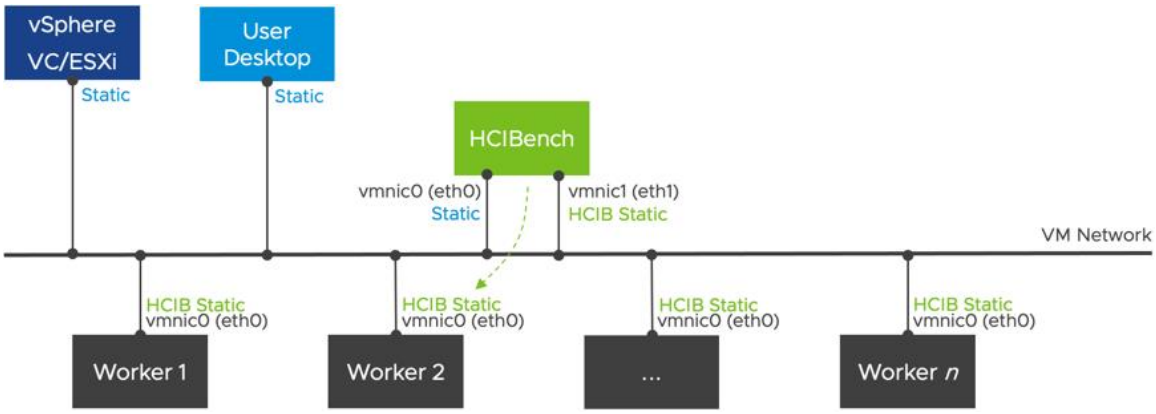


Figure 37. Single Network without DHCP

- Single network segment
- Single network segment carries 2 subnets
    - Subnet A: management
    - Subnet B: VM network
- Selected HCIBench static IP assignment subnet must not conflict or overlap with the management subnet
- Static IP assignment (subnet A)
    - vCenter
    - ESXi
    - User Desktop
    - HCIBench eth0
- HCIBench Static IP assignment (subnet B)
    - HCIBench eth1
    - Workers

Single Network with DHCP

Simple topology with no routing but leveraging an existing DHCP service. Management and worker traffic are on the same network and use the same subnet. All dynamic IP assignment depends on the external DHCP server.

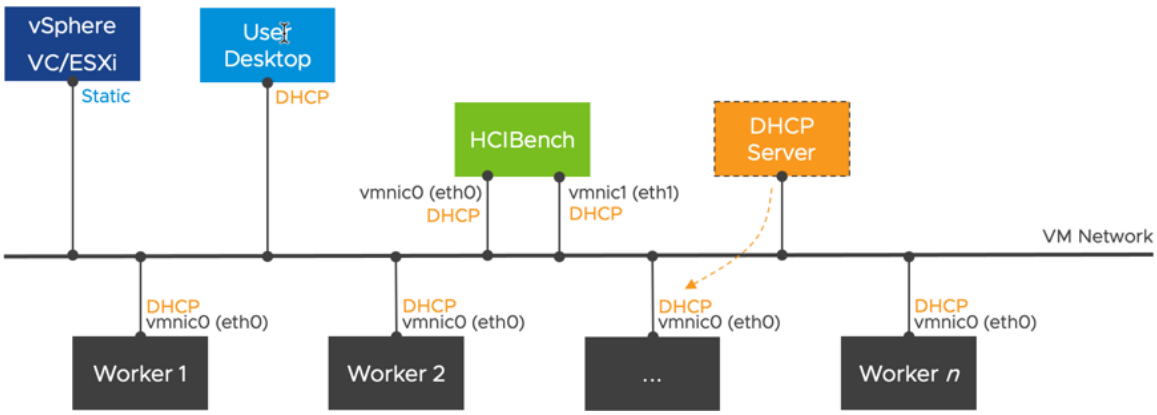


Figure 38. Single Network with DHCP

- Single network segment with DHCP service
- Single network segment carries 1 subnet
  - Subnet A: management and VM network
- Static IP assignment (subnet A)
  - vCenter
  - ESXi
- DHCP IP assignment (subnet A)
  - User Desktop
  - HCIBench eth0
  - HCIBench eth1
  - Workers

Single Routed Network without DHCP

Simple topology with routing to the vSphere assets. Management and worker traffic use different subnets but run on the same network segment.

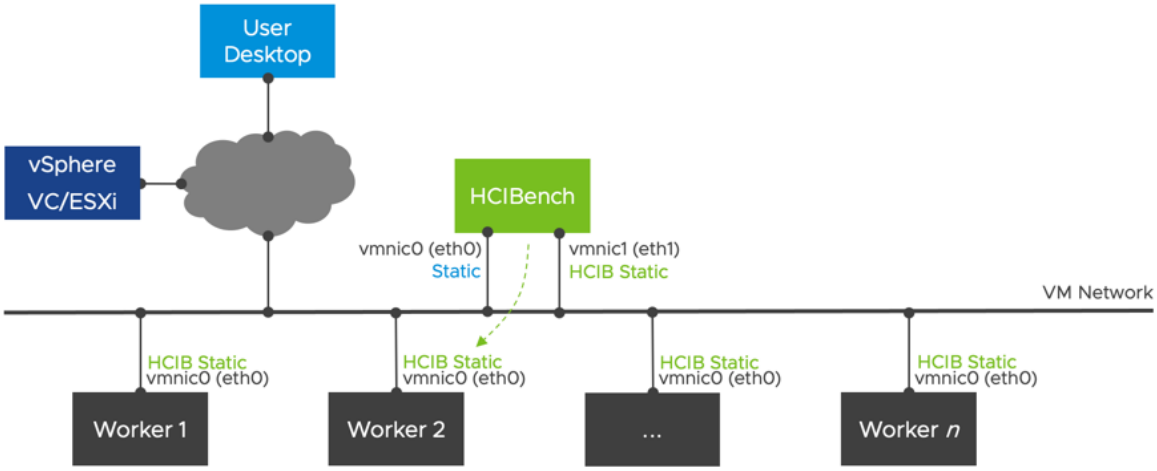


Figure 39. Single Routed Network without DHCP

- Single network segment
- Single network segment carries 2 subnets
    - Subnet A: management
    - Subnet B: VM network
- Selected HCIBench static IP assignment subnet must not conflict or overlap with the management subnet
- All vSphere assets, vCenter and ESXi hosts are on other unspecified network segment that are accessible to the management network
- Static IP assignment with default gateway configured (subnet A)
    - HCIBench eth0
- HCIBench Static IP assignment (subnet B)
    - HCIBench eth1
    - Workers

Single Routed Network with DHCP

Simple topology with routing to the vSphere assets and DHCP service. Management and worker traffic use the same subnet with IP assigned from the DHCP server.

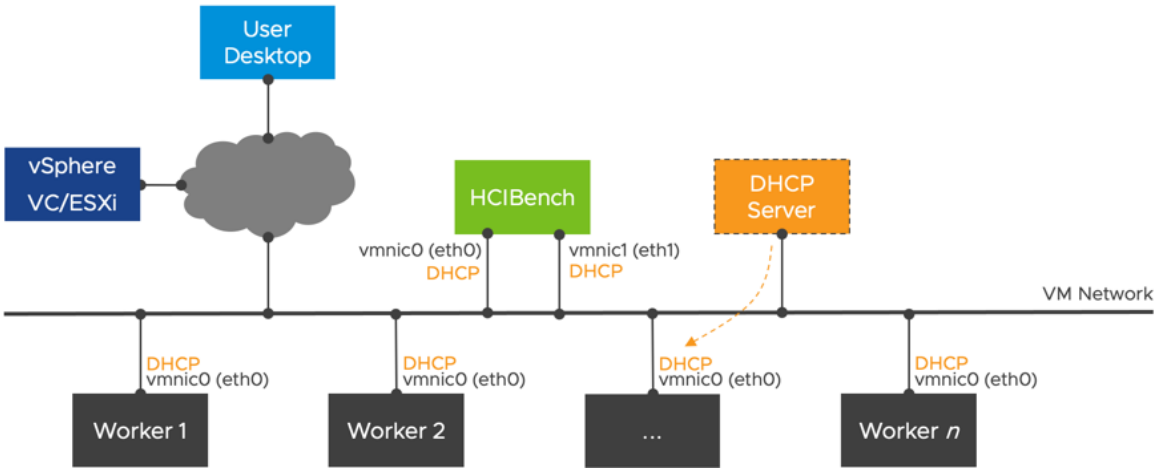


Figure 40. Single Routed Network with DHCP

- Single network segment
- Single network segment carries 1 subnet
  - Subnet A: management and worker
- All vSphere assets, vCenter and ESXi hosts are on other unspecified network segment that are accessible to the management network
- DHCP service must configure a valid default gateway
- DHCP IP assignment (subnet A)
  - User Desktop
  - HCIBench eth0
  - HCIBench eth1
  - Workers

Management Network and VM Network

Topology with a management network and a VM network. HCIBench is connected to both networks.

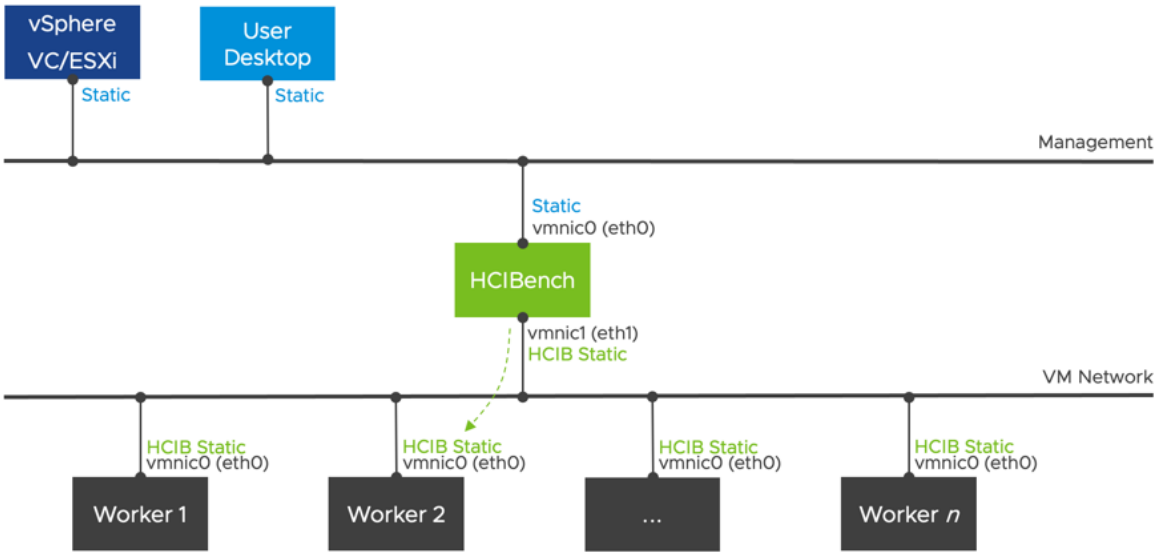


Figure 41. Management Network without DHCP and VM Network without DHCP

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Selected HCIBench static IP assignment used on network segment 2 must not conflict or overlap with the subnet on the management network segment
- Static IP assignment (subnet A)
  - vCenter
  - ESXi
  - User Desktop
  - HCIBench eth0
- HCIBench Static IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Management Network with DHCP and VM Network

Topology with a management network and a VM network. A DHCP service is running on the management network.

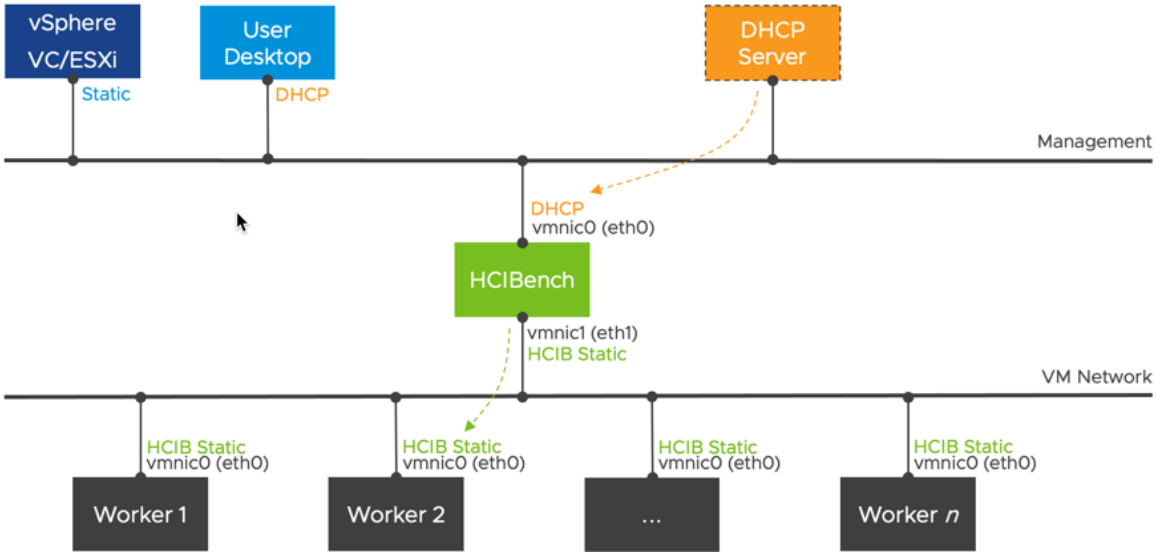


Figure 42. Management Network with DHCP and VM Network without DHCP

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Selected HCIBench static IP assignment used on network segment 2 must not conflict or overlap with the subnet on the management network segment
- Static IP assignment (subnet A)
  - vCenter
  - ESXi
- DHCP IP assignment (subnet A)
  - User Desktop
  - HCIBench eth0
- HCIBench Static IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Management Network and VM Network with DHCP

Topology with a management network and a VM network. A DHCP service is running on the VM network.

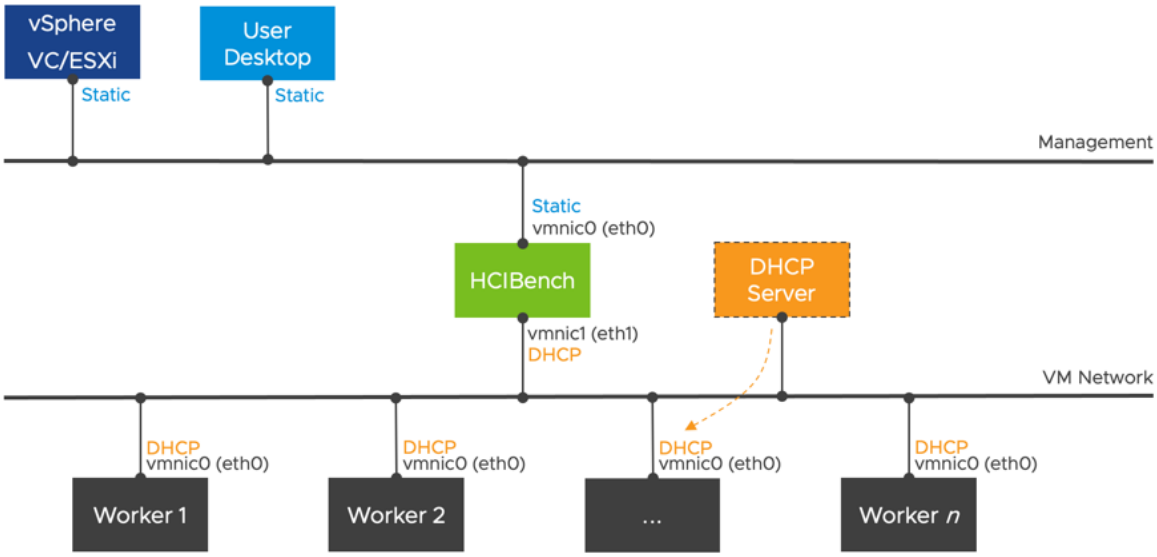


Figure 43. Management Network without DHCP and VM Network with DHCP

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Static IP assignment (subnet A)
  - vCenter
  - ESXi
  - User Desktop
  - HCIBench eth0
- DHCP IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Management Network with DHCP and VM Network with DHCP

Topology with a management network and a VM network. Both network segments have a DHCP service running.

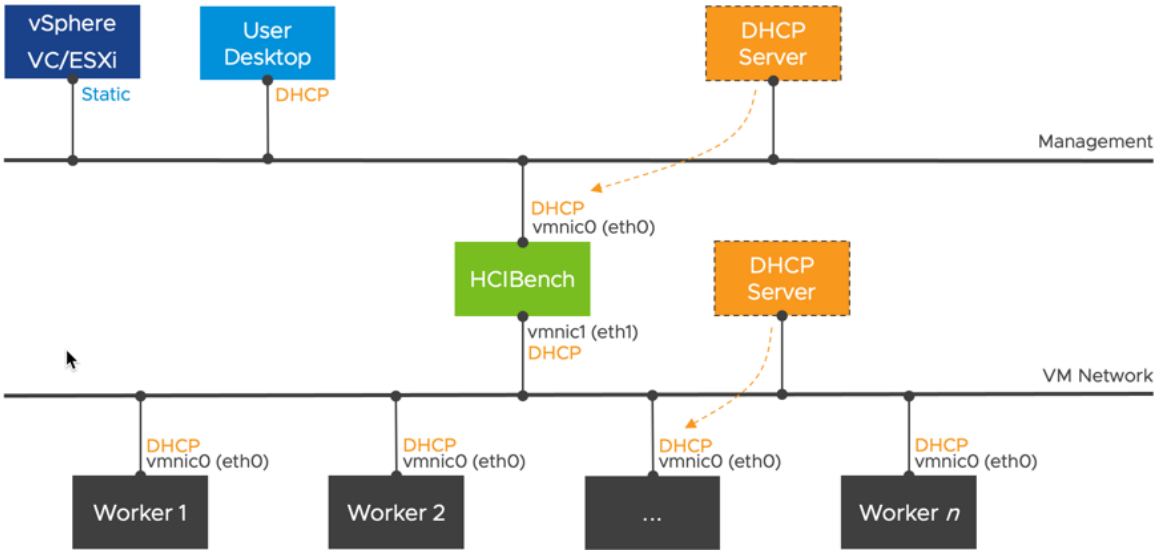


Figure 44. Management Network with DHCP and VM Network with DHCP

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Static IP assignment (subnet A)
  - vCenter
  - ESXi
- DHCP IP assignment (subnet A)
  - User Desktop
  - HCIBench eth0
- DHCP IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Routed Management Network and VM Network

Topology with a management network and a VM network. vSphere assets are located on other networks and depend on properly configured routing on the management network.

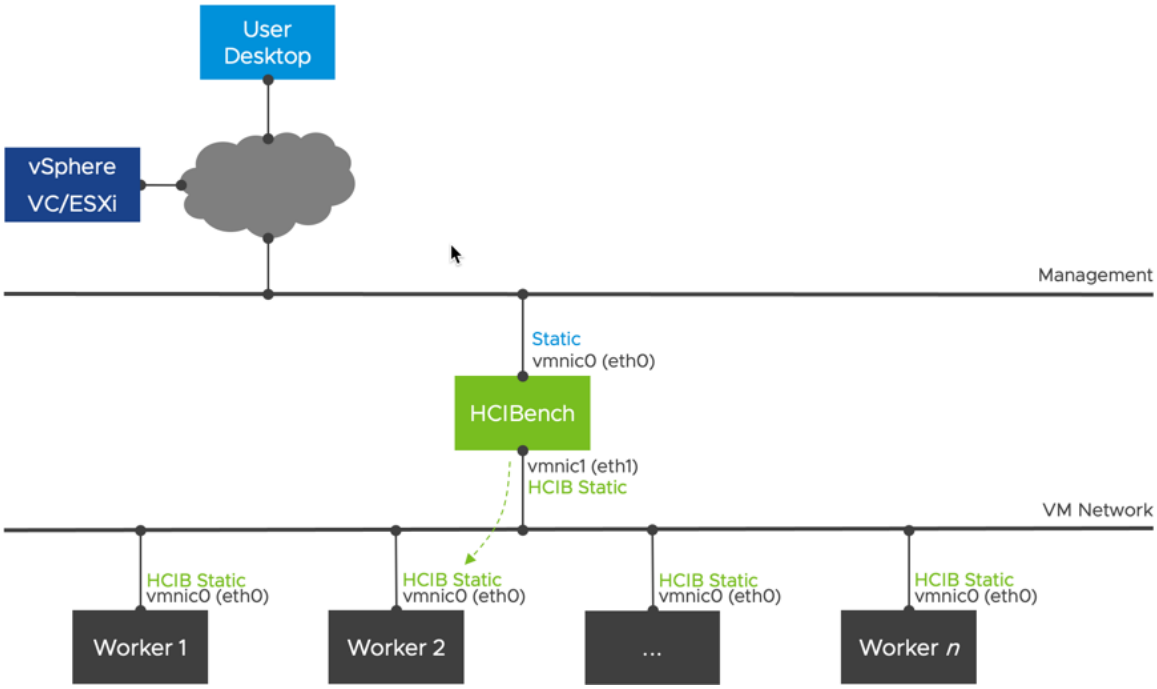


Figure 45. Routed Management Network and VM Network

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Selected HCIBench static IP assignment used on network segment 2 must not conflict or overlap with the subnet on the management network segment
- Static IP assignment with default gateway configured (subnet A)
  - HCIBench eth0
- HCIBench Static IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Routed Management Network and VM Network with DHCP

Topology with a management network and a VM network. vSphere assets are located on other networks and depend on properly configured routing on the management network. A DHCP service is running on the VM network.

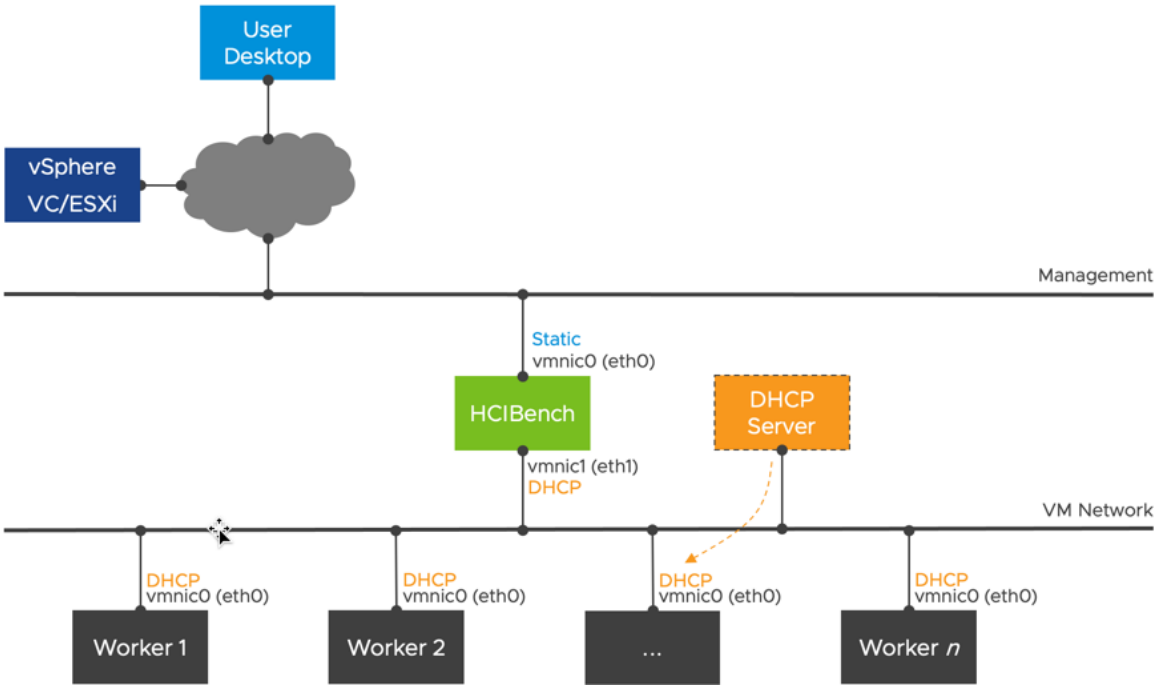


Figure 46. Routed Management Network and VM Network with DHCP

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Static IP assignment with default gateway configured (subnet A)
  - HCIBench eth0
- DHCP IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Routed Management Network with DHCP and VM Network with DHCP

Topology with a management network and a VM network. vSphere assets are located on other networks and depend on properly configured routing on the management network. A DHCP service is running on both the management and VM network.

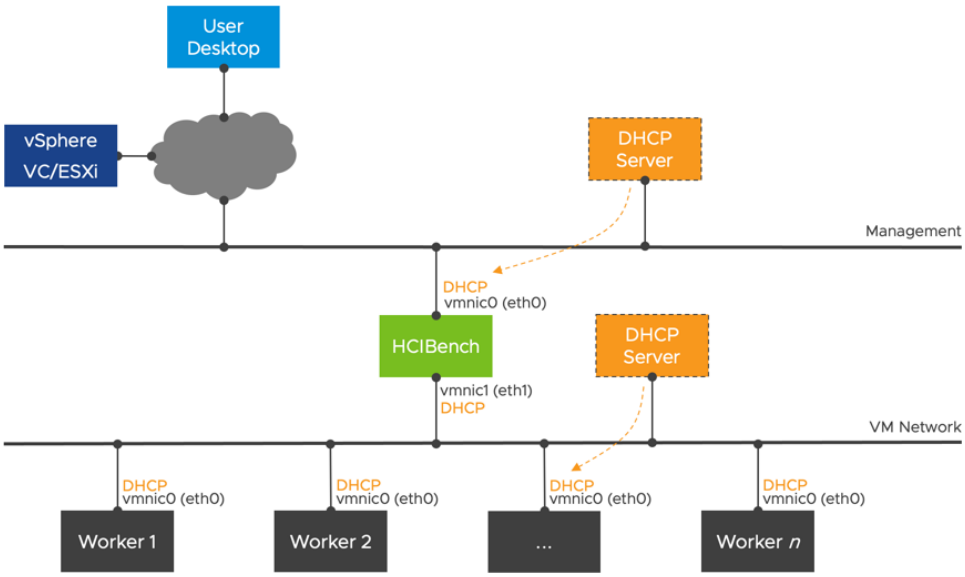


Figure 47. Routed Management Network with DHCP and VM Network with DHCP

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- DHCP IP assignment with default gateway configured (subnet A)
  - HCIBench eth0
- DHCP IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Routed Management Network with DHCP and VM Network

Topology with a management network and a VM network. vSphere assets are located on other networks and depend on properly configured routing on the management network. A DHCP service is running on the management network.

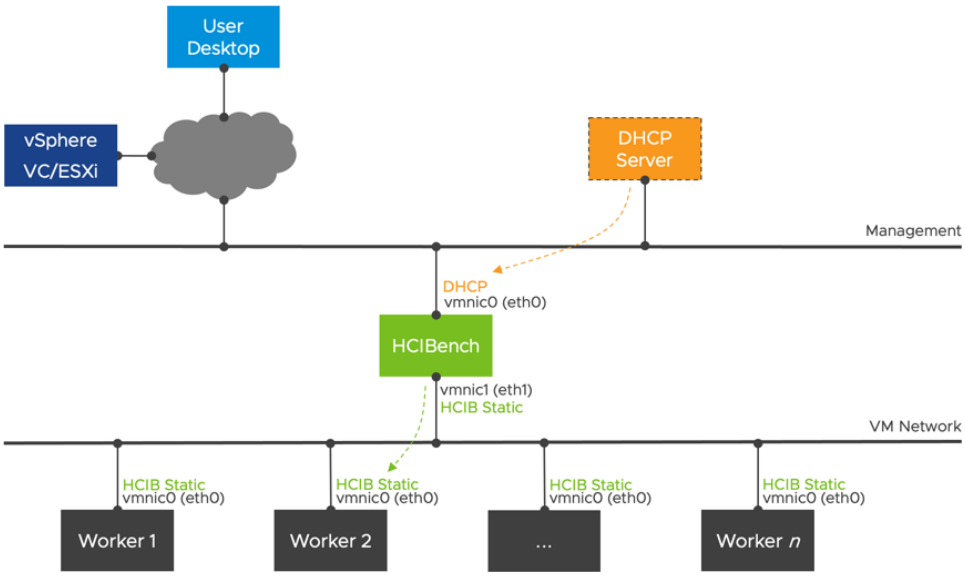


Figure 48. Routed Management Network with DHCP and VM Network

- Two network segments
  - Network Segment 1 (subnet A): management
  - Network Segment 2 (subnet B): VM network
- Selected HCIBench static IP assignment used on network segment 2 must not conflict or overlap with the subnet on the management network segment
- DHCP IP assignment with default gateway configured (subnet A)
  - HCIBench eth0
- HCIBench Static IP assignment (subnet B)
  - HCIBench eth1
  - Workers

Troubleshooting

Basic Checklist

To help pinpoint connectivity problems the following table contains questions that isolate functionality and provide common issues to validate. This troubleshooting assumes a correctly configured vSphere environment where vCenter and all ESXi hosts are deployed and operational. Any underlying vSphere connectivity problems should be addressed before investigating HCIBench connectivity issues.

Step	Question	Check Items

1	Can the user desktop ping HCIBench?	• Ensure correct HCIBench IP (eth0) is being pinged
		• Ensure HCIBench vmnic0 is connected to the correct portgroup
		• Ensure HCIBench (eth0) has a valid IPv4 address for the portgroup connected to vmnic0
		• If the desktop and HCIBench are on different networks
		  ○ Ensure routing is configured between networks
		  ○ Ensure HCIBench has a default gateway configured
		• Ensure there are no ACL or firewall rules blocking ping requests (ICMP)

2	Can the user desktop SSH into HCIBench?	• Ensure there are no ACL or firewall rules blocking SSH connections (TCP port 22)
3	Can the user desktop connect to the HCIBench web interfaces (HTTP/HTTPS)?	• Ensure there are no ACL or firewall rules blocking HTTP connections (TCP port 80) • Ensure there are no ACL or firewall rules blocking HTTPS connections (TCP port 8443)

4	Can HCIBench ping the vCenter?	• Ensure HCIBench (eth0) has a valid IPv4 address for the portgroup connected to vmnic0
		• Ensure HCIBench vmnic0 is connected to the correct portgroup
		• If vCenter and HCIBench are on different networks
		  ○ Ensure routing is configured between networks
		  ○ Ensure HCIBench has a default gateway configured
		• If HCIBench is connecting to vCenter by hostname
		  ○ Ensure HCIBench has DNS configured
		  ○ Ensure HCIBench can reach the DNS servers
		  ○ Check if HCIBench can ping vCenter by IP
		• Ensure there are no ACL or firewall rules blocking ping requests (ICMP)

5	Can HCIBench RVC into the vCenter?	• Run “rvc VCIP” in HCIBench command line • If a connection is established, ensure username and password are valid • Check if there is any ACL or firewall rules blocking HTTPS connections (TCP port 443)

6	Can HCIBench ping all the ESXi host?	• Ensure HCIBench (eth0) has a valid IPv4 address for the portgroup connected to vmnic0
		• Ensure HCIBench vmnic0 is connected to the correct portgroup
		• If the ESXi hosts and HCIBench are on different networks
		  ○ Ensure routing is configured between networks
		  ○ Ensure HCIBench has a default gateway configured

		• If HCIBench is connecting to ESXi host by hostname
		  ○ Ensure HCIBench has DNS configured
		  ○ Ensure HCIBench can reach the DNS servers
		  ○ Check if HCIBench can ping vCenter by IP
		• Ensure there are no ACL or firewall rules blocking ping requests (ICMP)

7	Can HCIBench SSH into all the ESXi hosts?	• If a connection is established, ensure username and password are valid • Check if there is any ACL or firewall rules blocking SSH connections (TCP port 22)
8	Can HCIBench RVC into all the ESXi hosts?	• Run “rvc ESXiIP” in HCIBench command line • If a connection is established, ensure username and password are valid • Check if there is any ACL or firewall rules blocking HTTPS connections (TCP port 443)
9	Does the HCIBench pre-validation pass the network validation?	Standard Virtual Switch • Ensure the physical switch ports are all configured the same for all the ESXi hosts • Ensure the portgroup is configured correctly for the physical network (VLAN) • Ensure the switches have a physical uplink configured • Ensure all hosts use the same portgroup configuration Distributed Virtual Switch • Ensure the physical switch ports are all configured the same for all the ESXi hosts • Ensure the portgroup is configured correctly for the physical network (VLAN) • Ensure the switches have a physical uplink configured NSX-T segments • Ensure the segment is configured correctly, attached to the correct transport zone, and the transport zone is properly configured • If traffic must be routed between different networks, ensure the edges are working and routing is properly configured

10	Can HCIBench ping the workers?	• Ensure HCIBench vmnic1 is connected to the correct portgroup
		• If using DHCP
		  ○ Ensure HCIBench eth1 has a valid IPv4 address
		  ○ Ensure there are enough IP for the workers
		• If using HCIBench static IP assignment
		  ○ Ensure the subnet selected does not conflict with other networks
		  ○ Ensure there are no other devices on the VM network using the same subnet
		• Ensure there are no ACL or firewall rules blocking ping requests (ICMP)

11	Can HCIBench SSH to the workers?	• Check if there is any ACL or firewall rules blocking SSH connections (TCP port 22)

Known Issues

Network fabrics with long MAC aging can experience IP resolution problems

Symptom: HCIBench is unable to connect to some workers after the initial validation passes. Subsequent validations also fail.

Cause: Some network fabrics have long MAC address aging periods that result in IP resolution problems when VM are rapidly destroyed and created. The default aging period is often 900 seconds (15 minutes). When a VM is created vSphere generates a unique MAC address for every virtual NIC. When the VM boots and assigned an IP address, the MAC-to-IP association is learned by the physical network fabric. If the VM is deleted the MAC address entry persists for the duration of the aging period and does not allow a new entry for the same IP but different MAC. Since HCIBench can rapidly delete and create VM, the IP address fails to resolve to the new MAC, but instead resolves to the original entry.

Workaround: This issue is caused by the underlying network, not HCIBench. Possible workarounds are changing the fabric aging period to a lower value, using different IP ranges for each test (including validation), or waiting out the aging period between deletion and creation.

Note: This issue should already be fixed as HCIBench 2.5, please email us vsanperformance@vmware.com if seeing this issue again with HCIBench 2.8.1

Network conflict with the HCIBench Docker Subnet Network

Symptom: Users are unable to connect to the Grafana service. All other features appear to work correctly. Either HCIBench eth0, eth1, or both interfaces have an IP address included in the 172.17.0.0/16 subnet.

Cause: HCIBench runs the Grafana service in a Docker container. The containers communicate with HCIBench using an internal network with subnet 172.17.0.0/16. By design, the HCIBench static IP service does not allow this selection. However, a conflicting subnet could be assigned to the management interface (static or DHCP) or the VM network interface (DHCP).

In the following example the management uses 172.17.0.0/24 which is a subnet included in 172.17.0.0/16. The docker interface (docker0) is assigned 172.17.0.1 and the container are assigned sequential IP starting at 172.17.0.2. The HCIBench routing table has an entry for 172.17.0.0/16 and a more specific entry for 172.17.0.0/24. When HCIBench tries to connect to the 172.17.0.2 the more specific route will be preferred resulting in the packets erroneously sent out from eth0 (Figure 49.

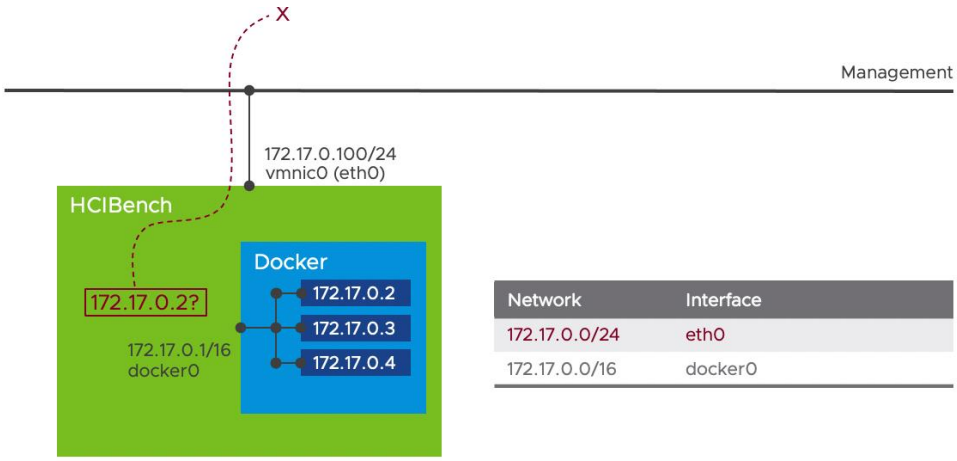


Figure 49. Docker Network Conflict Type 1

Workaround: There are two possible workarounds.

1. Ensure HCIBench does not use a subnet contained in the 172.17.0.0/16 subnet for eth0 or eth1.
2. Change the internal Docker network to another subnet that does not conflict.

HCIBench unable to connect to remote resources with IP belonging to the 172.17.0.1/16 subnet

Symptom: HCIBench does not have IP addresses on eth0 or eth1 that are included in the 172.17.0.0/16 subnet. When trying to connect to a vCenter or ESXi host with an IP included in the 172.17.0.0/16 subnet, it is unable to connect. Other devices on the management network can connect.

Cause: HCIBench runs the Grafana service in a Docker container. The containers communicate with HCIBench using an internal network with subnet 172.17.0.0/16 resulting in the HCIBench system to have an entry in the routing table listing the docker virtual interface (172.17.0.1) as the gateway for the 172.17.0.0/16 subnet. This entry will override the default gateway for any subnet included in 172.17.0.0/16.

In the following example the vCenter has an IP 172.17.100.54. When HCIBench tries to connect to the vCenter it finds the 172.17.0.0/16 entry in its routing table that is more specific than the default gateway and uses the specified docker0 interface (Figure 50).

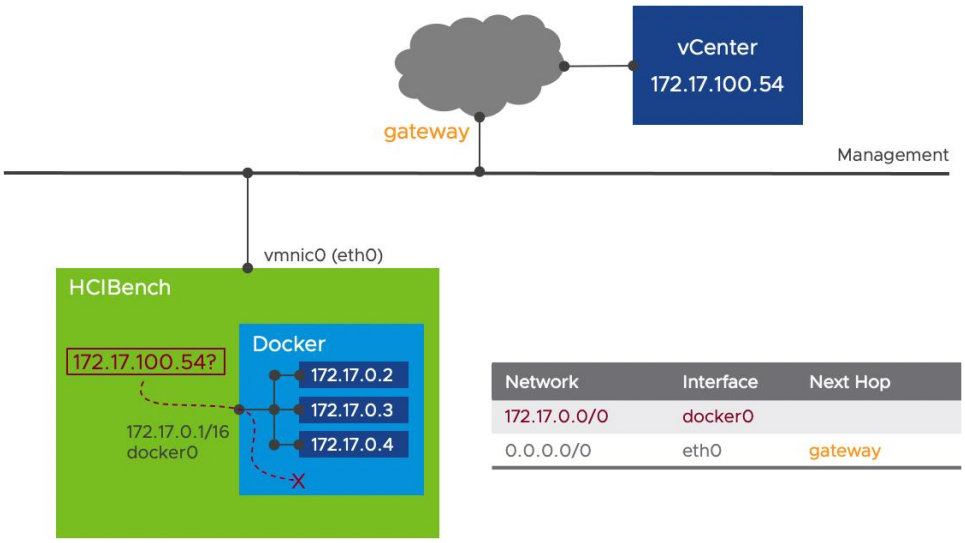


Figure 50. Docker Network Conflict Type 2

Workaround: There are two possible workarounds.

1. Ensure the remote systems do not use a subnet contained in the 172.17.0.0/16.
2. Change the internal Docker network to another subnet that does not conflict.

About the Author and Contributors

Charles Lee, Chen Wei, and Victor Chen in the VMware Product Enablement team wrote the original version of this paper. Catherine Xu, technical writer in the Product Enablement team, edited this paper to ensure that the contents conform to the VMware writing style.