Charon in the Azure Cloud Getting Started Guide

Contents

About this Guide

Contents

Intended Audience and Documentation Overview

This Getting Started guide is intended for anyone who needs to install, configure, or manage the Stromasys Charon emulators in the Azure cloud. It can be used as a starting point for installations that use the prepackaged Charon marketplace images available on the Azure Cloud marketplace, or when creating a Linux server in the cloud for a conventional RPM installation of Charon emulator products. A general working knowledge of Linux and its conventions is expected.

Please note:

- This Getting Started guide is a generic guide for all Linux-based Charon emulator products that are to be run in the Azure cloud. Therefore, you will find that some of the content is not applicable to your product. The focus of the document is on the common aspects of implementing a Charon emulator in the Azure cloud.
- The descriptions in this Getting Started guide are for Charon host systems running a **supported Linux operating system**. Windows host systems are not within the scope of this guide.

Overview of the relevant documentation for Charon emulators in cloud environments:

- This **Getting Started Guide** covers basic **cloud-specific aspects** when installing a Charon emulator product in the cloud. It can be used as a starting point for installations that use the prepackaged Charon marketplace images available on the Azure cloud marketplace, or when creating a Linux server in the cloud for a conventional RPM installation of Charon emulator products.
- The general **User's Guides** for each emulator product (please refer to the Stromasys documentation page) cover **features, configuration, and management of the respective Charon emulator product**.
- The **VE License Server User's Guide** in the Licensing Documentation section of the Stromasys Product Documentation site covers features, installation, and management of the **VE (Virtual Environment) license server** and the **VE licenses**.
- The **Release Notes** of your product provide important information regarding known problems and possible workarounds.

For additional information about this product, please contact Stromasys at the regional offices listed below in *Obtaining Technical Assistance or General Product Information*, send an email to **Team.Support.Azure@Stromasys.com**, or contact your Stromasys VAR.

Document Structure

- Introduction to Charon Emulators: overview of emulator concepts.
- Creating an Azure Cloud Instance for an Emulator Host: basic steps to create and launch a cloud instance to be used as a Charon host system.
- Installing the Charon-SSP Manager (Charon-SSP only): steps to install the main management tool for the cloud-based Charon-SSP host instance (if using the Charon-SSP emulator).
- Installing the VE License Server Software: steps to install the VE license server package if VE licenses are to be used and there is no existing VE license server yet.
- Accessing the Charon Cloud Instance: explains how to use SSH, SFTP, and the Charon-SSP Manager to access the cloud-based Charon host instance for management and file transfer, and how to set the initial management password.
- Additional Azure Instance Configuration for Charon: adding additional storage and network interfaces; cloud-specific networking aspects.

Please note:

- Cloud providers may change their management GUI without prior warning. Hence, the screenshots in this document may not always reflect the latest GUI appearance of the cloud provider. However, they will still provide an illustration of the described configuration steps.
- In general, the sample outputs in this document may show different versions than the one documented in this manual, but they are still representative of what a user will see.

Obtaining Documentation

The latest released version of this manual and other related documentation are available on the Stromasys support website at Product Documentation and Knowledge Base.

Obtaining Technical Assistance or General Product Information

Obtaining Technical Assistance

Several support channels are available to cover the Charon virtualization products.

If you have a support contract with Stromasys, please visit http://www.stromasys.com/support/ for up-to-date support telephone numbers and business hours. Alternatively, the support center is available via email at support@stromasys.com.

If you purchased a Charon product through a Value-Added Reseller (VAR), please contact them directly.

Obtaining General Product Information

If you require information in addition to what is available on the Stromasys Product Documentation and Knowledge Base and on the Stromasys web site you can contact the Stromasys team using https://www.stromasys.com/contact/, or by sending an email to info@stromasys.com.

For further information on purchases and the product best suited to your requirements, you can also contact your regional sales team by phone:

Region	Phone	Address
Australasia-Pacific	+852 3520 1030	Room 1113, 11/F, Leighton Centre 77 Leighton Road, Causeway Bay, Hong Kong, China
Americas	+1 919 239 8450	Stromasys LLC 871 Marlborough Ave, suite 100, Riverside CA 92507 USA
Europe, Middle-East and Africa	+41 22 794 1070	Avenue Louis-Casai 84 2nd Floor 1216 Cointrin Switzerland

Conventions

Notation	Description
$	The dollar sign in interactive examples indicates an operating system prompt for VMS. The dollar sign can also indicate non superuser prompt for UNIX / Linux.
#	The number sign represents the superuser prompt for UNIX / Linux.
>	The right angle bracket in interactive examples indicates an operating system prompt for Windows command (cmd.exe).
User input	Bold monospace type in interactive examples indicates typed user input.
<path>	Bold monospace type enclosed by angle brackets indicates command parameters and parameter values.
Output	Monospace type in interactive examples, indicates command response output.
[]	In syntax definitions, brackets indicate items that are optional.
...	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
dsk0	Italic monospace type, in interactive examples, indicates typed context dependent user input.

Definitions

| Term	Description
Host	The system on which the emulator runs, also called the Charon server
Guest	The operating system running on a Charon instance, for example, Tru64 UNIX, OpenVMS, Solaris, MPE or HP-UX

Related Documents

- Charon emulator documentation
- *VE License Server User's Guide* in Licensing Documentation

Introduction to Charon Emulators

Emulator Concepts

The Charon emulator products allow users of SPARC, PA-RISC, Alpha, and VAX legacy systems to replace their native hardware in a way that requires little or no change to the original system configuration. This means you can continue to run your applications and data without the need to switch or port to another platform. The Charon emulator software runs on commodity, x86 64-bit systems ensuring the continued protection of your investment.

The image below shows the basic concept of migrating physical hardware to an emulator:
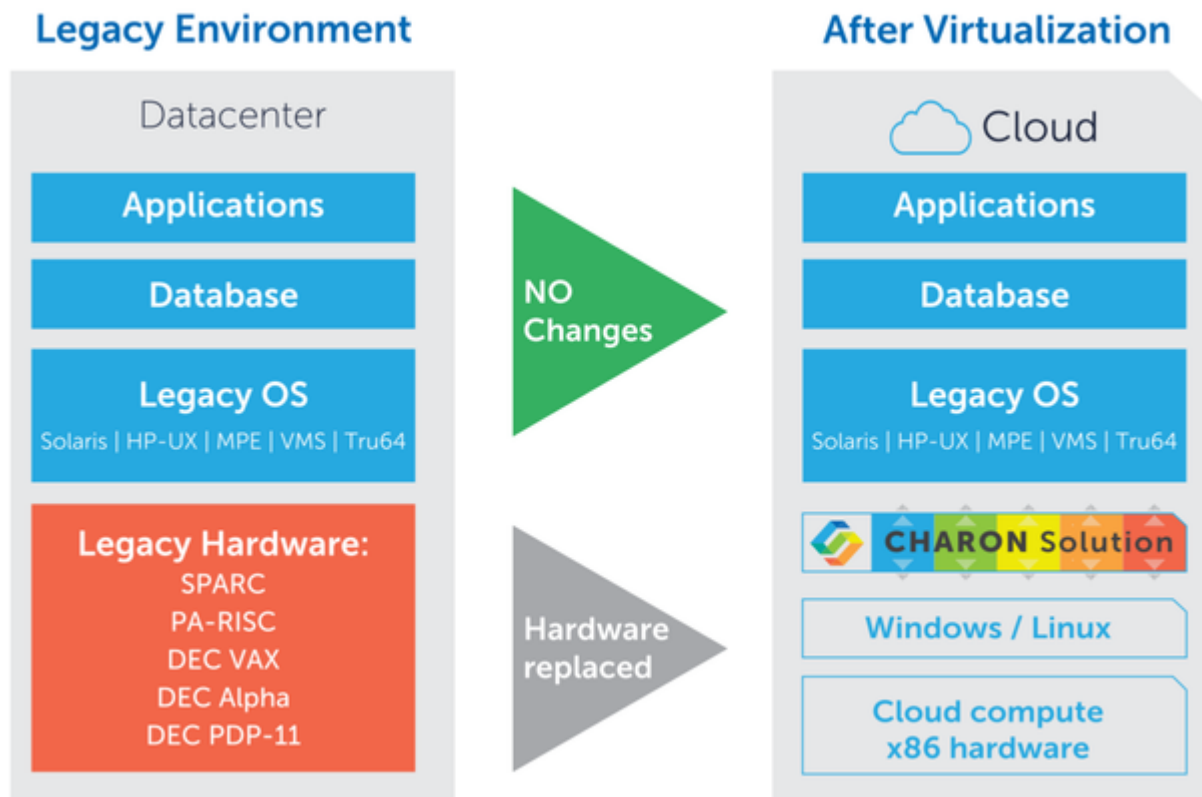
Charon Emulator Products

Stromasys currently provides the following emulator products:

- Charon-AXP/VAX (including PDP)
- Charon-SSP
- Charon-PAR

Please note: this guide is focused on an **implementation on Linux**.Therefore, it will not deal with the PDP emulator which is only available on Microsoft Windows.

General Installation Options for Emulators on Linux Cloud Instances

In general there are the following installation options for emulator products on Linux hosts in the cloud:

- Installation using **RPM packages**. This option is available for all products. The products were tested on AWS, GCP, Azure, Nutanix, and OCI. However, there may still be requirements of a legacy operating system that cloud environments cannot provide. Please contact your Stromasys representative for more information to discuss potential limitations, and if you would like to run Charon in a different Cloud environment.
- Installation based on **Stromasys-provided marketplace images** in selected clouds (not available for all products and all clouds). Such marketplace images include the Linux host operating system and the Charon emulator product. They can be provided for two different types of product licensing:
  - **Automatic Licensing (AL)** for which billing of cloud instance and Charon product is handled by the cloud provider. Public license servers for this option are operated by Stromasys (private AutoVE licensing with customer-operated license server is possible as an alternative).
  - **Virtual Environment Licensing (VE)** for which the billing of the cloud instance is handled by the cloud provider, the billing for the Charon product is handled by Stromasys or its partners. The license server is operated by the customer.

Marketplace image availability at the time of writing	
AWS	Charon-SSP: Amazon Linux (AL)
	Charon-SSP: VE
	Charon-PAR: VE
	Charon-AXP/VAX: VE - planned
Azure	Charon-SSP: VE
OCI	Charon-SSP: VE
	Charon-PAR: VE
GCP	Charon-SSP: VE - planned

For up-to-date availability information of marketplace images (in particular for AL marketplace images), **please contact your Stromasys representatives**.

At the time of writing the VE license server provided by Stromasys was available **for Linux systems** running on the following platforms:

VE license server availability at the time of writing	
Cloud platforms	- AWS - Azure - GCP - OCI - IBM - Nutanix
Other platforms	- VMware - Physical servers

VE licensing availability for Charon emulator products at the time of writing:

At the time of writing, the following Charon emulator products supported VE licensing:

- Charon-SSP
- Charon-PAR
- Charon-AXP/VAX (Linux only)

Management Options for Charon Emulator Instances in the Cloud

All Charon emulators running on a Linux host can be managed from the command-line.

Stromasys provides additional management tools for the different products. Availability of these products in cloud environments:

- Charon-SSP: the Charon Manager and the Charon Director fully support cloud-based Charon instances.
- Charon-PAR and Charon-AXP/VAX: at the time of writing, the Linux toolkit was not fully cloud-enabled yet. A new version supporting cloud-relevant features is planned (no release date known at the time of writing) and will be made available by Stromasys via the normal software distribution channels.

Creating an Azure Cloud Instance for an Emulator Host

This chapter describes how to set up a basic Charon host instance in Azure.

Contents

Charon Licensing for Stromasys Emulators in the Azure Cloud

Contents

General Information

This chapter describes the basic licensing options available for Charon instances in cloud environments. Not all options may be available in all clouds or for all products. Please refer to the User's Guide and the Release Notes of your product for detailed information.

Charon emulator products require a license to run. There are different licensing models. As **availability may differ depending on cloud environment and product,** please contact your Stromasys representative for any questions about product availability and licensing options.

The following options that are typically used in a cloud environment are described briefly in the sections below:

1. The cloud-specific, prepackaged **Charon AL** (Automatic Licensing) marketplace image utilizing either

- a **public, Stromasys-operated cloud-specific license server**, or
- a **private customer-operated VE license server** operating in **AutoVE** mode.

This licensing option offers pay-as-you-go billing for Charon-SSP. Please contact Stromasys Sales should you be interested in this option.

2. **Charon VE Licensing** (Virtual Environment) utilizing a **customer-operated, private VE license server** in a supported cloud environment. The license server can operate in **general VE mode** or in **AutoVE mode**. Charon emulators using general VE licensing are available as a

- prepackaged marketplace image on some cloud platforms, and
- in RPM package format for a conventional installation.

Guest operating system licensing requirements:

The user is responsible for any licensing obligations imposed by the legacy operating system running as a guest system inside the emulator, and has to provide the appropriate licenses.

Automatic Licensing Overview

Currently, only available for Charon-SSP and only on selected clouds. Please contact Stromasys Sales should you be interested in this option.

When a cloud instance is launched from a Charon AL marketplace image, this instance requires a license to run emulated systems. This license is obtained automatically from a Stromasys-operated license server upon first launch of the Charon AL instance.

Please note the following points:

- You will be billed by the cloud provider for your use of the Charon-SSP AL instance. Stromasys will not bill you directly.
- Charon AL instances require Internet access (via their own public IP address or NAT) for the license mechanism to work. If NAT is used, the gateway must be an instance in the same cloud (i.e., the source IP address must be from the address range of the same cloud provider in which the Charon host instance runs). The public, Stromasys-operated license servers must be reachable on port 8080 (SSP before version 5.5.5) or port 8081 (SSP version 5.5.5 or higher). Also, a DNS service must be reachable to resolve the host names of the license servers, or corresponding entries in *etc/hosts* must exist. The license server details are provided below.
- If you change the instance type after first launching the instance and thereby change the number of CPU cores (or if the number of CPU cores is changed by any other method), **the license will be invalidated** and you will have to create a new instance.
- Some licensing problems or other requirements (e.g., additional CPU cores needed) may make it necessary to move the emulator to a new instance. Therefore, it is strongly recommended to store all relevant emulator data on a separate disk volume that can easily be detached from the old instance and attached to a new instance.
- If you need to set up a new Charon host instance with AL licensing, you must create it via launching a new instance from the appropriate marketplace image and (as necessary) copying/moving the emulator data and configuration to the new instance. Cloning an instance with AL licensing will result in a system with an invalid license.
- Should access to the license be lost, there is a grace period of 24 hours. If license access is not restored within this period, the emulator will stop (if a guest system is running at the time, this is the equivalent of disconnecting the power without clean shutdown, i.e., it may lead to loss of data).

At the time of writing, the public Stromasys-operated license servers in Azure were

- **cloud-azure1.stromasys.com** and
- **cloud-azure2.stromasys.com**
- TCP port 8080 for SSP version before 5.5.5 (old certificates)
- TCP port 8081 for SSP version 5.5.5 or higher (new certificates)

VE Licensing Overview

Charon VE License Characteristics

The main characteristics of VE licenses are the following:

- Software licenses only.
- Installed on the Charon host or on a separate license server. A separate license server reduces the risk of accidentally invalidating a license.
- Require the Charon VE license server software (the RPM package is included in the prepackaged, cloud-specific marketplace Charon VE image).
- Require matching Charon emulator software (preinstalled on the prepackaged, cloud-specific marketplace Charon VE image).
- Different modes of operation:
  - For **general VE mode**, the customer is billed by Stromasys depending on the number and type of the emulated systems allowed by the installed license(s). The license server software itself is free of charge.
  - **AutoVE mode** is an extension of automatic licensing and introduces metered billing (by the cloud-provider) for VE licenses in cloud environment. It defines how many Charon host instances can be run based on the respective license. The number of emulated systems on each host instance is limited by the host resources, not the license. The emulator host instance must be based on an Charon AL marketplace image, and license server and emulator host must be in the same cloud. At the time of writing, this option was available only for Charon-SSP on selected clouds. **Please contact Stromasys Sales should you be interested in this option**.
  - The license server for both modes is managed by the customer.

If supported by the cloud provider, the VE license server instance can be moved to a different subnet, as long as the original instance can be moved. It is also possible to backup and restore (to the same instance) the license server data. However, the following actions will **invalidate the license**:

- Changing the number of CPU cores of the license server system.
- Copying the license server data to a different instance.
- Seriously damaging the root filesystem of the license server system.
- Re-installing the license server system.
- Copying the virtual machine on which the license server runs. This includes cloning a virtual machine, or recovering a backup into a new virtual machine.

Please note: Charon-SSP 5.5.5 and VE license server 2.1.3 introduce new certificates and the option to use user-defined certificates. Please review the Virtual Environment (VE) License Server Documentation for details and to avoid possible compatibility problems between old and new versions.

Charon VE License Server Communication Requirements

For proper operation, the system on which the license server runs has the following communication requirements:

Communication with the cloud infrastructure:

For proper functionality, the Azure cloud instance on which the license server runs must be able to communicate with the cloud infrastructure:

- The metadata server of the cloud environment (169.254.169.254)

It must also be able to communicate with the client systems using the license. The following ports are used for this communication **by default**:

- **TCP/8083**: must be permitted from the client to the license server to enable the use of the license by the client.
- **TCP/8084**: must be permitted by the license server for any system that should access the web interface to display license information. It is not required for providing a license to a license client system.

In current VE license server versions, the default ports can be changed in the file **/opt/license-server/config.ini**. Please refer to the VE license server guide for details.

Basic License Installation Steps Before an Emulator Can be Started

If there is no VE license server running already, decide on which cloud instance it should run and **install the VE License Server package** on the selected system. The VE License Server RPM package is included in the prepackaged Charon VE marketplace images. Alternatively, Stromasys will provide a download location. See *Installing the VE License Server Software*.

- If you don't already have a license, contact your Stromasys representative to procure an appropriate license.
- Log in on your Charon VE License Server instance.
- Create a C2V file and send it to the email address Stromasys will provide to you.
- Install the V2C file you will receive from Stromasys.
- Configure the emulator instance(s) to use the license server.

Please refer to the Licensing Documentation for more information.

Charon Host Cloud Instance Prerequisites

Content

Cloud Instance Prerequisites for Charon

The cloud instance for running a Charon emulator is also called the Charon host. When creating a cloud instance for Charon, there are **hardware and software prerequisites**.

Hardware prerequisites of a cloud instance:

For a cloud instance, the hardware characteristics of the instance (e.g., how many CPU cores and how much memory your virtual Charon host system will have) are defined by selecting an instance type or shape. This determines the virtual hardware that will be used for the Charon host instance in the cloud.

Software prerequisites of a cloud instance:

If you use a Charon marketplace image to launch your instance, all Linux host operating system requirements are fulfilled. For conventional emulator installations, the Charon host cloud instance must be based on a supported Linux version.

Cloud Instance Software Prerequisites Overview

This section provides a basic overview of supported Linux distributions and versions for the current versions of the different emulator products (at the time of writing).

Please always refer to the Charon emulator documentation of your emulator product and version for details and up-to-date information.

Emulator product	Red Hat Linux	Rocky Linux	Oracle Linux	CentOS
Charon-SSP version 5.6.7	7.0 - 7.9 8.1 - 8.6 9.0 - 9.2	8.1 - 8.6 9.0 - 9.2	7.0 - 7.9 8.1 - 8.6 9.0 - 9.2	7.0 - 7.9
Charon-AXP/VAX 4.12	7.x 8.x 9.x			7.x
Charon-PAR 3.0.12	7.x (min. 7.4) 8.x 9.x	8.x 9.x	7.x (min. 7.4) 8.x 9.x	7.x (min. 7.4)

Only 64-bit versions of Linux are supported.

Cloud Instance Hardware Prerequisites Overview

The sections below provided an overview of basic hardware prerequisites **for the different emulator products**. **Please note the following points regarding the sizing guidelines**:

- The sizing information below—in particular regarding number of host CPU cores and host memory—show the **minimum requirements**.
- Every deployment situation must be reviewed and the **actual host sizing** has to be adapted as necessary.
- In a **hyper-threading environment**, for best performance, the number of CPU cores (i.e., real/physical CPUs) should be sufficient to fulfill CPU requirements of the active emulators, thus avoiding high-workload threads sharing one physical CPU core. If possible, hyper-threading should be disabled.

Please always refer to the Charon emulator documentation of your emulator product and version for details and up-to-date information.

Hardware Prerequisites for Charon-SSP Cloud Host Instances

Charon-SSP General Notes

Please note the following points regarding the sizing guidelines:

- The sizing guidelines below—in particular regarding number of host CPU cores and host memory—show the **minimum requirements**. Every deployment situation must be reviewed and the actual host sizing has to be adapted as necessary. For example, the number of CPU cores available for I/O must be increased if the guest applications produce a high I/O load. Also, a system with many emulated CPUs is typically able to create a higher I/O load and thus the number of CPU cores available for I/O may have to be increased. In a hyper-threading environment, for best performance, the number of CPU cores (i.e., real/physical CPUs) should be sufficient to fulfill CPU requirements of the active emulators, thus avoiding high-workload threads sharing one physical CPU core.
- The CPU core allocation for emulated CPUs and CPU cores for I/O processing is determined by the configuration. See *CPU Configuration* in the general *Charon-SSP User's Guide* for more information about this and the default allocation of CPU cores for I/O processing.

Important general information:

- To facilitate a fast transfer of emulator data from one cloud instance to another, it is strongly recommended to store all relevant emulator data on a separate disk volume that can easily be detached from the old instance and attached to a new instance.
- Please make sure to dimension your instance correctly from the beginning (check the minimum requirements below). The Charon-SSP license for **Charon-SSP AL** is created when the instance is first launched. Changing later to another instance size/type and thereby changing the number of CPU cores will invalidate the license and thus prevent Charon instances from starting (new instance required). If planning to use the **Charon-SSP AL instance in AutoVE mode**, be sure to include the AutoVE server information before first launch, otherwise the public license servers will be used. The license for **Charon-SSP VE** is created based on the fingerprint taken on the license server. If the license server is run directly on the emulator host and the emulator host later requires, for example, a change in the number of CPU cores, the license will be invalidated, unless it is transferred out of the system beforehand - then the resulting transfer file and a new C2V file from the changed system can be used to obtain a new license from Stromasys.

Charon-SSP Cloud Instance Prerequisites

General CPU requirements: Charon-SSP requires modern x86-64 architecture processors. This could be Intel Servers based on Haswell v3 processors or later, or Desktop Core I7 (CPU frequency at least 3.0 GHz). AMD processors of the same or higher performance are also supported.

Minimum requirements for Charon-SSP:

- Minimum number of host system CPU cores:
  - At least one CPU core for the host operating system, plus:
  - **For each emulated SPARC system:**
    - One CPU core for each emulated CPU of the instance, plus:
    - At least one additional CPU core for I/O processing (at least two, if server JIT optimization is used). See the *CPU Configuration* section mentioned above for configuration options. By default, Charon will assign 1/3 (min. 1; rounded down) of the number of CPUs visible to the Charon host to I/O processing.

- Minimum memory requirements:
  - 4GB or more of RAM for the Linux host operating system. The actual requirements may be higher and will depend on the requirements of the non-emulator services running on the Linux host. The previous recommendation of at least 2GB of RAM for the Linux host will still be valid for many systems, but the increasing requirements of the Linux operating system and applications have led to the updated recommendation for new installations. Plus:
  - **For each emulated SPARC system:**
    - The configured memory of the emulated instance, plus:
    - 2GB of RAM (6GB of RAM if server JIT is used) to allow for DIT optimization, emulator requirements, run-time buffers, SMP and graphics emulation.
- If hyper-threading is enabled on modern x86-64 CPUs, two threads can run on one physical CPU core providing two logical CPUs to the host operating system. If possible, disable hyper-threading on the Charon-SSP host. However, this is frequently not possible in VMware and cloud environments, or it is unclear whether hyper-threading is used or not. The Charon-SSP hyper-threading option enables Charon-SSP to adapt to such environments. See the *CPU Configuration* section in your general *Charon-SSP User's Guide* mentioned above for detailed configuration information. **Please note**: for best performance, Charon-SSP threads should not share a physical CPU core – enough physical cores should be available on the host system to satisfy the requirements of the configured emulator(s).
- One or more network interfaces, depending on customer requirements.
- Charon-SSP/4U+ and Charon-SSP/4V+ must run on **physical** hardware supporting Intel VT-x/EPT or AMD-v/NPT (baremetal instances) and therefore **cannot run in all cloud environments**. Please check your cloud provider's documentation for the availability of such hardware. In addition, note the following points:
  - Charon-SSP/4U+ and Charon-SSP/4V+ are only available when using a Linux kernel supported by Stromasys. Please refer to the general Charon-SSP user's guide for details (see CHARON-SSP for Linux).
  - Please contact Stromasys or your Stromasys VAR if you need this type of emulated SPARC hardware to discuss your requirements in detail.

Hardware Prerequisites for Charon-AXP Cloud Host Instances

Charon-AXP Number of CPU Cores and CPU Speed

Each Charon-AXP emulated CPU requires a corresponding physical core. So the total number of the host CPUs must exceed the number of emulated CPUs since some of the host CPUs must be dedicated to serving CHaron I/O operations and host operating system needs. If several Charon instances run in parallel, the required number of CPU cores is cumulative.

The following table lists the minimum and recommended number of CPUs required for each virtual HP Alpha instance (note that each Charon instance is able to run on 2 CPU cores hosts, but this configuration does not support emulation of all the virtual CPUs):

CHARON-AXP product	Minimum number of host CPU cores	Recommended number of host CPU cores
HP AlphaServer 400 - HP AlphaServer 4100	2	2
HP AlphaServer DS10/DS10L/DS15	2	2
HP AlphaServer DS20/DS25	4	4
HP AlphaServer ES40/ES45	6	8
HP AlphaServer GS80	10	16
HP AlphaServer GS160	18	32
HP AlphaServer GS320	34	48

When starting, the Charon-AXP software checks the available number of host CPU cores. This check is based on the maximum number of AXP CPUs that can be emulated if this number is not restricted by the **n_of_cpus** parameter. If the available number of host CPU cores is below this number, Charon-AXP will issue a warning message even if the requirements for the configured number of AXP CPUs are fulfilled. The Charon-AXP software will work despite this warning if the requirements for the configured number of AXP CPUs are fulfilled.

Hyper-threading must be switched off completely. Normally, hyper-threading is disabled in the BIOS settings. In cloud environments, it depends on the cloud provider if and how the number of vCPUs per core can be limited to one.

CPU speed: the general recommendation is that a higher CPU frequency will lead to better performance of the emulated HP Alpha system. **The minimum recommendation is at least 3 GHz**.

Charon-AXP Memory Requirements

Minimum host memory size:

- It depends on the amount of Alpha memory to be emulated and on the number of Charon-AXP instances to be run on one host.
- It is calculated using the following formula:
  Minimum host memory = (2GB + the amount of HP Alpha memory emulated) per Charon-AXP instance

Charon-AXP Disk Storage

The instance needs sufficient disk space for the host and guest operating system and all the emulated storage devices of the legacy system.
To facilitate a fast transfer of emulator data from one cloud instance to another, it is strongly recommended to store all relevant emulator data on a separate disk volume that can easily be detached from the old instance and attached to a new instance

Charon-AXP Ethernet Adapters

Charon- AXP networking requires one of the following:

- One or more dedicated host Ethernet adapters; their number must be equal to the emulated adapters to be configured in Charon-AXP. One adapter must be available to the host for TCP/IP networking, management interface, etc. For this option, you must take into account the cloud-typical restrictions when using dedicated network adapters (e.g., IP unicast only, MAC must not be changed, cloud-provided IP address must be used by guest). Please refer to your cloud-providers documentation and the section *Charon Cloud Networking Information* in this guide.
- Virtual network interfaces connected to a local bridge configured inside the Charon host.

Hardware Prerequisites for Charon-VAX Cloud Host Instances

Charon-VAX Number of CPU Cores and CPU Speed

Each CHARON emulated CPU requires a corresponding physical core. So the total number of the host CPUs must exceed the number of emulated CPUs since some of the host CPUs must be dedicated to serving CHARON I/O operations and host operating system needs. If several CHARON instances run in parallel, the required number of CPU cores is cumulative.

The table below lists the minimum and recommended number of CPUs required for each product:

CHARON-VAX model	Minimal number of CPU cores	Recommended number of CPU cores
VAX 6610	2	4
VAX 6620	3	4
VAX 6630	4	6
VAX 6640	6	8
VAX 6650	8	12
VAX 6660	8	12
Other models	2	2

When starting, the CHARON-VAX software checks the available number of host CPU cores. This check is based on the maximum number of VAX CPUs that can be emulated. Therefore the number of host CPU cores recommended for the maximum number of emulated CPUs - as shown in the right column of the table above - must be available. If the available number of host CPU cores is below this number, CHARON-VAX will issue a warning message. The CHARON-VAX software will work despite this warning.

Hyper-threading must be switched off completely if possible. Normally, hyper-threading is disabled in the BIOS settings. In cloud environments, it depends on the cloud provider if and how the number of vCPUs per core can be limited to one.

CPU type and speed:

- Since Charon-VAX utilizes the LAHF instruction in VAX CPU emulation don't use early AMD64 and Intel 64 CPUs in a Charon host system since they do not support this instruction. AMD introduced the instruction with their Athlon 64, Opteron and Turion 64 revision D processors in March 2005 and Intel introduced it with the Pentium 4 G1 stepping in December 2005.
- CPU speed: the general recommendation is that the higher the CPU frequency is, the better the emulated VAX performances will be. The minimum recommendation is at least 3 GHz.

Charon-VAX Memory Requirements

Minimum host memory size:

- It depends on the amount of VAX memory to be emulated and on the number of Charon-VAX instances to be run on one host.
- It is calculated using the following formula:
  Minimum host memory = (2GB + the amount of VAX memory emulated) per Charon-VAX instance

The maximum amount of VAX memory that can be created in the CHARON-VAX/66x0 products and supported by OpenVMS/VAX is *3584* Mb.

Charon-VAX Disk Storage

The total amount of disk space required for Charon-VAX can be calculated as a sum of all the disk/tape image sizes plus 50 MB for the Charon software plus the space required for the host operating system. Temporary disk storage is often needed when setting up a new virtual machine (for source disks backups storage, software installation kits, etc...).

When virtual disks/tapes are used to represent physical disk drives / magnetic tapes, the disk/tape image files have the same size as their hardware equivalent, regardless of their degree of utilization.

Charon-VAX Ethernet Adapters

Charon- VAX networking requires one of the following:

- One or more dedicated host Ethernet adapters; their number must be equal to the emulated adapters to be configured in Charon-VAX. One adapter must be available to the host for TCP/IP networking, management interface, etc. For this option, you must take into account the cloud-typical restrictions when using dedicated network adapters (e.g., IP unicast only, MAC must not be changed, cloud-provided IP address must be used by guest). Please refer to your cloud-providers documentation and the section *Charon Cloud Networking Information* in this guide.
- Virtual network interfaces connected to a local bridge configured inside the Charon host.

Hardware Prerequisites for Charon-PAR Cloud Host Instances

The host hardware on which Charon-PAR runs must fulfill at least the following requirements:

- Intel or AMD x86-64 hardware platform (AMD support starting with Charon-PAR version 3.0.6)
- At least 3GHz, 3.4GHz or higher is recommended
- SSE 4.2 and FMA required
- **CPU cores required**:
  - at least one CPU core for the host operating system, **and**
  - at least 2 cores per emulated CPU (3 cores for future advanced DIT)
- **RAM requirements**:
  - 4GB RAM plus 1.1 times the emulated RAM size for the emulator. Additional memory is required for I/O buffering and other Linux processes (2 to 4 GB can be used as a starting point).
  - at least 24GB RAM for N4000 models
- One Ethernet interface for the host system and one Ethernet interface for each emulated Ethernet NIC. Alternatively, TAP interfaces can be used. Please note that the use of TAP interfaces is restricted to internal bridges in cloud environments (that is, a bridge cannot be linked to a NIC connected to the cloud LAN).
- At least one available USB port if HASP hardware licenses are to be used.
- Disable NUMA balancing.
  - To check the status, use the command: `cat /proc/sys/kernel/numa_balancing` (0 = off, 1 = on).
  - To disable NUMA balancing temporarily, use the command: `sysctl -w kernel.numa_balancing=0`
  - To make the configuration permanent you must create a **sysctl** configuration entry:
    - Create a file in **/etc/sysctl.d/** (e.g., 90-numa.conf)
    - Add the line `kernel.numa_balancing=0`
    - **Please note**: it still works to add the line `kernel.numa_balancing=0` directly to **/etc/sysctl.conf** but this file has been deprecated in newer Linux versions.

- If possible in your cloud environment, restrict the number of vCPUs on each CPU core to one (no hyper-threading).

Please note:

- The sizing guidelines above—in particular regarding number of host CPU cores and host memory—show the **minimum requirements**. Every use case has to be reviewed and the actual host sizing has to be adapted as necessary.
- If the host CPU does not support the required extensions (SSE and FMA), the emulator will not start. Instead, it will stop with an error message indicating the missing features.
- To identify the capabilities of the host system CPU, use one of the following commands: `lscpu` or `cat /proc/cpuinfo`.

Hyper-Threading on Cloud Instances

Some emulator products require that hyper-threading (two threads per physical CPU Core) be disabled. Normally, this is handled via the system BIOS. However, not all cloud providers offer an easy way to configure an instance without hyper-threading. In such cases, it may be necessary to perform an equivalent configuration on the Linux level.

Cloud-specific options:

- AWS: the threads per CPU core can be configured. See https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-specify-cpu-options.html
- GCP: the threads per CPU core can be configured. See https://cloud.google.com/compute/docs/instances/set-threads-per-core
- Azure: manual configuration on the Linux level
- OCI: manual configuration on the Linux level
- IBM: manual configuration on the Linux level. See https://cloud.ibm.com/docs/vpc?topic=vpc-disabling-hyper-threading

Basic steps of a manual configuration on the Linux level:

Please note: not all cloud instances may support disabling hyper-threading. Please refer to your cloud providers documentation.

- Check if hyper-threading is enabled on your system. If the **lscpu** command shows two threads per core then hyper-threading is on. Example:

```
$ lscpu
Architecture:          x86_64
  CPU op-mode(s):      32-bit, 64-bit
  Address sizes:       39 bits physical, 48 bits virtual
  Byte Order:          Little Endian
CPU(s):                8
  On-line CPU(s) list: 0-7
Vendor ID:             GenuineIntel
  Model name:          Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
    CPU family:        6
    Model:             94
    Thread(s) per core: 2
....
```

- Check which threads share one CPU core. In the example below threads 0 and 4 share core 0, threads 1 and 5 share core 1, etc.

```
$ cat /sys/devices/system/cpu/cpu*/topology/thread_siblings_list
0,4
1,5
2,6
3,7
0,4
1,5
2,6
3,7

or

$ lscpu --extended
CPU NODE SOCKET CORE L1d:L1i:L2:L3 ONLINE   MAXMHZ   MINMHZ       MHZ
  0    0      0    0 0:0:0:0          yes 4000,0000 800,0000 2100,0669
  1    0      0    1 1:1:1:0          yes 4000,0000 800,0000 2100,0020
  2    0      0    2 2:2:2:0          yes 4000,0000 800,0000 2100,0920
  3    0      0    3 3:3:3:0          yes 4000,0000 800,0000 2100,0281
  4    0      0    0 0:0:0:0          yes 4000,0000 800,0000 2099,9751
  5    0      0    1 1:1:1:0          yes 4000,0000 800,0000 2100,0039
  6    0      0    2 2:2:2:0          yes 4000,0000 800,0000 2100,0371
  7    0      0    3 3:3:3:0          yes 4000,0000 800,0000 2100,0769
```

- Disable threads such that there is only one active thread per core. Example:

```
$ sudo bash -c 'echo 0 > /sys/devices/system/cpu/cpu4/online'
...
[wer@square GettingStartedGuides]$ lscpu --extended
CPU NODE SOCKET CORE L1d:L1i:L2:L3 ONLINE    MAXMHZ    MINMHZ        MHZ
  0    0      0    0 0:0:0:0          yes 4000,0000 800,0000 2200,1321
  1    0      0    1 1:1:1:0          yes 4000,0000 800,0000 2200,0020
  2    0      0    2 2:2:2:0          yes 4000,0000 800,0000 2200,0300
  3    0      0    3 3:3:3:0          yes 4000,0000 800,0000 2200,5020
  4    -      -    - -                 no        -        -          -
  5    0      0    1 1:1:1:0          yes 4000,0000 800,0000 2200,0171
  6    0      0    2 2:2:2:0          yes 4000,0000 800,0000 2200,0020
  7    0      0    3 3:3:3:0          yes 4000,0000 800,0000 2200,0891
```

- Make configuration permanent by using a custom Linux startup script.

Creating and Configuring an Azure Cloud Instance for Charon

Contents

General Prerequisites

As this description shows the basic setup of a Linux instance in Azure, it does not list specific prerequisites. However, depending on the use case, the following prerequisites should be considered:

- To set up a Linux instance in Azure, you need an Azure account.
- Secondly, **prerequisites will be different depending on the planned use of the instance**:
  - Option 1: the instance is to be used as a **Charon emulator host system**:
    - Refer to the hardware and software prerequisite sections of the *User's Guide* and/or *Getting Started guide* of your Charon product to determine the exact hardware and software prerequisites that must be fulfilled by the Linux instance.
      The **image** you use to launch your instance and the **instance type** you chose determine the software and hardware of your cloud instance.
    - A Charon product **license** is required to run emulated legacy systems. Contact your Stromasys representative or Stromasys VAR for details.
  - Option 2: the instance is to be used as a dedicated **VE license server**:
    - Refer to the VE License Server Guide for detailed prerequisites.
- Certain legacy operating systems that can run in the emulated systems provided by Charon emulator products require a license of the original vendor of the operating system. The user is responsible for any licensing obligations related to the legacy operating system and has to provide the appropriate licenses.

Azure Login and New Instance Launch

Logging in to your Azure account

To log in perform the following steps:

- Go to **portal.azure.com**. You will see a Microsoft Azure login screen.
- Enter your login credentials.
- Upon successful login, the Azure home screen will be displayed as shown in the example below:
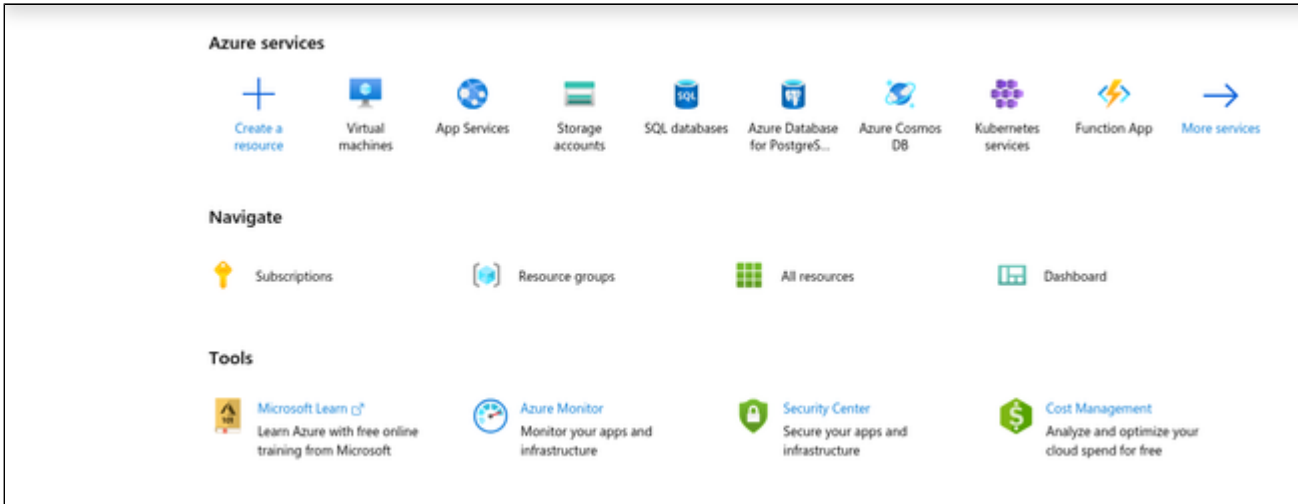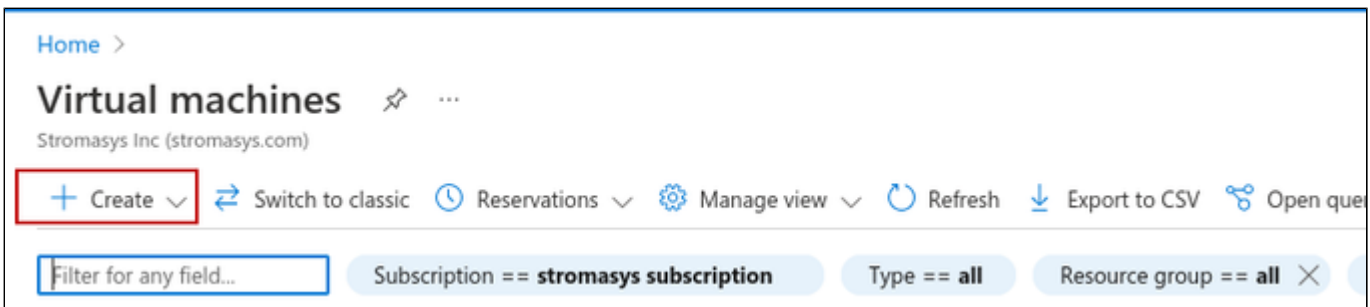
Creating a Virtual Machine

Step 1: Click on the **Virtual machines** or on the **Marketplace** icon on the home page. If you create your instance via the Marketplace icon, please select the Charon listing from the Marketplace offerings, select to create an instance, and continue with Step 3.
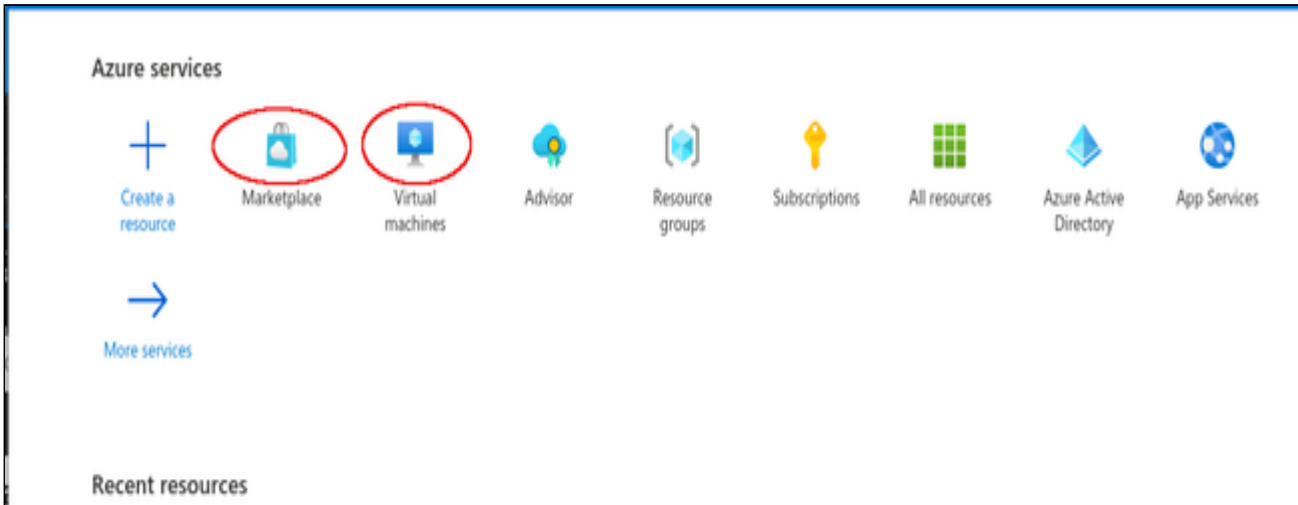
Clicking on **Virtual machines** opens the virtual machines overview list.

Step 2: Click on the **Create** link in the overview list.

For a basic setup, select **Azure virtual machine** from the drop down list opened by the **Create** link. This opens the **Basics** tab of the **Create a Virtual Machine** window.

Step 3: Enter your data on the **Basics** tab. Mandatory data are, for example:

- Your subscription
- Existing resource group (or click on **Create new**)
- Virtual machine name (cannot be changed after launching the instance)
- Region for the virtual machine
- The Azure image from which to launch your instance. Click on **See all images** to select the correct image. If installing a prepackaged marketplace Charon image, select the matching image. If you plan to install Charon using RPM packages, use a Linux version supported by your Charon emulator product.
- Size of your VM (click on **See all sizes** to see a list of available sizes)

Basics tab **upper part** sample:

Select the image from which to launch your instance and the correct size of your instance (please review the sizing requirements above). Enter the other information in accordance to your environment.

Home > Virtual machines >

Create a virtual machine   · · ·

| Basics | Disks | Networking | Management | Advanced | Tags | Review + create |

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. Learn more ⬀

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ	stromasys subscription ⌄
└─ Resource group * ⓘ	stromasys-lab ⌄
	Create new

Instance details

Virtual machine name * ⓘ	we-testingAL ✓
Region ⓘ	(US) East US ⌄
Availability options ⓘ	No infrastructure redundancy required ⌄
Security type ⓘ	Standard ⌄
Image * ⓘ	Charon-SSP-v5.4.3-el8-build1 - Gen1 ⌄
	See all images	Configure VM generation
Azure Spot instance ⓘ	☐
Size * ⓘ	Standard_D4s_v3 - 4 vcpus, 16 GiB memory ($140.16/month) ⌄
	See all sizes

| Review + create | | < Previous | Next : Disks > |

Basics tab **lower part** sample:

- Enter the user **sshuser** as the administrative user.
- Select **SSH public key** authentication. You can then use **one of the following** steps to install your SSH public key.
  - Let Azure create a new key-pair for you.
  - Use the public key from a key-pair on your computer. As shown in the example below, you will have to past your public key into the field provided.
  - Use a key-pair previously created on Azure.
- The default allowed inbound port will allow SSH connections without limiting the source IP range. Some images may also have preconfigured access rules that cannot be changed during the launch of the instance. In either case, remember to adapt the rules to your requirements after creating the instance or in the Networking tab (advanced) during the creation of the instance.

Please note: if your management system supports it, for RHEL 9.x, Rocky Linux 9.x, and Oracle Linux 9.x use SSH key types ECDSA or ED25519. This will allow connecting to these Charon host Linux systems using an SSH tunnel without the default crypto-policy settings on the Charon host having to be changed for less secure settings. This is, for example, important for the Charon-SSP Manager. See also: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9/html/security_hardening/using-the-system-wide-cryptographic-policies_security-hardening.

Click on **Next: Disks**. This will open the **Disks** tab of the VM creation window.

Step 4: Define the disks for your VM.

Please note: By default, Azure VMs have one operating system disk and a temporary disk for short-term storage (mounted on /mnt/resource and not persistent). The recommended minimum system disk size is 30GB. You can attach existing additional data disks, or create new disks and attach them.

Disks tab sample:

Home > Virtual machines >

Create a virtual machine   ...

Basics   **Disks**   Networking   Management   Advanced   Tags   Review + create

Azure VMs have one operating system disk and a temporary disk for short-term storage. You can attach additional data disks. The size of the VM determines the type of storage you can use and the number of data disks allowed. Learn more ⬡

Disk options

OS disk type * ⓘ	Premium SSD (locally-redundant storage)	⌄

Delete with VM ⓘ   ☐

Encryption at host ⓘ   ☐

> ⓘ Encryption at host is not registered for the selected subscription. Learn more about enabling this feature ⬡

Encryption type *	(Default) Encryption at-rest with a platform-managed key	⌄

Enable Ultra Disk compatibility ⓘ   ☐
Ultra disk is supported in Availability Zone(s) 1,2,3 for the selected VM size Standard_D4s_v3.

Data disks for we-testingAL

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching	Delete with VM ⓘ

Create and attach a new disk   Attach an existing disk

Review + create   < Previous   Next : Networking >

Click on **Next: Networking**. This will open the **Networking** tab of the VM creation window.

Step 5: Enter the necessary information in the **Networking** tab.

On this tab, you can define the network configuration of your VM:

- Virtual Network (existing or new)
- Subnet (default or other subnet)
- Whether a public IP should be assigned or not (note that if you use an image requiring a public, Stromasys-operated license server, this server must be accessed via a public IP address from the Azure range)
- Basic, advanced, or preconfigured security settings (which ports are open for access to the VM).
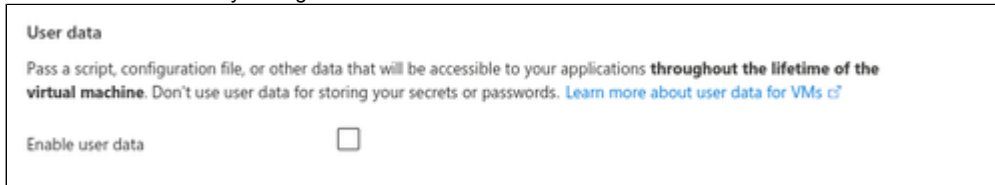
Networking tab sample:

Optionally, you can proceed to the **Management**, **Advanced**, and **Tags** tabs to configure additional details of your VM. However, for a basic test, this is not required. Click on **Review + Create** to proceed to the review screen.

Step 6: additional configuration for **AutoVE** setup.

If the instance is launched from a Charon AL marketplace image and is planned to use AutoVE licensing (instead of the public license servers), you must add the corresponding information to the instance configuration **before** the first launch of the instance:
The AutoVE license server information is entered as instance **User Data**. In the initial instance configuration window, go to the **Advanced** section.

- Open it and scroll down to the **User Data** section.

- Enable the **User Data** by ticking the checkbox.

User data

Pass a script, configuration file, or other data that will be accessible to your applications **throughout the lifetime of the virtual machine**. Don't use user data for storing your secrets or passwords. Learn more about user data for VMs ⊡

Enable user data ☐

- Then enter the correct primary and backup AutoVE servers as shown in the example below:
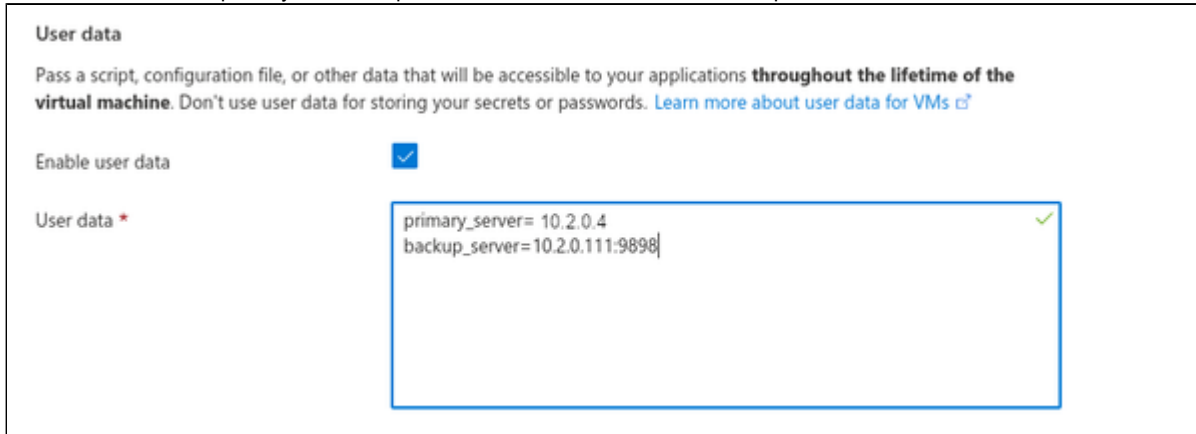
User data

Pass a script, configuration file, or other data that will be accessible to your applications **throughout the lifetime of the virtual machine**. Don't use user data for storing your secrets or passwords. Learn more about user data for VMs ⊡

Enable user data ☑

User data *
```
primary_server= 10.2.0.4
backup_server=10.2.0.111:9898
```

Valid User Data configuration options:

- `primary_server=<ip-address>[:<port>]`
- `backup_server=<ip-address>[:<port>]`

where

- *<ip-address>* stands for the IP address of the primary and the backup server as applicable, and
- *<port>* stands for a non-default TCP port used to communicate with the license server (default: TCP/8083).

Please note: at least one license server must be configured at initial launch to enable AutoVE mode. **Otherwise, the instance will bind to one of the public license servers operated by Stromasys.**

Step 7: Check the data on the Review + Create screen and create VM.

Verify that the checks passed successfully and click on **Create** to create the VM.

Sample **Review+Create** screen:

Home > Virtual machines > Create a virtual machine

Create a virtual machine

✓ Validation passed

Basics Disks Networking Management Advanced Tags Review + create

PRODUCT DETAILS

Standard D2s v3
by Microsoft
Terms of use | Privacy policy

Subscription credits apply ⓘ
0.0810 EUR/hr
Pricing for other VM sizes

TERMS

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the Azure Marketplace Terms for additional details.

⚠ **You have set SSH port(s) open to the internet.** This is only recommended for testing. If you want to change this setting, go back to Basics tab.

Basics

Subscription	Free Trial
Resource group	we-test1
Virtual machine name	we-test-vm2
Region	(US) East US
Availability options	No infrastructure redundancy required
Authentication type	Password
Username	charon
Public inbound ports	SSH

Create   < Previous   Next >   Download a template for automation

If key-pair was newly created, download private key:

If you chose to let Azure create a new SSH key-pair, you will be asked to download the private key after clicking on the Create **button**, this step is very important as this is the only opportunity to download the private key, which is required to access your VM. The image below shows a sample of this prompt:
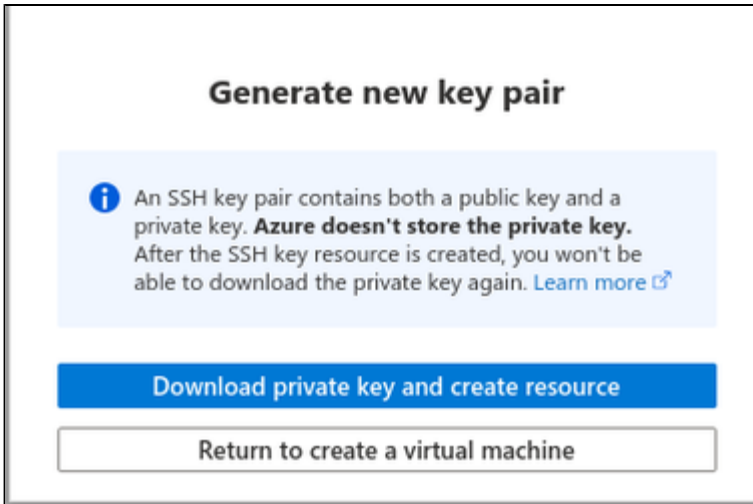
The Deployment page:

Create will take you to the **Deployment** page (possibly after downloading the private SSH key) where the current status of the deployment is displayed. Once the VM has been fully deployed, the **Deployment Complete** screen will be displayed.

Sample **Deployment Complete** screen:
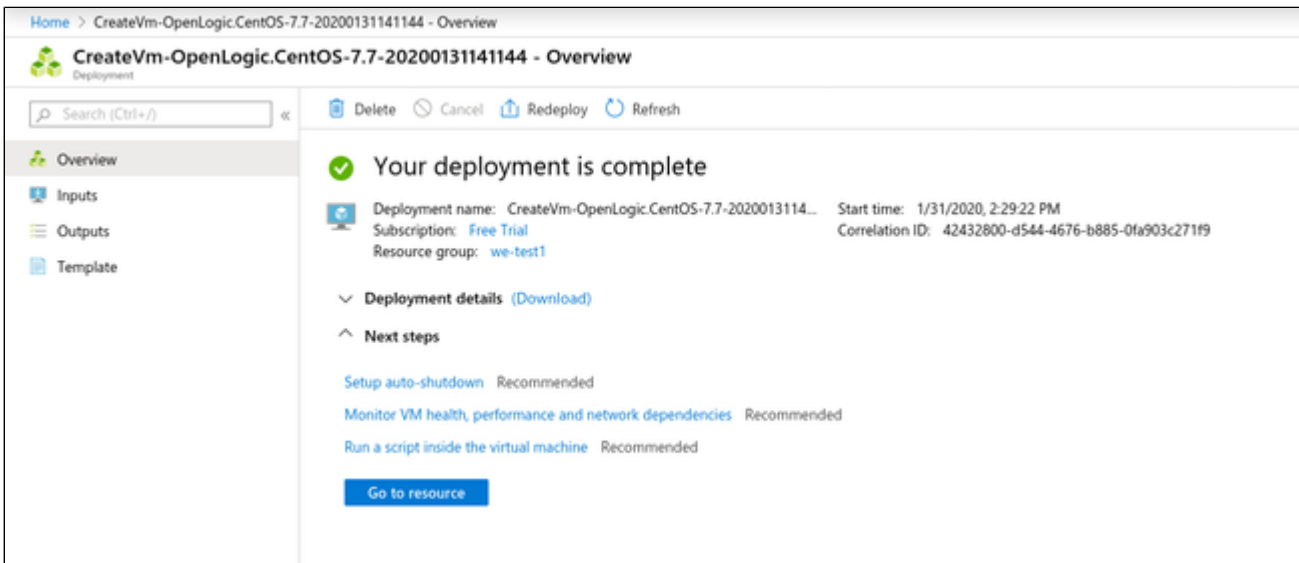
Click on **Go to resource** to get to the details page of the newly created VM. The image below shows a sample of a detail page:
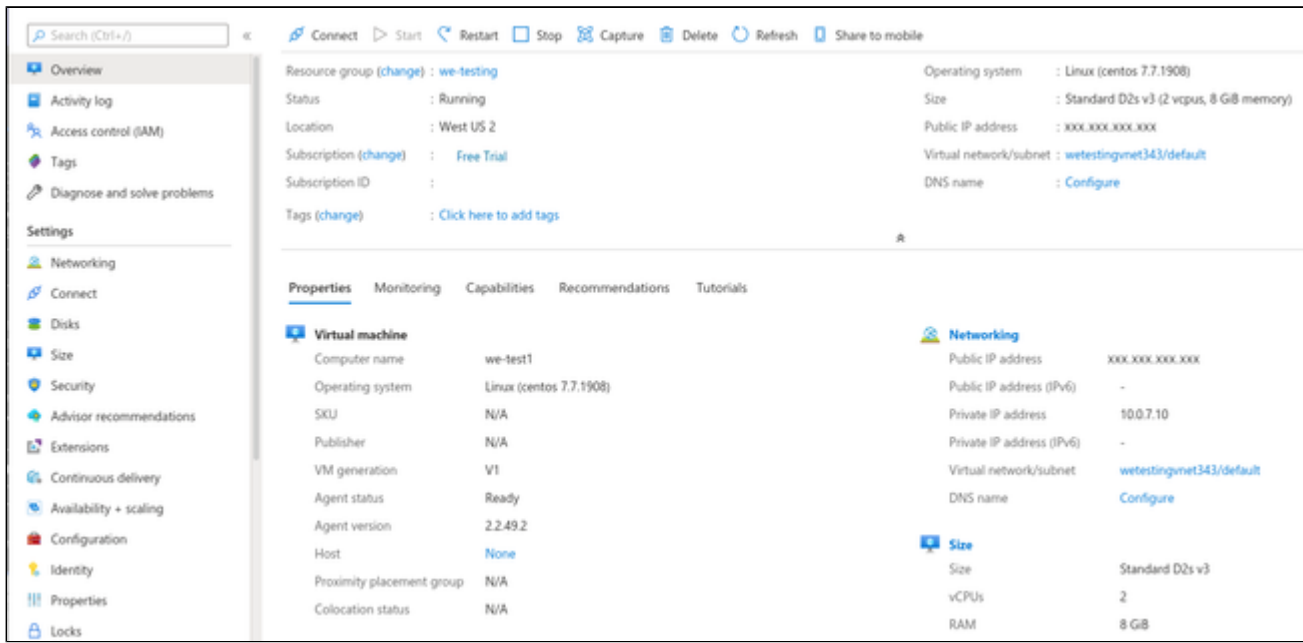
Installing the VE License Server Software

Charon emulators that are to use Virtual Environments (VE) licenses require at least one VE license server on the Charon host system itself or an a separate license server. It is recommended to run the VE license server on a **dedicated system** to avoid license invalidation caused by changes to the system which are more likely to occur on a system used for other purposes as well, for example, to run a Charon emulator. It is also recommended to install a backup license server to ensure continued operation in case of a failure or invalidation of the primary license.

Please note:

- This section provides a basic description on how to install the VE license server kit. Please consult the *VE License Server user's guide* in the Licensing Documentation section of the Stromasys documentation site for a detailed description of the prerequisites for running the VE license server and the configuration options available.
- If an instance was installed from a prepackaged Charon emulator marketplace image, the installation package is already stored in */charon/storage*. Please check, if there are newer versions available that would be preferable for your environment.
- For conventional installations, Stromasys will provide you with a download location.

In the description below, the placeholders used have the following meaning:

- *<mykey>* is the private key of the key-pair you associated with your cloud instance
  (for an on-premises VMware installation or an installation on a physical system where logging in with username/password is allowed, this is not needed).
- *<user>* is the user associated with your license server instance (e.g., *opc* on OCI, *centos* for a CentOS instance on AWS, or the custom user on your VMware virtual machine;
  for an instance installed from a Stromasys-provided Charon AL or VE emulator marketplace image, use user *charon* for SFTP and user *sshuser* for interactive login.
- *<linux-ip>* is the ip address of your license server system.

Perform the following steps to install the VE License Server software:

1. **Copy the license server software package to the license server host (if needed)**:
   a. For example, use **sftp** to connect to the VE license server system.
      # sftp -i ~/.ssh/<mykey> *<user>*@*<linux-ip>*
   b. Copy the software package to the license server system using the following SFTP command:
      > put <local-path-to-license-server-package>

2. **Use ssh to log in on the license server host**.
   # ssh-i ~/.ssh/<mykey> *<user>*@*<linux-ip>*

3. **As a privileged user (root) go to the directory where you stored the installation package and install the package**:
   a. Become the root user: **# sudo -i**
   b. Go to the package location: **# cd** <path-to-package-directory>
      If you used SFTP to copy the package to an instance installed from a prepackaged Charon marketplace image, the home directory of the *charon* user and the default location for file transfers is */charon/storage.*
   c. For VE license server 2.2.4 and above, unpack the archive and agree to the end-user license agreement:
      i. **# sh ./license-server-**<version>**.rpm.sh**
         This will display the EULA. After agreeing to it, for version 2.2.4, the RPM installation package will be unpacked in the current directory. For version 2.2.5 and later, the EULA and the RPM package will be unpacked in a subdirectory (*license-server-<version>.rpm*) of the current working directory.
   d. Install the package:
      i. Go to the directory in which the RPM package is located.
      ii. Linux 7.x: **# yum install license-server*.rpm**
      iii. Linux 8.x and 9.x: **# dnf install license-server*.rpm**

Below, you find the sample output of an installation (version 8.x of the supported Linux distributions; assuming that the RPM is in the current working directory):

```
# dnf install license-server-2.0.1.rpm
Last metadata expiration check: 0:19:36 ago on Di 03 Mai 2022 13:20:02 CEST.
Dependencies resolved.
================================================================================
 Package              Architecture Version          Repository        Size
================================================================================
Installing:
 license-server        x86_64       2.0.1-1          @commandline      53 M

Transaction Summary
================================================================================
Install  1 Package

Total size: 53 M
Installed size: 85 M
Is this ok [y/N]: y
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                            1/1
  Running scriptlet: license-server-2.0.1-1.x86_64              1/1
  Installing       : license-server-2.0.1-1.x86_64             1/1
  Running scriptlet: license-server-2.0.1-1.x86_64              1/1
Created symlink /etc/systemd/system/multi-user.target.wants/licensed.service  /etc/systemd/system/licensed.service.

  Verifying        : license-server-2.0.1-1.x86_64              1/1

Installed:
  license-server-2.0.1-1.x86_64

Complete!
```

Installing the Charon-SSP Manager (Charon-SSP only)

Contents

Overview

The Charon-SSP Manager is the main interface for managing the emulated SPARC systems running on a Charon-SSP cloud host. Therefore, the Charon-SSP Manager must be installed on every system that will be used to manage the Charon instances running on the Charon-SSP cloud host. Configuring and managing Charon-SSP instances from the command-line is also possible, but outside the scope of this Getting Started Guide. Please refer to the general Charon-SSP User's Guide for information about using the command-line.

Typically, for the management of a remote Charon host, the Charon Manager is installed on a system on customer premises, and uses an encrypted connection to manage the Charon host in the cloud. The Charon Manager can also be installed on the Charon host itself and be accessed via X11-Forwarding across an SSH connection. The latter currently requires additional package installation (via standard or local repository) on the Charon host.

Stromasys provides Charon-SSP Manager installation packages for the following operating systems:

- **Linux distributions and versions**:
    - Oracle Linux, Red Hat Enterprise Linux, and CentOS: 7.x or higher (64-bit versions only). Please note that as of 1 January 2022 CentOS 8 is EOL. For new deployments, it is recommended to use a non-EOL alternative. For existing installations, the possible negative impacts of staying with an EOL host operating system should be carefully evaluated.
    - Rocky Linux version 8.x (64-bit) or higher
    - Ubuntu 17 or higher (64-bit)
- **Microsoft Windows:** versions 7, 8, 10, and (starting with Charon-SSP 5.6.1) version 11

Restriction: the Charon-SSP Manager is not supported on Linux hosts using Wayland when they run in a VMware instance with 3D-graphics. The Manager will show erratic behavior in such cases.

Installation Packages

Installation packages are available in RPM or Debian package formats for Linux and as a ZIP-file for Microsoft Windows.

Please note: starting with SSP version 5.6.1, the RPM packages are distributed in an self-extracting archive. The archive required for the Charon-SSP Manager is **charon-gui-**<version>**.sh**. It also contains the Charon-SSP Agent which must be installed on the Charon host system to be managed by the Charon Manager. The archive must be unpacked on a Linux system (even if you need the kit for Microsoft Windows).

Use the following command to unpack the RPM packages:

- Go to the directory containing the self-extracting archive.
- Run the script: **# sh charon-gui-**<version>**.sh**
- Read the end-user agreement and accept it.
- The RPM packages will be extracted in a subdirectory (*charon-gui-<version>*) of your current working directory.

Names of the Charon-Manager installation packages:

- RPM package: **charon-manager-ssp-**<version>**.rpm**
- Ubuntu package: **charon-manager-ssp-**<version>**.deb**
- Microsoft Windows package: **charon-manager-ssp-**<version>**.zip**

<u>There are different ways to obtain the Charon-SSP Manager installation packages. They are briefly described below:</u>

a) For installation on a management system on customer premises if using a prepackaged cloud marketplace image:

The packages are included in the Charon-SSP cloud-specific image (in */charon/storage*). Once a new instance has been launched, you can download the Charon-SSP Manager archive from the running instance:

- Connect to the public IP address of the instance via SFTP using the private key assigned during launch and the user **charon**:
  $ sftp -i *<path-to-private-key>* **charon@***<public-ip-of-cloud-instance>*
- Download the required package:
  sftp> get charon-gui-<*version*>**.sh**

b) For installation on a Charon host where a conventional RPM installation was performed: Stromasys will provide you with a download link. The Charon Manager packages are also included in the Charon agent RPM and available in */opt/charon-agent/ssp-agent/bin/* once the agent has been installed.

Charon-Manager Installation on Linux

Prerequisites

The Charon Manager can be installed on the Charon host itself or on a remote management system. For the Charon Manager to work, the **Charon Agent must have been installed on the Charon host system**. The Charon Manager communicates with the Agent to configure and manage the emulator instances.

When the Charon Manager is installed on a Linux host with a graphical user environment, the prerequisites are often already fulfilled. However, when installing the Charon Manager on the Charon-SSP host in the cloud or on a Linux server without graphics (for example, to display it via a remote X11-connection) instead of on a local management system, **additional packages** may have to be installed that normally are already available in a workstation environment.

In particular, the Charon-SSP Manager requires the following packages:

- libX11
- xorg-x11-server-utils
- gtk2
- xorg-x11-xauth (only required for X11-Forwarding)

If you install the Charon Manager with the **yum** or **dnf** command, these packages (with the exception of xorg-x11-xauth) and any dependencies that these packages themselves may have, are resolved automatically if a package repository is available. The xorg-x11-xauth package must be installed separately (also with yum). If your server does not have access to the standard operating system repositories, refer to this document for instructions on setting up a local repositories.

Please note:

- The exact list of additionally required packages depends on what is already installed on the server.
- To install dependencies on Ubuntu, please refer to your Linux documentation.

Installation Steps on Linux

The following table describes the installation steps for Charon-SSP Manager:

Step	Description	
1	Installation on a Linux management system on customer premises (typical installation): • Log in to the Linux management system as the **root** user (denoted by the **#** prompt). • Copy the installation package to your local Linux management system (from one of the sources described above).	Installation on the Charon-SSP host system in the cloud (non-typical installation): • Log in and become the root user on the Charon host using the following commands: **$ ssh -i** *<path-to-private-key>* **sshuser@***<cloud-instance-ip>* **# sudo -i** • **Please note:** if the Charon host was not installed using a prepackaged marketplace image, the username may be different and the installation package will have to be copied to the Charon host in a separate step.
2	Go to the directory where the package has been stored: **# cd** *<package-location>*	
3	**Unpacking the shell archive**:	
	• Run the script: **# sh charon-gui-***<version>***.sh** • Read the end-user agreement and accept it. • The RPM packages will be extracted in a subdirectory (*charon-gui-<version>*) of your current working directory	
4	**Installing the package:**	
	Assuming you are in the subdirectory containing the RPM file, use the following commands for supported Linux systems with RPM package management: • Linux 7.x: **# yum install** *<filename-of-package>* • Linux 8.x and higher: **# dnf install***<filename-of-package>* (For an installation on the cloud host system, check if xorg-x11-xauth is already installed if X11-Forwarding is planned.)	
	For systems with Debian package management (Ubuntu): **# dpkg -i** *<filename-of-package>*	

Installation Steps on Microsoft Windows

The Charon-SSP Manager for Windows software is shipped as a zipped archive package which is contained in the **charon-gui-*<version>*.sh** archive. After unpacking the archive on a Linux system, copy the ZIP file to your Microsoft Windows system and use the following instructions to complete the installation.

1. **Right-click** on the zip archive charon-manager-ssp-{version}.zip and select **Extract All**.
2. A window titled **Extract Compressed (Zipped) Folders** opens. In this window:
   a. Click on the **Show extracted files when complete** checkbox.
   b. Click on the **Extract** button.
3. A new Windows Explorer window opens showing the extracted packages.
4. **Double-click** on the **setup.exe** executable to begin the installation.
5. If you are presented with an **Open File - Security Warning** window, click on the **Run** button.
6. You should now see the Charon-SSP Manager Setup Wizard. To proceed with the installation, click on the **Next** button. If the Windows Installer reports that Charon-SSP Manager for Windows is already installed, you must deinstall the currently installed software before you can install a different version. Normally, several versions can coexist.
7. To accept the default installation options, simply click on **Next** without modifying any options. Alternatively, the following installation options can be adjusted:
   a. Click on **Browse** to select an alternative installation target.
   b. Click the appropriate radio button, **Everyone** or **Just for Me**, to specify system-wide or private installation respectively (the system-wide installation will prompt for the administrator password if you are not using the administrator account).
   c. To determine the approximate disk usage after the installation, click on the **Disk Cost** button.
   d. Once all options have been set, click on **Next**.
8. Proceed with the installation by clicking on **Next**.
9. Once the installation has completed, click on **Close** to exit the SSP-Manager Setup Wizard.
10. The installation process creates:
    a. A Charon Manager icon on the desktop
    b. A Charon Manager entry in the Start menu (folder Stromasys)

Accessing the Charon Cloud Instance

Firewall Considerations

Access to a cloud cloud instance may be controlled by

- an external firewall,
- the operating system firewall of the instance,
- cloud-specific firewall and security rules and groups.

In addition to allowing SSH access, the different firewall levels must be configured to permit at least TCP port 8080 to enable license server access.

Azure Network Security Groups

Network Security Groups (NSG) can be associated to interfaces or subnets. Security rules in network security groups enable you to filter the type of network traffic that can flow in and out of virtual network subnets and network interfaces. When a cloud instance is created, you can assign a default security group to its interface (allowing SSH). Please refer to the following tutorial for more information: https://docs.microsoft.com/en-us/azure/virtual-network/tutorial-filter-network-traffic.

Connecting to the Cloud Instance

During the configuration of your instance you should have created security rules allowing at the minimum SSH access to the instance. If this has been done correctly, you can, for example, use SSH from the command-line or from a tool such as PuTTY to access the command-line of the user **sshuser** (for prepackaged Charon emulator marketplace images) or your custom user (for RPM installations) on the Charon host instance.

You will need the following:

- Access to the private key associated with the public key you uploaded during the configuration of the instance.
- The public IP address of the instance.

Please note:

- The file permissions of the private key file must be set such that the file is only readable by the user (e.g., `# chmod 400 <private-key-file>`).
- PuTTY uses a different key file format. It comes with tools to convert between its own `.ppk` format and the format of OpenSSH used by the default Linux tools.

There are several ways to connect to your Charon cloud instance using this basic SSH protocol access. Some of them are described in the following sections below.

- SSH and SFTP Command-Line Access
- Connecting with the Charon-SSP Manager

SSH and SFTP Command-Line Access

Initial Access to the Instance

Once you have access to the instance, you can create the access you require for your applications. This section just shows the basic steps for initial access to the instance.

SSH Interactive Access

To connect to the instance interactively, you must connect as the management user of your instance. Use the following command:

```
$ ssh -o ServerAliveInterval=30 -i <path-to-your-private-key> <management-user-name>@<cloudhost-IP-address>
```

The parameter `ServerAliveInterval` will protect the connection from timing out.

Please note:

- Depending on the type of connection, you will have to use either the public IP address of the cloud system or its address in a customer-specific VPN.
- The **private key** used must correspond to the public key installed in the *authorized_keys* file of the cloud instance management user. This is usually done during initial cloud instance launch.
- The management user account normally allows sudo access to privileged commands (use **sudo -i**).
- If the instance was created using a Stromasys-provided AL or VE marketplace image, the management user for **interactive login** is the user **sshuser**.

File Transfer with SFTP

SFTP enables file transfers to and from the cloud instance. Use the management user of your instance. The security rules must allow SSH access to allow SFTP access to the cloud instance.

Please note: Depending on the type of connection, you will have to use either the public IP address of the cloud system or its address in a customer-specific VPN.

To connect to the instance, use the following command:

```
$ sftp -i <path-to-your-private-key> <management-user-name>@<cloudhost-IP-address>
```

Please note:

- Depending on the type of connection, you will have to use either the public IP address of the cloud system or its address in a customer-specific VPN.
- The **private key** used must correspond to the public key installed in the *authorized_keys* file of the cloud instance management user. This is usually done during initial cloud instance launch.
- If the instance was created using a Stromasys-provided AL or VE marketplace image, the management user for **file transfer** is the user **charon**.
- If the user **charon** is used to transfer files, the home directory for the file transfer will be */charon/storage*.

Connecting with the Charon-SSP Manager

Contents

General Information

To use the management GUI for Charon-SSP and the emulated SPARC systems, you must connect to the Charon-SSP cloud instance with the Charon-SSP Manager. The Charon-SSP Manager is the main interface to all important functions of the Charon-SSP software. Managing Charon-SSP via the command-line is possible but outside the scope of this document (please refer to the user's guide of the conventional product for more information).

Notes:

- Typically, **Charon-SSP Manager** is installed either on the Charon host itself (if this system has a graphical interface) on a management system on customer premises. **This is the use-case described in this section**. Other configurations are possible. For example, the Charon Manager could be installed on a non-graphical Charon host in the cloud or in a VMware environment and be displayed on a remote system using X11-Forwarding via an SSH connection.
- **For accessing a Charon host instance in a cloud across the Internet using its public IP address**:
  - The **security configuration** on your Charon host instance must at least allow SSH access. This allows the **built-in SSH tunneling** of the Charon-SSP Manger to work. Should you not use SSH tunneling, you must open up additional ports. However, if the connection runs over the Internet without a general VPN, Stromasys strongly recommends to use SSH tunneling to protect your Charon-SSP cloud instance and any emulated systems running on it.
  - You must know the public IP address of the Charon-SSP host instance in the cloud. To determine this address, refer to the instance information displayed on the cloud management console.
  - To use the Charon Manager integrated SSH tunnel, you need the private SSH key of the key-pair associated with your instance.
- **For access a Charon host instance in a cloud via an SSH-based VPN or another VPN solution**:
  - Active SSH-based VPN (see *SSH VPN - Connecting Charon Host and Guest to Customer Network* in the Charon-SSP User's Guide) or other active VPN solution
  - Private IP address of the Charon-SSP host in the VPN

Information about the initial management password configuration:
Before connecting to a Charon-SSP host with the Charon Manager for the first time after the initial installation you must set the management password. This can either be done via the command line (see *SSH Command-Line Access*) or via the Charon Manager itself as described below.

Starting the Charon Manager and Login to Charon Host

Starting the Charon Manager

To start the Charon-SSP Manager on Linux and to open the Charon Manager login window, use the following command:

```
$ /opt/charon-manager/ssp-manager/ssp-manager
```

To start the Charon-SSP Manager on Microsoft Windows, click on the Desktop icon or use the entry in the Start menu.

The steps above will open the Charon Manager login window which has **two tabs**.

Entering Charon Manager Login Information and Connecting to Charon Host

Step 1: the Charon Manager **Login** tab

If the management password has not yet been set, perform the following steps:

- Enter the IP address of your Charon-SSP host instance in the **IP address** field.
- Leave the **Password** field empty.
- For cloud instances enable the SSH tunnel configuration (select **ON**).
  Set to **OFF** if connected to *localhost*. The SSH tunnel can generally be used if key-based SSH login is enabled on the target system.
- Change to the SSH tab to fill in the required information if the SSH tunnel has been enabled.

If the management password has already been set, perform the following steps:

- Enter the IP address of your Charon-SSP instance in the **IP address** field.
- Enter the Charon-SSP management password.
- Enable the SSH tunnel configuration for communication across a public network unless you use a secure VPN connection (key-based SSH login required).
- If the SSH tunnel is enabled, change to the SSH tab to fill in the required information there.

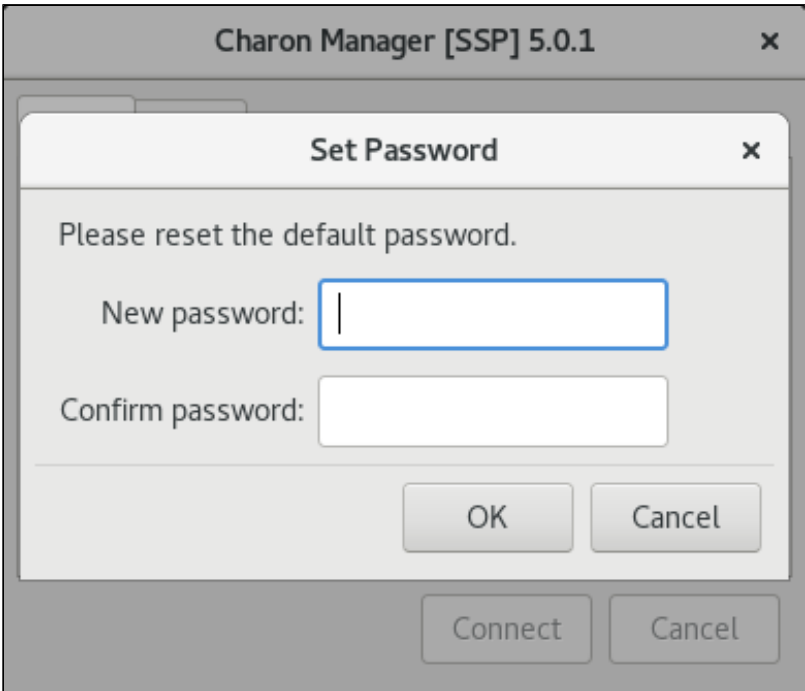**Step 2**: the Charon Manager **SSH** tab

If you use the integrated SSH tunnel, perform the following steps:

- Enter the Charon-SSP user in the **Username** field. For prepackaged images, use **charon** or **sshuser**; for RPM installations use the user for whom the correct public key has been installed.
- Enter the path to the private key file (click on the three dots next to the **Private key** field to open a file browser). You typically associated your cloud instance with this key-pair during instance creation.
- Enter the passphrase for the private key if required.
- Adjust the server port (default 22) if required.

Please note: the public key of the key-pair must be in the `.ssh/authorized_keys` file of the user entered above (**sshuser** and **charon** for prepackaged images).
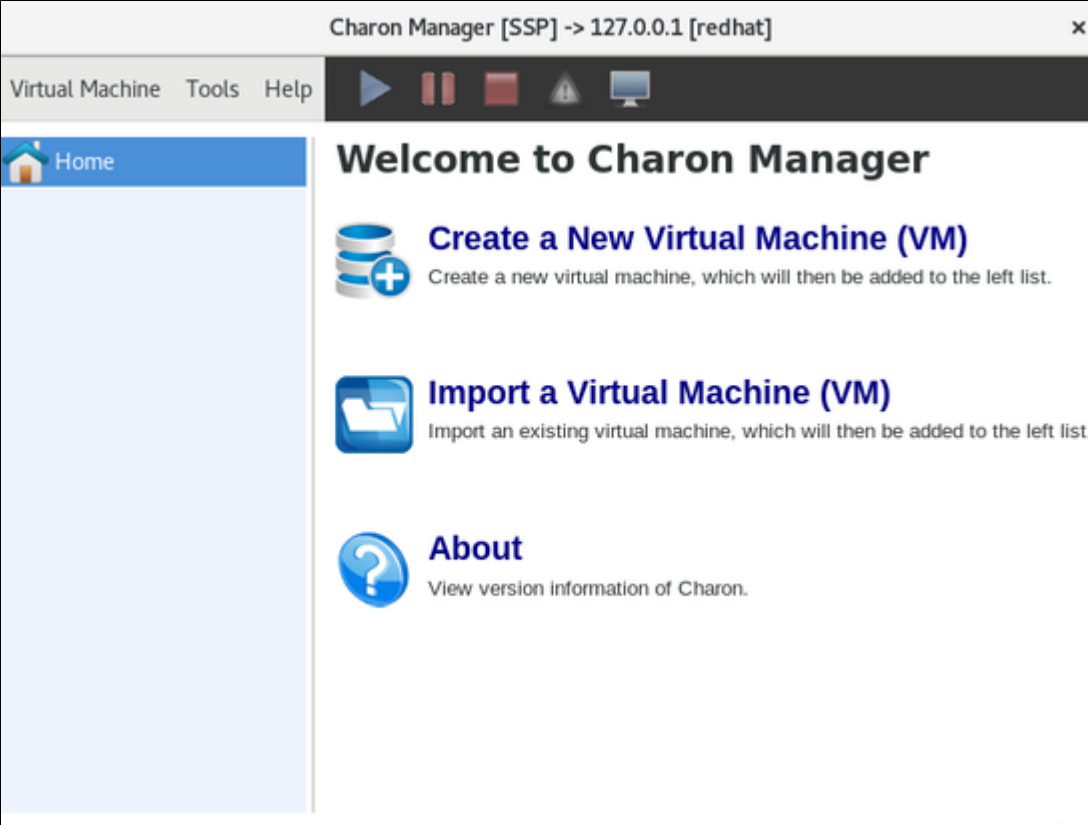
Step 3: connecting to the Charon host system

After entering all the required information, click on **Connect** to connect to the Charon-SSP instance. **If the management password still needs to be set,** you will receive a prompt to enter the new password:

<table>
<tr><td>

Charon Manager [SSP] 5.0.1 ✕

Set Password ✕

Please reset the default password.

New password: [ ]

Confirm password: [ ]

OK   Cancel

Connect   Cancel

</td><td>

- Enter the desired password in the **New password** field and confirm it in the **Confirm password** field. This management password is then valid for all subsequent logins by the same or a different user until it is changed again. It is not removed if Charon-SSP is reinstalled. Note that older versions of the product will not prompt for the password at first login but will use a default password (**stromasys**). If you need to reset a forgotten management password, please refer to the Charon-SSP user's guide.
- Then click on **OK**.
- The login process continues.

</td></tr>
</table>

After a connection has been successfully created, the Charon Manager welcome screen opens. Example of the Charon Manager welcome page:

Charon Manager [SSP] -> 127.0.0.1 [redhat] ✕

Virtual Machine   Tools   Help   ▶ ⏸ ⏹ ⚠ 🖥

🏠 Home

Welcome to Charon Manager

Create a New Virtual Machine (VM)
Create a new virtual machine, which will then be added to the left list.

Import a Virtual Machine (VM)
Import an existing virtual machine, which will then be added to the left list.

About
View version information of Charon.

Please note: the **title bar** of this screen indicates the managed system type in square brackets (conventional Red Hat installation in the example).

Removing Charon from the Azure Environment

Please note: the steps described here assume that the cloud instance and its resources are dedicated to the Charon installation that is to be removed and that they are not used for any other important applications.

If you want to **completely remove** Charon from the cloud environment and free up all resources used by it, you must perform several steps:

1. Cleanly shutdown any running guest operating systems.
2. Stop the emulator.
3. Back up any emulator data (configuration, vdisks, vtapes, ISO files, and any other customized emulator and instance settings) that you want to keep and copy the data to a non-cloud resource.
4. Delete all resources associated with the cloud instance that was used for the emulator. This includes the cloud instance itself, any additional storage associated with it, any static IP addresses allocated to it, associated vNICs, etc.

If more than one cloud instance is dedicated to the Charon installation, the steps must be performed for all cloud instances that are no longer needed.

Additional Azure Instance Configuration for Charon

This section describes some additional Azure configuration options that can be used with the Charon Azure instance.

Contents

- Storage Management
- Charon Cloud Networking Information

Storage Management

To add additional disk storage to your Charon Azure instance (for example, for storing virtual disk containers), perform the steps described below.
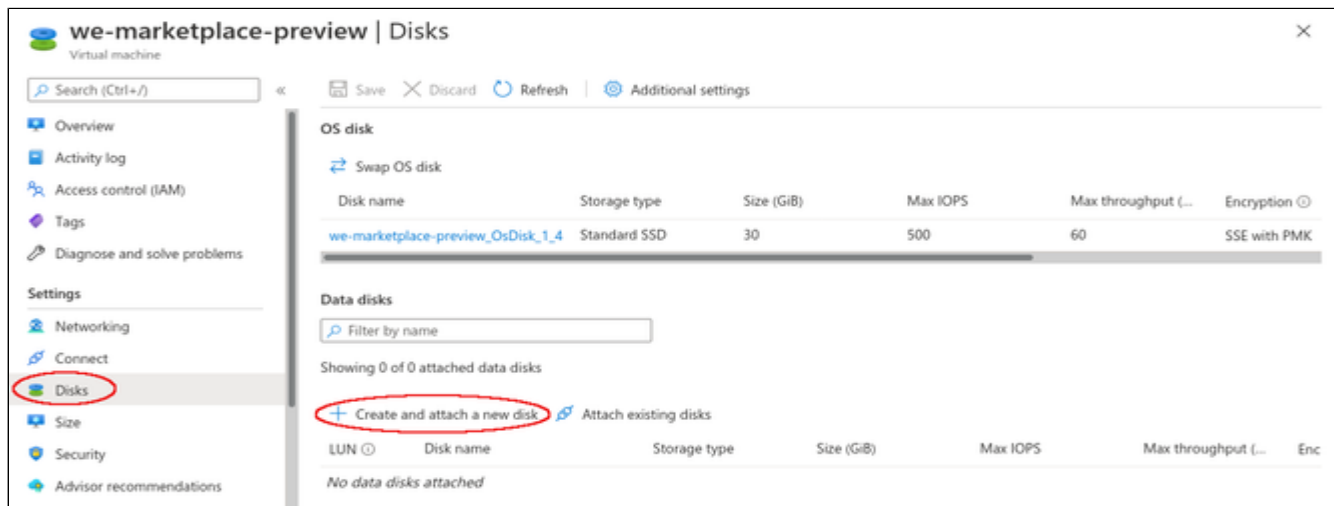
Contents

Steps in the Azure Management Environment

Step 1: To start the process of adding a disk, open the disk section of your VM (from the VM details section) by clicking on **Disks** in the left-hand pane.

The following image shows a sample.

If no disk has been added before, only the system disk is displayed on this screen. The temporary storage disk is not shown.

To add a new disk, click on **Create and attach new disk**. This will display the disk configuration area. If you select to attach an existing disk, you will be offered the options to select it.

Step 2: Create a new disk.

The following sample shows the configuration options of a new disk:

Step 3: Enter the disk configuration data.

Add at least the following information in the disk creation area:

- Unique LUN
- Disk name
- Disk size and type
- Disk performance data
- Encryption setting

Once you have filled in all the necessary data, click on **Save** at the top of the screen to create the disk and attach it to your VM.

Step 4: Review the new disk configuration.

On the disk overview screen, you will see the selected (or newly created) disk. It is now attached to your VM.
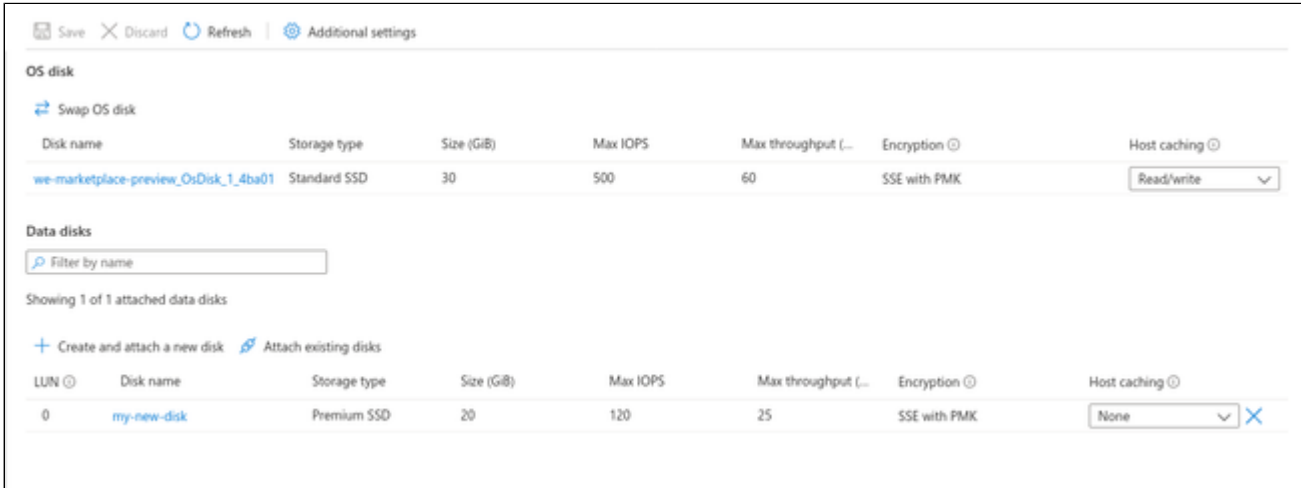
Steps on the Charon Host System

Mounting a Newly Attached Volume Using the Storage Manager (SSP AL only)

After the volume has been attached to the instance, it must be included in the Charon-SSP host system configuration. On Charon-SSP AL instances, this is achieved via the Charon-SSP Manager.

1. Open the Charon-SSP Manager on your local system and connect to your Charon cloud instance.
2. Select **Tools > Azure Cloud > Storage Manager**.
3. In the **Storage Manager** window, perform the following steps:
   a. Select the new device.
   b. Click on the cog-wheel symbol.
   c. **Only if required**, select **Format Volume** to create a filesystem on the new device.
      Please note: this will delete all data on the volume.
   d. Click on the cog-wheel symbol and select **Mount the Filesystem**.

This will mount the new volume under `/charon/storage/media/<UUID>/`. The following image shows a sample storage manager window:
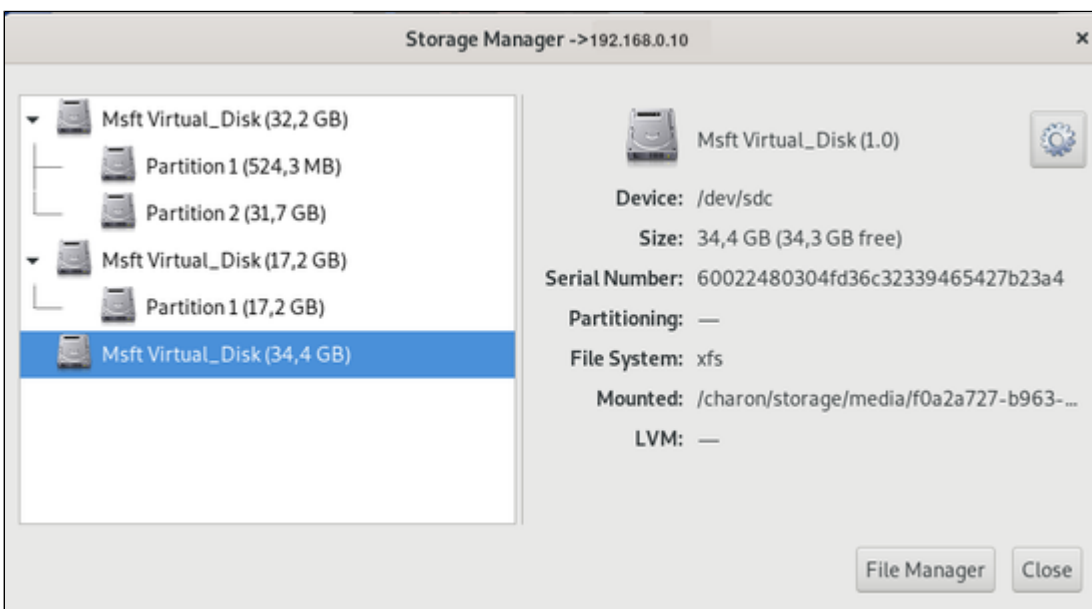
Once the filesystem has been mounted, the space is available to the Charon-SSP host system. After the first mount via the Storage Manager, the filesystem will be automatically mounted after a restart of the Charon host instance.

Mounting a Newly Attached Volume Manually

This is an example of how to mount (and if necessary partition) an additional disk on a Charon host system. Please refer to the Linux manual pages for details.

The general tasks on the Charon host system require to identify the disk, add a file system to it (if this has not been done before), and mount the disk on a suitable mount-point.

Please note: the different cloud environments may offer the disk volumes to the Linux instance using different names. However, the basic steps will be the same as in the examples below.

Step 1: Identify new disk

After logging in on the system, you can identify the new disk using the **lsblk** command:

```
# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   20G  0 disk
sda1    8:1    0  200M  0 part /boot/efi
sda2    8:2    0 19,8G  0 part /
sdb      8:16   0  200G  0 disk
```

In the example above, the new disk is **/dev/sdb**. The output shows no mount-point, i.e., the disk is not mounted yet. It also does not have any partitions.

Please note:

- A disk without partitions can also have a filesystem and data on it. Hence be sure that the disk really does not have any important data on it before you partition it.
- If a system has many disks, it is helpful to run the **lsblk** command before the new disk is added. This makes it easy to identify the new disk in the output after it has been added.

Step 2: Partition disk (fdisk or parted) - **only if required**

Please note: This step is only meant for new disks or to re-partition an existing disk. **It will destroy all data on an existing disk.**

Please refer to the manual pages (**$ man parted** and **$ man fdisk**) of your Linux distribution for details on the disk-partitioning commands. If the whole disk is used for one filesystem, it is not strictly required to create a partition. The decision of which disk layout is required depends on the customer requirements is the responsibility of the user.

After creating one partition on disk with fdisk (**# fdisk /dev/sdb**), the **lsblk** output shows the new partition:

```
# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   30G  0 disk
sda1    8:1    0  500M  0 part /boot
sda2    8:2    0 29,5G  0 part /
sdb      8:32   0   64G  0 disk
sdb1    8:33   0   64G  0 part
```

Step3: Create a filesystem on the new partition(s)

Use the **mkfs** command to create a new filesystem. The selection of a filesystem depends on customer requirements. For example, to create an XFS filesystem, use

```
# mkfs.xfs /dev/sdb1
```

Please refer to the documentation of your Linux distribution for details about the **mkfs** command.

Step 4: Create a mount-point and mount the new filesystem

The following example shows how to create a mount-point and mount the file system.
To keep the example consistent with the sample outputs above, **/dev/sdb1** is used in this example. However, as the */dev/sdX* device names are not guaranteed to be persistent across reboots, it is strongly recommended to use names from the /dev/disk/by-* hierarchy (for example by-uuid) for permanent, production use.

```
# mkdir /space
# mount /dev/sdb1 /space
```

The **df** command shows the mounted filesystem:

```
# df
Filesystem      1K-blocks      Used Available Use% Mounted on
devtmpfs          4065684         0   4065684   0% /dev
tmpfs             4077556        16   4077540   1% /dev/shm
tmpfs             4077556      9224   4068332   1% /run
tmpfs             4077556         0   4077556   0% /sys/fs/cgroup
/dev/sda2        30929148   1677416  29251732   6% /
/dev/sda1          508580     65512    443068  13% /boot
tmpfs              815512         0    815512   0% /run/user/1000
/dev/sdb1        65923628     53272  62498580   1% /space
```

Step 5: Mount the disk automatically at system boot

To mount the disk automatically when the system boots, you must add it to the file **/etc/fstab**.

Please note: The device naming /dev/sdXN (e.g., /dev/sdb1) is not guaranteed to be persistent across reboots. Hence, it is advisable to use a persistent name from the **/dev/disk/by*** hierarchy (for example, the UUID).

You can use the **ls** or the **blkid** command to identify the UUID. Examples:

```
$ ls -l /dev/disk/by-uuid/
total 0
lrwxrwxrwx. 1 root root 10 2020-08-14 21:14 0c523909-fb78-48cb-9dc8-e7a08197a673 -> ../../dm-4
lrwxrwxrwx. 1 root root 10 2020-08-14 21:14 31fa8e8c-a6c0-45f7-9892-da13ba81e0e5 -> ../../sdb1

$ blkid |grep sdb1
/dev/sdb1: UUID="31fa8e8c-a6c0-45f7-9892-da13ba81e0e5" BLOCK_SIZE="4096" TYPE="xfs" PARTUUID="db62deaa-f25f-43d4-b958-
700c1c13d844"
```

To add the device to **/etc/fstab** perform the following steps:

1. As the root user, open the file **/etc/fstab** with a text editor.
2. Add the mount command to the file. **Please note**: The following is for illustration only. The exact options depend on your requirements.
   Sample fstab entry:
   UUID=*31fa8e8c-a6c0-45f7-9892-da13ba81e0e5* /space **xfs defaults 1 2**
3. Save the file.
4. Test if the automatic mount works correctly.

Unmounting a Volume

To **unmount** a volume before perform the following steps:

- Stop all Charon instances that might use the volume that is about to be unmounted.
- On host systems based on AL images:

  - In Charon Manager go to **Tools > Azure Cloud > Storage Manager**.
  - Select the volume.
  - Click on the cogwheel symbol and select **Unmount the Filesystem**.
- On other systems:

  - Use the command **# umount** *<device-path>* or **# umount** *<mount-point>*
  - To make this permanent, remove the corresponding entry in **/etc/fstab**.

Please note: When you delete a virtual machine in Azure, by default, the disks that are attached to the VM will not be deleted. If you do not require a disk anymore, delete it to avoid unnecessary cost.

Charon Cloud Networking Information

Contents

General Information

This section provides some basic information about networking questions that are likely to affect Charon when running in the cloud.

If the information in this chapter is not sufficient, please refer to the additional Charon documentation provided by Stromasys and the documentation provided by your cloud provider for up-to-date and comprehensive information.

Linux Versions and NetworkManager

There are significant differences regarding the NetworkManager in the different Linux versions (RHEL 7, 8, 9 and derivatives). This section provides an overview of some important differences.

There are **two basic network configuration systems** in the relevant Linux systems:

- The network service with the network configuration based on **ifcfg-files** in */etc/sysconfig/network-scripts*. This requires the **network-scripts** package.
- The **NetworkManager** with its own configuration file syntax. Persistent configuration files are stored in */etc/NetworkManager/system-connections*. The NetworkManager has a plugin (ifcfg-rh) to handle ifcfg-files. This plugin does not support all configuration options of the network-scripts system (e.g., tunnel and tap interfaces are not supported).

Linux 7.x:

- The **network-scripts** and **NetworkManager** methods coexist. It is possible to disable the NetworkManager completely or only for certain interfaces (parameter **NM_CONTROLLED=no** in the ifcfg-file).
- The default NetworkManager plugin is the ifcfg-rh plugin.
- Interfaces managed by the Charon-SSP Manager must have an ifcfg-file and be removed from NetworkManager control (unmanaged interfaces).

Linux 8.x:

- The **network-scripts** package is deprecated. It is not installed by default, but available in the Linux package repositories.
- The default NetworkManager plugin configuration is **ifcfg-rh, keyfile**. The *keyfile* plugin is responsible for handling the native NetworkManager configuration file syntax.
- If virtual bridge configurations including TAP interfaces are configured using **ifcfg-files**, the **network-scripts** package is required. Otherwise, the TAP interfaces cannot be activated (missing support in the ifcfg-rh plugin). Alternatively, such interfaces can be configured as native NetworkManager connections.
- There is an **ifup** command which by default points to nm-ifup. Once the network-scripts package is installed, it points to the ifup command contained in this package.
- The loopback interface (lo) cannot be managed by the NetworkManager.

- Interfaces managed by the Charon-SSP Manager must be under NetworkManager control (managed interfaces).

Linux 9.x:

- The **network-scripts** package is no longer available in the Linux package repositories.
- The default Networkmanager plugin configuration is **keyfile, ifcfg-rh**.
- Existing ifcfg-files can still be read and written, but only if supported by the ifcfg-rh plugin.
- A new **nmcli** command option (`nmcli connection migrate <con-name>`) helps with the conversion of ifcfg configuration files to native NetworkManager connection profiles. However, this command only works for connections supported by the ifcfg-rh plugin. This means, for example, that TAP interfaces that were previously configured via ifcfg-files must now be recreated using nmcli commands or another NetworkManager configuration tool. Before using the migration command, take a **backup copy** of the content of /etc/sysconfig/network-scripts.
- By default, there is no ifup command. If it is needed, the NetworkManager variant of the command can be installed (**NetworkManager-initscripts-updown**).
- The loopback interface (lo) cannot be managed by the NetworkManager in versions before 9.2.
- Interfaces managed by the Charon-SSP Manager must be under NetworkManager control (managed interfaces).

Additional information about the ifcfg-rh plugin:

The **ifcfg-rh plugin** is used by the NetworkManager to read/write the traditional **ifcfg-files** in */etc/sysconfig/network-scripts*. Each NetworkManager connection corresponds to one ifcfg-file. The plugin does not support all the connection types supported by the original **network-scripts** package. The plugin currently supports Ethernet, Wi-Fi, InfiniBand, VLAN, Bond, Bridge, and Team connections. This means that, for example, TYPE=Tap is not supported and cannot be handled by the NetworkManager in the ifcfg-file format. In Linux 7.x and Linux 8.x, the network-scripts package can be used to support the ifcfg-file format. In Linux 9.x, this package is no longer available. Thus, unsupported connection types must be manually recreated.

Interface MTU Considerations

When configuring a dedicated network interface for an emulator, **ensure that the MTU of the Charon host interface used is not smaller than the MTU used by the legacy guest operating system**. Failing to do so will cause network problems. For further information, please refer to the chapter *Interface MTU Considerations* in this guide.

Host to Guest Communication Considerations

There are several ways a communication between the host operating system and the legacy guest operating system can be implemented. For example:

1. **Internal virtual bridge on the host system**:
   Such a bridge has several TAP interfaces. The host and the guest systems are connected to this bridge and can communicate directly with one another using L3 and L2 protocols. The bridge uses its own IP subnet that can be defined by the user. For Charon-SSP, setting up such a configuration is supported by the Charon-SSP Manager (leave the default gateway field empty for the bridge interface). Several hosts configured with guest systems and such an internal bridge can communicate across the cloud-internal LAN and the host systems can route the private IP subnets of the bridges between themselves. L2 protocols are not possible if routing across the cloud LAN is used.

2. **Communication via the cloud-internal subnet LAN**:
   In this case, a second interface is added to the Charon host system. The second interface is then assigned to the emulated guest system. After configuring the interface correctly, the host and guest can communicate across the cloud-internal LAN using IP. **Please note**: L2 protocols or any protocols that require changing the MAC address to something different than the MAC address assigned to the second interface by the cloud provider will not work.
   To connect the guest system to the LAN, the following basic configuration steps must be performed:

   - Add the additional interface to the Charon host system.
   - Make a note of the private IP address assigned to the second interface by the cloud provider, and remove it from the Linux configuration (if it has been configured).
   - Create a configuration for the additional interface such that it is activated at boot but does not have an IP address on the Linux level. This can be done via configuration files on Linux 7.x. For Linux 8.x **nmcli** commands, the **nmtui** utility, or (for SSP) the Charon Manager can be used.
   - Assign the interface to the emulated system. This can be done by modifying the emulator configuration file or by using the Charon-SSP Manager.
   - Set the MAC address of the emulated system to the same value as the one used on the host system Ethernet interface. For Charon-SSP, this configuration can be implemented using the Charon Manager.
   - On the guest operating system, configure the private IP address that was previously assigned to the second interface on Linux and configure the appropriate default route for the LAN.

Please note:

- The section *Dedicated NIC for Guest System* provides some hints on how to configure the second interface in the different situations. Please refer to your cloud-provider's documentation for up-to-date comprehensive information.
- If Layer 2 communication between guests on different Charon hosts is required, a bridged overlay solution must be set up between the two Charon host systems. Direct L2 or non-IP L3 communication across the cloud LAN is not possible.

External Communication Considerations

In addition to allowing SSH access to the host system for management purposes, it may be necessary to enable Internet communication to the host and guest system or connect host and guest to the customer's network.

Please note: Charon hosts based on Charon AL (Automatic Licensing) marketplace images and using the public license servers always need either direct Internet access or Internet access via NAT from a NAT gateway in the same cloud as the Charon host to access the license server.

Recommended way to connect the Charon host and legacy guest systems (e.g., Solaris, HP-UX) to the customer network:

To ensure data traffic between the Charon host and guest systems and the customer network is encrypted, it is strongly recommended to use a VPN connection. An example of a simple VPN connection based on an SSH tunnel is described in *SSH VPN - Connecting Charon Host and Guest to Customer Network*. This connection is based on a bridge between Charon host and guest system and (via an encrypted SSH tunnel) the remote end-point in the customer network. The connection supports L3 and L2 protocols.

Cloud providers usually also provides a VPN gateway instance that can be added to the customer cloud network to connect the cloud network to the customer network (for a charge).

Recommended way to connect the guest system to the Internet:

The Internet connection can be implemented across the VPN to the customer network. In this case, the customer can allow the legacy guest operating system to access the Internet exactly following the security policies defined by the customer.

Access to the Internet from subnets or guest operating systems with only private IP addresses:

Access to the Internet for subnets with only private IP addresses is possible across a gateway instance providing VPN access to the customer network and allowing (NATted) Internet access via this path. Alternatively, a NAT gateway in the cloud can be used to map the private addresses to public addresses. The NAT gateway can be implemented on a Charon host system, a dedicated customer-operated gateway, or it can often be provided by the cloud provider for a charge.

Please note: a Charon AL host system that uses the public license servers always needs either direct Internet access or Internet access via NAT from a NAT gateway in the same cloud as the Charon host to access the public license server.

Direct guest system access to the Internet:

This not a recommended standard solution for security reasons. However, should it be required, two interfaces with public IP addresses can be assigned to the Charon host.
One of these interfaces is then dedicated to the guest system which uses the private interface address and the MAC address assigned to the Charon host by the cloud provider (see also *Dedicated NIC for Guest System*).

Guest to Guest Layer 2 Communication Considerations

Should L2 protocols be required between two guest systems on different host systems, a bridged overlay solution must be set up between the two host systems to allow the L2 traffic to pass. Such a solution could be, for example, a simple SSH tunnel similar to the one described in *SSH VPN - Connecting Charon Host and Guest to Customer Network,* a VXLAN overlay network based on native Linux features, or a commercial solution. In all cases, it always requires thorough testing to verify that it fulfills the customer requirements.

Asymmetric Routing Considerations

This section applies to the case where several interfaces are configured on an instance and they all have **IP addresses configured on the Linux level**.

When you add a secondary NIC to a Linux instance, a new interface (that is, an Ethernet device) is added to the instance and automatically recognized by the OS. Depending on the cloud-provider, DHCP may not be active for the secondary VNIC, and you must configure the interface with a static IP address and add any routes that are relevant for the new interface.

Connectivity problems caused by asymmetric routing arise if traffic arrives through one interface and, when the service replies, the reply packets (with the incoming interface's IP address as the source address) go out the other interface. Policy-based routing is required to ensure that packets are sent out via the interface configured with the same IP address that is used as the source IP address in the packet, and to find the correct default gateway (if needed).

Please note:

- The steps below show a simple non-persistent example (can be made persistent via a startup script, or via a persistent network configuration). Please refer to your Linux documentation for details.
- The actual steps depend on your configuration and may vary slightly depending on the specific cloud environment and Linux version. Please always refer to your cloud provider's documentation.

Assumptions:

- The Linux instance has a **primary Ethernet interface** (eth0) with address **10.2.0.8/24** and a public IP address (**PubIP-0**). The default route points to this interface.
- The Linux instance has a **secondary Ethernet interface** (eth1) with address **10.2.0.9/24** which also has a public IP address (**PubIP-1**).
- All firewalls on the operating system and cloud level are set to allow ICMP traffic to both interfaces.
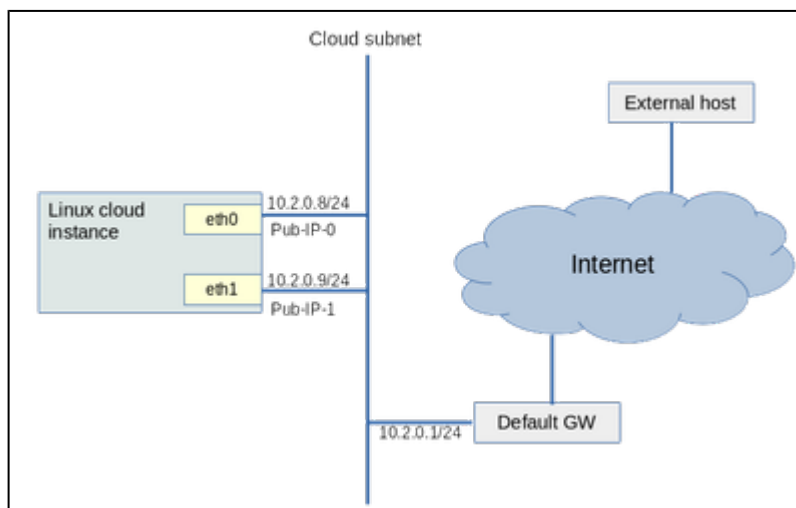
Problem description:

- A ping from an external host to the public IP address *PubIP-0* of the primary Ethernet interface works.
- A ping from an external host to the public IP address *PubIP-1* of the secondary Ethernet interface fails.
- A network trace on the cloud host shows that the ICMP packets to *PubIP-1* arrive at the cloud instance on eth1 as expected, but there is no answer. The reason is that the answer follows the default route via the primary interface, and this traffic is blocked by the cloud provider.

When adding a second IP interface to the Charon host, the routing problems described above can occur. They can be solved by creating a second routing table and adding a routing policy as shown in the following **example which uses the data provided in the assumptions above**:

```
ip route add 10.2.0.9/32 dev eth1 table 99
ip route add default via 10.2.0.1 dev eth1 table 99
ip route add 10.2.0.0/24 dev eth1 table 99
ip rule add from 10.2.0.9 lookup 99
```

The example has the following effect:

- It creates a **non-default routing table** (table ID 99) and adds the routes required for the secondary interface to this table. In particular, any primary or alias IP address assigned to the interface must be added.
- It then defines a **routing policy** that any traffic with the source address of the secondary Ethernet interface must use the non-default routing table. This forces traffic sent to the IP address of the secondary Ethernet interface to also leave the system via this interface.

You can verify the configuration using the commands:

- `ip route show table 99`
- `ip rule show all`

Once you found a configuration solving your problem, you can make the configuration permanent by adding it to a startup script.

Please refer to the Linux man pages for **ip rule** and **ip route** for more information.

Additional information for Charon-SSP marketplace images: the home directory of the **sshuser** contains a script named **active_sec_network.sh**. This script is **only an example** that illustrates how to create a **systemd** service to activate necessary routes and rules during system boot (instead of using steps 7 and 8 above). Do not use this script without carefully adapting it to your requirements - failing to do so, may make your system unreachable.

Cloud Instance and IP Forwarding

If a Charon cloud instance is to forward IP packages between its interfaces (act as a router), in addition to configuring IP forwarding on Linux (`/sbin /sysctl -w net.ipv4.ip_forward=1`), an additional configuration step is required in the configuration of the cloud instance. This configuration has different names in the different cloud environments.

- **AWS and OCI**: source/destination checking must be disabled for all relevant interfaces of the instance.
- **Azure**: IP forwarding must be enabled for all relevant interfaces of the instance.
- **GCP**: IP forwarding must be enabled for an instance **when it is created**.
- **IBM**: IP spoofing must be enabled for all relevant interfaces of the instance.

Without this configuration, the cloud providers block packets that do not contain the IP address of the cloud instance interface in either the source or destination field.

Interface Configuration Basics

This section shows some basic approaches on how to configure the network interfaces on a Charon host for use by the guest system. That is, the interface should be **activated at boot, but without an IP address**. The IP address assigned by the Cloud provider can then be used by the guest system.

It is by no means a complete documentation but should provide a starting point. Further information can be found in the documentation of your Charon Linux host and the documentation of your cloud provider. Please refer to them for any additional information beyond the basic examples below.

The examples show possible configuration steps on

- Linux systems with file-based network configuration (mostly Linux 7.x), and
- Linux system with NetworkManager-based network configuration (mostly Linux 8.x and later)

Please note: the **interface names** used in the following section are for illustrative purposes only. Please familiarize yourself with the interface naming conventions used in your cloud environment.

Expected result of the example:

1. The system should still be reachable via **eth0**.
2. Interface **eth1** should be up without having an IP address configured.

Basic File-based Interface Configuration without NetworkManager

This configuration applies to systems with a file-based network configuration where the NetworkManager is either not active, or where network interfaces should be excluded from NetworkManager control (e.g., to be managed by the Charon Manager). The NetworkManager is disabled by default in older Charon-SSP marketplace images that are based on Centos 7.

Please note:

- The sample configuration assumes a CentOS 7 system and that the interface is configured outside the control of the NetworkManager.
- Should the NetworkManager be active, the plugin **ifcfg-rh** must be enabled in section **main** of the NetworkManager configuration file */etc/NetworkManager/NetworkManager.conf*. It enables the NetworkManager to read and write ifcfg-files.
- After the initial creation of the ifcfg-file, the interface can be managed by the Charon-SSP Manager.
- For the full feature-set of the file-based network configuration, the **network-scripts** package is required.

To make the second interface usable for the Charon guest system, perform the following steps:

1. Add a second interface to your instance as described in the cloud-specific Getting Started guide and your cloud provider's documentation.

2. Log into the instance and become the root user (use: `sudo -i`)

3. Identify the names of the two Ethernet interfaces:
   `# ip link show`

4. Create an interface configuration file for the second interface.

   a. A file for the first interface may exist depending on the default of the cloud environment. In this case, you can copy Example (use correct interface name for your configuration):
      `# cp /etc/sysconfig/network-scripts/ifcfg-eth0 /etc/sysconfig/network-scripts/ifcfg-eth1`

   b. If there is no file that can be copied, you must create the ifcfg-file for the new interface manually.

5. Edit this file to match the characteristics of **eth1** (use correct interface name for your configuration). The private IP address used for this interface will be assigned to the Solaris guest. Therefore, configure the Linux Interface without IP address, similar to the example below.

   ```
   BOOTPROTO=none
   DEVICE=eth1
   NAME=eth1
   ONBOOT=yes
   TYPE=Ethernet
   USERCTL=no
   NM_CONTROLLED=no
   ```

   Please note:
   On some cloud platforms, the automatic cloud-specific configuration prevents the entries in the **ifcfg**-file to take effect (for example on GCP). Please refer to your cloud-provider's documentation and the *additional cloud-specific networking* sections in this guide for more information.

6. Restart the network:
   `# systemctl restart network`
   Please note: Should there be an error when executing this command, kill the DHCP client process and retry the command.

Basic Interface Configuration with NetworkManager

This configuration applies to systems where the NetworkManager is active and network interfaces are under NetworkManager control. The NetworkManager is enabled by default in newer Charon-SSP marketplace images that are based on Rocky Linux 8.x.

Please note:

- The **interface names** used in the following section are for illustrative purposes only. Please familiarize yourself with the interface naming conventions used in your cloud environment.
- The sample configuration assumes a Rocky Linux 8.x system and that the interfaces are under the control of the NetworkManager.
- On some cloud platforms, the automatic cloud-specific configuration prevents the operating system configuration to take effect (for example on GCP). Please refer to your cloud-provider's documentation and the further sections in this document for additional information.

In NetworkManager environments, you have different options to configure network interfaces for use by the guest system. The main options are the following:

1. On a Charon-SSP system, use the Charon Manager Network Settings utility. For this, the interfaces must be under the control of the NetworkManager.
2. On a Linux system with a graphical user interface, use the provided graphical network management tools. This is typically not available in cloud environments.
3. On a Linux system without a graphical user interface, use the **nmtui** utility or **nmcli** commands.
4. Manually create and modify ifcfg-files in /etc/sysconfig/network-scripts. This requires the installation of the deprecated **network-scripts** package in Linux 8.x. In Linux 9.x this method only works for interface types supported by the NetworkManager **ifcfg-rh** plugin which does not have the full **network-scripts** functionality.

The following sections show samples for options 1 and 3.

Charon-SSP Manager Network Settings

The Charon-SSP Manager provides basic network configuration options.

- To access them, start the Charon Manager and open the menu option:
  Tools > Network Settings
- To configure a host system for use by the emulator perform the following steps:
  - Select the correct interface.
  - In the **IP setting** field select **None**.
  - Click on **Apply**.

Please note:

- For RHEL 7 and derivatives, the interfaces to be managed by the Charon Manager must be removed from NetworkManager control and an ifcfg-file must exist.
- For RHEL 8 and later (and derivatives), the interfaces to be managed by the Charon Manager must be under NetworkManager control.

Using the *nmtui* Utility

The **nmtui** utility provides a method to configure the network settings for the NetworkManager in a text-based environment without having to know the *nmcli* commands. It is provided via the **NetworkManager-tui** package.

The following **basic example** shows how to remove the IP address from the interface and how to reactivate the interface afterwards.

- Start the tool as the root user: `# nmtui`
- Use the up/down and left/right arrows to navigate.
- Select **Edit a connection** and press **RETURN**.
- Select the interface you want to configure.
- Select **Edit** on the right side and press **RETURN**.
- To make the interface come up without an IP address at boot, set the IP configuration to disabled (pressing **RETURN** on the value field will open a menu), and enable the automatic connection.
- Select **OK** and press **RETURN**.
- Navigate back to the main screen.
- Select Activate a connection and press **RETURN**
- **Select the interface** you want to reactivate and select **Deactivate** on the right. Press RETURN.
- Repeat the steps for the **Activate** option to reactivate the interface.
- Navigate back to the main screen and end the session.

Using *nmcli* Commands

To configure the interface dedicated to the emulator such that it receives no IP address but is activated at start, you could use command similar to the following:

1. Identify the NetworkManager connection to configure. The interface may have been automatically activated by the NetworkManager. In the example, it is "Wired connection 1" on device eth1.

```
# nmcli conn show
NAME                UUID                                    TYPE      DEVICE
System eth0         5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03    ethernet  eth0
Wired connection 1  027a1c2b-3397-37fb-a6e2-f2e02eb59992    ethernet  eth1
```

If there is no connection for the interface yet, check if the device is visible using the command `nmcli dev status` or `ip link show`.

2. For an existing connection:

    a) Configure an appropriate name for the connection if required:

```
# nmcli conn mod "Wired connection 1" con-name eth1
```

    b) Set the IP configuration such that no IP address is assigned:

```
# nmcli conn mod eth1 ipv4.method "disabled" ipv6.method "disabled"
```

    c) Configure automatic interface activation at boot:

```
# nmcli conn mod eth1 connection.autoconnect yes
```

3. **If no connection for the second interface exists**:

    Add a new connection (with automatic interface activation, without IP address):

```
# nmcli conn add con-name eth1 type ethernet ifname eth1 autoconnect yes ipv4.method "disabled" ipv6.method "disabled"
```

4. (Re-)Activate the connection (this command may time out if IP connection check is enabled):

```
# nmcli con up eth1
```

Further Information

The following sections provide additional information:

- Network Interface Management
- SSH VPN - Connecting Charon Host and Guest to Customer Network
- Dedicated NIC for Guest System
- Interface MTU Considerations

Network Interface Management

To add an additional network interface to an instance or to remove an interface from your instance perform the steps described below.

Please note: The steps below only provide a basic overview. The exact tasks required will vary depending on your network design. Please refer to the Azure documentation for details.

Contents

- Adding an Additional Network Interface
- Adding a Public IP address to the Second NIC
    - Creating a Public IP Address
    - Attaching the Public IP Address to an Interface
- Address Assignment Information

Adding an Additional Network Interface

It is possible to add additional NICs to the Charon host system in the cloud. Additional NICs can be used to connect the instance to additional subnets. And an additional NIC can also be used to provide an emulated SPARC system running in the emulator with a dedicated network interface. This section shows the basic steps to add a NIC to an existing Azure cloud instance.

Please note: The VM must be stopped before the new interface can be attached to the system. The VM also must be stopped before detaching an interface.

To add an existing or newly created NIC to the Charon host cloud instance, perform the following steps:

Step 1: Open the **Networking** screen of your Charon host cloud instance. The sample below shows a VM with only one NIC.

Step 2: Click on **Attach network interface** at the top. This will either let you choose an existing interface or offer the option to create a new one as shown in the sample below:
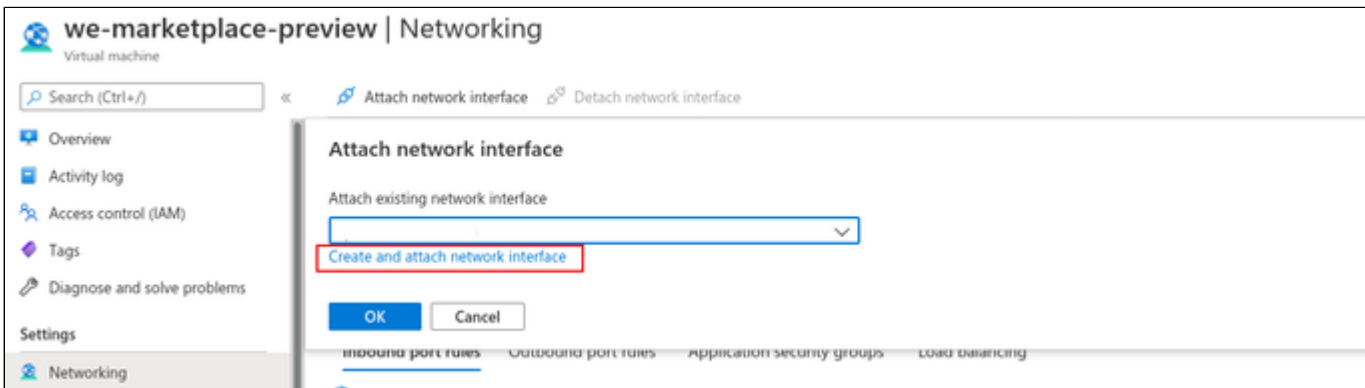
Step 3: Click on **Create network interface** to open the interface creation screen.


Add the mandatory information:

- Name
- Subnet
- Private IP (if static IP is selected)
- Resource group (same as VM)

Once the information has been added, click on **Create** to create the new NIC. This will return you to the **Networking** screen.

Step 4: Attach the selected or newly created interface to your cloud VM.

Please note: make sure to stop the instance before attaching the interface.

On the Networking screen, you should now see the new interface (if required, open the subsection of the screen by clicking on **Attach network interface**). Click on **OK** to add the NIC to your VM. The image below shows an example:
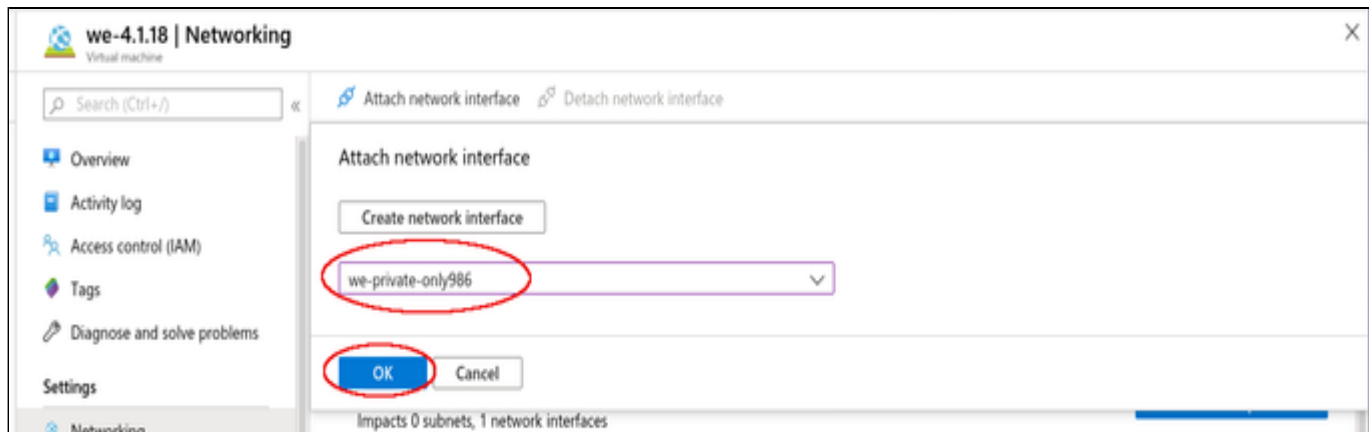
The **Networking** screen should now show the two interfaces in two different tabs, as displayed below:
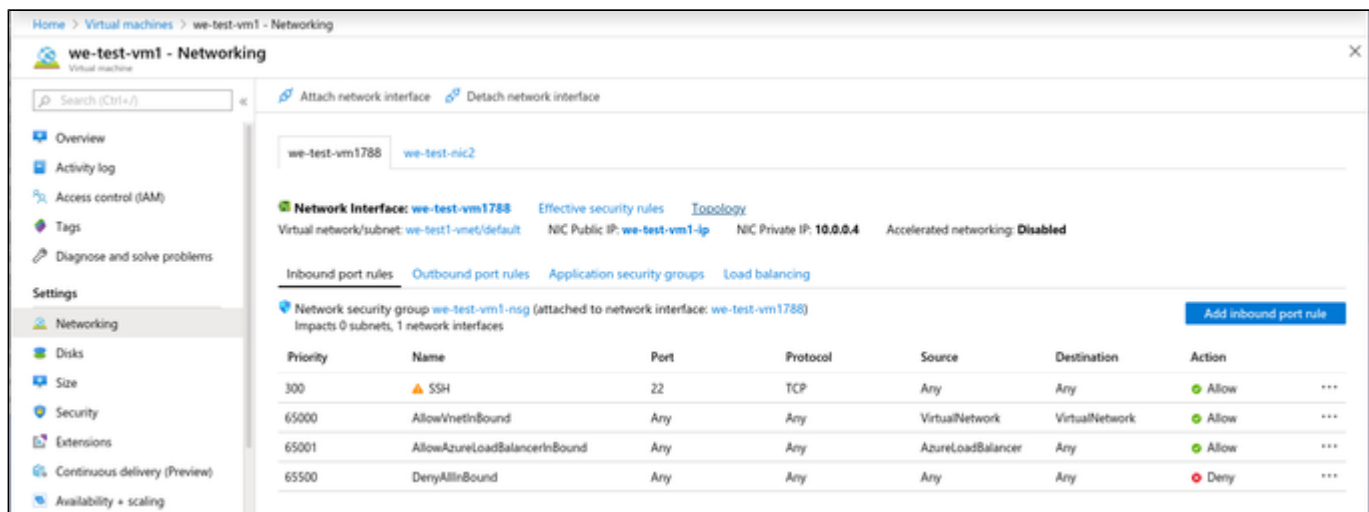
Step 5: Restart your VM, log in and check the availability of the second interface.

The output of the **ip link show** command now shows the second interface (eth1 in the example).

```
[root@we-test-vm1 charon]# ip link show |grep eth
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT group default qlen 1000
    link/ether 00:0d:3a:8f:da:f2 brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT group default qlen 1000
```
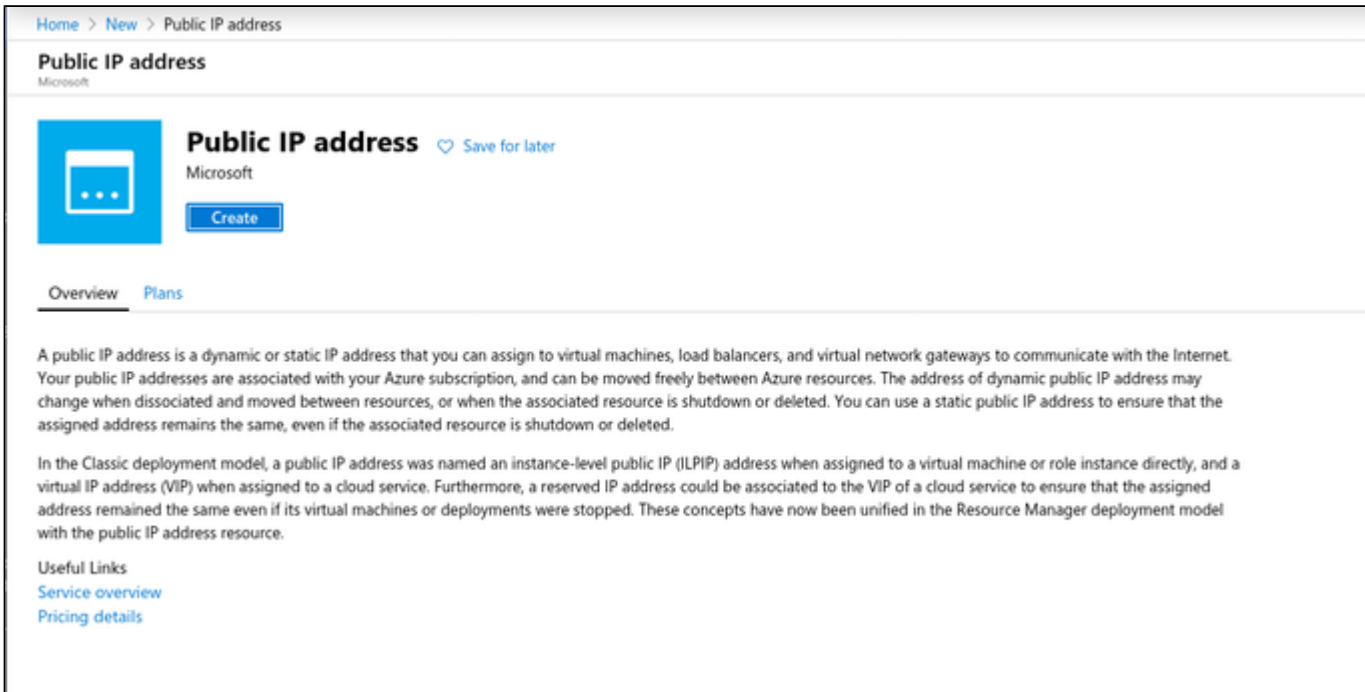
Note that for the new interface DHCP or another IP configuration on the host operating system level is not active automatically. Depending on the underlying Linux version, an ifcfg file for the interface has to be created manually in */etc/sysconfig/network-scripts/* (mostly Linux 7.x) or set up via **nmcli** or **nmtui** (mostly Linux 8.x).

Adding a Public IP address to the Second NIC

Please note: If this NIC is used as a dedicated NIC for a legacy guest operating system, it is not recommended to expose the legacy guest operating system to the Internet by adding a public IP to the dedicated NIC. It is highly recommended to use a secure, encrypted VPN connection for any external traffic of the guest system. The user is responsible for making sure that any external connection complies with the relevant rules and policies of the environment.

Creating a Public IP Address

A public IP address is a resource that can be created using the **Create a resource** icon on the **Home** screen. On the overview page, search for public IP address. This will open the initial screen to create public IP address:

Clicking on **Create** takes you to the IP address configuration screen:

Add the necessary data, in particular the following:

- Name
- Dynamic (changes every time the instance is restarted) or Static (persistent)
- Resource group (same as your interface)
- Location

Click on **Create** to create the address resource.

Attaching the Public IP Address to an Interface

To attach the new public IP address to the second NIC of your instance, perform the following steps:

- Select your VM in the overview list.
- Go to the **Networking** section.
- Select the interface tab of the relevant interface.
- On the interface tab page, click on the **interface name link** next to the **Network interface** label.
- On the network interface page, under **Settings** (in the left-hand pane), select **IP configurations**.
- Click on the name of the IP configuration (in the list at the bottom of the page) to which you want to add the public IP address. This will open the IP configuration screen.

All services > Virtual machines > we-testingAL > we-testing2-nic2 >

ipconfig1 ...
we-testing2-nic2

🖫 Save   ✕ Discard

Public IP address settings
Public IP address
(Disassociate) Associate

Private IP address settings

Virtual network/subnet
stromasyslabvnet422/default

Virtual machine
we-testingAL

> ℹ The associated virtual machine 'we-testingAL' must be either stopped or deallocated in order to be able to edit the private IP address.

Assignment
(Dynamic   Static)

IP address
10.2.0.9

On this screen select **Associate** for the **Public IP address**. This will allow you to select the new public IP address.

Then click on **Save** (at the top of the section) to save the changes.

Please note: do not forget to add an appropriate Network Security Group to protect the interface.

Address Assignment Information

You can divide a VNet into multiple subnets for organization and security. Each NIC in a VM is connected to one subnet in one VNet. NICs connected to subnets (same or different) within a VNet can communicate with each other without any extra configuration. You can change the subnet a VM is connected to after it's created, but you cannot change the VNet. Each NIC attached to a VM is assigned a MAC address that doesn't change until the VM is deleted.

Reserved addresses in each subnet:

The addresses x.x.x.0 to x.x.x.3 in every subnet and the last addresses in every subnet are reserved.

- x.x.x.0: network address
- x.x.x.1: default gateway
- x.x.x.2, x.x.x.3: DNS services
- x.x.x.255: broadcast address

Please note:

- The automatically configured default router of a subnet does not respond to a **ping** command.
- The **traceroute** command does not work.
- Multicast, broadcast, and GRE are not supported.

Public IP addresses:
By default, public IP addresses are **dynamic** and the address associated to them may change when the VM is stopped or deleted. To ensure that the VM always uses the same public IP address, create a **static** public IP address. One public IP address can be assigned to an instance at instance creation. Additional public IP addresses can be created using by creating a new resource
(**Home > Create a Resource > search for Public IP address and select resource > Create**) and assigning the resource to an instance interface.

SSH VPN - Connecting Charon Host and Guest to Customer Network

Contents

Applicable Charon Products

The configuration shown in this chapter should work for any Charon product.

Prerequisites:

- Charon host system is a supported Linux system.
- The product must support TAP interfaces for the emulated system Ethernet interface.
- The remote endpoint is a Linux system.

Overview

If the connection between the Charon host system, including the configured legacy guest systems, and the rest of the customer's network runs over a public network as is the case for Charon in cloud environments it is necessary to secure the traffic against unauthorized access.

The example in this section describes how to configure a bridged SSH-based VPN tunnel between the Charon host and a remote Linux system across a public network.

Please note:

- The customer is responsible for ensuring that any VPN solution meets the requirements of his or her company's security guidelines. The example in this chapter is only for illustrative purposes.
- The advantage of a bridged connection is that L2 protocols are also supported.

Once the sample configuration has been set up, it can be used for

- communication between host and guest system,
- communication between customer network and guest system.

The following image shows an overview of the sample topology:
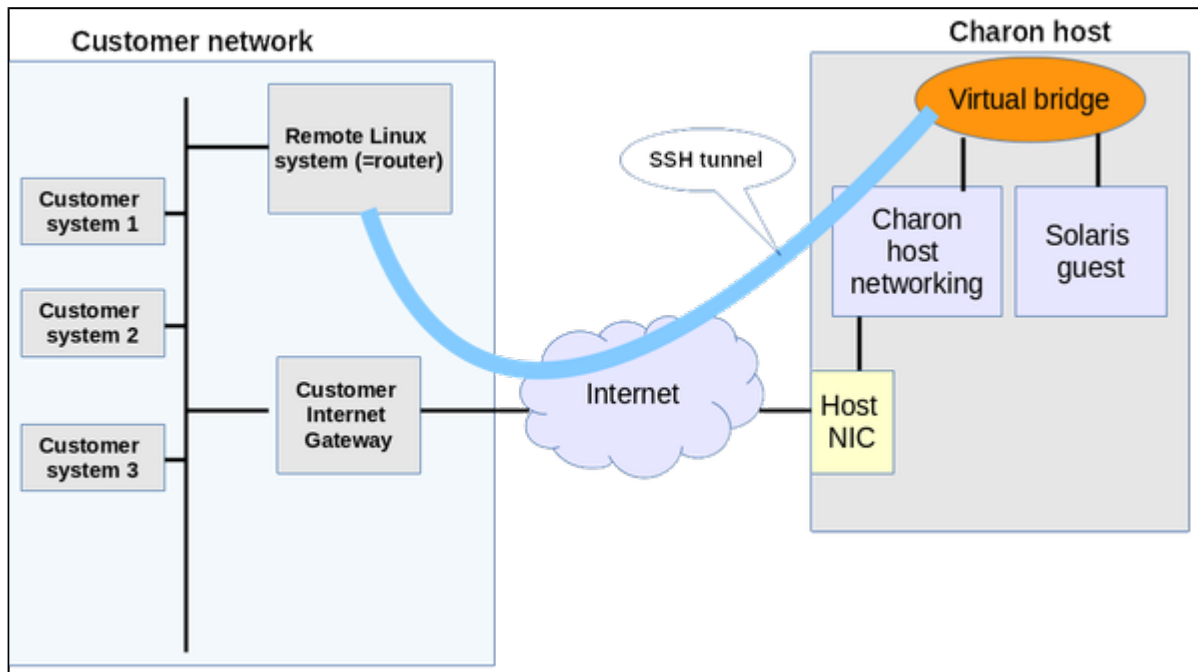
Prerequisites

The example shows how to create a local virtual bridge connecting host and guest system on the host, and a set of commands on the remote Linux System to create an SSH VPN tunnel.

For this configuration to work, the following prerequisites must be met:

- The remote Linux system must have access to the public IP address and the SSH port of the Charon host.
- The private key necessary to access the instance must be available on the remote Linux system and the public key must be installed on the Charon host system for the user used to set up the tunnel.
- *autossh* package must be installed on the remote Linux system.
- If using the Charon-SSP Manager on a Linux 7.x system, the *bridge-utils* package is also required.

Key-Pair Creation and Public Key Installation on Host System

The SSH tunnel requires a key-pair consisting of a private and a public key. The private key is used by the remote Linux system to identify itself with the Charon host system, which must have the matching public key installed.

The creation of the key-pair and the installation of the public key require different steps depending on the Charon product used.

Conventional Linux Installation

Creating a Key-Pair

This step is performed **on the remote Linux system** that will be used as the tunnel end-point.

Please note: If the key-pair is not created automatically during the launch of the instance, you can create it using a command similar to the following:

```
Sample 1:  # ssh-keygen -t rsa -b 4096 -f ~/.ssh/<keyname> -q
Sample 2:  # ssh-keygen -t ecdsa -f ~/.ssh/<keyname> -q
```

By default, the key files are stored in the **.ssh** directory of the current user. The resulting key-pair can then be associated with instance during instance creation or later by adding it to the *authorized_keys* file of the correct user on the target system. This will allow the key to be used to create an encrypted SSH connection.

Please note: if your management system supports it, for RHEL 9.x, Rocky Linux 9.x, and Oracle Linux 9.x use SSH key types ECDSA or ED25519. This will allow connecting to these Linux systems using an SSH tunnel without the default crypto-policy settings on the Charon host having to be changed for less secure settings. See also: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9/html/security_hardening/using-the-system-wide-cryptographic-policies_security-hardening.

Installing the Public Key on the Charon Host

Connect to the Charon host (as user **root**) via SFTP or SCP and copy the **public key file** to the system. The following example shows the steps for SFTP:

$ **sftp root@**<charon-ssp-host-ip>

sftp> **put** <path-to-public-key> **<keyname>.pub**

sftp> **bye**

Log in to the Charon host system and add the content of **<keyname>.pub** to **/root/.ssh/authorized_keys** (using a text editor). Make sure, the file permissions of the authorized_keys file are set such that the owner has read/write access to the file and others have no access. If required, also add restrictions regarding the allowable commands to the file using the '**command="***filename***"**' command at the beginning of the line containing the SSH key.

Charon Cloud-Specific Marketplace Images

The key-pair is (optionally) created and then assigned to the Charon cloud instance at first launch. If a new key-pair is created, the private key can be downloaded during creation.

Make sure to store the private key in a safe place. If it is lost, access to the instance may be permanently lost.

Adapting the SSH Daemon Configuration of the Charon Host System

The default configuration of the SSH daemon must be adapted to allow tunnel setup and root login by SSH key only. To do this, perform the following steps:

1. Login as the **root** user.
2. Open the file **/etc/ssh/sshd_config** with a text editor.
3. Set the following parameters:
   ```
   PermitRootLogin without-password
   PermitTunnel yes
   ```
4. Restart the SSH service:
   ```
   # systemctl restart sshd
   ```

Setting up the VPN Tunnel

The image below shows a more detailed picture of the connection between the remote Linux system and the Charon host. This section describes how to configure this sample setup.
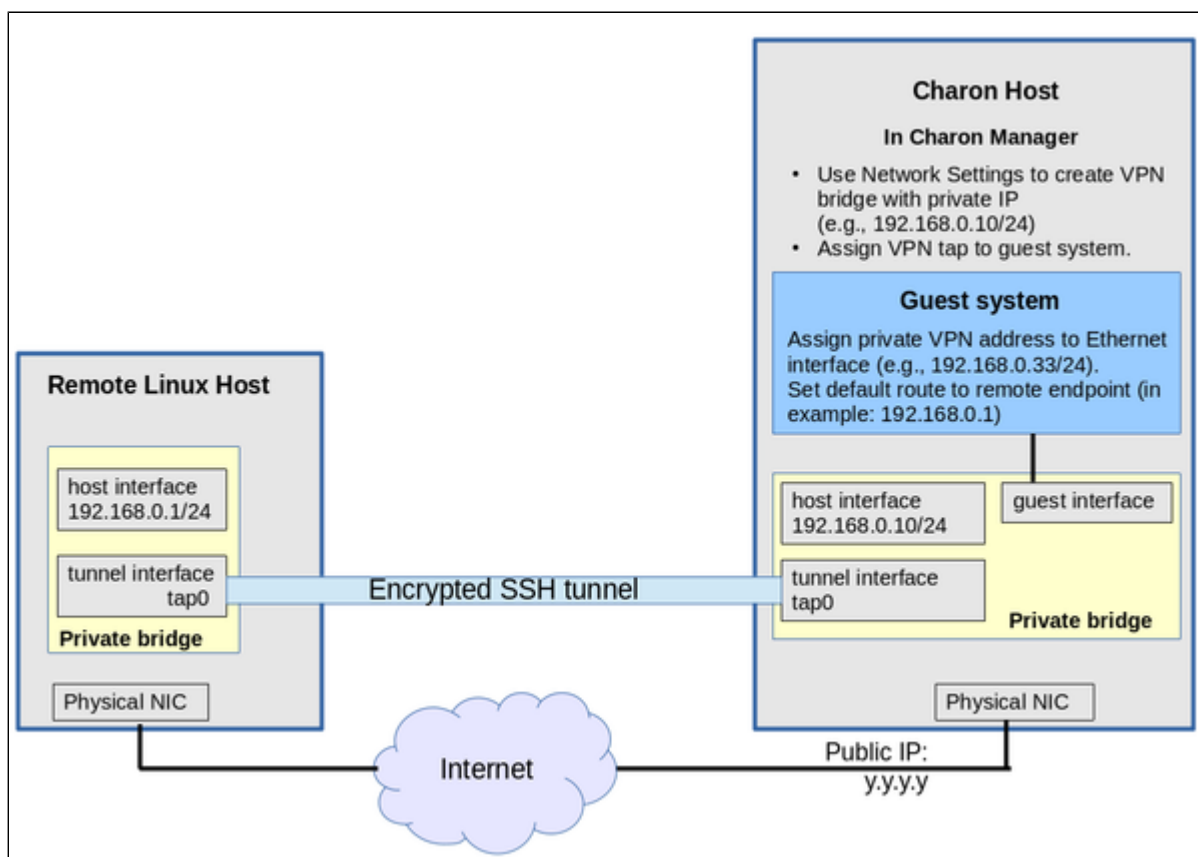
Steps on the Charon Host System

Creating a VPN Bridge Manually

To create a bridge for the VPN tunnel manually, use commands similar to the following:

```
ip tuntap add dev tap0_vpn0 mod tap
ip link set tap0_vpn0 up

ip tuntap add dev tap0 mod tap
ip link set tap0 up

ip link add name br_vpn0 type bridge
ip link set br_vpn0 up
ip addr add 192.168.0.10/24 dev br_vpn0

ip link set tap0_vpn0 master br_vpn0
ip link set tap0 master br_vpn0
```

The above commands are not persistent across reboots.

To make the configuration permanent, you can use **ifcfg-files**, **nmcli commands**, **or a custom startup script** - depending on your requirements and your host operating system version.

Linux version	network-scripts method (ifcfg-files)	NetworkManager (nmcli)
7.x	Installed by default. **NM_CONTROLLED=no** forces ifcfg-file use	Installed by default.
8.x	Deprecated but available; needed if TAP interfaces are to be configured in ifcfg-files.	Preferred configuration method. Used by the Charon-SSP Manager starting with Linux 8
9.x	No longer available. For interface types supported by the ifcfg-rh plugin, ifcfg-files can be used.	Only method with full functionality. Must be used for TAP interfaces.

Sample ifcfg-files for **CentOS/RHEL 7**:

```
# cat /etc/sysconfig/network-scripts/ifcfg-br_vpn0
DEVICE=br_vpn0
NAME=br_vpn0
TYPE=Bridge
ONBOOT=yes
DEFROUTE=yes
STP=no
BOOTPROTO=none
IPADDR=192.168.0.10
NETMASK=255.255.255.0
NM_CONTROLLED=no

# cat /etc/sysconfig/network-scripts/ifcfg-tap0
DEVICE=tap0
NAME=tap0
BRIDGE=br_vpn0
TYPE=Tap
ONBOOT=yes
NM_CONTROLLED=no

# cat /etc/sysconfig/network-scripts/ifcfg-tap0_vpn0
DEVICE=tap0_vpn0
NAME=tap0_vpn0
BRIDGE=br_vpn0
TYPE=Tap
ONBOOT=yes
NM_CONTROLLED=no
```

Note that on **CentOS/RHEL 8** the interfaces must be under NetworkManager control (that is, the NM_CONTROLLED command **must be removed or set to yes**) if the interfaces are later to be managed by the Charon-SSP Manager.

Please note that the **network-scripts** package is no longer available in **RHEL 9** and derivatives. While the **ifcfg-rh plugin** can handle many of the legacy ifcfg file configurations, **it cannot handle TAP interfaces**. Therefore **nmcli** should be used to make the bridge configuration permanent (as shown in the sample below).

```
nmcli conn add type bridge con-name br_vpn0 ifname br_vpn0 ipv4.method manual ipv4.addresses 192.168.0.10/24 \
      ipv6.method disabled
nmcli conn add type tun mode tap autoconnect yes con-name tap0_vpn0 ifname tap0_vpn0 master br_vpn0
```

Creating a VPN Bridge using the Charon-SSP Manager

To configure the SSH VPN connection, you must setup a private VPN bridge (called a virtual network in the Charon context) using the Charon Manager. Use the following steps to perform this task:

1. Open the Charon-SSP Manager and log in to the Charon-SSP host.

2. In the Charon Manager, open the Network Settings window by clicking on **Tools > Network Settings**. This will open the **Network Settings** window.

3. Click on **Add** and then on **Virtual Network** to open the virtual network configuration window. This will open the **Add Virtual Network** configuration window as shown below.

4. Enter the required information as shown below:

Perform the following steps to configure a VPN bridge:

- Set **Create for SSH VPN** to **ON**,
- Enter the **Number of virtual adapters** (TAP interfaces) required. These interfaces will be assigned to the emulated SPARC systems as Ethernet interfaces.
- Configure the **IP address** for the bridge interface.
- Set the **Netmask**.

Please note: this interface and the interface on the remote Linux system must be in the **same IP subnet.**

Click on **OK** to save your configuration.

Assigning the Guest Ethernet Interface

All Charon Emulator products

Adapt the configuration file to use the tap0_vpn0 interface in the above example for the emulated Ethernet interface.

Charon-SSP Manager

One of the TAP interfaces created in the step above, must be assigned to the Solaris guest system to add it to the LAN that will be tunneled across SSH to the remote Linux system.

Perform the following steps:

1. Open the Charon-SSP Manager and log in to the Charon-SSP host.

2. In the Charon Manager, select the guest system and then the **Ethernet** configuration category on the left. Assign one of the created TAP interfaces to the guest (see example below).

Virtual Machine Settings

Device	Summary
Model	SUN-4U
CPU	1
DIT	Client JIT
Memory	1 GB
Graphics	Disabled
SCSI	SCSI 0, SCSIX 6
TTYA	9000
TTYB	Disabled
Audio	Disabled
Ethernet	tap0_vpn0

Ethernet

Add-on adapter model: HME

Interface	Model	MAC Address
tap0_vpn0	HME	

Click on **OK** to save the configuration change.

If the emulated instance is currently running, the guest must be shut down and the emulated instance must be restarted for the change to become active.

Steps on the Remote Linux System

Please note: the steps on the Charon host must be performed first.

As the user **root** on the remote Linux system, perform the following steps to set up the VPN tunnel according to the overview image above:

Action	Command
Create TAP interface	`# ip tuntap add dev tap0 mod tap`
Enable TAP interface	`# ip link set tap0 up`
Create bridge	`# ip link add name br_vpn0 type bridge`
Enable bridge interface	`# ip link set br_vpn0 up`
Define IP address for bridge	`# ip addr add 192.168.0.1/24 dev br_vpn0`
Add TAP interface to bridge	`# ip link set tap0 master br_vpn0`

Start the SSH tunnel
 	
autossh is a program to start a copy of **ssh** and monitor it, restarting it as necessary should it die or stop passing traffic.
 	
Once started, you can move the program to the background.
	`# autossh -M 9876 -o ServerAliveInterval=60 -o Tunnel=ethernet \`
	`  -w 0:0 -t -i <path-to-private-key> -NCT <username>@<public-AWS-instance-IP>`
	
	The value for **username** depends on the Charon product:
	
	All products: **root** (or another configured user with the correct privileges and authorized_keys file). Charon-SSP Baremetal and Charon marketplace images: **sshuser**
	
	-M defines the monitoring port autossh uses to monitor the connection
	-o sets SSH options (bridged tunnel and keepalive)
	-i denotes the path to the private key matching the public key copied to the host system.
	-w denotes the number of the local and remote tunnel interfaces for tunnel device forwarding (e.g., the 0 in interface tap0).
	-N denotes that no remote command should be executed
	-T disables pseudo-terminal allocation
	-C requests data compression

Steps on the Guest System

Set the IP address on the Ethernet interface to an address within the VPN subnet. To follow the example above, you would set the address to 192.168.0.33/24.

Making the configuration permanent on a Solaris system (as an example):

- Change the address in **/etc/hosts** for the hostname specified in **/etc/**_<interfacename>_**.hostname**.
- On Solaris 11, use the commands `ipadm create-ip net`_X_ and `ipadm create-addr -T static -a` _<ip-address>_/_<netmask>_ `net`_X_`/v4`.

Functionality after Creating the Tunnel

After the tunnel has been created, the guest system can communicate with

- the Charon host system, and
- the remote Linux system.

Routing between the Guest System and the Customer Network

After following the description above, the guest system can be reached from the systems that are also connected to the virtual bridge (in the example: remote Linux system and host system). To enable the guest system to **communicate with other systems** in the customer network (or the Internet) over the VPN connection, perform the following steps:

Steps on the Guest System

Add the VPN address of the remote Linux system as the default gateway for the guest system. To follow the example above, use the command

```
# route add default 192.168.0.1
```

Steps on the Remote Linux System

To turn the remote Linux system into a router for other systems in the customer network, perform the following steps:

1. Enable IP Forwarding using the command:
   `# sysctl -w net.ipv4.ip_forward=1`
   To make permanent, add `net.ipv4.ip_forward=1` to the file **/etc/sysctl.conf**.
2. If the Linux firewall is enabled (firewalld assumed), allow the forwarding of packets through the firewall:
   `# firewall-cmd --permanent --direct --add-rule ipv4 filter FORWARD 0 -i` _<tunnel-bridge-interface>_ `-o` _<NIC-to-LAN>_ `-j ACCEPT`
   `# firewall-cmd --permanent --direct --add-rule ipv4 filter FORWARD 0 -o` _<NIC-to-LAN>_ `-i` _<tunnel-bridge-interface>_ `-j ACCEPT`

Steps on Other Systems in the Customer Network

To enable communication between other systems in the customer network, configure a static route to the SSH tunnel LAN subnet via the Linux router.

Functionality after Setting up the Sample configuration

After the sample configuration has been created, the Solaris guest can communicate with

- the Charon host system,
- the remote Linux, and
- other systems on the customer LAN that have been configured with the correct route via the Linux router.

Sample output from a system inside the customer LAN:

- IP address in different IP subnet
- Routing entry via the Linux router
- Ping to guest system

```
$ ifconfig enp0s25
enp0s25: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.118  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 2003:ee:f03:5e56:2d06:a50d:7762:7dc4  prefixlen 64  scopeid 0x0<gl
obal>
        inet6 fe80::7b0c:403e:8cec:7103  prefixlen 64  scopeid 0x20<link>
        ether f0:de:f1:87:7c:b9  txqueuelen 1000  (Ethernet)
        RX packets 600969  bytes 881599304 (840.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 294711  bytes 26134015 (24.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xf2600000-f2620000

$ ip route get 192.168.0.33
192.168.0.33 via 192.168.2.80 dev enp0s25 src 192.168.2.118 uid 1000
    cache
$
$ ping 192.168.0.33
PING 192.168.0.33 (192.168.0.33) 56(84) bytes of data.
64 bytes from 192.168.0.33: icmp_seq=1 ttl=254 time=1.97 ms
64 bytes from 192.168.0.33: icmp_seq=2 ttl=254 time=2.90 ms
64 bytes from 192.168.0.33: icmp_seq=3 ttl=254 time=1.12 ms
```

Sample login to the guest system from host inside customer LAN:

```
$
$ telnet 192.168.0.33
Trying 192.168.0.33...
Connected to 192.168.0.33.
Escape character is '^]'.


SunOS 5.6

login: root
Password:
Last login: Wed Aug 14 20:22:05 from 192.168.2.118
Sun Microsystems Inc.   SunOS 5.6       Generic August 1997
#
# uname -a
SunOS ent450-1 5.6 Generic_105181-03 sun4u sparc SUNW,Ultra-4
#
# ifconfig -a
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
        inet 127.0.0.1 netmask ff000000
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
        inet 192.168.0.33 netmask ffffff00 broadcast 192.168.0.255
        ether c8:8c:69:33:d1:2
```

Dedicated NIC for Guest System

Contents

Providing a dedicated NIC for guest operating systems is the standard method in non-cloud environments. However, this configuration poses some challenges in cloud environments where MAC address / IP address combinations are fixed parameters set by the cloud provider.

This section will provide some information about how to configure such a setup in a cloud environment. **It is not specific to one cloud provider. Hence, the descriptions may refer to different cloud providers if appropriate.**

Basic Concept

The following images illustrates the basic concept when working with a dedicated network interface for the guest operating system. There are, of course, many variations depending on the specific environment.

Scenario: host and guest system have a dedicated NIC. The NIC used by the Charon host has a private and a public IP address, the NIC used by the guest system a private IP address and optionally a public IP address. The Internet and VPN gateways are only used for illustration and are not part of this example.

Charon host instance with two NICs
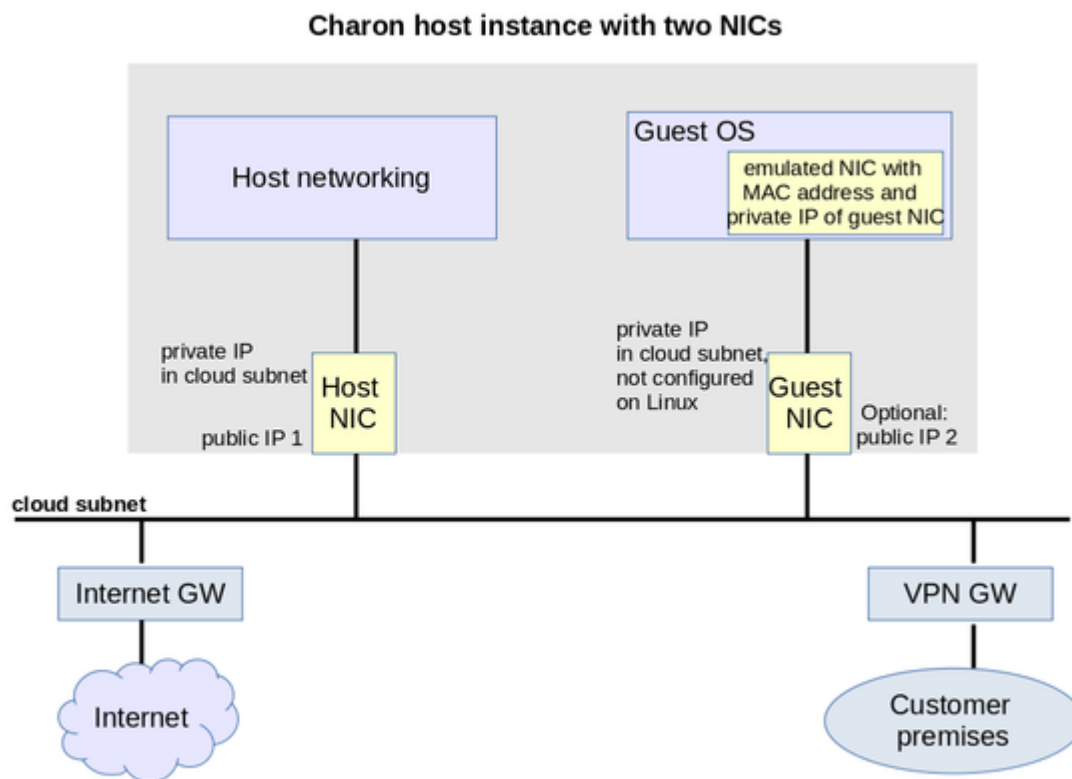
Please note: If the NIC dedicated to the guest OS does not have a public IP address, the guest system may still be able to access the Internet via the customer network reachable across a VPN gateway. This will depend on the customer specific network configuration. This type of connection is the recommended way to provided external network access to the guest system as the VPN ensures that traffic across a public network is encrypted.

The basic steps to implement the above configuration are as follows:

- Create a cloud instance in which the Charon host system runs.
- Add two NICs to the Charon host system. One for the Charon host and one for the guest system.
- Configure the appropriate access rules for instance and NICs.
- One NIC is dedicated to the Charon host, one to the guest system. Configure a private and public IP address for the NIC used by the Charon host. Configure a private IP address for the NIC used by the guest system (and optionally a public IP address - not recommended).
- On the Charon host (Linux level), remove the private IP address from the NIC dedicated to the guest system if it was automatically configured and ensure that the interface will be enabled when the system starts.
- Assign the appropriate NIC to the guest system.
- Configure the guest system MAC address to be the same as the one of the NIC selected for the guest (if this is not done automatically by your emulator product).
- After booting the guest system, configure the private IP originally assigned by the cloud provider to the NIC dedicated to the guest as the IP address of the guest Ethernet interface.
- Set the default route of the guest system to the default gateway or VPN gateway of the LAN.

Depending on firewall rules and cloud-specific security settings, the guest system should then be able to communicate with the following systems:

- The host system.
- The other systems in cloud-internal network (e.g. other guest and host systems).
- The customer internal network via a previously configured VPN gateway.
- Directly with the Internet if a public IP address was configured for the interface (not recommended).

The additional sections in this chapter show the basic configuration steps for the above example.

Please note:

- In this scenario any direct traffic between host and guest system (if configured with a public IP address) and external systems reachable via the Internet gateway is not encrypted by default. If this traffic runs across a public network, it is exposed to being monitored and even modified by third parties. The user is responsible for ensuring data protection conforming to the user's company security rules. It is strongly recommended to use encrypted VPN connections for any sensitive traffic.

- Guest operating systems are often old and no longer maintained by the original vendor. This means they are more easily compromised by attacks from the Internet. Therefore, direct Internet access for the guest system is not recommended.
- The actual configuration steps vary depending on the cloud environment used. The sample configuration below will have to be adapted to the specific environment.
- If you are using the **Charon-PAR emulator**, it is recommended not to assign the dedicated NIC directly to the emulator, but use a MACVTAP interface connected to the dedicated NIC instead. Please refer to the Ethernet configuration section of the Charon-PAR user's guide for detailed information. The preparation of the dedicated NIC for use with a MACVTAP interface is the same as described below.

Configuration Examples

Important information:

- The example assumes that the host operating system is a RHEL version 7 or 8 compatible Linux system. If you use a different host operating system version, you must adapt the example accordingly.
- **For AWS**, remember that any automatically assigned public IP addresses will be removed by the cloud provider once the instance is restarted with a second NIC. Hence, AWS Elastic IP addresses must be used as public IP addresses.
- **For Google cloud**, note the following:

  - The default is that all interfaces are configured with IP addresses automatically by GCP services on the Linux host. Please refer to the *Network Management* section for information on how to disable this automatic configuration.
  - Some base images used to create a Charon host instance may be configured to use /32 netmasks for additional interfaces, and only ARP requests for the default gateway are answered by Google. This can cause communication problems between the legacy guest system (e.g., Solaris) and other instances on the same subnet (ARP requests are not answered). The workaround is to use static ARP entries on the legacy guest system. Please refer to the *Getting Started* guide for more information. Current images provided by Stromasys use /24 netmasks, so this point does not apply to them.
- The interface names used in this example (eth0 and eth1) may be different on your system. Please verify the names on your system and refer your cloud provider's documentation for more detail. **Make sure you use the correct names**!
- The example uses only a private address for the dedicated interface. If a public address is required, the basic steps for making the interface available to the guest system are the same.

Step 1: Prepare Second NIC on Host for Use by the Guest System

The host system interface configuration must ensure that the private IP address allocated to the new interface by the cloud provider is not configured on the Linux Ethernet interface. This address will be used by the guest system.

The configuration depends on whether the network configuration on the Linux host is file-based (typically Linux 7.x) or NetworkManager-based (typically Linux 8.x and above**)**. Examples for both are shown below. Typically, the NetworkManager is disabled on Charon-SSP marketplace images based on Linux 7.x, and enabled on Charon-SSP marketplace images based on Linux 8.x.

Expected result of the examples in this section:

1. The system should still be reachable via **eth0**.
2. Interface **eth1** should be up without having an IP address configured.

Please note: if you are using the **Charon-PAR emulator**, it is recommended not to assign the dedicated NIC directly to the emulator, but use a MACVTAP interface connected to the dedicated NIC instead. Please refer to the Ethernet configuration section of the Charon-PAR user's guide for detailed information. The preparation of the second NIC for use by the guest system should be performed as described here.

File-based Configuration Examples

This configuration applies to systems with a file-based network configuration where the NetworkManager is either not active, or where network interfaces should be excluded from NetworkManager control (e.g., to be managed by the Charon Manager). The NetworkManager is disabled by default in older Charon-SSP marketplace images that are based on Centos 7.

Please note:

- The sample configuration assumes a CentOS 7 system and that the interface is configured outside the control of the NetworkManager.
- Should the NetworkManager be active, the plugin **ifcfg-rh** must be enabled in section **main** of the NetworkManager configuration file */etc /NetworkManager/NetworkManager.conf*. It enables the NetworkManager to read and write ifcfg-files.
- After the initial creation of the ifcfg-file, the interface can be managed by the Charon-SSP Manager.
- For the full feature-set of the file-based network configuration, the **network-scripts** package is required.

To make the second interface usable for the Charon guest system, perform the following steps:

1. Add a second interface to your instance as described in the cloud-specific Getting Started guide and your cloud provider's documentation.

2. Log into the instance and become the root user (use: **sudo -i**)

3. Identify the names of the two Ethernet interfaces:
   ```
   # ip link show
   ```

4. Create an interface configuration file for the second interface.
   a. A file for the first interface may exist depending on the default of the cloud environment. In this case, you can copy Example (use correct interface name for your configuration):
      ```
      # cp /etc/sysconfig/network-scripts/ifcfg-eth0 /etc/sysconfig/network-scripts/ifcfg-eth1
      ```

   b. If there is no file that can be copied, you must create the ifcfg-file for the new interface manually.

5. Edit this file to match the characteristics of **eth1** (use correct interface name for your configuration). The private IP address used for this interface will be assigned to the guest system. Therefore, configure the Linux Interface without IP address, similar to the example below.

   ```
   BOOTPROTO=none
   DEVICE=eth1
   NAME=eth1
   ONBOOT=yes
   TYPE=Ethernet
   USERCTL=no
   NM_CONTROLLED=no
   ```

   Please note:
   On some cloud platforms, the automatic cloud-specific configuration prevents the entries in the **ifcfg**-file to take effect (for example on GCP). Please refer to your cloud-provider's documentation and the *Network Management* section in the *Getting Started Guide* of your version for additional information.

6. Restart the network:
   ```
   # systemctl restart network
   ```
   Please note: Should there be an error when executing this command, kill the DHCP client process and retry the command.

NetworkManager-based Configuration Examples

The following sections show two examples:

- Configuration using **nmcli** commands
- Configuration using the Charon-SSP Manager

Using nmcli Commands

To configure the interface dedicated to the emulator such that it receives no IP address but is activated at start, you could use command similar to the following:

1. Identify the NetworkManager connection to configure. The interface may have been automatically activated by the NetworkManager. In the example, it is "Wired connection 1" on device eth1.

```
# nmcli conn show
NAME                UUID                                   TYPE      DEVICE
System eth0         5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03   ethernet  eth0
Wired connection 1  027a1c2b-3397-37fb-a6e2-f2e02eb59992   ethernet  eth1
```

If there is no connection for the interface yet, check if the device is visible using the command **nmcli dev status** or **ip link show**.

2. For an existing connection:

   a) Configure an appropriate name for the connection if required:

```
# nmcli conn mod "Wired connection 1" con-name eth1
```

   b) Set the IP configuration such that no IP address is assigned:

```
# nmcli conn mod eth1 ipv4.method "disabled" ipv6.method "disabled"
```

   c) Configure automatic interface activation at boot:

```
# nmcli conn mod eth1 connection.autoconnect yes
```

3. **If no connection for the second interface exists**:

   Add a new connection (with automatic interface activation, without IP address):

```
# nmcli conn add con-name eth1 type ethernet ifname eth1 autoconnect yes ipv4.method "disabled" ipv6.method "disabled"
```

4. (Re-)Activate the connection (this command may time out if IP connection check is enabled):

```
# nmcli con up eth1
```

Using the Charon Manager

The Charon-SSP Manager provides basic network configuration options.

- To access them, start the Charon Manager and open the menu option:
  Tools > Network Settings
- To configure a host system for use by the emulator perform the following steps:
  - Select the correct interface.
  - In the **IP setting** field select **None**.

- Click on **Apply**.

Please note:

- For RHEL 7 and derivatives, the interfaces to be managed by the Charon Manager must be removed from NetworkManager control and an ifcfg-file must exist.
- For RHEL 8 and later (and derivatives), the interfaces to be managed by the Charon Manager must be under NetworkManager control.

Step 2: Add the Dedicated NIC to the Emulator Configuration

Please refer to the documentation of your Charon emulator product for information on how to adapt the emulator configuration. The basic steps to be performed are the following:

- For Charon-SSP: start the Charon Manager and open the configuration window for the emulated system.
- For other emulator products: open the configuration file with a text editor.
- Configure the emulated system with the dedicated Ethernet interface as its interface.
- If your emulator product does not do so automatically, set the MAC address to the same value as used by the host interface (the value assigned by your cloud provider).
- Save your configuration.

Step 3: Configure the Guest system to Use the Private Cloud IP Address

Please refer to the documentation of your guest operating system for detailed information on how to configure networking on your guest system. Below is a basic example for Solaris.

Solaris Example

Using the steps below, the Solaris guest system is configured to use the second NIC configured on the host system (please refer to your Solaris documentation for configuration details).

1. Boot Solaris and configure the IP address assigned to the dedicated guest NIC for the Solaris Ethernet interface as shown in the examples below:
   # ifconfig *<interface-name> <private-guest-nic-ip>/<netmask* **up** (Solaris 10 example)
   or
   # ifconfig *<interface-name> <private-guest-nic-ip>* **netmask**<*mask*> **up** (Solaris 2.6 example)
   or
   # ipadm create-ip netX and **ipadm create-addr -T static -a** *<private-guest-nic-ip>*/*<netmask>* **net**X/**v4** (Solaris 11 example)
   For Solaris versions before version 11, make permanent by editing **/etc/hosts** and set the new address for the systems hostname. Then edit **/etc/netmask** and add the netmask for the subnet-network.

2. Add default route on Solaris:
   # route add default *<default-gateway-of-cloud-lan> <metric>*
   Make permanent by editing **/etc/defaultrouter** and add the address of the gateway (use route -p for newer Solaris versions).

3. Add DNS server to Solaris (if needed)

   a. Edit **/etc/resolv.conf** and add a nameserver line for the DNS server.
   b. Make sure, DNS is used for hostname translation: ensure that **/etc/nsswitch.conf** is configured to allow **dns** (in addition to **files**) for the hostname resolution.

   For Solaris 11, please refer to the Oracle Solaris documentation.

Expected result (depending on security rules and firewalls):

1. The guest system should be able to communicate with the host system across the cloud LAN using the private IP addresses.
2. The guest system should be able to communicate directly with the Internet if the dedicated NIC has a public IP address (not recommended).

Please note: Do not forget that traffic transmitted across the Internet by the guest system is not encrypted by default. Take appropriate measures to protect your data. It is strongly recommended to protect the Solaris guest system by an appropriate firewall and security group configuration. If possible, any communication across the Internet should be encrypted (e.g., by using a VPN).

Potential Problems for Guest Systems in Azure

The Azure cloud LAN default gateway does not respond to pings. This may cause problems for guest operating systems that use ICMP as a mechanism of "Dead Gateway Detection".

On example for this is HP-UX: once it marks the default gateway as unusable, any additional traffic that would require the default gateway fails.

Possible workaround for HP-UX:

- Check if dead gateway detection is active:
  # ndd -get /dev/ip ip_ire_gw_probe
  a **1** indicates that the feature is active
- Reboot and immediately disable dead gateway detection (before the first gateway check occurs):
  # ndd -set /dev/ip ip_ire_gw_probe 0
- Test connectivity.
- If the command solved the problem, you can make it permanent by editing **/etc/rc.config.d/nddconf** and adding the following lines:
  ```
  TRANSPORT_NAME[0]=ip
  NDD_NAME[0]=ip_ire_gw_probe
  NDD_VALUE[0]=0
  ```

Possible workaround for other guest operating systems (or if the above is not an option for HP-UX):

- Add static routes for the custom network routes to the guest OS in addition to the default route.
- The static routes should point to the correct next hop. In some cases, it is also possible to point them to the default gateway (tested on HP-UX 11.11).

Interface MTU Considerations

Contents

- Interface MTU Introduction
- Fragmentation
  - Possible Problems with IPv4 Fragmentation
    - Performance Problems
    - Connectivity Problems
- Implications for Charon Emulators and the Legacy Guest Systems

Interface MTU Introduction

The MTU (Maximum Transfer Unit) is the maximum packet size that can be transmitted across an interface. The MTU of an interface depends on the type of interface. This section only applies to Ethernet interfaces. The MTU value includes the IP and additional (e.g., TCP) protocol headers.

TCP/IP example for an Ethernet interface for which the default MTU is 1500:

```
1460 byte TCP application payload (Transport
layer)

+ 20 byte TCP header (Transport layer)

+ 20 byte IPv4 header (Routing layer)

------------------------------------------------

= 1500 byte MTU (Ethernet payload)

+ 14 byte Ethernet frame header (datalink layer)

+ 4 byte Frame Check Sequence (FCS)

------------------------------------------------

= 1518 byte Ethernet frame
```

The maximum size of the payload data varies depending on the transport layer protocol used. It can also be smaller as in the example above, for example, if a VPN is used which requires its own protocol informatin within the packet. In case of TCP/IP, the actual maximum payload size is also called MSS (Maximum Segment Size).

Fragmentation

The IP protocol includes the option to split up data packets into smaller packets (fragments) that are too big for the MTU of an interface. The recipient of the packet is responsible for reassembling the packets.

Possible Problems with IPv4 Fragmentation

The need to fragment IP packets can cause several problems. Some common problems are described below.

Performance Problems

Excessive fragmentation and reassembly causes additional network load due to more packets being transmitted. It also causes additional CPU load on systems that are required to fragment and reassemble packets. In addition, the loss of one fragment causes the retransmission of the whole packet. So in case of network instability, additional load is added to an already unstable network connection.

Connectivity Problems

If there are intermediate systems between two communication partners that have smaller MTUs than the two communication partners, data transmission may fail, for example in the following cases:

- If the intermediate systems do not allow fragmentation or block packet fragments.
- If the intermediate systems block the ICMP messages necessary for Path MTU Discovery.

For a TCP connections, the two communication partners come to an agreement regarding the MSS to be used based on their own interface MTU. If any intermediate systems have a smaller MTU but either the two communication partners don't know this (failed Path MTU Discovery) and the intermediate systems cannot fragment the data packets, any packet that is too large will be dropped.

For protocols without the ability to negotiate the payload size or cannot handle fragmented traffic, data will be lost as soon as a packet is larger than the smallest MTU on the path.

Example with two ICMP data sizes across a DSL connection that is limited to a MTU of 1492. The IP + ICMP headers are 28 bytes in total. In the example, fragmentation is disabled (`-M do`). The source system has an MTU of 1500 bytes.

```
$ ping -c 1 -s $((1493-28)) -M do www.stromasys.com
PING stromasys.com (192.124.249.190) 1465(1493) bytes of data.

--- stromasys.com ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms



$ ping -c 1 -s $((1492-28)) -M do www.stromasys.com
PING stromasys.com (192.124.249.190) 1464(1492) bytes of data.
1472 bytes from cloudproxy10190.sucuri.net (192.124.249.190): icmp_seq=1 ttl=58 time=21.6 ms
```

The first attempt with an attempted size of 1493 bytes fails, the second with a packet size of 1492 succeeds.

Implications for Charon Emulators and the Legacy Guest Systems

When dedicating a Charon host NIC to the emulator for use by the legacy guest system, **ensure that the MTU of the dedicated NIC is not smaller than the MTU interface visible in the guest system**.

Typically, legacy guest systems, such as Solaris, HP-UX, Tru64, and OpenVMS have a default MTU of 1500 bytes.

Physical Charon hosts usually also have a default MTU of 1500 bytes.

However, in cloud environments, the default MTU of Ethernet interfaces configured for a Charon host can vary. The following table provides an overview:

Platform	Default MTU size	Comment
Legacy OS	1500	
AWS	9001	Jumbo frames are not supported by Charon emulators (Charon-SSP has implemented a workaround). Therefore, all end-systems involved in the communication with the legacy guest system should use an MTU no larger than 1500.
Azure	1500	
GCP	1460	A different MTU size can be defined when creating the VPC to be used for the instance.
OCI	9000	Jumbo frames are not supported by Charon emulators (Charon-SSP has implemented a workaround). Therefore, all end-systems involved in the communication with the legacy guest system should use an MTU no larger than 1500.
IBM	1500	

Caveat: ensuring that the Charon host MTU is not smaller than the guest system MTU prevents problems arising at the emulator host. However, it cannot prevent problems arising from smaller MTU sizes along the complete communication path if fragmentation does not work properly. Therefore, it is recommended to identify the actual MTU, for example, of the path across a VPN between cloud and on-premises networks and to either ensure that fragmentation and path discovery work correctly, or adapt the MTU of the legacy guest system accordingly.

Next Steps

Once you have set up your Charon instance in the cloud, please proceed to the general *Charon emulator product User's Guide* for your Charon product (see Stromasys documentation site) and the VE License Server User's Guide for more information about configuring and managing your Charon emulator product.