

## Contents

Lab topology and components	3
Lab topology	3
Lab components	5
Planning lab deployment	7
On LabHost PC:	11
Deploying the VyOS router	14
On LabHost PC:	14
Creating Check Point Base VM	20
On LabHost PC:	20
Creating Windows Server 2019 Standard Base VM	26
On LabHost PC:	26
On WINBASE VM:	27
On LabHost PC:	27
Preparing cloned Windows hosts VMs	27
Preparing Active Directory and the domain controller VM	29
Preparing cloned Check Point hosts	31
Preparing SmartConsole VM	32
Summary	34

# Check Point Lab Rapid Deployment Guide (replacing Chapters 3 and 4)

In this guide, we will create our **Check Point** virtual lab environment. If you are studying for your **Check Point Certified Security Administrator (CCSA)** certification using **Check Point's Authorized Training Center**, all the non-Check Point components in the labs are provisioned for you in advance. In our case, we'll have to build the lab from scratch. This should not take long if you follow the provided instructions. The upside of building the lab is that you're likely to gain a deeper understanding of the interaction between the different infrastructure components.

In this chapter, we're going to cover the following main topics:

- Lab topology and components
- Planning lab deployment
- Downloading the prerequisites
- Installing Oracle VirtualBox and VirtualBox Extension Pack
- Deploying the VyOS router
- Creating Windows and Check Point base VMs
- Creating linked clones of Windows and Check Point Base VMs
- Completing initial configuration of individual Windows and Check Point VMs

Technical requirements

For this chapter, you will need a PC with at least 4 CPU cores(eight with hyperthreading), 32 GB of RAM, 200 GB of HDD space (preferably SSD), and a monitor with at least a 1,920 x 1,080 resolution running **Windows 10**. References to the software used in the lab are provided in later sections. You will also need **Google Chrome** or **Microsoft Edge** as a browser and an SSH terminal emulator.

**Oracle VirtualBox** has been chosen because it is free for personal use, can run on either **Windows**, **Linux**, or **macOS**, and is a capable platform. My personal production labs are built on **VMware** products, but those could either be expensive or require dedicated hardware. The *LabHost* could be running either Windows or any OS supported by Oracle VirtualBox as a host operating system. A full list of compatible host operating systems can be found here:

<https://www.virtualbox.org/wiki/Downloads> in **VirtualBox platform Packages** section.

#### Important Note

If you are experienced with (and have access to) different virtualization platforms, such as **VMware Workstation**, **Fusion**, **ESXi**, **Microsoft Hyper-V**, **Linux KVM**, or **EVE-NG**, you can adapt the lab to any of these with minor modifications and platform-specific scripting.

This chapter (and the following one) takes less than two hours to go through all the steps to have the lab implemented.

## Lab topology and components

Let's start by looking at the topology of the lab, identifying its components, and discussing the sequence in which they are going to be deployed.

### Lab topology

Please note that to keep the topology diagram more readable, the IP addresses of the cluster and its members will be depicted as follows:

	<b>10.0.0.1/24 Virtual IP of the Cluster's interface</b>
10.0.0.1(2,3)/24 =	<b>10.0.0.2/24 IP of the Cluster Member 1 interface</b>
	<b>10.0.0.3/24 IP of the Cluster Member 2 interface</b>

Figure 3.1 – Cluster IP address notation abbreviation

The following diagram (*Figure 3.2*) shows the lab topology, its constituent components, and their IP addresses:

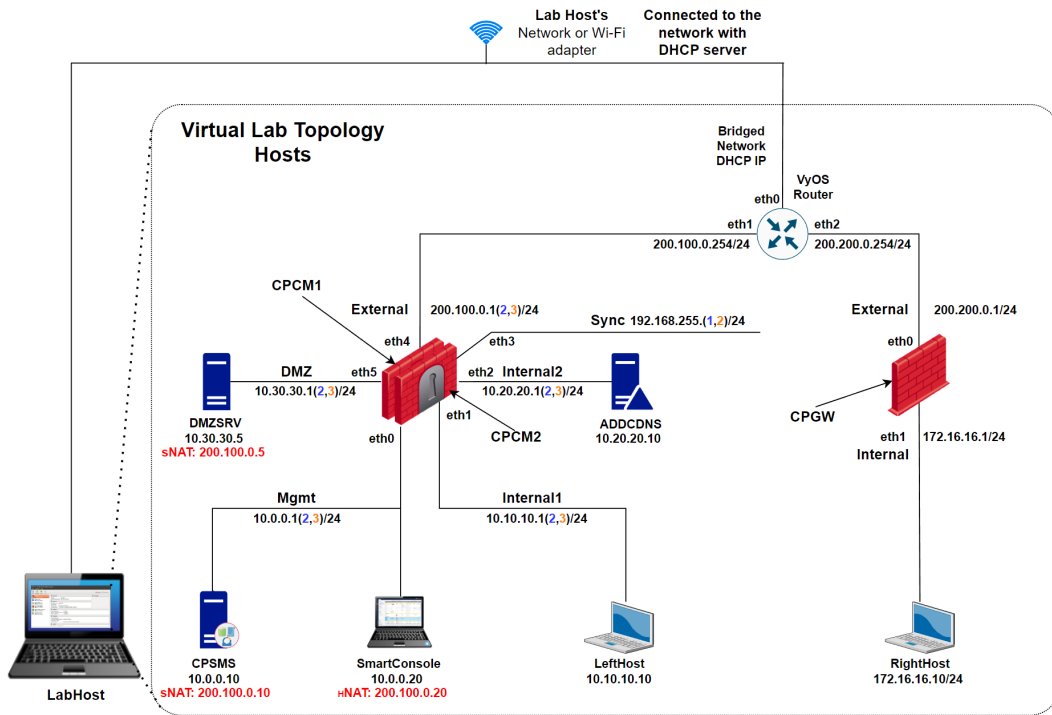


Figure 3.2 – Lab topology, components, and IP addresses

*LabHost* is the PC on which you are about to create your lab. It does not have to be renamed, as it is used for references only.

Since we are creating virtual networking, we will be relying on consistently named networks. The network names are shown in the following diagram (*Net\_<IP>*):

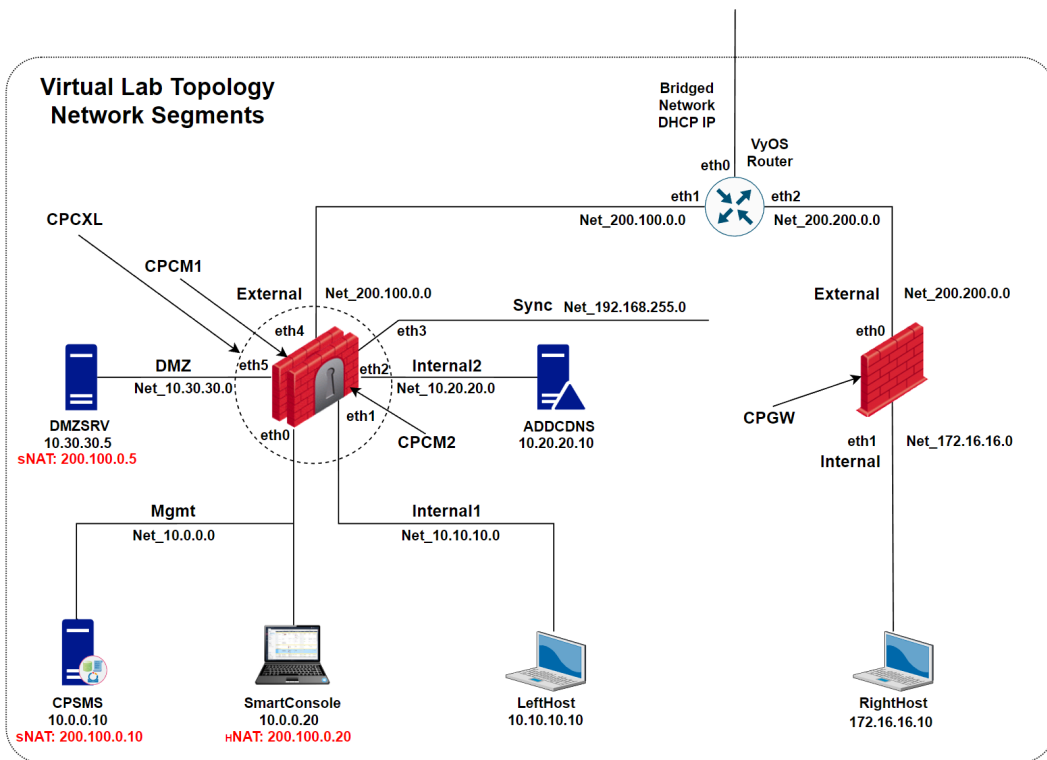


Figure 3.3 – Topology and virtual network segments

## Lab components

For easy reference, *Figure 3.4* shows a table of the lab components, the IP addresses of their interfaces, and the network segments they are connected to:

VyOS Router					
Interface	IP		Description	Network	
eth0	DHCP		Outside	Bridged Adapter	
eth1	200.100.0.254		Left	Net_200.100.0.0	
eth2	200.200.0.254		Right	Net_200.200.0.0	

Cluster (VirtualIPs, CPCM1 and CPCM2)					
Interface	Virtual IP	CPCM1	CPCM2	Description	Network
eth0	10.0.0.1	10.0.0.2	10.0.0.3	Mgmt	Net_10.0.0.0
eth1	10.10.10.1	10.10.10.2	10.10.10.3	Internal1	Net_10.10.10.0
eth2	10.20.20.1	10.20.20.2	10.20.20.3	Internal2	Net_10.20.20.0
eth3		192.168.255.1	192.168.255.2	Sync	Net_192.168.255.0
eth4	200.100.0.1	200.100.0.2	200.100.0.3	External	Net_200.100.0.0
eth5	10.30.30.1	10.30.30.2	10.30.30.3	DMZ	Net_10.30.30.0

Single Gateway (CPGW)			
Interface	IP		Network
eth0	200.200.0.1		Net_200.200.0.0
eth1	172.16.16.1		Net_172.16.16.0

Security Management Server (CPSMS)			
Interface	IP		Network
eth0	10.0.0.10		Net_10.0.0.0

SmartConsole VM		
Interface	IP	
Ethernet	10.0.0.20	
	Net_10.0.0.0	

LeftHost		
Interface	IP	
Ethernet	10.10.10.10	
	Net_10.10.10.0	

ADDCDNS		
Interface	IP	
Ethernet	10.20.20.10	
	Net_10.20.20.0	

DMZSRV		
Interface	IP	
Ethernet	10.30.30.10	
	Net_10.30.30.0	

RightHost		
Interface	IP	
Ethernet	172.16.16.10	
	Net_172.16.16.0	

Figure 3.4 – Hosts, interfaces, IP addresses, and networks

Please note that all subnet masks in the lab are /24 (or 255 . 255 . 255 . 0).

Now that we have seen the topology of the lab, we can move on to obtaining the software.

# Planning lab deployment

Since our lab is comprised of 10 virtual machines, PCs even those dedicated solely to the lab and equipped with 32 GB of RAM may, at times, experience difficulties running all of them. We should examine our current baseline CPU and RAM utilization to determine what resources are available to us.

To do that, on your *LabHost PC*, open **Task Manager** [1], click on **Performance** tab [2], and click on **Memory** option [3]. Note the values displayed in **In use (compressed)** and in **Available** fields [4]:

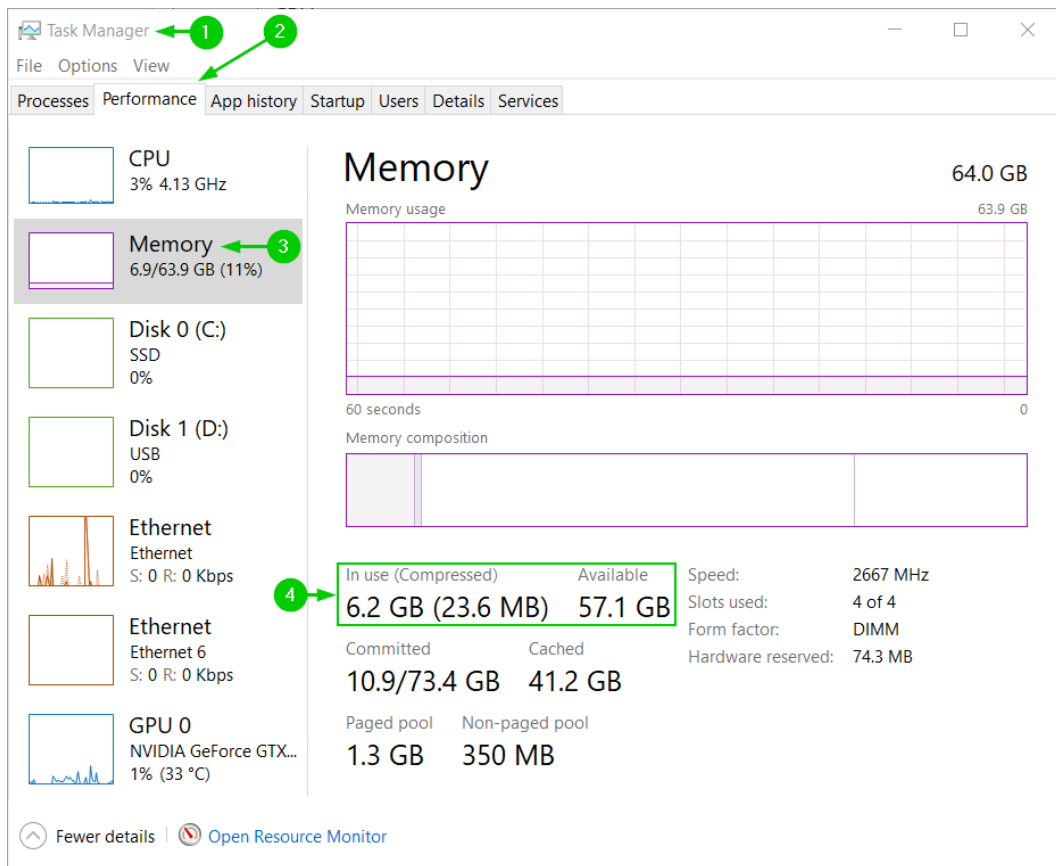


Figure 3.5 – LabHost PC resources and baseline utilization (64GB RAM LabHost example)

The difference between **Available** and **In use (compressed)** is the RAM available for your labs' concurrently running VMs.

Depending on our findings, keep those VMs that are not needed for specific lab exercises, powered down. For instance, *CPGW* and *RightHost* VMs are not needed until *Chapter 12*, so there is no reason to strain your LabHost's resources until then.

Resources necessary for deployment in production environments are described in Check Point versions release notes. For our lab, we may skimp on storage, but not on memory or CPU cores.

No.	VM Name	RAM			vCPU Cores
		Minimum	Recommended	Lab Script	
1	Router	1	2	1	1
2	LeftHost	1	2	2	2
3	RightHost	1	2	2	2
4	DMZSRV	1	2	2	2
5	ADDCDNS	1	2	2	2
6	SmartConsole	4	4	4	2
7	CPSMS	6*	8	6	4
8	CPCM1	4	4	4	4
9	CPCM2	4	4	4	4
10	CPGW	4	4	4	2
Total		27	34	31	Immaterial**

Figure 3.6 - LabHost Memory and Compute resources for the lab

\* 6GB RAM for *Management Server* is allowed according to *Check Point sk104848*:  
[https://supportcenter.checkpoint.com/supportcenter/portal?eventSubmit\\_doGoviewsolutiondetails=&solutionid=sk104848](https://supportcenter.checkpoint.com/supportcenter/portal?eventSubmit_doGoviewsolutiondetails=&solutionid=sk104848)

\*\*vCPU cores are allocated from the pool of total physical (including hyperthreaded) cores of the LabHost PC. So long as you are running any relatively modern quad core (or higher), **Intel** or **AMD** CPU with hyperthreading, 8 cores (or more) are shared among all the VMs.

Note, that the Total values listed for RAM are *IN ADDITION* to the memory consumed by LabHost itself, (The values we have seen in the baseline *Figure 3.5*). Assuming that, at the very least, on your LabHost with 32 GB of RAM you are running:

- **VirtualBox**
- **Adobe PDF reader** (to use the electronic version of the book for references)
- A **browser** instance
- **PuTTY**
- **NotePad++**



You will either have to stick to the values listed in the Minimum RAM column of the table depicted in *Figure 3.6* or to selectively shutdown VMs that are not needed for specific chapters and exercises. Of course, if you have sufficient RAM to allocate the values in **Recommended** column, do so.

Initially, RAM will be allocated to VMs according to the values listed in the **Lab Script** column of the table depicted in the *Figure 3.6*. If you need to adjust those, after completion of the specific VM configuration, shut it down, change RAM parameters in VirtualBox VM settings and power it up again.

**Important Notes:**

Do NOT allocate more vCPUs than listed in the vCPU Cores column- hypervisors scheduling execution of multithreaded applications on all (or most) cores assigned to VMs. This means that the VM with more cores may have to wait longer for corresponding number of physical cores on LabHost to become idle to complete the execution of transactions.

I am changing the HDD space allocation to CPBASE from 60 GB, used in the book, to 80 GB in the `create_CPBASE.bat` script. This will not impact overall lab space requirements of 200 GB, as we are going to utilize a **Linked Clones** capability of VirtualBox that will create delta disks for individual VMs from common Windows and Check Point **Virtual Disk Images (VDIs)**.

In the interest of accomplishing this lab build in as little time as possible, we'll be relying, for the most part, on scripts.

I encourage you to examine each of the `.bat` and `.ps1` scripts downloaded from the book's **GitHub** repository before execution, to see what they are doing.

There are two groups of scripts:

First, the batch files invoking `VBoxManage.exe` to create, clone and modify virtual machines.

Second, the **batch files** invoking corresponding **PowerShell** scripts, while bypassing default **Windows Execution Policy** to perform such actions as `sysprep`, rename hosts, assign their IP addresses, and configure VM-specific features and services.

When necessary, we'll be reverting to the GUI of applications and interacting with VMs' console sessions to accomplish tasks that require user input.

**Important**

Follow the process described in this chapter to complete the lab build in under two hours. Please pay close attention to subsequent section headings indicating on which machine- LabHost or a specific VM, particular scripts should be executed.

This is the plan we are going to follow:

1. Create nested folders for the lab resources.
2. Download prerequisite software and ISO images, copy them to appropriate folders.
3. Download the scripts from this book's GitHub repository and move them to appropriate folders.
4. Install prerequisites on LabHost PC.
5. Using script, VirtualBox console and SSH, install and configure the router VM.
6. Using scripts and VirtualBox console, create and configure the Check Point **CPBASE** VM.
7. Using scripts and VirtualBox console, create and configure the Windows **WINBASE** VM:
  - A. Using script, replace `win_nt6_unattended.xml` with the file from GitHub repository.
  - B. Using script, create WINBASE VM with pre-installed Windows Server.
  - C. Using VirtualBox console Install additional software on WINBASE VM.
  - D. Using script, perform sysprep on WINBASE VM.
8. Using script, eject installation media and **create snapshots** of CPBASE and WINBASE VMs.
9. Using script, **clone** and modify CPBASE and WINBASE snapshots to create the rest of the lab.
10. Using VirtualBox consoles, complete configuration of individual Windows VMs by executing VM-specific scripts and changing default browser to Chrome.
11. Using VirtualBox consoles, configure few initial parameters on individual Check Point VMs.

## On LabHost PC:

Create CPBook folder with nested folders by copying and pasting following lines in your Windows CMD prompt:

```
mkdir C:\CPBook
mkdir C:\CPBook\LabShare
mkdir C:\CPBook\LabShare\ISOs_and_OVAs
mkdir C:\CPBook\LabShare\Software
mkdir C:\CPBook\LabShare\Scripts
```

1. Download and move prerequisite ISOs to C:\CPBook\LabShare\ISOs\_and\_OVAs:
  - A. Windows Server 2019 Evaluation:
  - B. <https://info.microsoft.com/ww-landing-windows-server-2019.html>
  - C. Check Point Recommended release R81.10 (this URL is a one-liner):  
[https://supportcenter.checkpoint.com/supportcenter/portal/role/supportcenterUser/page/default.psml/media-type/html?action=portlets.DCFileAction&eventSubmit\\_doGetdcetails=&fileid=115148](https://supportcenter.checkpoint.com/supportcenter/portal/role/supportcenterUser/page/default.psml/media-type/html?action=portlets.DCFileAction&eventSubmit_doGetdcetails=&fileid=115148)
  - D. VyOS Router:  
<https://vyos.net/get/nightly-builds/>
2. Download and move prerequisite software to C:\CPBook\LabShare\Software:
  - A. **Check Point SmartConsole** (this URL is a one-liner):  
[https://supportcenter.checkpoint.com/supportcenter/portal/role/supportcenterUser/page/default.psml/media-type/html?action=portlets.DCFileAction&eventSubmit\\_doGetdcetails=&fileid=126445](https://supportcenter.checkpoint.com/supportcenter/portal/role/supportcenterUser/page/default.psml/media-type/html?action=portlets.DCFileAction&eventSubmit_doGetdcetails=&fileid=126445)
  - B. PuTTY Terminal Emulator:  
<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.78-installer.msi>
  - C. **NotePad++**:  
<https://notepad-plus-plus.org/downloads/>

D. **WinSCP:**

<https://winscp.net/eng/download.php>

E. **Google Chrome offline installer** (uncheck **Help make Google Chrome better...** option):

<https://www.google.com/intl/en/chrome/?standalone=1>

3. Download and move scripts from **GitHub** (this URL is a one-liner): <https://github.com/PacktPublishing/Check-Point-Firewall-Administration-R81.10-/tree/main/Chapter04/> to C:\CPBook\LabShare\Scripts.
4. Download and move **VirtualBox**, **VirtualBox Extension Pack** and **Microsoft Visual C++ X64 redistributable** package to C:\CPBook\LabShare:
  - A. **VirtualBox platform package** (for Windows hosts) and **VirtualBox Extension Pack (All supported platforms)**: <https://www.virtualbox.org/wiki/Downloads>
  - B. **MS Visual C++ redistributable**: <https://learn.microsoft.com/en-US/cpp/windows/latest-supported-vc-redist?view=msvc-170> (and chose X64 download link).
5. Install following software on LabHost PC
  - **PuTTY**
  - **WinSCP**
  - **NotePad++**
  - **VirtualBox:**
    - If prompted that **MS C++** is required, click **OK**, click **Finish**, click **OK**, to abort the installation and proceed to *Step 2* below. If you do not see this warning, you already have **MS C++** installed as a prerequisite to some other application running on your LabHost PC. In this case, skip *Step 2* and continue with *Step 4*.
    - Install **MS C++** by double-clicking **VC\_redist.x64.exe**. In **Open File – Security Warning** window, click **Run**. Accept **MS EULA** and click **Install**. Click **Close**.
    - Rerun **VirtualBox** installation file.
    - On **Custom Setup** screen, click on **VirtualBox Python Support** and click on red X **Entire feature will be unavailable**.

- When prompted with **Warning: Network Interfaces**, save your work in any open session of stateful applications before clicking **Next**.
- **VirtualBox Extension Pack** (becomes executable, after VirtualBox is installed)

## 6. Start VirtualBox

# Deploying the VyOS router

In our lab, the VyOS router will act as both the internet simulator (by providing routing between the left and the right sides of our setup) and the internet gateway (by routing outbound traffic to the destinations outside of our lab environment).

It will be configured with three interfaces. One, connected to the internal bridge on your virtualization LabHost, is configured to obtain the IP address and the default gateway (via **DHCP**) on your physical network. The second and third interfaces are going to be defined as internal interfaces to your LabHost, connected to the external interfaces of the Check Point cluster (on the left) and a single gateway (on the right).

## On LabHost PC:

Open Windows command prompt. From it, change directory to `C:\CPBook\LabShare\Scripts` folder and execute `create_router.bat`.

1. In Oracle VM, select newly appeared **router** VM and click on **Settings**:

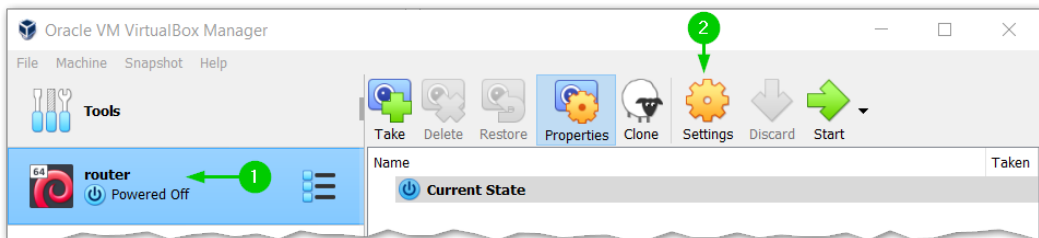


Figure 3.7 – router VM Settings

2. In the **router - Settings** options, select **Network** [1]. In **Adapter 1** tab [2], change the value of the **Attached to:** field to **Bridged Adapter** [3], using drop-down menu. Click on **Advanced** icon [4], to expand the view and verify that **Cable Connected** checkbox is checked [5]. Click **OK** [6]:

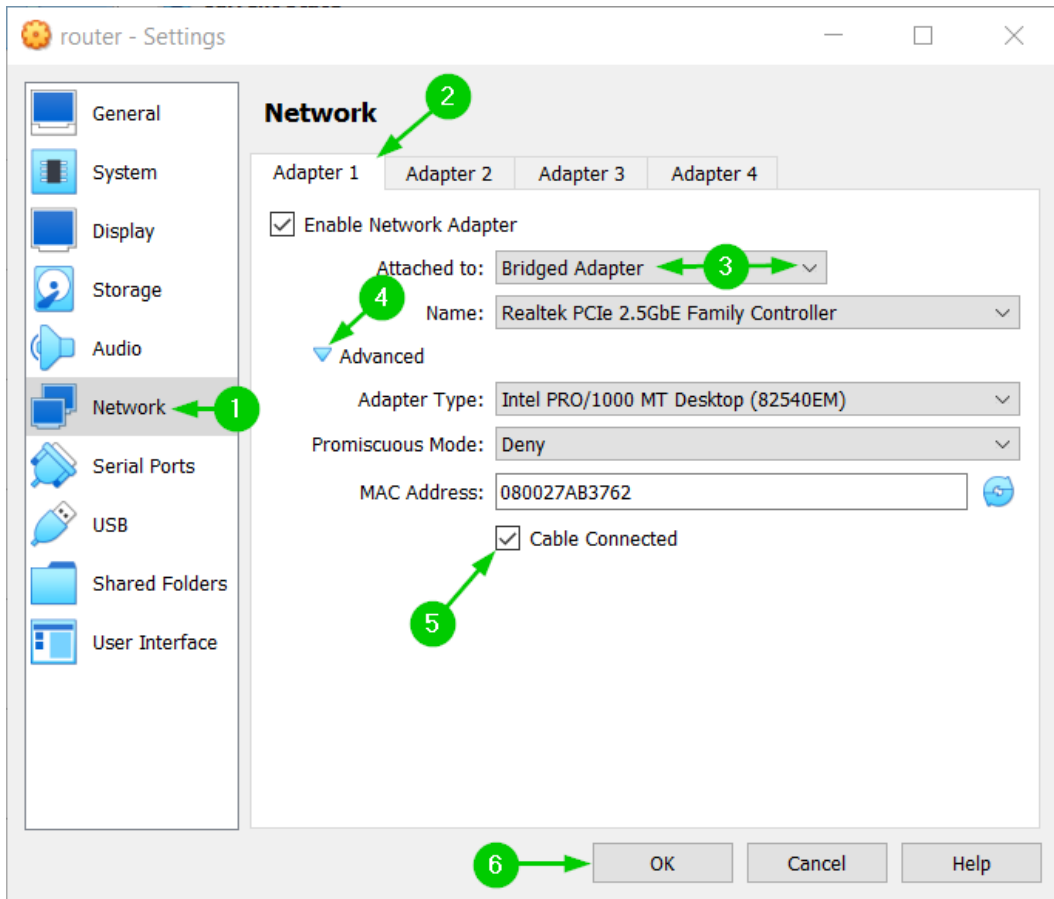


Figure 3.8 – Configuring router VM Network Adapter 1

3. In **Oracle VM VirtualBox Manager**, select **router** VM [1], and click **Start** [2]:

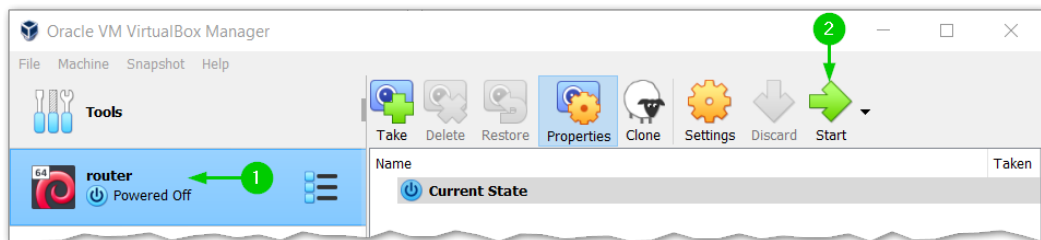


Figure 3.9 – Starting router VM

4. Select **router [Running]** virtual console window [1] and click once inside, to evoke keyboard capture by VM. The **Live** boot process [2] (from mounted ISO) will proceed automatically [3]:

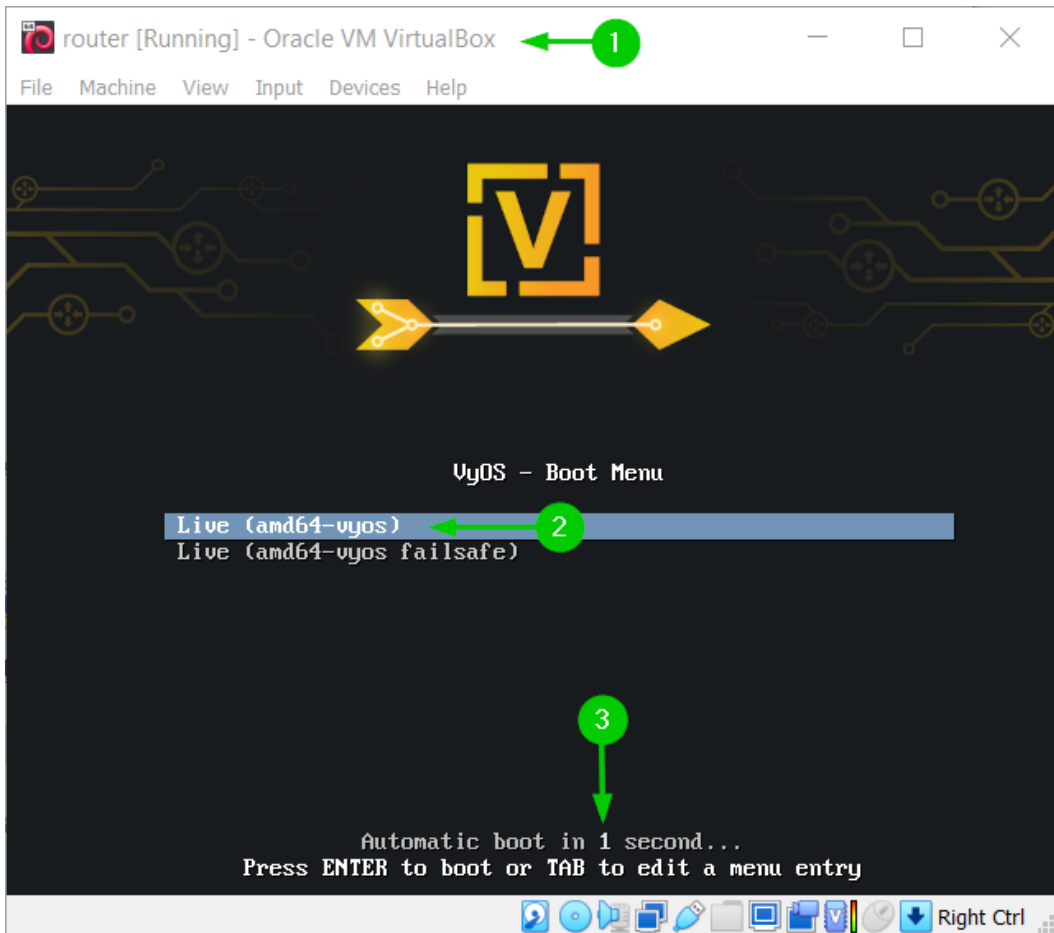


Figure 3.10 – router VM Live boot process

5. When boot process is finished, at **vyos** login: prompt type **vyos** for both, a user name and a password [1]:

```

router [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
[ OK ] Starting User Login Management...
[ OK ] Started VyOS Router.
[ OK ] Starting Permit User Sessions...
[ OK ] Finished arpwatck service.
[ OK ] Started Deferred execution scheduler.
[ OK ] Started Atop process accounting daemon.
[ OK ] Finished OpenBSD Secure Shell session cleanup.
[ OK ] Started Atop advanced performance monitor.
[ OK ] Finished Permit User Sessions.
[ OK ] Started Getty on tty1.
[ OK ] Reached target Login Prompts.
[ 7.198998] vyos-router[574]: Waiting for NICs to settle down: settled in 0.5s
[ OK ] Started System Logging Service.
[ OK ] Started User Login Management.
[ OK ] Started LSB: Brings up/down network automatically.
[ 9.726634] vyos-router[654]: Started watchfrr.
[ 12.761172] vyos-router[574]: Mounting VyOS Config...done.
[ 18.308717] vyos-router[574]: Starting VyOS router: migrate configure.
[ 18.699054] vyos-config[588]: Configuration success

Welcome to VyOS - vyos tty1

vyos login: vyos
Password: _

```

Figure 3.11 – VyOS router VM log in

6. At **vyos@vyos:** prompt, type `install image` and press *Enter* [1]. Accept default values throughout the installation process by pressing *Enter* [2]. When prompted with the warning **This will destroy all data on /dev/sda**, type `Yes` and press *Enter* [3]. When prompted for password and password confirmation, type `vyos` at both prompts [4]. After seeing **Done! Please reboot now**, at **vyos@vyos:~\$** prompt, type `reboot` and press *Enter* [5], type `Y` and press *Enter* (not shown):



```

vyos@vyos:~$ install image ← 1
Welcome to the VyOS install program. This script
will walk you through the process of installing the
VyOS image to a local hard drive.
Would you like to continue? (Yes/No) [Yes]: Enter ← 2
Probing drives: OK
The VyOS image will require a minimum 2000MB root.
Would you like me to try to partition a drive automatically
or would you rather partition it manually with parted? If
you have already setup your partitions, you may skip this step

Partition (Auto/Parted/Skip) [Auto]: Enter ←

I found the following drives on your system:
sda      8589MB

Install the image on? [sda]: Enter ←

This will destroy all data on /dev/sda.
Continue? (Yes/No) [No]: Yes ← 3

Looking for pre-existing RAID groups...none found.
How big of a root partition should I create? (2000MB - 8589MB) [8589]MB: Enter ↓

Creating filesystem on /dev/sda1: OK
Done!
Mounting /dev/sda1...
What would you like to name this image? [1.4-rolling-202207011759]: Enter ←
OK. This image will be named: 1.4-rolling-202207011759
Copying squashfs image...
Copying kernel and initrd images...
Done!
I found the following configuration files:
/opt/vyatta/etc/config/config.boot
/opt/vyatta/etc/config/boot.default
Which one should I copy to sda? [/opt/vyatta/etc/config/config.boot]: Enter ←

Copying /opt/vyatta/etc/config/config.boot to sda.
Enter password for administrator account
Enter password for user 'vyos': vyos Enter ← 4
Retype password for user 'vyos': vyos Enter ←
I need to install the GRUB boot loader.
I found the following drives on your system:
sda      8589MB

Which drive should GRUB modify the boot partition on? [sda]: Enter ←

Setting up grub: OK
Done! Please reboot now.
vyos@vyos:~$ reboot ← 5

```

Figure 3.12 – VyOS router image installation process

7. Once router VM rebooted, login again using vyos for username and the password.
8. Type configure and press Enter.

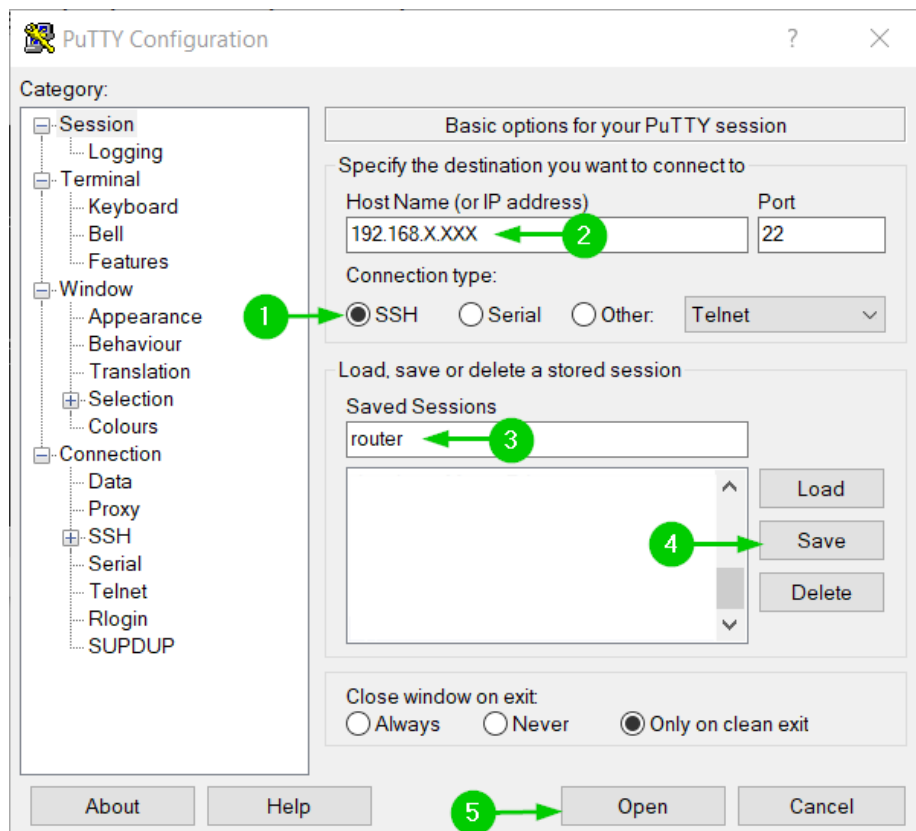
9. In configuration prompt (#), type following commands pressing *Enter* after each:

```
set interfaces ethernet eth0 address dhcp
set service ssh
commit
save
exit
```

10. At user prompt (\$) type `show interfaces ethernet eth0 brief` and press *Enter*

11. Note the IP Address assigned to the Interface **eth0**

12. On your *LabHost PC*, open **PuTTY** and create an **SSH** session [1] to the **IP address** noted in the previous step [2]. Name this session **router** [3] and click **Save** [4]. Click **Open** [5]:



13. In **PuTTY Security Alert** popup window [1], click **Accept** [2].
14. Enter `vyos` in both, **login as:** and the `vyos@<IP Address>`'s password: prompts.
15. Type `configure` and press *Enter*
16. From your LabHost PC's `C:\CPBook\LabShare\Scripts` folder, open the `VyOS_router_config_code_block.txt` file in your notepad. Copy the section of the code-block that follows the line **#At vyos@router# prompt, copy and paste this codeblock:** and paste it to the PuTTY session.

**Note**

If your time-zone is not US/Eastern, set it to the appropriate value.

17. After execution, type the following lines pressing *Enter* after each:

```
commit
```

```
save
```

```
exit
```

18. Test connectivity to the hosts on internet by running: `ping 9.9.9.9` and `ping www.yahoo.com`. Use *Ctrl + C* to abort.

Router VM is now fully configured. Let's move on to creating our Check Point Base VM.

## Creating Check Point Base VM

### On LabHost PC:

1. In Windows **CMD**, change directory to `C:\CPBook\LabShare\Scripts`
2. In `C:\CPBook\LabShare\Scripts>` prompt, execute `create_CPBASE.bat`
3. In each screen of Check Point initial configuration, use *Tab* key to select **OK**, keeping default values everywhere, except for **Management Interface configuration**. When prompted with **Management Interface configuration** screen, accept the default values for the IP address and the Subnet mask, but **DELETE** the value automatically populated for **Default Gateway** using *backspace* key:

- A. Click once inside **CPBASE [Running]** window [1], press the *Tab* key to select **Install Gaia on this system** [2], and press *Enter*:

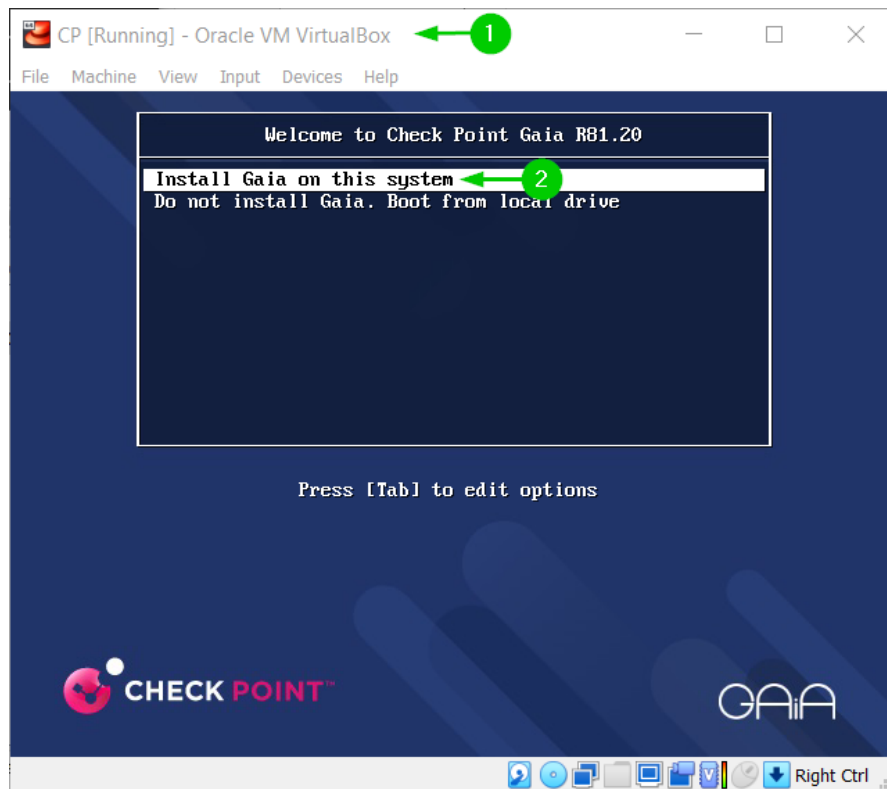


Figure 3.14 – Choosing to install Gaia

- B. In **Warning** window [1], *Tab* to select and click **OK** [2] to proceed with the installation:

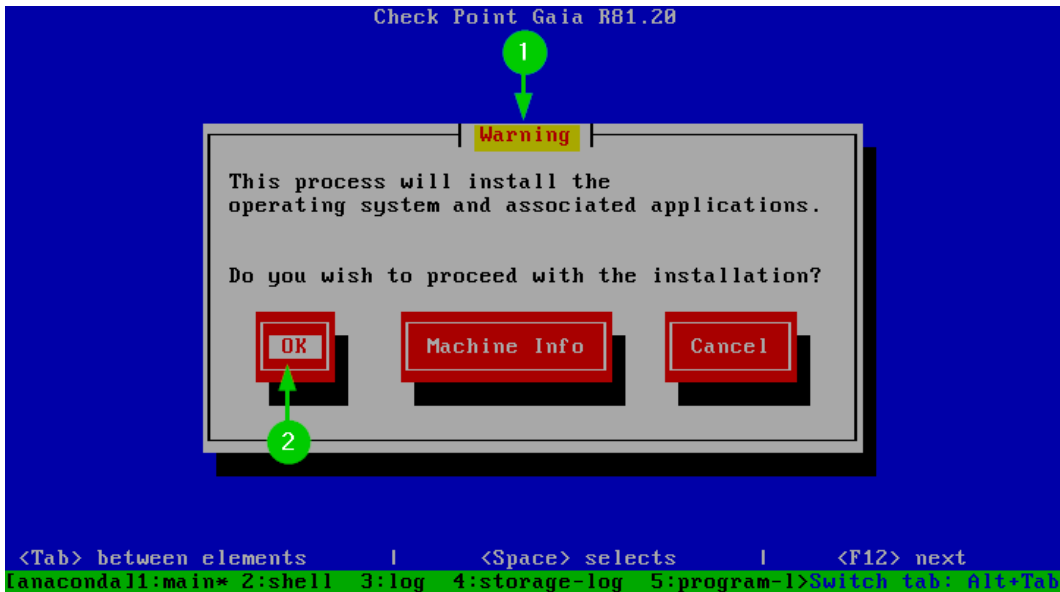


Figure 3.15 – Gaia installation Warning prompt

- C. In **Keyboard Selection** [1], choose your locale [2], *Tab* to select and click **OK** [3]:

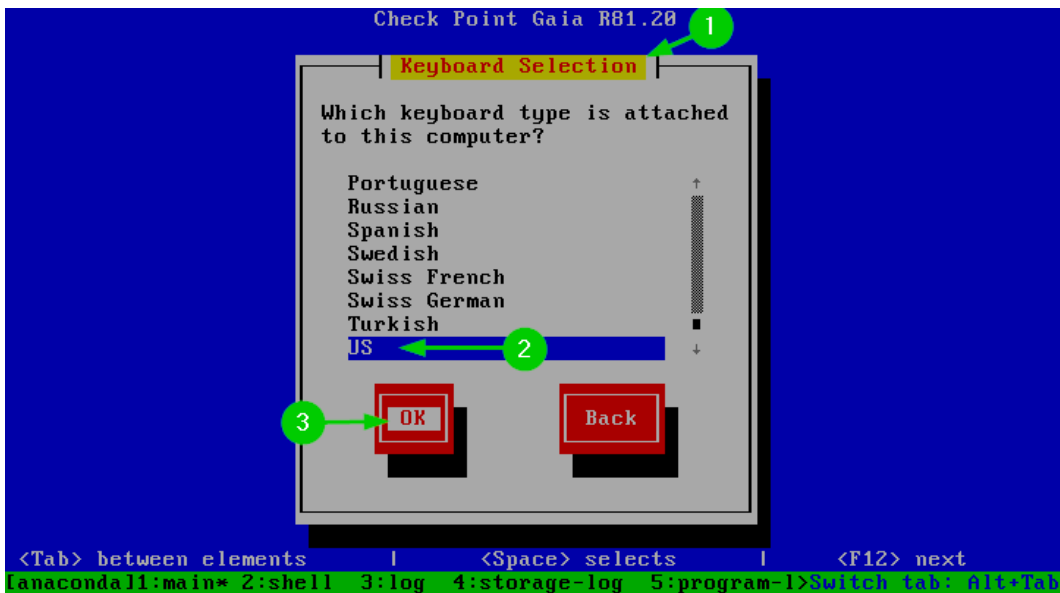


Figure 3.16 – Gaia installation Keyboard Selection prompt

- D. In **Partition Configuration** [1], keep the default values [2]. *Tab* to select and click **OK** [3]:

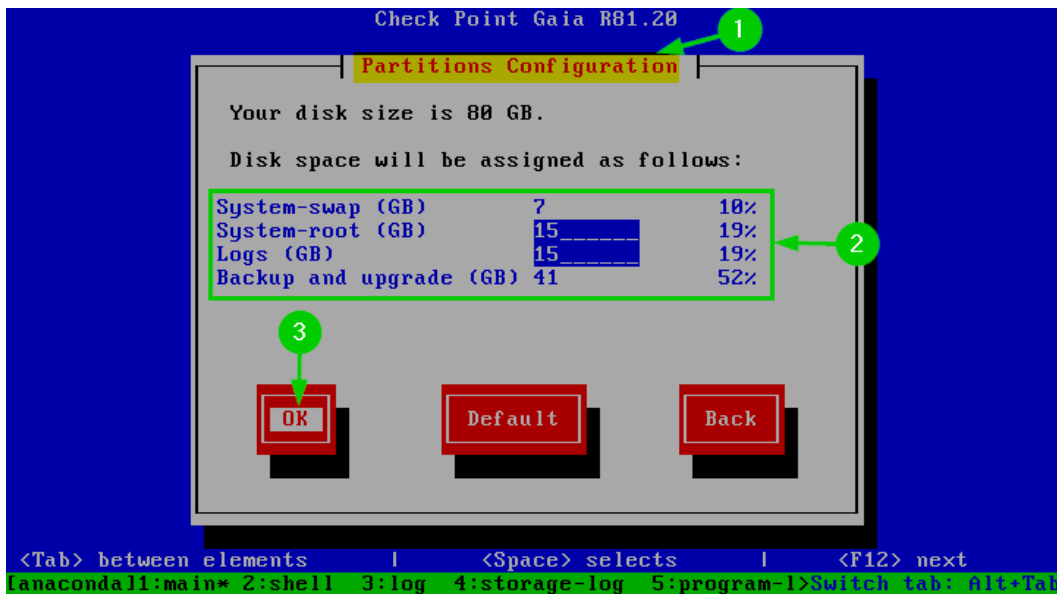


Figure 3.17 – Gaia installation Partition COnfiguration prompt

- E. In **Account Configuration** [1], enter our lab password, CPL@b8110, in both fields [2]. *Tab* to select and click **OK** [3]:

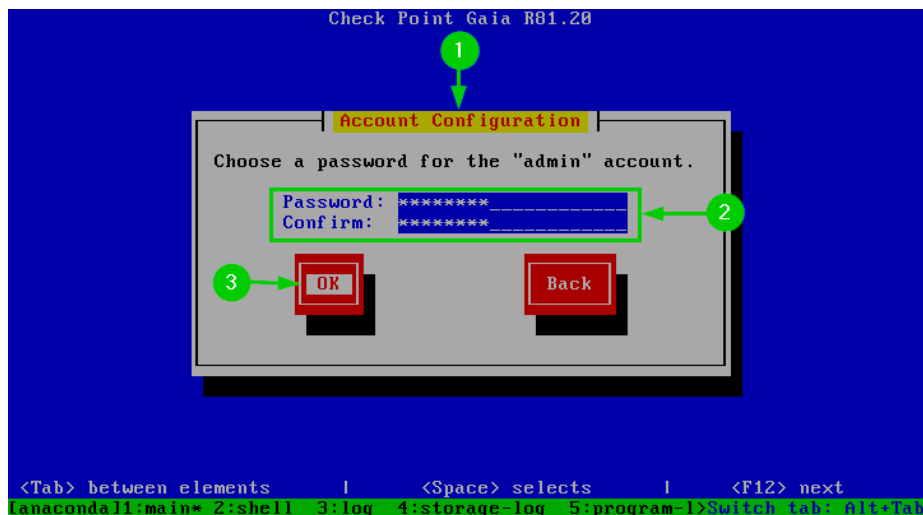


Figure3.18 – Gaia installation Account Configuration prompt

- F. For R81.20 or higher only, enter the **Maintenance password** [1], use the same CPL@b81110 [2]. *Tab* to select and click **OK** [3]:

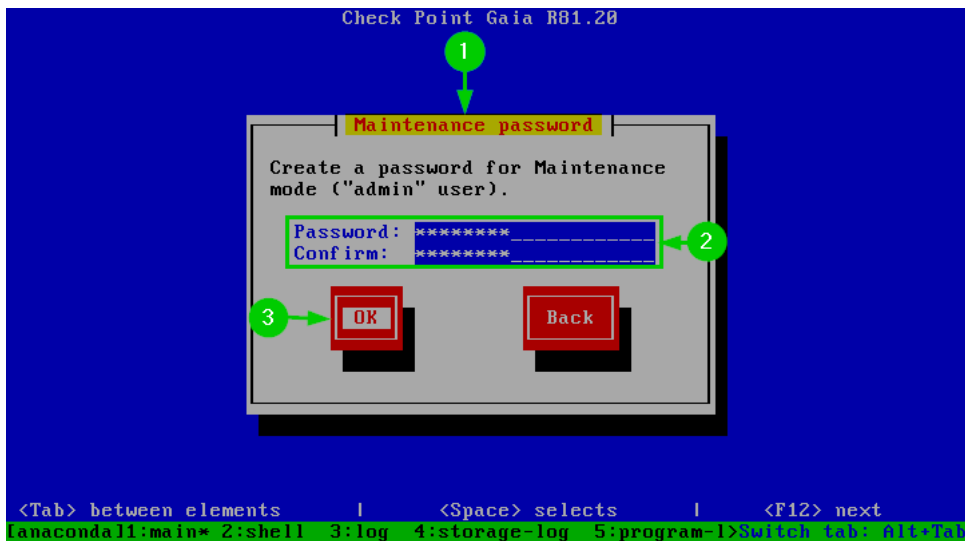


Figure3.19 – Gaia installation Maintenance password prompt (R81.20 or higher)

- G. For **Management Interface eth0** [1], keep the temporary **IP address** and **Netmask** [2] and delete the automatically populated entry for the **Default gateway** using *backspace* [3]. *Tab* to select and click **OK** [4]:

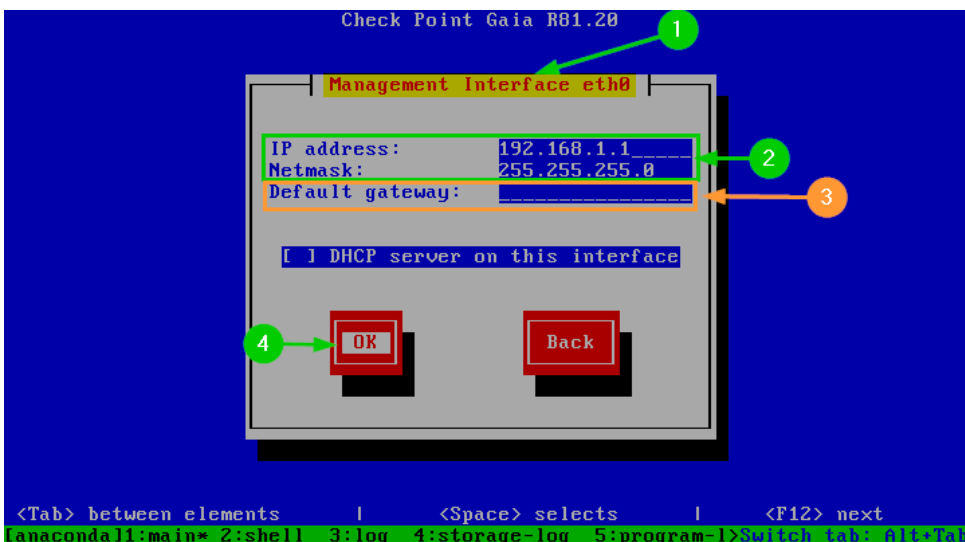


Figure 3.20 – Gaia installation Management Interface prompt

- H. In **Confirmation** window [1], *Tab* to select and click **OK** [2]:

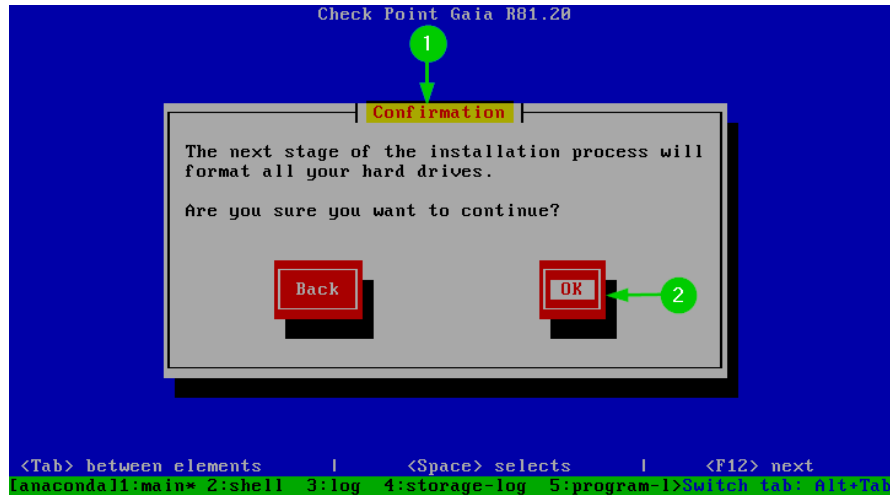


Figure 3.21 – Gaia installation Confirmation prompt

- I. Observe installation progress until you see the **Installation complete** screen [1]. This final step in R81.10 and R81.20 look a bit different but require identical action: you must press *Enter* to reboot and finalize the installation process. R81.20 has somewhat incorrect wording for this step: **Press return to quit** [2]. It does not actually quit the installation process, but rebooting the system to complete it:

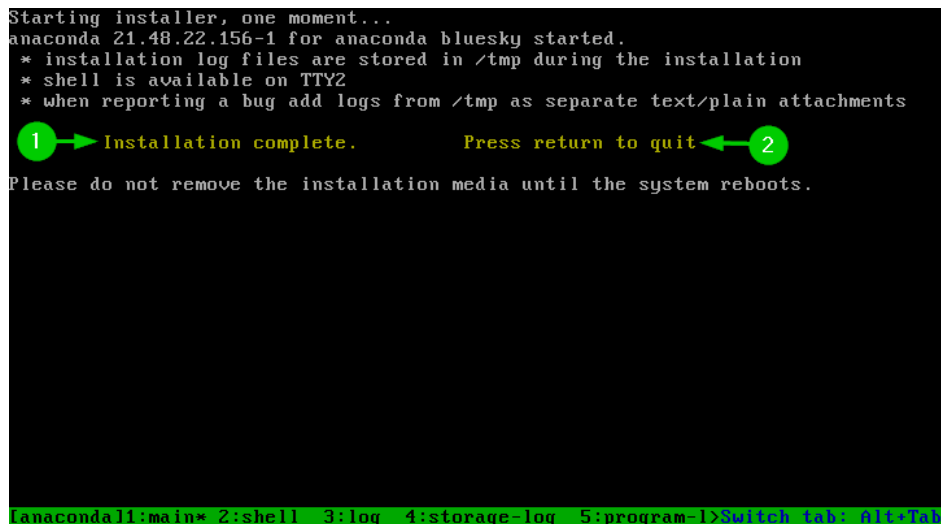


Figure 3.22 – Gaia Installation complete prompt (R81.20)



- J. In **CPBASE [Running]** – **Oracle VM VirtualBox** console, login to VM using credentials of **admin** and **CPL@b8110**. Change hostname to **CPBASE**, save configuration and halt **CPBASE** VM using following commands:

```
set hostname CPBASE
save config
halt
y
```

- K. Press *Right Ctrl* button to release keyboard captured in VM console window.

**Important Note:**

The process described above is only relevant to Check Point installations in virtual environments, on open servers, or when re-installing on Check Point appliances from bootable USB media or via LOM. Under normal circumstances, Check Point hardware appliances and the cloud-based images are pre-configured, and you are not required to perform these actions.

## Creating Windows Server 2019 Standard Base VM

### On LabHost PC:

1. From *ELEVATED CMD*, execute `unattended_swap.bat`
2. Exit *ELEVATED CMD*
3. From *CMD*, execute `create_WINBASE.bat`
4. Once **WINBASE** is up and running, reboot it. This will complete the integration of **VirtualBox Extension** and mount the LabShare (`\\VBoxSvr`) (`F:`) drive. We will also be referring to this location, throughout the book, as a LabShare.

### On WINBASE VM:

1. From LabShare (`F:`) drive, Install **Google Chrome**, **PuTTY**, **WinSCP** and **Notepad++**
2. In Windows command prompt, from `F:\Scripts>`, execute `WINBASE_sysprep.bat`
3. Observe *sysprep* progress
4. Once *sysprep* is finished, **WINBASE** VM will shutdown

## On LabHost PC:

1. In Windows command prompt, from `C:\CPBook\LabShare\Scripts>` execute:
  - A. `eject_and_snapshot_WINBASE.bat`
  - B. `eject_and_snapshot_CPBASE.bat`
  - C. `cloneVMs.bat`
2. In VirtualBox UI, observe the creation of lab component VMs.
3. In command prompt window, where this script was executing, you can see a list of all VMs in your environment.

## Preparing cloned Windows hosts VMs

1. Power-up all Windows VMs, except **ADDCDNS**
2. On each, complete setup by selecting appropriate time zone, accepting **EULA** and entering the password `CPL@b8110`
3. Once the VM you are currently working on is up, login using Administrator and `CPL@b8110`
4. On each Windows VM, unpin **Internet Explorer** from the taskbar and set Google Chrome as your default browser in **Start|Settings|Default apps|Web browser**, click on **Internet Explorer** and, in **Choose an app** popup, click **Google Chrome**.
5. On each Windows VM, *except* **ADDCDNS**, open `LabShare\Scripts` and execute the batch file with the name corresponding to the VM you are working on\*:
  - A. For **SmartConsole**, execute `SmartConsole.bat`
  - B. For **LeftHost**, execute `LeftHost.bat`
  - C. For **RightHost**, execute `RightHost.bat`
  - D. For **DMZSRV**, execute `DMZSRV.bat`

\*Alternatively, if you would like to observe the execution of PowerShell commands contained in `<VMNAME>.ps1` files triggered by the `,<VMNAME>.bat` files, you may open corresponding `.ps1` for editing on each VM, select and copy their content *WITHOUT* empty last line, and paste them into **PowerShell** console of each VM. To complete the execution of the last, non-empty line, press *Enter*.

These batch scripts will perform the following tasks on corresponding VMs:

- Disable automatic Windows Update services.
- Disable the Google update service.
- Disable Microsoft MapsBroker.
- Change the IP address.
- Configure DNS Server.
- Change hostname.
- Reboot the VM for changes to take effect.

## Preparing Active Directory and the domain controller VM

1. On **ADDCDNS** VM, open *LabShare\Scripts*, execute `addcdns_step1.bat`, this will rename and reboot the Windows server.
2. Once **ADDCDNS** VM is up and running again, login and execute `addcdns_step2.bat`, this will perform following tasks:
  - Disable automatic Windows Update services.
  - Disable the Google update service.
  - Disable Microsoft MapsBroker.
  - Change the IP address.
  - Set the local DNS client to point to itself.
  - Install Active Directory and its management tools.
  - Configure this server as a domain controller and a DNS server.
3. When prompted, enter the password for **SafeModeAdministratorPassword** (also known as **Directory Services Restore Mode (DSRM)**) and confirm it (CPL@b8110).
4. Type *Y* and press *Enter* to confirm that this server will be configured as a domain controller.
5. Observe the script in action and wait for the VM to reboot.

**Note:**

After reboot, this VM will go through a long **Applying Computer Settings** cycle, as it is getting ready to become your first domain controller.

6. Log on as **AD\administrator** with the **CPL@b8110** password.
7. When you log on, **Server Manager** will automatically start. Dismiss the **Try managing servers with Windows Admin Center** popup permanently by checking **Don't show this message again** and clicking on **X** in the top-right corner of the window.
8. Your **Server Manager | Dashboard** should look like this:

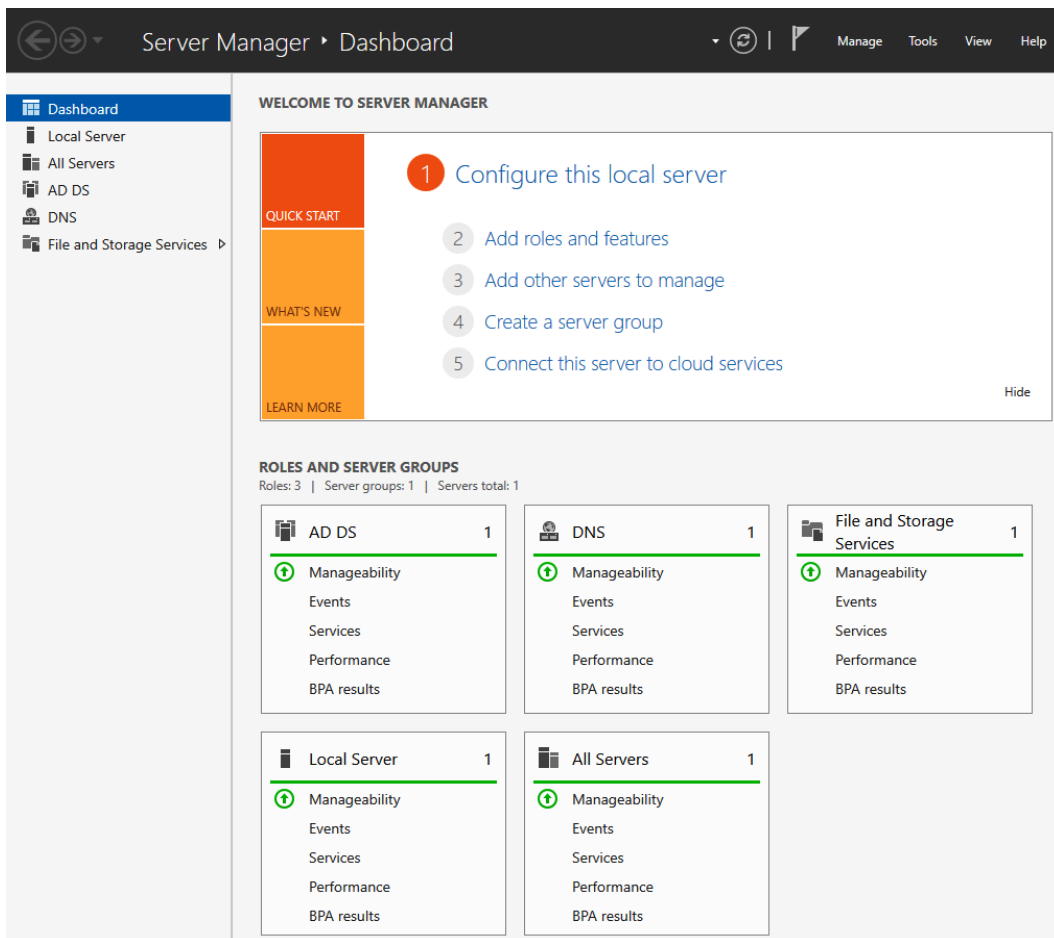


Figure 3.23 – Server Manager on ADDCDNS

9. From the **ADDCDNS** VM's command prompt, execute the `F:\Scripts\Subnets_and_dns_ADDCDNS.bat` file.  
This script will create domain networks, DNS zones, DNS A and PTR records for our lab environment.
10. Exit from **PowerShell** and **CMD**.

This concludes the preparation of Windows servers in the lab. With all Windows VMs configured, we can move on to the Check Point lab components.

## Preparing cloned Check Point hosts

Each cloned Check Point VM has to be assigned an individual hostname and IP address in the corresponding network segment, as shown below:

1. Preparing **CPSMS** VM:
  - A. In Oracle **VM VirtualBox Manager**, start the **CPSMS** VM.
  - B. Wait for the login prompt.
  - C. Log on using admin and `CPL@b8110` credentials.
  - D. Change the hostname and the IP address of the **eth0** interface and save the configuration by typing these lines manually and pressing *Enter* after each:

```
set hostname CPSMS
set interface eth0 ipv4-address 10.0.0.10 mask-length
24
set interface eth0 comments "Mgmt"
save config
```

2. Preparing **CPCM1** VM:
  - A. In Oracle **VM VirtualBox Manager**, start the **CPCM1** VM.
  - B. Wait for the login prompt.
  - C. Log on using admin and `CPL@b8110` credentials.

- D. Change the hostname and the IP address of the **eth0** interface and save the configuration by typing these lines manually and pressing *Enter* after each:

```
set hostname CPCM1
set interface eth0 ipv4-address 10.0.0.2 mask-length
24
set interface eth0 comments "Mgmt"
save config
```

3. Preparing VM:

- A. In **Oracle VM VirtualBox Manager**, start the **CPCM2** VM.
- B. Wait for the login prompt.
- C. Log on using admin and CPL@b8110 credentials.
- D. Change the hostname and the IP address of the **eth0** interface and save the configuration by typing these lines manually and pressing *Enter* after each:

```
set hostname CPCM2
set interface eth0 ipv4-address 10.0.0.3 mask-length
24
set interface eth0 comments "Mgmt"
save config
```

## Preparing SmartConsole VM

1. Start and log on to the **SmartConsole** VM.
2. Click **Start**, type **putty**, and press *Enter*.
3. In **PuTTY Configuration** window, in the **Host Name (or IP address)** field, enter 10.0.0.10 [1], and in **Saved Sessions**, enter **CPSMS** [2]. Click **Save** [3]:

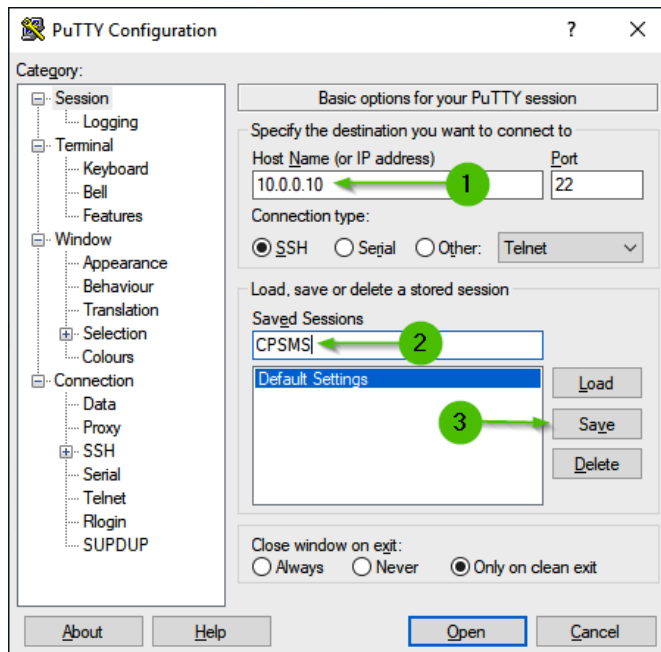


Figure 3.24 – Creating PuTTY sessions

4. Repeat this process, creating sessions for **CPCM1** (10.0.0.02) and **CPCM2** (10.0.0.03).
5. Once all three hosts' sessions are saved, double-click on **CPSMS**.
6. When prompted with **PuTTY Security Alert**, click **Accept** [1]:

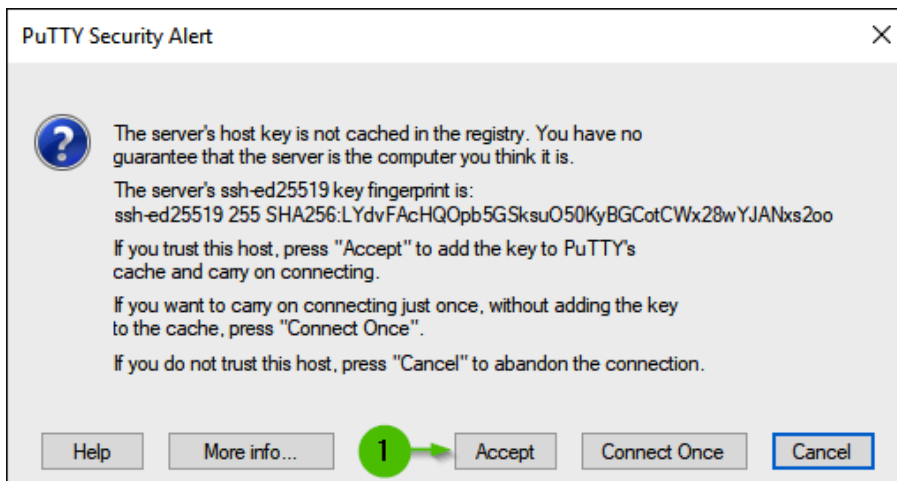
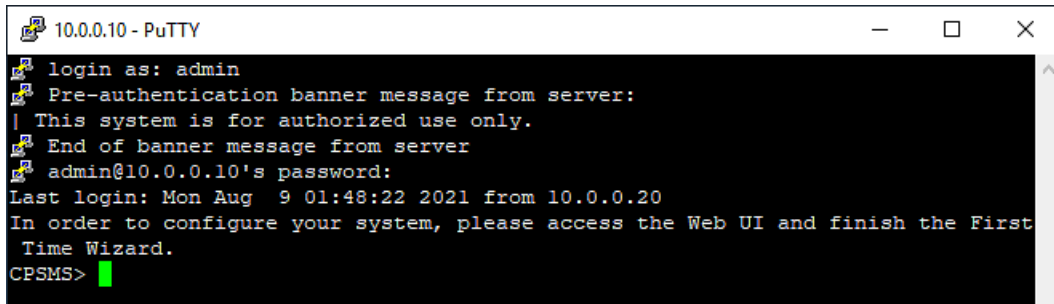


Figure 3.25 – Accepting hosts' keys

7. You should see similar screens for all three hosts on the left side of your topology:



```
10.0.0.10 - PuTTY
login as: admin
Pre-authentication banner message from server:
| This system is for authorized use only.
End of banner message from server
admin@10.0.0.10's password:
Last login: Mon Aug 9 01:48:22 2021 from 10.0.0.20
In order to configure your system, please access the Web UI and finish the First
Time Wizard.
CPSMS>
```

Figure 3.26 – SSH session to the Check Point host

8. While **PuTTY** is running, right-click on its icon in the **Windows Taskbar** and click on **Pin to taskbar** for convenience.

This concludes the preparation of the Check Point components in the lab, as well as overall initial lab hosts configuration process.

## Summary

In this guide, we have created our lab environment containing typical elements of the production infrastructure. We were able to take advantage of the native VirtualBox and batch scripting to accomplish some of these tasks and we have used PowerShell scripts to create or modify Windows components. You now have the necessary skills for building your own labs with different topologies in the future and can modify and reuse your scripts and commands to accomplish that. Now that we have all the lab components in place and understand how all of them are interconnected, we are ready to start working on our Check Point component configuration using the Gaia OS web UI and CLI.