BAHIR DAR UNIVERSITY

INSTITUTE OF TECHNOLOGY

FUCULITY OF COMPUTING

Department: Software engineering

Operating system and system programming individual assignment

Course code- Seng2034 Section-B

Prepared by :ABEBE WORKINEH

ID:BDU1601007

Submitted to:Wondimu.B

Submitted date:16/08/2017

E.C

Table of content page

HAIKUS OPERATING SYSTEM……………………………………………………3

1. INTRODUCTION……………………………………………………………… 3

1.1 Background of the Haiku Operating System……………..……………………3

1.2 Motivation Behind the Haiku Operating System………..……………………4

2. Objectives of the Haiku Operating System…………………..…………………5

2.1 Binary Compatibility with BeOS……………………………………………5

2.2 Performance and Efficiency………………………………..……………………5

2.3 Modular and Clean System Architecture………………..……………………6

2.4 User-Friendly and Intuitive Desktop Experience………………………………6

2.5 Open-Source Development and……………………………………………6

2.6 Modern Hardware and Software Support……………………………………7

2.7 Future-Proofing and Innovation……………………..……………………7

3. Hardware and Software Requirements for Haiku Operating System……………7

3.1Minimum Hardware Requirements……………………………………………7

3.2 Software Requirements for Development……………………………………8

4. Steps to install haiku operating system using vmware workstation 17 pro…………8

5. File system support……………………………………………………………19

6. Advantage and disadvantage of haiku OS……………………………………20

7. conclustion…………………………………………………………………21

8. Future outlook………………………………………………………………21

9. Virtualization Tools…………………………………………………………22

10. Implementation of fork() system call in haiku operating system………………23

HAIKUS OPERATING SYSTEM

1. INTRODUCTION

1.1 Background of the Haiku Operating System

Haiku is an open source operating system under constant development. Specifically targ
eting personal computing, Haiku is a fast, efficient, simple to use, easy to learn, and yet v
ery powerful system for computer users of all levels. Additionally, Haiku offers somethin
g over other open source platforms which is quite unique: The project consists of a singl
e team writing everything from the kernel, drivers, userland services, tool kit, and graphics
stack to the included desktop applications and preflets. While numerous open source pro
jects are utilized in Haiku, they are integrated seamlessly. This allows Haiku to achieve a
unique level of consistency that provides many conveniences, and is truly enjoyable to u
se by both end-users and developers alike.

Origins and Inspiration

The Haiku operating system traces its roots back to the now-defunct BeOS, a highly effic
ient and multimedia-focused operating system developed by Be Inc. in the 1990s. When
Palm, Inc. acquired Be Inc. in 2001 and discontinued BeOS, a group of enthusiasts, led b
y developer Michael Phipps, sought to preserve its innovative design by launching an op
en-source reimplementation called OpenBeOS on August 18, 2001. The project aimed to
replicate BeOS's performance, simplicity, and binary compatibility while avoiding propriet
ary code. In 2004, due to trademark concerns with Palm, the project was renamed Haiku
—a nod to the minimalist and elegant error messages in BeOS's NetPositive browser.

Development Philosophy and Goals

Haiku was designed to retain BeOS's strengths, such as its **fully threaded architecture** fo
r optimal multicore CPU utilization, a **responsive custom kernel**, and a **database-like file
system (BFS)** with metadata indexing. Unlike other BeOS-inspired projects (e.g., Cosmoe,
BlueEyedOS), Haiku prioritized **binary compatibility** with BeOS R5, allowing legacy applic

ations to run without modification. The project also emphasized **modularity**, with teams independently developing components like the kernel, GUI toolkit (Interface Kit), and media services.

Key Milestones

Early development focused on foundational components, such as the **app_server** (2002) for GUI rendering and the **Tracker** file manager (2005). By 2008, Haiku achieved **self-hosting**, meaning it could compile its own code. The first alpha release (R1/Alpha 1) arrived in 2009, followed by iterative updates adding features like **64-bit support** (2013), **UEFI booting** (2018), and **NVMe storage** (2019). The project transitioned to beta in 2018, with **Haiku R1 Beta 3** released in 2021.

Community and Governance

Haiku is driven by a **global volunteer community** and supported by **Haiku, Inc.**, a non-profit founded in 2003 to oversee funding and trademarks. Early contributors like Axel Dörfler (kernel development) and Stephan Aßmus (GUI tools) played pivotal roles. The project also participated in **Google Summer of Code** and **Outreachy** to mentor new developers

Technical Legacy and Future

Haiku's hybrid **GCC2/GCC7** environment balances compatibility with modern software, though its reliance on an outdated GCC 2.95 fork for BeOS apps poses challenges. Plans for **R2** include enhanced hardware support, multi-user functionality, and a modernized compiler. Despite its niche status, Haiku remains a testament to BeOS's influence, offering a **fast, consistent, and open-source alternative** for personal computing .

1.2 Motivation Behind the Haiku Operating System

Preserving the Legacy of BeOS

The primary motivation behind Haiku was to revive and preserve the innovative design of **BeOS**, an operating system developed by Be Inc. in the 1990s. BeOS was known for its **high performance, multimedia capabilities, and responsive user experience**, but after Be Inc. was acquired by Palm in 2001, the OS was discontinued. A group of enthusiasts, led by developer **Michael Phipps**, launched the **OpenBeOS** project (later renamed Haiku) to recreate BeOS as an open-source system, ensuring its concepts would not be lost to history.

Addressing the Limitations of Modern Operating Systems

Many developers and users felt that mainstream operating systems like Windows, macOS, and Linux had become **bloated, slow, and overly complex.** BeOS, in contrast, was designed for **efficiency, real-time performance, and simplicity**. Haiku aimed to bring back these qualities by offering:

- **A lightweight, responsive kernel** optimized for modern hardware.
- **A fully threaded architecture** that maximizes CPU utilization.
- **A consistent, clean API** (unlike Linux's fragmented ecosystem).
- **A focus on desktop usability** rather than server or enterprise features.

Creating a Truly Personal Computing Experience

Haiku was envisioned as an OS that **"gets out of the user's way"**—prioritizing speed, stability, and ease of use. Unlike Linux distributions, which often require extensive configuration, Haiku sought to provide a **cohesive, out-of-the-box experience** similar to classic BeOS. This included:

- **A unified, intuitive GUI** with minimal distractions.
- **A database-like filesystem (BFS)** for fast metadata searches.
- **Strong multimedia support**, making it ideal for creative work.

Fostering an Open-Source Alternative with Binary Compatibility

While other BeOS-inspired projects existed (e.g., Cosmoe, Zeta), Haiku stood out by focusing on **open-source development** while maintaining **binary compatibility** with BeOS R5 applications. This allowed legacy BeOS software to run without modification, providing a smooth transition for former BeOS users.

Encouraging a Passion-Driven Development Model

Unlike corporate-backed operating systems, Haiku was built by **a community of volunteers** who believed in its vision. The project embraced **transparency, modularity, and incremental progress**, with developers contributing out of passion rather than commercial incentives. Initiatives like **Google Summer of Code** helped bring in new talent to sustain development.

Future Goals: Keeping Haiku Relevant

While Haiku remains a niche OS, its development continues with goals such as:

- **Improved hardware support** (Wi-Fi, GPUs, ARM architecture).
- **Modernizing the tool chain** (transitioning from GCC2 to newer compilers).
- **Adding multi-user support** for broader usability.

Ultimately, Haiku's motivation lies in **proving that a fast, elegant, and user-focused operating system is still possible**—free from the bloat and fragmentation of mainstream alternatives.

2.Objectives of the Haiku Operating System

2.1Binary Compatibility with BeOS

One of Haiku's primary objectives was to maintain **binary compatibility** with BeOS R5 ap plications. This ensured that legacy BeOS software could run seamlessly on Haiku witho ut requiring modifications or recompilation. By preserving compatibility, Haiku aimed to provide a smooth transition for former BeOS users and developers, allowing them to con tinue using their favorite applications while benefiting from modern improvements. This objective also helped attract the original BeOS community, fostering early adoption and development contributions.

2.2 Performance and Efficiency

Haiku was designed to be a **lightweight, high-performance** operating system, inheriting BeOS's efficiency in handling multimedia and real-time tasks. Unlike modern OSes that o ften suffer from bloat, Haiku's objectives included:

- **Optimized kernel design** for fast boot times and low latency.
- **Fully threaded architecture** to maximize multi-core CPU utilization.
- **Minimal system overhead**, ensuring responsiveness even on older hardware.
  These goals made Haiku an attractive choice for users seeking a fast, streamlined computing experience.

2.3 Modular and Clean System Architecture

Haiku aimed to avoid the complexity and fragmentation seen in other open-source opera ting systems. Its design objectives included:

- **A cohesive, well-documented API** (unlike Linux's multiple competing framework s).
- **Strict adherence to clean, maintainable code** to ensure long-term sustainability.
- **Modular components** (e.g., kernel, app server, drivers) that could be independently improved.
  This approach made Haiku easier to develop, debug, and extend compared to mor e monolithic systems.

2.4 User-Friendly and Intuitive Desktop Experience

While many open-source projects prioritize technical flexibility over usability, Haiku aime d to provide a **polished, consistent desktop environment** inspired by BeOS. Key objective s included:

- **A unified, aesthetically pleasing GUI** with no unnecessary complexity.

- **A responsive file manager (Tracker)** with metadata and query-based search.
- **Predictable behavior** (e.g., no sudden slowdowns or crashes).
  By focusing on usability, Haiku sought to appeal to both technical and non-techni cal users.

2.5 Open-Source Development and Community Collaboration

Haiku was built as a **community-driven, open-source project**, distinguishing it from pro prietary alternatives. Its objectives in this area included:

- **Encouraging volunteer contributions** through mentorship programs like Google S ummer of Code.
- **Transparent governance** via Haiku, Inc., a non-profit overseeing development.
- **Avoiding corporate control**, ensuring the OS remained true to its original vision.
  This collaborative model helped sustain development despite limited funding.

2.6Modern Hardware and Software Support

While staying true to its BeOS roots, Haiku also aimed to **evolve with modern computing needs**. Objectives included:

- **Expanding hardware compatibility** (Wi-Fi, GPUs, NVMe, UEFI boot).
- **Gradual transition to 64-bit** while maintaining 32-bit support.
- **Improving software ecosystem** (ports of modern apps like Firefox, LibreOffice).
  These efforts ensured Haiku remained usable on contemporary systems.

2.7Future-Proofing and Innovation

Beyond replicating BeOS, Haiku's long-term objectives included:

- **Multi-user support** for broader adoption.
- **Enhanced security features** (sandboxing, modern permissions).
- **Support for ARM architecture** (Raspberry Pi, mobile devices).
  By balancing legacy compatibility with forward-looking improvements, Haiku aim ed to remain relevant in the ever-changing OS landscape.

3. Hardware and Software Requirements f or Haiku Operating System

3.1 Minimum Hardware Requirements

Haiku is designed to be lightweight and efficient, allowing it to run on older hardware while still supporting modern systems. The minimum requirements vary slightly between 32-bit and 64-bit versions:

32-bit (x86) Version

- **CPU:** Pentium II (400 MHz or higher)
- **RAM:** 384 MB (minimum), 512 MB recommended
- **Storage:** 1.3 GB (for installation), 3+ GB recommended for additional software
- **Graphics:** VESA-compatible or dedicated GPU with open-source driver support (Intel, AMD, or NVIDIA with basic acceleration)
- **Network:** Ethernet or FreeBSD-compatible Wi-Fi drivers (Atheros, Intel, Ralink, etc.)

64-bit (x86_64) Version

- **CPU:** 64-bit Intel/AMD processor (Core 2 Duo or later recommended)
- **RAM:** 512 MB (minimum), 1 GB+ recommended
- **Storage:** 2.5 GB (for installation), 5+ GB recommended
- **Graphics:** Same as 32-bit, with better support for modern GPUs via UEFI

Recommended Hardware for Optimal Performance

For a smoother experience, especially with modern applications:

- **CPU:** Multicore (dual-core or better) for Haiku's threaded architecture.
- **RAM:** 2 GB+ (web browsing and multitasking benefit from extra memory).
- **Storage:** SSD (faster BFS file system performance).
- **Wi-Fi:** Cards with FreeBSD driver support (e.g., Intel Dual-Band Wireless-AC 3160)

3.2 Software Requirements for Development

To build Haiku from source or develop applications:

On Haiku (Self-Hosting)

- **Tools:** `pkgman install cmd:python3 cmd:xorriso devel:libzstd.`
- **Compiler:** GCC2 (for BeOS compatibility) or GCC13 (modern apps).

Cross-Compiling from Linux/BSD/macOS

- **Essential Packages:** Git, NASM, GCC, Python 3, autotools, and Haiku's custom Jam build tool.
- **Platform-Specific:**
  - **Debian/Ubuntu:** `sudo apt install git nasm bc autoconf automake texinfo flex bison gawk build-essential unzip wget zip less zlib1g-dev libzstd-dev xorriso libt`

ool gcc-multilib python3.

- **macOS:** Requires a case-sensitive file system and Homebrew/MacPorts fo r dependencies

Compatibility Considerations

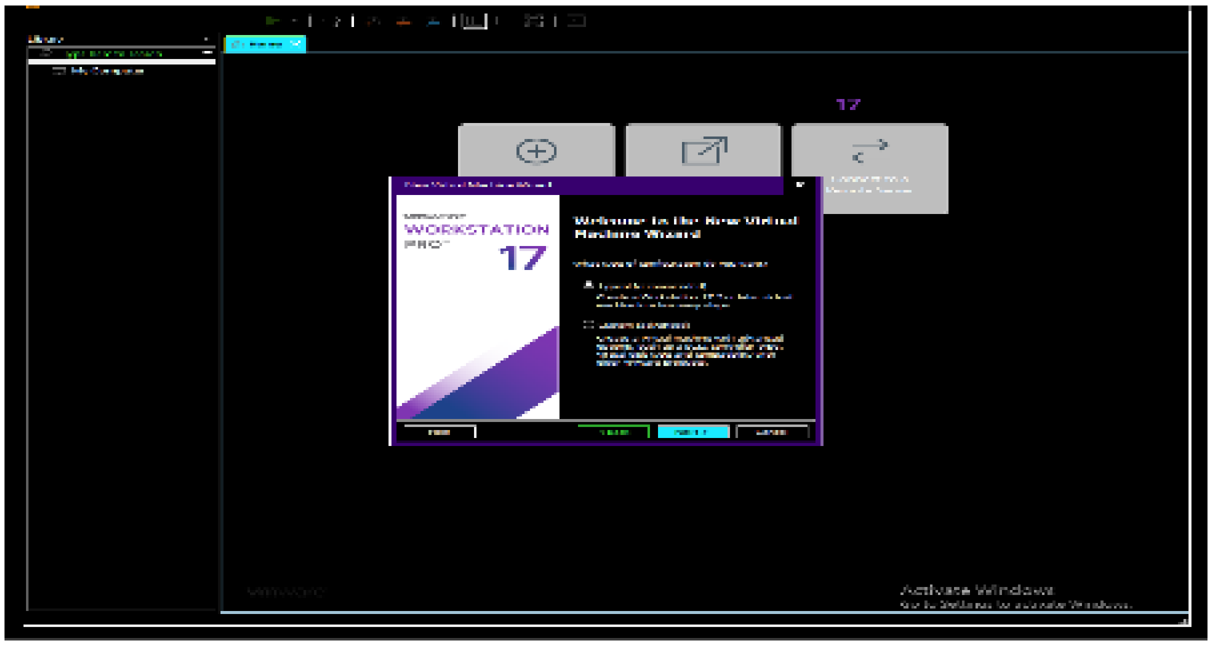
- **BeOS Binary Compatibility:** Only available on 32-bit Haiku (GCC2).
- **File Systems:** Supports BFS, ext2/3/4, NTFS (read-only), and FAT32.
- **Peripherals:** USB keyboards/mice work, but Bluetooth is experimental

4.Steps to install haiku operating system us ing vmware workstation 17 pro

Step 1:download vmware workstation 17 pro and then install it.
Step 2:download haiku operating system from the website
Step 3:create a new virtual machine

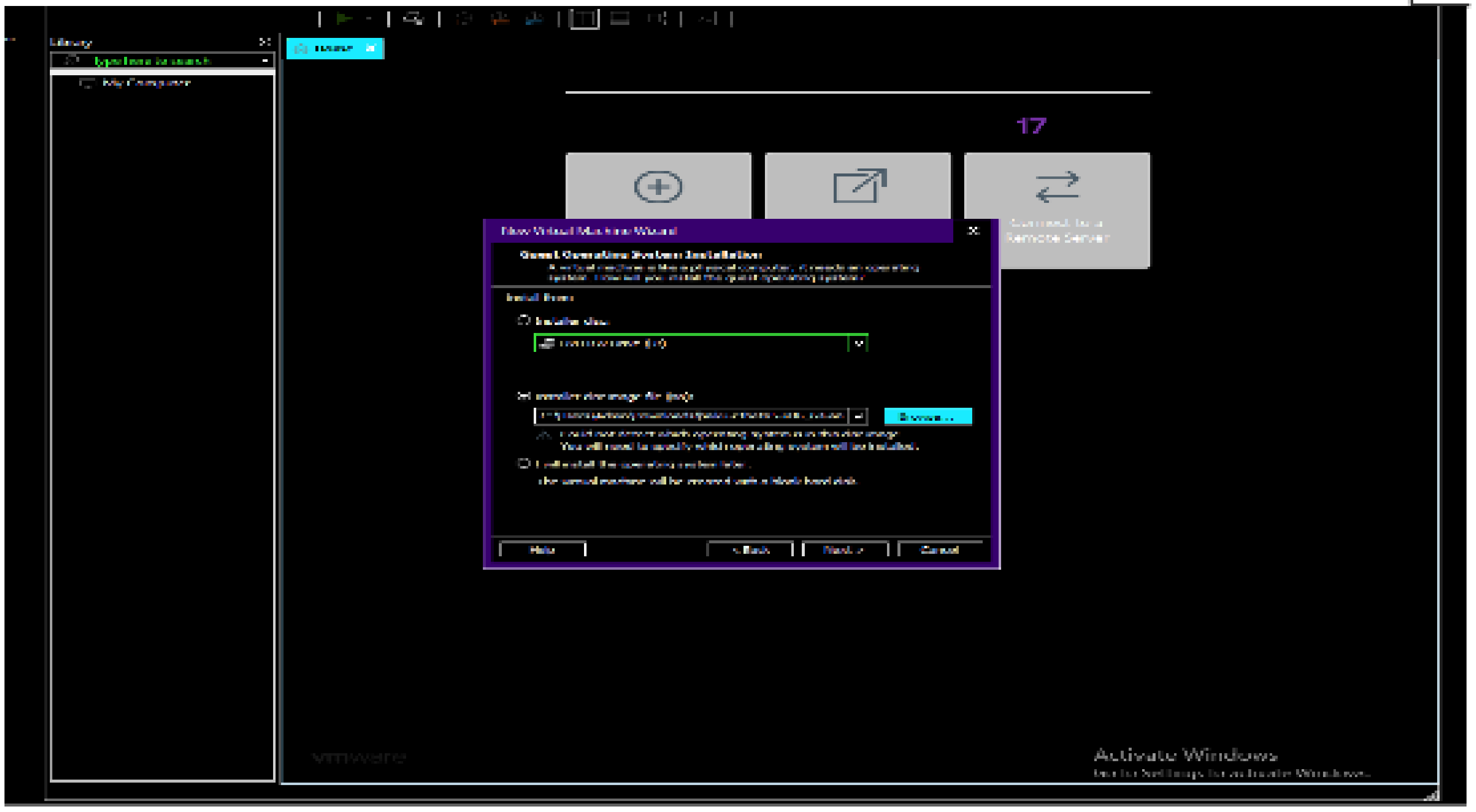
➢ Open vmware workstation pro 17->click "create a new virtual machine"->select "Typic al(recommended)"->click next

➢



➢ choose installation method

✓ Select "installer disk image file(iso)->browse and select the downloaded haiku iso fil e->click next.


New Virtual Machine Wizard

Guest Operating System Installation

Install from:

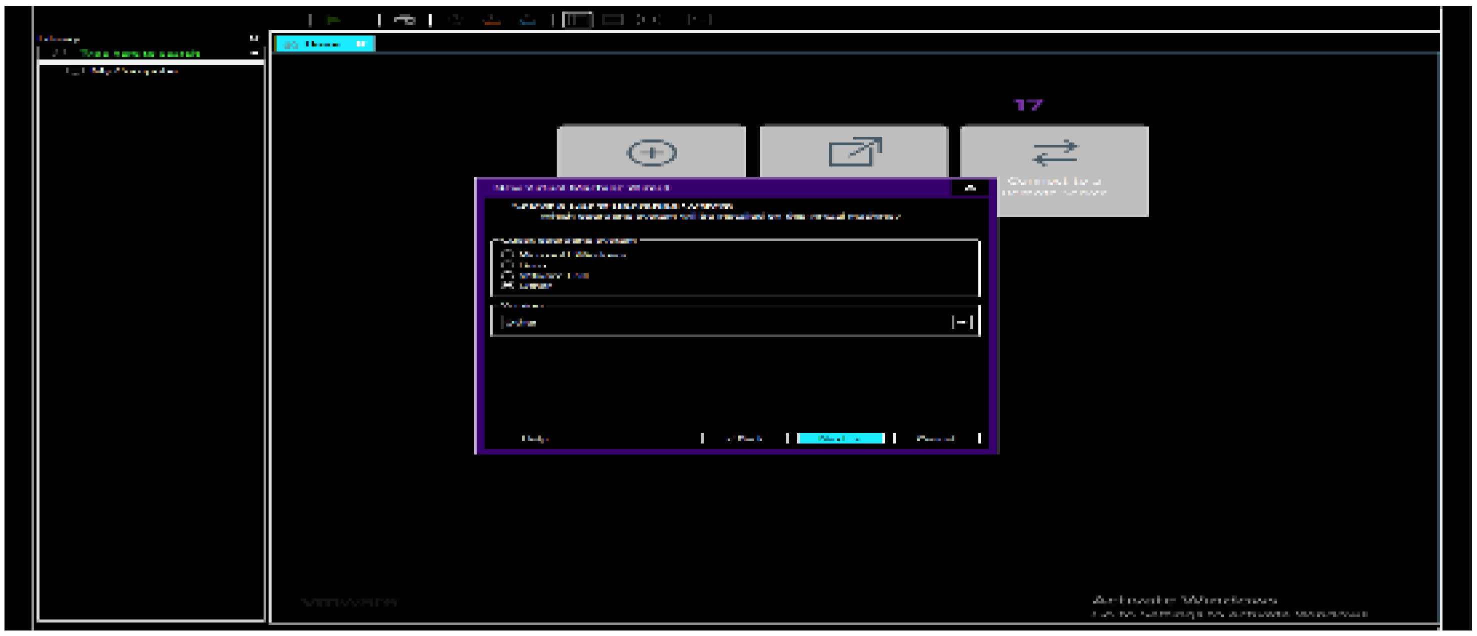
Installer disc:

Installer disc image file (iso):

I will install the operating system later.

The virtual machine will be created with a blank hard disk.

- ➢ Select Guest OS
- ✓ Choose other as the Guest OS
- ✓ From the dropdown select other since haiku is not listed by default
- ✓ Click "next"

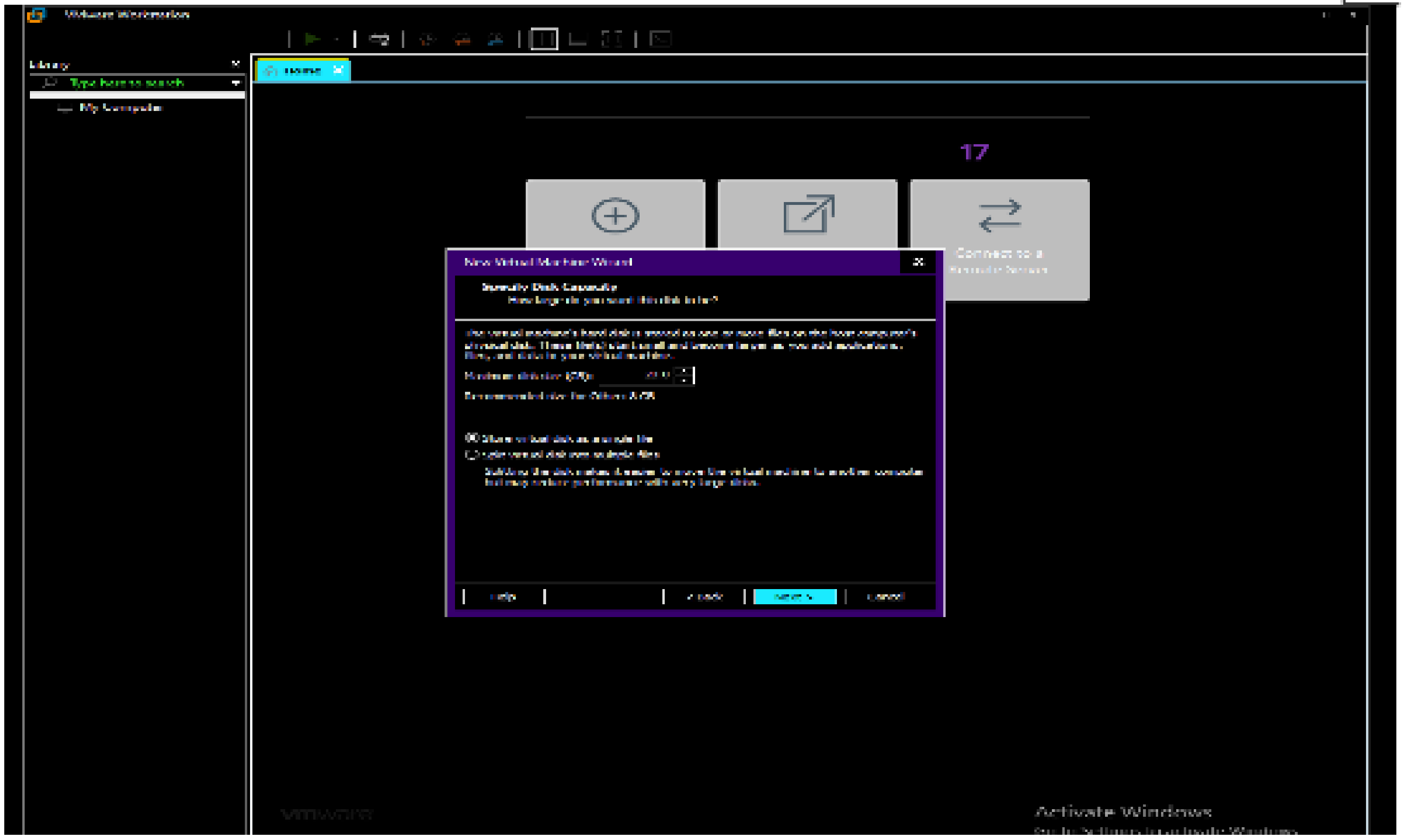

New Virtual Machine Wizard

Select a Guest Operating System

Microsoft Windows
Linux
VMware ESX
[✓ Other]

Version: Other

- ➢ specify disk capacity
- ✓ Allocate 20GB
- ✓ Select "store virtual disk as a single file"
- ✓ Click "next".



New Virtual Machine Wizard

Specify Disk Capacity
How large do you want this disk to be?

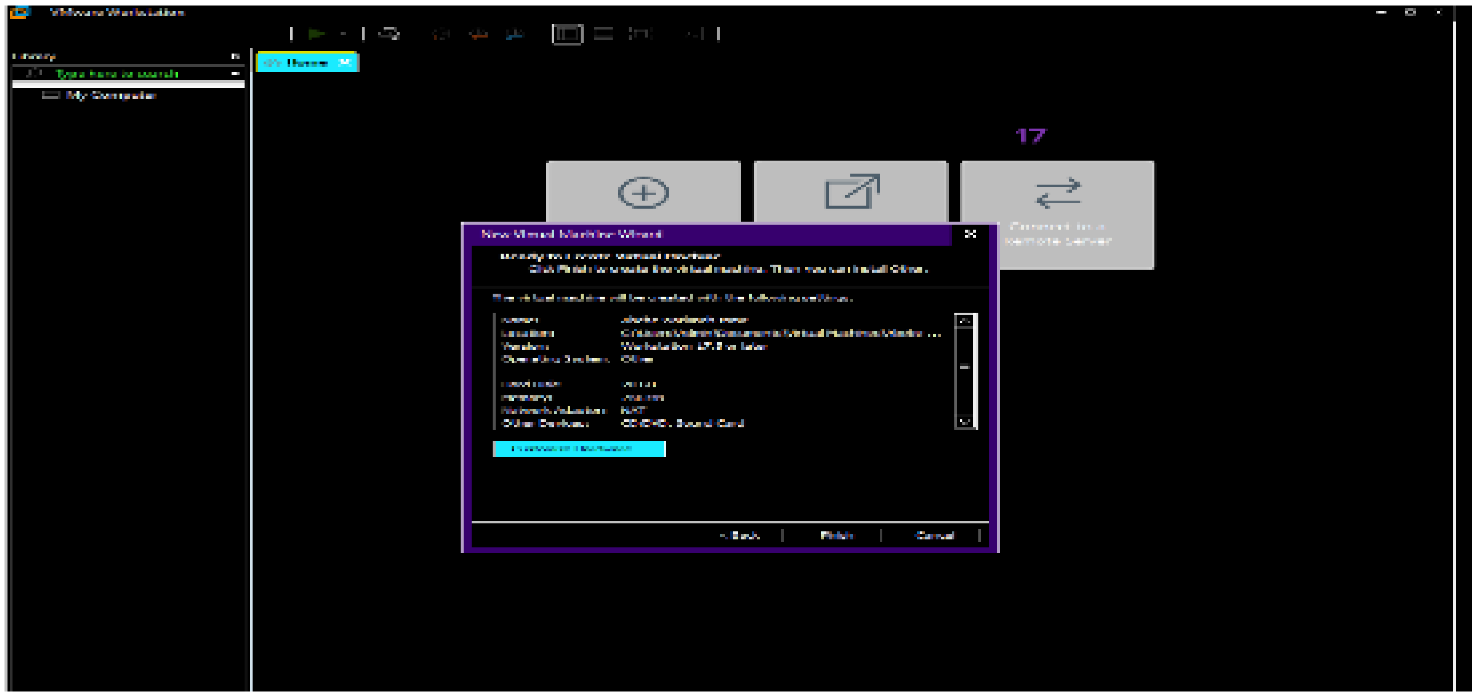
[illegible]

Maximum disk size (GB): [illegible]

Recommended size for Other: 8 GB

[✓ Store virtual disk as a single file]
Split virtual disk into multiple files
[illegible]

➢ Click “finish”.



New Virtual Machine Wizard

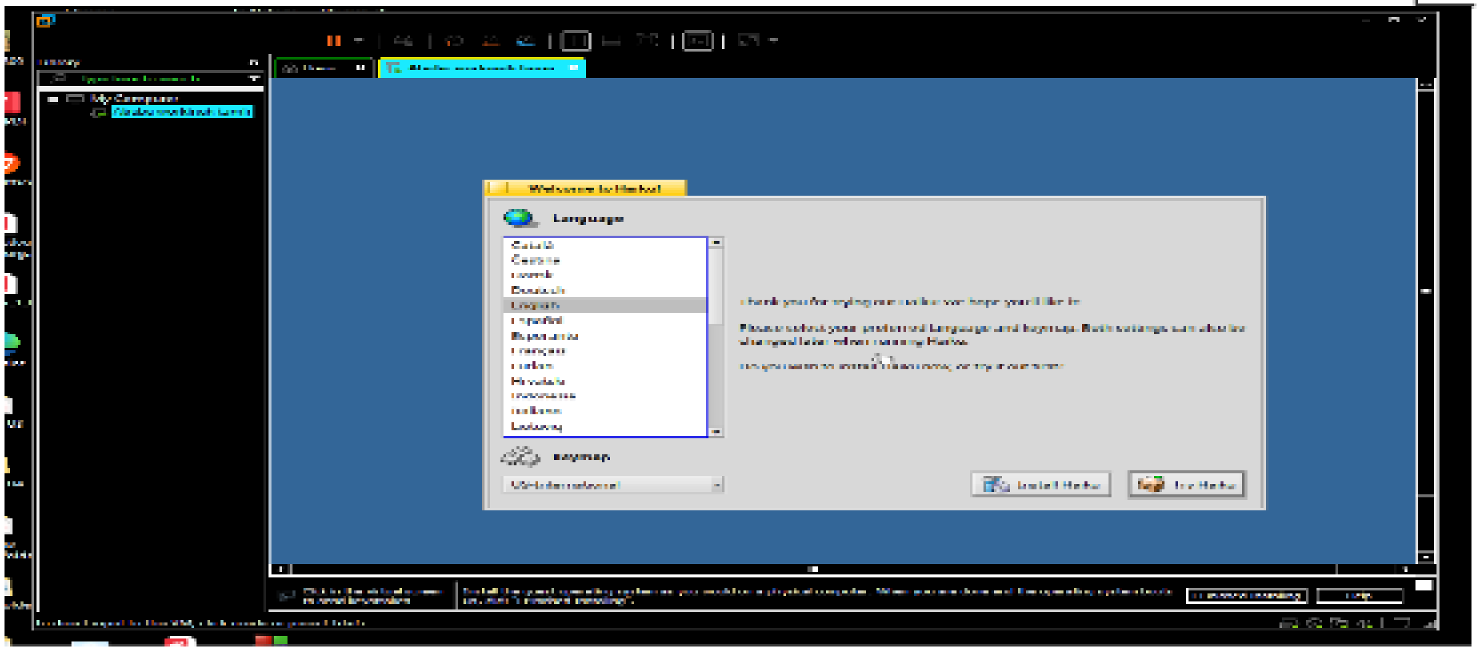
Ready to Create Virtual Machine
Click Finish to create the virtual machine. Then you can install Other.

The virtual machine will be created with the following settings:

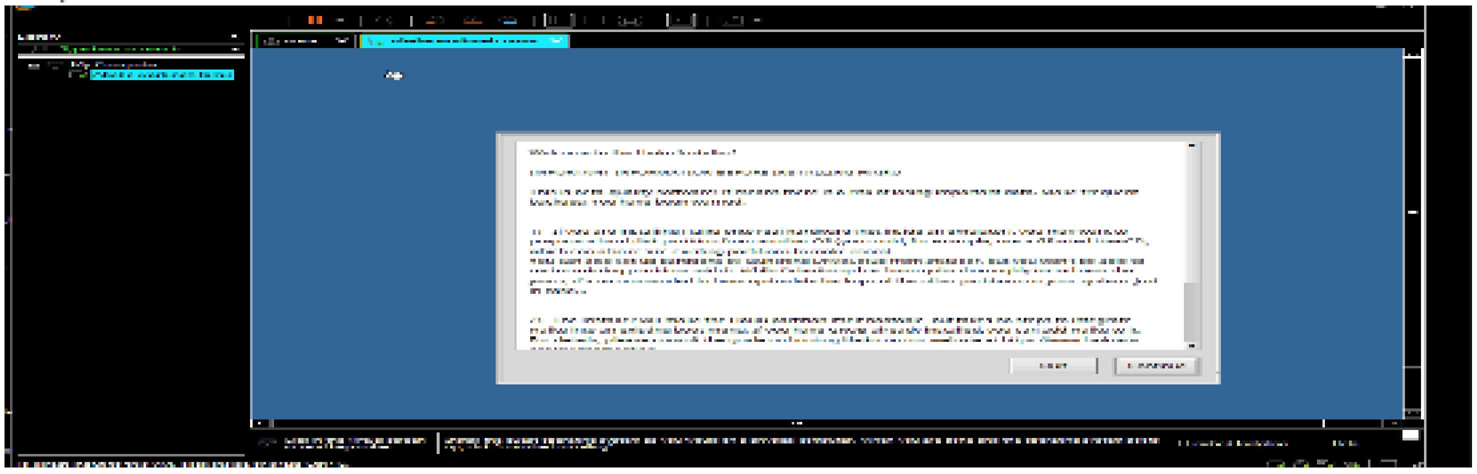
```
Name:               [illegible]
Location:           C:\Users\[illegible]\Documents\Virtual Machines\[illegible] ...
Version:            Workstation 17.5 or later
Operating System:   Other

Hard Disk:          [illegible]
Memory:             [illegible]
Network Adapter:    NAT
Other Devices:      CD/DVD, Sound Card
```

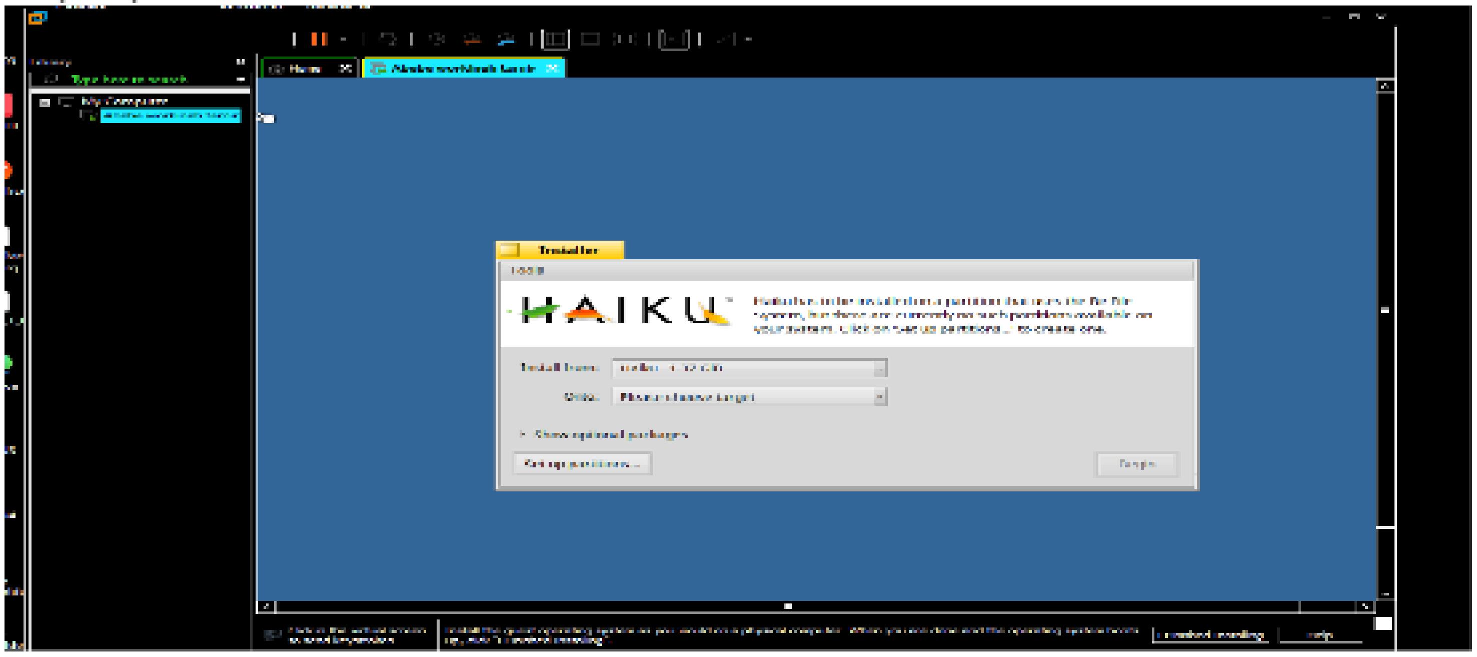
Step 4:select language and click “Install Haiku”

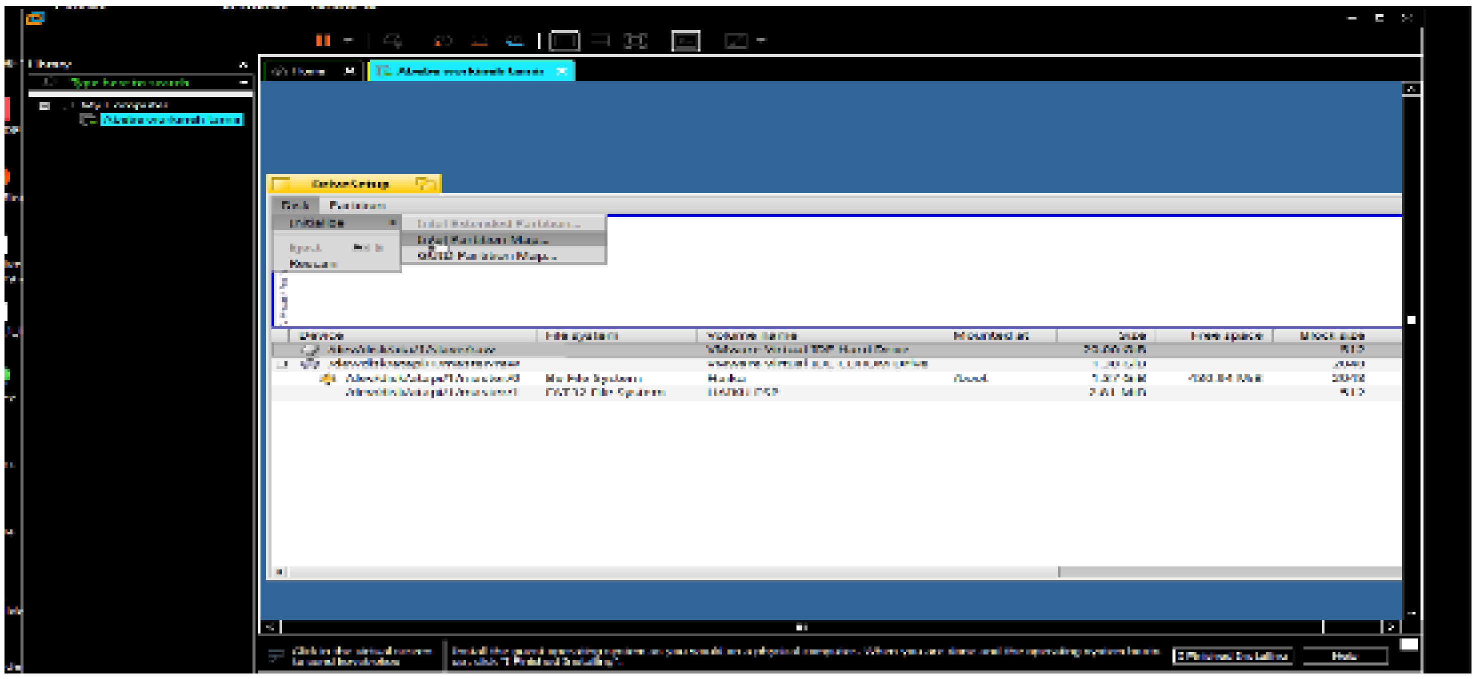


Step 5: click "continue"

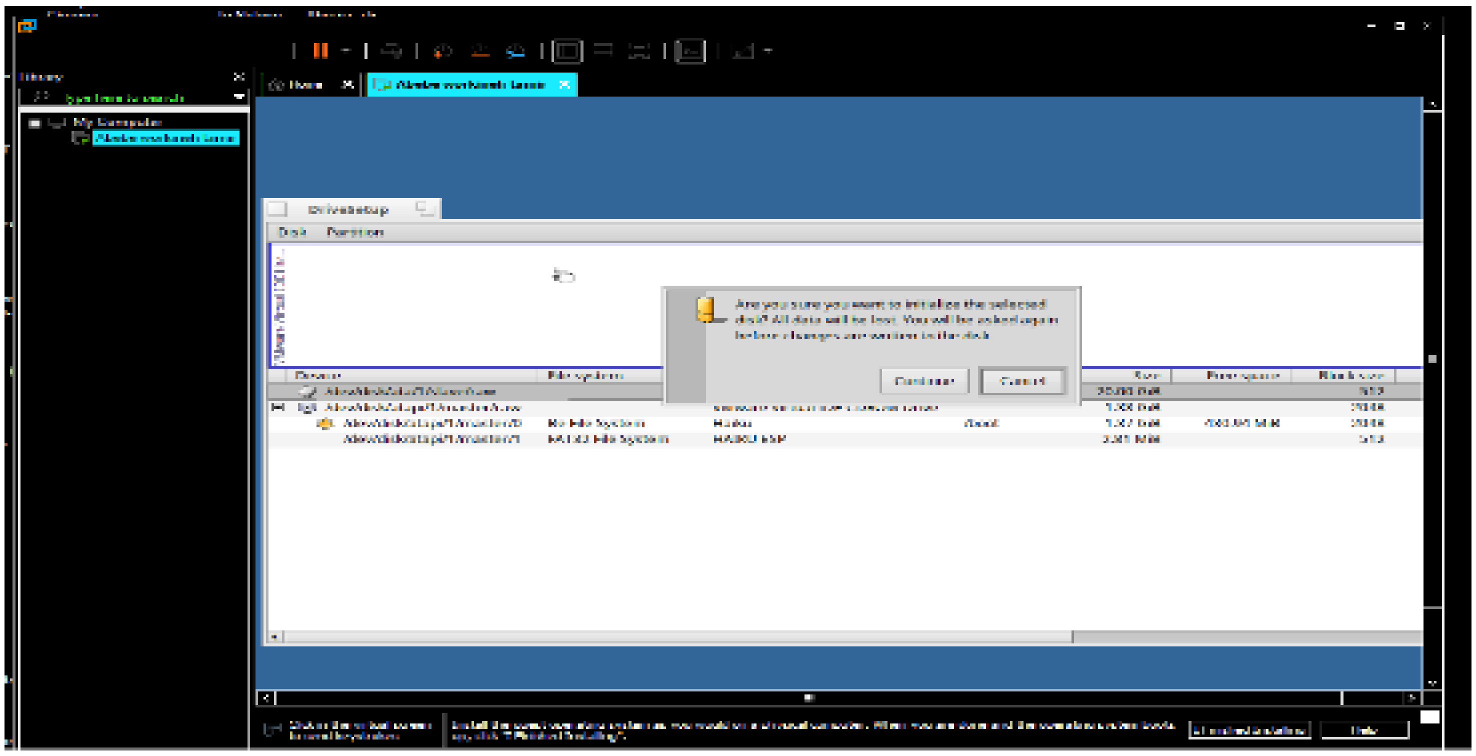


Step 6: partition the disk

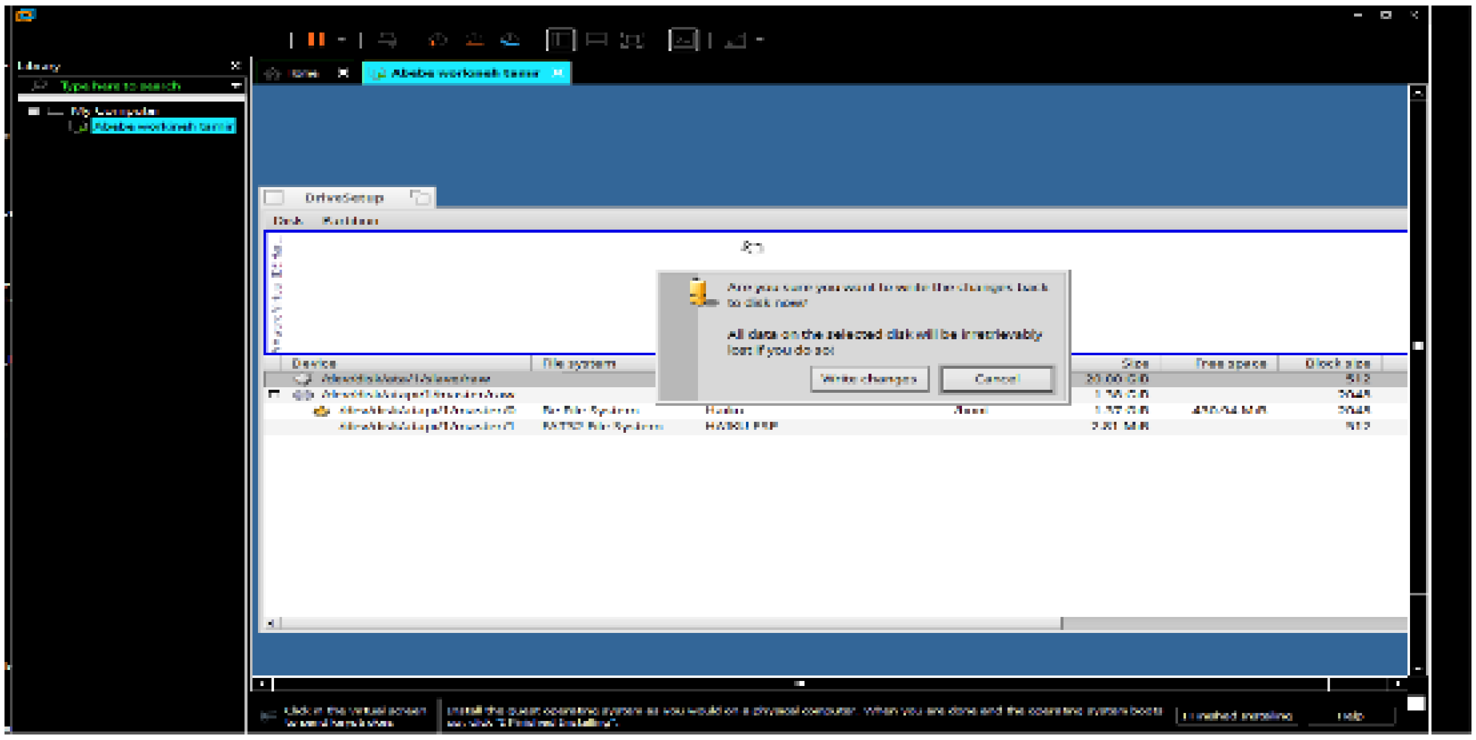




Device	File system	Volume name	Mounted at	Size	Free space	Block size
/dev/disk/ata/1/slave/raw		VMware Virtual IDE Hard Drive		20.00 GiB		512
[illegible]		[illegible]		[illegible]		2048
[illegible]	No File System	Haiku	/boot	1.37 GiB	[illegible]	2048
[illegible]	FAT32 File System	HAIKU ESP		[illegible]		512



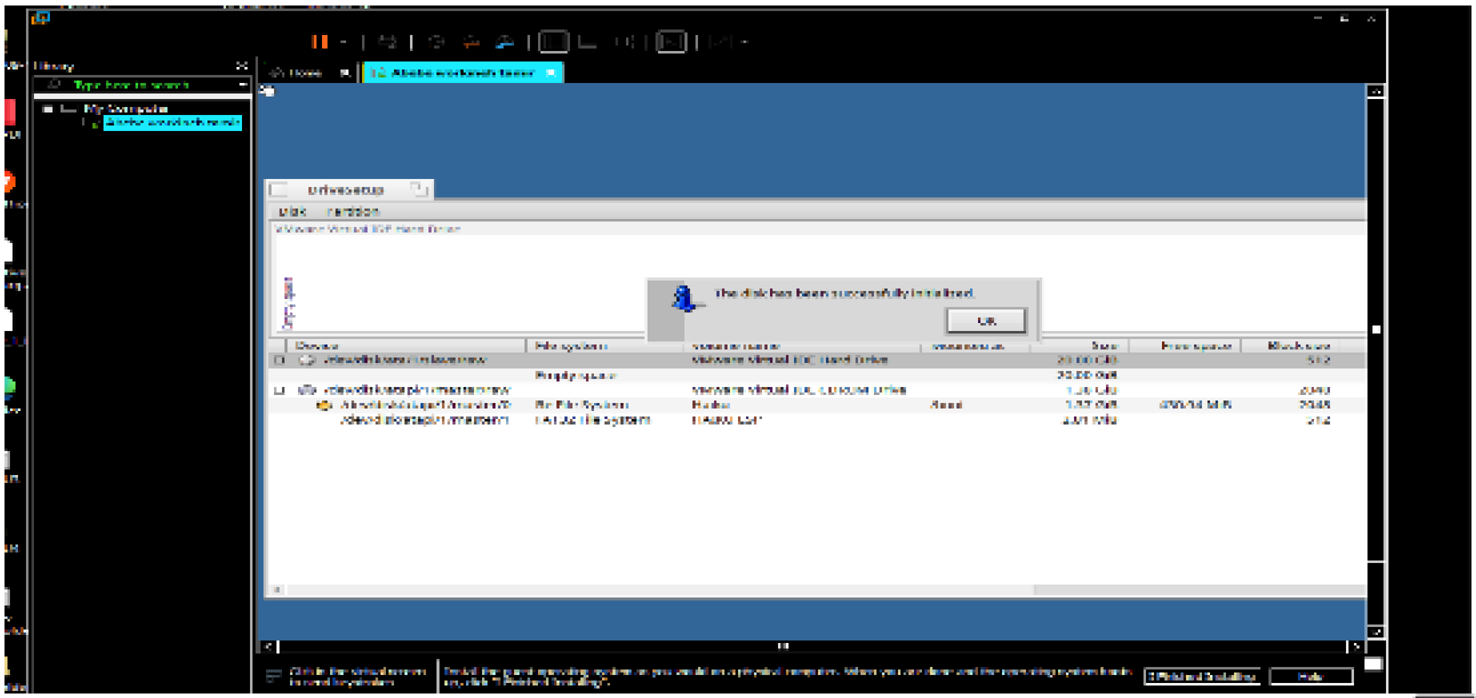
Are you sure you want to initialize the selected disk? All data will be lost. You will be asked again before changes are written to the disk.



Are you sure you want to write the changes back to disk now?

All data on the selected disk will be irretrievably lost if you do so!

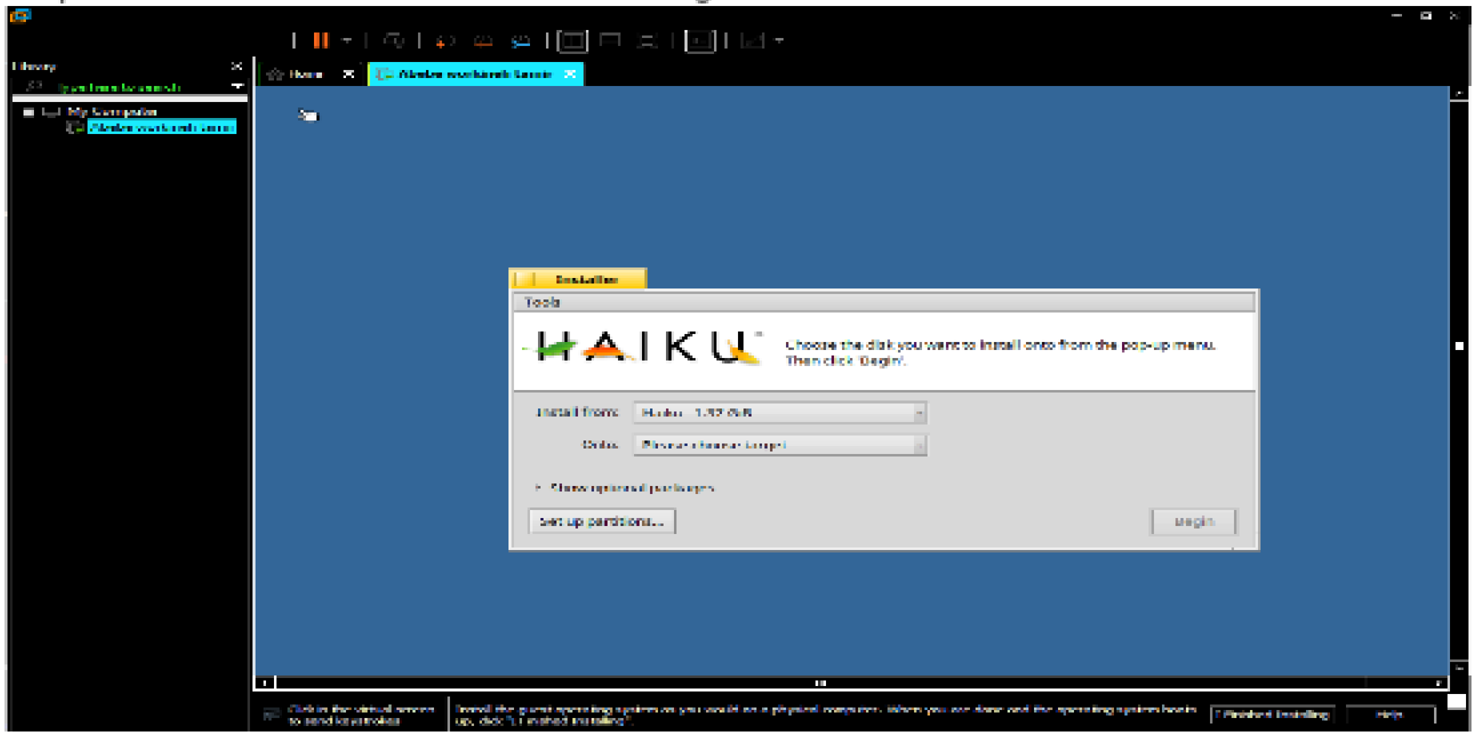
Install the guest operating system as you would on a physical computer. When you are done and the operating system boots up, click "I Finished Installing".



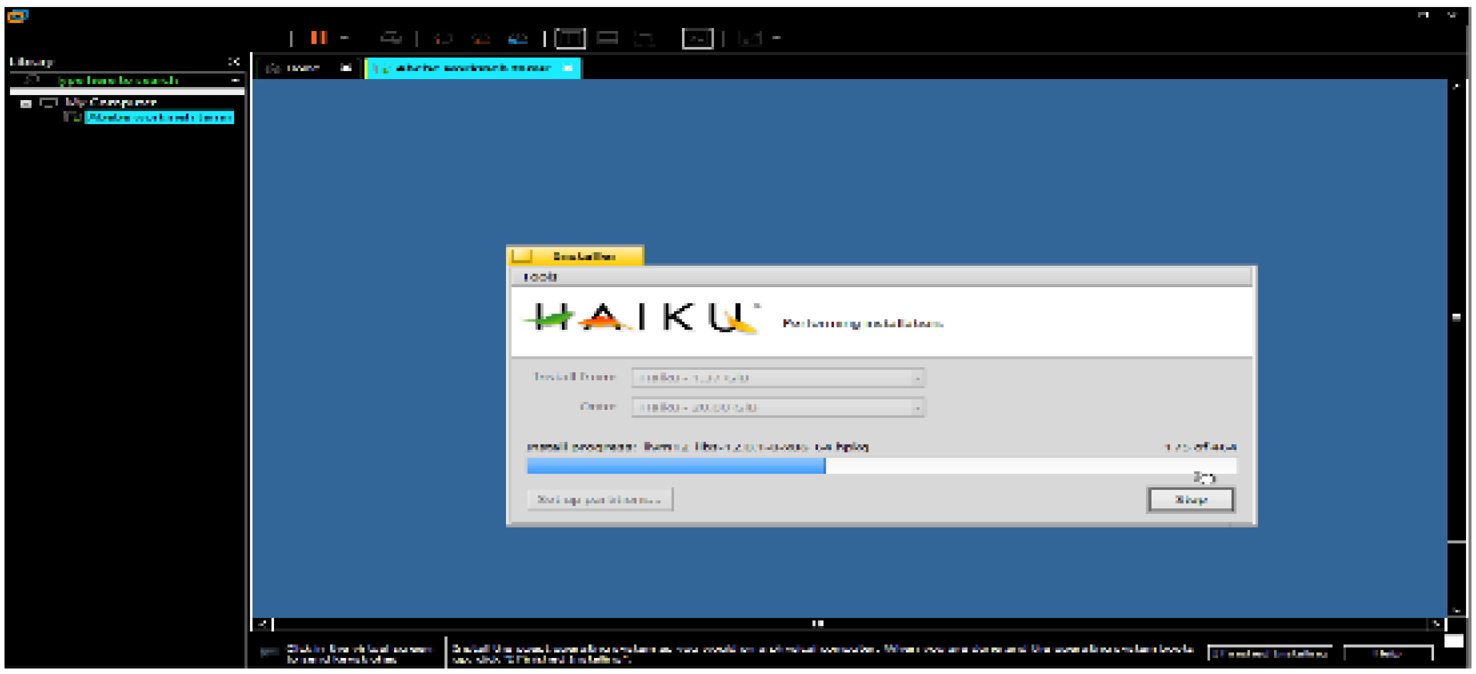
The disk has been successfully initialized.

Install the guest operating system as you would on a physical computer. When you are done and the operating system boots up, click "I Finished Installing".

Step 7: choose the disk and then click "Begin"

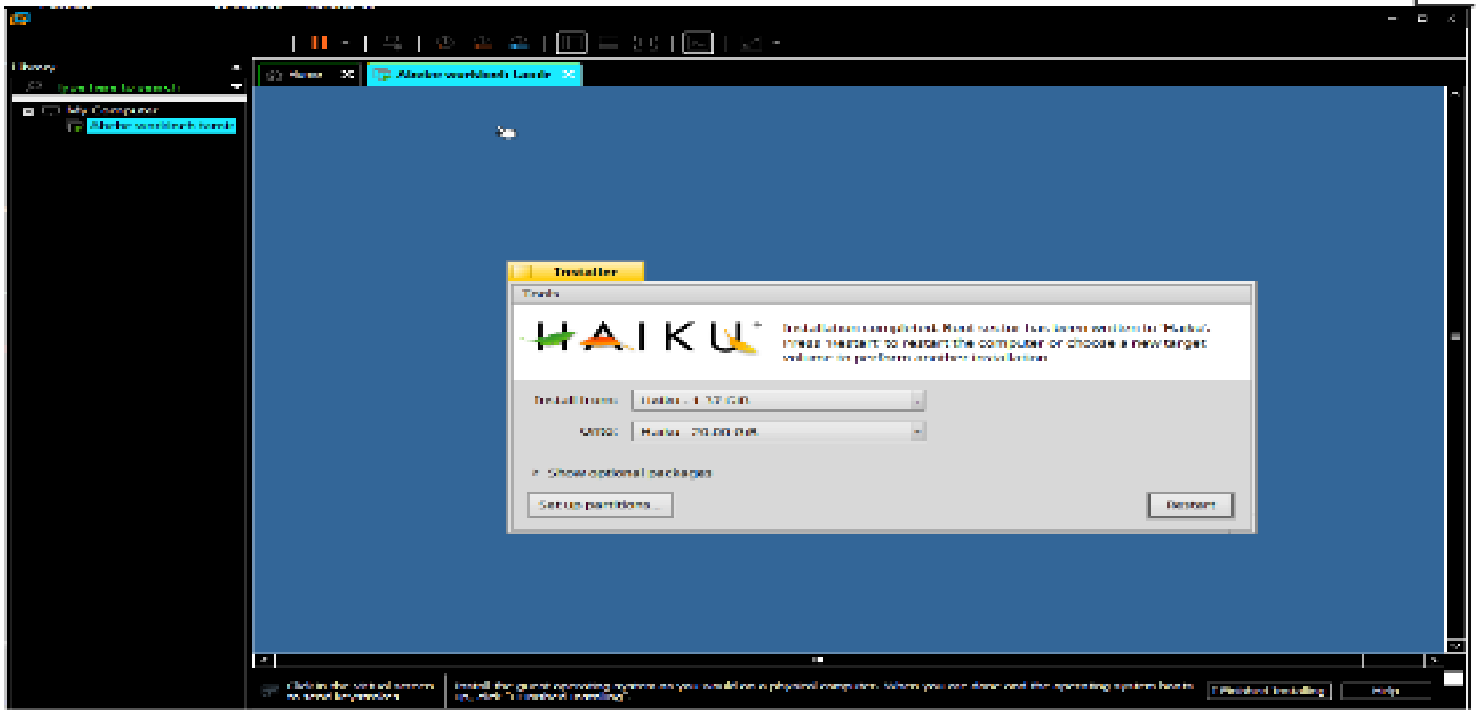


Choose the disk you want to install onto from the pop-up menu. Then click 'Begin'.



Performing installation.

Step 8: click "Restart"



Installer

Installation completed. Boot sector has been written to 'Haiku'. Press Restart to restart the computer or choose a new target volume to perform another installation.

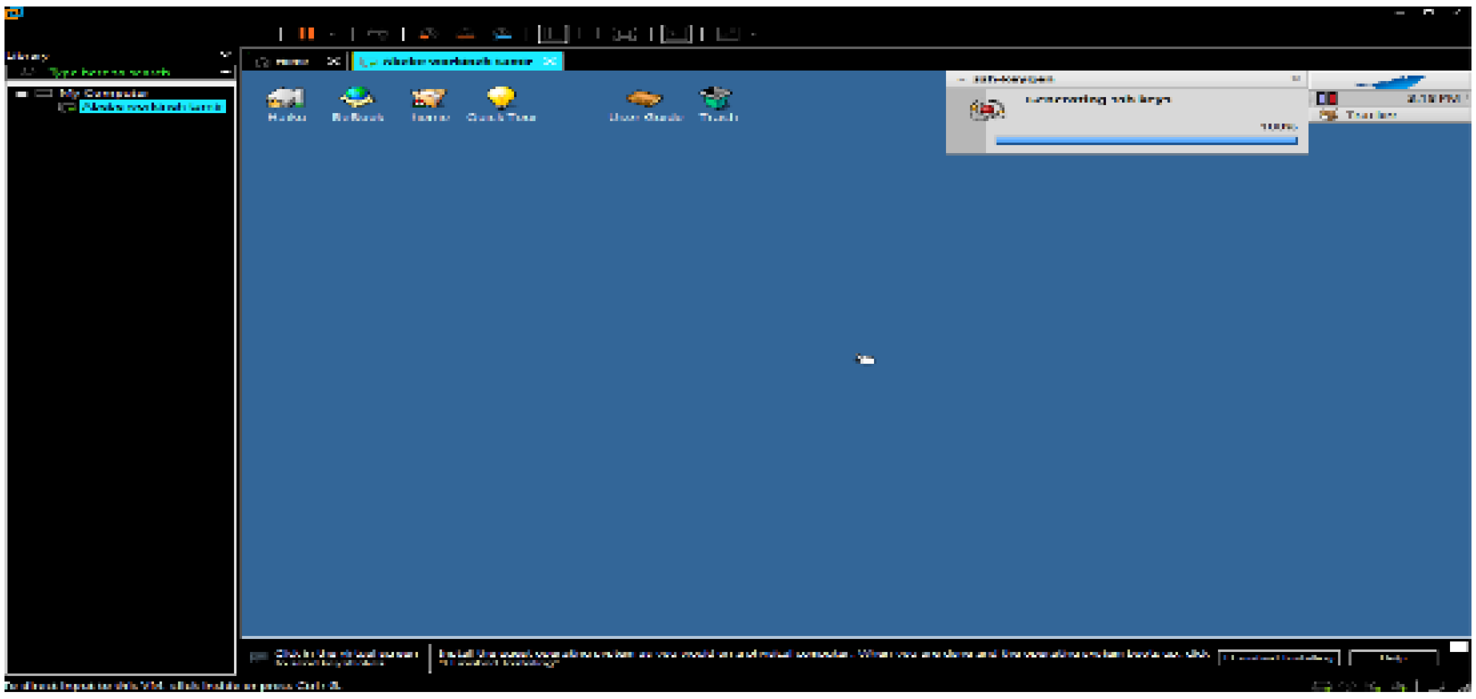
Install from: Haiku - 1.37 GiB

Onto: Haiku - 20.00 GiB

Show optional packages

Click in the virtual screen to send keystrokes

Install the guest operating system as you would on a physical computer. When you are done and the operating system boots up, click "I Finished Installing".



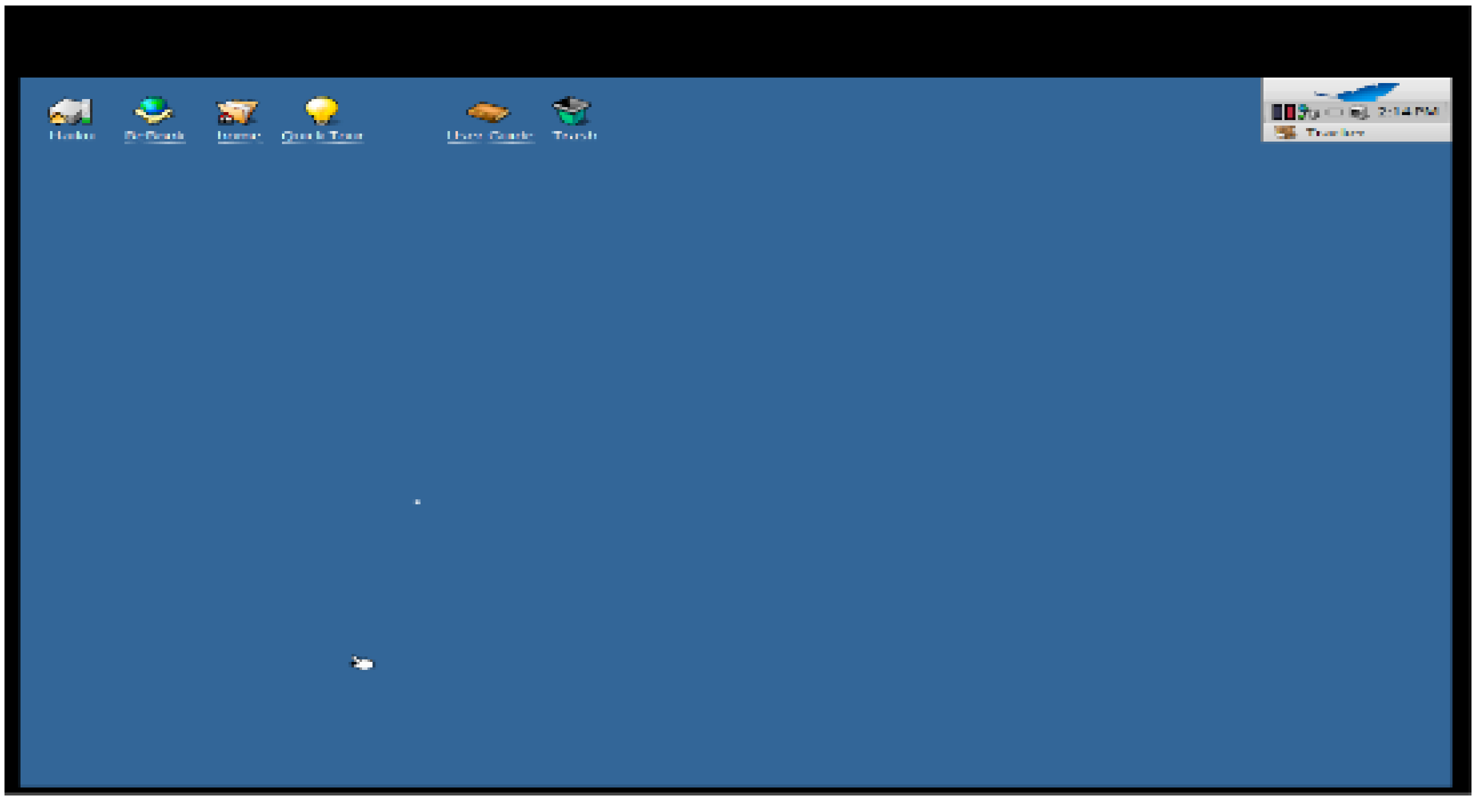
Generating ssh keys

100%

Click in the virtual screen to send keystrokes

Install the guest operating system as you would on a physical computer. When you are done and the operating system boots up, click "I Finished Installing".

To direct input to this VM, click inside or press Ctrl+G.



Problem faced during the installation process

When I tried installing the Haiku operating system on VMware Workstation Pro 17, ev erything seemed to go smoothly—until I reached the point where I needed internet acc ess. I noticed that the OS wasn't connecting to the network at all. I couldn't open a bro wser, download updates, or even ping an external address. At first, I thought it might be a bug in Haiku, but after a bit of digging, I realized the problem was related to VM ware's network settings and driver compatibility.

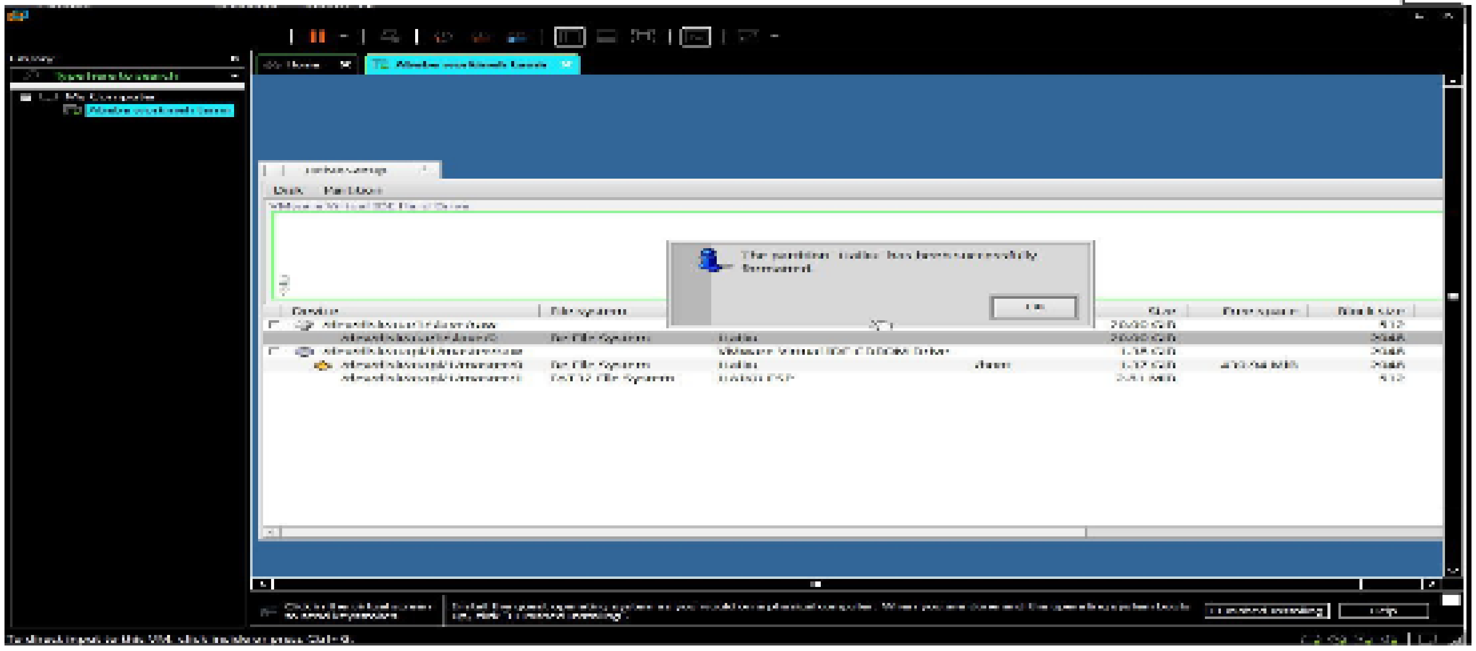
Solution of this problem

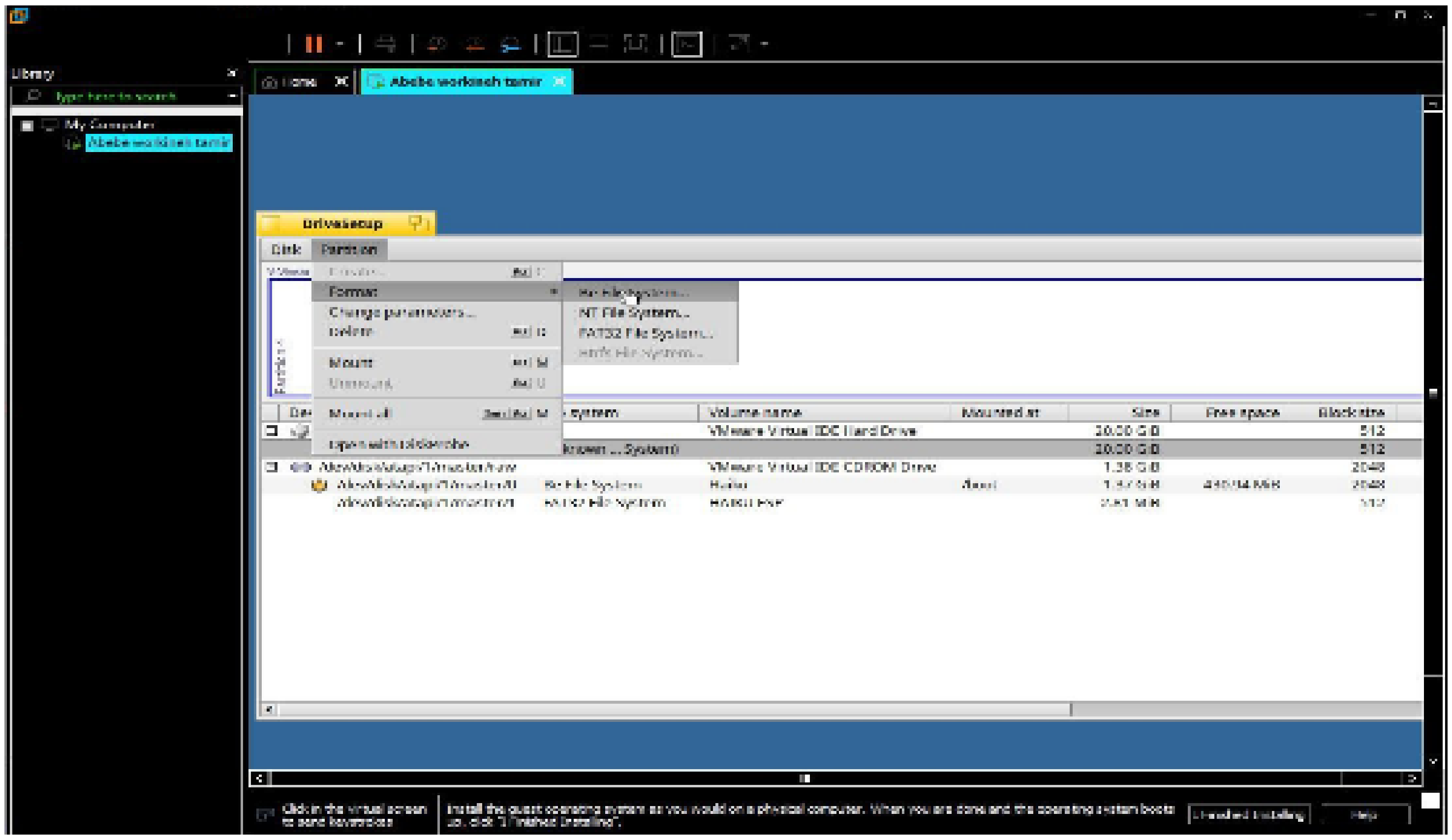
To fix the network problem in Haiku OS on VMware Workstation Pro 17, shut down th e VM and go to its settings. In the Network Adapter section, change the adapter type t o Intel PRO/1000 MT Desktop (e1000) and set the network connection to NAT or Brid ged. Start the VM, open Network Preferences in Haiku, and enable DHCP. This should connect Haiku to the internet. If it still doesn't work, try using the latest Haiku nightly build.

5.File system support

The Haiku operating system primarily uses its native BFS (Be File System), but it also supports other file systems for compatibility.

BFS (Be File System) – The Native File System


The partition "Haiku" has been successfully formatted.



Device	File system	Volume name	Mounted at	Size	Free space	Block size
[illegible]		VMware Virtual IDE Hard Drive		20.00 GiB		512
[illegible]	[illegible]			20.00 GiB		512
/dev/disk/atapi/1/master/raw		VMware Virtual IDE CDROM Drive		1.38 GiB		2048
/dev/disk/atapi/1/master/0	Be File System	Haiku	/boot	1.37 GiB	440.04 MiB	2048
/dev/disk/atapi/1/master/1	FAT32 File System	HAIKU ESP		2.81 MiB		512

Why?

- Designed for high performance and metadata handling, BFS was originally developed for BeOS, which Haiku is based on.
- Supports 64-bit file sizes, journaling (for crash resilience), and extended attributes (for rich metadata).
- Optimized for real-time multimedia operations, a key feature of Haiku's design philosophy.
- Indexed attributes allow fast searching and database-like functionality.

6.Advantage and disadvantage of haiku OS

Advantages:

Speed and Performance:

Haiku OS is known for being fast, especially on older or less powerful hardware. This is d
ue to its lightweight nature and efficient design.

User-Friendly Interface:

Haiku's interface is designed to be intuitive and easy to use, inspired by BeOS. Many user
s find it similar to the classic Mac OS

Open Source:

Haiku OS is an open-source project, meaning its code is freely available for anyone to ins
pect, modify, and distribute. This fosters community involvement and allows for continu
ous improvement.

Lightweight:

Haiku OS is designed to consume fewer system resources, making it suitable for older h
ardware or resource-constrained environments.

Unique Features:

Haiku OS has a unique file system (BFS) and other features that set it apart from other o
perating systems.

Lower Barrier to Entry for Contributing:

The Haiku community is known for being welcoming, making it easier for new contributo
rs to get involved in the development process.

Disadvantages:

Smaller Software Ecosystem:

Compared to Windows or Linux, Haiku has a smaller selection of available software.

Limited Hardware Support:

While Haiku OS aims for broad hardware compatibility, it may not offer the same level of support for all devices as more mature operating systems.

Smaller Community:

The Haiku community, while active, is smaller than those of Windows or Linux, which ca n lead to fewer resources and support.

Less Mature than Mainstream OSes:

Haiku OS is still under development and has not reached the level of maturity of mainstr eam operating systems like Windows or Linux.

7.conclustion

Installing Haiku OS on VMware Workstation Pro 17 offers a convenient and risk-free way to experience this lightweight, BeOS-inspired operating system. The process is relatively simple, with the Haiku ISO booting smoothly in VMware, and the installation wizard guidi ng users through an uncomplicated setup. Since Haiku is designed to be efficient, it runs well even with modest VM resources, such as 1-2GB of RAM and a single CPU core, maki ng it an excellent option for testing and development. However, there are some limitation s to consider, including VMware's lack of full driver support for certain features like 3D ac celeration and audio, which may not work optimally. Additionally, Haiku does not have V Mware Tools or guest additions, meaning seamless integration features like dynamic res olution scaling and shared folders are unavailable. Despite these drawbacks, VMware re mains a practical choice for exploring Haiku's unique environment, particularly for develo pers and enthusiasts who want to experiment without altering their primary system. For t hose seeking better performance or more complete hardware support, running Haiku on native hardware or dual-booting may be preferable. Overall, VMware Workstation Pro 17 provides a stable and accessible platform for discovering Haiku OS, even if some advan ced functionalities are limited in a virtualized setup.

8.Future outlook

The future outlook for Haiku OS appears promising, particularly as a lightweight, open-s ource alternative for users seeking a fast, responsive, and user-friendly operating system. With its BeOS-inspired design, Haiku excels in efficiency and simplicity, making it ideal f or legacy hardware, media production, and niche computing needs. Continued developm