UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería de Computadores

Trabajo Fin de Grado

Monitor para máquinas virtuales para laptops

UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería de Computadores

Trabajo Fin de Grado

Monitor para máquinas virtuales para laptops

Autor: Rubén Morgado Berrocal

Tutor: Francisco Marcelino Andrés Hernández

ÍNDICE GENERAL DE CONTENIDOS

Contenido

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El objetivo principal de este trabajo es habilitar la posibilidad de tener al mismo tiempo varios sistemas operativos sin emplear grandes cantidades de tiempo y sin disponer de unos conocimientos avanzados en el campo de la virtualización.

Este software libera al usuario de la necesidad de gestionar un sistema operativo base, desde el cual poder virtualizar otros sistemas, o de la tarea de instalar varios físicamente y la obligación de reiniciar el sistema para poder cambiar de uno a otro. Esta funcionalidad permite la simultaneidad de sistemas operativos sobre un mismo hardware.

Actualmente, la información almacenada en los ordenadores personales es mínima, toda la información sensible es guardada en la nube, lo que provoca que el almacenamiento local esté destinado a aplicaciones. Debido a este procedimiento, la dependencia con los sistemas operativos se ha visto en declive, haciendo que cuando surge un problema sea más sencillo eliminar por completo y reinstalar el sistema que intentar solucionar el problema.

Estas acciones dejan un sistema local dónde lo único que se almacena es el sistema operativo, su configuración y las aplicaciones. Con las herramientas actuales de virtualización y un desarrollo adecuado se puede lograr un sistema que ayude a los usuarios a realizar una administración de sus propios sistemas de manera sencilla.

1. INTRODUCCIÓN

En este proyecto se ha desarrollado un sistema software capaz de ejecutar simultáneamente varios sistemas operativos sobre el mismo ordenador.

Este desarrollo se basa en dos puntos fundamentales de la virtualización: la independencia respecto al sistema operativo y la transparencia respecto al hardware.

Haciendo uso de las herramientas actuales de virtualización se ha logrado una abstracción completa de todos los componentes del ordenador, lo que permite que dichos componentes puedan ser administrados como unidades lógicas, ficheros, despreocupándonos de la capa física.

Agregando una capa que facilita una interacción sencilla con estas unidades lógicas, interfaz gráfica, obtenemos un sistema potente con una gran flexibilidad y con un tiempo de aprendizaje mínimo.

El sistema es completamente independiente, no necesita programas de terceros, actualizaciones manuales… No es necesaria ninguna intervención, más que la de su instalación guiada. Una vez instalado, el usuario podrá comenzar a hacer uso del sistema a través de su interfaz.

2. OBJETIVOS

El objetivo principal de este trabajo se asienta en la posibilidad de gestionar uno o varios sistemas operativos independientemente del hardware sobre el que estén y en caliente. Los sistemas operativos deberán ser instalados, eliminados y modificados con el equipo arrancado, además para alternar entre ellos sólo será necesario una combinación de teclas, como si de una ventana se tratara.

Objetivos:

- Abstracción del hardware
- Transparencia de las funciones ejecutadas.
- Administración de las máquinas virtuales.
- Interacción sencilla con el usuario.

3. ANTECEDENTES / ESTADO DEL ARTE

Las primeras máquinas virtuales fueron creadas por IBM en los años 60. El primer ordenador diseñado para la virtualización fue el mainframe IBM S/360 Modelo 67, como una forma de compartir sistemas principales.

Este concepto todavía se sigue aplicando a los sistemas actuales de IBM, aunque dicho concepto sobre una máquina virtual se ha ampliado y aplicado a muchas áreas fuera de la virtualización.

¿Qué es una máquina virtual?

Una máquina virtual es un emulador de dispositivos físicos y lógicos de una máquina que nos sirve para un ordenador común.

¿Para qué sirve?

Su utilidad es la posibilidad de utilizar varios sistemas operativos de forma simultánea sobre un mismo hardware.

Origen de la máquina virtual

El primer sistema operativo que utilizó virtualización fue el Conversational Monitor System (CMS). A comienzos del año 1970 IBM introdujo máquinas virtuales que ejecutaban múltiples sistemas operativos de usuario individual.

El área de virtualización que IBM hizo popular se le conoce como virtualización de plataforma (o sistema). En esta forma la plataforma de hardware subyacente es virtualizada con el fin de ser compartida con varios sistemas operativos y múltiples usuarios.

Otro objetivo de la máquina virtual es que ofrece una propiedad de independencia de máquina. A esta forma se le llama virtualización de aplicación (o proceso). Consiste en crear un entorno para una aplicación, haciéndolo independiente de la parte física.

Historia y evolución de la máquina virtual

Uno de los primeros usos de la virtualización de aplicación fue en 1960 para el BASIC COMBINED PROGRAMMING LANGUAGE (BCPL). El BCPL es un lenguaje imperativo desarrollado por Martin Richards en la Universidad de Cambridge [6].

El BCPL tiene un lenguaje de alto nivel y el código intermedio que genera recibe el nombre de O-code. Este código intermedio podía ser interpretado por una máquina física o compilado para el lenguaje de la máquina nativa del host. Como al abstraer el O-code, este podría ser interpretado por una variedad de host y también podía ser compilado por la máquina nativa, permitía el desarrollo de un compilador y varios compiladores que traducen el O-code para la máquina nativa. Esta independencia hizo portátil el lenguaje entre máquinas.

A principios de1970, se implementó la máquina virtual para la ejecución de Pascal, pero ya compilado. La representación intermedia se llamó P-code. Esto buscó la independencia del hardware subyacente para simplificar el desarrollo del compilador Pascal [6].

En 1972, Xerox PARC introduce el lenguaje de Smalltalk que dependía de una máquina virtual para ejecutarse [6]. Este lenguaje y el P-code influenciaron en uno de los lenguajes basados en máquinas virtuales más prominentes que existe actualmente: JAVA.

Java apareció por primera vez en 1995 y fue desarrollado por Sun Microsystem. Este lenguaje desarrolló la idea de una programación que es independiente de la plataforma mediante la JAVA VIRTUAL MACHINE [4,6]. Por lo tanto, Java incrementó el conocimiento sobre las tecnologías de las máquinas virtuales y dio paso a tecnologías que conectaron la interpretación de la ejecución nativa usando técnicas JIT (justo a tiempo).

Las máquinas virtuales actuales

El objetivo actual de las máquinas virtuales es proporcionar una abstracción al host físico, esta idea va evolucionando y encuentra aplicación. A continuación, aparecerán algunas de las soluciones de fuente abierta más recientes:

- DALVIK VM [7,10,11]: Es una máquina virtual que utiliza la plataforma para dispositivos móviles. Esta máquina virtual permite ejecutar aplicaciones (programadas en java). DVM sacrifica la portabilidad de java para poder crear aplicaciones con un menor consumo de energía y un mejor rendimiento.

- PARROT [8,9]: Es una máquina virtual basada en registros que pretende la ejecución de programas escritos en diferentes lenguajes dinámicos. Esta máquina virtual tiene como objetivo hospedar lenguajes clientes y permitir la interpretación entre ellos.

Virtualización

A lo largo de la historia de la informática la tecnología de la virtualización ha ido variando, pero siempre con un mismo objetivo y significado, la abstracción de los recursos de una computadora, llamada Hypervisor o VMM (Virtual Machine Monitor) [1].

La virtualización consiste en crear una capa entre el hardware de la máquina física (host) el sistema operativo de virtual (virtual machine, guest). Esta capa gestiona cuatro recursos principales de una computadora (CPU, memoria, red y almacenamiento) que reparte dinámicamente entre todas las máquinas virtuales creadas.

Tipos de virtualización

La virtualización se puede agrupar en tres grandes grupos:

- Virtualización completa: Esta es en donde la máquina virtual simula un hardware suficiente para permitir un sistema operativo huésped sin modificar (uno diseñado para la misma CPU) para ejecutar de forma aislada. Típicamente, muchas instancias pueden ejecutarse al mismo tiempo. Este enfoque fue el pionero en 1966 con CP-40 y CP [-67]/CMS, predecesores de la familia de máquinas virtuales de IBM [12,13].
    - Oracle VM VirtualBox
    - Virtual Iron
    - Adeos
    - XenServer

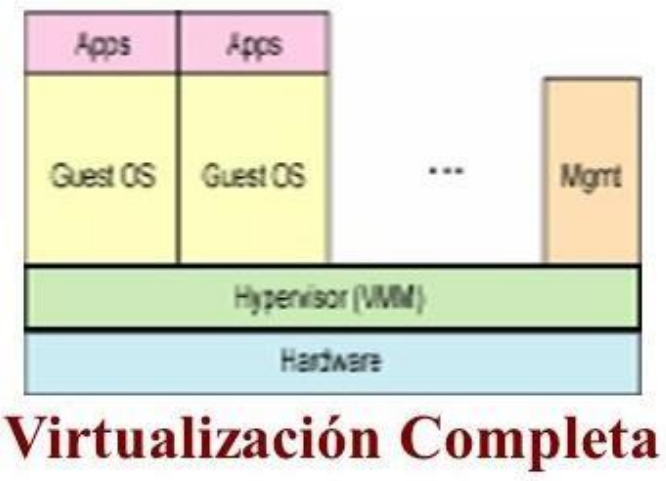

Ilustración 1: Virtualización Completa

- Virtualización parcial: La máquina virtual simula múltiples instancias de gran parte (pero no de todo) del entorno subyacente del hardware, particularmente los espacios de direcciones. Tal entorno acepta compartir recursos y alojar procesos, pero no permite instancias separadas de sistemas operativos huésped. Aunque no es vista como dentro de la categoría de máquina virtual, históricamente éste fue un importante acercamiento, y lo usaron en sistemas como CTSS, el experimental IBM M44/44X [12], y podría mencionarse que en sistemas como OS/VS1, OS/VS2 y MVS [12].

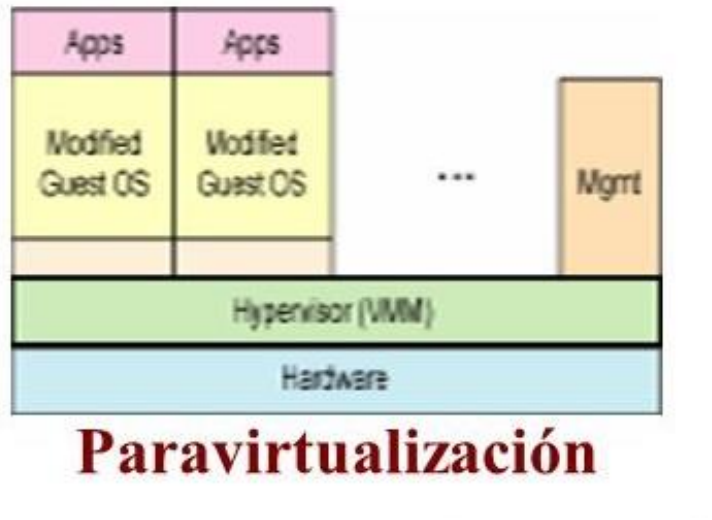

Ilustración 2: Paravirtualización

- Virtualización por S.O o Semi Parcial: La Virtualización de SO mejora el rendimiento, gestión y eficiencia. En la base reside un sistema operativo anfitrión estándar, como en el caso de Parallels Virtuozzo [12] que incluye Windows y un sistema con núcleo Linux. A continuación, encontramos la capa de virtualización, con un sistema de archivos propietario y una capa de abstracción de servicio de kernel que garantiza el aislamiento y seguridad de los recursos entre distintos contenedores. La capa de virtualización hace que cada uno de los contenedores aparezca como servidor autónomo. Finalmente, el contenedor aloja la aplicación o carga de trabajo.

Ilustración 3: Virtualización de sistema Operativo

Virtualización actual

En la actualidad, un usuario que quiera usar esta tecnología, en general, suele usar un programa de terceros sobre su sistema operativo, como pueden ser VirtualBox, VMware, Hyper-V… Estos softwares tienen algunas desventajas que para un usuario medio pueden ser grandes barreras como son la complejidad, están diseñados para tener un cierto grado de conocimientos, el coste, muchas de estas implementaciones están destinadas a grandes empresas y, por último, el rendimiento, ejecutar un sistema operativo sobre otro requiere de un nivel de recursos muy alto, lo que implica tener una computadora potente y esto, a su vez, son más costes.

4. MATERIAL Y MÉTODO

Para lograr los objetivos establecidos disponemos de los siguientes recursos:

- **Sistema base**, que consume los mínimos recursos, sobre el que se implementa el sistema completo.
- **Software que permite la virtualización del hardware**.
- **Interfaz** que permite una interacción sencilla con el usuario.

Esquema de las capas del sistema

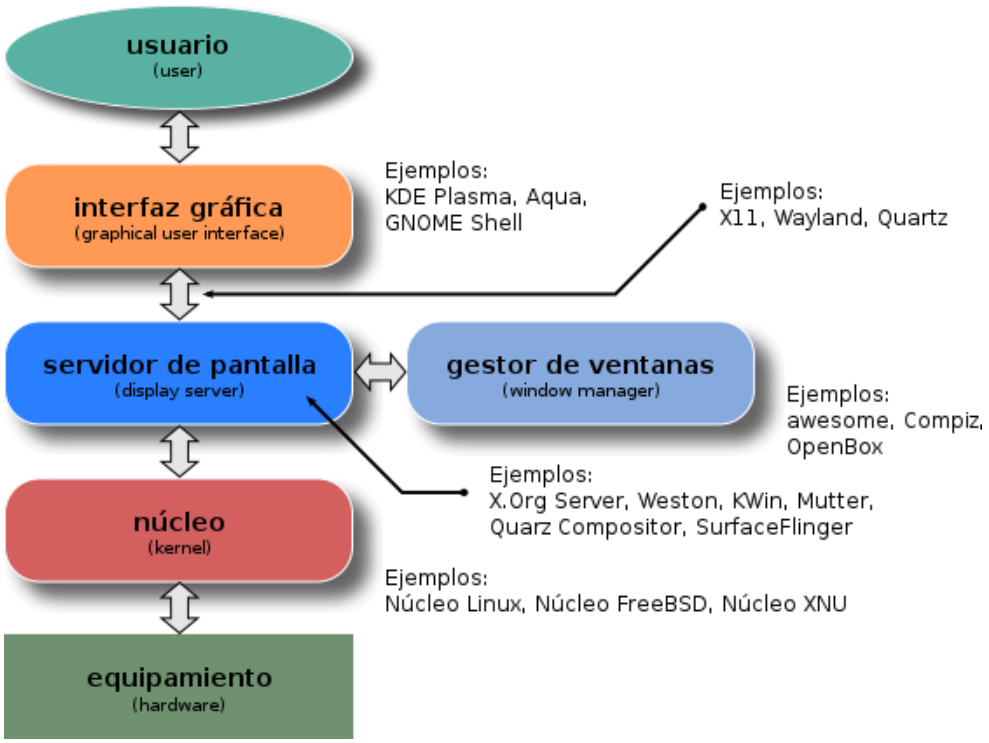

Ilustración 4: Esquema de las capas de la interfaz gráfica de usuario

- **Núcleo:** Linux
- **Servidor de pantalla**: X.Org.

X.Org es una implementación de código abierto del sistema de ventanas X Windows y desarrollado por la X.Org Foundation. La mayor parte de las distribuciones GNU/Linux han optado por este servidor gráfico.

Además, X.Org se integra dentro de los estándares descritos en la Freedesktop, cuyo objetivo es establecer las aplicaciones y especificaciones de los entornos de escritorio libres.

- **Gestor de ventanas:** Openbox.

  Openbox es un gestor de ventanas libre para el sistema de ventanas X, disponible bajo licencia GPL. Está diseñado para ser rápido y consumir una mínima cantidad de recursos. Además, Openbox cumple plenamente las especificaciones ICCCM y EWMH.

Libvirt

Libvirt nos ofrece una API agnóstica a hipervisores para administrar con seguridad sistemas operativos huéspedes que se están ejecutando en un host.

La API Libvirt se divide en varias secciones: API de conexión al hipervisor, API de dominio, API de red, API de volumen de almacenamiento y API del bloque de almacenamiento.

Virsh

Virsh es una herramienta de línea de comando para administrar a los huéspedes y al hipervisor. La herramienta Virsh se crea en la API de administración Libvirt y funciona como una alternativa para el comando xm y el Administrador de huésped gráfico (virt-manager)

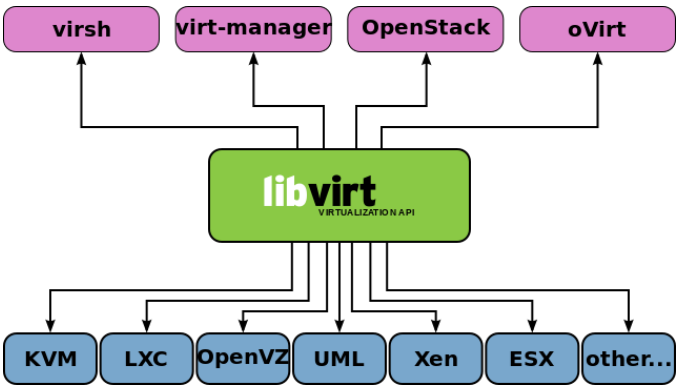

Ilustración 5: Esquema Virsh

Arquitectura básica

Libvirt [1] es un conjunto de interfaces de programación de aplicaciones que ha sido diseñado para ser utilizado por una aplicación de gestión.

Utiliza un mecanismo específico basado en hipervisores y se comunica con cada uno cuando están disponibles para realizar solicitudes de API.

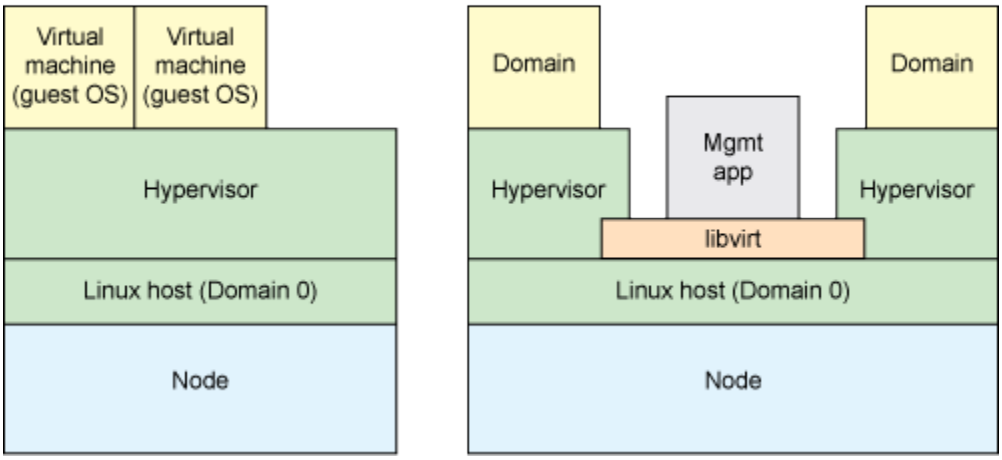

Ilustración 6: Arquitectura básica y Libvirt

Medios de control

Al utilizar libvirt se posee dos medios distintos de control.

- Local:

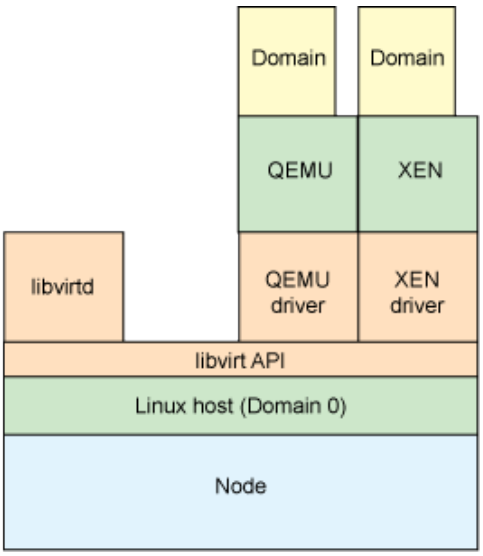

Ilustración 7: Medio de control local

Usada para administrar las máquinas locales con virsh.

- Remota:

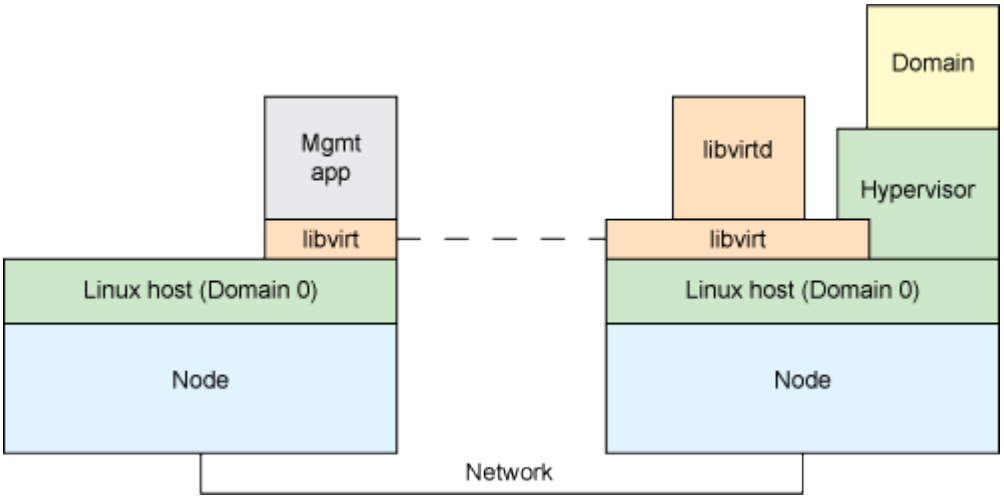

Ilustración 8: Medio de control remoto

Usada para administrar las máquinas virtuales remotas con Qemu.

QEMU

QEMU [1] es un emulador de procesadores que se basa en la conversión de código binario de la arquitectura fuente en código que puede entender la arquitectura huésped. Además, QEMU también tiene capacidades de virtualización dentro de un sistema operativo.

Su principal objetivo es emular un sistema operativo dentro de otro sin tener que realizar particiones en el disco duro.

Hipervisores que soporta Libvirt

Hipervisor	Descripción
Xen	Hipervisor para arquitecturas IA-32, IA-64 y PowerPC 970
QEMU	Emulador de plataforma para diversas arquitecturas
Máquina virtual basada en el kernel (KVM)	Emulador de plataforma Linux
Contenedores de Linux (LXC)	Contenedores de Linux (ligeros) para la virtualización del sistema operativo
OpenVZ	Virtualización a nivel sistema operativo basado en el kernel Linux
VirtualBox	Hipervisor para virtualización de x86
User Mode Linux	Emulador de plataforma Linux para diversas arquitecturas
Prueba	Driver de prueba para un hipervisor falso
Almacenamiento	Drivers de bloque de almacenamiento (disco local, disco de red, volumen iSCSI)

Tabla 1: Hipervisor que soporta Libvirt

Lenguaje de programación

El lenguaje de programación empleado es **Python3** [15].

Este lenguaje es multiparadigma, ya que soporta programación orientada a objetos, programación imperativa y programación funcional. Además, es un lenguaje interpretado, usa un tipado dinámico y es multiplataforma.

Librerías Python

- **Librería GTK (GIMP Tool Kit)** [16]: es una biblioteca del equipo GTK+, la cual contiene los objetos y funciones para crear la interfaz gráfica de usuario. Maneja widgets como ventanas, botones, menús, etiquetas, deslizadores, pestañas, etc.
- **Librería Paramiko** [17]: es un módulo que implementa todos los protocolos de seguridad de SSH2 (tanto encriptación como autenticidad).
- **Otras librerías: json, gi, subprocess**

Sistema operativo base

El sistema operativo base está generado a partir de Ubuntu server 16.10. Es una distro personalizada mediante la herramienta Cubic, la cual te permite abrir una imagen ISO y mediante una consola integrar y configurar completamente la imagen.

En la imagen se han integrado los siguientes componentes relevantes:

- Hipervisor
- Xorg
- Openbox

Además de todas las dependencias, paquetes menos significativos y configuraciones necesarias.

Instalación

La instalación se realiza mediante la imagen ISO personalizada, su modo de instalación es mediante un USB o disco. Los pasos para su instalación son los mismo que para instalar Ubuntu server 16.10, todos los paquetes y configuraciones necesarias ya están integrados y automatizados dentro de la imagen ISO.

Modos

El monitor de máquinas virtuales dispone de un interfaz con la cual el usuario puede interactuar de forma gráfica y sencilla. Esta interfaz dispone de dos modos de uso:

- Modo básico
- Modo avanzado

Modo básico

Es el modo principal del programa, es un interfaz simplificado con la que el usuario pueda estar cómodo y realizar las funciones más importantes de una forma rápida y sencilla.

Ilustración 9: Modo básico del programa

Acciones que permita al usuario:

- Crear máquina virtual: crea una máquina mediante un menú de configuración mínimo:

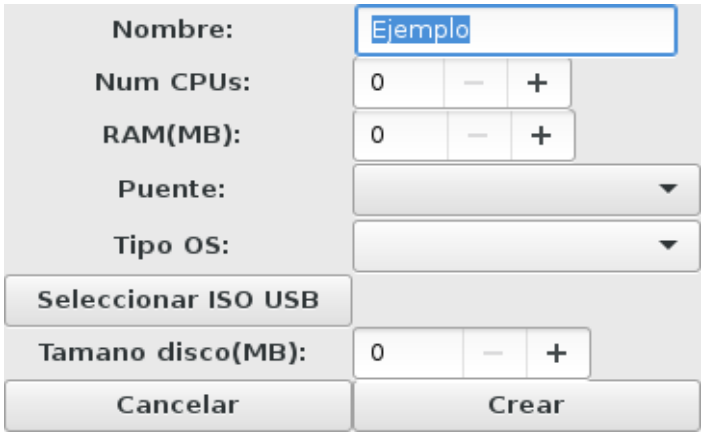

Ilustración 10: Crear máquina virtual

- Las imágenes ISO de los sistemas operativos se proporcionan mediante una memoria USB. La aplicación detectará automáticamente el dispositivo y proporcionará una lista con las imágenes ISO disponibles en la memoria.
- Duplicar máquina virtual: duplica una máquina ya existente.
- Activar/desactivar autoarranque: si está activado la máquina virtual arrancará y se visualizará al inicio del sistema.
- Abrir modo de opciones avanzadas: cambia al modo de opciones avanzas.

Modo avanzado

Este modo, aunque también simplificado, está dirigido a usuarios más avanzados. Dispone de una amplia gama de acciones para administrar tanto máquinas virtuales locales como remotas.

Ilustración 11: Acciones para administrar máquinas virtuales remotas y locales

Dispone de un menú siempre disponible en la barra superior que permite las siguientes acciones:

- Opciones básicas: cambia al modo de opciones básicas
- Conectar servidor: permite una conexión con un servidor remoto Linux.
- Apagar Sistema: apaga el sistema.
- Reiniciar Sistema: reinicia el sistema.
- Conexiones: permite configurar las conexiones de red.

Al seleccionar un servidor o el servidor local (localhost) aparecen nuevas opciones de bajo del menú principal que permiten administrar los servidores (crear máquina virtual, desconectar, reconectar y eliminar).

Los datos de los servidores son almacenados y recuperados en cada inicio para mantenerlos en el mismo estado en el que están al cierre de la aplicación.

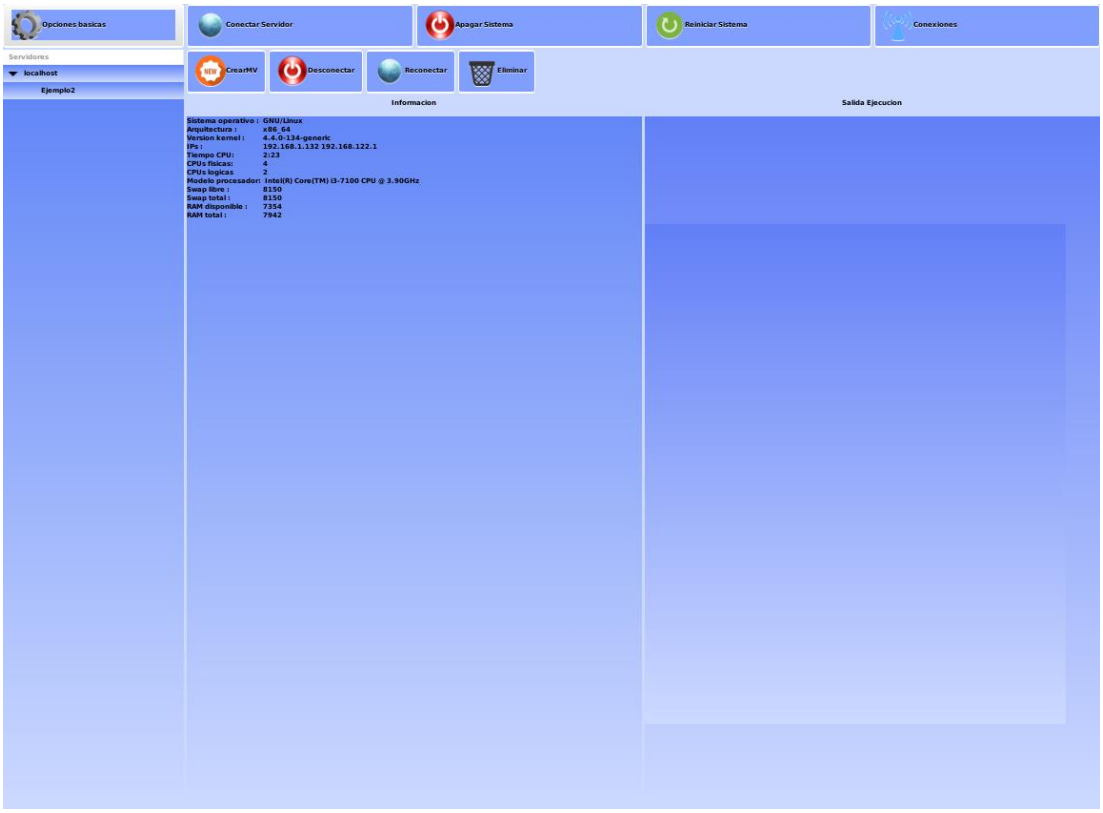

Ilustración 12: Funciones de un servidor

Al seleccionar una máquina aparecerán nuevas opciones de bajo del menú principal que permiten administrar las máquinas virtuales (estado, salida y entrada):

Cambiar estado: apagar, iniciar, reiniciar… una máquina virtual.

Ilustración 13: Cambio de estado

Salida: visualizar, crear snapshot, clonar…

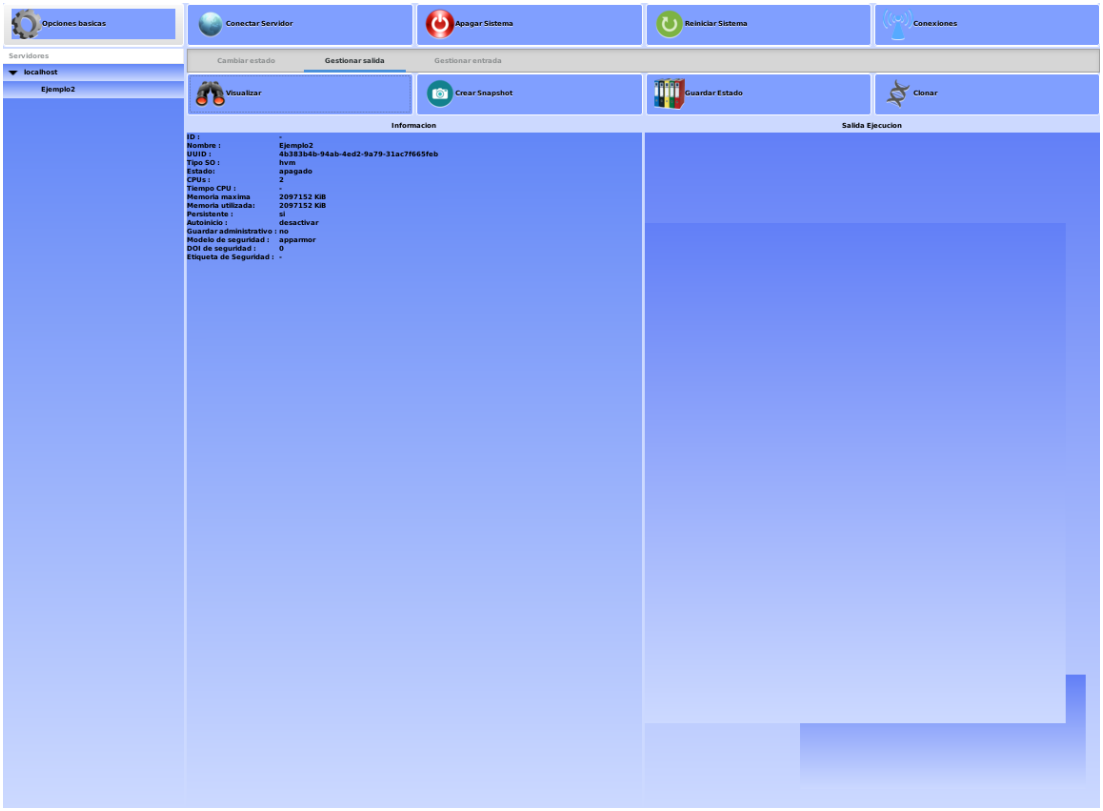

Ilustración 14: Salida

Entrada: cargar snapshot, cargar estado, eliminar…

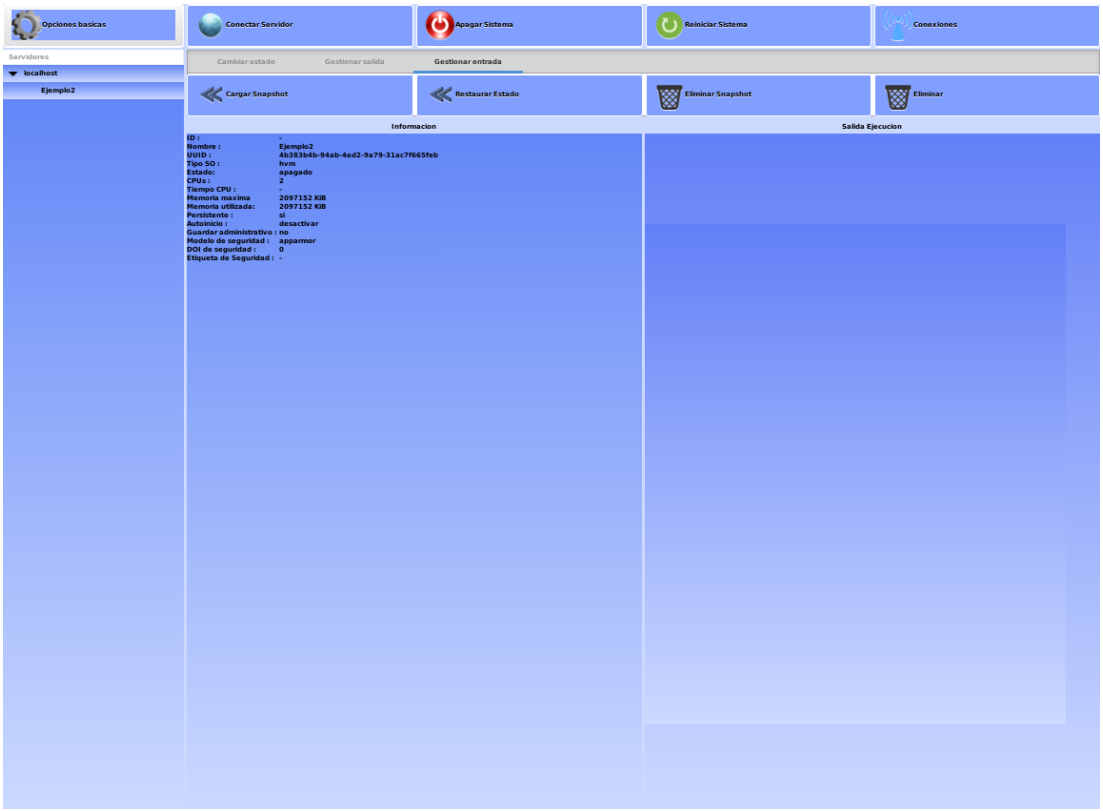

Ilustración 15: Entrada

5. RESULTADOS Y DISCUSIÓN

El resultado es una distribución personalizada que te permita la instalación de un sistema completo para la virtualización de múltiples sistemas operativos ejecutándose simultáneamente sobre un mismo hipervisor y la posibilidad de establecer conexiones remotas con servidores Linux.

Rompe la dependencia creada entre el sistema operativo y el usuario.

Permite una libre administración de los sistemas operativos instalados (eliminar, clonar, crear copias de seguridad…).

Este sistema tiene un coste de recursos mínimo que se restan de los recursos totales disponibles para la creación de nuevas máquinas virtuales, poco más de 200 MB de RAM y 6 GB de almacenamiento.

Gracias a los dos modos de los que dispone el sistema, modo básico y modo avanzado, permite un uso tanto a usuarios medios como expertos, lo que cubre las necesidades de la mayoría de los usuarios.

6. CONCLUSIONES

La virtualización es una herramienta muy potente y versátil, su uso está dirigido, en su gran mayoría, a entornos empresariales o al sector informático. Su uso en ordenadores personales es mínimo. Si esta tecnología se aplicara y diseñara para todo tipo de usuarios, se podría mejorar la experiencia de estos proporcionándoles una herramienta con la que puedan sentirse cómodos para administrar sus propios sistemas sin necesidad de grandes conocimientos.

El futuro de los sistemas y de la virtualización está en la nube, en unos años todos los datos y la mayor parte de la computación se realizará desde la nube, dejando en los sistemas físicos de los usuarios unos recursos mínimos, centrados en la visualización y la conexión. De esta manera el usuario será completamente ajeno a tareas como mantenimiento, actualizaciones, resolución de errores, creación de copias de seguridad… pudiendo centrarse en sus tareas de usuario.

Hasta que las conexiones evolucionen lo suficiente, para soportar el tráfico masivo que los usuarios generarán, es necesario un paso intermedio. Ese paso intermedio es un sistema de virtualización local, que permita comenzar a desentender al usuario del hardware.

Con este proyecto se ha logrado realizar ese paso intermedio, crear una herramienta que despreocupe al usuario del hardware y con la cual pueda administrar de forma transparente todos los recursos que este le proporciona.

Posibles mejoras

- Interfaz: la interfaz podría mejorarse para hacerla más atractiva para el usuario.
- Conexiones remotas: actualmente sólo dispone de conexiones remotas SSH, podría incluirse conexiones RDP, XEN…
- Gestión y monitorización de recursos: un monitor en el cuál el usuario pueda variar los recursos que asigna a cada máquina, los recursos totales y los que aún están libres.
- Aceleración hardware mediante GPUs: se podrían usar las GPUs para acelerar las visualizaciones de las máquinas virtuales para dotar de una mejor experiencia a los usuarios.
- Ayuda al usuario: Agregar un pequeño manual de guía al que puedan recurrir los usuarios para resolver las dudas que les surgen durante el uso del sistema.

REFERENCIAS BIBLIOGRÁFICAS

[1] Anatomía de la biblioteca de virtualización libvirt, Ibm.com [online]. Available from: https://www.ibm.com/developerworks/ssa/linux/library/l-libvirt/index.html

[2] Sistema de ventanas X, Es.wikipedia.org [online]. Available from: https://es.wikipedia.org/wiki/Sistema_de_ventanas_X

[3] Virtualbox vs. Vmware vs. Hyper-V – Cybrary, Cybrary [online]. Available from: https://www.cybrary.it/0p3n/virtualbox-vs-vmware-vs-hyper-v/

[4] Historia de la Virtualización, Virtualizacion [online]. Available from: http://www.virtualizacion.com/virtualizacion/

[5] Virtualización, Es.wikipedia.org [online]. Available from: https://es.wikipedia.org/wiki/Virtualizaci%C3%B3n

[6] Virtualización de aplicaciones pasado y futuro, Ibm.com [online]. Available from: https://www.ibm.com/developerworks/ssa/linux/library/l-virtual-machine-architectures/index.html

[7] Dalvik, Es.wikipedia.org [online]. Available from: https://es.wikipedia.org/wiki/Dalvik

[8] Máquina virtual Parrot, Es.wikipedia.org [online]. Available from: https://es.wikipedia.org/wiki/M%C3%A1quina_virtual_Parrot

[9] Android Runtime, Es.wikipedia.org [online]. Available from: https://es.wikipedia.org/wiki/Android_Runtime

[10] 2.3. Dalvik VM - Software de Comunicaciones, Sites.google.com [online]. Available from: https://sites.google.com/site/swcuc3m/home/android/generalidades/dalvikvm-1

[11] Dalvik bytecode | Android Open Source Project, Android Open Source Project [online]. Available from: https://source.android.com/devices/tech/dalvik/dalvik-bytecode

[12] Juan Quesada, Proyecto, Dis.um.es [online]. Available from: http://dis.um.es/~lopezquesada/documentos/IES_1718/LMSGI/curso/UT4/xhtml/xhtml22/html/index2.html

[13] Jonathan Echeverria, virtualizacion completa | | Jonathan Echeverría, Jonathanecheverria.com [online]. Available from: https://www.jonathanecheverria.com/tag/virtualizacion-completa

[14] Virtualización, Es.wikipedia.org [online]. Available from: https://es.wikipedia.org/wiki/Virtualizaci%C3%B3n

[15] Python 3.0 Release, Python.org [online]. Available from: https://www.python.org/download/releases/3.0/

[16] Apuntes FP, GTK y algunos apuntes básicos para entenderlo, entreunosyceros [online]. Available from: https://entreunosyceros.net/conceptos-gtk/

[17] Python – Conexión vía ssh utilizando la librería paramiko – Alberto Pascual, Alberto Pascual [online]. Available from: http://www.apascualco.com/python-conexion-via-ssh-utilizando-la-libreria-paramiko/

ANEXOS

Anexo I: manual de usuario