UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Estudio e instalación de un sistema de escritorio seguro basado en el particionado

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Ignacio Serna Merino
Tutor: Ismael Ripoll

Curso 2016-2017

Resumen

En este trabajo fin de grado se aborda la situación actual de los sistemas de protección digital, cada vez más ineficaces ante el creciente desarrollo de *software* malicioso. Se propone una solución basada en la virtualización de sistemas operativos sobre un equipo para dividir las diversas acciones a tomar en función a su peligrosidad. Aunque es posible realizar esta división mediante programas más comunes como VMware o VirtualBox, se escoge Qubes OS por ofrecer herramientas para una gestión ágil de los sistemas virtualizados, así como un mayor rendimiento en los mismos. Mediante una pequeña porción de *software* llamada hipervisor XEN, sobre la que se ejecuta Qubes, se consigue un entorno más seguro y eficiente que a través de otras soluciones de virtualización.

Palabras clave: Qubes OS, Linux, seguridad, virtualización

Resum

En aquest treball fi de grau s'aborda la situació actual dels sistemes de protecció digital, cada vegada més ineficaços davant el creixent desenvolupament de *softwarei* maliciós. Es proposa una solució basada en la virtualització de sistemes operatius sobre un equip per a dividir les diverses accions a prendre en funció de la seva perillositat. Encara que és possible realitzar aquesta divisió mitjançant programes més comuns com VMware o VirtualBox, s'escull Qubes OS per oferir eines per a una gestió àgil dels sistemes virtualitzats, així com un major rendiment en els mateixos. Mitjançant una petita porció de textit programari denominada hipervisor XEN, damunt la qual s'executa Qubes, s'aconsegueix un entorn més segur i eficient que a través d'altres solucions de virtualització.

Paraules clau: Qubes OS, Linux, securitat, virtualització

Abstract

This end-of-degree paper addresses the current situation of digital protection systems, increasingly ineffective in the face of the growing development of malicious software. It proposes a solution based on the virtualization of operating systems on a computer to divide the various actions to be taken in function of its dangerousness. Although it is possible to perform this division through more common programs like VMware or VirtualBox, Qubes OS is chosen because it offers tools for an agile management of the virtualized systems, as well as greater performance. By means of a small portion of software called XEN hypervisor, on top of which Qubes is executed, a more secure and efficient environment is achieved than through other virtualization solutions.

Key words: Qubes OS, Linux, security, virtualization

Índice general

Índice de figuras

CAPÍTULO 1
Introducción

La seguridad en los computadores modernos se ha convertido en una necesidad, desde un equipo personal hasta grandes máquinas empresariales son objetivo de una gran cantidad de *software* malicioso. Esto ha sido así desde los inicios de la informática, no obstante este tipo de programas ha evolucionado para pasar de ser algo "molesto", que dejaba un rastro de publicidad y mensajes invasivos, a ser complejas piezas de *software* difíciles de eliminar, capaces de ocultarse para robar información, encriptar el almacenamiento y pedir grandes cantidades de dinero para su recuperación, e incluso tomar el control directo del sistema.

La realidad es que los mecanismos de seguridad típicos actuales, tales como programas antivirus y *anti-malware*, avanzan por detrás del desarrollo de las amenazas, luchando por mantenerse actualizados contra estas. Además, este tipo de *software* de protección normalmente consume una cantidad considerablemente alta de recursos del sistema, reduciendo el rendimiento del mismo.

Es por esto que se busca una alternativa para otorgar seguridad a un equipo, y en lugar de buscar algún tipo de *software* de seguridad, se estudia y se implementa una metodología sobre la que los propios sistemas operativos se fundamentan, el particionado o aislamiento. Esta técnica, en lugar de intentar prevenir o neutralizar un ataque, asume la posibilidad de un ataque satisfactorio pero concede la posibilidad de aislarlo, analizarlo e incluso eliminarlo de raíz.

Por tanto en este proyecto se aborda una implementación de la metodología del particionado mediante la virtualización de sistemas operativos. Para ello se escoge una distribución de Linux llamada Qubes OS, basada en el hipervisor Xen, que despliega un entorno multisistema basado en máquinas virtuales.

1.1 Motivación

El principal factor que despertó el interés para la elaboración de este proyecto es la creciente necesidad de seguridad en equipos de uso personal, cada vez más vulnerables a robos de información, acoso con publicidad, suplantación de identidad y otros muchos ataques. De ahí que se busque una solución adaptable a cada usuario, gratuita y a la vez fiable.

Un único clic en el archivo adjunto de un correo electrónico o en un ejecutable desconocido puede comprometer el sistema completo, poniendo en riesgo gran parte de la vida digital del usuario, incluyendo contraseñas, cuentas bancarias, trabajos... Partiendo de la base de que es imposible que esto no ocurra ocasionalmente durante el uso común de un equipo, surge la necesidad de protegerse de estos ataques, no solo mediante meca-

nismos de seguridad sino particionando los distintos componentes de la vida digital de un usuario, de manera que una amenaza quede atrapada y limitada a la partición en la que se encuentra. Por tanto la solución aportada está basada en una técnica "permisiva", que asume el error humano y proporciona mecanismos rápidos y fáciles para eliminar una amenaza.

Por otro lado, se tiene en cuenta que el sistema debe ser seguro pero funcional; uno de los problemas actuales de cualquier distribución Linux es la escasez de *software* propietario de alta calidad, presente únicamente en sistemas operativos más comerciales, como Windows o Mac OS. Este inconveniente se tiene presente en la selección de la implementación escogida, siendo realmente fácil desplegar un sistema Windows virtualizado en un entorno seguro y controlado.

Se ha considerado la opción de emplear virtualización hospedada, mediante un programa de virtualización como VMWare o VirtualBox, sin embargo se ha escogido Qubes OS por sus ventajas en cuanto a seguridad y rendimiento. Mediante el uso de un hipervisor de tipo-1, Qubes es capaz de ofrecer una interacción más rápida con el *hardware*; además, para tomar control del equipo completo, es necesario atacar satisfactoriamente el propio hipervisor, una tarea realmente complicada. Además cuenta con la ventaja de ofrecer mecanismos totalmente seguros de transferencia de archivos y texto entre máquinas virtuales.

1.2 Objetivos

El objetivo principal del proyecto es estudiar la capacidad de otorgar seguridad a un sistema mediante el uso de la técnica del particionado, así como realizar una instalación práctica y establecer unas guías para promover un uso correcto del mismo.

Por otra parte, se pretende obtener un sistema seguro pero útil y funcional, con posibilidad de ejecutar programas propietarios presentes únicamente en sistemas de baja seguridad como Windows. Mediante el sistema operativo Qubes OS se consigue integrar distintos sistemas operativos virtualizados bajo una interfaz común, capaz de gestionarlos ágil y eficientemente.

CAPÍTULO 2
Ataques y prevención

Este capítulo tratará de conformar una visión global de cómo se estructura un sistema Linux y por qué es un sistema apto para el despliegue de ciertas medidas de seguridad contra amenazas que provengan tanto de la red como de un acceso físico al equipo. A su vez, se enumeran diversas distribuciones que añaden privacidad y seguridad a este tipo de sistemas, así como los tipos de ataques más típicos que se realizan sobre los mismos.

2.1 GNU/Linux

GNU/Linux es el nombre que se le da a la combinación del sistema operativo GNU y el núcleo o *kernel* Linux, actualmente es uno de los proyectos en desarrollo de *software* libre más importantes a nivel mundial. El hecho de ser *software* libre implica que todo su código fuente puede ser usado, modificado y distribuido según indica la GPL (Licencia Pública General de GNU), por lo que su evolución se ve determinada por las colaboraciones del conjunto de desarrolladores adheridos al proyecto, una especie de mente colectiva global.

Tanto su condición de *software* libre, como su flexibilidad y fiabilidad, han generado una alta concentración de empresas centradas en el desarrollo de distribuciones que cumplan distintas condiciones para adaptarlas a un público concreto. Existen abundantes ejemplos de distribuciones centradas en distintos aspectos como usabilidad, *hacking*, análisis de red, privacidad, seguridad, etc... Este proyecto se centra en la seguridad para el usuario, no confundir con privacidad, la cual suele acompañar a la seguridad pero nunca la sustituye. La privacidad se centra en proteger los datos e información personal del usuario, en cambio la seguridad busca evitar posibles ataques al equipo.

Un sistema operativo basado en GNU/Linux puede verse abstraido en dos capas lógicas; en la capa inferior se encuentra el núcleo o kernel de Linux, la parte más importante del sistema operativo, mientras que en la superior está el gran conjunto de programas y librerías que hacen posible la utilización de la máquina.

La principal función del kernel es la de gestionar la comunicación con el *hardware* del equipo y por ello es una parte "protegida" del sistema operativo, los programas no tienen acceso a este a no ser que empleen un mecanismo de peticiones gestionado por el propio kernel denominado "llamadas al sistema".

Por otro lado, los sistemas Linux cuentan con dos modos de ejecución totalmente distintos, dotados de procedimientos diferentes y aislados en memoria RAM. El núcleo del sistema se ejecuta siempre en modo kernel, mientras que los programas y librerías emplean el modo usuario. Cada modo de ejecución dispone de una sección de memoria, inhabilitando el acceso a esta por parte del modo contrario.

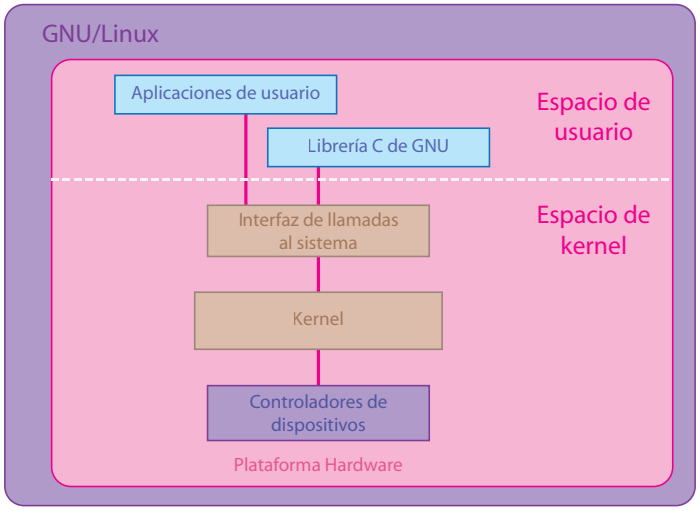

Figura 2.1: Estructura Linux

Este aislamiento entre las dos partes del sistema operativo lleva a diferenciar entre un espacio de usuario en el que se ejecutan los programas y librerías, donde el usuario puede gestionar el *software*, y un espacio de kernel, directamente inaccesible por los programas y que funciona aislado en memoria. Esto dota al sistema de estabilidad y seguridad, ya que evita que cualquier programa pueda causar un error que atente contra la estabilidad del sistema, ya sea causado por algún tipo de acción maliciosa, o por un error de programación en el *software*.

2.2 Seguridad en el Sistema Operativo

El sistema operativo es la pieza de *software* con la que el usuario interactúa para ejecutar programas y manejar archivos en el equipo. Debe ofrecer una administración automática de los recursos físicos e interactuar con el núcleo del sistema para realizar un uso correcto de los accesos al *hardware*; a su vez, debe presentar un equilibrio adecuado entre facilidad de uso y seguridad.

Cuando el usuario de un equipo personal piensa en mejorar la seguridad de la máquina, suele pensar de forma automática en un anti-virus, pero la realidad es que un anti-virus no asegura que la máquina sea inaccesible por *software* malicioso. Además, este tipo de programas llegan a consumir una gran cantidad de recursos requeridos para ejecutarse en segundo plano. Como dijo John McAfee, "la muerte del antivirus como programa necesario para nuestros ordenadores ya se ha producido" [1], muchos no son desinstalables, ofrecen servicios innecesarios, introducen publicidad y en general causan un efecto negativo en el rendimiento y uso del equipo.

Según estadísticas publicadas por diversos desarrolladores de *software* de seguridad, los sistemas operativos más afectados por amenazas son los desarrollados por Microsoft y Apple; aunque es cierto que estos son los sistemas más populares, por tanto más expuestos a ser el objetivo de ataques y contenido malicioso, cuentan con un sistema de seguridad y contención de amenazas que no se encuentra a la altura de las necesidades de los usuarios que realmente quieren proteger su sistema.

Se puede comprobar como los sistemas populares son también los que más vulnerabilidades contienen, además de ofrecer una capacidad de configuración notablemente

rank	operating system	number of vulnerabilities
1	Apple OS X	384
2	Microsoft Windows Server 2012	155
3	Canonical Ubuntu Linux	152
4	Microsoft Windows 8.1	151
5	Microsoft Windows Server 2008	149
6	Microsoft Windows 7	147
7	Microsoft Windows 8	146
8	Microsoft Windows Vista	135
9	openSUSE	121
10	Debian Linux	111
11	The Linux Kernel	77
12	Microsoft Windows 10	53
13	Fedora Linux	38
14	Microsoft Windows 2003	36
15	Xen OS	34

Figura 2.2: Vulnerabilidades por sistema operativo [9]

baja en cuanto a seguridad. También se observa una baja cantidad de vulnerabilidades en los sistemas empleados por Qubes OS (Fedora y Xen), lo que reafirma a Qubes como una mejor opción para preparar un sistema dotado de una seguridad eficiente.

Descartando la posibilidad de un sistema operativo común como los ofrecidos por Microsoft, donde, además de carecer de la seguridad de un sistema Linux, existe una cantidad mucho mayor de *software* malicioso, se presenta como mejor opción el uso de una distribución basada en Linux y Xen, con un sistema realmente efectivo contra acciones perjudiciales para el equipo o usuario.

Existen ciertas ventajas que ofrece un sistema convencional Linux con respecto a otros sistemas operativos y que hacen de este un entorno más protegido. Los sistemas basados en Unix, como lo es Linux, aíslan la parte del sistema operativo accesible por los usuarios estándar de la parte únicamente disponible para el administrador del sistema de dos maneras. Por un lado, como se ha mencionado en la sección anterior, los programas de usuario se manejan en un modo de ejecución que prohibe el acceso a la sección de memoria del núcleo; por otro lado, el sistema de ficheros del sistema se encuentra protegido por un mecanismo de gestión de permisos que identifica al administrador del sistema, siendo el único capaz de alterar la porción del sistema de ficheros perteneciente al propio sistema operativo. Ya que por defecto todos los programas se ejecutan en modo usuario, estos no suponen una amenaza a la estabilidad del sistema a no ser que de alguna manera obtengan la contraseña de administrador del sistema, para así obtener acceso a la parte del sistema de ficheros protegida mediante permisos de administrador.

Posee una configuración por defecto más restrictiva y aislada de posibles amenazas, aunque notablemente más adaptable que la mayoría del resto de sistemas, pudiendo alcanzar el nivel de seguridad realmente deseado por el usuario. Por otro lado, no alberga el concepto de un registro centralizado, sino que se basa en un sistema de archivos de configuración descentralizado y más complicado de alterar. Otra de las mayores ventajas es el ser un sistema modular, esto significa que se encuentra segmentado en diversas partes (o "programas" para simplificar) las cuales juntas conforman el sistema; por este hecho es posible desactivar una parte del sistema en caso de que hubiese sido infectada e incluso eliminarla y sustituirla por otra limpia. Por último, cabe mencionar que Linux es un sistema constantemente actualizado con parches de seguridad, manteniéndose a la altura del cada vez más presente desarrollo de *software* malicioso.

En cuanto a distribuciones de Linux, se presentan varias opciones que centran sus esfuerzos en crear un entorno seguro y privado:

Subgraph OS: diseñado especialmente para evitar ataques de red y *malware*. Emplea el método de *sandboxing* para aislar las aplicaciones clave del sistema y cuenta con un

núcleo alterado con parches de seguridad que lo hacen más resistente a ataques. La meta de este sistema operativo es proporcionar un sistema seguro con el que reducir los riesgos al establecer comunicaciones entre organizaciones de todo el mundo, evitando posibles escuchas y robos de datos.

Whonix: se centra directamente en garantizar la privacidad y anonimato del usuario, basado en el desarrollo de la red Tor también ofrece mejoras con respecto a seguridad aunque no tan abundantes y claras como otras distribuciones ofrecen. Esta distribución está más orientada a usuarios que prefieran priorizar su anonimato, evitando la vigilancia gubernamental o el rastreo de los movimientos por la red. Si bien otorga cierto grado de seguridad ante amenazas en red, no se considera como una buena opción para la configuración de un sistema de alta seguridad.

TENS: desarrollado por las Fuerzas Aéreas estadounidenses, esta distribución está pensada para ejecutarse en modo *live*, de manera que cualquier posible amenaza es eliminada en el apagado. Aunque es uno de los sistemas más seguros, ya que siempre se inicia como un sistema "nuevo", no constituye un buen sistema a largo plazo, ya que prácticamente cualquier usuario necesita modificar o instalar *software* en la máquina y sería realmente incómodo tener que realizar esta acción después de cada inicio.

Qubes OS: emplea un sistema de aislamiento de actividades mediante el hipervisor Xen para garantizar la seguridad del equipo. Centra la atención en la seguridad, a pesar de tener ciertas remarcables ventajas con respecto a privacidad. Qubes no establece un sistema infalible contra cualquier tipo de amenaza, sino que asume la posibilidad de que un ataque se realice satisfactoriamente contra el equipo. La metodología de este sistema consiste en aislar la amenaza en una pequeña porción del equipo, afectando únicamente a esta y no al sistema entero.

Aunque existe infinidad de posibilidades, cada distribución destaca en un ámbito, por lo que la elección depende de la finalidad o uso que se vaya a dar al equipo. Para configurar un entorno relativamente básico con el que asegurar las comunicaciones en una empresa, la mejor opción sería Subgraph OS. En cambio, para conseguir un sistema que nos permita pasar inadvertidos en la red, típicamente *hackers* y especialistas en seguridad, Whonix es la distribución que más destaca. Si en cambio lo que se pretende es no dejar rastro en la máquina, TENS es una distribución preparada para ejecutarse sin prácticamente dejar evidencia.

Qubes OS por otro lado enfoca sus esfuerzos en el desarrollo de un sistema innovador de protección, que proporciona una seguridad mayor que las distribuciones anteriores manteniendo todas o casi todas las funcionalidades y características de estas. Es capaz de ejecutar de manera virtualizada casi cualquier otra distribución Linux, por lo que es posible aprovecharse de las ventajas que una distribución pueda ofrecer, de hecho Qubes cuenta con una máquina virtual Whonix lista para ejecutarse después del primer arranque, así como de un mecanismo para ejecutar virtualizaciones "temporales", que al igual que TENS, no almacenan dato alguno ni dejan rastro de uso.

2.3 Posibles Ataques

Un equipo concreto, esté conectado o no a la red, es vulnerable a multitud de ataques que pueden robar datos, afectar al rendimiento, instalar publicidad e incluso tomar el control total del mismo. Es razonable clasificar los ataques en dos grupos generales, según se trate de un ataque que se presenta sobre la información y el *software*, o en cambio se trata de un ataque en el que se manipula el equipo físicamente. Cabe destacar que Qubes no protege directamente contra los ataques físicos (a excepción de Evil Maid en

caso de configurar el *software* adecuado), no obstante es necesario tener conocimiento de que existen y de que cabe la posibilidad de que un equipo se vea afectado por alguno de este tipo de ataques; para evitarlos, las soluciones reales pasan por la ocultación y la protección física del hardware y puertos del sistema.

Al margen de los ataques realizados a un equipo concreto, existe un tipo de programas llamados *malware*, que se manifiestan en varias formas de *software* malicioso, y en prácticamente todos los casos es necesario que el usuario inicie la acción de instalarlo. No obstante, el propio instalador puede ir camuflado en otro tipo de archivo, como un documento Word o un PDF, de manera que este se autoinstale al abrir dicho archivo. Una vez instalado en el sistema, el *malware* puede emprender diversas acciones, desde tomar el control total de la máquina, hasta monitorizar las pulsaciones de teclado o enviar datos confidenciales al atacante. Aunque no es un ataque como tal, este tipo de programas se propagan por la red infectando equipos desprotegidos, pudiendo fácilmente encontrarse en un correo electrónico, una descarga de una web supuestamente de confianza o adherido a instalaciones de otros programas. Es por ello que conforman una amenaza real para cualquier equipo y deben considerarse al establecer un entorno seguro.

2.3.1. Ataques Lógicos

Phising

La mayoría de usuarios evitan abrir archivos adjuntos desconocidos o sospechosos, donde posiblemente exista algún tipo de *malware*, por ello los hackers suelen recurrir a técnicas de *phising*. Este tipo de ingeniería social se basa en hacerse pasar por una persona o empresa de confianza en algún tipo de comunicación electrónica para recopilar información de la víctima, como contraseñas, tarjetas de crédito y otros datos personales. Para evitar este tipo de ataques, es necesario comprender la importancia de la verificación de interlocutores en todo tipo de comunicación electrónica como correos electrónicos, chats, páginas web, etc.

Ataques por inyección SQL

SQL son las siglas para describir un lenguaje de programación usado en la comunicación con bases de datos. Muchos de los servidores que contienen datos críticos emplean SQL para su gestión, esto abre la posibilidad de un ataque por inyección. Típicamente, este tipo de ataque aprovecha una vulnerabilidad en el *software* que emplea la base de datos para introducir una sentencia SQL que pueda extraer o modificar información de la misma. Por ejemplo, un cuadro de búsqueda de una web no protegida ante este tipo de ataques podría ser usado para introducir código SQL.

***Cross-Site Scripting* (XSS)**

Otra manera de efectuar un ataque a través de una página web es mediante la inclusión de un pequeño *script* que se vea automáticamente ejecutado por el navegador del usuario al cargar la página. Este ataque no se centra en la web ni sus datos, sino en la posible información que un determinado usuario pueda entregar. De esta manera, es posible introducir dicho código en algún punto de la página, como en un comentario de un blog, para que al abrir esta se muestre algún tipo de mensaje requiriendo credenciales, información bancaria, etc...

Denegación de servicio (DoS)

Este tipo de ataque consiste en saturar el tráfico hacia un determinado servicio en red, de manera que este quede paralizado. Esto se consigue lanzando peticiones constantes contra el servicio, hasta que este alcance un punto en el que no pueda atender más y por tanto se vuelva inaccesible. En algunos casos, el ataque se puede lanzar desde distintas

máquinas alrededor del planeta (ataque DDoS, o denegación de servicio distribuida), haciendo a este mucho más difícil de contrarrestar.

Ataques *man-in-the-middle*

Mientras un equipo se encuentra navegando por internet, este efectúa una gran cantidad de transacciones con distintos servidores de todo el mundo, identificándose y solicitando distintos servicios o páginas. Una conexión entre una máquina y un servidor debería ser privada e inaccesible por terceras partes; no obstante, es posible que un atacante capture la identificación de la máquina o servidor, permitiendo hacerse pasar por el emisor o receptor y así obteniendo información de ambos extremos, esto es lo que se denomina un ataque *man-in-the-middle*.

Reutilización de credenciales

Los usuarios de hoy en día deben recordar un número tan elevado de nombres de usuario y contraseñas que la mayoría de las veces simplemente repiten la misma contraseña para distintas cuentas, un hecho que los *hackers* aprovechan. La realidad es que en el mercado negro es posible conseguir amplias colecciones de nombres de usuario y contraseñas de páginas que hayan sido atacadas, por lo que aunque pertenezcan a un simple foro, es posible que el usuario emplee esa misma contraseña para acceder a su cuenta bancaria, dando pie a que el *hacker* tenga pleno acceso a esta.

2.3.2. Ataques Físicos

Ataques vía USB

Gran parte de los mayores ataques informáticos de la historia se han realizado de manera física sobre el equipo al que atacar. Una de las manera más típicas de acceder es mediante la inserción de un dispositivo de almacenamiento USB infectado que introduzca el *software* malicioso en la máquina. Además, actualmente es posible introducir el código malicioso en el propio *firmware* del dispositivo, complicando notablemente su detección.

Ataques de acceso directo a memoria (DMA)

Este ataque se aprovecha de la capacidad de acceder directamente a la memoria del equipo a través de ciertos puertos como el Firewire, CardBus, PCI o PCI Express. Mediante estos es posible evitar la mayoría de los controles de seguridad del sistema operativo, y acceder directamente al espacio de direcciones físicas para obtener claves de encriptado, instalar *malware* o controlar otros dispositivos conectados.

Evil Maid

En esencia, se trata de un ataque en el que se tiene acceso al equipo en varias ocasiones sin el conocimiento del propietario. En un primer contacto, y estando el equipo apagado y no vigilado, el atacante inserta un dispositivo de almacenamiento USB conteniendo un sistema de arranque alternativo, se inicia y se instala un capturador de pulsaciones de teclado; posteriormente y sin tener conocimiento de ello, el propietario inicia el ordenador con normalidad e introduce las claves de encriptación; finalmente el atacante vuelve a acceder al equipo para obtener la clave, ganando acceso completo al equipo.

Ataque de arranque en frío (*Coldboot*)

Para efectuar un arranque en frío, es necesario apagar repentinamente la máquina sin que a esta le dé tiempo a hacerlo limpiamente. Después, para efectuar el ataque, existen dos posibilidades; se carga un sistema operativo ligero desde la unidad de disco extraíble para volcar el contenido de memoria en un archivo, o se extraen físicamente las memorias para posteriormente introducirlas en otro equipo y realizar el volcado de datos. En este

ataque se suelen emplear métodos de refrigeración sobre las memorias para ampliar el tiempo de vida de los datos en su interior.

CAPÍTULO 3
Seguridad mediante aislamiento

El aislamiento es una metodología que separa distintos aspectos del equipo, sobre este concepto se fundamenta la base de los sistemas operativos actuales. Dada la elección de un sistema Linux, en este capítulo se procede a explicar dos implementaciones basadas en aislamiento que generan seguridad en este sistema, la gestión de procesos y la virtualización de un sistema operativo. Posteriormente se expone una introducción a los hipervisores, especificando las diferencias entre los dos tipos existentes. Por último se aborda concretamente el hipervisor Xen, utilizado por Qubes.

3.1 Gestión de procesos

En un sistema operativo, a los programas que se encuentran en ejecución se les denomina procesos y están aislados entre si. Esta es una aplicación de la técnica del aislamiento que otorga seguridad y estabilidad al sistema, evitando que los procesos interfieran el uno con el otro. A estos procesos se les asigna un identificador único y son gestionados por un servicio del núcleo que mantiene información acerca de cada proceso, como su estado y espacio de memoria accesible. A su vez, este servicio mantiene una cola de ejecución en la que cada proceso espera su turno para ocupar el procesador.

Por otro lado, los procesos están separados entre los que se ejecutan en modo privilegiado y los que no, limitando las acciones que estos últimos pueden tomar sobre el sistema. Los procesos generados por un usuario (voluntariamente o no) son comúnmente ejecutados en modo no privilegiado, reduciendo el rango de acción de un proceso malicioso. Existe la posibilidad de otorgar privilegios a archivos, directorios e incluso procesos para que puedan emplear funciones solo accesibles desde este modo, generalmente se emplean para otorgar privilegios temporalmente a procesos que necesitan realizar una tarea específica. Para garantizar que un proceso no puede tomar el control de otro proceso, ni del propio sistema, se le asigna un espacio de memoria individual e independiente, y pasa a formar parte de la cola del planificador de procesos para acceder al procesador cuando esté disponible.

Aunque el funcionamiento de un sistema está basado en el aislamiento, un proceso debe ser capaz de generar otro proceso, terminarlo cuando se requiera y comunicarse con otros procesos; así pues, surge la necesidad de establecer una jerarquía de procesos para facilitar la gestión de los mismos y delimitar el alcance de acción de cada uno.

En Linux, un solo proceso es el encargado del arranque de la máquina; una vez este proceso, llamado *init*, ha acabado de ejecutar todas las rutinas necesarias para el arranque, es su labor generar otro proceso denominado *getty* cuya misión es encargarse del acceso mediante usuario y contraseña. Sin embargo, posteriormente se necesitan muchos

más procesos para manejar las distintas acciones simultáneas requeridas por el sistema y el usuario, por ello es necesario alguna técnica de agrupación algo más compleja que una típica relación de padre-hijo. Estos sistemas se basan en la agrupación de procesos, que a su vez están agrupados en sesiones. Estos grupos son utilizados principalmente para la distribución de señales, una señal que se vea dirigida a un grupo es recibida por todos los procesos dentro de este. A su vez, un proceso no puede migrar de una sesión a otra, pero sí entre grupos de la misma sesión; tampoco puede crear otros procesos o grupos de procesos fuera de la sesión a la que pertenecen.

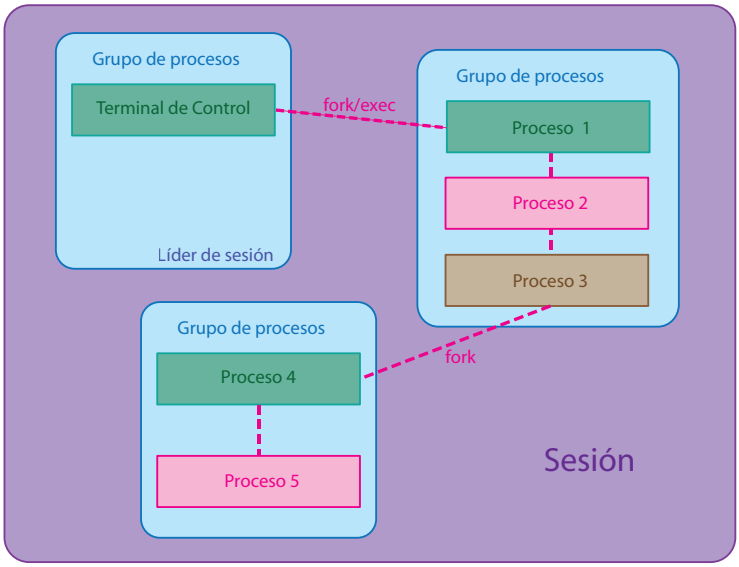

Figura 3.1: Grupos de procesos y sesiones en sistemas Linux

3.2 Virtualización de sistemas operativos

En la anterior sección se ha visto como los sistemas operativos emplean el aislamiento para separar y gestionar los procesos, pero es de notar que esta condición no garantiza seguridad, únicamente otorga un sistema básico para la prevención de ciertas metodologías que resultarían perjudiciales para el equipo. Para llevar la técnica del aislamiento un paso más lejos, es posible abstraer completamente un sistema operativo, de manera que este actúe en todo su conjunto como un programa más, gestionado por otro programa denominado hipervisor; esta técnica es la llamada virtualización.

El concepto de virtualización se emplea desde el nacimiento de los computadores, aplicado directamente sobre distintos aspectos de la computación; sin embargo, la virtualización de un sistema completo ha sido, en los últimos años, algo asociado a grandes sistemas, nunca como una solución posible a nivel de usuario. Gracias al continuo avance de la tecnología, y más concretamente al desarrollo de los microprocesadores, hoy es posible realizar virtualizaciones de sistemas operativos completos en prácticamente cualquier dispositivo moderno.

La virtualización, además de abaratar costes en diversos ámbitos como la gestión de redes, aporta un nivel más de seguridad mediante el aislamiento; los sistemas virtualizados son realmente igual de seguros (o de inseguros) que un sistema nativo, pero un posible ataque satisfactorio contra él quedaría retenido en la máquina virtual, incapaz de alcanzar el resto del sistema. La única posibilidad de que un *software* malicioso afecte

al resto del sistema es que este se haga con el control del hipervisor, lo que actualmente resulta extremadamente complicado.

Como ventajas destacables de la virtualización en cuanto a seguridad, cabe destacar:

- Los sistemas virtualizados con almacenamiento centralizado evitan la pérdida de datos en caso de que uno de los dispositivos se pierda.

- El aislamiento entre máquinas virtuales previene la propagación de ataques entre los sistemas.

- La migración de un sistema virtualizado a otra máquina física es más sencillo y rápido que el de un sistema nativo.

- Si una máquina virtual queda infectada, es posible devolverla a un estado previo a la infección.

- La limitación de acceso al *hardware* y la reducción de la variedad de dispositivos accesibles desde un sistema virtualizado complican un ataque.

- Los hipervisores son piezas de *software* realmente pequeñas, la poca cantidad de código reduce considerablemente la "superficie" atacable.

Ya que el aislamiento no ofrece, ni mucho menos, una seguridad infalible por si mismo, se ha de emplear de una manera óptima para reducir el riesgo de ataque. Ciertas prácticas hacen de la virtualización un buen método de protección.

- Una máquina virtual no debe tener conexión de red si no es estrictamente necesaria.

- Nunca se debe emplear la misma máquina para efectuar una acción potencialmente dañina que para otra acción que requiera de seguridad y certeza de no estar siendo monitorizado.

- Mantener las máquinas virtuales apagadas de no estar siendo usadas.

- Guardar las copias de seguridad de las virtualizaciones en un almacenamiento separado.

- No usar sistemas de compartición de ficheros o texto entre virtualizaciones y el sistema huésped.

3.3 Introducción a los hipervisores

Un hipervisor es una plataforma que, aplicando técnicas de control de virtualización, permite el uso de distintos sistemas operativos ejecutándose concurrentemente en la misma computadora. Es una extensión del término "supervisor", el cual se aplica a los *kernel* tradicionales de los sistemas operativos. Fueron originalmente creados en los 70, donde se empleaban para ejecutar varios sistemas operativos en un *mainframe* (un equipo más grande y más potente) y así segmentar los distintos sectores de las empresas, grupos de usuarios, etc... Esto además de reducir los costes en infraestructura, garantizaba la estabilidad de los sistemas en caso de que uno en concreto fallase.

Se pueden clasificar en dos tipos según la manera en la que se ejecutan.

Nativo o tipo-1

Este tipo de hipervisores se ejecuta directamente sobre el *hardware* del equipo, siendo lo primero que se carga desde BIOS al iniciar. Una vez cargado, su tarea es arrancar los sistemas operativos invitados y ofrecerles una interfaz para manejar los accesos directos al *hardware*. A pesar de que es la implementación de virtualización más antigua, actualmente sigue siendo la manera más eficaz, potente y segura de emplear la virtualización, por ello los grandes fabricantes emplean este enfoque.

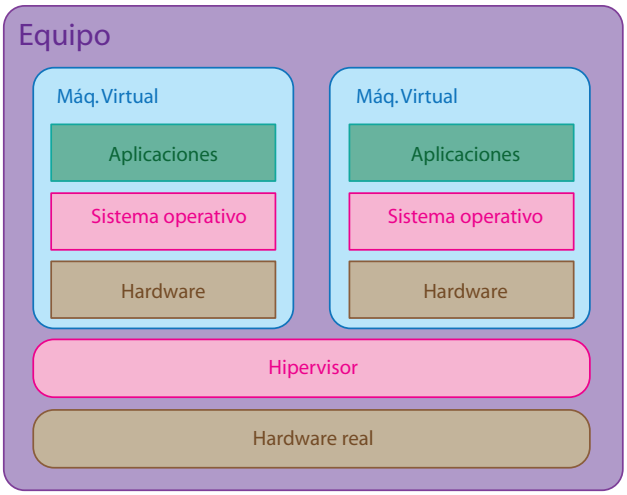

Figura 3.2: Hipervisor nativo

Existen varios hipervisores destacables de este tipo, tanto comerciales como de código libre, que ofrecen diversas características adaptándose a un entorno más centrado en un usuario estándar, o a un entorno empresarial. Entre ellos, se sitúa KVM (Kernel-Based Virtual Machine), un hipervisor que puede adherirse a casi cualquier sistema operativo Linux y que cuenta con versiones comerciales más avanzadas como RHEV (Red Hat Enterprise Virtualization). Por otra parte se encuentra Xen y su versión comercial Citrix XenServer, que aunque parecen ser menos soportados por los fabricantes, ofrecen un rendimiento ligeramente superior a su hipervisor análogo KVM. Destacan dos hipervisores más que aunque no se presentan como código libre, si tienen versiones gratuitas limitadas, estos son Microsoft Windows Server Hyper-V y VMware vSphere, siendo este último considerado como el más potente en su versión comercial más alta.

Hospedado o tipo-2

Frecuentemente denominados como monitores de máquina virtual (VMM), este tipo de hipervisores se ejecutan sobre un sistema operativo anfitrión, de la misma manera que

cualquier aplicación. Las máquinas virtuales se ejecutan en un tercer nivel, por encima del VMM. Estos hipervisores tienen el inconveniente de, en general, ser notablemente menos eficientes que los nativos, además de no ofrecer las ventajas de seguridad inherentes al tipo de hipervisor antagónico.

Un uso generalizado que se le da a estos hipervisores, es el de actuar como plataforma para la ejecución de *software* multiplataforma, como es el caso de .NET o las máquinas virtuales Java.

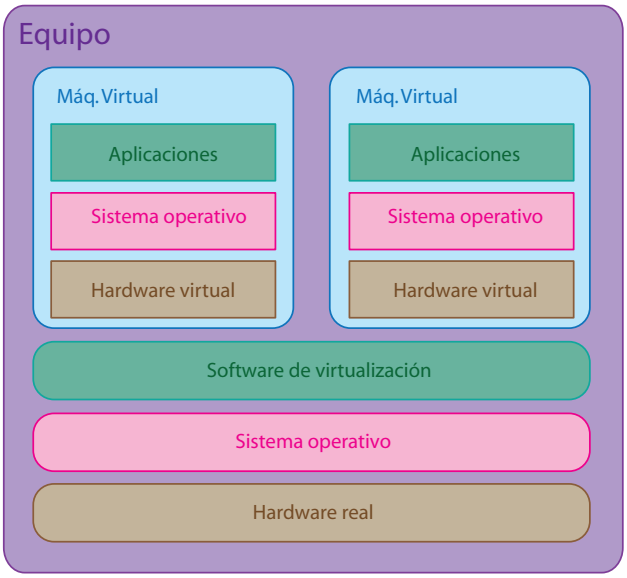

Figura 3.3: Hipervisor hospedado

VMware ofrece varias versiones de su hipervisor hospedado, desde una gratuita capaz de virtualizar un único sistema hasta otras más complejas que permiten diversos sistemas operativos simultáneos o incluso versiones del mismo. Microsoft tiene su propio hipervisor de tipo-2 denominado Virtual PC, capaz únicamente de virtualizar sistemas Windows. Oracle también ofrece un hipervisor de este tipo, VirtualBox; este se presenta como una opción gratuita aunque muy potente, y que comparte características con VMware vSphere y Microsoft Hyper-V. Cabe destacar que KVM posee características tanto de hipervisor de tipo-1 como de tipo-2, pudiendo convertir el propio kernel Linux en un hipervisor sobre el que las máquinas virtuales tengan acceso directo al *hardware*.

3.4 Hipervisor Xen

Xen es un hipervisor nativo de código abierto desarrollado por la Universidad de Cambridge, centrado en poder ejecutar todas las características de distintos sistemas operativos en una sola máquina. Es el hipervisor que Qubes OS emplea para su virtualización ya que los desarrolladores de este sistema argumentan que es el hipervisor más adaptable para una configuración de seguridad, además de poseer una "superficie de ataque" mucho menor que otros hipervisores donde la cantidad de código es mayor.

Existe una amplia colección de ventajas que aporta Xen respecto a otros hipervisores existentes. Caben destacar:

Soporte a distintos sistemas operativos

Actualmente Xen da soporte a los sistemas operativos más representativos, como Linux, Windows, NetBSD y FreeBSD, dando flexibilidad para escoger el apropiado.

Escalabilidad

Es capaz de manejar 4096 procesadores anfitriones con 16 tera bits de memoria. Usando la paravirtualización, el hipervisor soporta un máximo de 512 procesadores virtuales y 512 giga bits de RAM por cada sistema invitado. Usando la virtualización por *hardware*, un máximo de 128 procesadores virtuales y 1 tera bite de RAM por sistema invitado.

Rendimiento

Como indican diversas pruebas de rendimiento [2] y debido en gran medida a su arquitectura de hipervisor nativo, Xen supera notablemente a otros hipervisores en cuanto a rendimiento general y tiempos de respuesta.

Seguridad

Provee de un sistema de seguridad de alto nivel fundamentado en su estructura modular. A su vez cuenta con una serie de módulos de seguridad (XSM) desarrollados por NSA para casos de uso ultra seguros. XSM proporciona un sistema de control de grano fino para las acciones que el hipervisor pueda tomar consigo mismo y con el exterior.

Flexibilidad

Xen permite ajustar la instalación a las necesidades del usuario; desde elegir el tipo de virtualización para adaptarse a *hardware* antiguo o reciente, hasta elegir el sistema Unix o Linux sobre el que trabajar. Además, su arquitectura flexible promueve el soporte de grandes fabricantes para configurar el sistema con el *software* complementario requerido.

Modularidad

Una característica que otorga robustez, escalabilidad y seguridad es la división del dominio de control en diversos pequeños módulos, cada uno con un *kernel* mínimo y un controlador. Este concepto es denominado disgregación de dominio, y es similar a la separación de procesos en un sistema operativo convencional, pueden ser iniciados y parados en cualquier momento sin afectar al resto del sistema, reduciendo notablemente los cuellos de botella y complicando posibles ataques.

Código abierto

Código abierto significa que el proyecto permanece vivo gracias a una comunidad interesada en que este avance, no depende de los intereses económicos ni prioridades de una empresa a cargo. Además, cualquiera pueda colaborar con la elaboración del código e influenciar en la dirección que sigue el proyecto.

Soporte de fabricantes

Los sistemas operativos pueden adoptar una serie de modificaciones específicas para ejecutar Xen manteniendo la compatibilidad con las aplicaciones de usuario, lo que permite que Xen mejore el rendimiento de las virtualizaciones sin necesidad de un *hardware* específico. Gracias a diversas aportaciones de Intel y AMD, Xen es capaz de establecer cualquier sistema operativo sin modificar como huésped en sus máquinas virtuales. Además, Xen se beneficia de un gran número de fabricantes y empresas de servicios afiliados al proyecto e interesados en que este avance, esto otorga a los usuarios una rica selección de *software* y servicios completamente funcionales en el sistema.

En definitiva, Xen proporciona aislamiento para generar seguridad, mantiene un buen control de los recursos y permite el apagado y migración de máquinas virtuales en caliente.

Xen implementa dos tipos de virtualización, una mediante HVM (Hardware Virtual Machine - Máquina de *hardware* virtual) y otra mediante paravirtualización.

La paravirtualización es una manera eficiente y ligera de virtualizar sistemas operativos introducida por Xen. Al contrario que HVM, la paravirtualización no requiere de soporte por parte de la CPU, pero si son necesarias ciertas modificaciones en el *kernel* del sistema que hacen que sea posible su ejecución de forma nativa sobre el hipervisor, esto elimina el requerimiento de emular el *hardware*. Estos núcleos modificados existen para diversos sistemas como Linux, Solaris, NetBSD, FreeBSD, etc... La paravirtualización, aunque no es soportada por Windows, goza de gran soporte por parte de grandes contribuidores de código abierto, siendo completamente soportada por el *kernel* de Linux desde la versión 2.6.24, de modo que cualquier *kernel* a partir de esa versión debe funcionar sin necesidad de ninguna modificación. Es posible virtualizar sistemas con un *kernel* adaptado mediante HVM, estos emplearán automáticamente los controladores de paravirtualización en el acceso a disco y red.

HVM (también llamado *Fully virtualized*) es la otra opción para la emulación de sistemas, esta requiere de soporte por parte de la CPU (VT-x para Intel y AMD-V para AMD), y emplea una versión modificada de Qemu (otro virtualizador licenciado con GPL) para una emulación completa del *hardware*. Normalmente, las máquinas basadas en HVM son ligeramente más lentas que las paravirtualizadas, no obstante para mejorar el rendimiento, tanto Windows como Linux, se pueden beneficiar de unos controladores especiales que permiten el uso de la paravirtualización para el acceso a disco, red e interrupciones, estos controladores son de código abierto y están disponibles bajo el nombre de GPLPV y PV-on-HVM respectivamente; esta configuración es llamada PVHVM y, al contrario que un sistema con *kernel* modificado virtualizado con HVM, goza de acceso a los temporizadores e interrupciones de sistema mediante paravirtualización, esto hace que los controladores PVHVM sean generalmente una mejor opción.

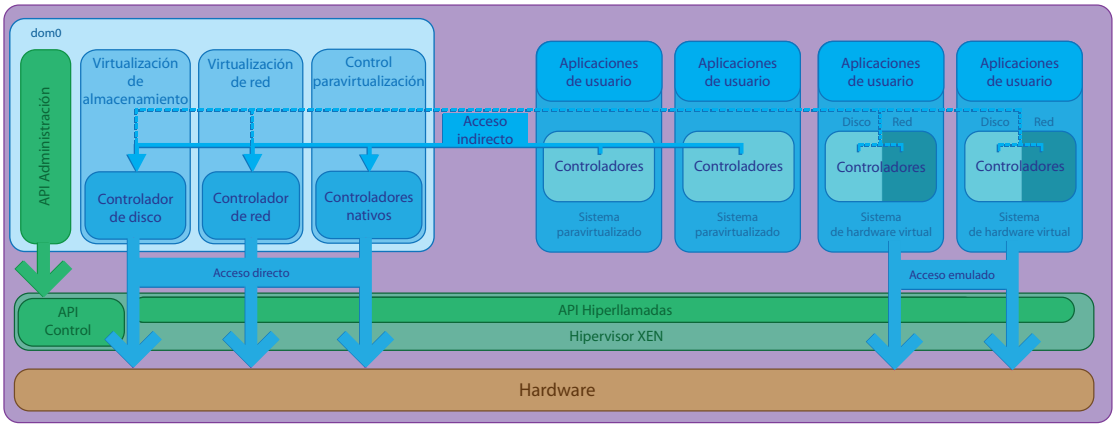

Figura 3.4: Estructura del hipervisor Xen

En la siguiente figura se compara a grandes rasgos el rendimiento ofrecido por los distintos tipos de virtualización disponibles en Xen. En particular, se muestra el tipo de acceso a distintos niveles del sistema de menor a mayor. El acceso más básico, el de disco y red, únicamente se ha de emular en un sistema HVM sin controladores de paravirtualización; las interrupciones y temporizadores de sistema son accesibles por paravirtualización en configuraciones con controladores PVHVM y en sistemas completamente paravirtualizados; a partir de este nivel, los accesos de los sistemas HVM deben ser bien virtualizados por *software* o por *hardware*.

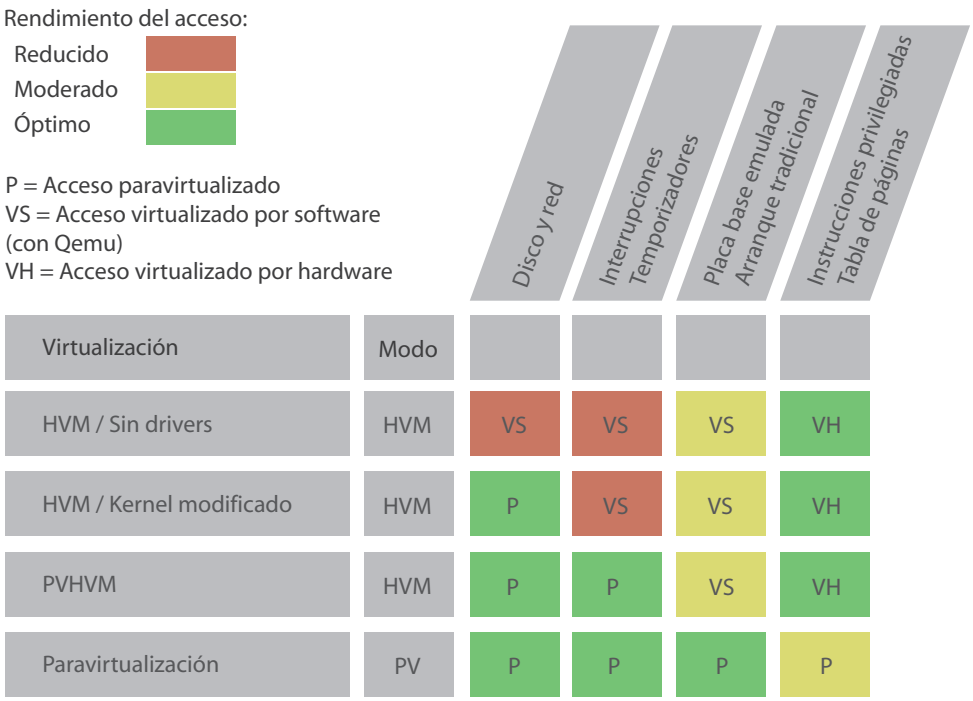

Figura 3.5: Rendimiento según el tipo de virtualización

Como se deduce de la figura, siempre que sea posible, es conveniente emplear la paravirtualización; sin embargo, la diferencia de rendimiento con una máquina HVM no es excesivamente grande, y con la configuración de los controladores PVHVM es posible alcanzar prácticamente el mismo rendimiento, siendo así posible una emulación completamente óptima para sistemas con núcleos no preparados para Xen, como Windows.

CAPÍTULO 4
Qubes OS

Por todas las características y ventajas mencionadas, se escoge Qubes OS como sistema para construir un equipo robusto y seguro. No obstante, no toda la seguridad recae sobre el sistema, gran parte de ella depende directamente del usuario; por ello, es necesario partir de la base de una instalación correcta y no comprometida. Además, un uso correcto del sistema reduce notablemente la posibilidad de un ataque, por lo que en este capítulo se tratarán estos dos temas.

4.1 Protección con Qubes OS

Qubes es un sistema operativo basado en el hipervisor Xen, y emplea el método de la compartimentación para generar seguridad, este separa las diversas partes de la "vida digital" de un usuario en diversos compartimentos aislados llamados *qubes*. Así pues, se vuelve prescindible el necesitar diversas máquinas físicas para las distintas actividades a emprender. Desde un *qube* se puede acceder a páginas web no seguras, desde otro acceder a la cuenta bancaria, y desde uno distinto desarrollar cualquier trabajo; de esta manera si algún *qube* es atacado, la vulnerabilidad permanece en dicho espacio, sin poder propagarse hacia otros *qubes*.

Este tipo de técnica de aislamiento se puede reproducir mediante un sistema operativo ejecutando diversas máquinas virtuales (comparable a los *qubes*), pero existen dos diferencias relevantes que hacen de Qubes una mejor opción en cuanto a usabilidad, rendimiento y seguridad.

En primer lugar, Qubes se encuentra alojado sobre un hipervisor de tipo nativo; como hemos mencionado anteriormente, este agiliza la ejecución de los sistemas operativos virtualizados mejorando el rendimiento general del equipo con respecto a un hipervisor hospedado.

Por otro lado, todos los *qubes* están integrados en un sólo sistema anfitrión con una interfaz gráfica segura y usable de forma muy similar a cualquier distribución de Linux, se le denomina como dom0. Cada *qube* se ejecuta en una ventana integrada en el sistema anfitrión a la que se le asigna un color de borde de ventana distinguible por el usuario, separando los distintos niveles de seguridad. Los principales focos de ataques informáticos, como la tarjeta de red o los puertos USB, están aislados en sus respectivos *qubes*, aunque son completamente accesibles mediante accesos de red seguros, *firewalls* y un manejador específico de dispositivos USB. También cuenta con un sistema de portapapeles que permite copiar y pegar texto o archivos entre distintos *qubes*.

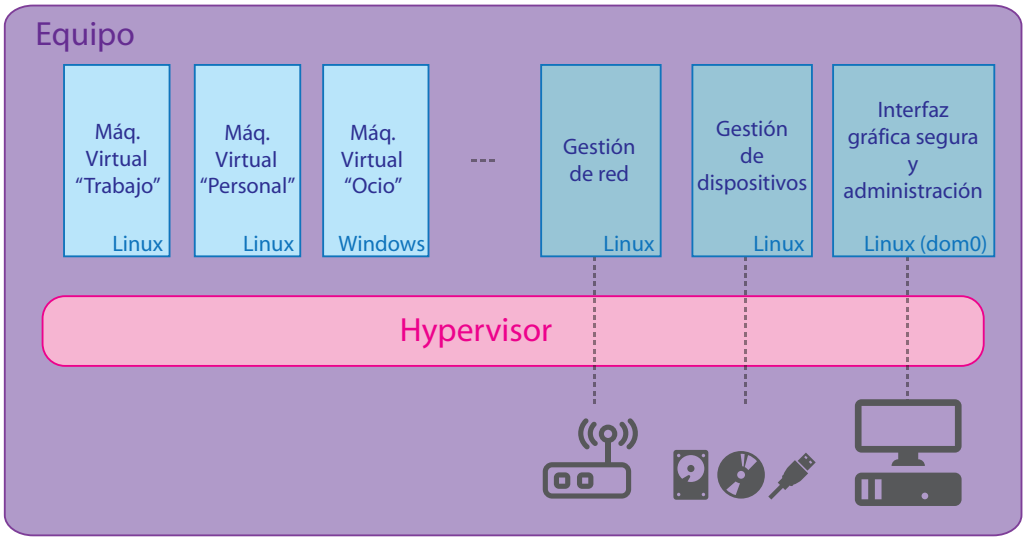

Figura 4.1: Estructura de un sistema Qubes

4.1.1. Conectividad y *firewall* en Qubes OS

Por defecto, todos los *qubes* están conectados a la red mediante la máquina virtual "FirewallVM", que es usada para forzar la aplicación de políticas a nivel de red, pero es posible crear más *qubes* para la gestión de red y configurarlos independientemente.

Para editar las reglas de *firewall* de un *qube*, basta con seleccionarlo en el administrador de *qubes* y presionar el botón de *"Firewall"*; mediante la interfaz gráfica se pueden especificar las reglas necesarias, aunque alternativamente se puede emplear el comando *qvm-firewall* desde dom0. Cabe destacar que la lista de reglas tiene un número máximo de reglas manejables, a partir de un valor cercano a 35, el *qube* no iniciará.

Aunque Qubes no permite parar una máquina virtual encargada de gestión de red, es posible que esta se pare por algún error o alguna acción tomada por el usuario; en estos casos basta con emplear un comando de terminal para restablecer la conexión con otros *qubes*:

```
$ qvm-prefs <vm>-s netvm <netvm>
```

Otra posible acción a tomar es activar la comunicación entre dos *qubes*, algo que de forma predeterminada está prohibido en Qubes. No obstante, es posible que surja la necesidad de configurar este tipo de conexión para el desarrollo de alguna aplicación o para testeo. En primer lugar, es necesario que ambos *qubes* estén conectados a la misma máquina de gestión *firewall*,

4.1.2. Ataque *Evil Maid*

Uno de los ataques a los que Qubes es vulnerable es el *Evil Maid*; como se ha explicado en el primer capítulo, este ataque se basa en un ataque físico a la máquina en el que se falsifica el arranque para que en el siguiente inicio, el usuario introduzca la clave de encriptado y esta quede guardada para que posteriormente el autor del ataque vuelva a recogerla.

Es posible evitar este ataque mediante el uso de un sistema llamado AEM *(Anti Evil Maid)*, pero este requiere, además de la instalación de *software* adicional, el uso de un

dispositivo de almacenamiento USB directamente en el sistema anfitrión dom0, lo que contradice las normas de aislamiento de dispositivos impuestas por Qubes.

Esto conlleva una elección de seguridad crítica para el sistema, se opta o por protegerse contra ataques *Evil Maid*, o proteger el sistema contra posibles dispositivos USB maliciosos. Dados los recientes descubrimientos de puertas traseras y *software* malicioso introducido directamente en el *firmware* de dispositivos USB incluso nuevos, cabe pensar que es más razonable protegerse contra este tipo de vulnerabilidad, no obstante, depende del uso específico del equipo. Un usuario que viaja constantemente con el equipo es mucho más propenso a sufrir un ataque *Evil Maid*, por lo que la elección es totalmente dependiente del uso.

4.1.3. Privacidad

Qubes es un sistema centrado en aportar seguridad, no confundir con privacidad o anonimidad. No obstante, la naturaleza multisistema de Qubes permite beneficiarse de distribuciones de Linux centradas en estos aspectos, como son Whonix o Tails. Un buen uso de la red Tor y una anonimización de la dirección MAC del equipo pueden garantizar un alto nivel de privacidad. Además se puede configurar un *qube* para ejecutar Signal, una aplicación para Android e iOS que permite la comunicación encriptada punto a punto mediante mensajes de texto, mensajes grupales, ficheros adjuntos y mensajes multimedia. Qubes también permite configurar un *qube* con Martus, un *software* gratis y libre desarrollado para recolectar y administrar información delicada sobre violación de leyes y derechos humanos.

Toda anonimización gira alrededor de la red Tor, rebotando las comunicaciones a distintos nodos de una red distribuida sustentada por otros usuarios de todo el mundo. De esta manera, alguien espiando las conexiones no sería capaz de ver qué sitios web se visitan y, de la misma manera, un sitio web no podría saber la localización física de la máquina desde la que se accede. Whonix hace uso de esta red de una manera especial, se basa en el mismo principio que Qubes, ofreciendo seguridad mediante aislamiento; consta de dos partes, una parte huésped desde la que se accede a la red normalmente, y otra parte llamada estación de trabajo que cifra mediante Tor todas las conexiones entrantes o salientes.

Por otro lado, el uso de Tails no requiere de una instalación propiamente dicha. Esta distribución está pensada para usarse como un sistema operativo “en vivo”, es decir, ningún cambio se guarda en el equipo a no ser que se especifique manualmente y todo rastro de su uso desaparece en el apagado de la máquina. Como Whonix, esta distribución emplea la red Tor para anonimizar las conexiones y cuenta con varias herramientas para la transmisión de mensajes y ficheros cifrados.

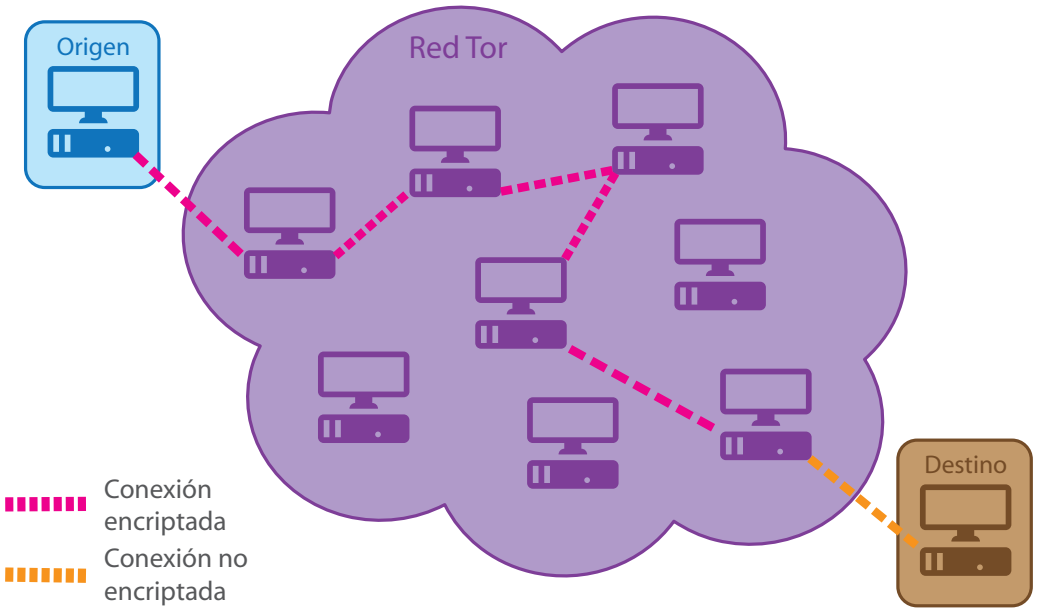

Figura 4.2: Estructura de la red Tor

4.2 Instalación del sistema

4.2.1. Elecciones e implicaciones

Antes de comenzar con la instalación del sistema, hay diversos aspectos a tener en cuenta, algunos afectan directamente a la seguridad del sistema y deben ser considerados previamente.

En primer lugar, los servidores para la descarga de la imagen no están bajo el control de los desarrolladores de Qubes OS; es decir, tras la descarga de la misma, es conveniente verificar la firma digital de la imagen con las claves GPG que ofrece Qubes OS y comprobar que no se encuentra comprometida.

Por otro lado, Qubes OS permite el encriptado completo del disco duro, una opción altamente recomendable. No obstante, si el usuario decide realizar una instalación conjunta con Windows en modo de arranque dual, la partición de disco encargada del arranque no puede ser encriptada, y el sistema podría verse comprometido por un ataque dirigido a esta. Es por esto que se explica como realizar una instalación individual de Qubes OS, ocupando el disco duro por completo.

También es necesario considerar el medio de instalación que se empleará para la instalación. Los medios que se consideran para la instalación son los discos ópticos y los dispositivos de almacenamiento USB. Ambos tienen ventajas e inconvenientes, el USB es notablemente más rápido y cómodo, pero a su vez es más propenso a ser atacado debido a su capacidad para reescribir los datos en él o a contener un *firmware* comprometido. Normalmente, una vez grabados datos en un disco óptico, estos son solo accesibles en modo de lectura por lo que resulta más difícil comprometerlos, en cambio tienen el inconveniente de no poder grabarse desde Qubes OS (impidiendo generar un medio para la actualización en el propio sistema), a no ser que se efectúe con una grabadora óptica externa conectada por USB o SATA. Este último inconveniente es suficientemente relevante como para que en este proyecto se escoja como medio de instalación un dispositivo USB, será necesario pues escoger un dispositivo sobre el que tengamos cierta certeza de no estar vulnerado.

4.2.2. Guía para prueba del sistema en modo “en vivo”

Este modo está actualmente en estado de *alpha*, lo cual quiere decir que sigue en desarrollo y puede presentar fallos y comportamientos erráticos. Además, Qubes presenta, debido a su naturaleza basada en Xen, una serie de problemas a la hora de implementar un sistema “en vivo” que el resto de distribuciones Linux no poseen. Es por ello que este modo tiene ciertas limitaciones a tener en cuenta, nótese que este modo no está ideado para actuar como un sistema de uso cotidiano, sino más bien actuar como una prueba para saber si funcionará correctamente sobre el equipo.

- Por ahora, sólo es posible emplear cinco *qubes* (red, firewall, no-fiable, personal, trabajo).

- Para poder guardar datos es necesario configurar manualmente una partición adicional, que haga de directorio personal persistente.

- No hay opción para instalar el sistema en este modo en un disco.

- El tamaño de disco disponible se ve limitado por la cantidad de RAM del equipo, esto acarrea serias limitaciones, como no poder restaurar varios *qubes* desde una copia de seguridad pesada.

- El sistema puede entrar fácilmente en fallo por falta de memoria disponible en dom0, debido a diversos *qubes* escribiendo simultáneamente en el espacio de ficheros del sistema.

- Actualmente es imposible iniciar el sistema "en vivo" mediante un arranque en modo UEFI.

4.2.3. Guía de instalación

Una vez se tiene una imagen verificada de Qubes OS (en este caso la versión r3.2), es necesario recurrir a otro equipo para preparar el medio de instalación, en nuestro caso, un dispositivo USB de almacenamiento. Cabe destacar que este equipo podría estar comprometido y por tanto poner en riesgo la instalación, por tanto es de vital importancia escoger un equipo que se asuma esté invulnerado.

Para la preparación del medio de instalación mediante alguna distribución de Linux, basta con emplear un comando en terminal:

```
$ dd if=imagen.iso of=/dev/sdX
```

Donde "imagen.iso" es la imagen descargada de Qubes OS y "/dev/sdX" el punto de montaje de la partición del dispositivo USB (p.e. /dev/sda1).

Para la preparación en Windows, es necesario el uso de una herramienta llamada Rufus. Al abrirla se presentan diversas opciones para la creación del medio, las cuales deberemos configurar como en la siguiente figura. El modo de imagen DD se selecciona después de seleccionar la imagen; es posible que se vuelva a preguntar por el modo de imagen al iniciar la copia, en este caso se vuelve a seleccionar el modo de imagen DD.

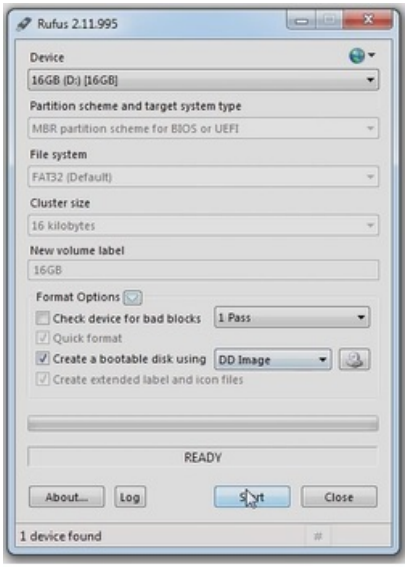

Figura 4.3: Configuración de Rufus

Una vez preparado el medio de instalación, se arranca la máquina donde se desea instalar Qubes OS presionando la tecla que activa el modo de selección de dispositivo de arranque (generalmente F8) y se selecciona el dispositivo USB, ya sea en modo UEFI para las placas compatibles, o en modo normal para las placas más antiguas. Esto lleva a otro menú de selección con varias opciones para la instalación y recuperación de un sistema Qubes.

El asistente de instalación de Qubes OS es notablemente directo y sencillo de seguir, suponiendo que seleccionamos el particionado automático, el cual ocupa un disco completo para la instalación del sistema. Para una mayor seguridad, se activa la opción de encriptar datos en el disco duro completo, por lo que será necesario establecer una contraseña para la encriptación.

En caso de necesitar una configuración personalizada de particionado pero relativamente similar al particionado original de Qubes, esta es la configuración por defecto de Qubes OS.

```
NOMBRE                      TIPO    PUNTO DE MONTAJE
sda                         disco
|---sda1                    part    /boot
|---sda2                    part
   |---luks-<UUID>          encrip
      |---qubes_dom0-root    lvm     /
      |---qubes_dom0-swap    lvm     [SWAP]
```

Una vez completada la instalación, el equipo se reinicia para ejecutar el asistente de configuración inicial de Qubes OS. En este se presentan siete opciones, aunque ninguna es obligatoria para la ejecución del sistema, estas proporcionan *qubes* preconfigurados realmente útiles para usuarios estándar.

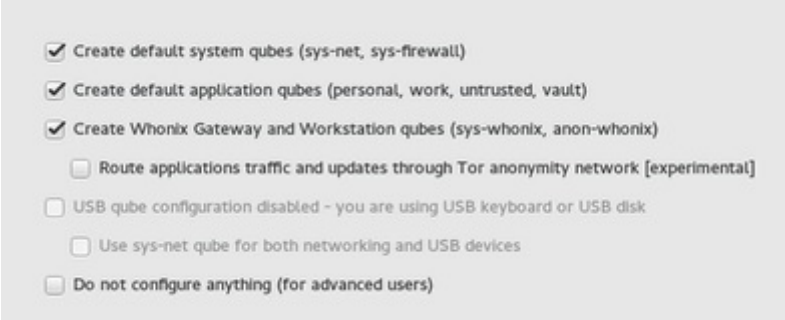

Figura 4.4: Configuración de inicio

La primera opción es casi obligatoria, esta genera un *qube* para ejercer de controlador de red y otro para hacer de *firewall*. Estas dos máquinas virtuales son configurables manualmente, pero su complejidad aleja esta opción de la mayoría de usuarios. La segunda opción genera los *qubes* predeterminados (personal, trabajo, no-fiable, baúl), también configurables manualmente. La tercera opción ofrece dos *qubes* empleados básicamente para navegar anónimamente con Tor en un sistema Whonix. La cuarta opción solo está disponible si no hay ningún dispositivo USB conectado, esta genera un *qube* para el manejo de los dispositivos USB (o asigna este trabajo a la máquina controladora de red, si se activa la subopción), no es una característica recomendada en caso de emplear un ratón, teclado o dispositivo de almacenamiento USB.

4.3 Manejo del sistema

Como se ha mencionado anteriormente, la seguridad del sistema se encuentra directamente relacionada con el empleo correcto del mismo por parte del usuario, por eso es necesario conocer las opciones que ofrece Qubes y saber elegir la adecuada para cada acción.

4.3.1. Tipos de *qubes*

Para comenzar, cabe destacar que en Qubes existen distintos tipos de máquinas virtuales o *qubes* disponibles, cada una adaptable al uso que se le vaya a dar. Estas son desplegables a partir de plantillas que ofrece Qubes y su comunidad, o directamente desde una imagen de instalación convencional. Ofrecen su sistema de archivos de sistema operativo a las máquinas que se hayan desplegado sobre ella, y aunque estas plantillas son sistemas operativos funcionales, están pensados para instalar, mantener y actualizar *software*, nunca para ejecutarlo; esta tarea se reserva para las máquinas que se ejecuten sobre ella.

TemplateVM

Estos *qubes* son plantillas preparadas para desplegar sistemas paravirtualizados. Qubes contiene ciertos sistemas totalmente funcionales, existentes de manera predeterminada, preparados para ejercer de plantilla, no obstante, la amplia comunidad ofrece distintas alternativas que podrían instalarse para formar parte de Qubes.

TemplateHVM

Al igual que los anteriores, estas máquinas sirven para desplegar *qubes*, no obstante estos no están basados en la paravirtualización, sino en la virtualización mediante hardware virtual. Qubes no contiene ninguna plantilla de este tipo por defecto, pero es posible generarlas para así poder desplegar otro tipo de *qubes* con ellas.

DVMtemplate

Este tipo de plantilla es utilizada para los sistemas desechables, actúa de la misma manera que las anteriores.

TemplateBasedVM

Representa al conjunto de máquinas virtuales desplegadas a partir de las plantillas de paravirtualización, están pensadas para ejecutar el *software* que previamente se ha preparado en la plantilla.

TemplateBasedHVM

De la misma manera que las anteriores, representa al conjunto de máquinas desplegadas mediante HVM.

DispVM (DVM)

Estos *qubes* tienen un carácter de ün único uso", son sistemas efímeros o desechables construidos únicamente para efectuar alguna acción concreta que pueda considerarse peligrosa; se eliminan en el apagado, borrando cualquier rastro de su uso en el sistema.

StandaloneVM

Una máquina virtual es denominada “StandaloneVM” cuando es un sistema independiente y no ha sido desplegado a través de una plantilla; así pues, su sistema de archivos no depende de ningún otro *qube*.

AppVM

Este tipo de *qube* se genera a través de cualquier plantilla, conforma un conjunto de aplicaciones basadas en las disponibles bajo la plantilla. Está pensado para ejecutar *software*, pero nunca para instalarlo o actualizarlo, ya que su sistema de archivos depende de la plantilla y estos cambios se eliminarían en el apagado.

ServiceVM

Se le denomina de esta forma a las máquinas destinadas a alguna función concreta, como ejercer de *proxy*, de máquina de red o de corta-fuegos.

ProxyVM

Una máquina virtual puede ejercer de puente para las conexiones de red, ofreciendo a otras máquinas la posibilidad de obtener conexión a través de ella. Una *firewallVM* es una máquina de este tipo en la que se filtra el flujo de datos.

NetVM

Pensada para interactuar directamente con la tarjeta de red y el adaptador inalámbrico, ofrece conexión al resto de *qubes*, o en caso de la configuración por defecto, únicamente al corta-fuegos.

FirewallVM

Un tipo de *proxyVM* configurable para ejercer de corta-fuegos, filtrando cierto tipo de conexiones entrantes o salientes.

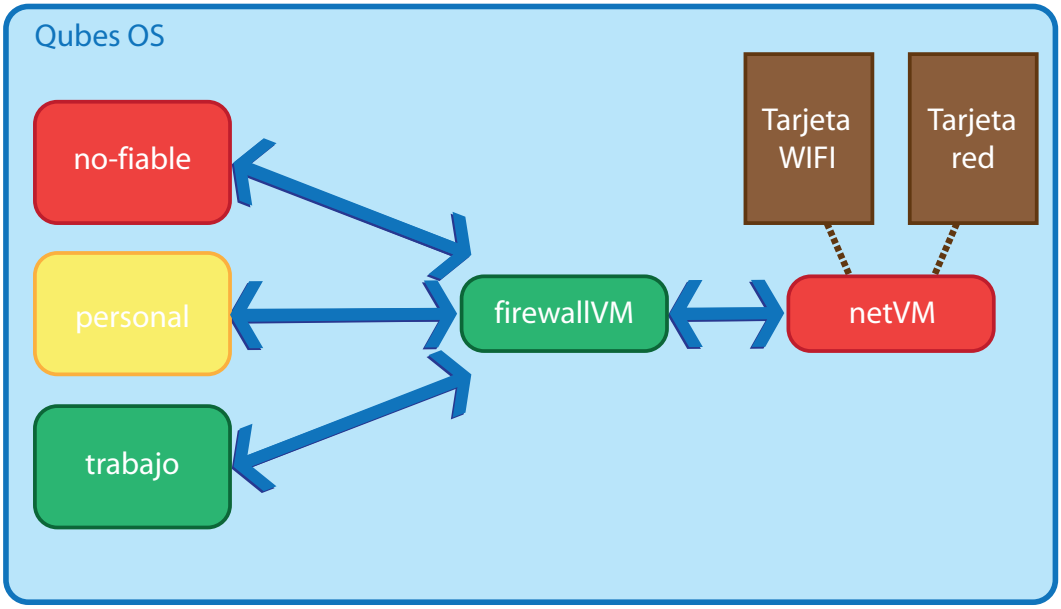

Figura 4.5: Configuración de red por defecto en Qubes

4.3.2. Qubes Manager

Todos los tipos de *qubes* mencionados se controlan mediante una pequeña interfaz llamada *Qubes Manager* que permite la creación, borrado y configuración de los mismos. En este aparecen todas las máquinas instaladas, así como información sobre las mismas; se puede observar si la máquina esta en marcha, la cantidad de RAM disponible y ocupada en cada máquina, uso de procesador, actualizaciones disponibles...

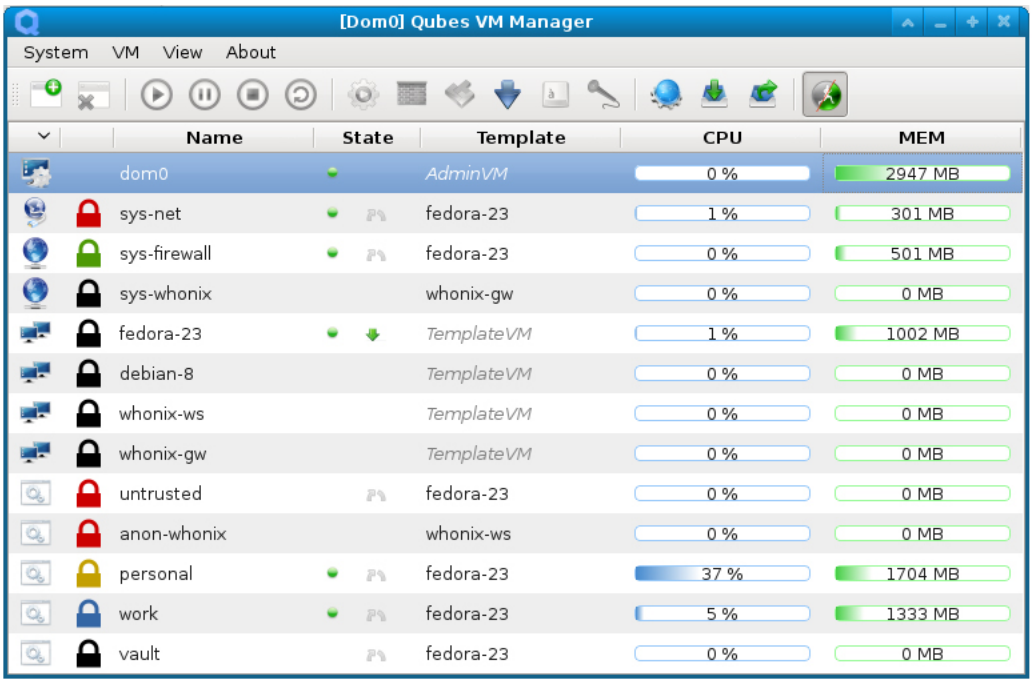

Figura 4.6: Vista principal de Qubes Manager

Para la gestión de las máquinas, destacan ciertas acciones esenciales:

Los controles típicos para el arranque, pausa, detención y reinicio de máquinas virtuales.

Inaccesible para dom0, este botón lanza un configurador para la máquina seleccionada Desde este configurador es posible alterar la cantidad de recursos otorgados a la misma, especificar el tipo de máquina, seleccionar una plantilla para el despliegue y escoger desde que otro *qube* se obtiene la conexión de red; también es posible escoger los dispositivos que estarán disponibles para la máquina, como por ejemplo la tarjeta de red asignada a la *NetVM*.

Muestra el manejador de reglas para ejercer de corta-fuegos, excepto ciertas máquinas como la *NetVM* que no pueden ser configuradas como corta-fuegos. Es posible agregar reglas basándose en protocolo, servicio y dirección de red.

Botón que permite elegir las aplicaciones que se mostrarán en la respectiva sección del menú de inicio.

Lanza la actualización de sistema disponible para la máquina seleccionada.

Cambia la distribución de teclado, el *qube* elegido debe estar en marcha.

Permite o prohíbe el uso del micrófono en la máquina.

Lanza el configurador global de Qubes, donde se presentan diversas opciones para la elección del *kernel* en uso, configuración de red y memoria, y elección de la plantilla por defecto.

Controles para realizar una copia de seguridad de una máquina concreta, o la restauración mediante una copia existente.

Muestra u oculta los *qubes* que no se encuentran en ejecución.

4.3.3. Configuración general

Copiar y pegar entre *qubes*

Qubes proporciona un método seguro y rápido para la transferencia de texto y datos entre *qubes*.

Para la copia de texto únicamente es necesario emplear un portapapeles especial accesible desde todas las máquinas virtuales. El proceso es muy similar al de cualquier sistema operativo, añadiendo los pasos de acceso al portapapeles global. Basta con seleccionar el texto deseado, presionar *Ctrl* + *C* como en cualquier sistema operativo, llevar la copia al portapapeles global con *Ctrl* + *Shift* + *C*; posteriormente, en la máquina destino, obtener el contenido de este con *Ctrl* + *Shift* + *V* y por último pegar el texto en el lugar de destino con *Ctrl* + *V*.

La copia de datos también es extremadamente sencilla, existiendo dos maneras de efectuar la transferencia, mediante el explorador de archivos o mediante línea de comandos. Con el explorador, simplemente se hace clic derecho sobre el archivo y se selecciona la opción "*Copy to Other AppVM*", eligiendo la máquina destino; esto deposita el archivo en la carpeta *"/home/user/QubesIncoming/"* del *qube* objetivo. Para efectuar la transferencia a través de un terminal o línea de comandos, se emplea la función *qvm-copy-to-vm*, cuyo primer argumento es la máquina destino y el segundo el fichero o carpeta a transmitir.

El esquema de copia y pega de texto y archivos es seguro, ya que no permite que una máquina virtual "robe" los datos que se están transmitiendo, de la misma manera que la máquina que emite los datos no tiene permiso para modificar archivos arbitrarios en la máquina destino, se limita a la carpeta *QubesIncoming*. Por otro lado, Qubes no emplea ningún tipo de emulación de *hardware* para la copia de archivos, sino que hace uso de la memoria compartida de Xen, esto elimina en gran medida la necesidad de ejecutar código adicional potencialmente inseguro.

A pesar de las medidas de seguridad, se debe tener en cuenta que el paso de un archivo de una máquina virtual a otra con un nivel de seguridad más alto, siempre es peligroso. El propio archivo podría estar contaminado, propagando la amenaza a los *qubes* más seguros y aislados. Este problema se presenta de la misma manera en sistemas de máquinas físicas independientes, siendo el buen manejo del sistema por parte del usuario la única manera de poder evitar efectivamente la propagación de estas amenazas.

Existe una excepción para la copia de archivos, se presenta al pretender transmitir un archivo de una máquina virtual a dom0. Esto compromete directamente la seguridad del sistema, ya que dom0 es una fina capa de un sistema operativo, pensada para gestionar el resto de *qubes*, no para ejecutar programas o manipular archivos. De esta manera, muy pocos motivos justifican esta acción, aunque es posible mediante el uso del comando *qvm-run* en un terminal de la máquina dom0. La sintaxis para el uso de este comando es:

```
$ qvm-run - -pass-io qube-fuente 'cat /ruta/qube/fuente' >/ruta/destino/dom0
```

***Qubes* desechables (DispVM)**

En Qubes, se pueden tomar diversas acciones mediante el lanzamiento de una máquina virtual exclusiva y ligera, que desaparece al finalizar su uso; normalmente se emplea para hospedar una sola aplicación, como un navegador o un visor de documentos, de manera que si se ve comprometida por alguna amenaza, no afecte a ningún otro *qube* y esta se elimine en el apagado. Estas máquinas heredan la configuración de red de la máquina de la que parten, de manera que usará la misma *NetVM o ProxyVM* para obtener conexión de red.

Para abrir un archivo en una máquina aislada, solo se requiere clic derecho sobre el archivo en cuestión, seleccionar la opción "*Scripts*" y la subopción "*Open in DisposableVM*". También es posible ejecutar un navegador en una máquina desechable desde el menú de inicio, nótese que el navegador es la única aplicación que se recomienda lanzar a través de una *DispVM* directamente desde dom0.

Dos comandos resultan útiles para generar máquinas desechables a través de un terminal, estos son *qvm-open-in-dvm* para abrir un archivo, y *qvm-run* para ejecutar una aplicación. La sintaxis de los mismos es:

```
$ qvm-open-in-dvm carpeta/archivo.ext
```

```
$ qvm-run '$dispvm' aplicación
```

Usar y administrar dispositivos USB

Qubes contempla la posibilidad de crear un *qube* para la gestión de los dispositivos USB, lo que añade seguridad al sistema. Esto es debido a que cualquier dispositivo conectado por USB es automáticamente leído por el sistema para determinar el tipo (teclado, ratón, almacenamiento...), comprometiendo el sistema completo en caso de realizar esta acción desde dom0. Esta opción es elegible en el inicio del sistema, después de la instalación, pero únicamente en caso de que no exista ningún dispositivo conectado por USB en el momento del primer arranque. A pesar de no haber sido configurado en el primer inicio, es posible configurarlo mediante ciertas operaciones con el terminal de dom0.

Para poner en marcha un *qube* de gestión USB por defecto, dos comandos son necesarios, el primero habilita la máquina por defecto para esta tarea (*sys-usb*), el segundo aplica la configuración predeterminada al mismo:

```
$ qubesctl top.enable qvm.sys-usb
```

```
$ qubesctl state.highstate
```

Alternativamente, es posible crearlo manualmente mediante la creación de un nuevo *qube* de tipo "NetVM" o "AppVM", dependiendo de si tendrá acceso a algún dispositivo de red o no. La asignación de los dispositivos USB o PCI se puede realizar a través del panel de configuración o mediante los comandos *qvm-usb* y *qvm-pci* respectivamente. Es importante destacar que un dispositivo asignado a un *qube* de gestión USB dejará de ser utilizable por el sistema dom0, lo que puede dejar inservible el sistema si este se encuentra alojado en algún tipo de almacenamiento USB.

Incluso gestionando los puertos USB mediante una máquina aislada, el equipo es vulnerable a un hipotético ataque durante el arranque de la misma; existe un breve periodo de tiempo en el que, antes de que el *qube* responsable de los USB tome control sobre estos, es dom0 el que los gestiona, por tanto se ve expuesto a una infección a través de los controladores de los dispositivos. Para evitar esta vulnerabilidad, existen dos opciones, siendo la primera desconectar todos los dispositivos USB durante el arranque.

La otra opción es "esconder" todos los dispositivos a dom0, una opción imposible para sistemas con *Anti-Evil-Maid* debido a que este servicio requiere un dispositivo USB conectado directamente a dom0; esto solo es necesario hacerlo para *qubes* de gestión USB generados manualmente (es decir, sin usar el comando *qubesctl* o la opción de configuración en el primer inicio), ya que en caso contrario esta acción se efectúa automáticamente. Para llevar a cabo esta alternativa de forma manual, se requiere la modificación de un archivo del sistema, concretamente "*/etc/default/grub*". Mediante un editor de texto, se busca la línea que comienza por "*GRUB_CMDLINE_LINUX*" y se añade a continuación "*rd.qubes.hide_all_usb*", después de guardar y cerrar el editor, se debe ejecutar el siguiente comando en el terminal de dom0:

```
$ grub2-mkconfig -o /boot/grub2/grub.cfg
```

Ocultar los dispositivos a dom0 acarrea una consecuencia importante, y es que este queda privado de acceso a dispositivos como el ratón y el teclado en caso de que estén conectados por puerto USB; si no se le da permiso de acceso a los mismos, el sistema puede quedar inútil al reiniciar al no poder introducir la contraseña para el inicio de sesión. Para otorgar permiso a dom0, se han de modificar dos archivos desde el *qube* que gestiona los dispositivos, uno para el ratón y otro para el teclado; los archivos son "*/etc/qubes-rpc/policy/qubes.InputMouse*" y "*/etc/qubes-rpc/policy/qubes.InputKeyboard*" respectivamente, se añade la siguiente línea al comienzo de cada archivo (sustituir *sys-usb* por el nombre del *qube* que se emplee):

```
$ sys-usb dom0 ask,user=root
```

Copia de seguridad y restauración

En Qubes es sencillo realizar copias de seguridad o restauraciones, así como migraciones entre máquinas físicas. Estas funciones están disponibles a través de la interfaz de "Qubes Manager", pero también es posible emplear los comandos *qvm-backup* y *qvm-backup-restore* desde un terminal en dom0. Este sistema se basa en un cifrado por clave relativamente débil, por lo que es altamente recomendado seleccionar una clave de alta entropía al generar copias de seguridad.

Para realizar copias de seguridad mediante el "Qubes Manager", se selecciona la subopción *Backup VMs* en la lista desplegable a través del botón *System* de la barra superior. En la interfaz que se muestra aparecen dos columnas, la de la izquierda muestra las máquinas virtuales disponibles para realizar una copia, y a la derecha aparecen las máquinas ya seleccionadas para la copia; mediante los controles, y teniendo en cuenta que las máquinas deben estar apagadas ya que en caso contrario aparecerán en color rojo y será imposible llevar a cabo la copia, se escoge un conjunto de *qubes* y se procede a la siguiente pantalla de la interfaz. Se selecciona el destino de la copia de seguridad, esto es, una "AppVM" o un dispositivo de almacenamiento USB conectado a algún *qube*, y, en caso de desearlo, se escoge una contraseña para el cifrado del archivo de recuperación generado. Esta contraseña es empleada tanto para la comprobación de integridad del fichero como para el descifrado.

Las restauraciones se efectúan de una manera similar, mediante la subopción *Restore VMs from backup* en la misma lista desplegable. En la interfaz, se selecciona el archivo de restauración, especificando la ruta del archivo, y si se encuentra alojado en alguna "AppVM". A continuación, aparecen tres opciones relevantes para el restaurado; activar "*Ignore missing*" hará que la restauración ignore si la máquina dependía de otra para su ejecución y esta no se encuentra presente en el actual sistema, "*Ignore username mismatch*" se emplea para forzar la restauración a pesar de que los nombres de usuario de los sistemas no coincidan, por último "*Skip dom0*" omite el directorio *home* en la restauración de una máquina virtual dom0.

Para la migración de *qubes* entre máquinas físicas, únicamente es necesario seguir el procedimiento anterior, llevando los archivos de restauración a la máquina destino y ejecutando desde la misma una restauración típica.

Pantalla completa

Qubes restringe que las máquinas virtuales tengan acceso a ocupar todo el espacio de pantalla disponible, a la vez que son "obligadas" a mostrar barras coloreadas alrededor de cada ventana, lo que ayuda al usuario a identificar claramente el dominio de cada una. Esta práctica otorga seguridad al sistema, ya que en caso de que los *qubes* tuviesen permiso para ocupar toda la pantalla, sería imposible para el usuario identificar cual de las máquinas ha activado este modo, o si alguna de ellas está simulando un escritorio alternativo para explotar alguna vulnerabilidad del equipo o del usuario.

No obstante, proporciona un método seguro para activar este modo de pantalla mediante la combinación de teclas "*Alt+F11*"; este método asegura que la activación de la pantalla completa es realizada por el usuario, y limita la acción a dom0, quien es encargado de gestionar las ventanas de escritorio. El resto de *qubes*, además de no tener permiso para la activación de la pantalla completa, están preparadas para no poder interceptar este atajo de teclado.

Seguridad adicional

Para evitar el clic accidental en enlaces dudosos en máquinas consideradas seguras, Qubes permite abrir un enlace con otra "AppVM" de manera que la segura no se vea afectada. Esto es posible mediante un terminal, ejecutando el comando *qvm-open-in-vm* que toma como argumentos el *qube* que tomará la acción, y el enlace a un archivo o una web cualquiera.

Es posible automatizar este proceso, de manera que un *qube* siempre abra los enlaces mediante otra máquina virtual. Solo es necesario abrir un editor de texto en el *qube* que se quiera proteger y copiar el siguiente texto:

```
[Desktop Entry]
Encoding=UTF-8
Name=BrowserVM
Exec=qvm-open-in-vm NOMBREDEQUBE
Terminal=false
X-MultipleArgs=false
Type=Application
Categories=Network;WebBrowser;
MimeType=x-scheme-handler/unknown;x-scheme-handler/about;text/html;text/
    xml;application/xhtml+xml;application/xml;application/vnd.mozilla.xul+xml;
    application/rss+xml;application/rdf+xml;image/gif;image/jpeg;image/png;
    x-scheme-handler/http;x-scheme-handler/https;
```

Después de reemplazar "NOMBREDEQUBE" con el nombre de la máquina virtual que ejecutará los enlaces, se debe guardar con el nombre *browser_vm.desktop* en la ruta */.local/share/applications/*. Por último se establece esta forma de abrir enlaces como navegador predeterminado con *xdg-settings set default-web-browser browser_vm.desktop*.

Por otra parte, ciertos sistemas operativos, como puede ser Windows, son propensos a sufrir filtrado de datos personales; es posible que ocurra un filtrado mínimo de estadísticas de uso por parte de una aplicación que captura las acciones que el usuario realiza mediante ella, o es también posible un filtrado masivo de documentos causado por un virus en red. Para prevenir este tipo de ataques, la solución típica en un sistema Qubes implica la creación de una plantilla para el sistema deseado; esta plantilla no contiene ningún dato del usuario en su arranque, por lo que se le otorga acceso de red y se actualiza e instala todo el software requerido; para emplear el sistema, se despliega una "AppVM" sin acceso a red sobre la que se trabaja sin riesgo de filtrado de datos.

4.3.4. Instalación y actualización de sistemas

Para una actualización completa del sistema a una nueva versión de Qubes, el método más seguro consiste en generar copias de seguridad de los *qubes* deseados, siguiendo de una instalación limpia de la nueva versión del sistema y finalmente una restauración de las máquinas; en caso de que las plantillas hayan sido actualizadas también, se deberá especificar las mismas en la configuración individual de cada máquina virtual restaurada.

La actualización e instalación de plantillas pasa por la ejecución de comandos específicos desde dom0 o desde la propia máquina virtual, dependiendo del caso. A continuación se explican los distintos procedimientos para las plantillas más relevantes de Qubes.

Actualización de plantilla Fedora

Fedora, la plantilla predeterminada de Qubes requiere de ejecución de comandos en ambas máquinas virtuales, dom0 y la propia plantilla, para su actualización. Concretamente se especifica el proceso para la actualización de un sistema Fedora 23 a Fedora 24, no obstante el proceso entre distintas versiones será similar, si no igual.

Desde un terminal en dom0, se asegura que la máquina en cuestión está apagada, se clona la misma y se inicia un terminal en la nueva copia de la plantilla mediante los siguientes comandos:

```
$ qvm-shutdown fedora-23
$ qvm-clone fedora-23 fedora-24
$ qvm-run -a fedora-24 gnome-terminal
```

Posteriormente, desde el terminal del clon de la plantilla (es posible que se requiera la confirmación clave GPG, en cuyo caso se introduce "y"):

```
$ sudo dnf clean all
$ sudo dnf –releasever=24 distro-sync
```

Se apaga la nueva plantilla desde dom0 y se "aligera" la misma mediante una herramienta de Qubes:

```
$ qvm-shutdown fedora-24
$ qvm-trim-template fedora-24
```

Esto deja el sistema con una plantilla totalmente funcional, lista para reemplazar a la antigua. Para ello basta con emplear la interfaz de Qubes Manager para seleccionarla como plantilla por defecto y asignar las máquinas existentes a la nueva plantilla mediante su panel de configuración individual. Por último, si se requiere, se elimina la plantilla obsoleta mediante el comando:

```
$ sudo dnf remove qubes-template-fedora-23
```

Actualización de plantilla Debian

Qubes ofrece los archivos binarios para la instalación de Debian 7 y Debian 8. Ambas instalables mediante un único comando ejecutado desde dom0:

```
$ sudo qubes-dom0-update qubes-template-debian-7
```

```
$ sudo qubes-dom0-update qubes-template-debian-8
```

No obstante, Qubes permite la actualización de una plantilla Debian 8 a Debian 9 mediante un proceso notablemente similar al anterior. Los primeros comandos deben ser introducidos en un terminal de dom0, estos aseguran que la plantilla no se encuentre en ejecución y, al igual que en el caso de Fedora, clonan la plantilla y ejecutan un terminal sobre ella:

```
$ qvm-shutdown debian-8
$ qvm-clone debian-8 debian-9
$ qvm-run -a debian-9 gnome-terminal
```

En este punto, se requiere actualizar los repositorios para que se use la nueva versión de los mismos; hecho esto, se procede a actualizar la lista de paquetes disponibles y la propia distribución de Debian. Durante este proceso, es posible que se requiera confirmación para el reemplazo de dos archivos, *qubes-r3.list* que si debe ser reemplazado, y *pulse/client.conf*, que deberá permanecer intacto. Como último paso sobre el terminal de la plantilla, se eliminan los paquetes adicionales innecesarios que se instalaron previamente.

```
$ sudo sed -i 's/jessie/stretch/g' /etc/apt/sources.list
$ sudo sed -i 's/jessie/stretch/g' /etc/apt/sources.list.d/qubes-r3.list
$ sudo apt-get update && sudo apt-get dist-upgrade -y
sudo apt-get autoremove
```

Los últimos pasos consisten en la ejecución de tres comandos sobre dom0; el primero apaga la plantilla, el segundo la "aligera" y el último elimina la plantilla obsoleta de Debian 8.

```
$ qvm-shutdown debian-9
$ qvm-trim-template debian-9
$ sudo yum remove qubes-template-debian-8
```

Actualización de plantilla Whonix

La actualización de este sistema es sencilla y se realiza desde los terminales de ambos *qubes* Whonix. Ambos deberán ser configurados para obtener conexión a la red.

Ya que Whonix se compone de dos máquinas virtuales, se debe ejecutar el siguiente comando sobre ambas plantillas:

```
$ sudo apt-get update && sudo apt-get dist-upgrade
```

Una vez finalizado, el terminal debe mostrar una línea informando de haber leído la lista de paquetes correctamente. Para que la actualización resulte efectiva, se reinician ambas plantillas Whonix.

Instalación de plantilla Archlinux

Archlinux cuenta con una plantilla precompilada y preconfigurada, mantenida por un miembro de la comunidad de Qubes. También es posible la compilación manual de un sistema Archlinux para Qubes, pero se aleja del objetivo de Qubes de generar seguridad fácilmente.

Para la instalación de la plantilla Archlinux, basta con la ejecución de un comando en un terminal sobre dom0:

```
$ sudo qubes-dom0-update –enablerepo=qubes-templates-community
$ qubes-template-archlinux
```

Instalación de plantilla Windows

El proceso para realizar una instalación satisfactoria de Windows y una posterior "adaptación" para que esta pueda ser usada como plantilla para desplegar máquinas virtuales AppVM es relativamente más complicado. Qubes ofrece soporte directo para Windows 7, por lo que el siguiente proceso está orientado a esta versión. Por otra parte, se asume que ya se posee una imagen de instalación de Windows alojada en el sistema de archivos de dom0.

Para comenzar, es necesario generar un nuevo *qube* de tipo HVM mediante el siguiente comando ejecutado en dom0, este asigna el color verde al gestor de ventanas de Windows:

```
$ qvm-create win7 –hvm –label green
```

Posteriormente, se inicia la nueva máquina asignando la imagen de instalación de Windows a la unidad virtual de reproducción de discos ópticos:

```
$ qvm-start win7 –cdrom=/usr/local/iso/win7_en.iso
```

Al iniciar la máquina, se presenta el típico instalador de Windows, el cual una vez finalizado deberá reiniciar la máquina virtual para completar la configuración; Qubes no permite que una máquina virtual se inicie automáticamente, por lo que es necesario iniciar la máquina manualmente cuando el instalador lo requiera.

Esto deja el sistema con un *qube* preparado para ejecutar una versión completa de Windows; sin embargo si se pretende emplear esta instalación simplemente como una plantilla para desplegar diversas AppVM, es necesario seguir una serie de pasos para realizar la instalación de unas herramientas que Qubes ofrece para la gestión de plantillas Windows.

En primer lugar, se deben obtener estas herramientas en la máquina anfitriona dom0, cabe destacar que este comando sólo prepara el *software* para ser enviado a la plantilla Windows, nunca se ejecuta sobre dom0:

```
$ sudo qubes-dom0-update qubes-windows-tools
```

Posteriormente, en el sistema Windows, se ha de desactivar el mecanismo de Windows que prohíbe la instalación de controladores no firmados; el paquete que se procede a instalar no está firmado por una entidad certificadora, no obstante el equipo de Qubes planea obtener las firmas necesarias en un plazo reducido de tiempo. Para ello, sólo se ha de iniciar la línea de comandos de Windows en modo administrador, ejecutar el comando *bcdedit /set testsigning on* y apagar la máquina virtual.

Por último, se emplea un terminal en dom0 para iniciar la plantilla Windows transfiriendo el paquete previamente obtenido.

```
$ qvm-start lab-win7 –install-windows-tools
```

El paquete debe aparecer como un medio de instalación por disco óptico, típicamente visible desde "Mi PC". Tras una instalación satisfactoria del paquete, el sistema Windows debe reiniciar de nuevo para finalmente estar preparado para actuar de plantilla. El proceso para generar sistemas basados en esta plantilla es el mismo que para el resto de sistemas.

4.3.5. Instalación y actualización de aplicaciones en máquinas virtuales

Como se ha mencionado en el capítulo tres, las máquinas de tipo "AppVM" obtienen su raíz de ficheros de la plantilla sobre la que se despliegan, y únicamente tienen como "propias" las carpetas */home* y */usr/local*; esto implica que cualquier aplicación instalada en este tipo de máquinas será eliminada en el apagado, excepto si esta es instalada en alguna de las mencionadas carpetas, lo cual es una práctica no recomendada. Por tanto, para instalar y mantener aplicaciones en las diversas máquinas, se emplea un terminal o un centro de *software* sobre la plantilla a tratar.

4.3.6. Personalización

Qubes impide la instalación de un entorno de escritorio cualquiera, este debe estar preparado para manejar las distintas características únicas de Qubes. Aunque esto limita las opciones que el usuario tiene para configurar el escritorio y las interfaces, Qubes actualmente cuenta con soporte para tres entornos, ofreciendo algo más de adaptabilidad a las preferencias del usuario.

Instalación de KDE

Previamente a la versión actual de Qubes (3.2), KDE era el escritorio por defecto; este ofrece una mayor capacidad para la personalización, a costa de un peor rendimiento y una mayor utilización de los recursos del equipo. Estos inconvenientes supusieron suficientes motivos para que Qubes optase por adoptar un entorno más ligero y sencillo como es XFCE.

No obstante, KDE es todavía completamente funcional y se encuentra disponible en los repositorios oficiales de Qubes. Para instalarlo, desde un terminal en el sistema dom0:

```
$ sudo qubes-dom0-update @kde-desktop-qubes
```

También es posible cambiar la pantalla de inicio de sesión por defecto en Qubes (*lightdm*), por la nueva de KDE (*sddm*). Es necesaria la modificación del archivo */etc/sddm.conf* para que sea acorde a la siguiente línea:

```
[XDisplay]
ServerArguments=-nolisten tcp -background none
```

Después, desde un terminal en dom0, se desactiva el servicio antiguo y se habilita el nuevo mediante:

```
$ sudo systemctl disable lightdm
$ sudo systemctl enable sddm
```

Por último, es necesario reiniciar el equipo completo.

Instalación de i3

Recientemente, Qubes ha introducido soporte para el entorno de escritorio i3; aunque este se encuentra en periodo de pruebas, está disponible para su descarga mediante los repositorios de *testing* y es completamente funcional. Para obtenerlo, junto a la configuración por defecto para Qubes, se ejecuta el siguiente comando mediante un terminal en dom0:

```
$ sudo qubes-dom0-update –enablerepo=qubes-dom0-current-testing i3-settings-qubes
```

Después de cerrar sesión, el nuevo entorno será seleccionable desde el desplegable de la pantalla de inicio de sesión.

Instalación de XFCE

Este es el entorno recomendado por Qubes, debido a su simplicidad y ligereza es capaz de ofrecer un mejor rendimiento y una reducida cantidad de errores de uso. Para las versiones anteriores a Qubes 3.2, o para un sistema en el que se haya desinstalado este entorno, es posible que surja la necesidad de instalarlo; además XFCE aún no cuenta con una configuración por defecto para Qubes, por lo que es necesario realizar alguna modificación para adaptarlo.

Para instalarlo en el sistema, se ejecuta una sola orden desde un terminal en dom0 y se reinicia la máquina:

```
$ sudo qubes-dom0-update @xfce-desktop-qubes
```

En cuanto a la configuración, se recomienda en primer lugar eliminar los accesos a aplicaciones de dom0 en el menú de inicio, ya que nunca las emplearemos directamente desde este.

Por otro lado, el menú de XFCE original no cuenta con las subsecciones para distinguir el conjunto de aplicaciones de cada máquina virtual. Esto se especifica mediante el fichero */.config/menus/favorites.menu*, el cual sigue la siguiente sintaxis:

```
<!DOCTYPE Menu PUBLIC //freedesktop//DTD Menu 1.0//EN"
"http://www.freedesktop.org/standards/menu-spec/1.0/menu.dtd»
<Menu>
    <Name>Favorites</Name>
    <DefaultAppDirs/>
    <DefaultDirectoryDirs/>
    <Directory>favorites.directory</Directory>
    <Include>
        <Filename>personal-gnome-terminal.desktop</Filename>
        <Filename>personal-firefox.desktop</Filename>
        <Filename>work-gnome-terminal.desktop</Filename>
        <Filename>work-firefox.desktop</Filename>
        <Filename>mail-mozilla-thunderbird.desktop</Filename>
        <Filename>mail-gnome-terminal.desktop</Filename>
        <Filename>banking-mozilla-firefox.desktop</Filename>
        <Filename>untrusted-firefox.desktop</Filename>
    </Include>
</Menu>
```

Una vez generado el archivo, mediante clic derecho en el panel, se selecciona como archivo de configuración del menú de XFCE.

CAPÍTULO 5
Análisis y evaluación

En este capítulo se presenta una evaluación del sistema real instalado en un portátil y se compara con respecto a un sistema operativo corriente en términos de rendimiento y usabilidad.

5.1 Entorno gráfico

En cuanto al entorno gráfico y al aspecto visual del sistema, este no difiere en gran medida de cualquier distribución de Linux como Debian o Ubuntu; posee el mismo sistema de ficheros, pudiendo instalar cualquier *software* para su gestión. El sistema de gestión de ventanas es el típico al que los usuarios están acostumbrados, únicamente agregando un color característico al borde de las ventanas para identificar a qué máquina virtual pertenecen. Por un lado, el sistema no requiere de un aprendizaje prolongado para hacer un buen uso del entorno gráfico, no obstante, sí obliga al usuario a establecer una organización estricta de las aplicaciones por dominios para aislar los posibles ataques.

Al iniciar el sistema, se presenta una pantalla de inicio de sesión típica de cualquier sistema Linux en la que se pide el nombre de usuario y la contraseña.

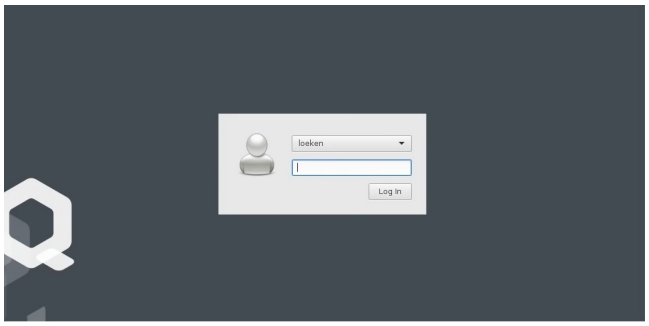

Figura 5.1: Pantalla de inicio de sesión

Tras esta pantalla Qubes presenta el escritorio y la barra de inicio, donde de forma predeterminada aparece el menú de aplicaciones, Qubes Manager y el gestor de ventanas. La apariencia es la de un escritorio corriente XFCE, la diferencia más obvia radica en la subdivisión del menú de aplicaciones, en este se estructuran las aplicaciones por carpetas que representan los dominios configurados en el sistema, identificados por un color que será usado para colorear los bordes de las ventanas de este dominio.

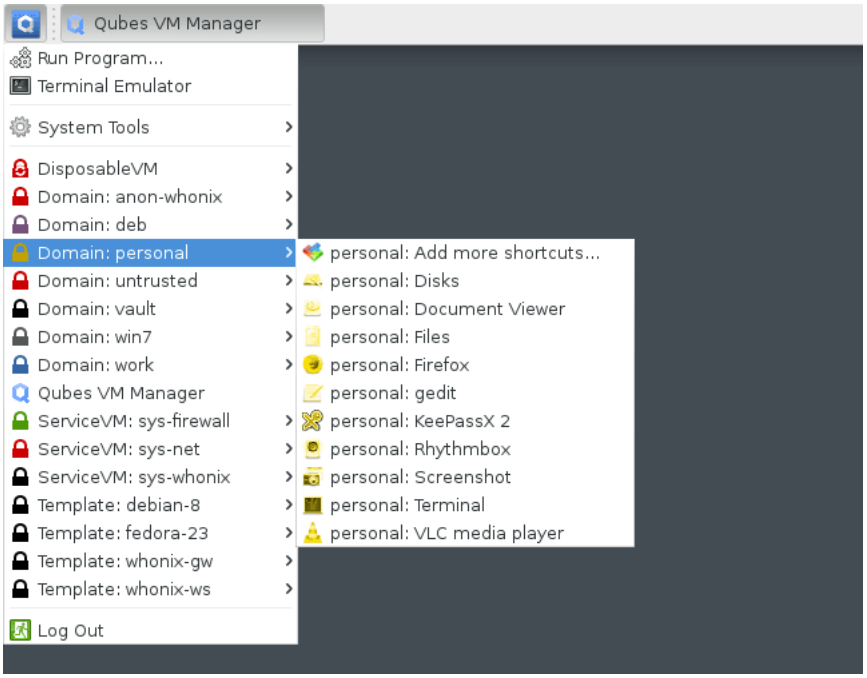

Figura 5.2: Menú de aplicaciones

El programa predeterminado para manejar el sistema de ficheros es Thunar, ofrece lo que cualquier programa de gestión de ficheros podría ofrecer, con una excepción. En Qubes, cada máquina virtual tiene su propio sistema de ficheros, por lo que copiar y pegar archivos entre máquinas virtuales no es tan trivial. Para ello, Qubes cuenta con modificaciones en estos programas que añaden la funcionalidad para poder enviar ficheros entre qubes.

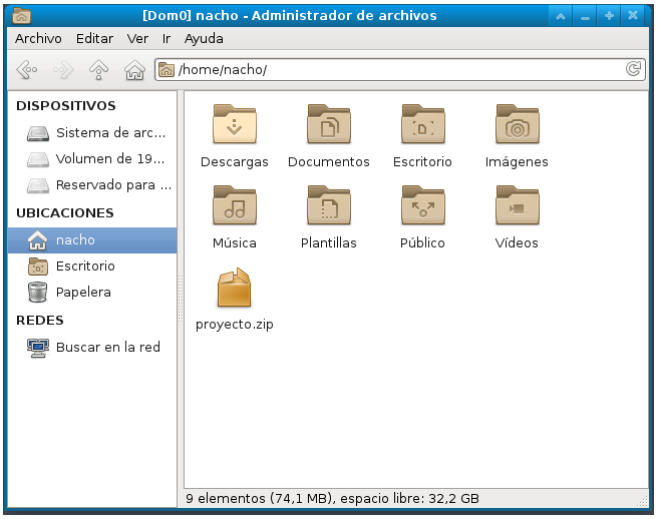

Figura 5.3: Explorador de archivos Thunar

Mediante el click derecho de ratón en un archivo, se muestran varias opciones adicionales para mover, copiar o abrir el archivo en una máquina virtual temporal (DispVM).

Por lo demás, Qubes es un sistema operativo aparentemente corriente, que cuenta con las limitaciones típicas de un sistema Linux para un usuario estándar. Es algo más tedioso para instalar programas, teniendo que emplear comandos en el terminal ocasio-

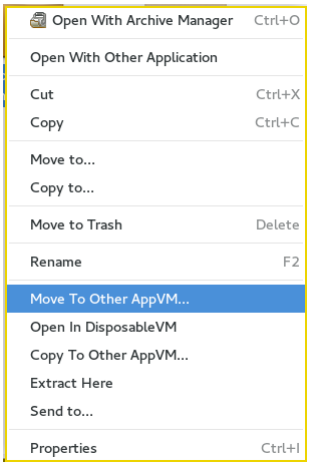

Figura 5.4: Menú click derecho

nalmente, se configura de forma distinta (aunque mucho más amplia) que un sistema Windows o Mac... No obstante, su capacidad para virtualizar de forma eficiente varias máquinas virtuales Windows, hacen de Qubes una buena opción para usuarios menos avanzados, pudiendo establecer una configuración mínima en dom0 para solo emplear máquinas virtuales Windows, aprovechando la capacidad de aislamiento frente ataques, pero conservando la usabilidad y la variedad de *software* propietario que ofrece el sistema de Microsoft.

5.2 Ventajas e inconvenientes

De forma general, se puede enumerar una serie de ventajas que claramente posicionan a Qubes como un sistema más seguro; no obstante, también existen inconvenientes que lo hacen menos "amigable".

En cuanto a las ventajas, resaltan ciertas características que posee Qubes frente a un sistema operativo nativo corriente:

- El aislamiento de partes del sistema por dominios, estableciendo un uso para cada uno y aislando amenazas.

- La capacidad de virtualizacion de multiples sistemas de manera eficiente.

- Ofrecer un innegable aumento en el rendimiento de los sistemas operativos virtualizados con respecto a una virtualización realizada sobre un sistema operativo corriente.

- Establecer un sistema de configuración de red complejo, permitiendo el aislamiento de ciertos dominios para que sean innaccesibles desde la red.

Los inconvenientes que presenta Qubes son principalmente debidos a su estructura centrada en la seguridad, y en su mayor parte afectan a la usabilidad del sistema, entre ellos se sitúan:

- Poseer un sistema de ficheros independiente para cada máquina virtual, aunque ofrece métodos para el paso de ficheros, el manejo de estos resulta algo más complicado que el de un sistema de ficheros unificado.

- Necesidad de configurar la conexión de red para cada dominio mediante el Qubes Manager.

- Consumir en torno a un 10 % de los recursos del sistema para la ejecución del propio hipervisor, ralentizando ligeramente la ejecución del sistema y los programas.

- No contar con controladores privativos, que aunque en gran parte existan versiones de código libre, ocasionalmente presentan errores.

5.3 Rendimiento

Qubes presenta un rendimiento alto en cuanto a virtualización de sistemas, lo cual no exime a este de presentar pérdidas de rendimiento con respecto a sistemas nativos. Descartando que este sistema sea apto para máquinas antiguas, que ni siquiera cuenten con la opción de virtualización por *hardware* en su procesador, se puede lidiar con la pérdida de rendimiento en máquinas modernas en las que usualmente el procesador no opera a pleno rendimiento en operaciones normales del sistema operativo.

La interfaz y, en general, la respuesta a las acciones del usuario, sigue siendo rápida en un portátil de gama media; aunque se puede percibir cierta latencia con respecto al sistema nativo, es mínima y no supone una incomodidad.

La memoria RAM podría ser un problema en estos sistemas si no se dispone de una cantidad mínima, ya que las diversas virtualizaciones simultáneas requieren de su porción aislada de memoria. En el sistema probado se dispone de 8GB de memoria, siendo esta cantidad suficiente para una operación normal. Si se pretendiese configurar un equipo para ejecutar un gran número de virtualizaciones, el uso de memoria sería notablemente más alto, requiriendo más capacidad RAM en el sistema.

Por otro lado, el uso de tarjetas gráficas de alto rendimiento está altamente limitado. Cualquier sistema Linux presenta esta desventaja, ya que los grandes fabricantes de tarjetas gráficas usualmente no ofrecen controladores adaptados para estos sistemas, y si los hay, suelen presentar fallos visibles. Sin embargo Linux no se caracteriza por ser un sistema orientado a videojuegos ni a desarrollo 3D, por lo que una controladora gráfica básica es suficiente para mantener un buen rendimiento en el sistema operativo.

El procesador es otro componente que se ve afectado por la necesidad de mantener en ejecución el hipervisor y las herramientas de virtualización; esta carga extra repercute en el rendimiento del equipo, pero el hipervisor Xen es capaz de reducir esta carga hasta situarse en torno al 10 % de pérdida de rendimiento. En concreto, en el portátil en el que se ha probado el sistema, se ha obtenido un resultado muy notable en una prueba de rendimiento.

Figura 5.5: Resultado de prueba de rendimiento

Figura 5.6: Valor medio del procesador en Geekbench

Como se ve en las figuras, el test de rendimiento realizado sobre el equipo resulta en una puntuación de en torno a 3700. El valor medio en la puntuación de este procesador

en la página web del servicio de tests es 2960. Estos valores dependen de la configuración de *hardware* del equipo, pero son suficientes para comprobar que la máquina se sitúa en un rendimiento muy aceptable.

Con estos datos, se puede afirmar que el rendimiento del sistema operativo es completamente apto para establecer un equipo de trabajo seguro, pero no para un sistema de diseño 3D o plataforma de juego. Esto satisface las expectativas de Qubes, pensado para proteger los datos y mantener un sistema limpio y estable.

CAPÍTULO 6
Conclusiones

En este capítulo se expone un análisis general del resultado del estudio y se plantean diversas aplicaciones de un sistema como el investigado.

6.1 Comprobación de objetivos

A través de Qubes OS se consigue la implementación de un sistema basado en el aislamiento, capaz no solo de agregar seguridad a un equipo, sino también de separar las áreas vulnerables y aislar una amenaza evitando su propagación.

Por otra parte, Qubes permite la ejecución virtualizada de la mayoría de los sistemas operativos actuales, lo que lo convierte en un entorno multisistema con soporte para programas basados en distintas plataformas.

No obstante, un sistema como el estudiado no proporciona una seguridad infalible; esta depende directamente del usuario, siendo el principal responsable del buen estado de la máquina. Es por esto que una sección del cuerpo de la memoria se destina a exponer diversos aspectos del manejo del sistema.

En definitiva, se consigue establecer un sistema seguro y altamente configurable, con capacidad de ejecutar aplicaciones de distintas plataformas .

6.2 Conclusión

Aunque Qubes OS requiere de unos conocimientos relativamente altos en configuración de sistemas, se presenta como una opción adaptable a diversos tipos de usuario para generar un entorno segmentado seguro. Además cuenta con asistentes de configuración que automatizan ciertos procesos de configuración, facilitando su despliegue.

Representa una solución apta para equipos personales de un uso más “doméstico”, pero a su vez suficientemente adaptable y compleja para poder ser implementada en entornos empresariales. Debido a su naturaleza basada en el particionado, ofrece mecanismos rápidos y fiables para frenar y aislar amenazas en el equipo.

Habiendo realizado este proyecto, puedo afirmar que los métodos tradicionales de seguridad en equipos informáticos están quedándose obsoletos, lejos de seguir el ritmo de desarrollo de las amenazas. Es por ello que se deben buscar otros métodos de protección digital como el particionado.

Bibliografía

[1] Adrian Raya, "El antivirus está muerto" John McAfee ¿Por qué tiene razón?, emitido el 3 de septiembre de 2015. Consultado en http://omicrono.elespanol.com/2015/09/muerte-del-antivirus-john-mcafee/.

[2] Michael Larabel, Ubuntu 15.10: KVM vs. Xen vs. VirtualBox Virtualization Performance, emitido el 22 de octubre de 2015. Consultado en http://www.phoronix.com/scan.php?page=article&item=ubuntu-1510-virt&num=1.

[3] Rapid7 Company, Common Types of Cybersecurity Attacks. Consultado en https://www.rapid7.com/fundamentals/types-of-attacks/.

[4] Mike Cardwell, Protecting a Laptop from Simple and Sophisticated Attacks, emitido el 26 de agosto de 2011. Consultado en https://www.grepular.com/Protecting_a_Laptop_from_Simple_and_Sophisticated_Attacks.

[5] Xen Project, Why Xen Project? Consultado en https://www.xenproject.org/users/why-the-xen-project.html.

[6] M. Tim Jones, La anatomía de un hipervisor Linux, emitido el 31 de mayo de 2009. Consultado en https://www.ibm.com/developerworks/ssa/library/l-hypervisor/index.html.

[7] Xen Project, Xen Wiki. Consultado en https://wiki.xen.org/wiki/.

[8] The Qubes OS Project and others, Qubes OS User Documentation. Consultado en https://www.qubes-os.org/doc/.

[9] 2015's MVPs – The most vulnerable players, TechTalk GFI Software. Consultado en https://techtalk.gfi.com/2015s-mvps-the-most-vulnerable-players/.

[10] Processos Benchmarks, Geekbench Browser. Consultado en https://browser.geekbench.com/processor-benchmarks.

[11] Brian Ward. *How Linux Works, what every superuser should know*. No Starch Press, San Francisco, Segunda edición, 2015.

[12] Michael K. Johnson, Erik W. Troan. *Linux Application Development*. Addison-Wesley Professional, Massachussets, Segunda edición, 2004.

[13] Carlos Pérez, Wladimiro Díaz. *La virtualización como soporte para SOA y Cloud Computing*. Novática: Revista de la Asociación de Técnicos de Informática, España, 2010.