Comparación de sobrecarga de rendimiento entre el hipervisor y el contenedor basado

Virtualización

Zheng Li∗, Maria Kihl∗, Qinghua Luy Jens A. Andersson∗

departamento de ∗ de Tecnología Eléctrica y de la Información, Universidad de Lund, Lund, Suecia

Correo electrónico: zheng.li,maria.kihl, jens a.andersson[eit.lth.se](mailto:@eit.lth.se) [.](mailto:@eit.lth.se)

•Colegio de Ingeniería Informática y de Comunicación, Universidad China de Petróleo, Qingdao, China

[Correo electrónico: dr.qinghua.lu@gmail.com](mailto:lu@gmail.com)

Resumen: la solución de virtualización actual en la nube se basa ampliamente en tecnologías basadas en hipervisores. Junto con la reciente popularidad de Docker, el virtu- alización basado en contenedores comienza rmás atención por ser una alternativa prometedora. Dado que ambas soluciones de virtualización no están exentas de recursos, sus sobrecargas de rendimiento darían lugar a impactos negativos en la calidad de los servicios en la nube. Para ayudar a comprender fundamentalmente la diferencia de rendimiento entre estos dos tipos de soluciones de virtualización, utilizamos una máquina física con recursos "bastamente" como línea de base para investigar la sobrecarga de rendimiento de un standalone contenedor de Docker contra una máquina virtual independiente (VM). Con hallazgos contrarios al trabajo relacionado, nuestros resultados de evaluación muestran que la sobrecarga de rendimiento de la virtualización podría variar no sólo en función de la característica, sino también trabajo a trabajo. Aunque la solución basada en contenedores es sin duda ligera, la tecnología basada en hipervisor no viene con una mayor sobrecarga de rendimiento en todos los casos. Por ejemplo, los contenedores Docker exhibenr QoS en términos de velocidad de transacción de almacenamiento.

arXiv:1708.01388v1 [cs. DC] 4 ago 2017

Términos de índice: servicio en la nube; Contenedor; Hipervisor; Perfor- mance Overhead; Tecnología de virtualización

1. Introducción

Como elemento clave de la informática en la nube, la virtualización desempeña varios roles vitales en el soporte de los servicios en la nube, desde el aislamiento de recursos hasta el aprovisionamiento de recursos. Las tecnologías de virtualización existentes pueden ser más o menos distintivashed entre las soluciones basadas en hipervisor y las soluciones basadas en contenedores. Teniendo en cuenta su propio consumo de recursos, ambas soluciones de virtualización inevitablemente introducen sobrecargas por formancia al ofrecer servicios en la nube, y losgastos generales de rendimientopodrían dar lugar a impactos negativos en la calidad de servicio (QoS) correspondiente. Por lo tanto, sería crucial que tanto los proveedores de la nube (por ejemplo, para mejorar la eficiencia de la infraestructura) como los consumidores (por ejemplo, para seleccionar servicios sabiamente) entiendan a qué extensión una solución de virtualización candidata incurre en influencia en la Calidad de Servicio de la nube.

Dadas las características de estas dos so-

luciones (cf. el fondo de la Sección 2), un hi-

latesis po podría ser: un servicio basado en contenedores muestra un mejor rendimiento que su servicio de máquina virtual basado en hipervisor correspondiente. Sin embargo, hay poca evidencia cuantitativa que ayude a probar esta hipótesis de una manera "manzana a manzana", except para esas discusiones cualitativas. Por lo tanto, des- cided para utilizar una máquina física con recursos "bastantes" como línea de base para investigar cuantitativamente y comparar las sobrecargas de rendimiento entre lasvirtualizaciones basadas en contenedores ybasadas en rvisor de bombo. En particular, dado que Docker es actualmente la solución de contenedor más popular [1] y VMWare es uno de los líderes en el mercado de hipervisores [2], elegimos Docker y VMWare Workstation 12 Pro para representar lassoluciones de virtualización tw orespectivamente.

Según las aclaraciones de [3], nuestra

investigaciones pueden ser reguladas por la disciplina de la informática ex perimental (ECS). Al emplear la metodología basada en conocimientos de dominio (DoKnowMe) recientemente disponible de ECS [4], exploramos experimentalmente los gastos generales de perfor- mance de diferentes soluciones de virtualización sobre unabase de característica porfe.

Los resultados y análisis experimentales muestran que el

hipótesis antes mencionada no es cierto en todos los casos. Por ejemplo, no observamos la dif- ferencia de rendimiento de cálculo entre esos tipos de servicio con respecto a la resolución de un problema de ajedrez combinatoriamente duro; y el contenedor incluso conduce a una mayor sobrecarga de rendimiento de almacenamiento que la máquina virtual al leer/escribir datos byte a byte. Además, encontramos que la notable pérdida de rendimiento incurrida por ambassoluciones de iones virtualizatgeneralmente aparece en la variabilidad del rendimiento. En general, las contribuciones de este trabajo son triples:

• Nuestros resultados y análisis experimentales pueden ayudar tanto a los investigadores como a los profesionales amantener mejorel rendimiento fundamental de las tecnologías de virtualización basadas en contenedores e hipervisores. No es de tablas que la ingeniería de rendimiento en ECS se pueda distinguir aproximadamente entre dos etapas: la primera etapa es to revelar el rendimiento primario de característicasespecíficas (sistema), mientras que la segunda etapa se basa generalmente en la evaluación de la primera etapa para investigar casos de aplicación reales. Por lo tanto, este trabajo puede ser visto como una base para sementales de evaluación más sofisticados en el futuro.

• Nuestro método de cálculo de la sobrecarga de rendimiento se puede aplicar o adaptar fácilmente a diferentes escenarios de evaluación por otros. La literatura muestra que la "sobrecarga por formancia" nose hautilizado oralmente en el contexto de discusiones cualitativas. Mediante la cuantificación de este indicador, nuestro estudio proporciona esencialmente una lente concreta en el caso de las comparaciones de rendimiento.

Aplicación VM

Bins/ Libs

Aplicación VM

Bins/ Libs

Aplicación de aplicación vm VM VM Aplicación

Papeleras/ Papeleras/ Bins/ Libs Libs Libs Libs

Invitado Invitado

Hipervisor del so del sistema operativo OS (Tipo 2)

Sistema operativo Host

Física MachiNe

Aplicación VM

Bins/ Libs

Contenedor

Aplicación

Bins/ Libs

Contenedor

Aplicación

Contenedor

Aplicación

• Toda la lógica de evaluación y los detalles notificados en este documento se pueden ver como una plantilla reutilizable y trazable para evaluar contenedores Docker. Dado que el proyecto Docker sigue creciendo rápidamente [5], la evalua-

Huésped

OS

Huésped

Hipervisor del sistema operativo

Huésped

OS

Bins/Libs

Contenedor contenedor

Aplicación Aplicación Aplicación

Bins/ Libs

Bins/Libs

Motor de contenedores

Sistema operativo Host

Máquina física

Motor de contenedores

resultados podrían estar gradualmente desactualizados. Beneficiándose de esta plantilla, las evaluaciones futuras pueden ser convenientemente repetidas o replicadas por diferentes evaluadores en diferentesmomentos y locciones.

El resto de este documento se organiza de la siguiente manera. Sec-

Sistema operativo host

Máquina física

(a) Ser- vicio virtual basado en hipervisor.

Sistema operativo host

Máquina física

(b) Servicio virtual basado en contenedores.

La ción 2 resume el conocimiento de fondo de las virtualizaciones basadas en contenedores y en los hipervisores y las luces altas del trabajo existente relacionado con sus comunicaciones de rendimiento. La Sección 3 introducela implementación de la evaluación del desempeño, incluida la metodología empleada en nuestro estudio. La investigación detallada de los gastos generales de rendimiento se divide en dos partes de la presentación de informes, a saber, las actividades pre-experimentales y los resultados experimentalesy losingenuos, y se describen correspondientemente en la Sección 3.2 y 3.3 respec- tively. Las conclusiones y algunos trabajos futuros se examinan en la Sección 4.

2. Antecedentes y trabajo relacionado

Cuando se trata de la virtualización en la nube, el ion solut de factoconsiste en emplear las tecnologías basadas en hipervisores, y el tipo de servicio en la nube más representativo ofrece máquinas virtuales (VM) [6]. En esta solución de virtualización, el hipervisor administra los recursos informáticos físicos y makes sectores aislados de hardware disponibles para crear máquinas virtuales [5]. Podemos distinguir aún más entre dos tipos de hipervisores, a saber, el hipervisor de metal desnudo que está en- estancado directamente en el hardware informático, y el hipervisor alojado que requiere un sistema operativo host (SO). Para hacer un mejor contraste entre los conceptos relacionados con el hipervisor y los relacionados con el contenedor, enfatizamos especialmente el segundo tipo de hipervisor, como se muestra en la Figura 1a. Dado que lavirtualización basada en hypervisor solo proporciona acceso al hardware físico, cada máquina virtual necesita una implementación completa de un sistema operativo invitado, incluidos los archivos binarios y las bibliotecas necesarias para las aplicaciones [7]. Como resultado, el sistema operativo invitado inevitablemente incurrirá en competition de recursos contra las aplicaciones que se ejecutan en el servicio de máquina virtual y, esencialmente, degradará el QoS desde la perspectiva de la aplicación.

Para aliviar la sobrecarga de rendimiento de los hipervisores

la virtualización, los investigadores y los profesionales comenzaron recientemente a promover una solución alternativa y ligera, a saber, la virtualización basada en contenedores. De hecho, la base de la tecnología de contenedores se remonta al comando chroot de Unix en 1979 [7], mientras que este technology finalmente se convierte en mecanismos de virtualización como Linux VServer, OpenVZ y Linux Containers (LXC) a lo largo de

Figura 1. Diferentes arquitecturas de hipervisor y basado en contenedores

servicios virtuales.

con el auge de Linux [8]. A diferencia de la solución de nivel de hardware de los hipervisores, los contenedores realizan la virtualización en el nivel de sistema operativo y utilizan sectores aislados del sistema operativo host para proteger sus aplicaciones contenidas [7]. En essence, un contenedor se compone de una o más imágenes ligeras, y cada imagen es un sistema de archivos prebaked y reemplazable que incluye los archivos binarios necesarios, bibliotecas o middlewares para ejecutar la aplicación. En el caso de varios images, los sistemas de archivos compatibles de solo lectura se apilan uno encima del otro para atender el sistema de archivos de capa superior grabable [1]. Con este mecanismo, como se muestra en la Figura 1b, los contenedores permiten que las aplicaciones compartan el mismo sistema operativo e incluso binarios/librarcuando corresponda. Como tal, en comparación con las máquinas virtuales, los contenedores serían más eficientes en los recursos al excluir la ejecución del hipervisor y el sistema operativo invitado, y más eficiente en el tiempo al evitar el arranque (y el apagado) de todo un sistema operativo [5], [9]. Menos,se ha identificado que las capas en cascada de las imágenes de contenedor vienen con una complejidad inherente y una penalización de rendimiento [10]. En otras palabras, la tecnología de virtualización basada en contenedor también podría afectar negativamente a laQoS correspondientedebido a su sobrecarga de rendimiento.

Aunque la ventaja de rendimiento de los contenedores

investigado en varios estudios pioneros [2], [8], [11], la solución de virtualización basada en contenedores no obtuvo la popularidad de signif- icant hastaque las recientes mejoras subyacentes en el kernel de Linux, y especialmente hasta la aparición de Docker [12]. Partiendo de un proyecto de código abierto a principios de

2013 [5], Docker se convierte rápidamente en la solución de con- tainer más popular [1] al facilitar laadministración de contenedores. Técnicamente, al ofrecer el conjunto de herramientas unificado y la API, Docker alivia la complejidad de utilizar las técnicas de nivel de kernel relevantes, incluido el LXC, el cgroup y un filesystem de copia enescritura. Para examinar el rendimiento de los contenedores Docker, se han utilizado un software de simulación de modelado molecular [13] y una aplicación Joomla basada en base de datos postgreSQL [14] para comparar el entorno de Docker con el entorno de máquina virtual..

El trabajo más cercano al nuestro es el estudio orientado a la CPU

[15] y el informe de investigación de IBM [16] sobre el rendimiento

comparación de contenedores de VM y Linux. Sin embargo, ambos estudios están incompletos (por ejemplo, el primero no se refería a las características que no sonde CPU, y el segundo no terminó la evaluación de la red del contenedor). Más importante aún, nuestro trabajo niega la constatación del informe de IBM de que "los contenedores y las máquinas virtuales no imponen casi ninguna sobrecarga en el uso de CPU y memoria" y también dudas sobre "Docker iguala o supera el rendimiento de KVM en todos los casos". Además, en adición a la sobrecarga de rendimiento promedio de las tecnologías de virtualización, estamos más preocupados por suvariabilidad de rendimiento de sobrecarga i n.

Tenga en cuenta que, aunque también hay estudios de rendimiento

en la implementación de contenedores dentro de máquinas virtuales (por ejemplo, [17], [18]), tal estructura redundante podría no ser adecuada para una comparación "manzana a manzana" entre Docker containers y máquinas virtuales, y por lo tanto no incluimos este escenario de virtualización en este estudio.

3. Implementación de la evaluación del rendimiento

3.1. Metodología de evaluación del rendimiento

Dado que la comparación entre los gastos generales de rendimiento del contenedor y la máquina virtual se basa esencialmente en su evaluación del rendimiento, definimos nuestro trabajo como un estudio de evaluación perfor- mance que pertenece al campo de ECS [3]. Considerando que "la metodología de evaluación sustenta toda la innovación en la informática experimental" [19], empleamos la metodología DoKnowMe [4] para guiar las implementaciones de eval- uation en este estudio. DoKnowMe es una metodología abstracta de evaluation sobre la analogía de la "clase" en la programación orientada a objetos. Mediante la integración de artefactos de conocimiento específicos del dominio, DoKnowMe se puede personalizar en metodologías específicas (por analogía de "objeto") para facilitar la evaluación desistemas informáticos concretos diferentes. Para estructurar mejor nuestro informe, dividimos nuestra implementación de evaluación impulsada por DoKnowMe en actividades premesur experimentales (cf. Sección 3.2) y resultados y análisis experimentales (cf. Sección 3.3).

3.2. Actividadespre-E xperimentales

3.2.1. Reconocimiento de requisitos. Después de DoKnowMe, toda la implementación de la evaluación está impulsada esencialmente por los requisitos reconocidos. En general, el reconocimiento de requisitos consiste en definir un conjunto de requisitos específicos tanto para facilitar la comprensión del problema del mundo real como para ayudar a lograr declaraciones claras del propósito de evaluación correspondiente. En este caso, el requisito básico es dar un hijo de comparación cuantitativa fundamentalentre las so- luciones de virtualización basadas en hipervisores y basadas en contenedores. Puesto que concretamos estas dos solu- ciones de virtualización en máquinas virtuales de VMWare Workstation y contenedores Docker respectivamente, este requisito se puede especificar endospreguntas:

RQ1: ¿Cuánta sobrecarga de rendimiento introduce un contenedor Docker stan- dalone sobre su máquina física base?

TABLA 1. METRICS Y BENCHMARKS PARA ESTE EVALUATION STUDY

Capacidad de propiedad física Métrico Versión de referencia

Rendimiento de datos de comunicación Iperf 2.0.5

Cálculo (latencia) Puntuación HardInfo 0.5.1

Rendimiento de datos de memoria STREAM 5.10

Velocidad de transacción de almacenamiento Bonnie++ 1.97.1

Rendimiento de datos de almacenamiento Bonnie++ 1.97.1

RQ2: ¿Cuánta sobrecarga de rendimiento introduce una máquina virtual stan- dalone sobre su ma- chine físico base?

Teniendo en cuenta que las tecnologías de virtualización podrían dar lugar a una variación del rendimiento del servicio [20], también nos preocupa la sobrecarga de variabilidad potencial del contenedor y de la máquina virtual, además de su sobrecarga de rendimiento promedio:

RQ3: ¿Cuánta sobrecarga de variabilidad de rendimiento introduce un contenedor Docker independiente sobre su máquina física base durante un período de tiempo determinado?

RQ4: ¿Cuánta sobrecarga de variabilidad de rendimiento introduce una máquina virtual independiente sobre su máquina física base durante un período de tiempo determinado?

3.2.2. Identificación de la función de servicio. Recuerde que tratamos los contenedores de Docker como un tipo alternativo de servicio en la nube a las máquinas virtuales. Mediante el uso de la taxonomía de los servicios en la nube evalu- ation [21], examinamos la comunicación-, la computación-

, aspectos de QoS relacionados con la memoria y el almacenamiento; y luego nos centramos en las características de servicio, incluido el rendimiento de los datos de comunicación, la latencia de cálculo, el rendimiento de los datos de memoria y la velocidad de las transacciones de almacenamiento y el rendimiento de los datos.

3.2.3. Listado y selección de métricas/puntos de referencia. La selección de métricas de evaluación suele depender de la capacidad de disponibilidad de los puntos de referencia. De acuerdo con nuestra expe- rience anterior de la evaluación de servicios en la nube, elegimos puntos de referencia relativamente ligeros y populares para tratar de minimizar el sesgo de benchmarking potencial, como se indica en la Tabla 1. Por ejemplo, Iperf se ha identificado para poder ofrecer resultados más precisos mediante el consumo de menos recursos del sistema. De hecho, a excepción de STREAM que es el punto de referencia de evaluación de memoria de facto incluido en lasuite HPC Challenge Benchmark (HPCC), los otros puntos de referencia son todas las utilidades integradas de Ubuntu.

En particular, aunque Bonnie++ solo mide la cantidad de datos procesados por segundo, las acciones de E/S de disco se realizan byte a byte al acceder a un tamaño pequeño de datos. Por lo tanto, consideramos medir la velocidad de las transacciones de almacenamiento al operar datos de tamaño de byte y medir el rendimiento de los datos de almacenamiento al operar datos de tamaño de bloque. En cuanto a la computación de propiedades,teniendo en cuenta la diversidad en los trabajos de CPU (por ejemplo, cálculos enteros y de punto flotante), empleamos HardInfo que incluye seis micro-puntos de referencia para generar puntuaciones de rendimiento.

Cuando se trata de la sobrecarga de rendimiento, utilizamos la relación de sobrecarga1 del dominio de iness debus como analogía a su medición. En detalle, tratamos la pérdida de rendimiento en comparación con una línea de base como el gasto, al tiempo que imaginamos que el rendimiento de referencia es el ingreso global, tal como se define en la ecuación (1).

Pm Pb?

• Tamaño de la memoria: La máquina física está equipada con una SDRAM DDR2 de 3 GB. Al ejecutar VMWare Workstation Pro sin iniciar ninguna máquina virtual, "vea

-n 5 free -m" muestra un uso de memoria de 817MB mientras deja 2183MB libre en la máquina física. Por lo tanto, establecemos el tamaño de la memoria en 2 GB para que la máquina virtual evite (al menos para minimizar) la posible memoria

Op á ?

−

Pb á

100% (1)

Intercambio.

• Tamaño de almacenamiento: Hay 120 GB de disco duro en el

donde Op se refiere a la sobrecarga de rendimiento; Pm denota el resultado de benchmarking como una medida de una entidad de servicio; Pb indica el rendimiento de línea base de la entidad de servicio; y luego . Pm á Pb? representa la pérdida de rendimiento correspondiente. Tenga en cuenta que la perfor- mance de la máquina física se utiliza como línea de base en nuestro estudio. Además, con- sidering posibles errores observacionales, permitimos un margen de error para el nivel de confianza hasta 99% con respecto a los resultados de benchmarking. En otras palabras, ignoraremos la diferencia entre el rendimiento medido y su línea de base si la sobrecarga de rendimiento calculada es menor que

1% (es decir, si Op < 1%, a continuación, Pm á Pb).

3.2.4. Listado y selección de factores experimentales. La identificación de factores experimentales desempeña un papel previo en el siguiente diseño experimental. Más impor- tantly, especificando los factores pertinentes sería necesario paramejorar la repetibilidad de los implemen- tations experimentales. Al hacer referencia al marco de factores experimentales de evaluación de servicios en la nube [22], elegimos los factores relacionados con los recursos y la carga de trabajo de la siguiente manera. En particular, con la posible compensación de sobrecarga de rendimiento de los potentes recursos informáticos, tratamos de enfatizar la condición ex perimentaria empleando un banco de pruebas "suficientemente suficiente".

Los factores relacionados con los recursos:

• Tipo de recurso: Dada la evaluación requierement, esencialmente hemos considerado tres tipos de recursos para soportar el servicio imaginario en la nube, a saber, máquina física, contenedor y máquina virtual.

• Alcance de comunicación: Probamos la comunicación entre nuestra máquina local y una instancia t2.micro de Amazon EC2. La máquina local se encuentra en nuestro laboratorio de banda ancha en la Universidad de Lund, y la instancia EC2 es de la zona disponible de Amazon ap-southeast-1a dentro de la región Asia Pacífico (Singapur).).

• índice Ethernet de comunicación: Nuestro lado local utiliza una conexión Gigabit a Internet, mientras que la instancia EC2 en el lado remoto tiene el rendimiento de red "Bajo a Moderado" definido por Amazon.

• ÍNDICE de CPU: Recuerde que hemos empleado "suficiente" recurso informático. El modelo de CPU de la máquina física se elige para ser Intel CoreTM2 Duo Pro- cessor T7500. El procesador tiene dos núcleos con el

Arquitectura de 64 bits, y su frecuencia base es de 2,2 GHz. Asignamos ambos núcleos de CPU a la máquina virtual independiente en la máquina física.

1[http://www.](http://www.investopedia.com/terms/o/overhead-ratio.asp) [investopedia.](http://www.investopedia.com/terms/o/overhead-ratio.asp) [com/terms/o/overhead-ratio.](http://www.investopedia.com/terms/o/overhead-ratio.asp) [asp](http://www.investopedia.com/terms/o/overhead-ratio.asp)

máquina física. Teniendo en cuenta el uso de espacio por parte del sistema operativo host, asignamos 100 GB a la máquina virtual.

• Sistema operativo: Dado que Docker requiere una in-stallation de 64 bits y los kernels de Linux anteriores a 3.10 no admiten todas lasuras de hazaña para ejecutarcontenedores Docker, elegimos el último Ubuntu 15.10 de 64 bits como el sistema de operación tanto para la máquina física como para la máquina virtual. Además, de acuerdo con las discusiones sobre imágenes base en la comunidad de Docker [23], [24], establecemos inten- cionalmente una imagen base del sistema operativo (especificando FROM ubuntu:15.10 en el Dockerfile) para todos los contenedores Docker en nuestros experimentos. Tenga en cuenta que la imagen base del sistema operativo de un contenedor es solo una representación del sistema de archivos, aunque no actúa como unsistema operativo guest.

Los factores relacionados con la carga de trabajo:

• Duración: Para cada experimento de evaluación, decidimos tomar una observación de todo el día más un calentamiento de una hora (es decir, 25 horas).

• Tamaño de la carga de trabajo: Las cargas de trabajo experimentales están predefinidas por los puntos de referencia seleccionados. Por ejemplo, la CPU de micromafiesta Fibonacci genera carga de trabajo calculando el número 42 de Fibonacci. En parte, el punto de referencia Bonnie++ distingue entre datos de tamaño de byte de lectura/escritura y datos de tamaño de bloque.

3.2.5. Diseño experimental. Está claro que los factores identificados están todos con un valor único, excepto el tipo de recurso. Por lo tanto, un diseño sencillo es ejecutar los puntos de referencia indi- viduales en cada uno de los tres tipos de recursos iindependientemente durante todo un día más una hora.

Además, siguiendo el modelo conceptual de IaaS

evaluación del rendimiento [25], registramos el diseño experimental en un proyecto tanto para facilitar nuestras implementaciones experimentales como para queotros evaluadores repliquen/repitan nuestro estudio. Debido al límite de espacio, compartimos el plano experimental en línea como un documento complementario. 2

3.3. Resultados y análisis experimentales

3.3.1. Resultado y Análisis de evaluación de la comunicación. Para lap urpose de comparación "manzana a manzana", forzamos tanto el contenedor como la máquina virtual a emplear la traducción de direcciones de red (NAT) para establecer conexiones salientes. Puesto que requieren enlace/reenvío de puertos para aceptar conexiones entrantes, solo probamos la comunicación saliente por formancia para reducir la posibilidad de ruido de configuración,

2[El blueprint experimental se comparte en línea en https:](https://drive.google.com/file/d/0B9KzcoAAmi43WTFuTXBsZ0NRd1U/)

[unidad.](https://drive.google.com/file/d/0B9KzcoAAmi43WTFuTXBsZ0NRd1U/) [google.](https://drive.google.com/file/d/0B9KzcoAAmi43WTFuTXBsZ0NRd1U/) [com/file/d/0B9KzcoAAmi43WTFuTXBsZ0NRd1U/](https://drive.google.com/file/d/0B9KzcoAAmi43WTFuTXBsZ0NRd1U/)

Máquina virtual de contenedor

Variabilidad Overhead 54.27986481 132.27808

TpApBpLE2. COMMUNICATION BENCHMARKING RE0SULTS USING IPERF0 ttt 0 0



DatosReTshoruorucgehTpyupteOverheadPromedio2e.002357S6ta7n3dard D5e5vi. a8t1io4n57202

Máquina física 29.066 Mbits/seg 1.282 Mbits/seg

Contenedor 28.484 Mbits/seg 1.978 Mbits/seg

Máquina virtual de contenedor de máquina física

CPU Blowfish

1

0.98

0.96

Máquina virtual 12.843 Mbits/sec 2.979 Mbits/seg

Sobrecarga de rendimiento de datos de sobrecarga de variabilidad

FPU Raytracing

0.94

0.92

0.9

CPU CryptoHash

140

120

Sobrecarga de variabilidad (%)

100

80

60

40

20

0

60

50

40

30

20

10

0

Sobrecarga de aire de datos (%)

Máquina virtual de contenedor

FPU FFT

0.88

CPU N-Queens

CPU Fibonacci

Figura 2. Rendimiento de datos de comunicación y su sobrecarga de variabilidad de un contenedor Docker independiente frente a una máquina virtual (mediante la referencia Iperf).

estableciendo la instancia EC2 remota en el servidor Iperf y utilizando la máquina local, el contenedor y la máquina virtual como cliente Iperfs.

Los resultados de evaluación comparativa de la repetición de iperf -c XXX.XXX.XXX.XXX -t 15 (con un intervalo de un minuto entre cada dos ensayos consecutivos) se enumeran en el Cuadro 2. El XXX.XXX.XXX.XXX denota la dirección IP externa del instan ce EC2utilizado en nuestros experimentos. Tenga en cuenta que, a diferencia de las otras características de rendimiento, el rendimiento de los datos de comunicación ofrece fluctuaciones periódicas y significativas, que podrían ser el resultado de la competencia de recursos de red tanto en nuestro lado local como en el lado EC2 durante las horas de trabajo. Por lo tanto, nos centramos particularmente en el período más largo de datos relativamente estables de la observación de todo el día, y por lo tanto los resultados aquí son sólo para referencia aproximada.

Dado el costo adicional de usar la red NAT para enviar y recibir paquetes, habría perfor-

Figura 3. Resultados de benchmarking de cálculo mediante HardInfo.

Contenedor (sobrecarga de variabilidad) Máquina virtual (sobrecarga de variabilidad) Container (sobrecarga de latencia (puntuación)) Máquina virtual (latencia (puntuación) sobrecarga)

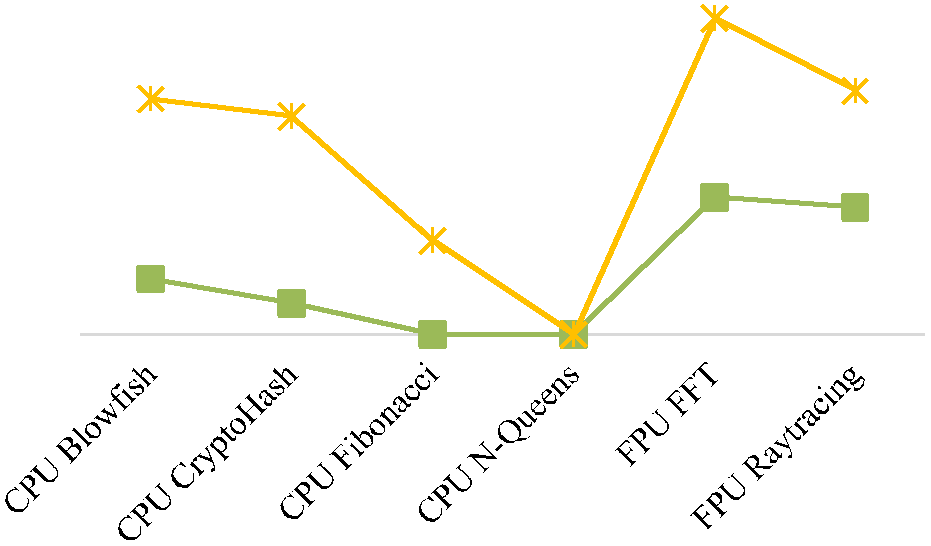
2500 10

Sobrecarga de lada- (puntuación) (%)

2000 8

1500 6

1000 4



500 2

Sobrecarga de variabilidad (%)

0 0

Figura 4. Latencia de cálculo (puntuación) y su sobrecarga de variabilidad de un contenedor Docker independiente frente a una máquina virtual (mediante el kit de herramientas HardInfo).

Sanciones para el contenedor y la máquina virtual. Usando

Ecuación (1), calculamos su rendimiento de comunicación

Benchmarking

Hbi =

Yo

1,2,...,n

(2)

los gastos generales, como se ilustra en la Figura 2.

Una tendencia clara es que, en comparación con la máquina virtual, el contenedor

max(Benchmarking )

1

pierde menos rendimiento de comunicación, con solo un 2% de sobrecarga de rendimiento de datos y una sobrecarga de variabilidad de alrededor del 54%. Sin embargo, es sorprendente ver más del 55% de los datos

LBi - Benchmarkingi

1

max( )

Benchmarking1,2,...,n

(3)

sobrecarga de rendimiento para la máquina virtual. Aunque hemos comprobado dos veces los parámetros de configuración relevantes y hemos rehecho varias rondas de experimentos para confirmar este fenómeno, todavía dudamos de la razón relacionada con el hipervisor detrás de una pérdida por formancia tangrande. Destacamos especialmente esta observación para inspirar más investigaciones.

3.3.2. Resultado y Análisis de evaluación de cálculo. Recuerde que los seis micro benchmarks de HardInfo ofrecen puntuaciones de CPU "más altas y mejores" y "más bajas". Para realizar un análisis experimental, utilizamos las dos ecuaciones de baja para estandarizar los resultados de benchmarking "superiores", mejores y "mejores", respectivamente.

donde HBi además punt obtiene una puntuación del tipo de recurso de servicio i mediante la estandarización del resultado de benchmarking "superior-mejor" Benchmarkingi; y de manera similar, LBi representa la puntuación de CPU "inferior-mejor" del servicio re- source type i. Tenga en cuenta que la ecuación (3) ofrece esencialmente los resultados de benchmarking "más bajos-mejor" una representación "superior-mejor" a través de la estandarización recíproca.

Por lo tanto, podemos utilizar una gráfica de radar para ayudar a contrastar intuitivamente el rendimiento de los tres tipos de recursos, como se muestra en la Figura 3. Por ejemplo, los diferentes tamaños de polígono indican claramente que el contenedor generalmente se calcula más rápido que la máquina virtual, aunque las diferencias de rendimiento se basan en caso por caso con respecto a diferentes tipos de trabajo de CPU.

3500

Cuenta de datos de memoria (MB/s)

3000

2500

2000

1500

1000

500

0

Máquina virtual de contenedor de máquina física

Copiar escala Añadir tríada

Operación de memoria por STREAM

1500

Sobrecarga de variabilidad (%)

1250

1000

750

500

250

Contenedor (sobrecarga de variabilidad) Máquina virtual (sobrecarga de variabilidad) Contenedor (sobrecarga de rendimiento de datos) Máquina virtual (sobrecarga de rendimiento de datos)

6

Tipo de aire de datos (%)

5

4

3

2

1

Figura 5. Resultados de la prueba comparativa de memoria mediante STREAM. Las barras de error indican las desviaciones estándar de los datos de memoria correspondientes a través de- put.



Sin embargo, nuestros resultados experimentales no muestran ninguna tendencia general en la variabilidad de laspuntuaciones de cálculo de esos res ources. Como se puede ver en las sobrecargas de rendimiento calculadas (cf. Figura 4), la máquina virtual ni siquiera muestra una variabilidad peor que la máquina física al ejecutar CPU CryptoHash, CPU N-Queens y FPU Raytracing. Por el contrario, hay una sobrecarga de variabilidad de casi el 2500% para la máquina virtual al calcular el número 42 de Fibonacci. En particular, las tecnologías de virtualización parecen ser sensibles a los trabajos de transformación de Fourier (el FFT FPU de referencia), ya que la sobrecarga de latencia de cálculo y la sobrecarga de variabilidad son relativamente altas tanto para el contenedor como para la máquina virtual.

3.3.3. Resultado y análisis de la evaluación de la memoria. STREAM mide el rendimiento sostenible de los datos de memoria mediante la realización de cuatro operaciones vectoriales típicas, a saber, Copiar, Escalar, Agregar y Tríada. Visualizamos directamente los resultados de marcado de banco en la Figura 5 para facilitar nuestra observación. Como primera impresión, parece que la máquina virtual tiene un rendimiento de datos de memoria un poco más pobre y hay poca diferencia entre la máquina física y el contenedor de Docker en el contexto de la ejecución de STREAM.

Al calcular la sobrecarga de rendimiento en términos de rendimiento de datos de memoria y su variabilidad, podemos ver la diferencia significativa entre estos tres tipos de recursos, como se muestra en la figura 6. Tome la operación Triad como ejemplo, aunque el contenedor porformularios, así como la máquina física en promedio, la sobrecarga de variabilidad del contenedor es más del 500%; de forma similar, aunque la sobrecarga de rendimiento de datos de la Tríada de la máquina virtual es de alrededor del 4% solamente, su sobrecarga de variabilidad es de casi 1400%. En otras palabras, la pérdida de rendimiento de memoria en la que incurren ambas técnicas de virtualización se encarna principalmente con el aumento de la variabilidad del rendimiento.

Además, también vale la pena destacar que el rendimiento promedio de los datos de copia del con- tainer es incluso ligeramente superior al de la máquina física (es decir. 2914.023MB/s frente a 2902.685MB/s) en nuestros experimentos. Recuerde que hemos considerado un margen de error del 1%. Puesto que esos dos valores están cerca uno del otro dentro de este margen de error, aquí ignoramos un fenómeno tan irregular como un error de observación.

0 0

Copiar escala Añadir tríada

Figura 6. Rendimiento de datos de memoria y su sobrecarga de variabilidad de un contenedor de Docker stan- dalone frente a una máquina virtual (mediante el punto de referencia STREAM).

3.3.4. Resultado y análisis de la evaluación de almacenamiento. Para la prueba de lectura y escritura de disco, Bonnie++ crea un conjunto de datos dos veces el tamaño de la memoria RAM implicada. Dado que la máquina virtual se asigna 2 GB de RAM, también restringimos el uso de memoria a 2 GB para Bonnie++ tanto en la máquina física como en el contenedor, ejecutando "sudo bonnie++ -r 2048

-n 128 -d / -u root". En consecuencia, las pruebas de marcado de banco se llevan a cabo con 4 GB de datos aleatorios en el disco. Cuando Bonnie++ se está ejecutando, lleva a cabo varias operaciones de almacenamiento que van desde la lectura/escritura de datos hasta la creación/eliminación de archivos. Aquí solo nos centramos en el rendimiento de los datos de tamaño de byte y bloque de lectura/escritura.

Para ayudar a resaltar varias observaciones diferentes, trazamos la trayectoria de los resultados experimentales a lo largo de la secuencia de prueba durante todo el día, como se muestra en la Figura 7. La primera observación sorprendente es que, los tres tipos de recursos tienen patrones regulares de fluctuación del rendimiento en la escritura en bloque, reescritura y lectura. Debido al límite de espacio, no informamos de su rendimiento de reescritura de bloques en este artículo. Al explorar la información de hardware, identificamos el modelo de unidad de disco duro (HDD) como ATA Hitachi HTS54161, y suespecificación describe "Almacena 512 bytes por sector y utiliza cuatro cabezales de datos para leer los datos de dos bandejas, girando a 5.400 revoluciones por minuto". Como sabemos, la superficie del disco duro se divide en un conjunto de tr acks concéntricamentecirculares. Dada la misma velocidad de rotación de un disco duro, las pistas externas tendrían un rendimiento de datos más alto que las internas. Como tal, esos patrones regulares podrían indicar que los cabezales de disco duro se transportan secuencialmente entre pistas externas e internas cuandoredactan/leen datos de bloques de forma ecutiva durante los experimentos.

La segunda observación sorprendente es que, a diferencia de la mayoría de los casos en los que la máquina virtual tiene el peor rendimiento, el contenedor parece significativamente pobre en el accesoal tamaño de bytes de los datos, aunque su variabilidad de rendimiento es claramente la más pequeña. Calculamos además la sobrecarga de rendimiento de almacenamiento para ofrecer una comparación más específica entre el contenedor y la máquina virtual y dibujar los resultados en la Figura 8. Tenga en cuenta que, en el caso de que la variabilidad del contenedor/VM sea menor que la de la máquina física, establecemos directamente la sobrecarga de variabilidad correspondiente en cero en lugar de

0.6

0.4

**MB/s**

0.2

0

1 150

**Número** secuencial **de**  **ensayos**

50

40

30

**MB/s**

**MB/s**

20

10

0

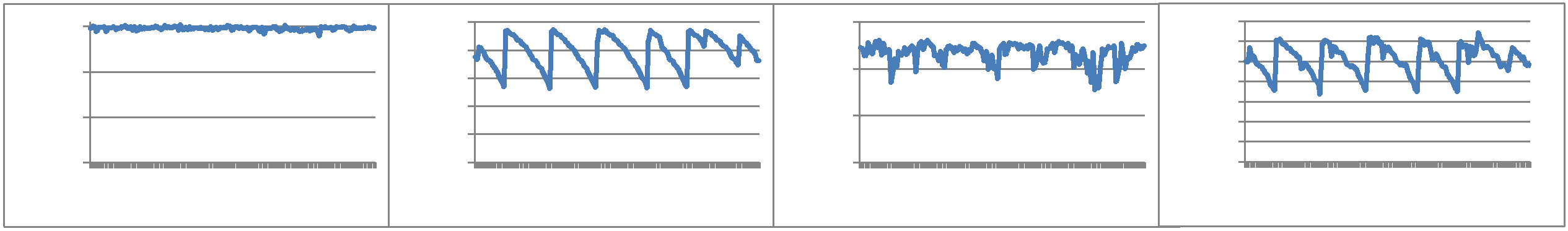
1 150

**Número** secuencial **de**  **ensayos**

3

2

1



0

1 150

**Número** secuencial **de**  **ensayos**

70

**MB/s**

60

50

40

30

20

10

0

1 150

**Número** secuencial **de**  **ensayos**

(a) La máquina física escribe bytes. (b) Máquina física escribe bloques. (c) Máquina física lee bytes. (d) La máquina física lee bloques.

0.6

0.4

**MB/s**

0.2

0

1 147

**Número** secuencial **de**  **ensayos**

50

40

30

**MB/s**

20

10

0

1 147

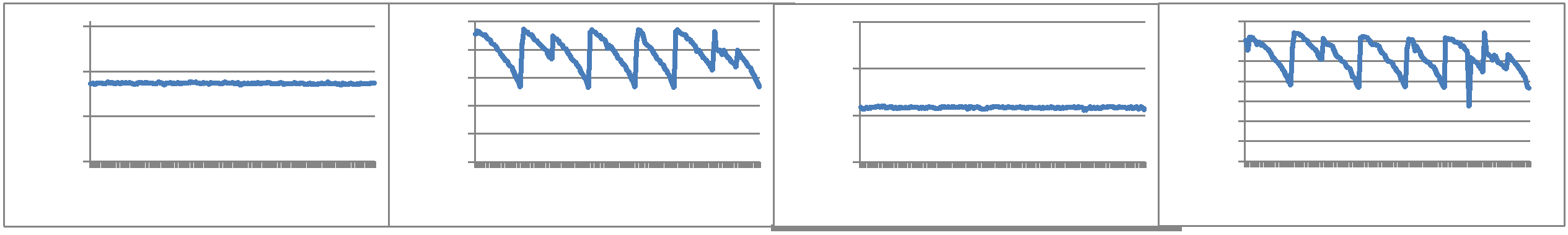
**Número** secuencial **de**  **ensayos**

3

2

**MB/s**

1



0

1 147

**Número** secuencial **de**  **ensayos**

70

60

50

**MB/s**

40

30

20

10

0

1 147

**Número** secuencial **de**  **ensayos**

(e) El contenedor escribe bytes.

(f) Contenedor escribe bloques.

(g) El contenedor lee bytes.

(h) El contenedor lee bloques.

0.6

0.4

**MB/s**

0.2

0

1 10

**Número** secuencial **de**  **ensayos**

50

40

30

**MB/s**

20

1

10

0

1 10

**Número** secuencial **de**  **ensayos**

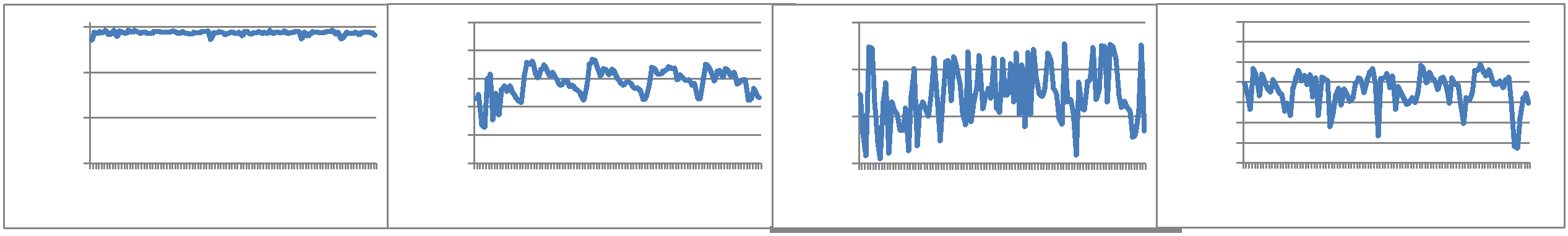
3

2

**MB/s**

1

1



0

1 10

**Número** secuencial **de**  **ensayos**

70

60

50

**MB/s**

40

30

20

1

10

0

1 101

**Número** secuencial **de**  **ensayos**

(i) La máquina virtual escribe bytes.

(j) La máquina virtual escribe bloques.

(k) La máquina virtual lee bytes.

(l) La máquina virtual lee los bloques.

Figura 7. Los resultados de la evaluación comparativa de almacenamiento mediante Bonnie++ durante 24 horas. La escala máxima del eje X indica el número de iteración de Bonnie++

prueba (es decir, la máquina física, el contenedor y la máquina virtual ejecutan 150, 147 y 101 pruebas respectivamente).

200

150

Sobrecarga de variabilidad (%)

100

50

0

Contenedor (sobrecarga de variabilidad) Máquina virtual (sobrecarga de variabilidad) Contenedor (sobrecarga de rendimiento de datos) Máquina virtual (sobrecarga de rendimiento de datos)

60

Tipo de aire de datos (%)

45

30

15

0

La máquina virtual tiene aproximadamente un 30 % de sobrecarga de rendimiento de datos en otros escenarios de E/S de disco, mientras que el contenedor apenas incurre en sobrecarga al leer o escribir un gran tamaño de datos.

Nuestra tercera observación es que, la sobrecarga de rendimiento de almacenamiento de diferentes tecnologías de virtualización también se puede reflejar a través del número total de pruebas iterativas de Bonnie++. Como apunta la escala máxima del eje X en la Figura

7, la máquina física, el contenedor y la máquina virtual pueden finalizar respectivamente 150, 147 y 101 rondas de pruebas de disco durante 24 horas. Dada esta información, estimamos que la sobrecarga de rendimiento de almacenamiento del contenedor y de la máquina virtual es del 2 % (147-150 a 150) y del 32,67 % (101-150-/150)

Datos de bytes

Escritura

Bloquear datos

Escritura



Datos de bytes

Lectura

Bloquear datos

Lectura

Respectivamente.

Figura 8. Rendimiento de datos de almacenamiento y su sobrecarga de variabilidad de un contenedor Docker stan- dalone frente a una máquina virtual(mediante el punto de referencia Bonnie++).

permitiendo que cualquier sobrecarga de rendimiento sea negativa. A continuación, las barras del gráfico indican que las sobrecargas de variabilidad de almacenamiento de ambas tecnologías de virtualización son casi insignificantes, excepto para los datos de tamaño de byte g de lecturaen la máquina virtual (hasta casi el 200%). Aunque el controlador de almacenamiento es un cuello de botella conocido para la E/S de disco interno de un contenedor, sigue siendo sorprendente que el contenedor aporte entre un 40% y un 50% de sobrecarga de rendimiento de datos al realizar operaciones de disco de perforbytes byte a byte. En otras palabras, la solución contenedora podría no ser adecuada para las aplicaciones en la nube que tienen una generación y un consumo de datos intermedios frecuentes e impredecibles. Por el contrario, lapérdida de rendimientorelativamente trivial en la escritura de datos de bytes de la máquina virtual. Sin embargo, el

4. Conclusión

Siguiendo la metodología de evaluación del desempeño Do- KnowMe, sacamos conclusiones principalmente respondiendo a las preguntas predefinidas de los requisitos. Impulsado por RQ1 y RQ2, nuestro resultado de evaluación confirma en gran medida las discusiones cualitativas antes mencionadas: El rendimiento promedio del contenedor es generalmente mejor que el de la máquina virtual e incluso es comparable al de la máquina física con respecto a muchas características. Específicamente, el contenedor tiene menos del 4% de sobrecarga de perfor- mance en términos de rendimiento de datos de comunicación, cálculo tardíoy, rendimiento de datos de memoria y rendimiento de datos de almacenamiento. Sin embargo, la virtualización basada en contenedores podría alcanzar un cuello de botella de velocidad de transacción de almacenamiento, con una sobrecarga de hasta el 50%. Tenga en cuenta que, como se mencionó anteriormente, interpretamos el rendimiento de los datos de tamaño de byte en

velocidad de transacción de almacenamiento, porque cada byte esencialmente llama a una transacción de disco aquí. Por el contrario, aunque la máquina virtual ofrece el peor rendimiento en la mayoría de los casos, podría funcionar tan bien como la máquina física al resolver el problema de N-Queens o escribir datos de tamaño pequeño en el disco.

Impulsado por RQ3 y RQ4, encontramos que la pérdida de rendimiento resultante de lasvi tualizaciones es más visible en la variabilidad del rendimiento. Por ejemplo, la sobrecarga de capacidad variable del contenedor podría alcanzar hasta más del 500% con respecto al cálculo de Fibonacci y la operación de la tríada de memoria. Similar,aunque el contenedor generalmente muestra menos variabilidad de rendimiento que la máquina virtual, todavía hay casos excepcionales: el contenedor tiene la mayor variación de rendimiento en el trabajo de calcular la transformación de Fourier, mientras que incluso la variabilidad de rendimiento de la máquina virtual no es peor que la de la máquina física al ejecutar trabajos CryptoHash, N-Queens y Raytracing.

En general, nuestro trabajo revela que el rendimiento sobre-jefes de estas dos tecnologías de virtualización podría variar no sólo en una característica-por característica, sino también sobre una base de trabajo a trabajo. Aunque la solución basada en contenedores es sin duda ligera, la tecnología basada en hipervisor no viene con una mayor sobrecarga de rendimiento en todos los casos. Basándonos en unestudio de valoración electrónica tanfundamental, aplicaremos gradualmente contenedores Docker a diferentes aplicaciones del mundo real en el futuro venidrán. Las prácticas orientadas a aplicaciones también se replicarán en el entorno virtual basado en hipervisor para estudios de casos de comparación adicionales.

Reconocimiento

Este trabajo cuenta con el apoyo del Consejo Sueco de Investigación (VR) para el proyecto "Cloud Control", y a través de los Centros de Excelencia LCCC Linnaeus y ELLIIT.

Referencias

[1] C. Pahl, "Containerization and the PaaS Cloud", IEEE Cloud Com- put., vol. 2, no 3, págs. 24–31, mayo/junio de 2014.

[2] J. P. Walters, V. Chaudhary, M. Cha, S. G. Jr., y S. Gallo, "Una comparación de tecnologías de virtualización para HPC", en Proc. 22o Int. Conf. Adv. Inf. Networking Appl. (AINA 2008). Okinawa, Japón: IEEE Computer Society, 25-28 de marzo de 2008, págs. 861–868.

[3] D. G. Feitelson, "Experimental computer science", Commun. ACM, vol. 50, No. 11, págs. 24–26, noviembre de 2007.

[4] Z. Li, L. O'Brien y M. Kihl, "DoKnowMe: Towards a domain knowledge-driven methodology for performance evaluation", ACM SIGMETRICS Perform. Eval. Reverendo, vol. 43, no. 4, pp. 23–32, Mar.

2016.

[5] D. Merkel, "Docker: contenedores Linux ligeros para un desarrollo e implementación coherentes", Linux J., vol. 239, pp. 76–91, Mar.

2014.

[6] X. Xu, H. Yu y X. Pei, "Un enfoque novedoso de programación de recursos en nubes basadasen contenedores," en Proc. 17th IEEE Int. Conf. Sci. óg. (CSE 2014). Chengdu, China: IEEE Computer Society, 19-21

Diciembre de 2014, págs. 257–264.

[7] D. Bernstein, "Containers and Cloud: From LXC to Docker to Ku- Bernetes", IEEE Cloud Comput., vol. 1, no. 3, págs. 81–84, septiembre de 2014.

[8] M. G. Xavier, M. Ⅴ. Neves, y C. Un. F. D. Rose, "Una comparación de rendimiento de los sistemas de virtualización basados en contenedores para clústeres MapRe- duce", en Proc. 22o Euromicro Int. Conf. Distrib paralelo.

Proceso basado en red. (PDP 2014). Turín, Italia: IEEE Press, 12-

14 de febrero de 2014, págs. 299–306.

[9] C. Anderson, "Docker", IEEE Software, vol. 32, No. 3, págs. 102–105, mayo/junio de 2015.

[10] T. Banerjee, "Entender las principales diferencias seanentre LXC y

[Docker", https://www.](https://www.flockport.com/lxc-vs-docker/) [flockport.](https://www.flockport.com/lxc-vs-docker/) [com/lxc-vs-docker/, agosto de 2014.](https://www.flockport.com/lxc-vs-docker/)

[11] J. Che, C. Shi, Y. Yu, y W. Lin, "Una evaluación de rendimiento sintético de OpenVZ, Xen y KVM", en Proc. 2010 IEEE Asia- Pacific Serv. Comput. (APSCC 2010). Hangzhou, China: IEEE Computer Society, 6-10 de diciembre de 2010, págs. 587–594.

[12] D. Strauss, "Los contenedores - no las máquinas virtuales - son la nube futura,"

Linux J., vol. 228, págs. 118–123, abril de 2013.

[13] T. Adufu, J. Choi, y Y. Kim, "¿Es la tecnología basada en contenedores un ganador para aplicaciones científicas de alto rendimiento?" en Proc. 17a Oper de Red Asia-Pacífico. Administrar. (APNOMS 2015). Busan, Corea: IEEE Press, 19-21 de agosto de 2015, págs. 507–510.

[14] A. M. Joy, "Comparación de rendimiento entre contenedores Linux y máquinas virtuales", en Proc. 2015 Int. Conf. Adv. Comput. ICACEA 2015. Ghaziabad, India: IEEE Press, 14-15 Feb. 2015, págs. 507–510.

[15] K.-T. Seo, H.-S. Hwang, yo.-Y. Luna, O.-Y. Kwon, y B.-J. Kim, "Análisis de comparación de rendimiento del contenedor Linux y la máquina virtual para la creación de la nube", Adv. Sci. Technol. Lett., vol. 66, pág. pp.

105–111, diciembre de 2014.

[16] W. Felter, A. Ferreira, R. Rajamony, y J. Rubio, "An updated performance comparison of virtual machines and Linux containers", en Proc. 2015 IEEE Int. Software (ISPASS

2015). Filadelfia, PA, Estados Unidos: IEEE Press, 29-31 Mar. 2015, pp.

171–172.

[17] R. Dua, A. R. Raja, y D. Kakadia, "Virtualization vs container- ization to supportPaaS", en Proc. 2014 IEEE Int. Cloud Eng. (IC2E 2015). Boston, Massachusetts, EE. UU.: IEEE Computer Society,

10-14 de marzo de 2014, págs. 610–614.

[18] S. F. Piraghaj, A. Ⅴ. Dastjerdi, R. N. Calheiros, y R. Buyya, "Efficient virtual machine sizing for hosting containers as a service", en Proc. 11o Congreso Mundial. Serv. (SERVICIOS 2015). Nueva York, Ee.UU.: IEEE Computer Society, 27 jun.-2 jul. 2015, págs. 31–38.

[19] S. M. Blackburn, K. S. McKinley, R. Garner, C. Hoffmann, A. M.

Khan, R. Bentzur, A. Diwan, D. Feinberg, D. Frampton, S. Z. G. M. H. A. H. M. J. H. Lee, J. E. B. Moss, A. Phansalkar, D. Stefanovik, T. VanDrunen, D. von Dincklage, y B. Wiedermann, "Despierta y huele el café: Metodología de evaluación para el siglo XXI", Commun. ACM, vol. 51, No. 8, págs. 83–89, agosto de 2008.

[20] A. Iosup, N. Yigitbasi, y D. Epema, "Sobre la variabilidad del rendimiento de los servicios en la nube deproducción", en Proc. 11o IEEE/ACM Int. Symp. Cluster Cloud Grid Comput. (CCGrid 2011). Newport Beach, CA, EE. UU.: IEEE Computer Society, 23-26 de mayo de 2011, págs. 104–113.

[21] Z. Li, L. O'Brien, R. Cai, y H. Zhang, "Hacia una taxonomía de evaluación performance de servicios comerciales en la nube", en Proc. 5th Int. (IEEE CLOUD 2012). Honolulu, Hawái, Ee. UU.: IEEE Computer Society, 24-29 de junio de 2012, págs. 344–351.

[22] Z. Li, L. O'Brien, H. Zhang, y R. Cai, "Un factor framework para el diseño experimental para la evaluación del rendimiento de los servicios comerciales en la nube", en Proc. 4th Int. (CloudCom 2012). Taipei, Taiwán: IEEE Computer Society, 3-6

Diciembre de 2012, págs. 169–176.

[23] StackOverflow, "¿Cuál es la relación entre el sistema operativo host docker y el sistema operativo de la imagen de base [del contenedor?"](http://stackoverflow.com/questions/18786209/what-is-the-relationship-between-the-docker-host-os-and-the-container-base-image)   [http:](http://stackoverflow.com/questions/18786209/what-is-the-relationship-between-the-docker-host-os-and-the-container-base-image)

[stackoverflow.](http://stackoverflow.com/questions/18786209/what-is-the-relationship-between-the-docker-host-os-and-the-container-base-image) [com/questions/18786209/what-is-the-relationship-](http://stackoverflow.com/questions/18786209/what-is-the-relationship-between-the-docker-host-os-and-the-container-base-image) between-the-docker-host-os-and-the-container-base-image, Sept.

2013.

[24] Reddit, "¿Necesito usar una imagen base del sistema operativo en mi Dockerfile o será [el sistema operativo](https://www.reddit.com/r/docker/comments/2teskf/do_i_need_to_use_an_os_base_image_in_my/)  [host?" https://www.](https://www.reddit.com/r/docker/comments/2teskf/do_i_need_to_use_an_os_base_image_in_my/) [reddit.](https://www.reddit.com/r/docker/comments/2teskf/do_i_need_to_use_an_os_base_image_in_my/) [com/r/docker/comments/](https://www.reddit.com/r/docker/comments/2teskf/do_i_need_to_use_an_os_base_image_in_my/)

2teskf/do i necesita usar una imagen base de os en mi/, enero de 2015.

[25] Z. Li, L. O'Brien, H. Zhang y R. Cai, "Sobre la conceptualización de la evaluación del rendimiento de los servicios de IaaS", IEEE Trans. Serv. Comput., vol. 7, no 4, págs. 628–641, octubre-diciembre. 2014.