Compartición de recursos entre MVs

v.20161220

Arquitecturas Virtuales

Depto. de Arquitectura de Computadores
Universidad de Málaga

Pregunta:

- “Y eso de la virtualización.. ¿funciona de verdad?”
  - Respuesta A: Los bancos ya están virtualizando sus infraestructuras.
    - “Cuando un banco se atreve a usar algo es que la tecnología ya está madura (o incluso obsoleta).”
  - Respuesta B (la correcta): “DEPENDE DE PARA QUÉ Y DE CUÁNTO DINERO TIENES”

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Objetivos

- Tomar conciencia de los aspectos **internos** de la virtualización y de cómo reducen o mejoran el rendimiento del hardware virtual (respecto al real).
- Saber cómo afecta al rendimiento la **coexistencia** de máquinas virtuales y cómo medir los recursos reales que necesita una MV.
- Conocer los mecanismos de control de **asignación** de recursos

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

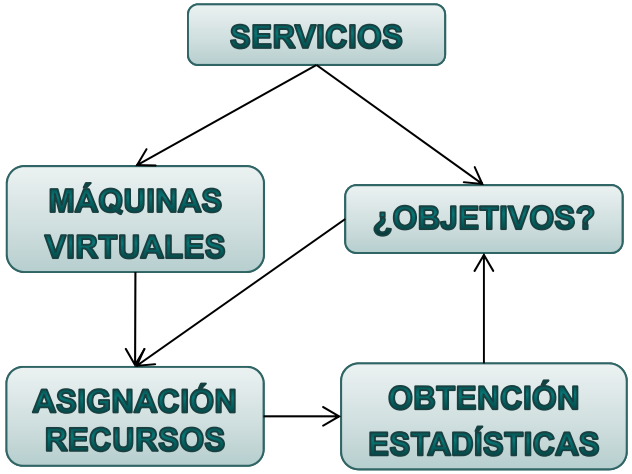
Indice

- **Sobrecostes** inherentes a la virtualización
- Influencia en el rendimiento de la **compartición** de recursos entre máquinas virtuales
- Paravirtualización y **optimización** del S.O. invitado

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Indice

- Obtención de estadísticas
- Asignación de recursos
  - Estática
  - Dinámica

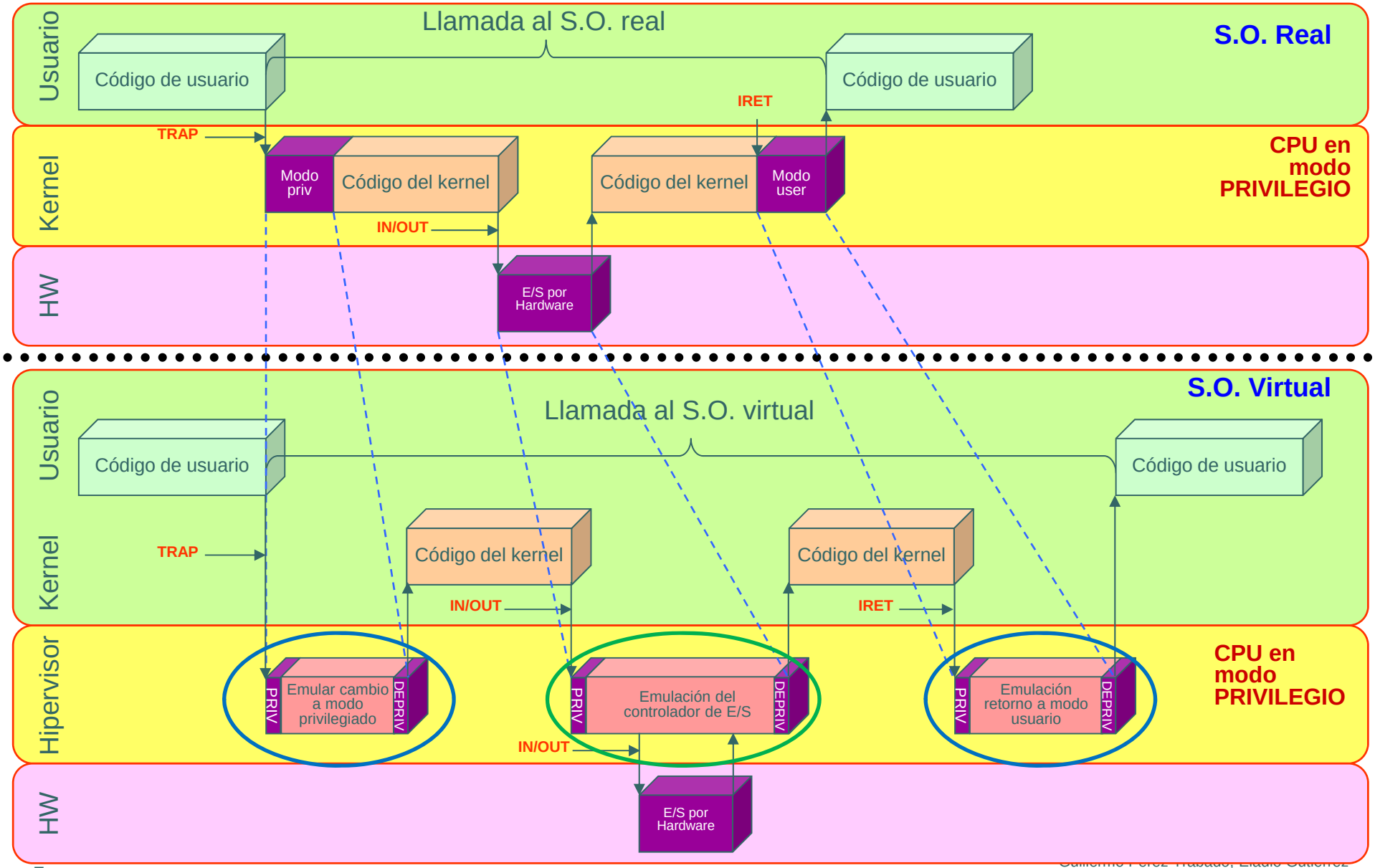


Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

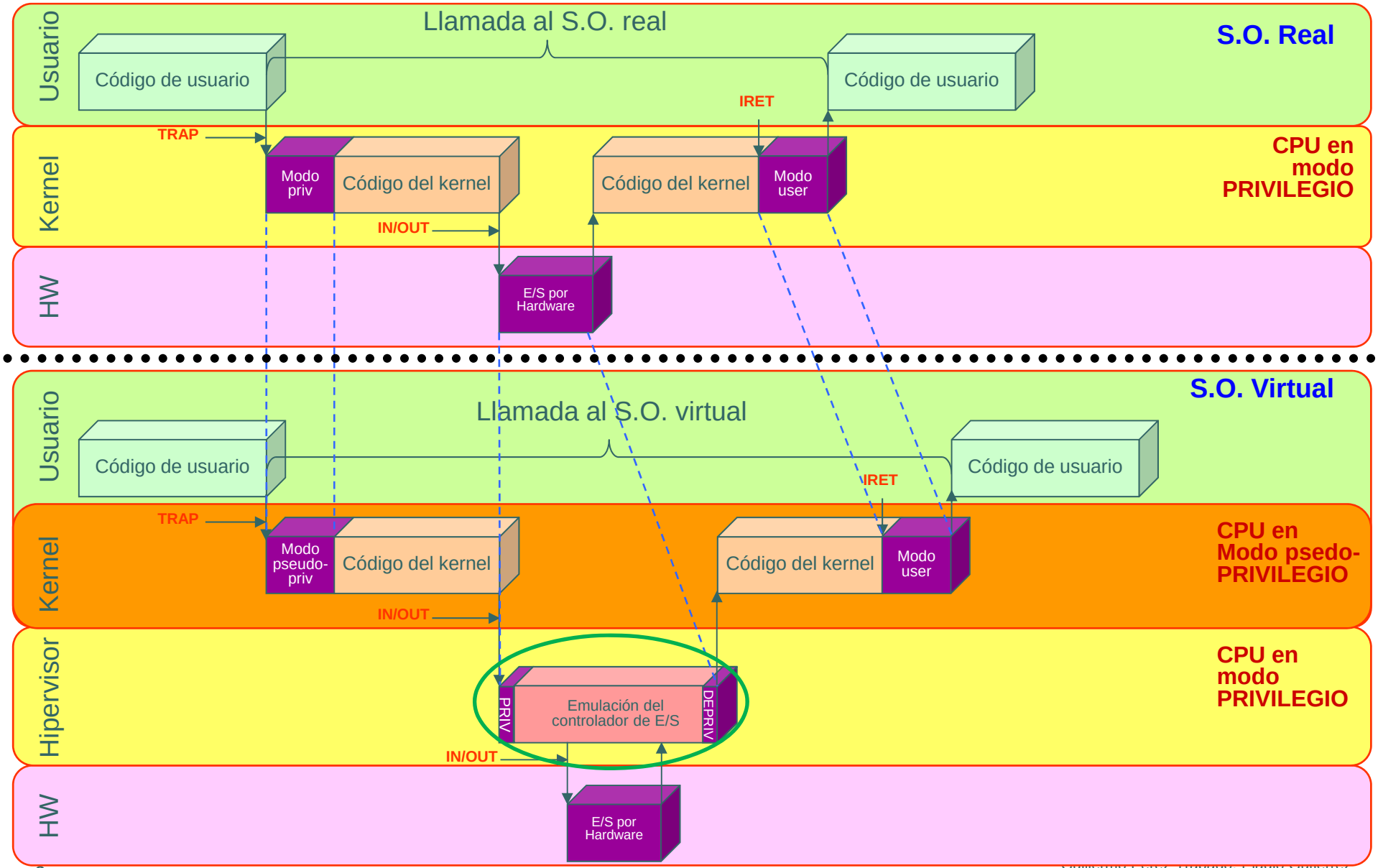
Sobrecostes inherentes a la virtualización

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Procesador: virtualización SW



Procesador: virtualización HW



Procesador

- El problema son las instrucciones **privilegiadas** y **E/S**

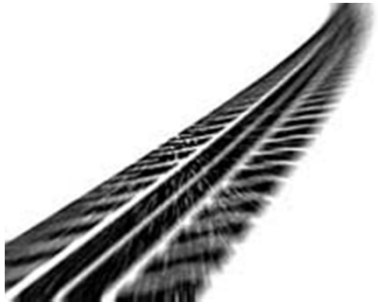
- **Traducción binaria** en tiempo de ejecución
  - Se traducen a un código seguro para la CPU real.
  - Incluye el código de emulación por software del hardware de E/S virtual (se evita en modo “by pass”).
  - La traducción lleva cierto tiempo adicional.

- ***Trap & Emulate***
  - Se generan excepciones en la CPU real, que debe emular dichas operaciones por software.

- El soporte hardware puede reducir el efecto de la E/S

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Procesador



- Pérdida de eficiencia por la virtualización S_v :

$$S_v = f_p \times N_e + (1 - f_p)$$

  - f_p: fracción de instrucciones privilegiadas
  - N_e: número medio de instrucciones que las reemplazan
    - Rutina de emulación
    - Excepción

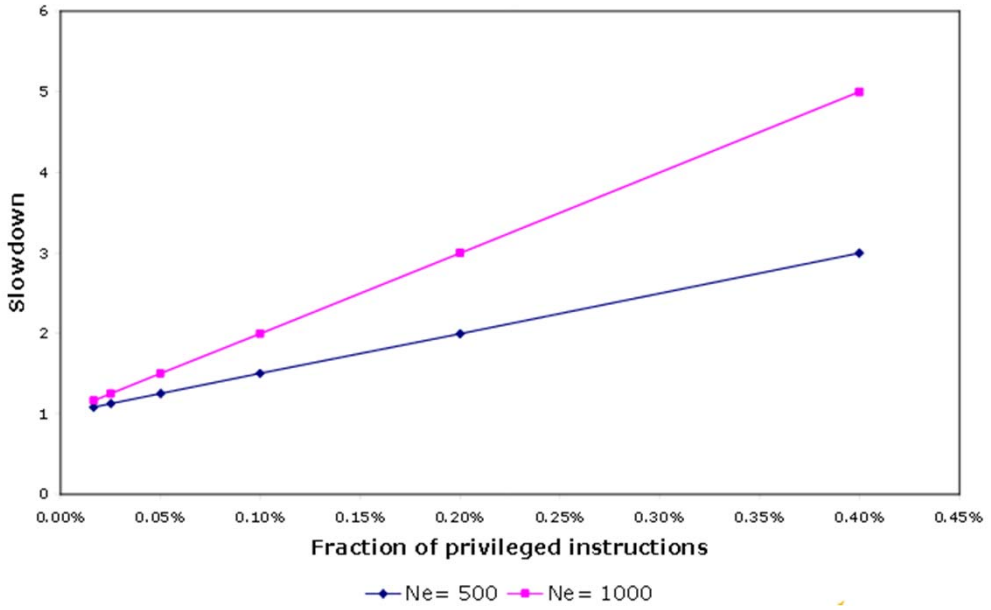
Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Procesador

☺
Cálculo intensivo.

☹
E/S Intensiva:
Bases de datos.

- f_p: dependerá de la aplicación
  - Bajo para CPU bound
  - Alto para I/O bound
- N_e: dependerá de la técnica de virtualización, pero serán cientos incluso miles



El problema de las excepciones

- Las excepciones invalidan todo el trabajo anticipado por el pipeline del procesador:
  - Prefetch de instrucciones y datos.
  - Localidad de la cache.
  - Predicción de saltos.
  - Otras optimizaciones agresivas del procesador y del compilador.
- A mayor longitud del pipeline, mayor pérdida de rendimiento en cada excepción.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

El problema de las excepciones

- Los programas I/O bound contienen una syscall (que implica un trap) cada pocas instrucciones.
  - Cuando se virtualiza hay que sumar el efecto de todos los traps de las instrucciones privilegiadas dentro de cada syscall.

```
while (!eof(f)) {
   l=read(f,buff,N);
   write (f,buff,l);}
```

  - El efecto de los traps suele ser menor que el de acceso a disco.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Memoria

- La máquina virtual requiere la simulación del espacio físico de memoria RAM y la unidad de gestión de memoria (MMU) del procesador:
  - Overhead de memoria y procesamiento.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Discos duros

- Se pueden implementar:
  - *By pass*: directamente sobre discos físicos, particiones, volúmenes lógicos.
  - *Ficheros*: del sistema de archivos del hipervisor

Discos duros

- Volúmenes físicos
  - Menos retardo adicional en cada operación de acceso.
  - Uso: aplicaciones críticas o para migrar un servidor existente.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Discos duros

- Implementados sobre ficheros
  - Overhead por las estructuras de datos del sistema de ficheros.
    - Menos velocidad de transferencia y más latencia.
  - Es posible no asignar el espacio libre del volúmen. Ahorra espacio, pero:
    - Puede provocar inanición por falta de disco real.
    - Puede introducir fragmentación del fichero(s) que simula el disco (adicional a la interna).

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Snapshots de disco

- Disminuyen la eficiencia del disco
  - En escritura hay que gestionar la asignación de espacio para las modificaciones.
  - En lectura hay que consultar el índice de bloques modificados que contiene el *snapshot*.
  - El fichero delta puede estar en un disco más lento que el disco original.
    - Se pueden desactivar para cada disco.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

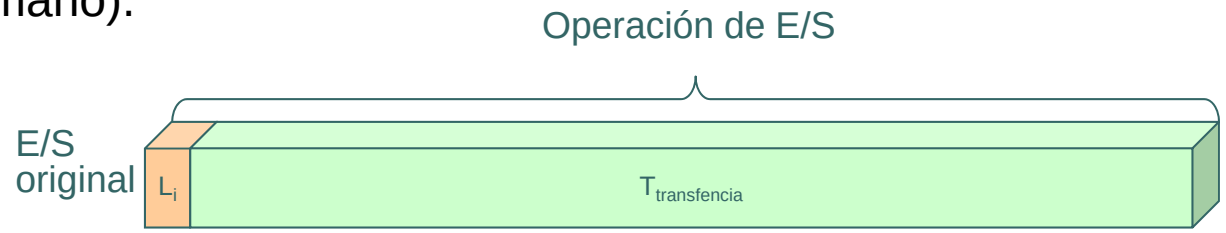
La red

- Requiere simular
  - Interfaces de red (NIC) en cada máquina virtual.
  - LANs virtuales (switches Ethernet).
- El paso de paquetes Ethernet entre NICs de máquinas virtuales conectadas al mismo switch virtual se hace mediante buffers de memoria del hipervisor.
  - La velocidad de paso de datos es muy elevada.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Eficiencia de la E/S

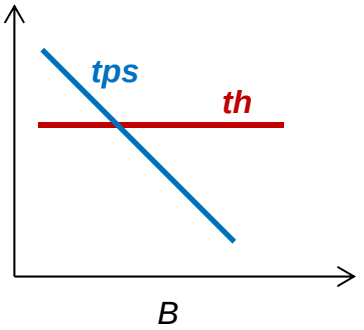
- Para evaluar el rendimiento en E/S hay que distinguir:
  - *Throughput*: velocidad máxima de transferencia en bytes/s.
  - *Latencia*: tiempo mínimo que tarda en transferirse el primer byte.
  - *tps*: transacciones por segundo máximas (del mínimo tamaño).



$$tps_{real} = \frac{1}{L_{inicial} + T_{Transferencia}}$$

$$th_{real} = B \cdot tps_{real}$$

$$B: \text{bytes por bloque}$$



Eficiencia de la E/S

- La virtualización afecta al rendimiento:
  - Aumenta la *latencia*
    - Código de simulación
    - Pérdida de rendimiento por las excepciones.
  - Se puede reducir con el uso de *buffering* en el sistema real.
  - Respecto al *throughput*
    - Una buena simulación lo mantiene al 100% del original
    - Puede degradarse por las copias entre buffers reales y virtuales

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Eficiencia de la E/S

- Comparación real vs virtual

REAL

$$tps_{real} = \frac{1}{L_{inicial} + T_{Transferencia}}$$

$$th_{real} = B \cdot tps_{real}$$

B: bytes por bloque

VIRTUAL

$$tps_{virtual} = \frac{1}{L_{inicial} + L_{virtual} + T_{Transferencia} + T_{buffer}}$$

$$th_{virtual} = B \cdot tps_{virtual}$$

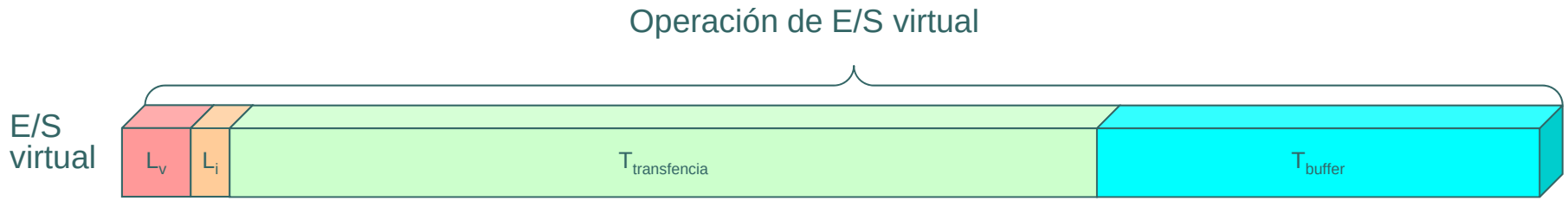
T_{buffer} : copia entre buffers

Operación de E/S virtual

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Eficiencia de la E/S

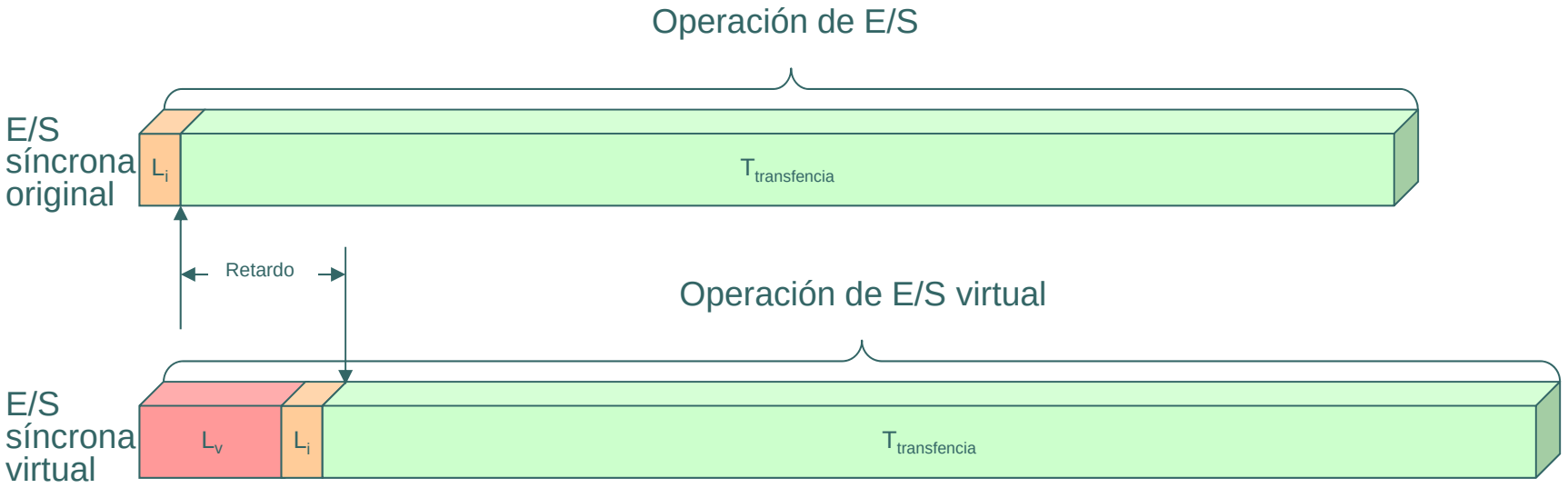
- El impacto de la virtualización depende de:
  - Tamaño de bloque de las transferencias de E/S:
    - Grandres vs Pequeños
  - Si hay copia entre buffers M. virtual y real
  - Tipo de escrituras:
    - Síncronas vs Asíncronas (¿debo esperar al final?)



Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

E/S con bloques grandes

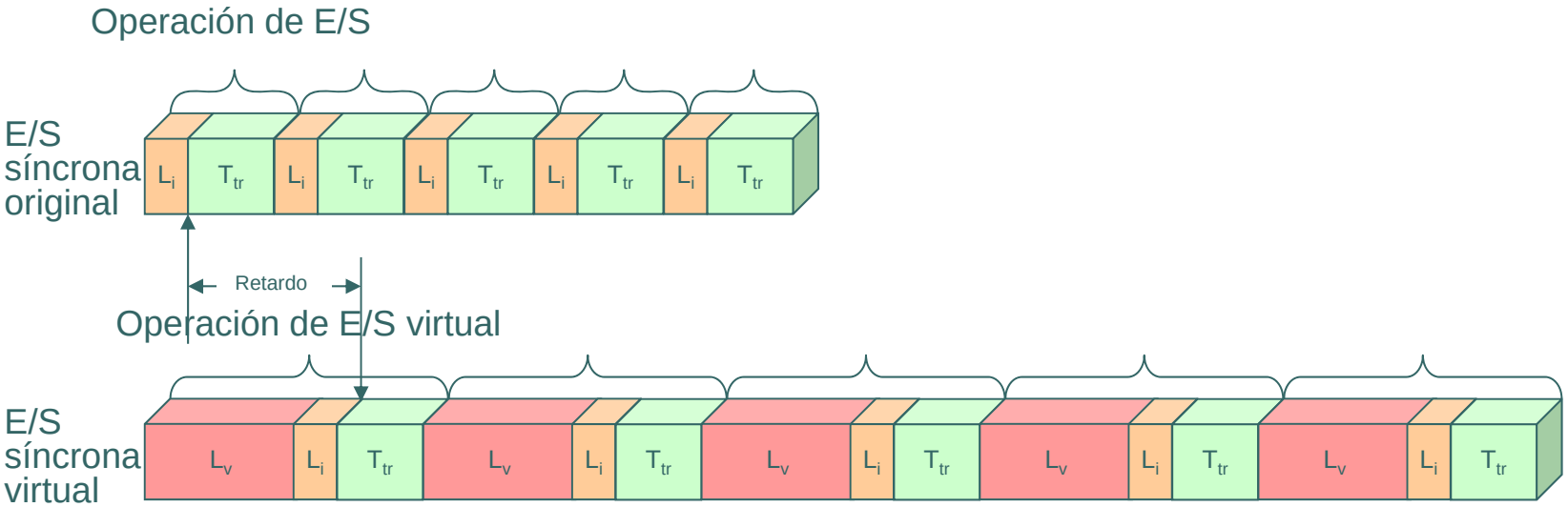
- El overhead extra en la *latencia* representa un porcentaje pequeño del tiempo total.
  - En cada operación puede observarse el retardo adicional.
- El número de *transacciones por segundo* se mantiene cerca del original.
- Se obtiene un *throughput* cercano al original en transferencias de grandes volúmenes de información.



Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

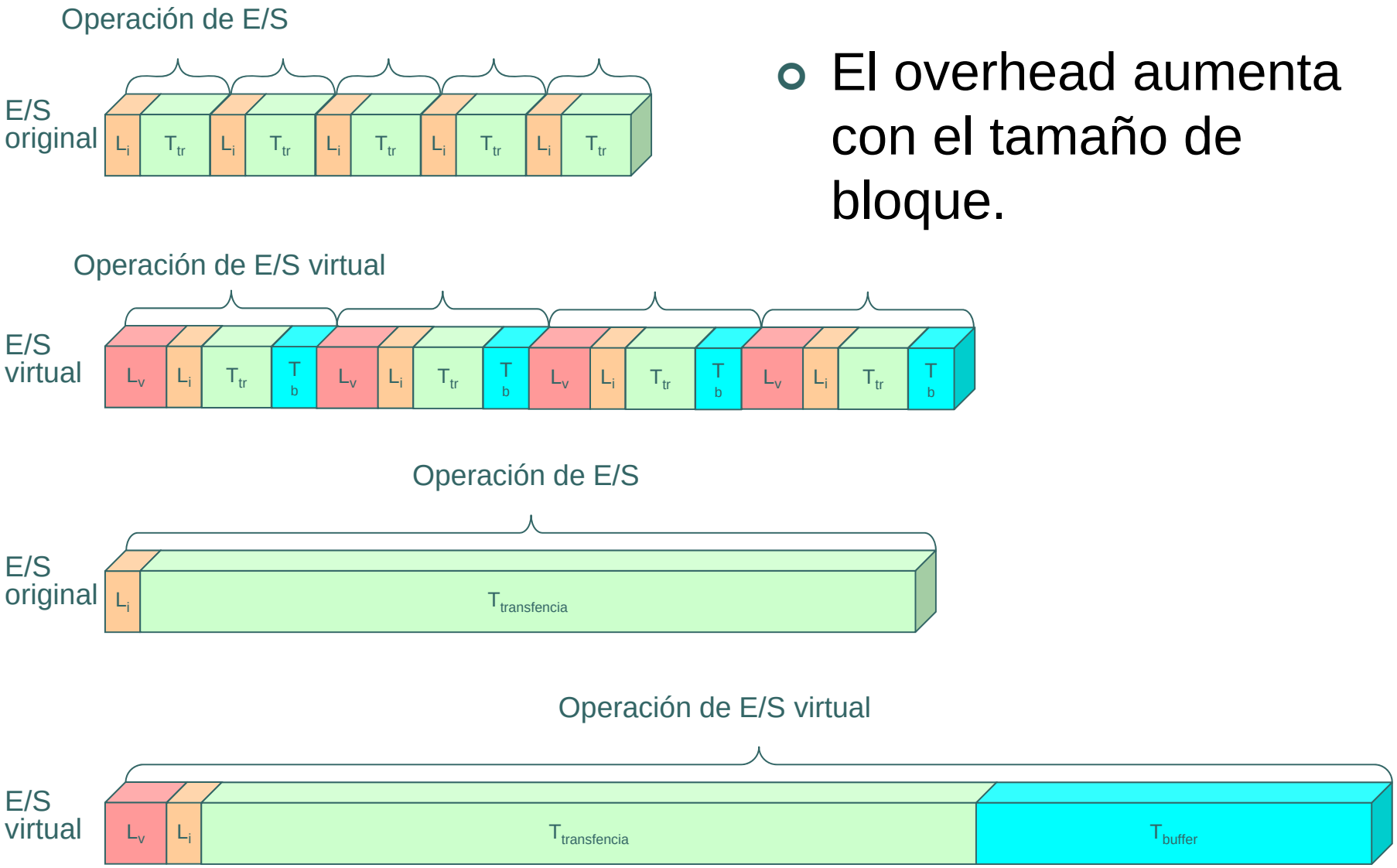
E/S con bloques pequeños

- El overhead extra en la *latencia* representa un gran porcentaje sobre el total.
  - En cada operación puede observarse el retardo adicional.
- El número de *transacciones por segundo* en transferencias continuadas decae de forma dramática.
- El *throughput* también decae.



Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

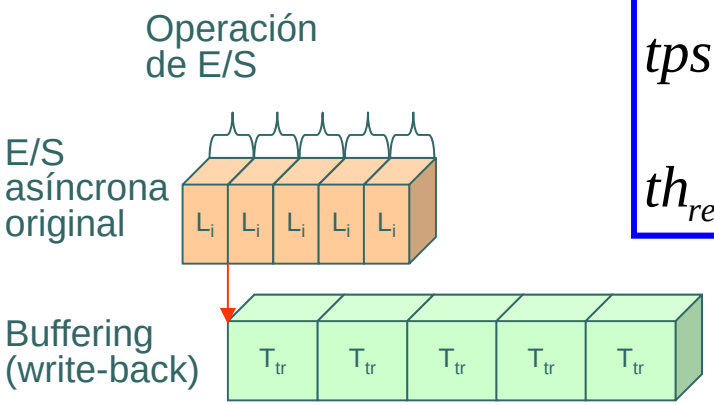
Impacto de copiar entre buffers



- El overhead aumenta con el tamaño de bloque.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

E/S asíncrona

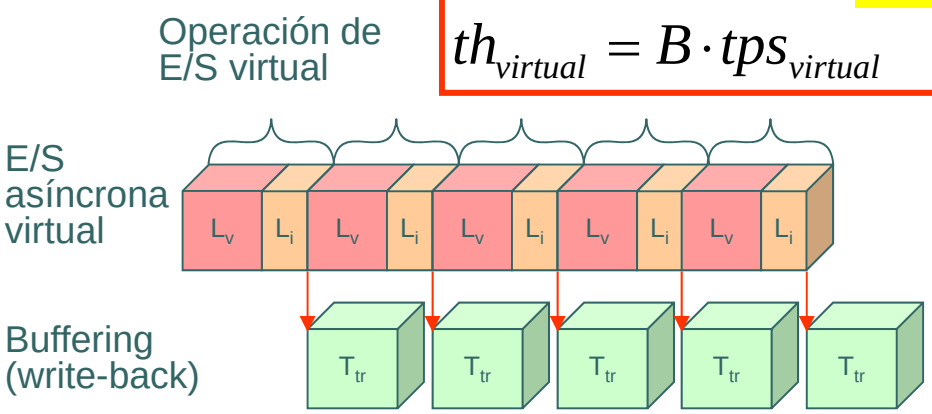


$$tps_{real} = \frac{1}{L_{inicial}}$$

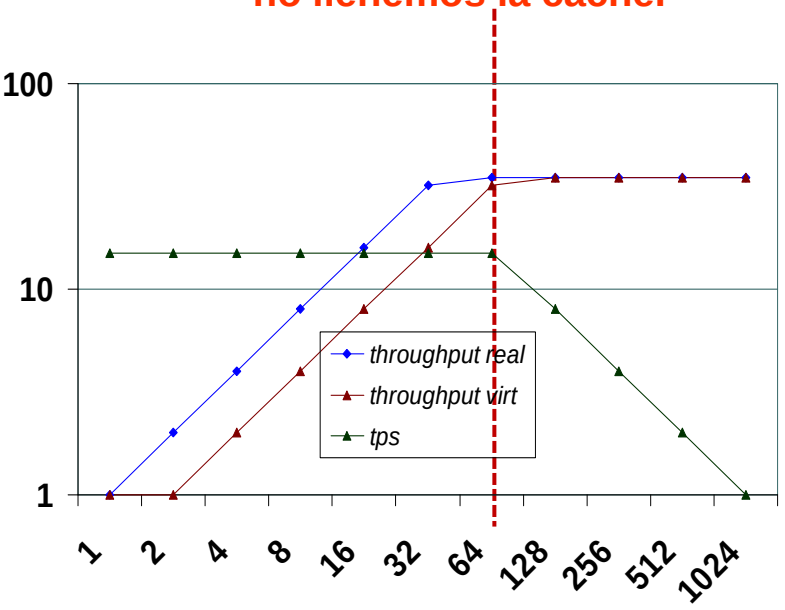
$$th_{real} = B \cdot tps_{real}$$

$$tps_{virtual} = \frac{1}{L_{inicial} + L_{virtual}}$$

$$th_{virtual} = B \cdot tps_{virtual}$$



- Afecta a la *latencia* de cada operación y a la velocidad en *tps*.
- La cache del SO virtual oculta el tiempo de copia entre buffers reales y virtuales (si existe).
- El *throughput* solo está limitado por las *tps*.
- A mayor tamaño de bloque mayor *throughput* **mientras no llenemos la cache.**



Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

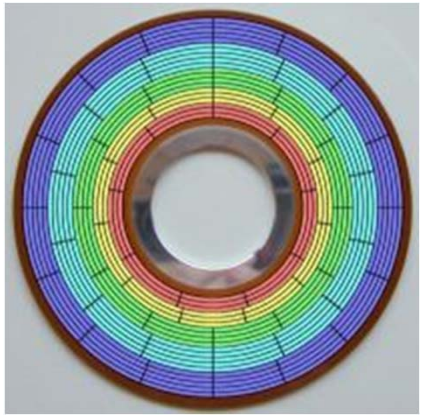
Eficiencia E/S. Conclusiones

- ¿Qué usar?
  - Bloques grandes ó
  - Bloques pequeños con E/S asíncrona
- ¿Cómo controlo el tamaño de bloque? Depende de la aplicación
  - Pe en BD puedo configurar los *buckets* o nº de registros modificados antes de actualizar la BD en disco
- *tps* y *throughput* son incompatibles, ¿cuál mejoro? Depende la aplicación
  - *tps* muchos clientes con datos pequeños (alta concurrencia)
  - *throughput* pocos clientes con datos grandes (pe servidor videos HD)

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Eficiencia E/S. Conclusiones

- Otras consideraciones



  - En un disco magnético
    - La velocidad de giro es constante (rpm)
    - La densidad de información es constante
      - Hay más sectores en las pistas exteriores
  - Por tanto, el *throughput* aumenta hacia el exterior (pista 0) del disco
    - Podríamos usar solo la primera mitad
  - Movimiento de cabeza lectora no optimizable globalmente (disco compartido: LUNs, RAIDs)
  - Los discos estado sólido (SSD)
    - No tienen posicionamiento (menor latencia)
    - Tienen mayor velocidad de transferencia (*throughput)*
    - Problema del desgaste (aumenta latencia)
    - Coste elevado (aún)

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

¿Y para qué virtualizamos?

- Si se pierde rendimiento…

  …¿qué ganamos?

- Ganamos ventajas de alto nivel:
  - Portabilidad indefinida:
    - Independencia de los fabricantes y la tecnología.
  - Rentabilidad económica:
    - Mayor aprovechamiento del hardware.
    - Menor consumo de energía y generación de calor.
  - Protección:
    - Tolerancia a fallos.
    - Recuperación de desastres.
  - Reducción de costes (hardware, administración, consumo de electricidad)

Mejora del rendimiento: Optimizaciones en el SO guest

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Paravirtualización

- Requiere reescribir el *kernel* del SO virtual, reemplazando las llamadas al núcleo con llamadas al hipervisor.
  - Usa el soporte de virtualización por HW del procesador.
  - Reduce el número de excepciones en la implementación de la E/S (no usa instrucciones IN/OUT para provocar la llamada al hipervisor).

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Paravirtualización

- Puede implementar optimizaciones internas de los periféricos antes de encargar la operación al hipervisor. Ejemplos:
  - Discos: simular reordenación de comandos.
  - E/S: eliminar copias entre buffers, usando E/S “in-place”.
  - Red: copiar paquetes directamente entre *kernels* de MVs.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Driver optimizado

- Requiere escribir *drivers* compatibles con el *guest* OS, usando llamadas al hipervisor
  - Más simple, solo reescribir el driver para el hardware virtual, no el real.
  - Similar a la paravirtualización, pero aplicada solo nivel de *drivers* (no todo el *kernel*).
  - La única opción si no tenemos los fuentes del SO (pe Windows)

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Driver optimizado

- También se realizan optimizaciones internas dentro del *driver* antes de llamar al hipervisor:
  - Simular reordenaciones de comandos de disco.
  - Reducir copias entre *buffers* usando E/S "*in-place*".
  - Optimizaciones de red.
    - Ejemplo. Para una red a 1.000 Mbs (unos 110 Mbytes efectivos):
      - Driver Flexible (en todos los SOs): 300 Mbps
      - Driver VMXNET 2 (requiere Tools): 975 Mbps

Comparativa

- La paravirtualización permite optimizaciones más amplias, ya que afecta a todo el *kernel*.
  - Reduce al mínimo el *overhead* de cada operación.
- Aumenta el número máximo de operaciones de E/S por segundo.
- La paravirtualización mejora notablemente la E/S de bloques pequeños:
  - Acceso a disco de Bases de Datos
  - Transacciones de red (*web services*)

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Comparativa

- El *driver* optimizado y la paravirtualización del *kernel* obtienen el mismo *throughput* (normalmente):
  - Reducen el número de excepciones eliminando instrucciones IN/OUT.
  - Usan copias "*in-place*".

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Comparativa

- El *driver* optimizado puede ser más fácil de implementar que el *kernel* paravirtualizado.
  - SLES y RedHat ofrecen *kernels* paravirtualizados para XEN y VMWare.
  - ¡¡¿Windows paravirtualizado?!! (Existió)
- Los rendimientos obtenidos son similares

Overcommiting

Compartición de recursos entre MVs

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Overcommiting

- Muchos servidores están inactivos casi todo el tiempo
- *Overcommiting*: multiplexación de recursos físicos entre máquinas virtuales
  - Supone un uso “ineficiente” del hardware disponible
  - Busca la reducción de costes mediante la compartición del mismo hardware por varios servidores virtuales
  - Se asignan más recursos virtuales de los que existen realmente
  - No existen garantía de que se proporcionen los recursos cuando se necesitan

Overcommiting

- Estadísticamente funciona porque el uso medio en conjunto de todas las MVs no supera los recursos existentes.

- Puntualmente, el uso total puede superar los recursos existentes (congestión)
  - No se puede prever ni su aparición ni su duración (puede prolongarse indefinidamente)
  - En esta situación, se bloquea a los solicitantes en espera de que los recursos estén disponibles
    - Puede reducir gravemente la eficiencia

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Compartición de la CPU

- Terminología
  - CPU = procesador físico
  - Core físico = core de CPU física
  - *Hyper-Threading* = dos instancias lógicas de un core físico

  - vCPU = CPU virtual, en la MV
  - vCore = núcleo de una vCPU
  - LCPU = CPU lógicas disponibles en el servidor que ejecuta el hipervisor

  - Ejemplo:
    - 2 CPU físicas, con 6 cores e hyper-threading activo => 24 LCPU => Ejemplos de configuraciones posibles:
    - 1 MV con 24 vCPU ó 4 vCPU de 6 vCores

Compartición de la CPU

- Recursos asignados:
  - Número de cores de la MV
  - Porcentaje de uso de esos cores
- Overcommiting de la CPU: la suma de las vCPUs de las MVs supera a las CPUs
- Si el tiempo de proceso de las vCPUs excede la capacidad de las CPUs
  - Se aplica *time-sharing* entre MVs
  - Los SOs guest observan tiempos de espera adicionales inexplicables
  - Podrían aparecer retardos que afectaran a la temporización de una aplicación crítica

Compartición de la RAM

- Terminología
  - Memoria **física**
    - Existente en la realidad (en el servidor que ejecuta hipervisor)
  - Memoria **configurada** en la MV
  - Memoria **reservada** para MV (garantizada)
    - Mínima que necesita para funcionar sin problemas
    - Menor (o igual) que la configurada
  - Memoria **asignada** a la MV (accedida)
    - Mayor o igual a reservada
    - Menor o igual a configurada
  - Memoria usada recientemente (***working set***)
    - Es la más crítica, debería estar en memoria física siempre (debería caber en reservada)
    - Cuando no es posible se produce *trashing* entre MVs

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Compartición de la RAM

- Overcommiting de la RAM: la suma de las RAMs configuradas en las MVs supera a la física del servidor
  - No tiene porqué ser un problema
  - El hipervisor implementa memoria virtual
    - Asignación bajo demanda
    - Espacio de *swap* en disco
- Los SOs *guest* observan tiempos de espera inexplicables al acceder a ciertas páginas de memoria
  - La paginación del *host* y el *guest* interfieren

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Compartición de la RAM

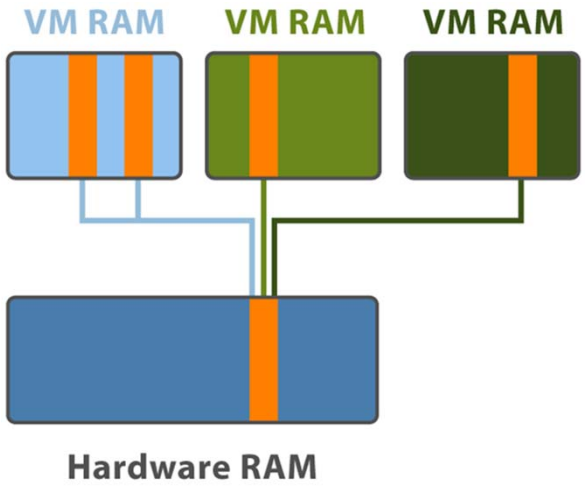
- En un SO paravirtualizado el hipervisor sabrá qué memoria puede recuperar de las MVs (no usada)

- Si no lo es, existen técnicas para intentar evitar que el rendimiento se degrade (en orden de prioridad)
  - Compartición de memoria
  - *Ballooning*
  - Compresion de memoria
  - *Swap*

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Compartición de la RAM

- Compartición de memoria (Transparent Page Sharing – TPS)
  - SOs similares en MVs distintas o creadas de la misma plantilla tienen páginas idénticas
  - Mientras no se escriban, se garda una sola copia compartida en RAM
  - La sobrecarga del escaneo para TPS es baja

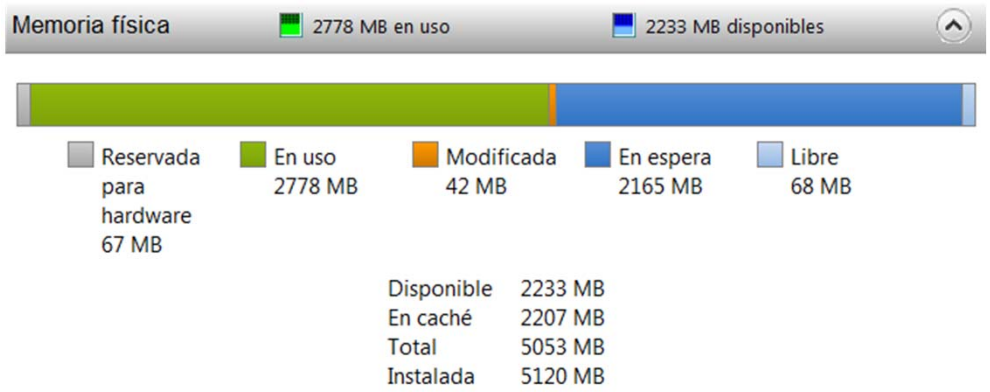


Compartición de Recursos entre MVs

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Compartición de la RAM

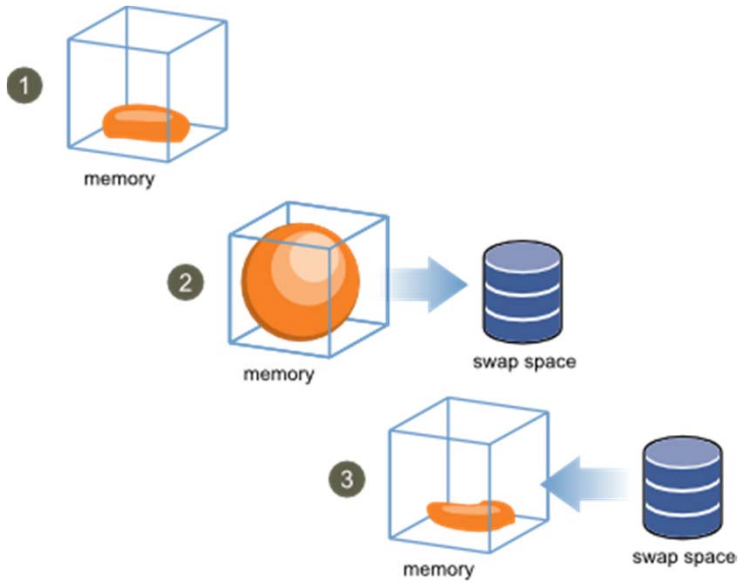
- Ballooning (*driver* de las Tools: `vmemctl`)
  - Se basa en que solo un bajo porcentaje de la RAM se usa intensivamente en el *guest*. Ejemplo Windows 7:



- **Reservada para hardware**: memoria reservada para uso por parte de la BIOS y algunos controladores para otros periféricos
- **En uso**: usada por los procesos, los controladores o el sistema operativo
- **Modificada**: memoria cuyo contenido debe escribirse en el disco antes de poder usarse con otra finalidad
- **En espera**: memoria que contiene datos y código en caché que no se usan de forma activa
- **Libre**: memoria que no contiene datos de valor y que se usará primero cuando los procesos, los controladores o el sistema operativo necesiten más memoria

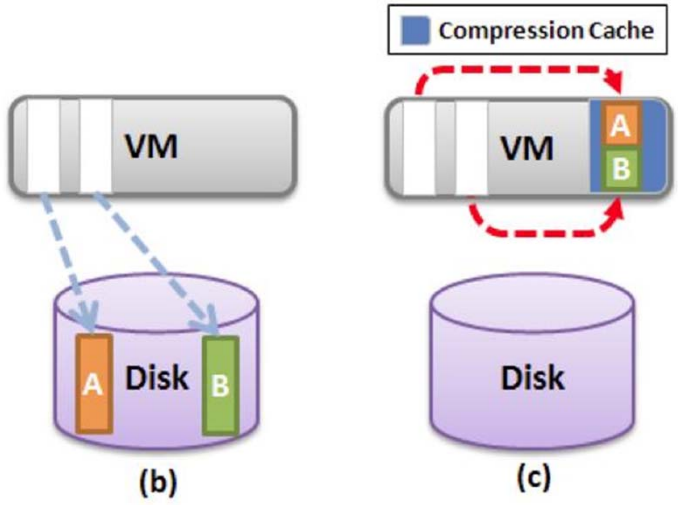
Compartición de la RAM

- Ballooning. Procedimiento
  - El balón se infla (reclama memoria)
  - Es SO *guest* libera memoria menos útil
  - El balón entrega memoria al hipervisor



Compartición de la RAM

- Compresión de memoria
  - Mismo principio que caché víctima
  - Nivel intermedio, antes del *swap*
  - Comprime la página y la almacena en un espacio protegido de memoria (4K->2K)
  - Es más rápido que el acceso al disco y no degrada mucho en rendimiento



Compartición de Recursos entre MVs

Compartición de la RAM

- Swap
  - Usa el disco como memoria
  - Es muy lento, usar como último recurso
    - Ejemplo: 1 seg RAM = 11.5 días en disco
  - No hay problema si el *working set* de la MV reside en la memoria física
  - Tamaño Swap = Mem configurada - reserva

Compartición E/S

- Discos Virtuales
  - El ancho de banda de los discos físicos será compartido
    - Existen mecanismos para balancear su uso entre MVs
  - Discos tipo *thin* (el espacio se reserva al usarse)
    - Puede haber problemas de inanición si reservo tipo *thin*
    - No se libera el espacio de un disco *thin* borrado
  - Ventajas de usar *linked-clones*
    - El espacio se ocupa solo una vez
    - Se mejora la concurrencia de E/S a disco si el hipervisor implementa una cache de disco para los bloques de los discos virtuales

Compartición E/S

- Red
  - El ancho de banda será compartido
    - Se puede limitar el tráfico saliente de cada MV
  - Comunicamos MVs sin necesidad de NICs físicos

- Paravirtualización y *drivers* pueden dar mejor rendimiento para E/S que en máquinas reales

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Asignación de recursos

Asignacion de recursos

- Para tomar decisiones de optimización reales hay que conocer las necesidades de recursos de cada aplicación.
- Se puede hacer una sobreasignación inicial y observar el uso real que hace la MV durante un periodo de tiempo.
- El hipervisor ofrece herramientas para almacenar estadísticas de uso durante largos periodos de tiempo.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Tasa de consolidación

- Número de MVs ejecutadas simultáneamente en el servidor
  - Respetando rendimiento y QoS exigido
- La tarea del buen administrador es maximizarla con ayuda de herramientas de gestión
- No existe un valor fijo, dependerá de la potencia del *host* y las aplicaciones ejecutadas en las MVs
  - El sobre-dimensionamiento perjudica el rendimiento
  - El infra-dimensionamiento es un derroche para la empresa

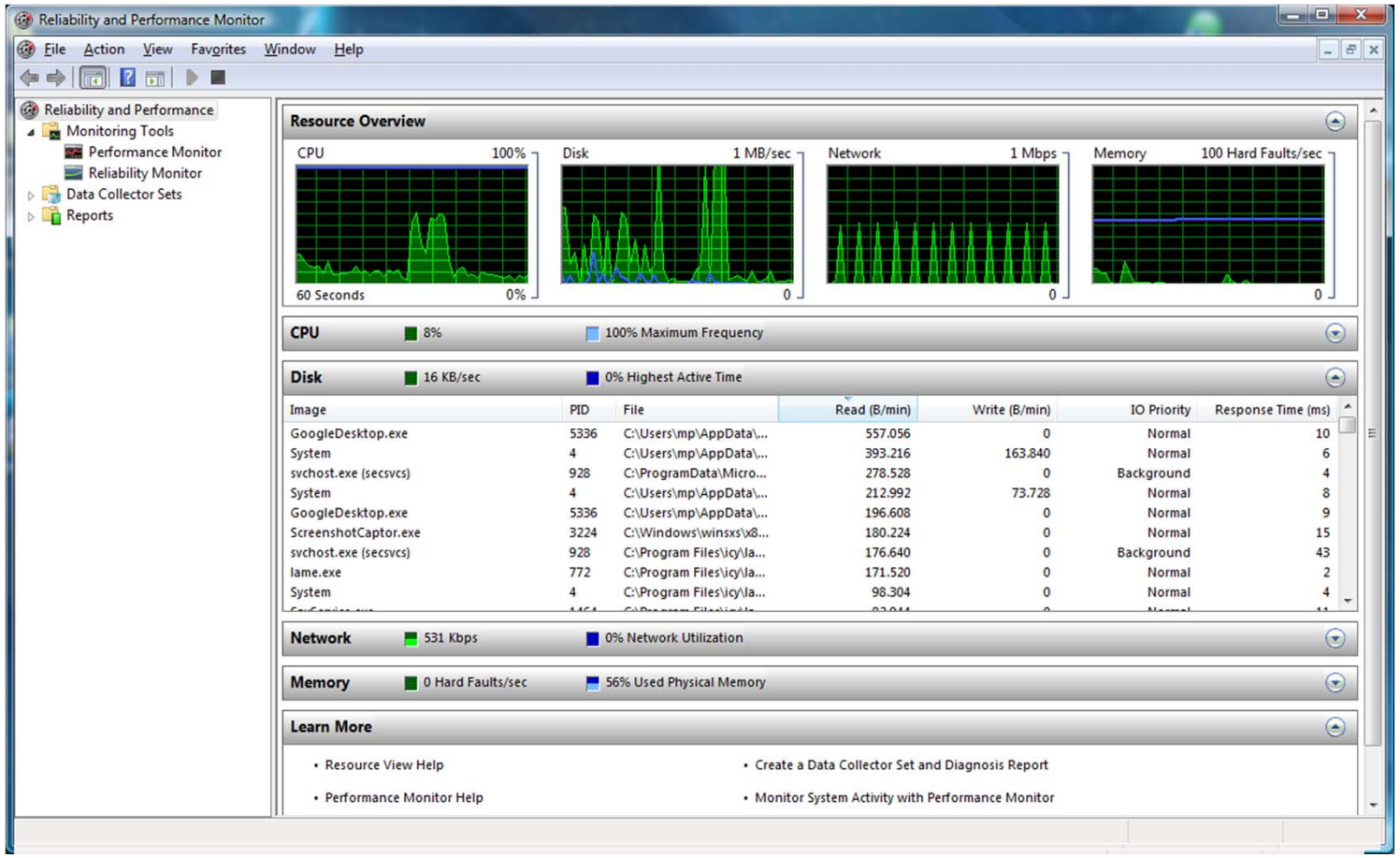
Agentes de monitorización

- Lo ideal es tomar datos tanto del interior del SO *guest* como del hipervisor.
  - Monitorizar el interior requiere instalar un agente de monitorización en el SO invitado.
  - El hipervisor suele contar con algún agente de monitorización integrado.

- Si queremos monitorizar un cluster, necesitamos recopilar centralizadamente los datos de todas las MVs y de todos los hipervisores.

Agentes de monitorización

- `perfmon` (Windows)



Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Agentes de monitorización

- esxtop

```
 10:09am  up 22:09, 16 worlds,  load average: 0.03, 0.01, 0.00, 0.00
PCPU:   3.49%,   1.95% :   2.72% used total
LCPU:   3.07%,   0.42%,   1.91%,   0.04%
MEM: 850944 managed(KB), 270336 free(KB) :  68.23% used total
SWAP: 1047552 av(KB), 0 used(KB), 1037080 free(KB) :    0.00 MBr/s,    0.00 MBw/s
DISK vmhba0:6:0:    0.00 r/s,    0.00 w/s,    0.00 MBr/s,    0.00 MBw/s
DISK vmhba0:0:0:    0.00 r/s,    7.57 w/s,    0.00 MBr/s,    0.02 MBw/s
NIC vmnic1:    0.00 pTx/s,   14.55 pRx/s,    0.00 MbTx/s,    0.01 MbRx/s
NIC vmnic0:    0.00 pTx/s,   14.55 pRx/s,    0.00 MbTx/s,    0.01 MbRx/s

 VCPUID  WID  WTYPE     %USED  %READY  %EUSED   %MEM
  129    129  idle     59.86    0.00   59.86   0.00
  128    128  idle     50.83    0.00   50.83   0.00
  131    131  idle     45.77    0.00   45.77   0.00
  130    130  idle     38.14    0.00   38.14   0.00
  127    127  console   2.31    0.02    2.31   0.00
  142    142  vmm       2.29    0.36    2.29  35.00
  143    143  vmm       0.76    0.22    0.76  15.00
  132    132  helper    0.02    0.22    0.02   0.00
  140    140  driver    0.00    0.00    0.00   0.00
  139    139  reset     0.00    0.00    0.00   0.00
  138    138  reset     0.00    0.00    0.00   0.00
  137    137  helper    0.00    0.00    0.00   0.00
  136    136  helper    0.00    0.00    0.00   0.00
  135    135  helper    0.00    0.00    0.00   0.00
  134    134  helper    0.00    0.00    0.00   0.00
  133    133  helper    0.00    0.00    0.00   0.00
```


Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Agentes de monitorización

- vmkusage

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Herramientas de análisis

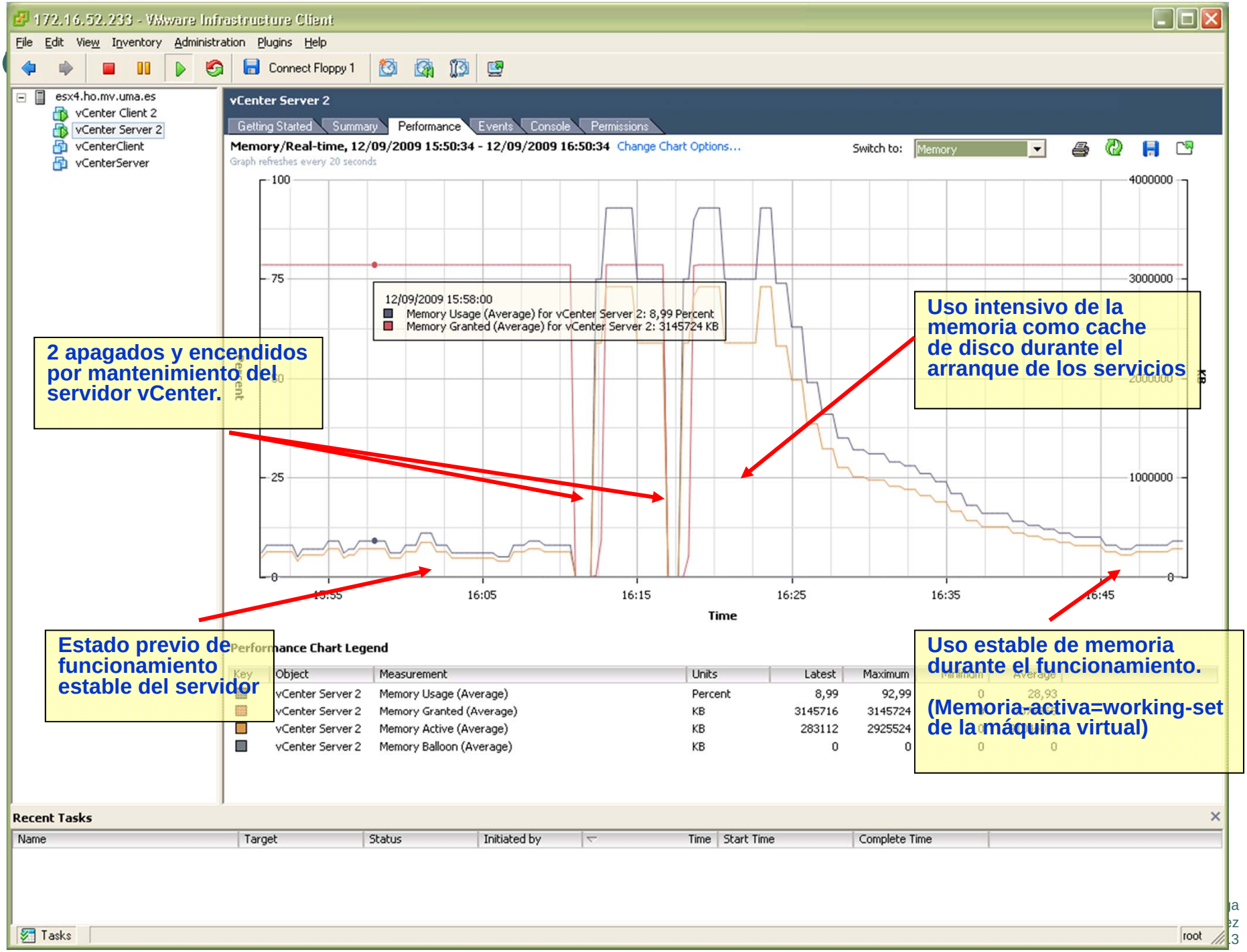
- Las herramientas gráficas permiten un vistazo rápido para detectar situaciones anómalas.
- Es deseable poder exportar las estadísticas para realizar análisis más completos mediante programas más complejos y trabajar sobre una gran cantidad de datos.
  - Máximos, mínimos, medias, desviación estándar, correlación entre los recursos usados en distintas máquinas de la misma aplicación, etc.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Ejemplo: arranque del servidor

- El arranque de un sistema implica consumo de recursos masivo.
  - Puesta en marcha de procesos de servicios en secuencia (pero sin espera ni control sobre el estado del servicio).
  - Solapamiento entre la inicialización de servicios arrancados en secuencia.
- Uso masivo de CPU
  - Los servicios no son procesos interactivos: no se bloquean hasta completar la inicialización.
- Uso masivo de memoria
  - Carga de aplicaciones (cache de disco y espacio de procesos)
  - Código de inicialización de las aplicaciones
  - Acceso a ficheros de configuración, etc.
- Se parte de un estado ineficiente con la cache de disco vacía.
  - Acceso físico a disco para todo

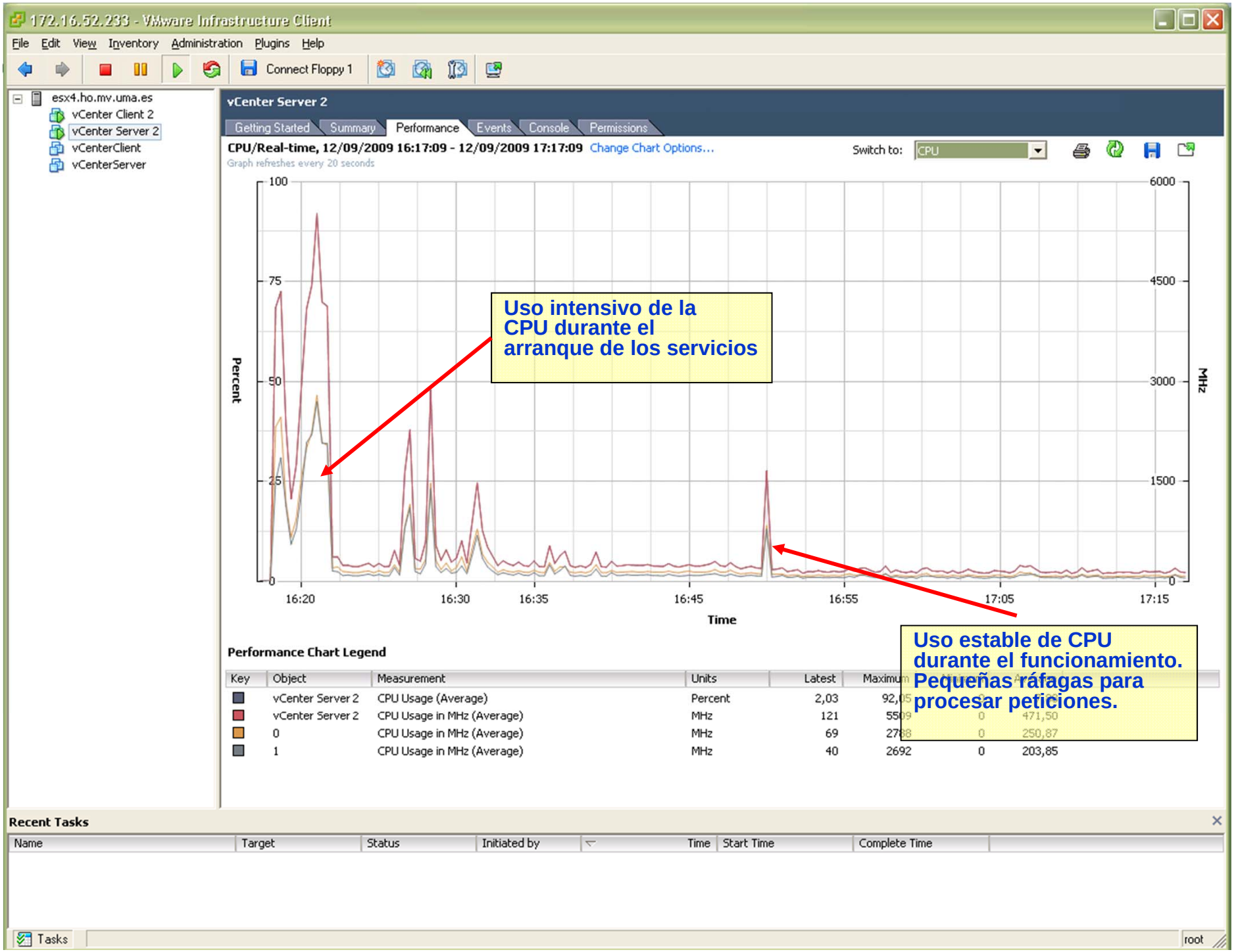
Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar


172.16.52.233 - VMware Infrastructure Client
File Edit View Inventory Administration Plugins Help
Connect Floppy 1
esx4.ho.mv.uma.es
vCenter Client 2
vCenter Server 2
vCenterClient
vCenterServer
vCenter Server 2
Getting Started Summary Performance Events Console Permissions
Memory/Real-time, 12/09/2009 15:50:34 - 12/09/2009 16:50:34 Change Chart Options...
Switch to: Memory
Graph refreshes every 20 seconds
12/09/2009 15:58:00
Memory Usage (Average) for vCenter Server 2: 8,99 Percent
Memory Granted (Average) for vCenter Server 2: 3145724 KB
Percent
KB
Time
Performance Chart Legend
Key Object Measurement Units Latest Maximum
vCenter Server 2 Memory Usage (Average) Percent 8,99 92,99
vCenter Server 2 Memory Granted (Average) KB 3145716 3145724
vCenter Server 2 Memory Active (Average) KB 283112 2925524
vCenter Server 2 Memory Balloon (Average) KB 0 0
Recent Tasks
Name Target Status Initiated by Time Start Time Complete Time
Tasks
root
2 apagados y encendidos por mantenimiento del servidor vCenter.
Uso intensivo de la memoria como cache de disco durante el arranque de los servicios
Estado previo de funcionamiento estable del servidor
Uso estable de memoria durante el funcionamiento.
(Memoria-activa=working-set de la máquina virtual)

Ejemplo: arranque del servidor

- Es un estado limitado
  - Se acaba tan pronto como se bloquean los servicios en espera de conexiones de red.
- El uso de recursos baja posteriormente gracias a la localidad
  - Solo se accede la memoria del “working-set” de los procesos, del kernel, y la parte de la cache de disco que contiene los datos usados.
  - Los procesos se bloquean y el consumo de CPU casi desaparece.
- No es tan grave…
  - Es tan ineficiente como en una máquina real.
- …pero
  - Se puede complicar si varias máquinas virtuales arrancan a la vez.
- Optimización
  - Es vital secuenciar el arranque de las máquinas para evitar la competencia por los recursos si no están garantizados (el disco es el más difícil de garantizar).
  - Definir un tiempo de gracia en el hipervisor suficiente para inicializar todos los servicios.
  - Esperar una señal desde dentro del S.O. invitado:
    - Arranque de VMWare Tools => Verificar que es el último servicio en arrancar.
- ¡Otro motivo más para instalar VMWare Tools!

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar


172.16.52.233 - VMware Infrastructure Client
File Edit View Inventory Administration Plugins Help
Connect Floppy 1
esx4.ho.mv.uma.es
vCenter Client 2
vCenter Server 2
vCenterClient
vCenterServer
vCenter Server 2
Getting Started Summary Performance Events Console Permissions
CPU/Real-time, 12/09/2009 16:17:09 - 12/09/2009 17:17:09 Change Chart Options...
Graph refreshes every 20 seconds
Switch to: CPU
Percent
MHz
Time
16:20 16:30 16:35 16:45 16:55 17:05 17:15
Uso intensivo de la CPU durante el arranque de los servicios
Uso estable de CPU durante el funcionamiento. Pequeñas ráfagas para procesar peticiones.
Performance Chart Legend
Key Object Measurement Units Latest Maximum
vCenter Server 2 CPU Usage (Average) Percent 2,03 92,05
vCenter Server 2 CPU Usage in MHz (Average) MHz 121 5509 0 471,50
0 CPU Usage in MHz (Average) MHz 69 2738 0 250,87
1 CPU Usage in MHz (Average) MHz 40 2692 0 203,85
Recent Tasks
Name Target Status Initiated by Time Start Time Complete Time
Tasks
root

Recursos del Servidor

Configuración de recursos

- Son parámetros de calidad de servicio (QoS)
- Permiten definir garantías de mínimos, límite de uso, prioridades, cuotas
  - La suma de las garantías mínimas no puede superar los recursos existentes.
  - La suma de los límites de uso sí puede superar el total.
- El administrador debe estudiar las necesidades y definir los requerimientos de recursos de cada máquina.

Configuración de recursos

- El hipervisor permite definir directivas estáticas de reparto de recursos entre MVs. Ejemplos:
  - Asignación de CPU (en %tiempo o ciclos por segundo).
  - Políticas de asignación de RAM (en MB absolutos).
- El hipervisor asigna los recursos en tiempo real siguiendo las directivas.

CPU

- Modo de virtualización: hardware o software
- Visibilidad modelo: portabilidad vs eficiencia
- Número de vCPU (activar SMP)
- Afinidad vCPU con CPUs

- Límite uso CPU (MHz) = máxima
- Garantía CPU (MHz) = mínima
- *shares*: importancia relativa de las máquinas
  - Deciden la prioridad en caso de competir por uso de la CPU

CPU. Ejemplo de *shares*

- Situación 1:
  - Servidor de 8GHz de capacidad agregada
  - Dos MVs *CPU-bound*
  - Si sus *shares* son **Normal**, cada una obtendrá
    - MV1 = 4 GHz; MV2 = 4 GHz

- Situación 2:
  - Se enciende una tercera MV *CPU-bound*
  - Su *share* está a **High** (doble de *shares* que las MVs con *shares* a Normal)
  - La nueva recive: MV3 = 4GHz
    - Las dos MVs antiguas: MV1 = MV2 = 2GHz

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

RAM

- Configurada = máxima
- Reserva = mínima
- *swap* = Configurada-Reserva
- *shares*: prioridad relativa
  - Pe: decide dónde hinflar el *balloon*
- *Idle memory tax*
  - Recuperar memoria inútil independientemente del share

RAM. Ejemplo *shares*

- Situación 1:
  - Servidor con 4 GB de RAM
  - 2 MVs con 4GB configurada cada una
    - MV1 High (ratio 4) y MV2 Low (ratio 1)
  - Reparto de memoria asignada
    - MV1 = 4/5 * 4GB = 3.2 GB
    - MV2 = 1/5 * 4GB = 800 MB
- Situación 2:
  - Nueva MV con 4GB y *share* High
    - MV1 = MV3 = 4/9 * 4GB = 1.77 GB
    - MV2 = 1/9 * 4GB = 444 MB

RAM. Ejemplo idle memory tax

- Por defecto recupera hasta un 75% de memoria no usada
- Sin *idle memory tax*
  - Servidor 4GB de RAM
  - 2 MVs de 4GB con *shares* Normal
    - MV1 poca actividad (usa 1GB de 4GB)
    - MV2 actividad intensa (usa 3GB de 4GB)
  - MV1 = MV2 = 2GB
- Con *idle memory tax* puede recuperar hasta el 75% de 1GB = 750 MB
  - MV1 = 1.25 GB
  - MV2 = 2.75 GB

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Disco

- Eficiencia en el acceso a disco
  - Control totalmente manual mediante la ubicación de los discos virtuales
  - Distribuir discos virtuales en discos físicos distintos
  - No usar *snapshots* o solo en el disco de sistema (nunca en los datos)
  - Usar directamente particiones para datos (nunca para el sistema)
  - A veces no tiene sentido tener caché de disco en RAM (puede acabar en disco si no hay RAM suficiente)

Clusters de Servidores

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Clusters de Servidores

- Un hipervisor distribuido controla varios servidores físicos
- Permite **migrar** máquinas virtuales entre servidores sin pararlas:
  - La RAM de la máquina migrada se copia al nuevo servidor junto con el contexto de la CPU.
  - Requiere una SAN (red de almacenamiento) para no mover los discos virtuales entre servidores físicos.
  - Todos los servidores deben estar conectados a la misma LAN para continuar el acceso de red sin interrupciones.

Universidad de Málaga
Guillermo Pérez Trabado, Eladio Gutiérrez Carrasco, Julián Ramos Cózar

Aplicaciones de la migración

- Reducir la **competencia** por los recursos entre máquinas virtuales
  - Tiempo de CPU y espacio de RAM.
  - Ocupación de buses de E/S (buses de E/S, NICs, HBAs, etc).
- Permitir **fallos** de hardware no fatales (en estado degradado) en servidores.
- Cuando se necesita **apagar** o mantener el servidor.
- Cuando se **añaden** nuevos servidores físicos.
- **Recuperación** de caídas de sistemas.
- **Balanceo** de cargas entre servidores
- **Green** Computing.
- …