XenServer ofrece un seguimiento detallado de las métricas de rendimiento, incluyendo CPU, memoria, disco, red, información de estados C/P y almacenamiento.

En su caso, estas cifras están disponibles en una base de datos almacenada localmente por host y un por VM. Estas métricas están disponibles directamente, o se puede acceder y ver gráficamente en XenCenter o consultando desde una herramienta de terceros.

Los administradores y clientes pueden monitorizar los host de XenServer y sus máquinas virtuales utilizando métricas del sistema de base de datos Round Robind (RRDS) . Estas métricas se pueden consultar tanto desde http como haciendo uso de la herramienta **RRD2CSV** , además de los gráficos generados por XenCenter, el cual es apoyado sobre RRDS.

**Nota**: Podemos obtener una lista de métricas disponibles en el manual de Administración de XS6.2. Sección 9.1.1 y 9.1.2.

**Herramientas necesarias**

RRDTool ; wget ; XML editor. (con versiones windows/mac/Linux)

SDK no es necesario para el alcance de este post.

**Analizando métricas**

XenCenter permite ver una representación gráfica de algunas de las métricas de XenServer, de forma sencilla en periodos que comprenden desde los 10 min hasta 1 año.

Existen, pero muchas métricas que no aparecen en esta sección gráfica. Ello no significa que no exista la métrica, únicamente que no esta activa para su consulta.

Como comentamos anteriormente, existen una seria de métricas disponibles, documentadas en las secciones 9. del manual de administración. Adicionalmetne, disponemos de un Suplemental Pack (descargable a parte) que introduce métricas adicionales de monitorización.

Es importante tener en cuenta el retorno del comando de consulta de métrica. Cada métrica consultada proporciona:

- Descripción completa de la métrica
- Unidades aplicadas a la métrica
- Rango de valores posibles

Para ver una lista de las métricas disponibles en nuestro host ejecutaremos:

xe host-data-source-list host=

Igual podemos obtener sobre una máquina virtual corriendo en nuestro host.

xe vm-data-source-list vm=

Estas métricas pueden habilitar-se (si no están) o des-habilitarse si no nos son necesarias. Para ello, seguiremos los siguientes pasos:

xe host-data-source-record data-source= host=

Para des-habilitarlos utilizaremos:

xe host-data-source-forget data-source= host=

**Nota**: Por defecto las métricas “tipo” incluidas en XS vienen activas por defecto. Tras instalar el Supplemental Pack, si queremos disponer de las mismas, deberemos habilitar-las.

**RRDS (Round Robin Database)**

XenServer almacena las métricas del sistema en una base de datos local. RRDS. Consiste en múltiples ficheros Round Robin (RRA) fijados en una base de datos.

El funcionamiento de RRDS es el siguiente: se trata la base de datos como si fuese un círculo, sobrescribiendo los datos almacenados con anterioridad una vez alcanzada la capacidad máxima de la misma. Esta capacidad máxima dependerá de la cantidad de información que se quiera conservar como historial.

Cada fichero en la base de datos contiene una métrica particular con una granularidad de tiempo especifica:

- Cada 5 segundos en los últimos 10 min. (RRA 0)

- Cada mínuto en las dos últimas horas (RRA 1-3)

- Cada hora en la última semana (RRA 4-6)

- Cada 24h durante el último año. (RRA 7-9)

Además, RRAs utilizar funciones de consolidación (CF). Entre ellas, son soportados los datos de tipo:

- Average (media)

- MIN

- MAX

Existen RRDS para cada VMs individual, asi como el propio dom-0. VMs RRDs son almacenadas en el host dónde corren, o en el Pool Master si estas están paradas.

Es importante conocer el indice de cada RRA y a que corresponde . Por ejemplo, imaginemos que queremos analizar la media de CPU utilizada en intervalos de 1 minuto. Con la información anterior, sabemos que debemos extraer las métricas del fichero RRA 1.

RRA 0 es llamado **PDPs** (Primary Data Points) que reflejan el valor actual de la métrica obtenida cada 5 segundos. El resto de RRAs son llamados CDPs (Consolidated Data Points) basados en la consolidación de los datos de los PDPs según los CF (Consolidation Function) aplicados al PDPs.

Es importante entender este ciclo, ya que si queremos obtener métricas por minuto de las últimas 24h, eso no será posible, y deberemos sacar metricas cada 2h para obtener la medición de 10 min. que deseamos y poder presentar los resultados necesarios.

**CF – Consolidated Functions**

Hemos visto en la sección anterior, los llamados CF. Es importante también entender el significado de los CF cuando estamos utilizando datos RRD para el seguimiento del rendimiento de nuestro sistema.

Cuando se analizan las métricas de nuestro sistema, es importante prestar atención a los valores de los DRR con granularidad limitada.

**Utilizando http para la consulta RRDS.**

RRDS puede ser descargado vía http, utilizando el controlador registrado http en host_rrd o vm_rrd. Ambas direcciones requieren de autenticación, ya sea http o proporcionada a traves de la API como un argumento de consulta.

wget http:///host_rrd?session_id=OpaqueRef:>

Dónde Server es la IP del servidor y SESSION HANDLE la métrica de consulta. Por ejemplo:

http://:@192.168.1.240/host_rrd?session_id=OpaqueRef:runstate_fullrun

Podemos obtener todas las métricas desde una fecha determinada utilizando la siguiente consulta.

http:///rrd_updates?start=1234567890

Esta consulta recupera las actualizaciones de métricas, con el fin de no obtener toda la bd de métricas.

Con el parámetro Start=xxx indicamos que nos de las métricas a partir de esta fecha y hora (mostrado en segundos desde el 1 de enero 1970). Obtendremos métricas desde la fecha indicada en formato XML RRD Xport, que contiene las métricas de cada VM (si no se especificaron métricas especificas para una VM determinada, como veremos a continuación).

Nota: para obtener las actualizacion de un host, debe ser incluido el parámetro “host=true”.

http:///rrd_updates?start=123456790&host=true

Para sacar todas las métricas rrd de nuestro host, podemos ejecutar:

http://:@/host_rrd

Podemos consultar las métricas de una VM mediante el UUID de la misma.

http://:@/vm_rrd?uuid=

**Importante:** RRD_Updates solo devolverá aquellos datos que estan siendo recogidos actualmente. VM_rrd le puede proporcionar datos historicos de una VM apagada, pero no podremos obtener datos con rrd_updates al no poder ser recogidos de forma inmediata.

http:///rrd_updates?session_id=OpaqueRef:&Start=1234567890&cf=AVERAGE

**RRD_Updates** proporcionará el archivo mas adecuadoa a su solicitud. Tal como indicamos anteriormente hay 4 tipos de ficheros de bd y 3 Cfs. Cuando especificamos el parametro Start indicamos desde cuando queremos que se nos suministren los datos y en base a esa definición

el sistema nos devolvera de forma automatica el tipo de archivo mas acertado.

## Obteniendo datos desde CLI

Igual que podemos obtener todas las métricas históricas en tiempo real y especificas, podemos obtener datos RealTime desde la linea de comandos y obtener y redirecionar a CSV que luego podremos editar.

Como vimos anteriormente, con xe [vm|host]-data-source-list podemos obtener el listado de métricas y sus datos. Utilizando este listado podemos consultar cualquier métrica que sea necesaria.

xe host-data-source-query host= data-source=memory_target

o para un vm

xe vm-data-source-query vm= data-source=

Por supuesto, podemos incluir estas consultas en un script que nos permitan consultar o monitorizar datos de nuestro servidor.

Por ejemplo, veamos un pequeño script que nos proporciona métricas de host especificas cada x tiempo.

while true ; do

sleep 5

xe host-data-source-query host= data-source=cpu_avg

done

El cual nos mostrará la métrica media de uso de cpu en el host, cada 5 segundos.

Scripts como este, nos permitirán observar en tiempo real métricas de nuestro servidor así como almacenar las mismas en ficheros que posteriormente nos servirán para analizar en caso que sea necesario.

**Utilizando RRDs**

Volvemos a la monitorización por RRDs. Como hemos visto antes, podemos obtener fichero XML con la representación de RRDs que es de nuestro interés.

Y que hacemos ahora? Podemos utilizar herramientas como rdtool que nos permitirán generar gráficos adecuados para las métricas obtenidas, o analizar, si así lo deseas los ficheros de forma manual.

Para el análisis de los ficheros obtenidos, existe una biblioteca de Python (parse_rrd) para el tratamiento de los ficheros RRDs.

**Un vistazo a RRDTool**

RRDTool es una herramienta que permite la representación gráfica y análisis de los datos en serie obtenidos por RRDS. RRDTool puede ser integrado fácilmente en Scripts y en aplicaciones perl, python, ruby, lua o tcl. Es un re-implementación del conocido MRTG.

RRDtool esta disponible para plataformas Windows/Unix en el siguiente link: http://www.mrtg.org/rrdtool/download.en.html

Rddtool no solo genera gráficos, si no, que nos permite el análisis de los ficheros RRDs obtenidos.

Consulta de datos: **rrdtool fetch file.rrd AVERAGE**

El retorno <**nan**> indica que no existe métrica en ese punto del tiempo. Utilizar rrdtool para el análisis puede ser interesante, pero lo que queremos en este punto, es generar una gráfica entendible del estado actual.

Es sin duda la característica mas potente de la herramienta. No cabe indicar, que disponemos siempre de la ayuda en linea “man” para la obtención de detalles.

**Generando gráficas con RRDTool.**

La sintaxis para generar un gráfico con rrdtool es la siguiente: rrdtool graph. Con “man rddgraph” podemos obtener ayuda sobre el uso del comando.

Lo primero que debemos tener es una serie de datos en variables, las cuales definimos a continuación:

**rrd**: path de la base de datos rrd

**last**: tiempo unix del último registro añadido

**file1**: fichero de salida utilizado (.png)

**fecha**: fecha actual para la generación de la gráfica

**Width** y **height**: Anchura y Altura de la gráfica.

Todas estas variables serán pasadas a nuestro rrdtool para la generación de la gráfica. Disponemos, pero, de otros parámetros del comando necesarios, que definiremos en ejecución del comando en lugar de usar con variable.

**Start**: tiempo

**DEF**: <>

Nota: La especificación del tiempo de inicio podemos utilizar un valor negativo, de esta forma podemos usar **-1day** o **-7day** para indicar datos de hace un día o una semana respectivamente.

Adicionalmente, tenemos el parámetro DEF, el mas complicado de entender, pero a priori el mas importante. Este parámetro nos permite definir un nombre “virtual” para una fuente de datos que podremos utilizar con rrdtool.

DEF:vname=rrd:ds-name:CF

vname: nombre ficticio que le damos a la consulta.

rrd: fichero rrd.

ds-name: nombre que le hemos dado en la creación de la bd rrd.

CF: funcion de consolidación que utilizaremos.

Nota: recordad que los CF hacen referencia al tipo (AVERAGE, MIN, MAX).

Con estas lineas ya podriamos crear nuestra gráfica, pero rrdtool nos permite anexionar comentarios entre otras opciones, de las que destacamos.

**COMMENT: “comentarion”**

GPRINT: escribe datos debajo de la gáfica. Su sintaxys es: **GPRINT:vnmae:CF:format**

LINE: definición de las lineas graficas, o lo que nos ayuda a especificar como serán las lineas. Su sintaxys: **LINE{1|2|3}:vname[#rrggbb_code[:legend]]**

LINE1 crea una linea mas gruesa que LINE2.

**Un ejemplo**

Vamos a ver un ejemplo práctico de cómo poner esto en practica. Mas vale un ejempo que mil palabras dicen.

En primera instancia vamos a obtener datos desde nuestro servidor de XenServer. Como vimos anteriormente, vamos a obtener un XML con todos los datos de nuestras metricas RRD.

wget http://user:password@host_ip/host_rrd

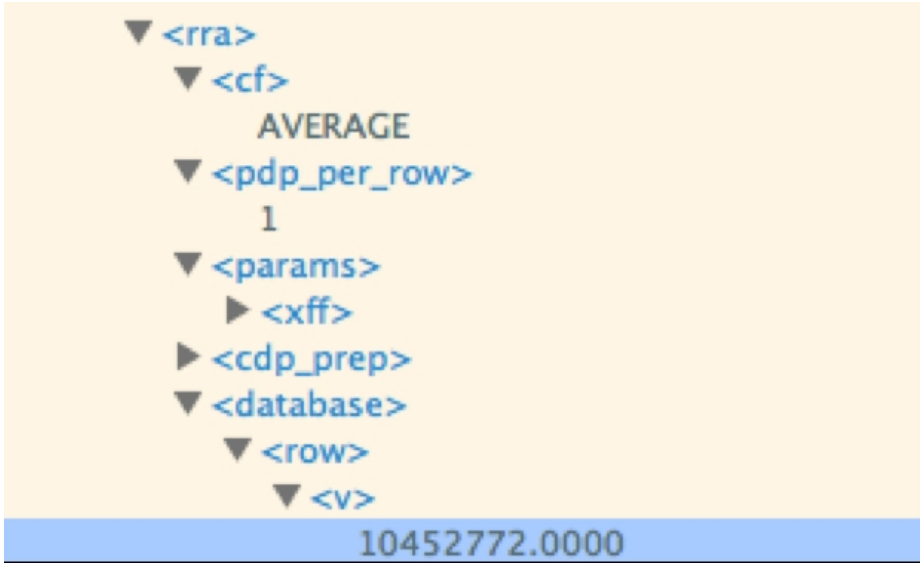
Ya tenemos nuestro fichero XML extraido del log RRD. Podemos editar el fichero con XMLEditor para ver que contiene y los respectivos tags del mismo, asi como valores especificos de recursos.

Estas metricas obtenidas son almacenados en datasets () dentro de una estructura XML. Dependiendo de los componentes de nuestro sistema (numero de Cpus, de Nics, etc) dispondremos de mas o menos datasets. Veamos un ejemplo de dataset.


El inicio del fichero contiene un TimeStamp con la fecha de colección de los datos.

Ademas cada DS contiene un tipo: GAUGE o DERIVE. Gauge hace referencia a valores absolutos. DERIVE es una comparativa entre dos mediciones. Por ejemplo, %CPU es siempre un valor DERIVE así como el número de KB de memoria es una valor GAUGE.

Tras los DS, veremos unos tags RRA. Anteriormente hablamos de los RRA, cada uno de los RRA posteriores, es referente a los tipos RRA0,1,2,3…




Las comprobaciones y revisiones desde el XML son muy confusas, por eso vamos a generar gráficos para ayudarnos a entender estos datos. Es importante en este caso, conocer el contenido del campo “name” de los datos que queremos analizar. Por ejemplo, para el analisis de la recepción de paquetes en la interface 0, deberemos usar el contenido englobado en pif_eth0_rx .

Posteriormente, vamos a convertir el XML en un base de datos RRD para su uso con RRDtool. Para ello ejecutaremos:

rrdtool restore filexml file.rrd

Ahora, ya disponemos de un fichero RRD con el que poder trabajar.

Podemos ver el cotenido con el comando.

rrdtool fetch file.rrd AVERAGE

Determinamos el inicio/fin: Para determinar el intervalo de datos que queremos mostrar en la gráfica primero deberemos determinar los tiempos para el RRA que queremos analizar. Los siguientes comandos van a mostrarnos el último punto de datos grabado en el RRD y el tiempo del primer registro que ha sido guardado en el segundo RRA en una notación de tiempo UNIX.

rrdtool last host_rrd.rrd

rrdtool first host_rrd.rrd --rraindex 1

```
hack:~ cristiansan$ rrdtool last host_rdd.rrd
1406277829
hack:~ cristiansan$ rrdtool first --rraindex 1 host_rdd.rrd
1406270640
```

**Generando el gráfico**

Ya vimos anteriormente como generar el gráfico, ahora, es momento de ponerlo en practica.

**Definiendo DataSets**

Lo primero que debemos hacer es definir el dataset (ds) que queremos que sea mostrado utilizando la propiedad Def de RRDTool.

DEF:CPU00_MAX=host_rrd.rrd:cpu0:MAX

Ahora aclaremos y entendamos esta linea.

DEF:CPU00_MAX= :: En esta linea definimos una variable CPU00_MAX a la que asignamos el valor de lo siguiente:

Host_rrd.rrd :: Es el nombre de la base de datos RRD generada que será consultada.

:cpu0 :: Es el nombre del set que asignaremos a la variable. Anteriormente indicamos que lo importante era conocer el  del “contador” a consultar. Este es el parametro .

MAX :: El CF utilizado identificado dentro del RRA.

Todo ello define CPU00_MAX. Con lo que podemos definir adicionalmente variaciones de la misma.

DEF:CPU00_AVG=host_rrd.rrd:cpu0:AVG

DEF:CPU00_MAX=host_rrd.rrd:cpu0:MAX

DEF:CPU00_MIN=host_rrd.rrd:cpu0:MIN

O sobre otras CPUs (si existen; comprobar en XML).

DEF:CPU00_AVG=host_rrd.rrd:cpu0:AVG

DEF:CPU01_AVG=host_rrd.rrd:cpu1:AVG

DEF:CPU02_AVG=host_rrd.rrd:cpu2:AVG

Ya tenemos los conjuntos de datos a definir. Ahora necesitamos donde empieza y dónde acaba la toma de datos que deber ser mostrada en la gráfica.

Si queremos generar una gráfica que contenga todos los datos del RRA consultado, utilizar la fecha de inicio y final obtenida del comando rrdtool “last” & “First” que comentamos anteriormente.

Así pues, nuestra sintaxis para el tiempo será definida como:

--start

--end

**Nota**: Podemos definir ventanas de tiempo menores dentro de esos dos valores. Unicamente debemos tener cuidad de que los valores indicados no engloben dos ficheros RRA.

**Definir las características del gráfico.**

Ya tenemos lo dificil. Ahora toca pintar :D y para pintar, debemos saber como. Tipo de linea, color, marco, tamaño del lienzo, etc.

Así pues ahora viene lo divertido. Pintemos!!! RRDtool puede crear lineas o areas para graficos. (un area comprende una linea con todo el campo inferior pintado completo)

Para ello utilizaremos el parametro LINE. Una linea comprende de varios punto. Incluye, un tamaño, un color y una leyenda. Por ejemplo, vamos a crear una linea doble Azul que incluira la leyenda CORE0 para el dato max. de cpu0. Y una linea simper verde que incluya datos para la cpu1.

LINE2:CPU00_MAX#0033FF:Core0

LINE1:CPU01_MAX#00CC66:Core1

Para finalizar, únicamente nos queda definir el Lienzo, que incluirá datos como Alto, ancho, limites de los indices, y un título.

--weidth 1024

--height 786

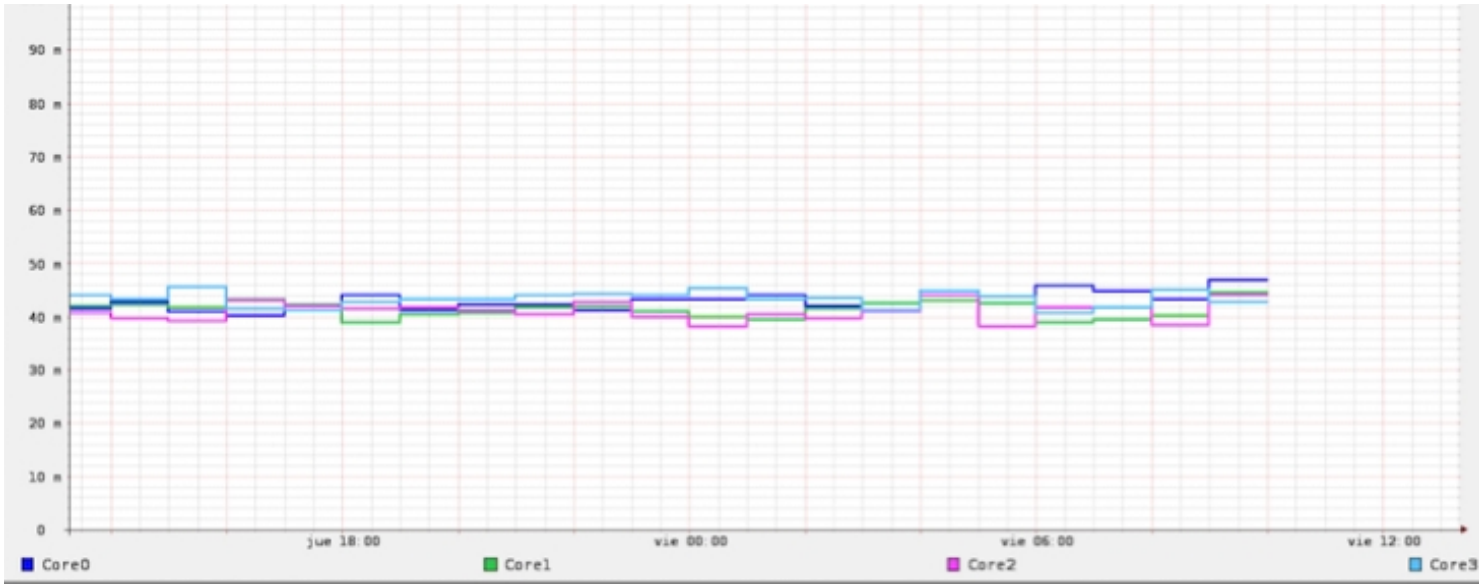
--uper-limit 100

--lower-limit 0

Y con eso ya tendremos nuestro grafico. Ahora simplemente es necesario incluir todo en una única linea mediante el uso de RRDtool. Podemos usar un fichero de texto dado el tamaño del comando y ejecutar a posterior.

rrdtool graph migrafica.png DEF:CPU00AVG=host_rdd.rrd:cpu0:AVERAGE
DEF:CPU01AVG=host_rdd.rrd:cpu1:AVERAGE
DEF:CPU02AVG=host_rdd.rrd:cpu2:AVERAGE
DEF:CPU03AVG=host_rdd.rrd:cpu3:AVERAGE LINE2:CPU00AVG#0033FF:Core0
LINE2:CPU01AVG#00CC66:Core1 LINE2:CPU02AVG#FF66FF:Core2
LINE2:CPU03AVG#66CCFF:Core3 --width1024 --height 786 --upper-limit 0.2 --lower-limit 0

Y ejecutamos para obtener nuestro gráfico.




**Otra vuelta de tuerca**

Esto nos permitira generar nuestras gráficas y disponer de nuestros scripts para generar nuestras graficas a petición cuando sean necesarias. Pero puede que en ocasiones queramos valores mas compresinbles (como vimos el XML es bastante tedioso) que poder exportar para generar nuestras propias gráficas mas elaboradas, como es el caso de usar Excel o Numbers. Para ello siempre podemos realizar capturas de datos en tiempo real y exportar-los a CSV...

**Creando CSV**

XenServer nos proporciona otra herramienta interesante. Rrd2csv. Esta nos permite crear ficheros CSV desde fichero RRD.

Estos son explicados en los post: **Playing with Counters**: Part1 , Part2 y Part3

**Cacti**

“Cacti es una completa solución para la generación de gráficos en red , diseñada para aprovechar el poder de almacenamiento y la funcionalidad para gráficas que poseen las aplicaciones RRDtool .
Esta herramienta, desarrollada en
PHP
, provee un pooler ágil,
plantillas
de gráficos avanzadas, múltiples métodos para la recopilación de datos, y manejo de usuarios.
Tiene una
interfaz
de usuario fácil de usar, que resulta conveniente para instalaciones del tamaño de una
LAN
, así como también para
redes
complejas con cientos de dispositivos.” - Wikipedia

Podemos obtener Cacti del siguiente Link: http://www.cacti.net/download_cacti.php

Disponemos de version Windos, Unix y para varias distribuciones Linux, y para instalar-lo en Mac, basta con ejecutar nuestro “port”.

**Recursos:**

Rdtool: http://www.mrtg.org/rrdtool/index.en.html

Rrdxport: http://www.mrtg.org/rrdtool/doc/rrdxport.en.html

Parse_rrd & SDK: http://www.xenserver.org/partners/developing-products-for-xenserver.html

Bulma: http://bulma.net

XenServer Manual: http://support.citrix.com/servlet/KbServlet/download/34969-102-704897/reference.pdf

Calc. RGB Values: http://www.calculatorcat.com/free_calculators/color_slider/rgb_hex_color_slider.phtml

Rrd2csv: http://support.citrix.com/article/CTX135033

Cacti: http://www.cacti.net/