Artículo
Mario Sanchez Macias · 16 ene, 2020 Lectura de 9 min

Plataformas de datos de InterSystems y su rendimiento - Parte 1

¡Hola Comunidad!

Voy a empezar a traducir los artículos de mi compañero Murray sobre rendimiento, pues son muy interesantes y merece la pena echarles un vistazo. Aprovecho para comentar que si tenéis preguntas y /o problemas de rendimiento os animéis a escribir y entre todos trataremos de ayudaros. Empecemos...

Tu aplicación está implementada y todo funciona bien. ¡Genial, enhorabuena! Pero, de repente, el teléfono empieza a sonar sin parar: son usuarios que se quejan de que la aplicación a veces se vuelve "lenta". Pero... ¿qué quiere decir eso? ¿A veces? ¿De qué herramientas dispones y a qué estadísticas deberías prestar atención para encontrar y resolver la lentitud? ¿La infraestructura de tu sistema está a la altura de la carga de usuarios? ¿Qué preguntas de diseño de infraestructura deberías haber hecho antes de haber pasado a producción? ¿Cómo puedes hacer la planificación de capacidad para nuevo hardware con confianza y sin gastar en más hardware del necesario? ¿Cómo hacer que el teléfono deje de sonar? ¿Cómo podrías haber evitado que sonara desde un principio?

Aquí puede ver un listado con otros artículos de esta serie >>

Esto será un largo viaje

Este es el primer artículo de una serie que explorará las herramientas y métricas disponibles para supervisar, analizar y resolver problemas de rendimiento de los sistemas, así como consideraciones de diseño de arquitectura y sistemas que afectan el desempeño. Por el camino nos desviaremos por varias ramas para entender el desempeño de Caché, los sistemas operativos, el hardware, la virtualización y otras áreas que cobren relevancia en base a los comentarios que vayáis haciendo.

Seguiremos el bucle de retroalimentación por el cual los datos de rendimiento permiten ver las ventajas y limitaciones de las aplicaciones y la infraestructura implementada, para finalmente volver a mejorar el diseño y la planificación de capacidad.

No debería ser necesario aclarar que las métricas de rendimiento se deben revisar constantemente. Es increíble la cantidad de veces que nuestros clientes se ven sorprendidos por problemas de rendimiento que se podrían haber detectado mucho antes, si hubieran prestado atención a los datos. Pero, claro, que la pregunta es: ¿qué datos? Comenzaremos el viaje recolectando algunas métricas básicas de Caché y del sistema, para poder tener una impresión general de la salud actual del sistema. En posteriores publicaciones analizaremos el significado de las métricas clave.

Hay muchas opciones disponibles para la monitorización del sistema, desde dentro de Caché y desde fuera, y examinaremos muchas de ellas en esta serie

Para empezar, veremos mi herramienta favorita para recogida continua de datos, que ya viene instalada en cada sistema Caché: ^pButtons.

Para asegurarte de que cuentas con la última versión de pButtons, por favor consulta la siguiente publicación:

https://community.intersystems.com/post/intersystems-data-platforms-and-performance-%E2%80%93-how-update-pbuttons

Recogida de métricas de rendimiento del sistema - ^pButtons

pButtons de Caché genera un informe de rendimiento HTML legible, a partir de los archivos de registro que crea. Las métricas de rendimiento generadas por pButtons se pueden extraer, representar mediante gráficos y analizar sin dificultad.

Algunos datos recogidos en el archivo html de pButtons son:

- Implementación de Caché: con configuración, drive mappings, etc.
- mgstat: métricas de rendimiento de Caché - la mayoría de los valores son promedios por segundo
- Unix: vmstat e iostat: métricas de rendimiento y recursos del sistema operativo
- Windows: performance monitor: métricas de rendimiento y recursos de Windows
- Otras métricas de utilidad

La recogida de datos de pButtons tiene muy poco impacto sobre el rendimiento del sistema, porque las métricas ya son recopiladas por el sistema. pButtons simplemente las empaqueta para facilitar su archivado y transporte.

Para el análisis de tendencias y la resolución de problemas, una buena práctica puede ser recoger un pButtons de 24 horas cada día (de medianoche a medianoche), para tener un ciclo completo. Un ciclo podría ser un mes o más, por ejemplo para capturar datos desde el procesamiento de fin de mes. Si no se cuenta con otro sistema externo de recogida o supervisión de rendimiento, se puede usar pButtons todo el año.

Hay que tener en cuenta los siguientes puntos clave:

- Cambia el directorio del registro a una ubicación separada de los datos de producción, para almacenar los archivos de salida acumulados y evitar discos llenos de problemas
- Ejecuta un script de sistema operativo o comprima de alguna otra forma, y archive frecuentemente el archivo pButtons. Esto es de particular importancia en Windows, ya que estos archivos pueden ser grandes
- ¡Revisa los datos con frecuencia!

En el caso de un problema que requiera de un análisis inmediato, puedes previsualizar los datos de pButtons (recogidos inmediatamente), mientras las métricas se siguen almacenando para ser recogidas al final del día.

Para más información sobre pButtons, incluyendo la vista previa, detener una ejecución y agregar una recogida de datos personalizada, consulte la "Guía de monitarización de Caché" en la documentación de Caché más reciente:

http://docs.intersystems.com

Los datos del archivo HTML de pButtons se pueden separar y extraer (por ejemplo a archivo CSV) para procesar y generar gráficas u otros análisis mediante scripts o simplemente copiando y pegando. Más adelante en el siguiente artículo veremos ejemplos de gráficos.

Por supuesto, si surgen problemas de rendimiento urgentes, debe contactar con el Centro de Soporte Internacional (WRC).

Programar una recogida de datos de pButtons de 24 horas

^pButtons se puede iniciar manualmente desde un terminal o se puede programar. Para programar una recogida diaria de 24 horas:

1. Inicia un terminal de Caché, pase al namespace %SYS y ejecute manualmente pButtons una vez para configurar las estructuras de archivos pButtons:

```
%SYS>d ^pButtons Current log directory: /db/backup/benchout/pButtonsOut/
Available profiles:
     1  12hours     - 12 hour run sampling every 10 seconds
     2  24hours     - 24 hour run sampling every 10 seconds
     3  30mins      - 30 minute run sampling every 1 second
     4  4hours      - 4 hour run sampling every 5 seconds
     5  8hours      - 8 hour run sampling every 10 seconds
     6  test        - A 5 minute TEST run sampling every 30 seconds
```

Selecciona la opción 6. para la prueba, una ejecución de TEST de 5 minutos con muestreo cada 30 segundos. Tenga en cuenta que la numeración podría ser distinta, pero el resto debería ser obvio.

Durante esta ejecución, ejecute un Collect^pButtons (como se muestra abajo). Verá la información, incluyendo el runid. En este caso, " 20160303_1851_test " .

```
%SYS>d Collect^pButtons
Current Performance runs:
    20160303_1851_test
        ready in 6 minutes 48 seconds nothing available to collect at the moment.
%SYS>
```

Ten en cuenta que a esta ejecución de 5 minutos le quedan 6 minutos y 48 segundos (?) pButtons agrega un período de gracia 2 minutos a todas las ejecuciones para permitir un tiempo para la recogida y compilación de los registros en formato html.

2. ¡IMPORTANTE! Cambia el directorio de salida de los registros pButtons. La ubicación de salida predeterminada es la carpeta <cache install path>. Por ejemplo, en Unix, la ruta al directorio de registros podría verse así:

```
do setlogdir^pButtons("/somewhere_with_lots_of_space/perflogs/")
```

Asegúrese de que Caché tenga permisos para el directorio y de que haya suficiente espacio en disco disponible para acumular los archivos de salida.

3. Crea un nuevo perfil de 24 horas con intervalos de 30 segundos. Ejecute lo siguiente:

```
write $$addprofile^pButtons("My_24hours_30sec","24 hours 30 sec interval",30,2880)
```

Verifica que el perfil se haya agregado a pButtons:

```
%SYS>d ^pButtons
Current log directory: /db/backup/benchout/pButtonsOut/
Available profiles:
     1  12hours     - 12 hour run sampling every 10 seconds
     2  24hours     - 24 hour run sampling every 10 seconds
     3  30mins      - 30 minute run sampling every 1 second
     4  4hours      - 4 hour run sampling every 5 seconds
     5  8hours      - 8 hour run sampling every 10 seconds
     6  My_24hours_30sec- 24 hours 30 sec interval
     7  test        - A 5 minute TEST run sampling every 30 seconds
select profile number to run:
```

Nota: puedes variar el intervalo de recogida. 30 segundos es adecuado para una supervisión de rutina. Yo no bajaría de 5 segundos para una ejecución de rutina de 24 horas (… " ,5,17280), ya que los archivos de salida

pueden volverse muy grandes cuando pButtons recoge datos en cada intervalo. Si está intentando resolver un problema en un momento específico del día y necesita datos más detallados, use uno de los perfiles predeterminados o cree un nuevo perfil personalizado con un período más breve, por ejemplo 1 hora con intervalos de 5 segundos (··· " ,5,720). Puede ejecutar múltiples pButtons a la vez, así que podría tener pButtons más cortos con un intervalo de 5 segundos ejecutándose al mismo tiempo que los pButtons de 24 horas.

4. Consejo Para sitios UNIX, consulte el parámetro "disk". Puede que los parámetros predeterminados usados con el comando "iostat" no incluyan los tiempos de respuesta de disco. Primero muestre los comandos de disco actualmente configurados:

```
%SYS>zw ^pButtons("cmds","disk")
^pButtons("cmds","disk")=2
^pButtons("cmds","disk",1)=$lb("iostat","iostat ","interval"," ","count"," > ")
^pButtons("cmds","disk",2)=$lb("sar -d","sar -d ","interval"," ","count"," > ")
```

Para recoger estadísticas de disco, use el comando adecuado para editar la sintaxis de su instalación de UNIX. Tenga en cuenta el espacio final. Estos son algunos ejemplos:

```
LINUX:     set $li(^pButtons("cmds","disk",1),2)="iostat -xt "
AIX:       set $li(^pButtons("cmds","disk",1),2)="iostat -sadD "
VxFS:      set ^pButtons("cmds","disk",3)=$lb("vxstat","vxstat -g DISKGROUP -i ","int
erval"," -c ","count"," > ")
```

Se pueden crear archivos html de pButtons muy grandes si deja ejecutándose tanto el comando iostat como el sar. Para análisis de rendimiento habituales, generalmente uso solo iostat. Para configurar solo un comando:

```
set ^pButtons("cmds","disk")=1
```

Puede encontrar más detalles sobre la configuración de pButtons en la documentación online.

5. Programa pButtons para que se inicie a medianoche en `Management Portal > System Operation > Task Manager:`

```
Namespace: %SYS
 Task Type: RunLegacyTask
 ExecuteCode: Do run^pButtons("My_24hours_30sec")
 Task Priority: Normal
 User: superuser
 How often: Once daily at 00:00:01
```

Recogida de datos de pButtons

La versión de pButtons incluida en las versiones más recientes de las plataformas de datos InterSystems incluyen recogida automática. Para recoger y compilar manualmente los datos en un archivo html: en el namespace %SYS, ejecute el siguiente comando para generar cualquier archivo de salida html de pButtons pendiente:

```
do Collect^pButtons
```

El archivo html estará en el logdir que configuraste en el paso 2 (si no lo hiciste, **¡ hazlo ahora!**). Si no, la ubicación

predeterminada es <Caché install dir/mgr>

Los archivos se nombran <hostnameinstanceNamedatetimeprofileName.html> por ejemplo vsan-tc-db1H20152016021802 55test.html

Consideraciones sobre Windows Performance Monitor

Si el sistema operativo es Windows, entonces puede usar Windows Performance Monitor (perfmon) para recoger datos de forma sincronizada con las otras métricas recolectadas. En distribuciones viejas de pButtons, deberás configurar perfmon manualmente en Windows.

Si resultara útil, puedo escribir una publicación sobre cómo crear una plantilla de perfmon para definir los contadores de rendimiento para supervisar y programar una ejecución para el mismo período e intervalo que pButtons.

Resumen

Con estos pasos, podemos empezar a recoger datos para analizarlos y ver qué significan.

http://docs.intersystems.com

#Administración del sistema #Rendimiento #Caché #InterSystems IRIS #InterSystems IRIS for Health

URL de fuente:https://es.community.intersystems.com/post/plataformas-de-datos-de-intersystems-y-su-rendimiento-parte-1