Artículo
Joel Espinoza · 27 feb, 2020 Lectura de 15 min

Crear OID's SNMP personalizados

Hola Comunidad:

Esta publicación está dedicada a la tarea de supervisar una instancia de Caché usando SNMP. Algunos usuarios de Caché probablemente ya lo hacen de una u otra forma. Hace ya mucho que el paquete de Caché estándar soporta la supervisión por SNMP, pero no todos los parámetros necesarios vienen "listos para usar". Por ejemplo, sería bueno poder supervisar el número de sesiones CSP, obtener información detallada sobre el uso de la licencia, KPI particulares del sistema en uso, etc. Después de leer este artículo, sabrás cómo agregar tus parámetros a la supervisión de Caché mediante SNMP.

Lo que ya tenemos

Es posible supervisar Caché con SNMP. En los archivos de <Installdir>/SNMP/ puedes encontrar una lista completa de todo lo soportado. Aquí encontrarás 2 archivos: ISC-CACHE.mib y ISC-ENSEMBLE.mib. Nos interesa el archivo destinado a Caché — ISC-CACHE.mib. En particular, quisiéramos saber qué información podemos obtener sobre licencias y sesiones. La tabla contiene los OID correspondientes, siempre que la jerarquía comience desde la raíz de InterSystems: 1.3.6.1.4.1.1656

OID	Nombre	Descripción
.1.1.1.1.10	cacheSysLicenseUsed	El número actual de licencias en us esta instancia de Caché
.1.1.1.1.11	cacheSysLicenseHigh	La cota máxima para las licencias us en esta instancia de Caché
.1.2.16	cacheLicenseExceed	Una solicitud de licencia ha superad licencias disponibles o permitida
.1.1.1.1.6	cacheSysCurUser	Número actual de usuarios en est instancia de Caché

Al paquete le faltan muchos parámetros importantes, como el número de sesiones CSP, información de licencia y, por supuesto, no tiene KPI específicos por aplicación.

Esto es un ejemplo de lo que nos gustaría saber:

- El número de usuarios CSP
- Limitations of our license in terms of the user count;
- License expiry date.

Podemos añadir también unos pocos parámetros para el análisis de desempeño. Los parámetros en sí están en el paquete, pero queremos conocer su incremento por minuto. Por ejemplo:

- El aumento del número de referencias “ globales ” por minuto
- La cantidad de comandos de ejecución por minuto
- La cantidad de llamadas de rutina por minuto

Cómo agregar “ tus ” parámetros

Puedes consultar el documento "Monitoring Caché using SNMP" .

La versión de Caché de nuestra instancia de prueba (CACHE2016) es 2016.2.0.721.0. El sistema operativo es Linux Fedora 24 (Workstation Edition). La instalación de Caché en Linux OS se describe al detalle aquí.

Este es nuestro plan:

1. Crear una clase para recolectar métricas
2. Registrar y activar una nueva clase en Caché usando ^%SYSMONMGR
3. Crear una MIB de usuario usando métodos de la clase MonitorTools.SNMP. Usaremos 99990 como PEN (Private Enterprise Number) temporal, pero necesitaremos registrarnos con IANA después. Este proceso es gratuito, lleva una semana o dos y requiere un intercambio de correos electrónicos acerca de "¿Para qué necesita su propio PEN?"
4. Comenzar un servicio de supervisión con un subagente de Caché conectado
5. Usar snmpwalk para asegurarse de tener acceso a todos los OID recién creados
6. Agregar los OID a un sistema de supervisión de otro proveedor. Usemos por ejemplo Zabbix. La documentación de Zabbix está aquí. Asegurémonos de que la supervisión esté funcionando
7. Agregar el inicio de la supervisión del sistema en el namespace TEST a la lista de inicio del sistema

Sigamos ahora el plan, punto por punto:

1. Crear una clase para recolectar métricas

La clase de recolección de métricas extiende %Monitor.Adaptor. En el terminal, cambiamos al namespace %SYS y exportamos la clase oculta Monitor.Sample:

```
%SYS>do $system.OBJ.Export("Monitor.Sample.cls","/tmp/Monitor_Sample.xml")
Exporting to XML started on 02/07/2017 21:39:56
Exporting class: Monitor.Sample
Export finished successfully.
```

Vamos a asumir que el namespace TEST es nuestra área de trabajo. Vamos a pasar a ella e importar aquí la clase Monitor.Sample. Ahora crearemos una clase que describa la implementación de un mecanismo de supervisión para las 6 métricas descritas en la sección "Lo que ya tenemos".

```
Class monitoring.snmp.Metrics Extends %Monitor.Adaptor
{
/// Give the application a name. This allows you to group different
/// classes together under the same application level in the SNMP MIB.
/// The default is the same as the Package name.
Parameter APPLICATION = "Monitoring";
/// CSP sessions count
Property Sessions As %Monitor.Integer;
/// License user limit
Property KeyLicenseUnits As %Monitor.Integer;
/// License key expiration date
Property KeyExpirationDate As %Monitor.String;
/// Global references speed
Property GloRefSpeed As %Monitor.Integer;
/// Number of commands executed
Property ExecutedSpeed As %Monitor.Integer;
/// Number of routine loads/save
Property RoutineLoadSpeed As %Monitor.Integer;
/// The method is REQUIRED. It is where the Application Monitor
/// calls to collect data samples, which then get picked up by the
```

```
/// ^SNMP server process when requested.
Method GetSample() As %Status
{
      set ..Sessions = ..getSessions()
      set ..KeyLicenseUnits = ..getKeyLicenseUnits()
      set ..KeyExpirationDate = ..getKeyExpirationDate()

      set perfList = ..getPerformance()
      set ..GloRefSpeed = $listget(perfList,1)
      set ..ExecutedSpeed = $listget(perfList,2)
      set ..RoutineLoadSpeed = $listget(perfList,3)       quit $$$OK
}
/// Get CSP sessions count
Method getSessions() As %Integer
{
     // This method will only work if we don't use WebAddon:
    // quit $system.License.CSPUsers()
    //
    // This will work even if  we use WebAddon:
    set csp = ""
    try {
        set cn = $NAMESPACE
        znspace "%SYS"
        set db = ##class(SYS.Stats.Dashboard).Sample()
        set csp = db.CSPSessions
        znspace cn
    } catch e {
        set csp = "0"
    }
    quit csp
}
/// Get license user's power
Method getKeyLicenseUnits() As %Integer
{
    quit $system.License.KeyLicenseUnits()
}
/// Get license expiration date in human-readable format
Method getKeyExpirationDate() As %String
{
    quit $zdate($system.License.KeyExpirationDate(),3)
}
/// Get performance metrics (gloref, rourines etc.)
Method getPerformance(param As %String) As %Integer
{
    set cn = $NAMESPACE
    znspace "%SYS"
    set m = ##class(SYS.Monitor.SystemSensors).%New()
    do m.GetSensors()
    znspace cn
    quit $listbuild(m.SensorReading("GlobalRefsPerMin"),
                    m.SensorReading("RoutineCommandsPerMin"),
                    m.SensorReading("RoutineLoadsPerMin"))
}
}
```

Asegúrate de que el método GetSample() realmente recupere los datos necesarios:

```
TEST>set metrics = ##class(monitoring.snmp.Metrics).%New()
```

```
TEST>write metrics.GetSample()
1
TEST>zwrite metrics
metrics=<OBJECT REFERENCE>[2@monitoring.snmp.Metrics]
+----------------- general information ---------------
|      oref value: 2
|      class name: monitoring.snmp.Metrics
| reference count: 2
+----------------- attribute values ------------------
|      ExecutedSpeed = 431584
|        GloRefSpeed = 881
|  KeyExpirationDate = "2017-02-28"
|    KeyLicenseUnits = 100
|   RoutineLoadSpeed = 0
|            Sessions = 1
+-----------------------------------------------------
```

2. Registrar y activar la nueva clase en Caché usando ^%SYSMONMGR

Abre el terminal y pasa al namespace TEST:

```
# csession cache2016 -U test
Node: server, Instance: CACHE2016
TEST>do ^%SYSMONMGR
1. Select item 5, Manage Application Monitor. 2. Select item 2, Manage Monitor Classe
s. 3. Select item 3, Register Monitor System Classes.
Exporting to XML started on 02/09/2017 11:22:57
Exporting class: Monitor.Sample
Export finished successfully.
Load started on 02/09/2017 11:22:57
Loading file /opt/intersystems/cache2016/mgr/Temp/Mb7nvq5xuovdHQ.stream as xml
Imported class: Monitor.Sample
Compiling class Monitor.Sample
Compiling table Monitor.Sample
Compiling routine Monitor.Sample.1
Load finished successfully. 4. Select item 1, Activate/Deactivate Monitor Class
Class??
Num MetricsClassName Activated
1) %Monitor.System.AuditCount N
…
15) monitoring.snmp.Metrics N
Class? 15 monitoring.snmp.Metrics
Activate class? Yes => Yes 5. Select item 6, Exit 6. Select item 6 again, Exit 7. Sel
ect item 1, Start/Stop System Monitor 8. Select item 2, Stop System Monitor
Stopping System Monitor… System Monitor not running! 9. Select item 1, Start System M
onitor
Starting System Monitor… System Monitor started 10. Select item 3, Exit 11. Select it
em 4, View System Monitor State
Component                    State
System Monitor                    OK
%SYS.Monitor.AppMonSensor         OK 12. Select item 7, Exit
```

3. Crear una MIB de usuario

Para crear una MIB de usuario, se usan métodos de clase de MonitorTools.SNMP. Para este ejemplo, usaremos un PEN (Private Enterprise Number) falso, 99990, pero será necesario registrar el PEN con IANA después. Aquí puedes ver los números registrados. Por ejemplo, el PEN de InterSystems es 16563.

```
16563
InterSystems
Robert Davis
rdavis&intersystems.com
```

Usaremos la clase MonitorTools.SNMP y su método CreateMIB() para crear un archivo MIB. Este método lleva 10 argumentos:

Nombre y tipo de argumento	Descripción	
AppName As %String	nombre de la aplicación	Valor del parám metrics.s
Namespace As %String	nuestro namespace	
EntID As %Integer	PEN de la empresa	
AppID As %Integer	OID de aplicación dentro de la empresa	
Company As %String	nombre de la empresa (mayúsculas)	
Prefix As %String	prefijo de todos los objetos SNMP que creemos	
CompanyShort As %String	prefijo corto de la empresa (mayúsculas)	
MIBname As %String	nombre del archivo MIB	
Contact As %String	información de contacto (en particular, dirección)	Dejemos el valo Somewhere in t
List As %Boolean	equivalente a verbose. Mostrar progreso de tarea para el archivo MIB	

Ahora llega la creación del archivo MIB:

```
%SYS>d ##class(MonitorTools.SNMP).CreateMIB("Monitoring","TEST",99990,42,"fiction","f
ict","fiction","ISC-TEST",,1)
Create SNMP structure for Application - Monitoring
     Group - Metrics
          ExecutedSpeed = Integer
          GloRefSpeed = Integer
          KeyExpirationDate = String
          KeyLicenseUnits = Integer
          RoutineLoadSpeed = Integer
          Sessions = Integer Create MIB file for Monitoring
     Generate table Metrics
          Add object ExecutedSpeed, Type = Integer
          Add object GloRefSpeed, Type = Integer
          Add object KeyExpirationDate, Type = String
          Add object KeyLicenseUnits, Type = Integer
          Add object RoutineLoadSpeed, Type = Integer
          Add object Sessions, Type = Integer
MIB done.
```

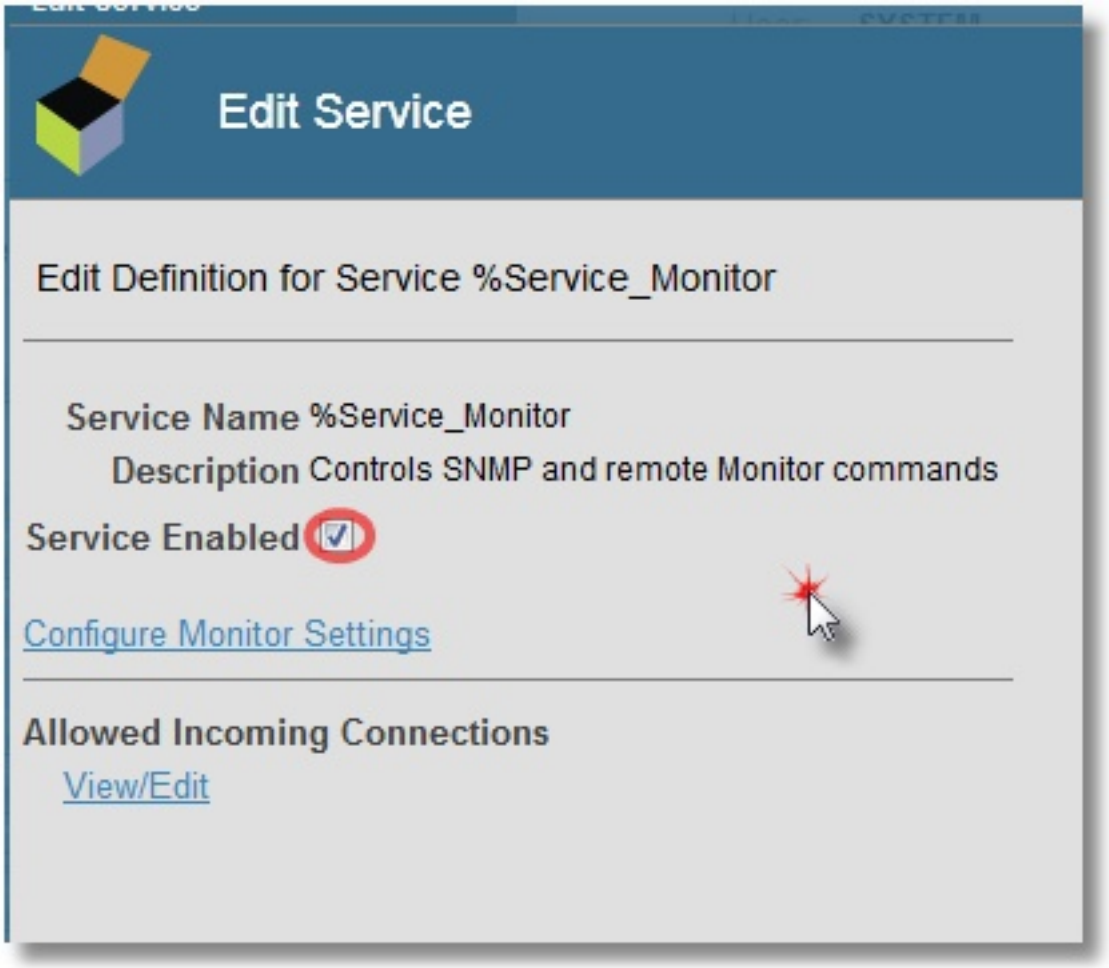
Ahora hay una nueva *MIB ISC-TEST.mib* en la carpeta <Installdir>/mgr/TEST.

4. Comenzar el servicio de supervisión con el subagente de Caché conectado

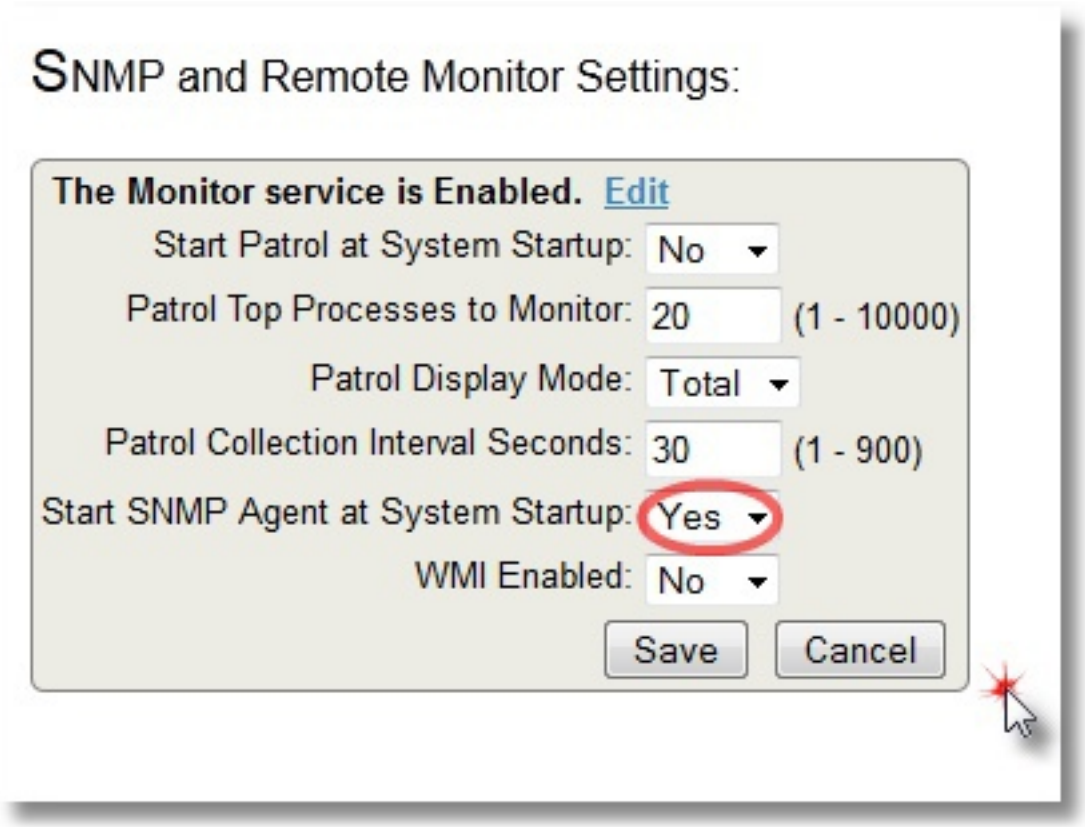
Abramos *System Administration -> Security -> Services -> %ServiceMonitor (click) -> Service Enabled (check)* .

Services are the primary means by which users and computers connect to Caché. The following services are currently available:

Name	Enabled	Public	Authentication Methods	Allowed Connections	Description	Two-Factor Enabled
%Service_Bindings	Yes	N/A	Password,Unauthenticated	Unrestricted	Controls SQL or Objects	No
%Service_CSP	Yes	Yes	Password,Unauthenticated	Unrestricted	Controls CSP Gateway access	No
%Service_CacheDirect	Yes	Yes	Unauthenticated	Unrestricted	Controls Cache Direct	No
%Service_CallIn	Yes	Yes	Unauthenticated	Unrestricted	Controls the Call-In Interface	No
%Service_ComPort	No	Yes	Unauthenticated	Unrestricted	Controls COM ports attached to a Windows system	No
%Service_Console	Yes	Yes	Unauthenticated	Unrestricted	Controls CTERM (TRM:pid) and the Windows Console	No
%Service_DataCheck	No	N/A		Unrestricted	Controls this system as a DataCheck source	No
%Service_ECP	No	N/A		Unrestricted	Controls Enterprise Cache Protocol (ECP)	No
%Service_Login	Yes	No	Password	Unrestricted	Controls SYSTEM.Security.Login	No
%Service_MSMActivate	No	N/A	Unauthenticated	Unrestricted	Controls MSM Activate Protocol	No
%Service_Mirror	No	N/A		Unrestricted	Controls Mirroring	No
%Service_Monitor	No	N/A		Unrestricted	Controls SNMP and remote Monitor commands	No
%Service_Shadow	No	N/A		Unrestricted	Controls if this system can be the source of a shadow	No
%Service_Telnet	No	Yes	Unauthenticated	Unrestricted	Controls Telnet sessions on a Windows server	No
%Service_Weblink	No	N/A	Unauthenticated	Unrestricted	Controls Weblink	No



También especificaremos que queremos iniciar el subagente SNMP cuando se inicie Caché (clic en Configure Monitor Settings):



En Linux usamos el paquete net-snmp para supervisión SNMP. Así que lo instalamos, lo configuramos para usar con subagentes y especificamos el puerto 705 como predeterminado para que el agente maestro hable con los subagentes.

```
# grep -i agentx /etc/services
agentx 705/tcp # AgentX
agentx 705/udp # AgentX
```

Puede ver un breve artículo sobre el archivo de configuración snmpd.conf que complementa al manual en cyberciti . Aquí están los ajustes finales

```
# yum install net-snmp
# grep '^[^#]' /etc/snmp/snmpd.conf
master agentx
agentXSocket TCP:localhost:705
com2sec local localhost public
group MyRWGroup v1 local
group MyRWGroup v2c local
group MyRWGroup usm local
view all included .1 80
view system included .iso.org.dod
access MyROGroup "" any noauth exact all none none
access MyRWGroup "" any noauth exact all all none
syslocation server (edit /etc/snmp/snmpd.conf)
syscontact Root <root@localhost> (configure /etc/snmp/snmp.local.conf)
dontLogTCPWrappersConnects yes
```

Reiniciamos los daemons snmpd y snmptrapd en Linux. Después, iniciaremos el servicio SNMP para activar el subagente de Caché SNMP:

```
%SYS>do start^SNMP %SYS>; Check SNMP subagent status %SYS>zwrite ^SYS("MONITOR")
^SYS("MONITOR","SNMP")="RUN"
^SYS("MONITOR","SNMP","NAMESPACE")="%SYS"
^SYS("MONITOR","SNMP","PID")=5516
^SYS("MONITOR","SNMP","PORT")=705
^SYS("MONITOR","SNMP","STARTUP")="SNMP agent started on port 705, timeout=20, winflag
=0, Debug=0"
^SYS("MONITOR","SNMP","STATE")="Terminated - 01/27/2017 04:15:01.2833PM"
^SYS("MONITOR","SNMP","WINSTART")=0
```

5. Verifique que solo estén disponibles sus propios OID nuevos

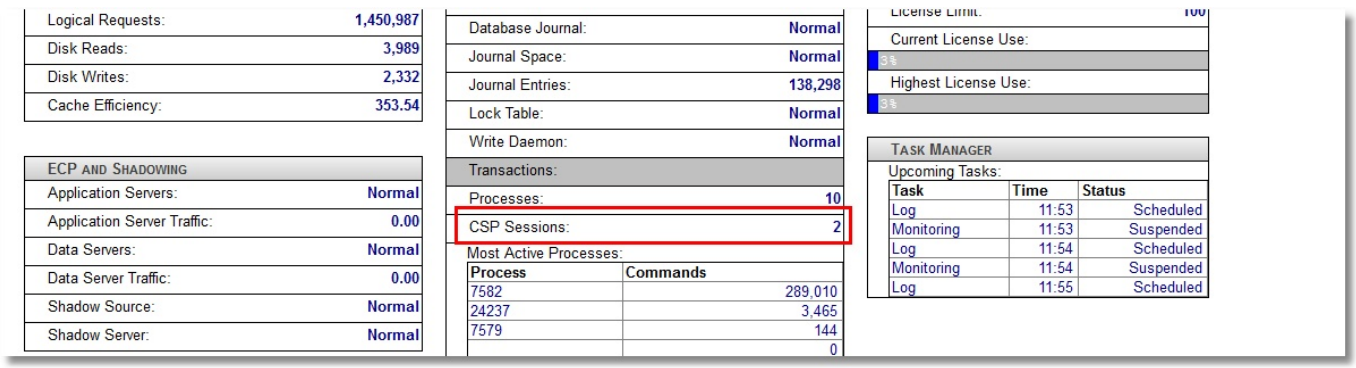
Para hacer esto, use snmpwalk. Mostraremos el OID que indica el número de sesiones CSP:

```
# snmpwalk -On -v 2c -c public localhost 1.3.6.1.4.1.99990
# snmpwalk -On -v 2c -c public localhost 1.3.6.1.4.1.99990
.1.3.6.1.4.1.99990.42.1.1.1.1.9.67.65.67.72.69.50.48.49.54 = INTEGER: 559851
.1.3.6.1.4.1.99990.42.1.1.1.2.9.67.65.67.72.69.50.48.49.54 = INTEGER: 973
.1.3.6.1.4.1.99990.42.1.1.1.3.9.67.65.67.72.69.50.48.49.54 = STRING: "2017-02-28"
.1.3.6.1.4.1.99990.42.1.1.1.4.9.67.65.67.72.69.50.48.49.54 = INTEGER: 100
.1.3.6.1.4.1.99990.42.1.1.1.5.9.67.65.67.72.69.50.48.49.54 = INTEGER: 0
.1.3.6.1.4.1.99990.42.1.1.1.6.9.67.65.67.72.69.50.48.49.54 = INTEGER: 2 # If you get
such result
# .1.3.6.1.4.1.99990 = No Such Object available on this agent at this OID
# try to restart SNMP subagent in Caché in this way:
# do stop^SNMP
# do start^SNMP
```

El archivo *ISC-TEST.mib* contiene la secuencia de nuestros OID ' s:

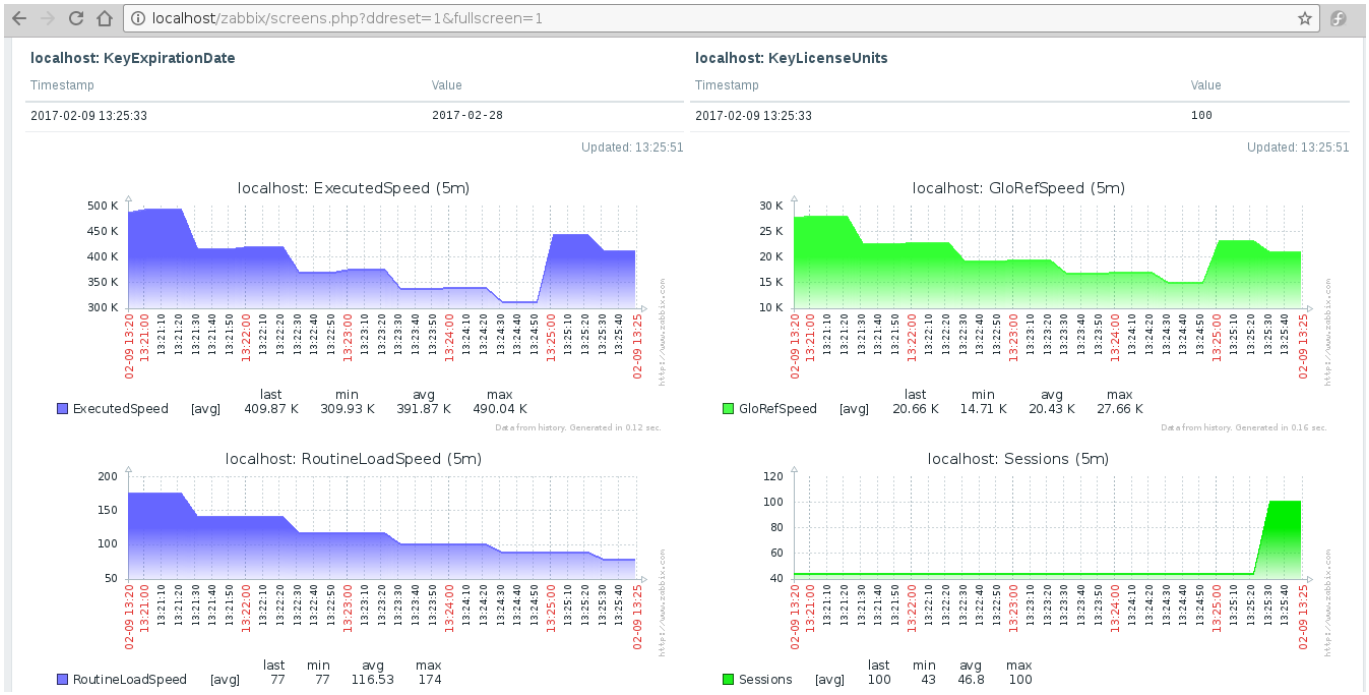
```
FictMetricsR ::=
       SEQUENCE {
        fictExecutedSpeed   Integer32,
        fictGloRefSpeed   Integer32,
        fictKeyExpirationDate   DisplayString,
        fictKeyLicenseUnits   Integer32,
        fictRoutineLoadSpeed   Integer32,
        fictSessions   Integer32
       }
```

Por tanto, el número de sesiones, por ejemplo, es el último OID 1.3.6.1.4.1.99990.42.1.1.1.6. Puede compararlo con el número de sesiones que se muestra en el tablero SMP:



6. Agregar nuestros OID a un sistema de supervisión externo

Usaremos Zabbix. La documentación de Zabbix está aquí. Puede encontrar una guía detallada de instalación y configuración de Zabbix en Linux aquí. Se eligió Zabbix como un sistema que no solo le permite trazar gráficas, sino también supervisar texto simple (en nuestro caso, fecha de vencimiento de licencia y unidades de licencia). Después de agregar nuestras 6 métricas a los elementos de nuestro host local (escriba: SNMPv2 agent) y crear 4 gráficas y 2 parámetros de texto simple (como elementos de pantalla), deberíamos ver la siguiente imagen:



Arriba podrá encontrar la información de vencimiento de licencia y número de licencias disponibles. Las gráficas hablan solas.

7. Agregar el inicio del supervisor de sistema a la lista de inicio de nuestro namespace TEST.

Hay un documento bastante bueno sobre las rutinas de usuario ejecutadas cuando Caché se inicia y se detiene. Se llaman %ZSTART y %ZSTOP, respectivamente.

Lo que nos interesa es que el supervisor del sistema (^%SYSMONMGR) se inicie en el namespace TEST durante el inicio del sistema. De forma predeterminada, este supervisor solo se inicia en el namespace %SYS. Por lo tanto, solo miraremos al programa ^%ZSTART. La fuente está en %ZSTART.mac (créelo y guárdelo al namespace %SYS).

```
%ZSTART; User startup routine.
SYSTEM;
    ; Cache starting
    do $zu(9,"","Starting System Monitor in TEST namespace by ^%ZSTART...Begin")
    znspace "TEST"
    set sc = ##class(%SYS.Monitor).Start()
    do $system.OBJ.DisplayError(sc)
    if (sc = 1) {
    do $zutil(9,"","Starting System Monitor in TEST namespace by ^%ZSTART...OK")
    } else {
    do $zutil(9,"","Starting System Monitor in TEST namespace by ^%ZSTART...ERROR")
    }
    ; Starting SNMP
    znspace "%SYS"
    do start^SNMP
    quit
LOGIN;
```

```
    ; a user logs into Cache (user account or telnet)
    quit
JOB;
    ; JOB'd process begins
    quit
CALLIN;
    ; a process enters via CALLIN interface
    quit
```

Otra forma de hacer lo mismo es usar ^%SYSMONMGR:

```
%SYS>do ^%SYSMONMGR
1. Select item 3, Configure System Monitor Classes.
2. Select item 2, Configure Startup Namespaces.
3. Select item 2, Add Namespace.
Namespace? TEST 4. Select item 1, List Start Namespaces.
Option? 1
     TEST 5. Select item 4, Exit.
6. Select item 3, Exit.
7. Select item 8, Exit.
```

Reiniciamos ahora Caché (de ser posible) para asegurarnos de que las estadísticas SNMP se sigan recolectando después de un reinicio.

Y terminamos. Quizás algunos cuestionen mi elección de parámetros supervisados o código, pero la tarea era mostrar la mera posibilidad de implementar dicha supervisión. Más tarde puede agregar parámetros extra o refactorizar su código.

¡Gracias por vuestra atención!

#Administración del sistema #Monitorización #Visualización #Caché

URL de fuente:https://es.community.intersystems.com/post/crear-oids-snmp-personalizados