記事
Tomohiro Iwamoto · 2020年6月5日 14m read

InterSystemsデータプラットフォームとパフォーマンス-パート5: SNMPによる監視

前回の記事では、pButtonsを使って履歴パフォーマンスメトリックを収集する方法を説明しました。 すべてのデータプラットフォームインスタンス（Ensemble、Cachéなど）にはpButtonsがインストールされていることがわかっているため、私はpButtonsを使用する傾向にありますが、 Caché パフォーマンスメトリックをリアルタイムで収集、処理、表示する方法はほかにもあり、単純な監視や、それよりもさらに重要な、より高度な運用分析とキャパシティプランニングに使用することができます。
データ収集の最も一般的な方法の1つは、SNMP（簡易ネットワーク管理プロトコル）を使用することです。

SNMPは、Cachéが管理・監視情報をさまざまな管理ツールに提供するための標準的な方法です。 Cachéのオンラインドキュメンテーションには、CachéとSNMP間のインターフェースに関する詳細が説明されています。 SNMPはCachéと「単純に連携」するはずですが、構成にはいくつかの技と罠があります。 私自身、はじめに何度も過ちを繰り返し、InterSystemsの同僚から助けを得ながらCachéをオペレーティングシステムのSNMPマスターエージェントにやっと接続できた経験から、皆さんが同じような困難を避けられるようにこの記事を書くことにしました。

この記事では、Red Hat
Linuxで実行するCachéにおけるSNMPのセットアップと構成について説明します。ほかの
*nixフレーバーでも同じ手順を使用できるはずです。
Linuxの場合はセットアップにもう少しコツが必要なので、Red
Hatを使ってこの記事を書いています。WindowsのCachéは、自動的にDLLをインストールして標準のWindows SNMPサービスに接続するため、より簡単に構成できます。

サーバー側でSNMPのセットアップが完了したら、複数のツールを使用して監視を開始できます。
私は
一般的な
PRTGツールを使
って監視を説明しようと思いますが
、ほかにもたくさんのツールがあります。ここに示すのはリストの一部です。

CachéとEnsembleのMIBファイルは
Cachéインストールディレクトリ/SNMP フォルダにあります。ファイルは ISC-CACHE.mib と ISC-ENSEMBLE.mib です。

このシリーズのこれまでの記事:

- パート1 - はじめの一歩。メトリックの収集。
- パート2 - 収集したメトリックの確認。
- パート3 - CPUに注目
- パート4 - メモリの確認。

まずはここから始めましょう...

まず、Cachéオンラインドキュメンテーションの「Monitoring Caché Using SNMP（SNMPによるCachéの監視）」を確認してください。

1. Cachéを構成する

Caché オンラインドキュメンテーションの「Managing SNMP in Caché（CachéでのSNMPの管理）」セクションにある手順に従って、Caché監視サービスを有効化し、Caché SNMPサブエージェントがCachéの起動時に自動的に開始するように構成します。

OSのプロセスリストを確認するなどして、Cachéプロセスが実行していることを確認します。

```
ps -ef | grep SNMP
root      1171  1097  0 02:26 pts/1    00:00:00 grep SNMP
root     27833     1  0 00:34 pts/0    00:00:05 cache -s/db/trak/hs2015/mgr -cj -p33
JOB^SNMP
```

これで、Cachéの構成は完了です！

2. オペレーティングシステムを構成する

ここでは、もう少しやることがあります。
まず、snmpdデーモンがインストール済みで実行中であることを確認してください。
そうでない場合はsnmpdをインストールして開始します。

次ようにして、snmpdのステータスを確認します。

```
service snmpd status
```

次のようにして、snmpdを開始または停止します。

```
service snmpd start|stop
```

snmpがインストールされていない場合は、OSの指示に従ってインストールする必要があります。次に例を示します。

```
yum -y install net-snmp net-snmp-utils
```

3. snmpdを構成する

Cachéドキュメンテーションで説明されるとおり、Linuxシステムで最も重要なタスクは、システム上のSNMPマスターエージェントがAgent Extensibility（AgentX）プロトコルと互換性があり（Cachéはサブエージェントとして実行）、マスターがアクティブで標準のAgentX TCPポート705で待ち受け状態であることを確認することです。

私が壁にぶつかったのは、この部分でした。 snmp.confファイルで基本的なミスをしてしまったせいで、Caché SNMPサブエージェントがOSマスターエージェントと通信していませんでした。
次の /etc/snmp/snmp.conf サンプルファイルはagentXを開始してCachéとEnsembleのSNMP MIBにアクセスを提供するように構成されています。

次の構成があなたの組織のセキュリティポリシーを順守しているかどうかを確認する必要があります。

システムの設定が反映されるように、少なくとも次の行を編集する必要があります。

例えば、

```
syslocation  "System_Location"
```

の行を次のように変更します。

```
syslocation  "Primary Server Room"
```

また、少なくとも次の2行を編集します。

```
syscontact  "Your Name"
trapsink  Caché_database_server_name_or_ip_address public
```

次と一致するように、既存の`/etc/snmp/snmp.conf`ファイルを編集するか置換します。

```
################################################################################
#
# snmpd.conf:
#   An example configuration file for configuring the NET-SNMP agent with Cache.
#
#   This has been used successfully on Red Hat Enterprise Linux and running
#   the snmpd daemon in the foreground with the following command:
#
#   /usr/sbin/snmpd -f -L -x TCP:localhost:705 -c./snmpd.conf
#
#   You may want/need to change some of the information, especially the
#   IP address of the trap receiver of you expect to get traps. I've also seen
#   one case (on AIX) where we had to use  the "-C" option on the snmpd command
#   line, to make sure we were getting the correct snmpd.conf file.
#
################################################################################

###############################################################################
# SECTION: System Information Setup
#
#   This section defines some of the information reported in
#   the "system" mib group in the mibII tree.

# syslocation: The [typically physical] location of the system.
#   Note that setting this value here means that when trying to
#   perform an snmp SET operation to the sysLocation.0 variable will make
#   the agent return the "notWritable" error code.  IE, including
#   this token in the snmpd.conf file will disable write access to
#   the variable.
#   arguments:  location_string

syslocation  "System Location"

# syscontact: The contact information for the administrator
#   Note that setting this value here means that when trying to
#   perform an snmp SET operation to the sysContact.0 variable will make
#   the agent return the "notWritable" error code.  IE, including
#   this token in the snmpd.conf file will disable write access to
#   the variable.
#   arguments:  contact_string

syscontact  "Your Name"

# sysservices: The proper value for the sysServices object.
#   arguments:  sysservices_number
```

```
sysservices 76

##############################################################################
# SECTION: Agent Operating Mode
#
#   This section defines how the agent will operate when it
#   is running.

# master: Should the agent operate as a master agent or not.
#   Currently, the only supported master agent type for this token
#   is "agentx".
#
#   arguments: (on|yes|agentx|all|off|no)
master agentx
agentXSocket tcp:localhost:705

##############################################################################
# SECTION: Trap Destinations
#
#   Here we define who the agent will send traps to.

# trapsink: A SNMPv1 trap receiver
#   arguments: host [community] [portnum]

trapsink  Caché_database_server_name_or_ip_address public

##############################################################################

# Access Control

##############################################################################
# As shipped, the snmpd demon will only respond to queries on the
# system mib group until this file is replaced or modified for
# security purposes.  Examples are shown below about how to increase the
# level of access.
#
# By far, the most common question I get about the agent is "why won't
# it work?", when really it should be "how do I configure the agent to
# allow me to access it?"
#
# By default, the agent responds to the "public" community for read
# only access, if run out of the box without any configuration file in
# place.  The following examples show you other ways of configuring
# the agent so that you can change the community names, and give
# yourself write access to the mib tree as well.
#
# For more information, read the FAQ as well as the snmpd.conf(5)
# manual page.
#
####
# First, map the community name "public" into a "security name"

#       sec.name  source          community
com2sec notConfigUser  default       public

####
# Second, map the security name into a group name:
```

```
#       groupName      securityModel securityName
group   notConfigGroup v1           notConfigUser
group   notConfigGroup v2c           notConfigUser

####
# Third, create a view for us to let the group have rights to:

# Make at least  snmpwalk -v 1 localhost -c public system fast again.
#       name           incl/excl     subtree         mask(optional)
# access to 'internet' subtree
view    systemview    included   .1.3.6.1

# access to Cache MIBs Caché and Ensemble
view    systemview    included   .1.3.6.1.4.1.16563.1
view    systemview    included   .1.3.6.1.4.1.16563.2
####
# Finally, grant the group read-only access to the systemview view.

#       group          context sec.model sec.level prefix read   write  notif
access  notConfigGroup ""      any       noauth    exact  systemview none none
```

/etc/snmp/snmp.confファイルの編集が完了したら、snmpdデーモンを再起動します。

```
service snmpd restart
```

snmpd
のステータス
を確認します。AgentXが開始されていることに注意してください。ステータス行に「**Turning on AgentX master support**」が表示されます。

```
h-4.2# service snmpd restart
Redirecting to /bin/systemctl restart  snmpd.service
sh-4.2# service snmpd status
Redirecting to /bin/systemctl status  snmpd.service
? snmpd.service - Simple Network Management Protocol (SNMP) Daemon.
   Loaded: loaded (/usr/lib/systemd/system/snmpd.service; disabled; vendor preset: di
sabled)
   Active: active (running) since Wed 2016-04-27 00:31:36 EDT; 7s ago
 Main PID: 27820 (snmpd)
   CGroup: /system.slice/snmpd.service
           ??27820 /usr/sbin/snmpd -LS0-6d -f

Apr 27 00:31:36 vsan-tc-db2.iscinternal.com systemd[1]: Starting Simple Network Manag
ement Protocol (SNMP) Daemon....
Apr 27 00:31:36 vsan-tc-
db2.iscinternal.com snmpd[27820]: Turning on AgentX master support.
Apr 27 00:31:36 vsan-tc-db2.iscinternal.com snmpd[27820]: NET-SNMP version 5.7.2
Apr 27 00:31:36 vsan-tc-db2.iscinternal.com systemd[1]: Started Simple Network Manage
ment Protocol (SNMP) Daemon..
sh-4.2#
```

snmpdを再起動したら、^SNMPルーチンを使用してCaché
SNMPサブエージェントを再起動する必要があります。

```
%SYS>do stop^SNMP()

%SYS>do start^SNMP(705,20)
```

オペレーティングシステムのsnmpdデーモンとCachéサブエージェントが実行しており、アクセスできるようになっているはずです。

4. MIBアクセスを検証する

MIBアクセスは、コマンドラインで以下のコマンドを使って確認できます。snmpgetは、単一の値を返します。

```
snmpget -mAll -v 2c -c public vsan-tc-
db2 .1.3.6.1.4.1.16563.1.1.1.1.5.5.72.50.48.49.53

SNMPv2-SMI::enterprises.16563.1.1.1.1.5.5.72.50.48.49.53 = STRING: "Cache for UNIX (R
ed Hat Enterprise Linux for x86-64) 2015.2.1 (Build 705U) Mon Aug 31 2015 16:53:38 ED
T"
```

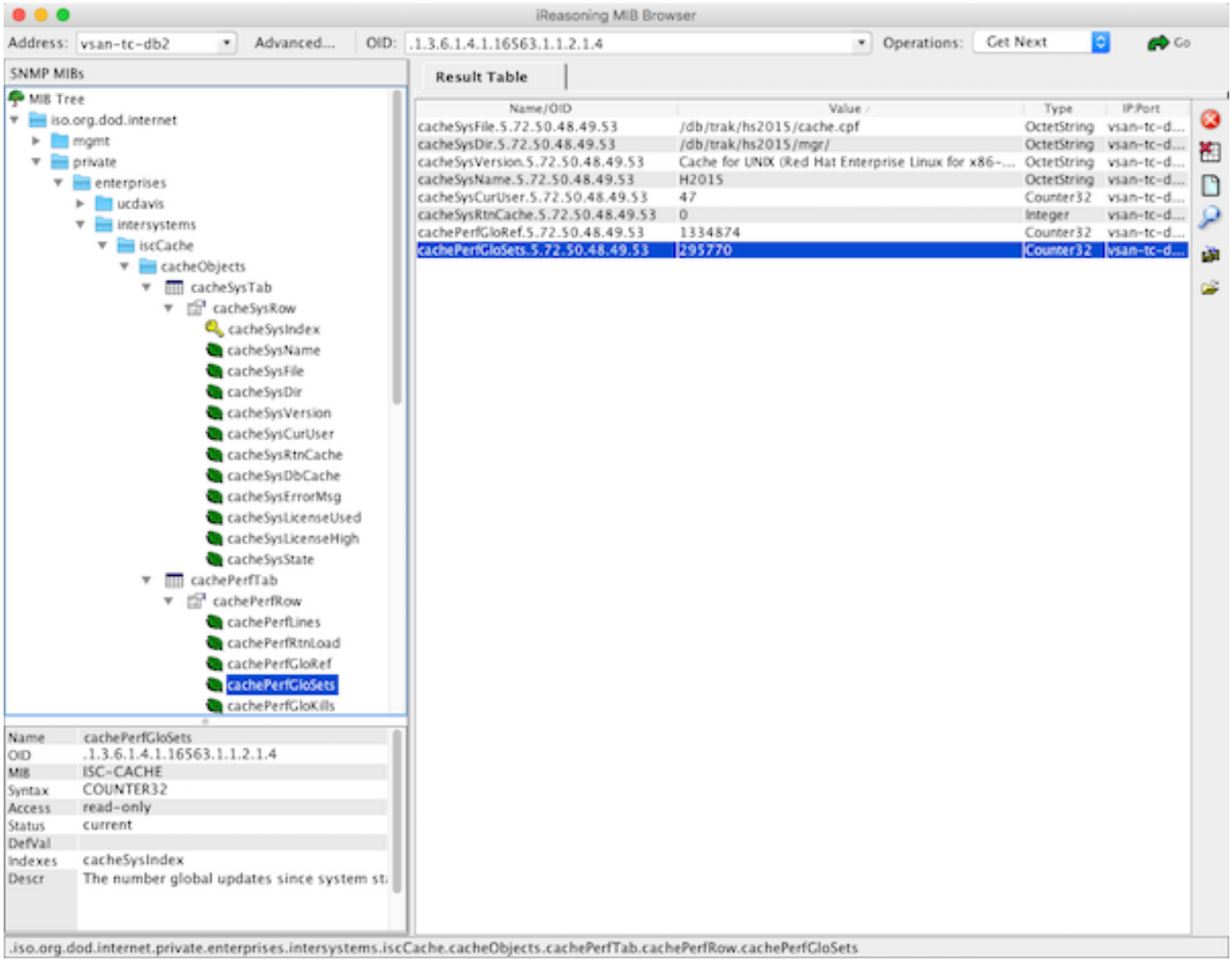
そして、snmpwalkは、MIBツリーまたはブランチを「ウォーク」します。

```
snmpwalk -m ALL -v 2c -c public vsan-tc-db2 .1.3.6.1.4.1.16563.1.1.1.1

SNMPv2-SMI::enterprises.16563.1.1.1.1.2.5.72.50.48.49.53 = STRING: "H2015"
SNMPv2-SMI::enterprises.16563.1.1.1.1.3.5.72.50.48.49.53 = STRING: "/db/trak/hs2015/c
ache.cpf"
SNMPv2-SMI::enterprises.16563.1.1.1.1.4.5.72.50.48.49.53 = STRING: "/db/trak/hs2015/m
gr/"
etc
etc
```

また、システムデータの表示には、さまざまなウィンドウと *nixクライアントを使用できます。
私は無料のiReasoning MIB Browserを使用しています。 MIBの構造を認識させるために、ISC-CACHE.MIBファイルをクライアントにロードする必要があります。

次の画像は、OS X上のiReasoning MIB Browserをキャプチャしたものです。

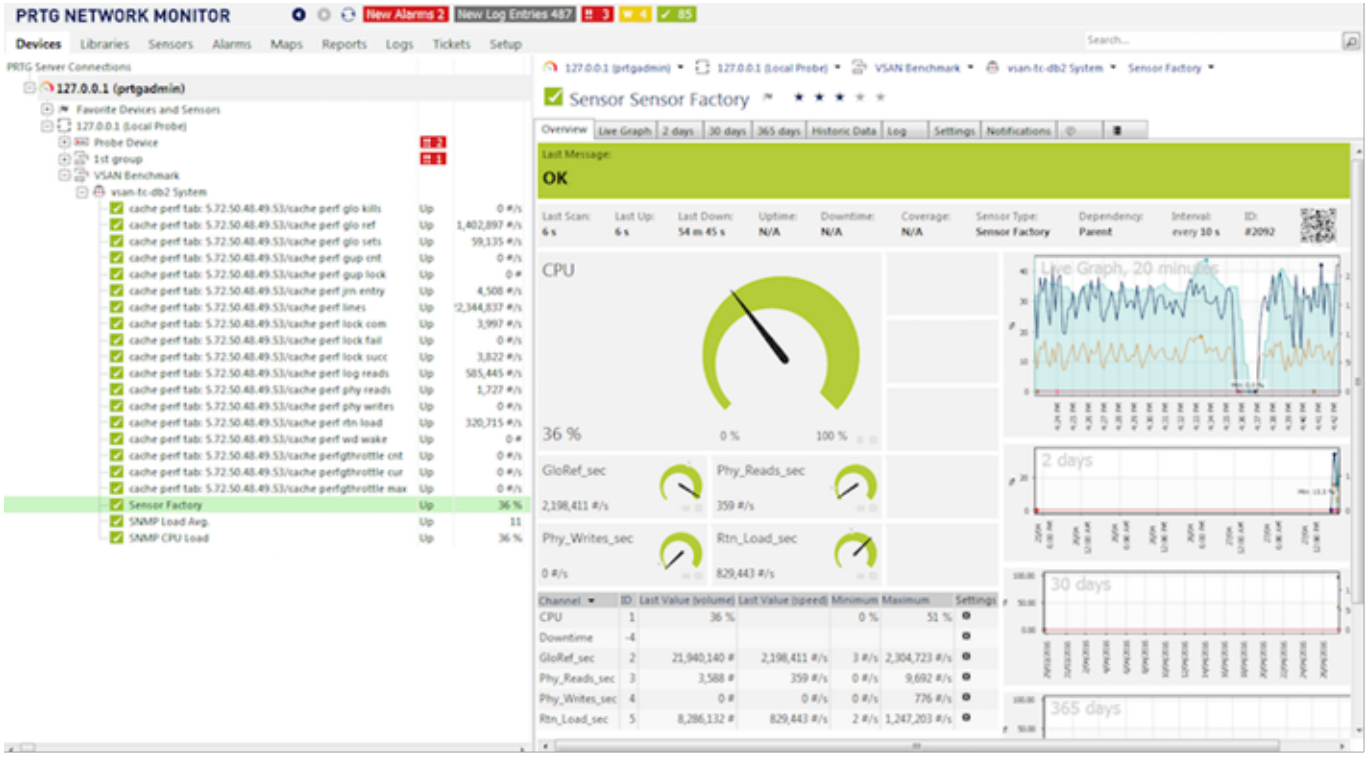


監視ツールに含める

これが、実装に大きな違いを生じる可能性のあるところです。
監視ツールや分析ツールの選択は、あなたにお任せすることにします。

システムの監視と管理に使用するツールとそれから得られる価値について、コメント欄で詳しくお聞かせください。 他のコミュニティメンバーにとっても非常に役立つと思います。

次は、一般的に使用されているPRTG Network Monitorのスクリーンショットで、Cachéメトリックが表示されています。 CachéメトリックをPRTGに含める手順はほかのツールと同じです。



ワークフローの例 - 監視ツールをCaché MIBに追加する。

ステップ 1.

オペレーティングシステムのMIBに接続できることを確認します。
ヒントは、Cachéではなく、オペレーティングシステムに対してトラブルシューティングを行うということです。
監視ツールは一般的なオペレーティングシステムのMIBをすでに認識しており、あらかじめ構成されている可能性が高いため、ベンダーやほかのユーザーから支援を得る方が簡単かもしれません。

選択した監視ツールによっては、SNMP「モジュール」または「アプリケーション」を追加する必要がある場合があります。これらは通常無料で、オープンソースです。
この手順では、ベンダーが作成した指示が非常にわかりやすいと思いました。

オペレーティングシステムのメトリックを監視できるようになったら、Cachéを追加します。

ステップ 2.

`ISC-CACHE.mib` と `ISC-ENSEMBLE.mib` をツールにインポートして、MIB構造を認識できるようにします。

ここでの手順はツールによって異なります。PRTGの場合は「MIB
Importer」ユーティリティが提供されています。
基本的には、ツールで `ISC-CACHE.mib`
テキストファイルを開いて、ツールの内部形式にインポートする、という手順です。
たとえば、SplunkではPython形式が使用されています。

注意: すべてのCaché
MIBブランチでセンサーを追加しようとすると、PRTGツールはタイムアウトしてしまいました。 ツリー全体をウォークし、プロセスリストなどのいくつかのメトリックでタイムアウトしたのだと思います。このトラブルシューティングに時間をかける代わりに、 `ISC-CACHE.mib`
からパフォーマンスブランチ（cachePerfTab）のみをインポートすることで問題を回避しました。

インポートと変換が完了したら、ネットワークのほかのサービスからデータを収集するためにMIBを再利用することができます。 上の図では、PRTGがSensor
Factoryセンサを使用して複数のセンサを1つのチャートに結合しています。

最後に

監視ツール、警告ツール、そして非常に高度なスマート分析ツールにはさまざまなものがあり、無料で提供されているものもあれば、サポートライセンス付きのものもあり、機能も多岐に渡ります。

システムを監視し、どのアクティビティが正常であり正常外であるのかを理解し、調査する必要があります。
SNMPは、CachéとEnsembleのメトリックを確認できるようにするための簡単な方法と言えます。

#インターシステムズビジネスソリューションとアーキテクチャ #システム管理 #パフォーマンス #監視 #Caché #InterSystems IRIS #InterSystems IRIS for Health

---

ソースURL:
https://jp.community.intersystems.com/post/intersystems%E3%83%87%E3%83%BC%E3%82%BF%E3%83%97%E3%83%A9%E3%83%83%E3%83%88%E3%83%95%E3%82%A9%E3%83%BC%E3%83%A0%E3%81%A8%E3%83%91%E3%83%95%E3%82%A9%E3%83%BC%E3%83%9E%E3%83%B3%E3%82%B9-%E3%83%91%E3%83%BC%E3%83%885%C2%A0snmp%E3%81%AB%E3%82%88%E3%82%8B%E7%9B%A3%E8%A6%96%C2%A0