



**Using Hobbit Monitor with SAS® 9.1.3 BI Servers –
An Appetizer**

February 2008

**THE
POWER
TO KNOW.**

Copyright Notice

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *Using Hobbit Monitor with SAS 9.1.3 BI Servers – An Appetizer*, Cary, NC: SAS Institute Inc., 2007, 2008.

Using Hobbit Monitor with SAS 9.1.3 BI Servers – An Appetizer

Copyright © 2007, 2008, SAS Institute Inc., Cary, NC, USA.

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, by any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc. Limited permission is granted to store the copyrighted material in your system and display it on terminals, print only the number of copies required for use by those persons responsible for installing and supporting the SAS programming and licensed programs for which this material has been provided, and to modify the material to meet specific installation requirements. The SAS Institute copyright notice must appear on all printed versions of this material or extracts thereof and on the display medium when the material is displayed. Permission is not granted to reproduce or distribute the material except as stated above.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software by the government is subject to restrictions as set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.

Copyright © 2007, 2008, SAS Institute Inc. All rights reserved.

www.sas.com

® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

Table of Contents

About this Guide	4
Who should read this Guide	4
Conventions used in this Guide.....	4
Chapter 1: Introduction and Overview	5
1.1 About Hobbit Monitor	5
1.2 Project Goals and Objectives	5
1.3 An Appetizer	6
1.4 Feedback	6
Chapter 2: Architecture and Systems Overview	7
2.1 SAS 9 Business Intelligence Architecture	7
2.2 Requirements for Hobbit Monitoring	7
Chapter 3: Hobbit Monitoring of the SAS BI Environment	8
3.1 – Configuring Hobbit to monitor SAS service ports	8
3.2 – Configuring Hobbit to monitor SAS server processes	9
3.3 – Configuring Hobbit to monitor SAS server logs	10
3.4 – Configuring Hobbit to use an external monitor	11
3.5 – Monitoring Windows systems with Hobbit and BBWin.....	12
3.6 – Using SAS to automatically generate Hobbit configuration files	14
3.7 – Using SAS to automatically generate BBWin configuration files.....	15
Appendix A	16
setinit.sh	16

About this Guide

Who should read this Guide

This guide is intended for system administrators with knowledge of, and practical experience with the following:

- Installation and administration of SAS Enterprise BI Server 9.1.3
- Installation and administration of Hobbit Monitoring
- Unix/Linux system administration

Conventions used in this Guide

Descriptive text	Descriptive text in the document will appear in Times New Roman type.
<code><variable></code>	Symbols in <i>italic Courier</i> monospace type and enclosed in angle brackets represent site-dependent values that you must provide such as file paths, names, or text strings.
<code>cat hobbit-clients.cfg</code>	Text in Courier monospace type represents commands, command options, flags, and file paths that should be entered as shown. Commands that you invoke will be shown on separate, indented lines from the descriptive text.

Chapter 1: Introduction and Overview

The SAS Business Intelligence (BI) architecture provides a complete portfolio of BI capabilities and applies the power of SAS analytics and data integration to create a complete and easy-to-use BI solution. SAS Enterprise BI Server provides a fully integrated and comprehensive suite of BI software that addresses the needs of IT Management as well as executives and business users. The distributed, heterogeneous nature of the BI architecture requires corresponding methods for IT organizations to monitor, manage and administer the environment. The integration of Hobbit Monitor with the SAS BI architecture provides an inexpensive, simplified system to monitor the operational status of SAS BI servers across a network of machines, using a web-based interface.

1.1 About Hobbit Monitor

Hobbit provides a web-based framework to monitor applications, systems resources, network services, and other features of a heterogeneous computing environment. Hobbit periodically generates requests to network services, noting whether or not the service is responding as expected. Hobbit uses local agents on each monitored machine to track data such as disk utilization, log files, and processes.

Monitoring results are accumulated at a central Hobbit server. That data is used to generate web pages showing the current status of the monitored items and in many cases can be presented as graphs showing historical trends.

Commercial monitoring systems such as IBM Tivoli and HP OpenView provide functionality similar to, or exceeding the capabilities of Hobbit, but require a high level of expertise to install and configure, and their acquisition costs may put them out of reach of smaller enterprises or company divisions.

Hobbit is an Open Source software package, available as a free download from the SourceForge project site¹. Hobbit is comparatively simple to configure and maintain, while offering significant flexibility and power for monitoring smaller computing environments. One of the greatest benefits of Hobbit is that it only requires a browser to be able to view the hobbit monitors.

1.2 Project Goals and Objectives

SAS Institute has developed a sample configuration of Hobbit Monitor for monitoring SAS Enterprise Business Intelligence servers for the occurrence of anomalous activity such as changes in server availability and other significant events.

The goals of this project are as follows:

1. to develop the tools and/or documentation needed to allow customers to integrate SAS 9.1.3 EBI servers with the Hobbit monitoring package.
2. to duplicate the level of monitoring offered by the EMI integrations of SAS 9.1.3 with IBM Tivoli and HP OpenView, including SAS server log file monitoring.
3. to investigate automating configuration of Hobbit by leveraging SAS installation site data generated by the SAS Configuration Wizard and/or Metadata initialization.

¹ The Hobbit Monitor – SourceForge.net (<http://hobbitmon.sourceforge.net/>)

1.3 An Appetizer

ap·pe·tiz·er [ap-i-tahy-zer]

–noun

1. a small portion of a food or drink served before or at the beginning of a meal to stimulate the desire to eat.
2. any small portion that stimulates a desire for more or that indicates more is to follow: *The first game was an appetizer to a great football season.*

- Dictionary.com²

This development project is ongoing. In the interest of gaining valuable feedback from customers, the Enterprise Management Integration group at SAS is releasing the Hobbit monitoring integration in stages as each is completed. This initial “appetizer” document describes how to configure Hobbit to monitor SAS BI servers by probing their related service ports.

The author of Hobbit operates a demonstration site showing Hobbit actively monitoring several systems (<http://www.hsw.n.dk/hobbit/>). Clicking on the colored icons will drill down into a more-detailed view of status for the particular category. Clicking on icons in the “trends” column for a given system will display a graphical plot of trends over time for that category.

1.4 Feedback

Read more about SAS Institute Inc Information Technology products and solutions online:

<http://www.sas.com/start/it.html>

and visit the SAS Research and Development online Focus Area site for related Enterprise Management Integration topics:

<http://support.sas.com/rnd/emi/>

Your questions, suggestions, and comments on this project are welcomed. Send e-mail mentioning “Hobbit” in the Subject: to

emifedback@sas.com

² "appetizer." *Dictionary.com Unabridged (v 1.1)*. Random House, Inc. 23 Aug. 2007. <Dictionary.com <http://dictionary.reference.com/browse/appetizer>>.

Chapter 2: Architecture and Systems Overview

2.1 SAS 9 Business Intelligence Architecture

The SAS 9 Business Intelligence Architecture represents a fundamental shift in design and methodology for deploying the analytical and statistical strengths of Foundation SAS. The SAS 9 BI framework includes multiple layers which may be deployed within a single environment or distributed across heterogeneous environments. The BI architecture includes a diverse collection of thin and thick clients represented by a **ClientTier**. These clients connect to a **MidTier** or **ServerTier** depending on the client capabilities. The MidTier is comprised of multiple SAS Java-based components that are deployed under a web application server. To harness the analytical strengths of SAS, the MidTier communicates with a suite of SAS Servers. Depending on the client requests and workloads, one or more servers may actually process a MidTier request for computing services. The SAS 9 BI architecture is hinged together via a SAS Metadata Server. The Metadata Server serves as the cornerstone for aggregating and correlating the configuration, management and deployment of the SAS 9 architecture, applications, and users that cooperate within the enterprise.

2.2 Requirements for Hobbit Monitoring

This document presumes that you already have a working installation of Hobbit. There is plenty of information on the SourceForge Hobbit site to get you to this point. You also need to know the ports you are running the different SAS services on. To track the SAS server processes, you need to know the SAS scripts that start the services (it's highly recommended that full paths are used wherever possible). Depending on your site's SAS installation, some of the services you may want to monitor include:

- SAS Metadata Server
- SAS Object Spawner
- SAS OLAP Server
- SAS Connect Server
- SAS Workspace Server
- SAS Stored Process Server and LB Connections
- SAS Remote Services
- SAS Analytics Platform (Enterprise Miner, Forecast Server, Model Manager)
- SAS Share Server
- SAS Scalable Performance Data Server (SPDS)

Chapter 3: Hobbit Monitoring of the SAS BI Environment

3.1 – Configuring Hobbit to monitor SAS service ports

One of the simplest methods for monitoring the SAS services is to monitor the ports associated with the various services. Hobbit parses output from the Unix “netstat” command to determine the status of TCP ports on a monitored server. This approach allows you to not only monitor for an active listener on a particular port, but you can also track how many connections are active (ESTABLISHED) on the port.

When the SAS services start up, they immediately begin listening on the ports assigned to them. These ports are determined prior to the SAS system installation and should be available from your SAS system administrator or installer. A list of default ports can be found in SAS Usage Note 15162 (<http://support.sas.com/kb/15/162.html>).

To begin monitoring ports, you will need to modify your `$HOBBITSERVER/etc/hobbit-clients.cfg` file. Note that `$HOBBITSERVER` is the path to where you installed your Hobbit server (e.g. `- /usr/local/hobbit/server`).

These examples will assume that you have a multi-machine SAS BI installation consisting of an application server, metadata server and midtier.

Figure 3-1 shows HOST rules in the Hobbit server configuration file that define service port monitors using the default port numbers for a typical set of SAS BI servers. As shown, the example uses a % wildcard pattern for the host names such that these monitoring definitions would apply to any machine at your domain that was named beginning with “appserver”, “midtier”, or “metadata” respectively. For example, you might have separate SAS BI installations for different business units:

```
appserver.widgets.yourcompany.com
appserver.gadgets.yourcompany.com
appserver.sales.yourcompany.com
```

For a simple configuration, the “HOST=” lines should be changed to the fully qualified node name(s) for your machines which supply the described services.

The TRACK keyword is used to generate trend data for monitored items. As shown, the sample configuration will track the number of established connections to the workspace server as well as connections to the metadata server. This will appear as a graph at the bottom of the ports monitor as well as on the trends page. This is a built-in feature of Hobbit.

```

HOST=%appserver.*
  PORT "LOCAL=%[\.:]8581$" state=LISTEN \
    "TEXT=Object Spawner Operator Port"
  PORT "LOCAL=%[\.:]8571$" state=LISTEN \
    "TEXT=Object Spawner LoadBalancer Port"
  PORT "LOCAL=%[\.:]5451$" state=LISTEN "TEXT=OLAP Server"
  PORT "LOCAL=%[\.:]7551$" state=LISTEN "TEXT=Connect Server"
  PORT "LOCAL=%[\.:]8591$" state=LISTEN "TEXT=Logical Workspace Server"
  PORT "LOCAL=%[\.:]8551$" state=LISTEN "TEXT=SAS Share Server"
  PORT "LOCAL=%[\.:]8601$" state=LISTEN "TEXT=SAS Stored Process Server"
  PORT "LOCAL=%[\.:](8611|8621|8631)$" state=LISTEN \
    "TEXT=SASMain Stored Process LB Connections"
  PORT "LOCAL=%[\.:]5160$" state=LISTEN "TEXT=SPD Server"
  PORT "LOCAL=%[\.:](6098|6099)$" state=LISTEN \
    "TEXT=SAS Analytics Platform" 2
  PORT "LOCAL=%[\.:]8591$" state=ESTABLISHED MIN=0 TRACK=sasmainws \
    "TEXT=Active Connections to SASMain Workspace Server"

HOST=%midtier.*
  PORT "LOCAL=%[\.:]5099$" state=LISTEN "TEXT=SAS Remote Services"

HOST=%metadata.*
  PORT "LOCAL=%[\.:]8561$" state=LISTEN "TEXT=Metadata Server"
  PORT "LOCAL=%[\.:]8561$" state=ESTABLISHED MIN=0 TRACK=omr \
    "TEXT=Active Connections to Metadata Server"

```

Figure 3-1: Sample hobbit-clients.cfg entries to monitor SAS BI ports

3.2 – Configuring Hobbit to monitor SAS server processes

Monitoring processes can be a bit trickier. Hobbit parses output from the Unix “ps” command and compares it to the PROC pattern strings in your `hobbit-clients.cfg` configuration. Figure 3.2 shows sample entries in the configuration file that will match against typical SAS server processes. Process name patterns that contain spaces must be enclosed in double-quotes. As in PORT monitoring, you can use the TRACK keyword to generate trend data for the monitored processes. As shown, the sample configuration entries will track the number of running processes that match the “spdsbase” PROC pattern string. This should give a good idea of the number of connections to SPDS at a given time. This information is graphed at the bottom of the PROC status page. This is a built-in feature of Hobbit.

If you run more than one SAS installation simultaneously on the same machine, Hobbit will not be able to distinguish between the SAS processes associated with a particular installation unless you modify the patterns below to contain a more fully qualified command path.

Ideally, you’d use the full process command path in the test, but on some systems (Solaris, for example) the command output of “ps” is truncated to 80 characters. This can cut off some critical path information if you nest your services deeply enough in the directory hierarchy.

```

HOST=%appserver.*
    PROC spdssnet
    PROC spdsaud
    PROC spdserv
    PROC spdsnsrv
    PROC spdslog
    PROC spdsbase TRACK=spdbase
    PROC "utilities/bin/objspawn"
    PROC "ConnectServer.sh start2"
    PROC "OLAPServer.sh start2"
    PROC "apserver start"

HOST=%metadata.*
    PROC "MetadataServer"

```

Figure 3-2: Sample hobbit-clients.cfg entries to monitor SAS BI processes

3.3 – Configuring Hobbit to monitor SAS server logs

In addition to monitoring SAS service ports and processes, Hobbit can monitor the log files generated by those SAS servers. The logs you wish to monitor are defined in `$HOBBITSERVER/etc/client-local.cfg` and the status change triggers are defined in `$HOBBITSERVER/etc/hobbit-clients.cfg`. Both of these files reside on the Hobbit server machine. The `client-local.cfg` provides a centralized source for configuration settings that are used by each Hobbit client when it runs on a monitored host.

To monitor a log file, you *MUST* configure both `client-local.cfg` and `hobbit-clients.cfg`. If you configure only the `client-local.cfg` file, the client will collect the log data that you can view it in the “client data” display, but it will not affect the state of the displayed “msgs” status. On the other hand, if you configure only the `hobbit-clients.cfg` file, then there will be no log data to inspect, and you will not see any updates of the “msgs” status either.

Figure 3-3 shows sample entries in the `client-local.cfg` that will monitor the Unix system messages log and the SAS Object Spawner logs on the “appserver” host, and will monitor the SAS Metadata log on the “metadata” host. The text in brackets specifies the hostname to which this section of `client-local.cfg` applies.

For the SAS Metadata Server log, the configuration entry uses back-quotes to indicate that the log filename to monitor is generated dynamically: in this case, the Unix “date” command is used to generate a name that corresponds to today’s instance of the rotating SAS Metadata Server log.

The “10240” specifies the maximum number of bytes to transfer to the Hobbit server. No more than 30 minutes of data will be transferred in any case.

Note: each “log:” line in Figure 3-3 should occur as a single line. Editing limitations forced line-wraps of the sample text.

```

[appserver]
log:/var/adm/messages:10240
log:/path/to/biconfig/Lev1/SASApp/ObjectSpawner/logs/objspawn.log:10240
log:/path/to/biconfig/Lev1/SASApp/ObjectSpawner/logs/objspawn_console.log:10240

[metadata]
log:`date
+ "/path/to/biconfig/Lev1/SASMain/MetadataServer/logs/MetadataServer_%Y.%m.%d.
log"~:10240

```

Figure 3-3: Sample client-local.cfg entries to monitor SAS BI server logs

3.4 – Configuring Hobbit to use an external monitor

Hobbit can be configured to use external monitors implemented as custom scripts (in your favorite language). For example, your site can monitor your SAS SID license files so that you will be warned when your license is about to expire. Modify `$HOBBITCLIENHOME/etc/clientlaunch.cfg` and add the lines as shown in Figure 3-4. This will periodically execute the external monitor script “setinit.sh”, redirecting output from the script to the specified file (LOGFILE), and will run the script once a day (INTERVAL 1d). The “setinit.sh” file is supplied in Appendix A.

```

# User Added Monitors

# SAS setinit monitor (alert before license expires)
[setinit]
  ENVFILE $HOBBITCLIENHOME/etc/hobbitclient.cfg
  CMD $HOBBITCLIENHOME/ext/setinit.sh
  LOGFILE $HOBBITCLIENHOME/logs/setinit.log
  INTERVAL 1d

```

Figure 3-4: Sample clientlaunch.cfg entries to monitor SAS license expiration

3.5 – Monitoring Windows systems with Hobbit and BBWin

Hobbit can be used to monitor the status of Windows systems and SAS services running on Windows nodes by using the BBWin client. Like Hobbit, BBWin is an Open Source software package, available as a free download from the SourceForge project site³ as a Windows “.msi” installer package. As of this writing, BBWin supports Windows Server 2003, Windows XP, Windows 2000, and Windows NT 4.0 (with some restrictions on the NT version).

The BBWin package installs into `C:\Program Files\BBWin`. Documentation and sample configuration files are supplied in subdirectories off of that path. Similar to the `hobbit-clients.cfg` and `client-local.cfg`, BBWin uses a `BBWin.cfg` file on each Windows node to control which local services, CPU utilization, disk utilization, and other metrics are relayed to the Hobbit server. The configuration file uses a specific XML grammar to define various monitoring parameters, including the DLL monitor agents to load (cpu, disk, memory, svcs, ...), the alert thresholds to monitor (warnlevel=“70%”), and the service names to track. Figure 3-5 shows part of a sample BBWin.cfg file.

Note: This is a partial configuration file meant to demonstrate aspects of the configuration process. The BBWin package supplies a fully-populated sample configuration file in `C:\Program Files\BBWin\etc\BBWin.cfg`

The BBWin documentation gives a full description of the various settings in the configuration file. For many configurations you can use the default values as supplied, or you may wish to change some of the threshold values to better reflect your operating environment. You will need to point the “`<setting name=“bbdisplay” value=“`” entry to your Hobbit server machine. This tells BBWin where to send monitored data for the local node.

BBWin monitors services such as a SAS Metadata Server through entries in the XML “`<svcs>`” block in the configuration file. A service is monitored by matching against a specific service name string. When you install SAS on your Windows system, you can define SAS servers as Windows Services using the *SAS Service Configuration Utility* (see “Starting SAS as a Windows Service” in the *SAS 9.1 Companion for Windows*, available online at http://support.sas.com/documentation/onlinedoc/91pdf/index_913.html).

The SAS Service Configuration Utility lets you specify the display name of the Windows Service used to identify the particular SAS server. For the Windows node described in the example BBWin configuration in Figure 3.5, the SAS Metadata Server was defined as a Windows Service named “SAS Lev1 MS”. The corresponding BBWin configuration line is:

```
<setting name="SAS Lev1 MS" value="started" autoreset="false" alarmcolor="red" />
```

This will notify the Hobbit server if the SAS Metadata service is not available. In this case, it will *not* attempt to restart the service automatically (autoreset=“false”). You can add configuration `<setting>` lines for any additional services that you wish to monitor.

³ BBWin – SourceForge.net (<http://sourceforge.net/projects/bbwin>)

```

<?xml version="1.0" encoding="utf-8" ?>
<configuration>
<bbwin>
  <setting name="bbdisplay" value="yourhobbit.yourcompany.com" />
  <!-- <setting name="bbdisplay" value="yoursecondbbdisplay:port" />-->
  <!-- BB Pager Part -->
  <!--<setting name="usepager" value="false" />
  <setting name="bbpager" value="yourfirstbbpager" />
  <setting name="bbpager" value="yourfirstbbpager" />
  <setting name="pagerlevels" value="red yellow" /> -->

  <!-- bbwin mode local or central -->
  <setting name="mode" value="local" />
  <setting name="configclass" value="win32" />

  <setting name="autoreload" value="true" />
  <setting name="timer" value="5m" />
  <load name="cpu" value="cpu.dll"/>
  <load name="memory" value="memory.dll"/>
  <load name="msgs" value="msgs.dll"/>
  <load name="svcs" value="svcs.dll"/>
  <load name="uptime" value="uptime.dll"/>
  <setting name="loglevel" value="3" />
  <setting name="logpath" value="C:\Program Files\BBWin\logs\BBWin.log"/>
  <!-- If true, the agent will report reporting failures as warning events -->
  <setting name="logreportfailure" value="false" />
</bbwin>
<cpu>
  <!-- If true, the agent will always report with green status -->
  <setting name="alwaysgreen" value="false" />
  <setting name="default" warnlevel="90" paniclevel="95" delay="3" />
</cpu>

<svcs>
  <!-- If true, the agent will always report with green status -->
  <setting name="alwaysgreen" value="false" />
  <!-- If true, the agent will restart all "automatic services" which would be
stopped-->
  <setting name="autoreset" value="false" />
  <setting name="alarmcolor" value="yellow" />
  <!-- The line bellow show you svcs rules. If the "Automatic Updates" service
is stopped, it will be automatically restarted and a red status will be sent
until the service get it's correct status -->
  <setting name="Automatic Updates" value="started" autoreset="true"
alarmcolor="red" />
  <setting name="Big Brother Hobbit Client" value="started" autoreset="true"
alarmcolor="red" />
  <setting name="SAS Lev1 MS" value="started" autoreset="false"
alarmcolor="red" />
</svcs>

<uptime>
  <setting name="delay" value="30m" />
</uptime>
</configuration>

```

Figure 3-5: Sample BBWin.cfg

3.6 – Using SAS to automatically generate Hobbit configuration files

One of the functions of the SAS Metadata Server is to act as a central, authoritative source for information on how to access other installed resources in your SAS BI hierarchy such as the SAS OLAP Server, SAS Stored Process Server, etc. The Foundation metadata repository contains objects describing each resource. For services, the object will contain connection information such as the machine name, service port, and other information used to gain access to the resource.

The Enterprise Management Integration group at SAS has developed a simple SAS program that will query a SAS Metadata Server Foundation repository for this resource connection information and will generate simple Hobbit configuration files to define monitors for the associated machines and service ports. The generated files can be added to your Hobbit configuration by means of the Hobbit “include” directive, or the resulting configuration information can be cut/pasted into your existing `hobbit-clients.cfg` and `bb-hosts` files.

The `HobbitAutoConfig.sas` program is available on the SAS Support website, under the Knowledge Base -> Focus Areas -> Enterprise Management Integration-> Hobbit Monitor page. (<http://support.sas.com/rnd/emi/Hobbit/index.html>).

At the beginning of the program source, you will need to modify the “metaserver” and “metaport” OPTIONS entries to reflect your environment. Run the program as an interactive SAS session. It will prompt you for the SAS metadata administrator user-ID and password, and will then query the metadata server to generate `hobbit-clients-sas-bi.cfg` and `bb-hosts-sas-bi` files for your BI configuration. The program does rudimentary error-checking.

Your questions, suggestions, and comments on this project are welcomed. Send e-mail mentioning “Hobbit” in the Subject: to

`emifedback@sas.com`

3.7 – Using SAS to automatically generate BBWin configuration files

In the case of a Windows-based server that is being monitored with a BBWin client, the necessary SAS configuration information is located in the Windows Services entries. This information can be queried from a Windows server using the Microsoft Windows Management Instrumentation (WMI) facility.

The Enterprise Management Integration group at SAS has developed a simple SAS program that will utilize this WMI functionality and query the Windows servers and determine the necessary <svcs> stanza entries to enable Hobbit to monitor the various SAS servers. These generated files (one for each Windows server) can be added to your Hobbit configuration by copying and pasting the block of <svcs> entries into your existing `BBwin.cfg` files on each of the monitored client machines.

The `BBwin_WinServices_Config.sas` program is available on the SAS Support website, under the Knowledge Base -> Focus Areas -> Enterprise Management Integration-> Hobbit Monitor page. (<http://support.sas.com/rnd/emi/Hobbit/index.html>).

At the beginning of the program source, you will need to modify the “`bbwin_loc`” macro variable to a writeable directory to create the configuration files. Run the interactive SAS program on each of the Windows servers that contains SAS servers. Invoke the SAS program with a userid that will have the capability to execute WMI queries on each of the servers. If the server contains Grid Management Service, the program will determine the necessary services for each of the configured Grid nodes. The program does rudimentary error-checking.

Your questions, suggestions, and comments on this project are welcomed. Send e-mail mentioning “Hobbit” in the Subject: to

`emifedback@sas.com`

Appendix A

setinit.sh

```
#!/bin/ksh

# setinit.sh
#
# Hobbit - SAS setinit monitor script
#
# (c) Copyright SAS Institute, Inc. 2007 All rights reserved.
#
#-----1-----2-----3-----4-----5-----6-----7-----8
#
# Scripts in the $HOBBITCLIENT/ext directory are only run if
# they are defined in the clientlaunch.cfg for the current host
# Add the following to your clientlaunch.cfg:
#
#[setinit]
#     ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
#     CMD $HOBBITCLIENTHOME/ext/setinit.sh
#     LOGFILE $HOBBITCLIENTHOME/logs/setinit.log
#     INTERVAL 1d
#
#
# Start of lines to put in $HOBBITCLIENT/etc/hobbitclient.cfg
#

# We need the GNU version of date and sed for date calculations
# Linux users can do:
#   GNUDATE=$DATE
#   GNUSED=$SED
#   SASPATH="/usr/local/sas/913/SAS_9.1/sas"
#   SASOPTS="-stdio -nofullstimer -nonews -nosource -nosource2 -pagesize max"
#
# Solaris and HPUX can do:
#   GNUDATE=/usr/local/bin/date;
#   GNUSED=/usr/local/bin/sed;
#   SASPATH="/usr/local/sas/913/SAS_9.1/sas"
#   SASOPTS="-stdio -nofullstimer -nonews -nosource -nosource2 -pagesize max"

#
# End of lines to put in $HOBBITCLIENT/etc/hobbitclient.cfg
#

#
# Test name: this will become a column on the display
# it should be as short as possible to save space...
# note you can also create a help for your test
# by modifying $HOBBITSERVER/etc/columndoc.csv. It
# be linked into the display automatically.
#
TEST="setinit"

# Leave null unless nondefault interval (5m) is needed
# or you can set it to 5m explicitly
INTERVAL="1d"

PIC="&green"
MSG=""

# Let's disable notes...can't do this globally as the setinit monitor
# depends on it being set.
SASOPTS="${SASOPTS} -nonotes"
```

```

# Set up the SAS code snippet
SASCODE="proc setinit;
        run;
"

# SELECT SOME LEVELS... GREEN IS THE DEFAULT...

WARN="14"  # Yellow if a license expires in WARN days
           # Gets reset to the default of the Grace period
           # unless grace period is less.
PANIC="7"  # Red if a license expires in PANIC days

get_status ()
{
#####
##### Setup some variables for use later
#####
COLOR="green"
LINE=""

# Check defaults have been set
if [ "$GNUDATE" = "" ]; then
    COLOR="yellow"
    GNUDATE=/usr/local/bin/date
    MSG="{MSG}"
    &$COLOR GNUDATE command is not defined in etc/hobbitclient.cfg - using default: $GNUDATE"
fi

if [ "$SASPATH" = "" ]; then
    if [ -x /usr/local/SAS/SAS_9.1/sas ]; then
        COLOR="yellow"
        SASPATH="/usr/local/SAS/SAS_9.1/sas"
        MSG="{MSG}"
        &$COLOR SASPATH command is not defined in etc/hobbitclient.cfg - using default: $SASPATH"
    else
        COLOR="red"
        MSG="{MSG}"
        &$COLOR SASPATH command is not defined in etc/hobbitclient.cfg - default does not exist:
$SASPATH"
    fi
fi

# USING SAS TO GRAB THE OUTPUT of PROC SETINIT
echo $SASCODE | $SASPATH $SASOPTS 2>&1 | $GNUSED -e 'ld' -e '/^

```

```

/d' -e '/^$/d' > $BBTMP/setinit.$$

# IF WE HAVE OUTPUT...
if [ -s "$BBTMP/setinit.$$" ]; then
    while read line; do
        (echo "$line" | $GREP "Expiration:") && EXPIRE=`echo "$line" \
            | $GNUSED -e 's/Expiration.*\([0-3][0-9]\.\{3\}[0-9]\{4\}\)\.$/\1/'` && MSG="{MSG}
    $line"
        (echo "$line" | $GREP "Grace Period") && GRACE=`echo "$line" \
            | $GNUSED -e 's/.* \([0-9]\+\) days.*/\1/'` && MSG="{MSG}
    &yellow $line"
        (echo "$line" | $GREP "Warning Period") && WARNING=`echo "$line" \
            | $GNUSED -e 's/.* \([0-9]\+\) days.*/\1/'` && MSG="{MSG}
    &red $line"

        if [ ! -z "$EXPIRE" -a ! -z "$GRACE" -a ! -z "$WARNING" ]; then
            break
        fi
    done < $BBTMP/setinit.$$

    if [ -z "$EXPIRE" -o -z "$GRACE" ]; then
        COLOR=red
        LINE="$LINE"
    &$COLOR SAS license on $MACHINE is invalid or already expired"
    else
        if [ $WARN -ne ` $EXPR $GRACE \+ $WARNING` ]; then
            WARN=` $EXPR $GRACE \+ $WARNING` # Yellow if a license expires in WARN days
                                                # Set to the default of the total grace period
        fi

        if [ $PANIC -ne $WARNING ]; then
            PANIC="$WARNING" # Red if a license expires in WARNING days
                                # Set to the default of the Warning period
        fi

        #####
        ##### Calculate license expiry dates.
        #####
        EXPIREEPOCH=` $GNUDATE --date="$EXPIRE" +%s`
        CUREPOCH=` $GNUDATE +%s`
        EXPIRESECONDS=` $EXPR $EXPIREEPOCH - $CUREPOCH`
        EXPIREDDAYS=` $EXPR $EXPIRESECONDS / 86400`
        # Add the total grace period before SID expires
        EXPIREDDAYS=` $EXPR $EXPIREDDAYS \+ $GRACE \+ $WARNING`

        if [ "$EXPIREDDAYS" -le "$PANIC" ]; then
            COLOR=red
            LINE="$LINE"
        &$COLOR SAS license on $MACHINE expires in $EXPIREDDAYS days"
        else
            if [ "$EXPIREDDAYS" -le "$WARN" ]; then
                if [ "$COLOR" = "green" ]; then
                    COLOR=yellow
                fi
                LINE="$LINE"
            &$COLOR SAS license on $MACHINE expires in $EXPIREDDAYS days"
            else
                LINE="$LINE"
            &$COLOR SAS license on $MACHINE expires in $EXPIREDDAYS days"
            fi
        fi
        unset EXPIREDDAYS

        MSG="
<center><b>SAS License Status</b></center>
    ${LINE}
    ${MSG}

` $CAT $BBTMP/setinit.$$`"
        fi
    else
        COLOR="clear"
    fi

```

```
fi
    export COLOR
}

get_status $MACHINE >/dev/null 2>&1

# NOW USE THE BB COMMAND TO SEND THE DATA ACROSS
if [ "$INTERVAL" = "X" ]; then
    $BB $BBDISP "status $MACHINE.$TEST $COLOR ` $DATE` ${MSG} "
else
    $BB $BBDISP "status+$INTERVAL $MACHINE.$TEST $COLOR ` $DATE` ${MSG} "
fi

# Clean up our mess
# Checking for existence of each file since the whole test may be optional
# and may not actually run on every client
#
if [ -f $BBTMP/setinit.$$ ]; then
    $RM $BBTMP/setinit.$$
fi
```

