

BigSister

Configuring and installing BigSister

COLLABORATORS

	<i>TITLE :</i> BigSister	<i>REFERENCE :</i>	
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Thomas Aeby, Didier Bringer, Joerg Fritsch, and Alexander Kerr	October 28, 2009	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME
1.0	11 December 2000	Initial release.	tom
1.01	March 2002	Docbook format	dda
1.01	March 2002	When describing commands, I tried to write arguments like in any Programmation Book ie command { mandatory args } [optional arguments] Skins for CGI	dda
1.5	August 2003	Done further work on the Docbook format. Introduced Big Sister logo. Introduced html chunks. Rewritten some parts in order to get them taylored to Big Sister rev. 0.98b3 and give the reader a better understanding.	jf
1.5.1	August 2003	Corrected english wording, grammar and style	ak
1.9	November 2005	Did some major improvements to reflect the current status of the project	ta

Contents

1	License - The GNU General Public License	1
1.1	Preamble	1
1.2	TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION	1
2	Installation	5
2.1	Prerequisites	5
2.2	Big Sister Components	5
2.3	Installing Big Sister	7
2.3.1	Paths to files	7
2.3.2	Installation from sources	7
2.3.3	Installing Big Sister from the RPM-packages	9
2.3.3.1	Fresh install	9
2.3.3.2	Upgrading from previous versions	9
2.3.4	Installing Big Sister on Debian	10
2.3.4.1	Download vs. APT	10
2.3.5	Installing Windows binary	10
2.4	Post-installation tasks	10
2.4.1	Configuring your webserver (Big Sister server only)	10
2.4.2	Installing Perl modules	11
2.4.3	Installing RRDTool	11
2.5	Install Big Sister plugins	12
3	Agent Configuration	13
3.1	First steps	13
3.2	How to start the daemons at boot time	13
3.3	Adding new hosts and services	14
3.3.1	The uxmon-net file	14
3.3.1.1	Multiple uxmon-net configuration files	15
3.3.1.2	The uxmon-asroot configuration file(s)	15
3.3.2	Network objects with simple names	15
3.3.3	Network objects with alias names	15

3.3.4	Health checks	16
3.3.4.1	Defining the scope of monitored systems: DESCR	16
3.3.4.2	Common syntax	17
3.3.4.3	Pointing the agent to your server	17
3.3.4.4	Check frequencies	18
3.3.4.5	Defaults	19
3.3.5	Using self contained healthcheks: testers	19
4	Server Configuration	20
4.1	Server security	20
4.1.1	Server vulnerability and server access	20
4.1.2	Server access: /etc/bigsisiter/permissions	20
4.1.3	Data integrity and authenticity	21
4.1.4	Tunneling Big Sister tcp-connections through Secure Shell	22
4.1.4.1	Agent to Server	23
4.1.4.2	Server to Server	23
4.2	Configure Alarming	23
4.2.1	Rules	23
4.2.2	Patterns - "when to do things"	24
4.2.2.1	Simple rule	24
4.2.2.2	Using wildcard in rules	24
4.2.2.3	Using groups in rules	25
4.2.2.4	Using criteria in rules	25
4.2.3	Description - "what to do"	25
4.2.4	PAGER rules: influencing alarm delivery	26
4.2.5	Definitions and their meaning	26
4.3	Use and configure SLA/Availability management	26
4.3.1	Setting it up Availability management	28
4.3.1.1	Running the reporting tool	29
4.3.1.2	The results	30
4.3.1.3	Using savelogs/archivelogs and availability management	30
4.4	Sending server commands	30
4.4.1	Using the telnet client	31
4.4.2	Using the bsadmin tool	31
4.5	Configuring the Server and customizing the Display	32
4.5.1	The main server configuration file: bb-display.cfg	32
4.5.2	The options section	32
4.5.3	The names and group section	33
4.5.3.1	Defining and using groups	33

4.5.3.2	Automatically joined groups	33
4.5.3.3	Deleting groups	35
4.5.4	The Webpages section	35
4.5.4.1	Skins	37
4.5.4.2	Skins in CGI programs	38
4.5.4.3	Frames	38
4.5.5	Using imagemaps	38
4.6	Deleting unused and stale network objects	39
4.7	Attaching the Big Sister server to a database	40
4.7.1	Enabling data logging in a database	40
4.7.2	Data logging in MySQL database	41
4.7.2.1	Prerequisites	41
4.7.2.2	Creating the database and database user	41
4.7.2.3	Configure Big Sister	41
4.7.3	Data logging without a database server: the file database	42
4.7.4	Database Structure	42
5	Appendices	44
5.1	Question and Answers : Q+A	44
5.1.1	Trend graphs do not appear though I followed the instructions on setting them up	44
5.1.2	Can I have some graphics showing disk/memory/cpu/... usage during the last few days?	44
5.1.3	No links to my performance data trend graphics are appearing on the web pages	44
5.1.4	Links to trend graphics appear but when I click on it no graph is displayed	44
5.1.5	After installing Big Sister my display works but I do not see any status from my clients	44
5.1.6	Some of my web pages look quite poor (no background, no lamps) while others look fine	45
5.1.7	The History and Alarms menu entries do not work	45
5.1.8	CGIs are so slow	45
5.1.9	The .conn (ping) test does always report failure	45
5.1.10	My display.history file is growing larger and larger	46
5.2	SNMP Agent	46
5.2.1	Big Sister as an SNMP trap source	46
5.2.2	MIB	46
6	Command Reference	50
6.1	testers	50
6.2	bsadmin	52

7	New-style Health Checks	54
7.1	accesstime	54
7.2	disk	55
7.3	diskload	56
7.4	etherport	57
7.5	ftp	58
7.6	http	58
7.7	ica	59
7.8	icmp_ping	60
7.9	ident	60
7.10	imap	61
7.11	ldap	61
7.12	load	62
7.13	mailq	63
7.14	memory	63
7.15	myexpedap	64
7.16	mysql	65
7.17	network	66
7.18	nntp	67
7.19	ntp	67
7.20	nutups	68
7.21	oracle	69
7.22	pop3	69
7.23	procs	70
7.24	procsname	71
7.25	readbandwidth	72
7.26	realhttp	73
7.27	service	73
7.28	smtp	74
7.29	snmpvar	75
7.30	ssh	75
7.31	tcp	76
7.32	telnet	77
7.33	users	77
7.34	writebandwidth	78

8	Old-style Health Checks	80
8.1	cpupload	80
8.2	statusfile	80
8.3	bbscript	81
8.4	http	81
8.5	tcp	81
8.6	ping	82
8.7	rpc	83
8.8	procs	83
8.9	diskfree	84
8.10	diskload	84
8.11	load	84
8.12	dumpdates	85
8.13	syslog	85
8.14	eventlog	85
8.15	bbdisplay	86
8.16	snmp	86
8.17	OV	87
8.18	metastat	87
8.19	ntp	87
8.20	mrtg	88
8.21	atmport or etherport	88
8.22	software	89
8.23	realhttp	89
8.24	memory	90
8.25	oracle	90
8.26	tripwire	92
8.27	ldap	92
8.28	command	93
8.29	ups	93
8.30	qmqueue	94
9	Server Command Reference	95
9.1	join	95
9.2	leave	95
9.3	displayname	96
9.4	status	96
9.5	page	96
9.6	savelogs	97
9.7	sendlogs	97
9.8	perf	97
9.9	remove	98
9.10	event	98

10 Server configuration reference: bb-display.cfg	99
10.1 %Option	99
10.2 %Autoconn	100
10.3 %Autojoin	100
10.4 %Autojoin pattern	101
10.5 %Port	101
10.6 %Pager	101
10.7 %Groups	102
10.8 %StatusTrap	102
10.9 %Page	103
10.10%Section	103
10.11%title	103
10.12%ref to	104
10.13%itemref	104
10.14%table	105
10.15%sort	105
10.16%select	106
10.17%select_names	106
10.18%select_items	106
10.19%Itemgroup	107
10.20%column_wrap	107
10.21%image	108
10.22%ref	108
10.23%skin	109
10.24%Logskin	109
10.25%rsync	109
10.26%frameset	110
10.27%include	110

List of Figures

2.1	Big Sister Components	6
3.1	Host Aliases	16
3.2	Agent to server	18
4.1	Status Changes result in Alarms	24
4.2	Group hierarchy example	34
4.3	Database Structure of Big Sister 1.01	43

List of Tables

2.1	paths to files	7
4.1	Accepted keywords and function	21
4.2	The alarm generator knows more definitions than just mail and pager settings	27
4.3	Files used for Availability management	28
4.4	Output of the default statistics file	30
4.5	Accepted keywords and function	31
4.6	Some available skins	38

Abstract

Abstract This is the official manual for the Big Sister network monitor. This documentation is intended for guiding new users during their first steps setting up agent nodes, configuring health checks and getting the display server up and running.

Chapter 1

License - The GNU General Public License

1.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. copyright the software, and
2. offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

1.2 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

Section 0

This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a 'work

based on the Program' means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term 'modification'.) Each licensee is addressed as 'you'.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does. *Section 1*

You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee. *Section 2*

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of [Section 1](#) above, provided that you also meet all of these conditions:

- a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License.

Exception:

If the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License. *Section 3*

You may copy and distribute the Program (or a work based on it, under [Section 2](#) in object code or executable form under the terms of [Sections 1](#) and [2](#) above provided that you also do one of the following:

- a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)
-

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code. *Section 4*

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance. *Section 5*

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it. *Section 6*

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License. *Section 7*

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License. *Section 8*

If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License. *Section 9*

The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation. *Section 10*

If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally. *Section 11*

NO WARRANTY

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION. *Section 12*

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Chapter 2

Installation

2.1 Prerequisites

The first step in getting a working installation of Big Sister is to download the source package or one of the binary release packages from <http://www.bigsister.ch/download/> or from the alternate download site at sourceforge.net http://sourceforge.net/project/showfiles.php?group_id=14120. At the time this manual was written there was a choice between

- The source package
- Big Sister for Windows systems
 - zip archive containing non-compiled but otherwise ready-to-use Big Sister installation (ActivePerl interpreter needed)
 - zip archive with completely pre-compiled binary distribution, where the required Perl interpreter is basically wrapped around the perl-sources in a binary form.
 - MSI package with binary distribution not requiring Perl to be installed, ready for installation via the Windows installer
- RPMs for Linux systems (independent from any specific platform or distribution)
- Debian packages

If you are not using the binary distribution (Windows Platforms only), a pre-installed and running Perl interpreter with Perl version ≥ 5.6 is *always* needed to run Big Sister. For the Big Sister server, additionally a webserver is required. You can certainly choose the webserver which suits your needs best. However, a suitable extension to the webserver configuration file is only supplied for the apache webserver (httpd).

2.2 Big Sister Components

Big Sister comes in three parts:

- `bigsister-[some rev. number].rpm`, this is the Big Sister base package which you will always need to install, whether you are going to install a Big Sister server or an agent.
 - `bigsister-server-[some rev. number].rpm`, this is an additional configuration file (`bb-display.cfg`) whose presence will make your installation feel and behave like a server. This configuration file must only be present on the server, otherwise you will run into problems.
 - `bigsister-agent-[some rev. number]`, this is an additional configuration file (`uxmon-net`) whose presence will make your installation feel and behave like an agent node. If it is desired that besides being a Big Sister server, the server is monitoring itself and its resources, this file should be present on the server also. It does not destroy anything and you can't run into problems.
-

A network monitored with Big Sister (a "Big Sister Environment") is composed of a Big Sister server and several agent nodes (see figure 1.1). The server is responsible for the storing, processing and visualisation of the data collected or received from the monitored systems. A "Big Sister Server" actually runs the Big Sister Server (bbd), the Big Sister Monitor (bsmon) and the Apache httpd. Bbd is responsible for the communication with the agents including checks if agents are permitted to do certain operations. Bbd then passes all the information it gets from the agents to bsmon as is. Bsmon processes this information and builds status pages, does alarming, etc.

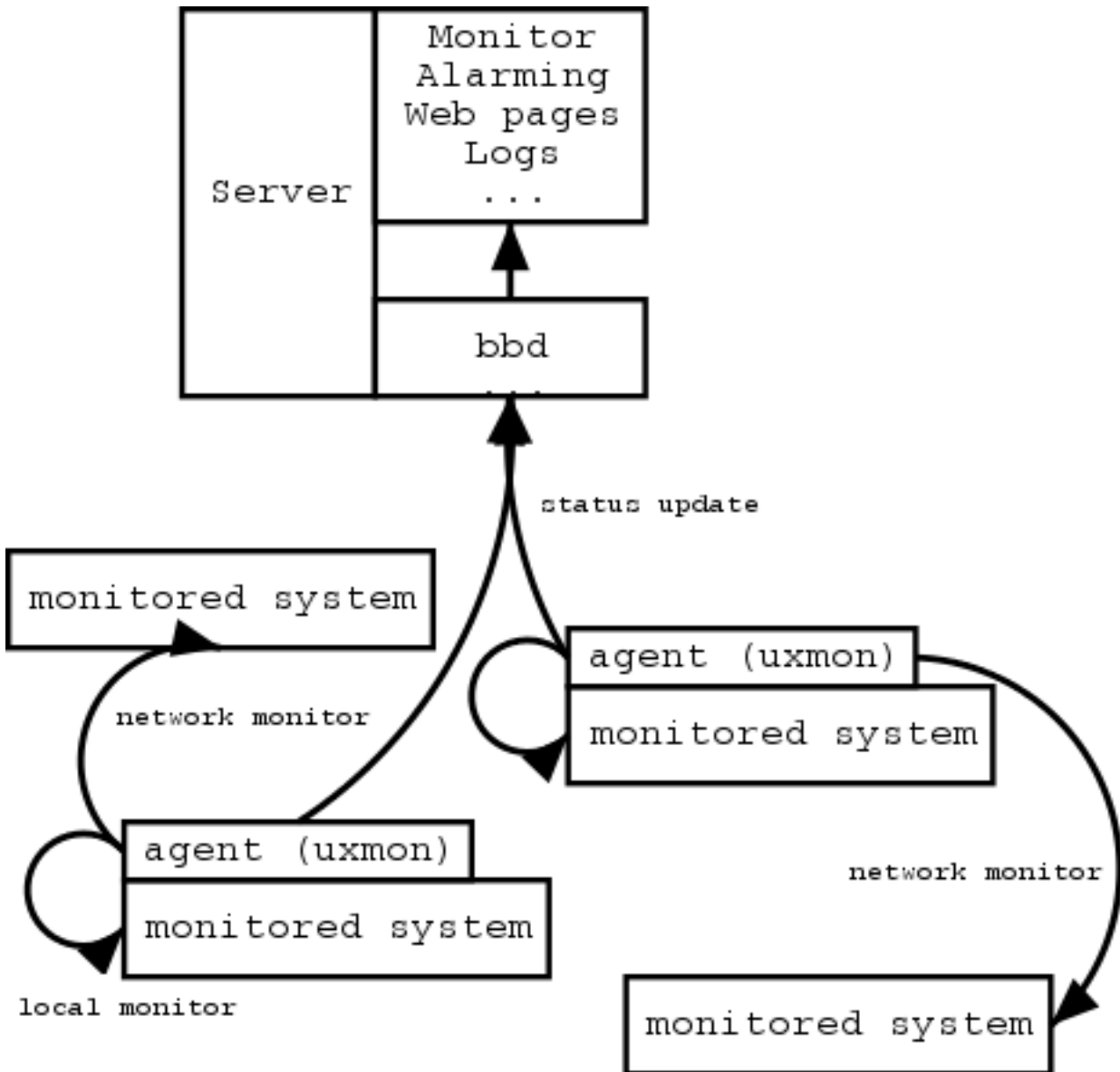


Figure 2.1: Big Sister Components

Every monitored system or service is monitored by a Big Sister Agent (uxmon). Some of the healthchecks performed are only applicable to the system hosting the agent while others are done via the network. Usually you will install agents on all the systems you want to monitor provided there is an agent implementation available for a particular system. Systems with agents installed will collect the specified data about their own state and send it to the Big Sister server for further processing. By default this is done with a *tcp* connection to *port 1984* on the server machine. Please keep in mind that you have to open *tcp* port 1984 on your firewall in case the agent(s) and the servers are on network segments separated by a local firewall or access-lists.

Note

In order to simplify your firewall rulebase and add an additional security level through encryption to your data connections, you can use a [Secure Shell \(ssh\) tunnel](#).

On systems which are not capable of running an agent (like switches, routers etc.) or where the installation of agent software is not desired (like web servers at remote sites), every agent node can also do remote healthchecks via the network. Remote healthchecks can be done (for example) by accessing OIDs via SNMP or http-requests for certain content. Only a limited set of checks are available via the network though. *For more information on the currently available healthchecks and their usage read the [Healthcheck Command Reference](#).*

It is possible to use the Big Brother agent as a replacement for the Big Sister agent (e.g. for Netware). However, some features of Big Sister (e.g. performance data collection) will not be available in this case.

2.3 Installing Big Sister

2.3.1 Paths to files

Big Sister puts its binaries and data in specific directories. Depending on your installation the paths pointing to these directories will differ. If you build Big Sister from source you are free to move most of these directories to the location you prefer. The Windows packages allow you to relocate the whole Big Sister package (the Root directory) but not each of the directories with special meanings. The RPM packages come with predefined Big Sister root-directory {ROOT} as well as other predefined paths (given in the table below) which fit nicely into the the filesystem layout of common Linux distributions.

What	Default (when build from source)	Windows	RPM / Debian
Root	/usr/local/lib/bs	C:\bigsis	/usr/share/bigsis
Binaries	{Root}/bin	{Root}\bin	{Root}/bin
Agent Binaries	{Root}/uxmon	{Root}\bin	{Root}/uxmon
Web Pages	{Root}/www	{Root}\www	/var/lib/bigsis/www
CGI Scripts	{Root}/cgi	{Root}\cgi	/var/lib/bigsis/cgi
RW-Files	{Root}/var	{Root}\var	/var/lib/bigsis
configuration files	{Root}/etc	{Root}\etc	/etc/bigsis
Release Notes and HOWTO	/usr/local/lib/bigsis/doc		/usr/share/doc/packages/bigsis
Default User Account	bs		bigsis
Big Sister URL	/bs	/bigsis	/bigsis
CGI URL	/cgi/	/cgi/	/bigsis/cgi/

Table 2.1: paths to files

The Big Sister- and the CGI-url are not physical paths - rather they are the path where the Big Sister HTML-Display and its CGI programs appear when accessed from the outside world via the web server.

2.3.2 Installation from sources

When installing Big Sister from the source package you will additionally need at least a working copy of the Make utility and the Bourne shell. This means that you will probably not be able to install Big Sister from source directly on a Windows machine and this section therefore just assumes you are installing on a Unix box.

Before you can proceed you should decide where to store your various Big Sister files (see section **). If this is your first installation it might be a good idea to just use the defaults. You should also consider a few points:

- The Web-Pages directory must be accessible via a web browser, probably you will run a web server for this purpose
- Other directories than the Web-Pages directory should not be accessible via a web server for security reasons

- Big Sister will excessively write to the Web-Pages and var directories. They should be located on a fast drive.
- Config files are split into two directories: the adm- and etc- directory. The idea behind this is to keep files shared (e.g. via software distribution) between different installations in etc- and the files local to the respective machine in the adm directory.

Usually you will install Big Sister under its own user account for security reasons. Big Sister daemons will run under this account. Before you can install Big Sister you have to create this account.

Are you still with us? Well, then we can start the installation. Unpack your Big Sister source package, log in as root and change directory into your package. Now you can run the configuration procedure by changing directory into the Big Sister package directory and run

```
./configure
```

Configure will try to use reasonable defaults, but it is a good idea to browse through its configuration options and look for things you want to have configured differently. Please run

```
./configure --help
```

to see all the supported options. Especially useful options in my eyes are:

```
--prefix=[dir]
                (install Big Sister elsewhere than in /usr/local)
```

```
--with-user=[login]
                (use some other user than the default "bs")
```

```
--with-cgi=[URL]
                (the URL of the directory your browser will find the Big Sister CGIs ←
                 under "/cgi")
```

```
--with-url=[URL]
                (the URL of the directory your browser will find the "www" directory of ←
                 Big Sister (defaults to "/bs"))
```

The RPM / Debian packages for instance are built using the following configure options:

```
./configure --enable-fhs --with-user=bigsis --with-cgi=/bigsis/cgi --with-url=/bigsis
```

Once you have run configure and you are happy with the configuration it generated you can run

```
make install
```

install the whole Big Sister package or

```
make install-agent; make install-modules
```

just install the agent, or

```
make install-server; make install-modules
```

just install the server.

Note that Big Sister allows installation of additional plugins. Many features even of the official distribution are implemented in their own plugin. The additional install-modules target used above ensures that all the plugins being included in the distribution are installed as well.

During the installation process a Big Sister boot script will be placed in the "init" directory of your system (/etc/init.d, /sbin/init.d, /etc/rc.d/init.d, whatever else). It's up to you to enable this script by placing the necessary links in the rc*.d directories or using chkconfig (if your system supports this).

2.3.3 Installing Big Sister from the RPM-packages

2.3.3.1 Fresh install

There is not much of a difference between the installation of an agent node and a Big Sister Server. In fact they differ only in two files located at `/etc/bigsister`. A Big Sister Server always needs the `bb-display.cfg` file to make it feel like a server. Agent node always need the `uxmon-net` file. The start script `/etc/init.d/bigsister` checks which file is present and starts the required daemons. In order to avoid any confusion and accidental installation of several servers in one network, the base rpm contains none of the above mentioned configuration files. The installation process using the RPMs is as follows: first you install the base package containing all the binaries, scripts and so on. Afterwards you install an additional RPM package containing the required configuration file to make your base installation a server or an agent node.

The following guidelines assume that you are using the `rpm` command line interface and not any GUI like `gnorpm` or others. If you are doing a fresh install,

```
rpm -i bigsister-[some rev. number].rpm
```

should do the job and install the base package. Afterwards you should use

```
rpm -i bigsister-agent-[some rev. number].rpm
```

to install an agent node or

```
rpm -i bigsister-server-[some rev. number].rpm
```

to make it a Big Sister server.

2.3.3.2 Upgrading from previous versions

Note

Please read the Release Notes of every new version before doing an upgrade.

Updating Big Sister is usually as simple as doing a

```
rpm -Uhv bigsister-[some rev. number].rpm
```

If your previous installation was done from the sources, a

```
make
```

with the same arguments you originally used should do the job. Files that are user-editable (config files, etc.) won't get overwritten, binaries will though.

Warning

Some things you should be aware of:



- never install Big Sister for running as a different user to your previous version (well, you'll have to do a 'chown' manually in this case)!
 - config files are usually compatible between releases. Anyway if you'd like to use new features you'll usually have to update your config files of course. And of course config files for newer releases won't necessarily work with older releases.
 - it's always a good idea to back up a working Big Sister installation before upgrading
 - never upgrade an installation from sources with an rpm and the other way around!
-

2.3.4 Installing Big Sister on Debian

2.3.4.1 Download vs. APT

The current version of Big Sister is always available via two ways: the deb packages are available for download on the Big Sister web server like every other package. Those who prefer using APT or one of its derivatives may wish to directly put the line

```
deb http://software.graeff.com/debian-graeff stable main
```

into their `/etc/apt/sources.list` file. You can then install Big Sister using

```
apt-get update
apt-get install bigsister-server bigsister-agent
```

or any other tool similiar to apt like aptitude, synaptic and the like.

If you prefer downloading the .deb-Package from the download server then you will install Big Sister from the downloaded packages with the command

```
dpkg -i bigsister_[some rev. number].deb
```

and one of or both

```
dpkg -i bigsister-agent_[some rev. number].deb
dpkg -i bigsister-server_[some rev. number].deb
```

2.3.5 Installing Windows binary

The Big Sister package for Windows has undergone massive changes during its history. At the time this document was written the first test-release of an MSI (Windows Installer) version of Big Sister was available. In future this will be the way binary distributions for Windows will be made.

In order to install the MSI package you download the package file and double-click it when being logged in with administrator priviledges. The Windows installer will start and present you a few options. The package will automatically register Big Sister as a service. However, the necessary configuration files are all installed as files with the extension `.default`. You have to browse through the `adm` directory and copy the default files into their active pendant.

During installation you will be asked if the Big Sister Agent should be run under the Administrator or LocalSystem account. The user priviledges of the LocalSystem account prevent the agent from accessing some network resources like mapped network drives. The Administrator account is less limited in this respect and might be the preferred choice if you need the agent to run health checks like the file access bandwidth.

If you are installing some non-MSI distribution please follow the instructions in the release notes accompanying the distribution or on the Big Sister web server.

2.4 Post-installation tasks

2.4.1 Configuring your webserver (Big Sister server only)

This only needs to be done on your Big Sister server. Currently the Big Sister start script `/etc/init.d/bigsister` does not check the availability and configuration of a webserver on the same system which is used to display any processed data.

In order to save you some time, each installation procedure leaves you with a `httpd.conf` snippet in the `/etc/bigsister` directory which is taylored with the paths and options you might have specified during the installation process. However, the Big Sister installation routine will not touch your web server configuration in any way. It relies on you to include this snippet into your standard `httpd.conf` by appending the line

```
Include /etc/bigsister/httpd.conf
```

This will

- make the documents in the `.../www` directory available by the specified url
- execute the CGI scripts on demand

2.4.2 Installing Perl modules

Note

If you are using the binary Windows package you already have all the necessary Perl modules implicitly.

If you can bare working with ActiveState Perl (a Windows Perl*interpreter*), it comes with its own easy way of installing modules (see <http://www.activestate.com/>).

For Linux systems many modules are available via RPM packages from your distributor. If you haven't got your Linux distribution handy or if a certain perl module seems to be missing, you can always get them from CPAN and follow their installation directives. You will find a list of CPAN mirrors on <http://www.cpan.org/>.

You might have to install additional perl modules to enable some Big Sister health checks or certain display functionality. For more information about specific health checks and their inter-dependancies please see under the appropriate section in chapter **. For general use, the following modules are highly recommended:

- SNMP - All the SNMP functionality in the agent and the server bases on Simon Leinen's SNMP module available from <http://www.switch.ch/misc/leinen/snmp/perl/>). The module is included in all Big Sister versions >0.99, so there is no need to install it manually any more.
- GD - Big Sister can present system status overviews as a graphical image map. This functionality bases on the GD module available from CPAN (install it on the server system).
- Net::SMTP - Alarming is usually done via E-Mail and the sendmail program. If you prefer your Big Sister server to directly transmit alarm mails via SMTP - e.g. because you are running Big Sister on a non-Unix system - you will have to install the Net::SMTP module available from CPAN on your server system.
- LWP::UserAgent and URI - are required for the realhttp test monitoring Web servers
- Crypt::SSLeay - is required for monitoring SSL-enabled Web servers

All of the above modules - except for the GD module - are easy to install. Just follow the instructions in the respective module. Please check carefully if the required module is already installed on your system before you touch your (running) perl installation. The command `perldoc perlmodlib` gives you a somewhat verbose list of all perl modules installed on your system. The command `perldoc [module name]` can be used to check for single modules and get even more verbose documentation (in case the requested module is installed).

Note

On most systems these commands cannot be used by the root user.

2.4.3 Installing RRDTool

Big Sister is able to collect performance data, store it in a database and provide you with nice trend graphics. Currently only one database exactly designed for such purposes is supported: Tobi Oetiker's RRDTool. For this feature to work you will have to download and install RRDTool. At the time this manual was written it was not necessary to install the RRDTool perl modules - having the `rrdtool` command installed in your path was enough. Anyway it is a good idea to install these modules. In the near future they might improve the performance of your server.

RRDTool is available from <http://ee-staff.ethz.ch/~oetiker/webtools/rrdtool/>

2.5 Install Big Sister plugins

Many of Big Sister's features have been moved out of the core implementation into plugins. During installation all of the commonly used plugins have been installed. You can upgrade existing plugins and install additional plugins via the

```
bsmodule
```

command. List the installed plugins with

```
bsmodule list
```

and install or upgrade modules with

```
bsmodule install [module-name]
```

The available plugins can be found on <http://www.bigsister.ch/plugins.html>. Unless you download the respective module file from there, `bsmodule` will automatically contact the software download server and try to download a module with the name you specified.

Chapter 3

Agent Configuration

3.1 First steps

Are you curious? Just want to start it up and see what's going on? Ok, lets go. Simply start the server by issuing

- ```
/etc/init.d/bigsister start
```

on unix systems

- Start Big Sister Monitor services on Windows systems

The server should immediately start listening to the default agent/server port (TCP port 1984) and create the initial web pages in the www directory. Try starting a Web browser and point it to the newly generated index.html file.

Now you can proceed to configure and bring up agents. First you should configure all agent nodes to point to your Big Sister server by editing the `uxmon-net` file. Locate the line `localhost bsdisplay`, replace `localhost` by the name (or the IP address) of the system hosting your Big Sister server and start the agent. On Windows systems start the Big Sister Agent service via the Control Panel / Service Manager, on Unix systems issue

```
/etc/init.d/bigsister start
```

---

#### Note

For more background information why this is the same command as on your Big Sister server please read the section "Installing Big Sister from the RPM-packages" subsection "Fresh Install".

---

Now go back to your web browser and reload the viewed pages. If you are using the `bb-display.cfg` initially installed on the server the agents should automatically join the Groups NEW, UNIVERSE and ALL and appear on the status page (this may take a few minutes).

Try the links Admin, Alarms and History to see if the CGI configuration of your web server is right.

### 3.2 How to start the daemons at boot time

---

#### Note

This section talks about how to get the Big Sister daemons started each time the box is rebooted. This is handy for ensuring the agents are always running.

---



Under Windows use the Control Panel / Service Manager to tell your system which Big Sister services it should startup on boot. See section 1.2 for determining what services should run on which systems.

With Unix the startup of Big Sister daemons is a little different. There is a *perl script* named `{ROOT}/bin/bb_start` meant to work like a System V init script, thus running

```
bin/bb_start start will start up the Big Sister daemons while bin/bb_start stop will shut them down and b-
in/bb_start restart will restart Big Sister.
```

Since the init directories should only contain *shell scripts* and no *perl scripts* a little shell script has been installed in your init directory which is basically a wrapper for the above mentioned perl script. Therefore you can create the necessary links in your `rc*.d` directories to the wrapper script `/etc/init.d/bigsister`. If you are running Big Sister on RedHat Linux you can use the `chkconfig` command to enable / disable Big Sister.

`bb_start` will look at the config files in the `/etc/bigsister` directory for deciding whether it should act like an agent or a server.

## 3.3 Adding new hosts and services

### 3.3.1 The uxmon-net file

The agent (uxmon) will read its configuration from the file `/etc/bigsister/uxmon-net`

A good and well structured uxmon-net file consists of:

- Global **DEFAULT** values (for certain arguments)
- **DESCR** iptions of the monitored systems, so the the subsequent healthchecks know what environment they can expect and which features are available on the tested platform
- The **healthchecks** together with the required parameters

---

#### Example 3.1 A simple uxmon-net configuration file

---

```
Set the default SNMP community to "public", the
default frequency is 1/5min anyway
DEFAULT community=public frequency=5 perf=5 ALL

Set the default version and protocol for rpc checks to "1" and "udp"
DEFAULT version=1 proto=udp rpc
DEFAULT proto=icmp ping

DESCR features=unix,linux,redhat,redhat71 localhost
DESCR features=unix,linux,suse,suse70 susix
DESCR features=unix,linux,redhat,redhat71 rejectix
DESCR features=unix,linux,redhat,redhat72 publizistix-ext

localhost perf=30 disk memory
localhost cpuload syslog
localhost fs=.0(20-60) dumpdates dns
localhost proc=sendmail min=1 max=30 procs

susix ping perf=5 service=smtp tcp
susix dns ntp

rejectix ping dns

publizistix-ext (publizistix) ping dns
publizistix-ext (publizistix) perf=5 service=smtp tcp
publizistix-ext (publizistix) perf=5 service=ssh tcp
```

---

### 3.3.1.1 Multiple uxmon-net configuration files

Under Unix multiple uxmon-net files may exist - their name must start with uxmon-net followed by an arbitrary suffix (note that e.g. uxmon-net.bak is a valid agent configuration file!). `bb_start` will in this case start up an instance of uxmon for each of these configuration files.

### 3.3.1.2 The uxmon-asroot configuration file(s)

Under Unix, the access to some (socket) commands might be prohibited for ordinary users and thereby for the `big sister` account. I.e. on some systems only the root user can access icmp socket commands to send icmp pings. For such cases, Big Sister offers the option to create a special configuration file named `uxmon-asroot`. For the healthchecks configured in the `uxmon-asroot` file, `bb_start` will start an uxmon instance running with root privileges.

As for the `uxmon-net` configuration file, multiple `uxmon-asroot` files with a name starting with `uxmon-asroot` and an arbitrary suffix can be used.

## 3.3.2 Network objects with simple names

---

### Note

A resolvable name or an IP address (reverse lookup is not required) is a necessity for the agent (uxmon) to find the system on which the health checks need to be done. It is fundamental to understand that this name does not need to be identical with the name under which the data collected from this host is processed and grouped by the Big Sister server. In some cases it might be convenient to use the hostname for both the agent and the network object name in the Big Sister server. In other cases a somewhat finer granularity might be desired.

---

The syntax of the `uxmon-net` file is kept very simple: each entry starts with the name or IP address of the system to be monitored, followed by a list of health checks that should be applied to this system, e.g.:

```
localhost fstype=ext2 disk
```

will create a network object named like the output of `/bin/hostname` on the Big Sister server and run the `diskfree` test against all mounted partitions holding an `ext2` file system of the local system. In the HTML status display on the server all test results will be grouped under the name of the network object. A working name resolution on the agent node, either through the `/etc/hosts` file or DNS, can be considered a prerequisite when using hostnames in the `uxmon-net` file.



### Warning

In the `uxmon-net` file either `localhost` or the output of `/bin/hostname` can be used to point checks to the local system. Unless no alias name (see section [Network objects with alias names](#)) is specified, the gathered data will be reported to a network object in the Big Sister server which has the system's hostname!

---

### 3.3.3 Network objects with alias names

Sometimes it is necessary to differ between the host you are running a check against and the name of the network object which is reported to the server. For instance imagine you are running a check against a multihomed host (a host with multiple IP addresses) and you want to access this target system via a well-defined network interface. In this case you can use a configuration line like the following:

```
192.168.1.17(myhost) ping
```

This will run the `ping` test against `192.168.1.17`. The result of the test however is then reported to be related to the network object `myhost` (see figure 2.2).

A host definition of the form

---

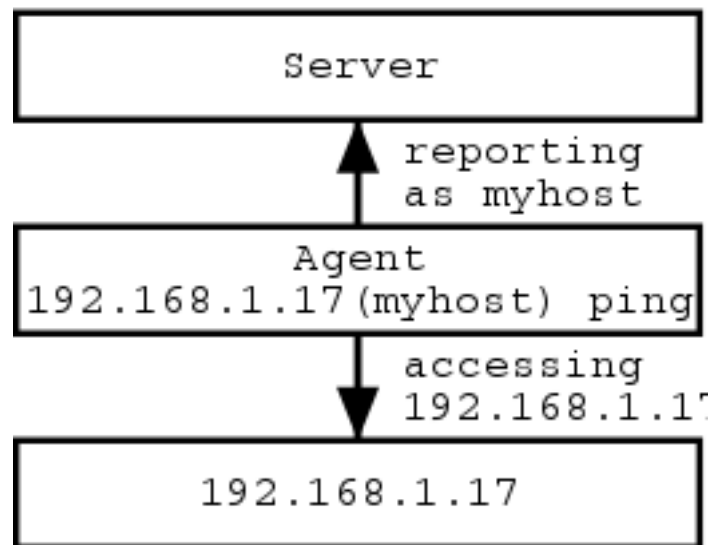


Figure 3.1: Host Aliases

```
name1 (name2)
```

is always treated by `uxmon` in the following way: `name1` is used internally by the agent while `name2` is the name reported to the Big Sister server. The server will be completely unaware of the `uxmon` internal name (`name1`).

#### Tip

The alias associated with a physical host can be chosen on a per healthcheck basis. Thus, healthcheck results can be displayed on the status pages under different logical host names. For instance, for a physical host A system health could be reported as A-system, while application healthchecks report under A-applications, making it possible to group check results on the Big Sister display in both a system/hardware and an application view.

### 3.3.4 Health checks

Big Sister is using intelligent health checks to ensure system, application and content availability.

- System availability can be ensured i.e. through remote ping checks on (one of) the system's NICs. Disk capacity, errors in syslog and other parameters can be made available for further processing through agent software (`uxmon`) installed on the monitored system.
- Application availability can be ensured through an agent on the system which is running the application or, if it is a network application through binding to the tcp-service port. Full path healthchecks which go beyond checks of the tcp-service port exist for some network applications i.e. mail-, web- or DNS-servers.
- Content availability might be desired especially on webservers. This special healthcheck tests if a specified URL is available.

#### 3.3.4.1 Defining the scope of monitored systems: DESCR

This section applies for Big Sister rev. 0.98beta6 or later. This has been developed for somewhat broader use, so that Big Sister can differentiate between *device types* (computer, router, switch and others) and tailor healthchecks for the *features* coming with the specified system. Currently Big Sister is using only one device type (computer).

The methods used for monitoring a certain computer depend on the operating system and whether it is the localhost or a remote system. I.e. an `uxmon` agent on the localhost will monitor the sendmail service by checking if there is at least one daemon

running. For testing a remote system a connection to tcp port 25 will be established and await certain patterns or return codes. A well formed configuration file will have one DESCR statement for every monitored system (see also [A simple uxmon-net configuration file](#)).

Big Sister knows the following features:

```
aix
bsd
hpux
linux
macos
netware
remote
local
solaris
sysv
tru64
true64
unix
windows
```

---

**Note**

The *remote* and *local* features are automatically chosen by uxmon depending on whether the target host is considered the host the agent is running on or some remote host. Adding remote or local explicitly to the features list in uxmon-net will override this automatism.

---

### 3.3.4.2 Common syntax

uxmon-net entries may span multiple lines. Usually a line end will automatically end the respective configuration entry. However if a line ends with a \ character the following line is assumed to be part of that entry also. If a '#' character is found all the characters behind # are treated as a comment.

Most of the checks accept arguments. Arguments are always preceding the check and are of the form

```
argument1=value1 argument2=value2 ... check
```

The argument list only applies to the check immediately following the last argument in the list, thus

```
myhost proto=icmp ping ping
```

will run two ping checks against the host myhost, the first one will do an ICMP ping, while the second will do a ping using the default protocol (usually UDP). The proto argument does not influence the second ping check, but of course you can do something (rather senseless) like

```
myhost proto=icmp ping proto=icmp ping
```

*You will find a complete and hopefully up to date list of available health checks with their arguments in the [reference part of this documentation](#)!*

### 3.3.4.3 Pointing the agent to your server

Two special pseudo-checks in the uxmon-net file point your agent to the server(s) that status reports should be sent to: the `bsddisplay` and `bbdisplay` checks. The line

```
myserver bsdisplay
```

for instance will force the agent to send status information to the server myserver where myserver talks the Big Sister protocol. You can use uxmon in conjunction with a Big Brother server by changing the above line into

---

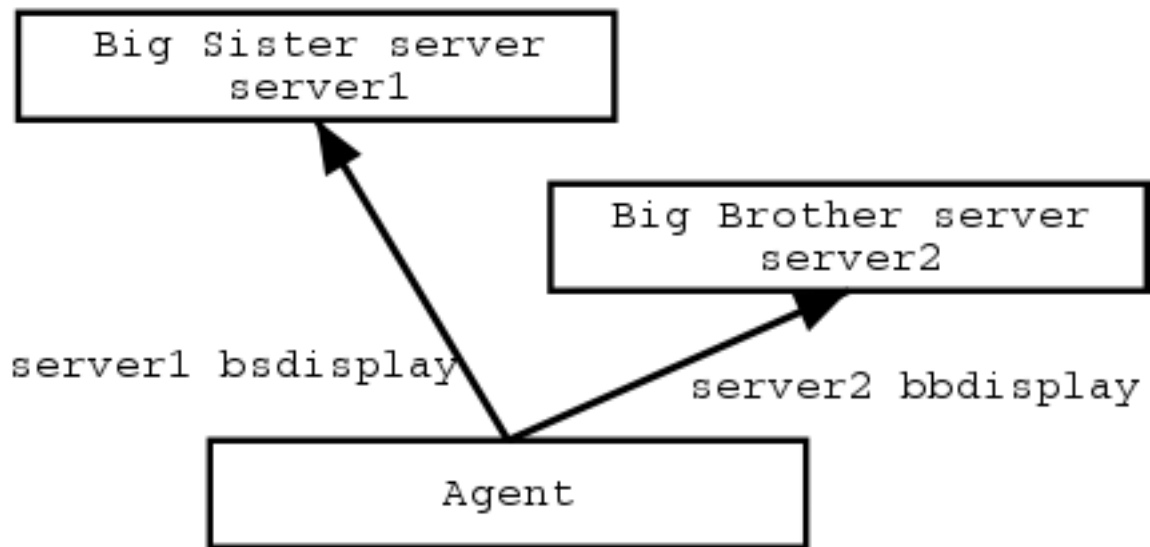


Figure 3.2: Agent to server

```
myserver bbdisplay
```

In this case the agent will suppress any non Big-Brother feature and use the Big Brother protocol to talk to the server (see [figure 2.2](#))

Of course multiple `bsdisplay` / `bbdisplay` lines may appear in `uxmon-net`. In this case the agent will report its status information multiple times to (potentially) different servers.

As one would expect the `bsdisplay` pseudo-check accepts a few arguments, e.g. the line

```
myserver fqdn=no bsdisplay
```

will report status information to `myserver` by stripping domains from all the host names.

Other useful arguments which are relevant if you want Big Sister to keep statistics on performance data and are listed in section ??.

### 3.3.4.4 Check frequencies

By default `uxmon` runs checks and reports information in 5 minutes intervals. Any-way, some checks might put some load on the target system, or are of no short-term relevance and you prefer to run them less frequently. Other checks might be of extreme importance and should be run more often. For such occasions you can define your own check frequencies. Every check (including the `bsdisplay` and `bbdisplay` pseudo-checks) accept the special argument `frequency`. E.g. the `uxmon-net` line

```
localhost frequency=180 metastat
```

```
importantmachine frequency=1 ping
```

will run the `metastat` check against the local machine every 180 minutes while the `ping` check is run every minute.

---

#### Frequencies

Note that the argument name `frequency` is a little bit misleading - it is not really a frequency but rather the time interval in minutes between two runs of a check. Big Sister 1.00 or later supports fractional frequencies making a configuration like `frequency=0.5` valid.

---

When defining check frequencies keep some rules in mind:

- Running a check more often than the fastest `bsdisplay` pseudo-check is senseless. Check results will only be reported during `bsdisplay` runs, not necessarily immediately after each check s run. So the above example only makes sense if you have got for instance the following `uxmon-net`:

```
localhost frequency=180 metastat
importantmachine frequency=1 ping
myserver frequency=1 bsdisplay
```

- You must run `bsdisplay` checks at least every 10 minutes. The server relies on the agents to report their status rather frequently. If no status information is coming in for 15 minutes the server will assume the agent or communication to the agent is dead and will set status to purple (no report).
- The above rule does not apply to other checks. Even if you run certain checks only every few hours the `bsdisplay` check will report the result of the last check run. So the server will not change status to purple.

### 3.3.4.5 Defaults

Proceeding to more complex `uxmon-net` files you will probably get bored by repeatedly listing the same check arguments again and again. Fortunately `uxmon` supports setting global defaults for certain arguments. For instance the configuration:

```
localhost frequency=2 type=ext2 diskfree
localhost frequency=2 memory
localhost frequency=2 procs=sendmail procs fileserver frequency=5 nfs
myserver frequency=2 bsdisplay
```

can be simplified to

```
DEFAULT frequency=2 ALL
DEFAULT type=ext2 diskfree
localhost diskfree
localhost memory
localhost procs=sendmail procs fileserver frequency=5 nfs
myserver bsdisplay
```

the `DEFAULT` statements in this example do

- set the default check interval for all (ALL) the checks to 2 minutes
- set the default type for all `diskfree` checks to `ext2`

Of course you can override defaults by just listing an argument with a non-default value as usual. For instance in the example above the interval for the `nfs` check is explicitly set to 5 minutes overriding the default interval of 2 minutes.

### 3.3.5 Using self contained healthcheks: testers

As of Big Sister rev 0.98 so called "self contained" healthchecks do exist. Self contained healthchecks are coupled with the `testers` command which can be accessed from the command line interface.

The `testers` command is used to query self contained healthcheks for the syntax and arguments available *on the system where the command is issued*. By default, the output is given in ascii text. Depending on the commandline options and software installed on your system, XML and man page are also valid output formats.

The `testers` command is located under the `{bigsisiter-root}/bin/` directory. Please read the section on [testers in the reference part of this manual](#) for a detailed description.

## Chapter 4

# Server Configuration

### 4.1 Server security

When talking about server security we need to consider two issues:

- server vulnerability
- integrity and authenticity of the collected data

#### 4.1.1 Server vulnerability and server access

Agent nodes communicate with the Big Sister server via a TCP based protocol (see sections [Sending server commands](#) and [Server Command Reference](#)). Whenever an agent needs to send status/group updates it connects to a dedicated port (defaults to 1984), sends one or more commands and then disconnects. The server will never acknowledge client commands, the protocol (currently) is one-way only.

Because of using the TCP/IP protocol stack for agent - server communication, the Big Sister server has some potential to be exploited by a hacker. Overall the server is considered rather secure since

- it is written in Perl and therefore does not suffer from buffer overrun problems
- it uses a simple protocol and therefore validity checking on incoming requests is rather trivial
- the functionality available through the server protocol is rather limited



#### **Warning**

Nobody will guarantee that there is absolutely no way of hacking Big Sister! In order to minimize your risk you should harden your server system (uninstalling any unnecessary software and fancy stuff including all unused daemons and network services) and limit the remote access to your server to systems who are running Big Sister agents.

---

#### 4.1.2 Server access: `/etc/big sister/permissions`

The permission file tells bbd which clients are allowed to connect and which operations they may perform. The file is read line by line. Each line contains both a pattern and a list of operations accepted or rejected for the matching clients. If a client matches multiple patterns the associated access lists are treated in a cumulative way and applied in the order they appear in the file.

The format of each line is

```
pattern => access list
```

---

accepted patterns are:

name [hostname] client name matches 'name'

ip [ip] client IP address matches 'ip'

anonymous no user is logged on on this connection

---

**Note**

name [name] patterns are ignored if the DNS option in bb-display.cfg (see above) is switched off. This is the default!

---

The access list is a list of keywords being associated with an operation or a group of client operations. Each keyword is preceded by either a "+" or a "-" character allowing or rejecting corresponding request. Accepted keywords are:

| Keyword      | Function                                                       |
|--------------|----------------------------------------------------------------|
| all          | all operations                                                 |
| authenticate | client is permitted to send user authentication                |
| status       | client is authorized to send status messages                   |
| page         | client is authorized to send page commands                     |
| grouping     | client is permitted to send group join/leave and name commands |
| archiving    | log file archiving operations                                  |
| alarm_acking | alarm acknowledging operations                                 |
| perf         | performance data transmission                                  |
| remove       | removal of staus records                                       |

Table 4.1: Accepted keywords and function

Empty lines and lines starting with '#' are ignored. Example:

```
name .* => -all
default - reject everything
#
hosts in mydomain.com may only send status
messages
name .*\.mydomain\.com => +status
#
archiver may only do archiving
name archiver => -all +archiving
#
localhost may do everything
name localhost => +all
#
unauthenticated clients must never page
anonymous => -page
#
selected hosts may send grouping information
name group1 => +grouping
name group2 => +grouping
```

### 4.1.3 Data integrity and authenticity

Data gathered by network monitoring software like Big Sister does not need to be considered sensitive enough for taking the trouble to ensure full integrity, authenticity and maybe even privacy.

Without any means auf authentication, a nasty user can at least annoy you by generating masses of false alarms, faking wrong system status, injecting wrong history data, etc.

A client is identified in two ways:

---



- by the host it is running on
- $\Sigma$  by the user it is authenticating *asunimplemented yet*

**Warning**

The second identification is not implemented yet - this makes less secure but at the same time things get really easy.

There are only a few ways how to identify a host: Either you identify it by its name or by its IP address. To make rules more powerful you can use wildcards in both cases. For instance

```
host 192.168.* => +all
```

will permit access to every host in the 192.168 network, while

```
host *.microsoft.com => +all
```

will do the same for every host in the microsoft.com domain.

**Note**

Note that the *strings are perl regular expressions* which are slightly different from any other wildcarded patterns!

To clarify if you are trying to match a name or an IP address it is always a good idea to use ip or name instead of host:

```
ip 192.168.* => +all name *.microsoft.com => +all
```

**Example 4.1** Example of refusing a status from certain machines

```
host .* => +all host 192\.168\.0\.88 => -status host
192\.168\.0\.89 => -status
```

**Note**

Actually Big Sister tries to mangle IP addresses, so that 001.002.003.004 works equally well as 1.2.3.4

**Warning**

This is in contrast with normal IP addressing rules, where a number starting with a 0 is considered octal, so that 010.020.030.040 will be interpreted as 8.16.24.32

**Tip**

Using host name to address translation via DNS or some other directory service on a monitoring machine is often a bad idea: The monitoring application should especially work when there are problems around - and there is some risk that these problems prevent name resolution from working. Therefore it is a good idea to avoid names in the permissions file and to set *%Option -DNS* in the server configuration file.

#### 4.1.4 Tunneling Big Sister tcp-connections through Secure Shell

It is very simple to build a secure (encrypted) tunnel for these status connections using Secure Shell (ssh) and thereby adding at least a basic security level to your environment.

#### 4.1.4.1 Agent to Server

On the host running the agent start up ssh forwarding, e.g.:

```
ssh -n displaymachine -L 10192:localhost:1984 sleep 600
```

(This will forward connections to 10192 to displaymachine port 1984 for 10 minutes)

In uxmon-net use the following server entry:

```
localhost port=10192 bsdisplay
```

#### 4.1.4.2 Server to Server

Use ssh the same way as above. In bb-display.cfg use the Rsync statement like e.g.:

```
%Rsync mastermachine:10192 prefix GROUP
```

## 4.2 Configure Alarming

Big Sister implements alarming in a *server based* manner. The agent is responsible for determining if a system or service is working correctly ("green"), if it is critical ("yellow") or it has failed ("red") - other stati do exist but are not relevant to alarming.

This status is noticed by the alarming module of the server. Depending on the configuration file `bb_event_generator.cfg` the server generates alarms on status changes.

The alarming configuration mainly consists of a set of rules. Each rule consists of a pattern matched against all status change, a definition of dependencies and a description of the action to be taken when an alarm is raised. The first two elements describe under what circumstances an alarm is to be raised while the last one describes how actually the alarm is raised.

Using this simple approach a few things can easily be configured either for individual checks, for individual hosts or for whole groups:

- wait for a defined time period before reporting an alarm and do not report an alarm if the problem goes away within this period
- regularly send reminders telling the administrator that a problem persists until the problem goes away
- do not repeatedly send alarms for a multiply occurring problem
- behave different depending on time of day or day of week (e.g. postpone alarms raised during the night to the early morning)
- suppress alarms depending on what status other systems/services are in (e.g. do not report that a system is unreachable when Big Sister already knows that the whole network the system is connected to is down)

The main disadvantage of the existing rule based alarming configuration is that it is very hard to find a simple way to explain how it works. Unfortunately you will just have to read the whole section and hopefully understand the configuration at the end.

### 4.2.1 Rules

An alarming rule in the `bb_event_generator.cfg` file always starts with a pattern followed by a description describing what actions should be taken if the pattern matches.

Whenever a status change is detected, `bb_event_generator.cfg` goes through the config file and looks for matching patterns. Each variable associated with the matching patterns is then set as described. If multiple patterns are matching the associated variables are set in order.

Every time a status change is noticed the alarm generator does two things:

- go through the pending alarms and check if the status change has some effect on one of them

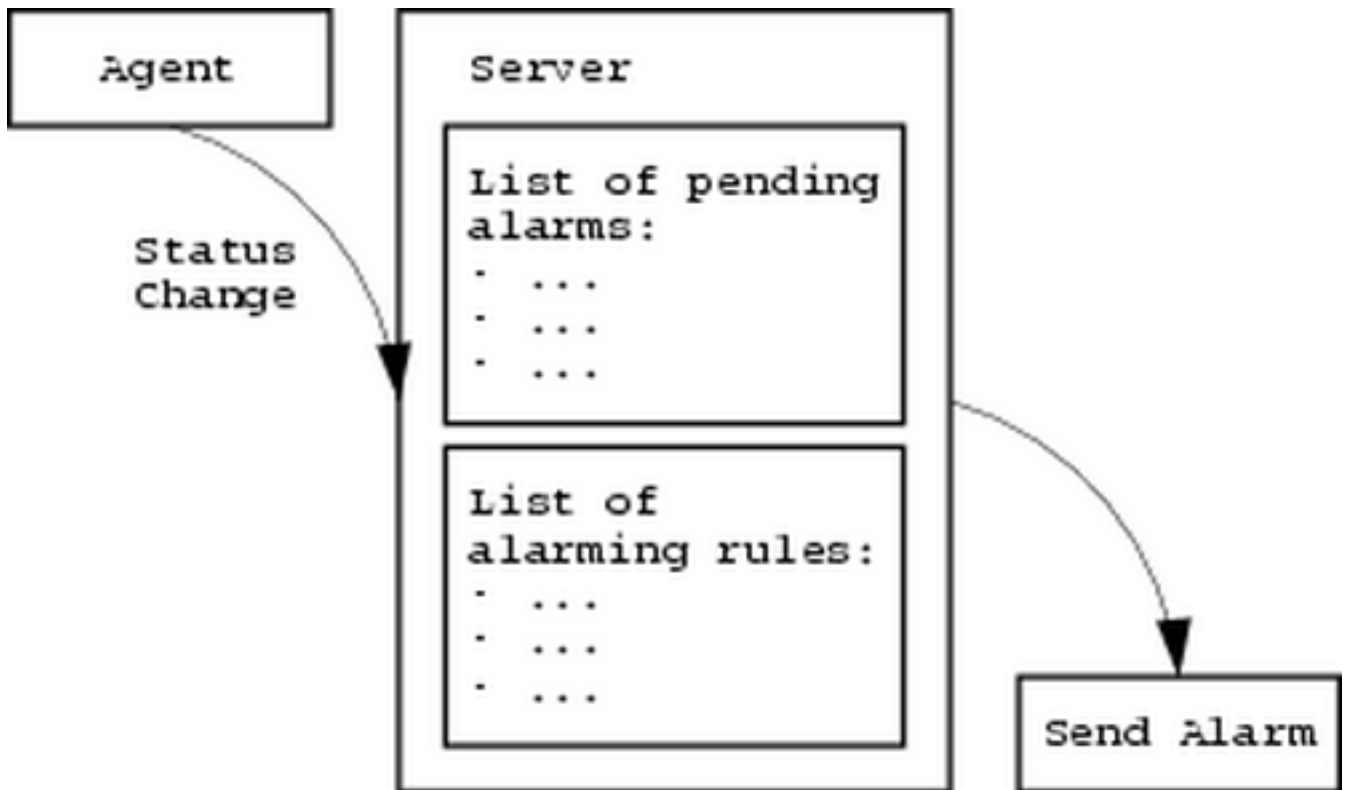


Figure 4.1: Status Changes result in Alarms

- if the status change is not related with one of the pending alarms: go through the list of rules, select all the matching rules and raise an alarm depending on their descriptive part

Usually each line in the configuration file represents one rule. Of course like in most Big Sister configuration files empty lines and lines starting with a '#' character are treated as comments and are therefore simply ignored. A rule may span multiple lines: Lines terminated with a '\ ' character are joined with their following line.

## 4.2.2 Patterns - "when to do things"

The most simple form of a pattern is a `host . check` pattern.

### 4.2.2.1 Simple rule

```
foo.cpu mail=nobody
```

(where `foo.cpu` is the pattern and `mail=nobody` is the description) for instance matches only status changes for the host `foo` and the `cpu` check. The above rule tells Big Sister to forward alarms for `foo.cpu` to the user `nobody`.

### 4.2.2.2 Using wildcard in rules

```
*.cpu mail=nobody
```

Now let's assume you do not want to list each individual system and check in the rule file. The Alarm Generator accepts one single wildcard - `*` - matching any check or any host

```
foo.* mail=nobody
```

And for matching any status change for any host.

```
 . mail=nobody
```

#### 4.2.2.3 Using groups in rules

Of course you may want to address a group of hosts - haven't you spent hours setting up groups after reading section 2.2.3? Exactly these groups are also visible to the alarm generator.

By prefixing a host name with a '@' character you point Big Sister to match a group rather than a single host so that a rule like

```
@USA.* mail=nobody
```

for instance applies to any status change reported for any system being member of the group USA.

#### 4.2.2.4 Using criteria in rules

So far so good. Sometimes it is very useful to be able to make alarming behave different depending on when a status change is detected - maybe you just refuse to be woken up by your beeper during the night or you want get alarms via another medium during working hours.

For this purpose the patterns can contain so-called pre-conditions. In the rule

```
@USA.*{weekday Sat,Sun} mail=pikett
```

the stuff in parenthesis is a pre-condition. The rule will only match status changes for any system being member of the group USA reported during the weekend. Another useful precondition is the daytime condition. This rule

```
.{daytime 17:00-07:00} down=never
```

for instance will suppress (down=never) any status change reported between 5pm and 7am. Of course conditions can be combined using and and or, so

```
.{daytime 17:00-07:00 or weekday Sat,Sun} \ down=never
```

will suppress any status change reported between 5pm and 7am or during week-end.

### 4.2.3 Description - "*what to do*"

Associated with each pattern there is a description in the form of a bunch of definitions. This set of definitions describes what actually will be done if a status change matching the pattern occurs. The rules will be processed in the order they appear in the configuration file and if multiple patterns match all the definitions will cumulate.



#### Warning

Definitions appearing later in the file will overwrite definitions appearing earlier

#### Example 4.2 Example default values

```
. mail=alarm@nowhere.org delay=5
*.cpu delay=100
```

If a status change for myhost.conn is reported then only the first pattern will match resulting in a description of:

```
mail=alarm@nowhere.org delay=5
```

while if a status change for myhost.cpu is reported both patterns will match and the resulting description would look like:

```
mail=alarm@nowhere.org delay=20
```

thus the mail definition will be taken from the first rule while the delay definition of the second matching rule will replace the concurring definition in the first rule.

---

#### Tip

It is a good idea to place more general rules near the start of the configuration file and more specific rules near the end. E.g. a rule associated with the pattern `.*` is working like default settings since it will match every single status change.

```
.* mail=alarm delay=5 down=yellow up=green prio=5
```

Placed at the very start of the configuration it will initialize the settings for mail, delay, down, up and prio. Later rules may re-set one of these settings by at the same time inheriting all the other settings.

---

### 4.2.4 PAGER rules: influencing alarm delivery

Version 0.98 introduced special rules allowing us to modify the way how alarms are delivered. E.g. the rules

```
PAGER{$mail eq "someaddress@somehost"} pager=myscript mail=someaddress
PAGER{$pager eq "sendmail" and $mail eq "test"} mail=addr1,addr2,addr3
```

will re-direct alarms sent to `someaddress@somehost` to the address `someaddress` and invoke `myscript` for sending the alarm (first rule). If the pager equals `sendmail` and the target address is `test` the alarm is redirected to the three addresses `addr1,addr2` and `addr3`.

The PAGER-Rules are applied once per target address, thus the above rules would also apply if the page originated e.g. from a

```
.* mail=someaddress@somehost,test pager=sendmail
```

(this would cause the event generator to apply the PAGER rules twice: once for `"someaddress@somehost"`, once for `"test"`)

---

#### Note

Target addresses are split into "pager" and "mail" before PAGER rules are processed. An address like

```
mail=sendmail:test@somewhat.strange
```

will appear to the PAGER rules as `mail=test@somewhat.strange pager=sendmail`

---

### 4.2.5 Definitions and their meaning

## 4.3 Use and configure SLA/Availability management

The following instructions show you how you can use the Big Sister log files to work out for how long a certain network device has been up/down during a certain time range. The result (usually a value like `m.n%`) can be used to evaluate if a certain device or service meets our expectations or not. I.e. it is quite common to negotiate a guaranteed availability (in %) of a leased line. This is called an SLA. Usually carriers give discounts or pay even penalties if they couldn't provide the negotiated availability.

There are two approaches to understanding reporting. First of all, there are two fundamental commands called `report_read` and `report_consolidate`.

`report_read` reads in a specified file with status information or dependency rules for one specific day and stores the results in `var/reportdb/day-yyyy-mm-dd.statuslog`. Multiple runs of `report_read` – most probably you will at least read in the `display.history` and some dependency definitions - may incrementally update the same file in `var/reportdb`. One of the functions of `report_read` is the "Cumulator" which builds `var/reportdb/*cumu*` files out of the `statuslog` files by applying service hours / holidays definitions to the status information. Usually you will have multiple `cumu` files since you are interested in seeing statistics for multiple service levels. The files generated by `report_read` are actually of minor interest to you – they just serve as a kind of cache in order to reduce time spend in daily statistics operation. Another effect of this caching is that you do not need storing status information (`display history`) for the whole time period you are interested in.

---

| Setting     | Purpose                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| mail        | the recipient address of an alert message. Multiple addresses are separated by comma, each address may be prefixed by a pager                                                                                                                                                                                                                                                                                        |
| upmail      | specifies the recipients of the message that is being sent when the alarm condition is cleared ("up"-Mail). If not specified upmail defaults to the same recipient list as specified in the mail definition                                                                                                                                                                                                          |
| prio        | priority level (0..100), this is not used by the alarm generator any more, instead you can use it in conditional rules                                                                                                                                                                                                                                                                                               |
| repeat      | if set, the alarm generator will send the alert message again all specified minutes until the alarm condition has cleared                                                                                                                                                                                                                                                                                            |
| repeatprio  | same as for prio, but is the priority level for repeated alerts                                                                                                                                                                                                                                                                                                                                                      |
| keep        | the number of minutes the alarm is not cleared by the alarm generator after the alarm condition is telling us that a service is up again                                                                                                                                                                                                                                                                             |
| norepeat    | the number of minutes new alerts for the same condition are suppressed                                                                                                                                                                                                                                                                                                                                               |
| delay       | the number of minutes between when the alarm condition is noticed by the alarm generator and an actual alert message is sent. If the alarm condition clears within this delay, no message is sent                                                                                                                                                                                                                    |
| check       | a boolean expression that is checked during the delay time interval. If the check condition is evaluating to false at least once during that interval, no alert message is sent                                                                                                                                                                                                                                      |
| down        | one out of "green", "purple", "yellow", "red", "never": tells the alarm generator which status should be seen at as "down". I.e. "yellow" means that if a service status goes yellow or below (red), then the corresponding service is to be considered down                                                                                                                                                         |
| up          | like down. Tells the alarm generator which status should be considered "up". I.e. "down=yellow up=green" will mean that a service is considered down from the time when it changes to yellow or red to the time when it goes to green again (but not if it is merely changing into purple)                                                                                                                           |
| maxmsg      | a numeric value which is the maximum size of the shortened alarm message used in the subject of the alert                                                                                                                                                                                                                                                                                                            |
| postpone    | if set, alarms shall not be sent for additional x minutes and rather stay in the queue. If during the postpone time period the alarm condition is cleared the alarm is silently dropped without ever sending an alert message. I.e. Postpone is meant to be used during night when you do not want to get woken up by alert messages but you want to get an alert as soon as you get to work if the problem persists |
| postpone_to | is basically the same as postpone, but the value is a time of day rather than a time interval (i.e. "06:00")                                                                                                                                                                                                                                                                                                         |
| pager       | use specified pager program rather than the default (sendmail)                                                                                                                                                                                                                                                                                                                                                       |
| skin        | use the skin specified here for alert messages                                                                                                                                                                                                                                                                                                                                                                       |
| trap        | if set, the alarm generator will send a trap to the specified trap destination whenever sending alert/acknowledgement messages. The value of trap should be something like community@host                                                                                                                                                                                                                            |
| maxpermin   | limit the maximum number of alert messages being sent in one minute to this value. When this limit is exceeded alert messages are silently discarded (but the alarm is handled and appears on the alarming page). As of rev 1.01 this defaults to 20                                                                                                                                                                 |

Table 4.2: The alarm generator knows more definitions than just mail and pager settings

`report_consolidate` then will go through all the `cumu` files in a specified time period, sum up time spent in specific status and create report files

```
var/reportdb/day-yyyy-mm-dd.statistics.cumuclass.name
```

for each cumulated file (`texttreport_read`) and each defined time period. These files are what you actually want to get.

### 4.3.1 Setting it up Availability management

Theory sounds rather complex, doesn't it? Let's go ahead to the real world then. In order to simplify the use of the reporting module an additional command `report_day` has been added. *Usually you will just forget about `report_read` and `report_consolidate` and just use `report_day` which tries to do a mixture of `report_reads` and `report_consolidates` itself.* As its name implies `report_day` is meant to be run on a daily basis. It will build statistics based on the following files:

| File                                | Purpose                                                                                                               |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| <code>display.history.*</code>      | Big Sister status history                                                                                             |
| <code>reporting/servicehours</code> | defining when system are expected to be working                                                                       |
| <code>reporting/holidays</code>     | defining holidays (aka. time periods when systems are notexpected to be in service)                                   |
| <code>reporting/cumulators</code>   | defining which service levels should be reported (linking servicehours and holidays to status information)            |
| <code>reporting/override</code>     | is meant to store manually maintained status information overriding <code>display.history</code>                      |
| <code>reporting/dependencies</code> | telling us which services should be watched and how they depend on status information in <code>display.history</code> |
| <code>reporting/statistics</code>   | defining which time periods should be consolidated and what exactly we would like to see in the resulting report      |

Table 4.3: Files used for Availability management

Best is to start with the simple things: `servicehours` and `holidays`. These files are rather self explaining – in `servicehours` you can define "in service" hours for each day of the week, while in `holidays` you can exclude whole days from service time. Note that the first column in each of these files contain a "class" specifier. This allows to define multiple levels of service – e.g. some systems might to be expected to run 24 hours 7 days a week while others are only expected to run from 8:00 till 17:00 on Monday through Friday. Define classes for all these service levels.

`Servicehours` and `holidays` will not be effective on their own. The rules actually linking service levels with status information are listed in the `cumulators` file. In this file you actually define your service levels based on `servicehours` and `holidays`.

Each rule looks like

```
levelname = service:class > holidays:class
```

Do not mind if you do not really understand what exactly ">" means. "levelname" is a symbolic name you can freely choose - at the very end you will get report files carrying "levelname" in their names.

It is time now for the more complex things: `dependencies`. In a real world you are usually not interested in seeing statistics for simple things like `myserver.conn` or `myserver.smtp` – in the SMTP case for instance you probably have multiple redundant mail servers increasing the overall mail service availability. So the mail service is available if any of your mail servers is up and running. In the `dependencies` file you tell the reporting modules which dependencies apply to your systems. The above rule would maybe look like:

```
mailservice = history:myserver1.smtp | history:myserver2.smtp
```

telling that `mailservice` is up if at least one of `myserver1.smtp` and `myserver2.smtp` is up. Note the leading "history:". Every information holder's name in the reporting tool is preceded by a prefix like e.g.:

```

history: - display.history information
service: - servicehours information
comp: - dependencies
holidays: - holidays information
override: - override information

```

Let's assume that the mail servers above depend on DNS to work correctly. So you define a DNS rule (let's assume you run two redundant DNS servers dns1 and dns2):

```
dns = history:dns1.dns | history:dns2.dns
```

Now you can use the result of the dns rule to make your mailservice rule more realistic:

```
mailservice = comp:dns \
(history:myserver1.smtp | history:myserver2.smtp)
```

do not be fooled by the fact that the rule is now put on two lines – this is just for readability (the at the end of line one tells the reporting module the rule will be continued on the next line). Note that "dns" is in fact "comp:dns" – dependencies get automatically prefixed with "comp:".

Note that the resulting statistics will only contain services defined via the dependencies mechanism. Also, dependencies with names starting with "\_" are not appearing in the final output file – you can use such names as "internal variables".

Of course some times your monitor will fail and report nonsense (e.g. because your agent or server got cut off). In this case you need a means for manually correcting such mistakes. This is done via the "override" mechanism. E.g. add the following line to the override file:

```
mailservice,28.6.01 12:00,29.6.01 09:00,green,I know it worked
```

saying that mailservice was completely ok from June 28 12:00 till June 29 09:00. This line on its own does not yet change the statistics result. You have to set up your dependencies accordingly. The full mailservice rule should then look something like

```
mailservice = comp:dns \
(history:myserver1.smtp | history:myserver2.smtp) \
> override:mailservice
```

The ">" (override) operator tells us to prefer the expression on the right side of ">" to the expression on the left side for the whole time period(s) the expression on the right side is defined. In other words: whenever override:mailservice is defined the whole expression basing on history status is ignored and overridden by override:mailservice – actually this is probably what you guessed a long time before.

There is only one thing left you need to setup before we can get the first statistics: the statistics file. In this file you specify what "things" should be reported and which time periods the statistics should cover. As for now we can go along with the default file. It does report:

Note again that only dependencies (names starting with comp:) will appear in the final output. So as long as you do not define dependencies you will get no results. A dependency can of course be as simple as

```
comp:myrouter = history:myrouter.conn
```

#### 4.3.1.1 Running the reporting tool

The easiest way to run the reporting tool is via the report\_day wrapper, e.g.:

```
bin/report_day
```



|                |                                                                                      |
|----------------|--------------------------------------------------------------------------------------|
| Down           | time (secs) a service was red                                                        |
| N/A            | time (secs) a service was purple                                                     |
| Up             | time (secs) a service was yellow or green                                            |
| Planned Outage | time (secs) a service was white                                                      |
| Service Hours  | time (secs) a service should have been in service (according to the cumulators file) |
| Down%          | Down/Service Hours                                                                   |
| Up%            | Up/Service Hours                                                                     |
| Availability   | The services's availability                                                          |

Table 4.4: Output of the default statistics file

`report_day` is meant to be run every day and will by default read in `status history / servicehours / holidays` for the last two days and `dependencies / cumulators / overrides` for the last 10 days.

When running `bin/texttreport_day` for the first time you might want to compute statistics for a longer time back, do it with e.g.:

```
bin/report_day -h 30 -c 30
```

```
-h: read history / servicehours / holidays 30 days back, -c: read the rest of the files ←
 also 30 days back
```

You will see that this will take some time because `report_day` will compute each individual day separately. This is very inefficient for long time periods since the whole reporting system is optimized for daily (incremental) use.

#### 4.3.1.2 The results

You will get a bunch of files in `var/reportdb`. The most interesting files are `day*.statistics.class.period`. E.g. a file

```
day-2001-06-28.statistics.level1.oneweek
```

contains the statistic for the time from June 22 to June 28 2001 and the service level `level1`.

Note that though you defined `oneweek` to consolidate status for 7 days the reporting tool will create a statistics file for every day containing the last 7 days. This is intentionally setup like that. Probably you are only interested in one of these files per week though.

#### 4.3.1.3 Using savelogs/archivelogs and availability management

The reporting tool is compatible with `savelogs/archivelogs` (see [Using the bsadmin tool](#) and [The reference section of the savelogs command](#)). Actually the `display.history` reader will not only read in `display.history` but also `display.history.*` files. If you archive your logs outside of `var/display.history.*` this will even drastically improve the efficiency of the daily incremental report generation steps since it is actually sufficient to have `display history` files available for 3 days back only!

Some of the file formats look very funny. Some of the file formats are actually CSV (well, more or less – do *not* try to use commas in cells!). That's why they look rather unfriendly to a `vi` user. The file formats were defined with the idea in mind that someone might use some spreadsheet or database application for editing or importing/exporting the files. CSV is one of the formats most applications understand.

## 4.4 Sending server commands

For accomplishing certain administrative tasks, sending commands to the server through the agent-server-protocol can come in very helpful. *Please read the [Server Command Reference](#) for a complete list of commands implemented in the Big Sister serverprotocol.*

### 4.4.1 Using the telnet client

Commands can be sent by establishing a telnet session to the tcp-server port and typing the desired command plus required arguments:

```
[root@proxy /root]# telnet [bigsisiter-server] 1984
Trying 192.168.170.13...
Connected to bigsisiter.joerg.cc.
Escape character is '^'.
[TYPE YOUR COMMANDS HERE]
```

Commands are text based. Each command starts with a keyword followed by the command arguments followed by a newline. Both "network" (CR/LF) and "unix" (LF) newlines are accepted. Newlines in a command argument must be replaced by the string "|>" before transmission. There is one exception to this: For compatibility reasons with Big Brother "status" and "page" commands may be multi-lined. The text beginning at a status command up to the next line looking like a valid command is treated as one single command. Do not use this feature!

Usually the client disconnects after successfully sending its commands. The Big Sister server does terminate a connection if one of the following conditions are met:

- an invalid command was sent
- a timeout occurred (no data received for a limited time)
- the client is not allowed to send a command

Whenever hostnames are fully qualified (domain name included) "." characters must be replaced by "\_" or "," before transmission.

By convention hostnames are written in lowercase letters while group names are written in uppercase letters.

Clients running in Big Brother compatibility mode should never send more than one command without closing/re-opening the TCP connection.

### 4.4.2 Using the bsadmin tool

Getting rid of unwanted status lights ("purples")

`bsadmin` is commandline based tool wich comes with Big Sister and offers a more comfortable way of sending server commands than the telnet client. It basically takes the same **commands and arguments** than the telnet client.

Whenever an agent stops reporting status messages for a certain host/check the Big Sister display server will still remember this host/check and display a purple ("no report") status light. This is the indended behaviour since by this means you notice which checks are currently not working (e.g. due to a connection loss with the respective agent).

However sometimes you will remove or rename a host/check and want those purples to just disappear. Big Sister 0.96 introduces a "remove" command for exactly this purpose. Agents now can send a "remove host.check" to the display server and the respective status light will disappear. The required procedure is actually a good example on how to send server commands using the telnet client or `bsadmin`.

| Function                    | Using bsadmin                                                    | Using the telnet client                                           |
|-----------------------------|------------------------------------------------------------------|-------------------------------------------------------------------|
| remove a single check       | <b>bsadmin -d</b> servername <b>remove</b><br>hostname.checkname | <b>telnet</b> servername port <b>remove</b><br>hostname.checkname |
| remove all checks of a host | <b>bsadmin -d</b> servername <b>remove</b><br>hostname.*         | <b>telnet</b> servername port <b>remove</b><br>hostname.*         |
| remove a host completely    | <b>bsadmin -d</b> servername <b>leave</b><br>hostname *          | <b>telnet</b> servername port <b>leave</b><br>hostname *          |

Table 4.5: Accepted keywords and function

In Big Sister versions prior to rev. 0.96, the remove command has not been implemented in the server protocol. Big Sister Version 0.96 to 0.98 need an additional `remove hostname.*` command for removing a host completely.

Never forget to limit access to this removal command (see [Server access: /etc/bigsisiter/permissions](#))

## 4.5 Configuring the Server and customizing the Display

### 4.5.1 The main server configuration file: bb-display.cfg

With very few exceptions every display related configuration appears somewhere in this file.

The suggested structure of the file is:

- Options
- System names and groups
- Web pages

Each of these sections is composed of a number of statements. Statements always start on a new line and are prefixed by a % character. Most statements take arguments, so a valid statement looks like e.g.:

```
%Autojoin new NEW(Autojoin with first argument new and second argument NEW. Please see ↔
 Automatically joined Groups for more information on the autojoin command)
```

See the section on bb-display.cfg in the [Server configuration reference](#) documentation for a complete list of known statements. Note that the above structure is mandatory.

### 4.5.2 The options section

In the options section you specify various not necessarily display related parameters. Statements belonging to the options section include

option: specify (boolean) options

autoconn: switch on the autoconn feature for specific hosts

pager: set the pager program used by Big Brother agents

A valid options section might look like

```
The port bbd will listen to
%Port 1984

Options ...
%Option -ImmediateHTML +KeepGroups +StartOK -DNS

this is only used by BB clients
%Pager /usr/local/lib/bb/bin/bb-page.sh

put newly appearing hosts into group NEW
%Autojoin new NEW

... and every host into ALL
%Autojoin all_hosts ALL

... and everything into UNIVERSE
%Autojoin all UNIVERSE
```

---

#### Note

The %Autoconn statement indicates to bbd (not bsmom!) for which hosts it should enable the autoconn feature. Whenever an agent connects to bbd for transmission of status information bbd looks up the system hosting the agent in the list of autoconn hosts. If it is listed there bbd automatically generates a green status message for this hosts conn (connection) status. When the agent does not connect to the server for more than 15 minutes the status is automatically changed to red.

---

## 4.5.3 The names and group section

### 4.5.3.1 Defining and using groups

Big Sister uses groups at various places. You will meet them when defining what systems' status will appear on which web page (see [The Web pages section](#)) as well as when setting up your alarming rules (see [Configure alarmin](#)).

In the system names and groups section of `bb-display.cfg` you specify displayed system and group names and define the group hierarchy. The most important statement you will want to remember is the Groups statement. All the lines between a Group statement and the next valid statement will be treated as group and/or name specifications.

Each line is of the form

```
hostname(Displayed Host Name) GROUP1 ... GROUPN
```

or

```
groupname(Displayed Group Name) GROUP1 ... GROUPN
```

where `hostname/groupname` is the name of a host (group resp.) and the string in parenthesis is the description of the host or group shown on web pages. `GROUP1` through `GROUPN` are names of groups `hostname/groupname` is a member of.

Groups are created when they are first referenced so you can use an arbitrary name for both the `groupname` or `GROUPN` arguments. Groups may themselves be members of other groups.



#### Warning

It is a good idea to avoid circular groups.

---

Time for an example: Assuming `washington`, `paris` and `london` are three systems monitored by Big Sister

```
%Groups
washington(Our server in Washington) USA SERVER
paris(Our server in Paris) EUROPE SERVER
london(Our server in London) EUROPE SERVER
EUROPE(Good old Europe) WORLD
USA(The United States) WORLD
WORLD(all the continents) UNIVERSE
SERVER(all our servers) UNIVERSE
```

the grouping statement above will create :

- a group called `USA` containing only the system `washington`,
- the group `EUROPE` containing the systems `paris` and `london`,
- the group `WORLD` containing the groups `EUROPE` and `USA`,
- the group `SERVER` containing all three systems,
- and finally the group `UNIVERSE` containing the groups `SERVER` and `WORLD` (see figure \*\*).

### 4.5.3.2 Automatically joined groups

Big Sister can automatically add a host to special groups at the time the first status message for the host comes in. Via the `Autojoin` statement you declare which groups Big Sister will use for this purpose. Basically there are three available `autojoin` features:

---

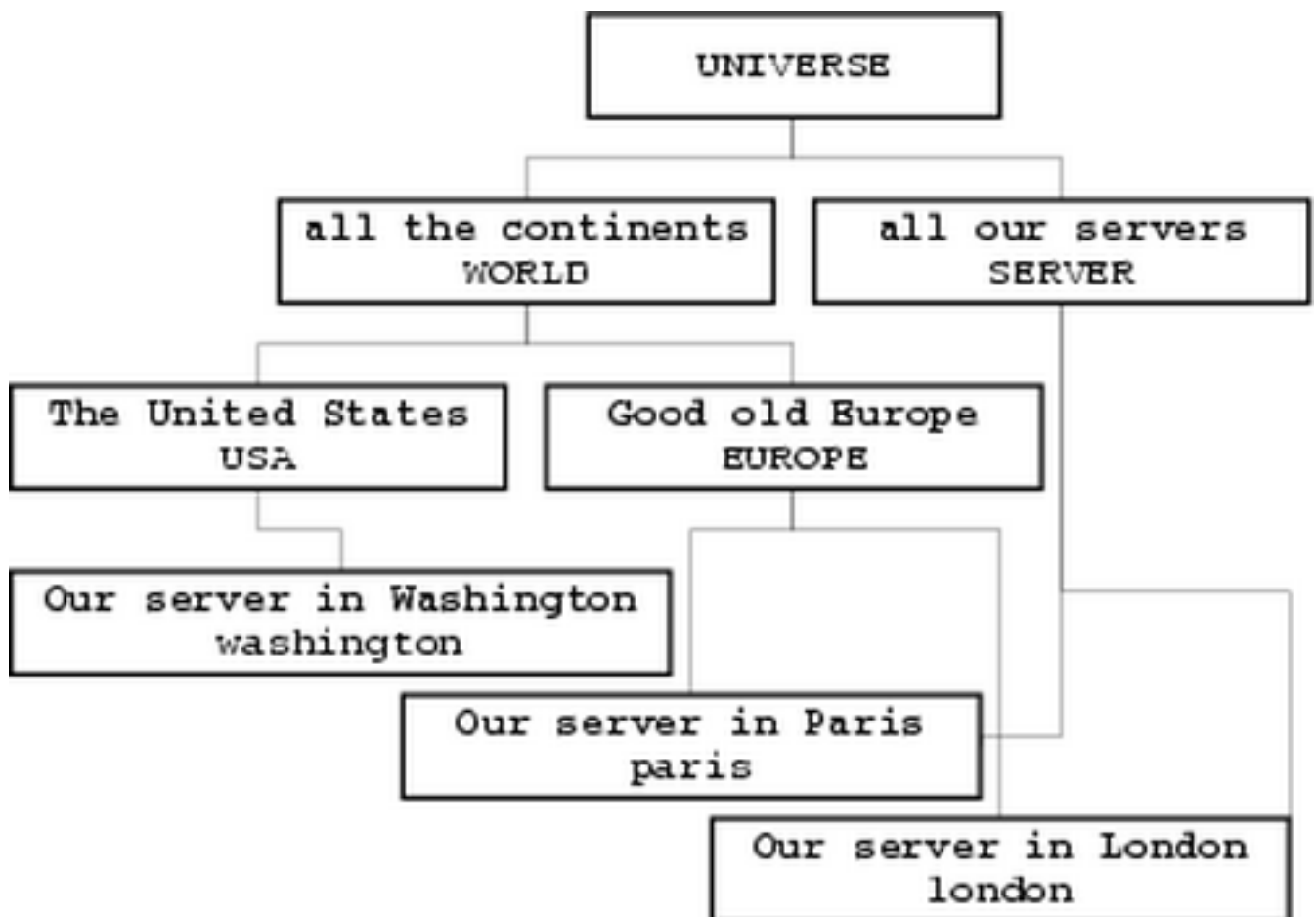


Figure 4.2: Group hierarchy example

- ```
%Autojoin new GROUPNAME
```

hosts joining in that are not member of any group yet (semantics: "they are new" / "not defined yet") will join the group named GROUPNAME. This is especially useful for detecting hosts which are already monitored by an agent but which are not correctly configured on server side.

- ```
%Autojoin all_hosts GROUPNAME
```

any host (but not groups!) becomes member of the group named GROUPNAME.

- ```
%Autojoin all GROUPNAME
```

any host or group becomes member of the group named GROUPNAME.

4.5.3.3 Deleting groups

4.5.4 The Webpages section

The web pages section defines what web pages are generated as well as what they look like. A few statements (like the skin and Logskin statements) are of a global nature while others (like title, table, etc.) are only applicable in conjunction with a Page statement.

Mainly you will define the basic look of your web pages via the skin and Logskin statements, then you describe the pages you would like to be generated and their contents via the Page statement and its "children" (title, refto, table, ref, itemref, image). Each page description starts with a Page statement as e.g.

```
%Page top The_main_page
```

It will create a web page called top.html with title "The main page"

Note

The page-title must not contain spaces. To bypass this limitation underscores will be replaced by spaces.

A single Page statement will not create much more than an empty page. The lines immediately following it should define some contents. Most important in this respect is the table statement. Followed by one or more groups it will insert one table per group containing the hosts or groups in the respective group and their status. So e.g. (the group names and members are taken from the example in section [Defining and using groups](#))

```
%Page top The_main_page %table EUROPE USA
```

will create a page with filename top.html containing two tables, one containing all the members of the group EUROPE (namely paris and london), the other containing all the members of the group USA.

Warning

That will only list the members of a group at the next hierarchy level. The command:

```
%table WORLD
```



will list EUROPE and USA, not washington, london, paris.

You can control how deep table digs by prefixing the group name(s) with a "+" sign. Multiple "+" signs will make table go deeper in the hierarchy.

```
%table +WORLD
```

This will list the group members two levels below WORLD (washington, london, paris).

All the other statements (except for the image statement documented in section [Using Imagemaps](#)) will not create contents themselves. Instead they modify the behaviour of the following table statement.

```
%title [ title | none | auto]
```

Specifies the tables title. if you use the special word none instead of a title tables are created without any title, the special word auto makes table use the display name (see [Defining and using groups](#)) of the groups the respective tables are created from.

Note

auto is the suggested variation.

```
%itemref [directory]
```

Specifies the sub-directory (of www) in which the file host.check.html is. So by clicking on a status light the browser will open the respective file. Two commonly used shortcut-arguments are:

- %itemref html point to the Big Sister html-ized status messages
- %itemref logs point to the un-html-ized (ASCII) status messages. This only works if the BBLog option is enabled.

```
%refto [name{ url | none }]
```

Whenever host names or group names appear in a table Big Sister will create a hypertext link pointing to url#hostname. Usually url will be the name of a page created via bb-display.cfg and is used to point from tables showing consolidated status information to more detailed tables. Big Sister will automatically create the necessary labels each time it inserts a table in a web page. Of course you are free to use URLs pointing somewhere outside the set of auto generated pages.

refto accepts a few special pseudo-URLs: refto none will suppress hypertext linking, refto self will create links pointing to the same page.

Note

refto always applies to all rows of a table.

refto name url - the refto statement explained above always takes effect on a whole table. However sometimes it is necessary to link individual hosts or group to individual pages. Another flavour of the refto statement supports exactly this. Via e.g.

Example 4.3 Example without name argument

```
%Page top Top_Page
%refto detailed
%table WORLD
%Page detailed Host_Details
%itemref html
%refto http://oursite.com/serverlist.html %table EUROPE USA
```

This will create two web pages top.html and detailed.html. On the first page only a single table containing rows for "Good old Europe" and "The United States" is shown. These two labels are linked to the second page where there are two tables, one listing all the hosts in the group EUROPE, one listing the hosts in the group USA. The labels in this two tables are linked to a manually created descriptive page ...serverlist.html.

Example 4.4 Example with name argument

```
%refto washington /servers/washington.html
```

you force the following table statements to link each appearing washington label to the page /servers/washington.html#washington. No matter which comes first an individual referto always overrides a global one, so that e.g.

```
%Page top Top_Page
%refto EUROPE europe
%refto USA usa
%refto all_the_rest
%table WORLD
%Page europe Europe
%refto none
%table EUROPE
%Page usa USA
%refto none
%table USA
```

will work correctly, thus creating three pages, one being an index page showing consolidated status for Europe and USA pointing to the respective pages listing the servers in Europe and USA.

Example 4.5 Example with special argument : clear

Individual referto entries are kept across Page statements. If you need to limit scope of entries you can use the special pseudo-url clear as in this example:

```
%Page top Top_Page
%refto EUROPE europe
%refto USA usa
%refto servers
%table WORLD
%refto clear
%table WORLD
```

This will create a table with links to the Europe and USA pages, then another table with links to the servers page.

4.5.4.1 Skins

While Big Sister strictly limits the contents of web pages to a few elements (actually tables and image maps) the way web pages look is configurable via the so called skin mechanism. A skin defines the layout of pages - to give some examples skins decide if tables get borders, if the background is white or yellow, if legends are at the top or at the bottom of a page or not present at all, if status lights are round and blinking or rather triangular and static, and so on.

Most skins do not define the whole palette of possible features. Instead, there is a basic skin (the default skin) and various skins modifying only a few layout elements - e.g. the white_bg skin will change the background to white, the static_lamps skin will replace the blinking status lights by non-blinking lights, etc. A whole set of such skins is called a skin set.

Skins appear in conjunction with the Logskin and skin statement. Logskin is used to specify the skin set Big Sister uses when creating the web pages for each checks detailed status while skin sets the default skin for everything else.

Example 4.6 Example on cgi's skins

```
%Logskin white_bg
%skin static_lamps,structured_bg,frames
```

This will make Big Sister create the HTML log pages with white background in place of the default one and all the other web pages with static status lights, a textured background and using HTML frames.

skin name	what is it doing with my display?
title_in_table	make table titles part of the table
white_bg	set pages' background to white in place of the default colored background
structured_bg	another alternate background
static_lamps	display non-blinking status lights
frames	frame optimized skin
bigbro13	Big Brother 1.3 like look
alt_contentsicons	especially ugly status lights in contents frame

Table 4.6: Some available skins

Note

The default skin is automatically part of any skin set, there is no need for explicitly listing it!

4.5.4.2 Skins in CGI programs

Skins are actually used by CGI programs too. In order to make Big Sister administrator's life a little bit more difficult (:-) CGIs will *not* read `bb-display.cfg`. Instead, you can set the skin they should use via the `adm/resources` file. An entry like

```
*.skin=structured_bg,static_lamps
```

in `/etc/bigsister/resources` will set the default skin used by every Big Sister component (including CGIs) to "structured_bg" and "static_lamps". If you replace "*" by the name of one specific utility or CGI then the setting only applies to it.

You can also select a skin via an argument to the CGI program, e.g. use

```
http://.../...cgi/bshistory?SKIN=structured_bg
```

4.5.4.3 Frames

Have you already read section concentrating on [skins](#)?

Pages displayed in frames are in fact just a special way of layouting pages, therefore you need only use a frames-enabled skin set (e.g. one including the skin frame) and you get a layout using frames.

Anyway, there is one little problem. It is not sufficient to create all the status pages, additionally an index page defining the frame set as well as the non-status frames are needed. This is the realm of the `Frameset` statement of `bb-display.cfg`:

```
%Frameset index top Monitored_by_Big_Sister
```

It will create an index page called `index.html`, the initial page displayed when entering `index.html` is `top.html` and the title of the frame set is "Monitored by Big Sister".

The statement will only work if the specified skin is frame-aware. If the skin defines a menu frame only the pages defined up to the `Frameset` statement will be respected.

4.5.5 Using imagemaps

Big Sister supports graphical image maps since version 0.22. To use this feature follow this procedure:

1. check if you've installed the Perl module `GD.pm`
2. create a background image you want to place your status lights on (e.g. a geographical map) and save it as a GIF, PNG or JPEG, e.g. `display_map.png`

Note

Older versions of GD only support GIF while newer versions support PNG and JPEG!

3. think about what you'd like to display on the image map. Note that you need to have a group for any of the buttons that should appear on the map. So configure the necessary groups in `bb-display.cfg`
4. think about where to place the buttons in the image map and get the display coordinates (sorry, you have to use your tools for that, but this should be not too much work)
5. create an image map config file, e.g. `display_map.cfg`. It could look like:

```
# read the background image from adm/display_map.png
template      adm/display_map.png

# use the png stat*.png for red/yellow/green/pruple/... status
red           www/skins/default/statred.png
yellow        www/skins/default/statyellow.png
green         www/skins/default/statgreen.png
purple        www/skins/default/statpurple.png
clear         www/skins/default/statclear.png
blue          www/skins/default/statblue.png

# remember display coordinates in the image canvas by name
name          15,308 genf
name          80,238 lausanne
name          96,115 central_switch

# insert status lamps for groups GENF and LAUSANNE at the
# position genf and lausanne
at            genf GENF
at            lausanne LAUSANNE

insert a status lamp for host/check switch1.conn at position central_switch
at            central_switch switch1.conn

# draw a line in the status color of GENF from position central_switch
# to position genf
line          central_switch genf GENF

# include a custom link/icon
link 90,238 http://myweb.com/lausanne/ adm/lausanneicon.png

# dump the resulting image out to www/map.png
dump          www/map.png
```

Note

The generated image will be called `www/mapxx.png`, where `xx` is a sequence number hold in `adm/display_map.cfg.seq`

6. in `adm/bb-display.cfg` add the line `%image some.path/display_map.cfg` to the `%Page` statement you want to appear the map in.

4.6 Deleting unused and stale network objects

After operating Big Sister for a while you will notice that it is not very hard to add newly monitored systems. However, if you remove health checks on agent-side the entries on the server side are not automatically disappearing since the Big Sister server does not know if the status reports disappeared accidentally or intentionally.

In order to remove a single health check for one host, follow the following procedure

- remove the health checks in the uxmon-net of the agent that was responsible for performing the test
- wait until the agent re-read its configuration (up to one minute) or re-start the agent
- execute

```
bsadmin -d localhost remove host.check
```

on the Big Sister server side (of course, host and check must be replaced by the actual host and check names)

- after up to one minute the status lamp in the row *check* of the host *host* will disappear on the server display

In order to remove a whole host with all its checks, follow this procedure

- remove all the tests in the uxmon-net files of the agents that were responsible for performing the tests
- wait until the agents re-read their configuration (up to one minute) or re-start the agent
- remove all group associations for the host in bb-display.cfg on the server side
- wait until the server re-read its configuration or re-start the server
- execute

```
bsadmin -d localhost leave host \*
```

on the Big Sister server side

- after up to one minute the machine will disappear

4.7 Attaching the Big Sister server to a database

As of revision 1.00 Big Sisters supports storing some of the status information in a database like i.e. MySQL for later processing. In 1.00 and 1.01 this is limited to performance data. Revision 1.02 will introduce a few more graphing capabilities making use of the stored performance data.

4.7.1 Enabling data logging in a database

By default, data storage in a database is disabled. In order to enable it, follow the following procedure:

- install the DBI perl module, and the DBD (database driver) module for the database you want to attach
- install the database server (MySQL, PostgreSQL, whatever), create a database, e.g. called "bigsister", and a database user granted permissions to create tables, update and select rows
- find the db.cfg file in the etc directory and copy it over to the adm directory
- edit the db.cfg file, comment the entries referring to the "Null" database, uncomment the entries matching your chosen database server or derive entries from the example
- Re-start the Big Sister server: The database tables should now be automatically created and Big Sister starts logging data

Tip

Big Sister is using a very simple data model allowing data to be spread over multiple databases or be stored in very simple, not really relational databases. Therefore, instead of setting up a database server you can even choose to store data in CSV files!

4.7.2 Data logging in MySQL database

The database interface of Big Sister is not limited to particular database servers. Anyway, since MySQL is very wide-spread we use it as an actual example to show how to set up database access for Big Sister.

4.7.2.1 Prerequisites

Before you can setup Big Sister data storage, you need to install the MySQL database server, of course. Also, Big Sister uses Perl's DBI (database interface) abstraction layer for accessing databases. You have to install DBI and the MySQL driver for DBI called `DBD::MySQL`. Both are directly available from [CPAN](#) or most probably your operating system or Perl distribution does offer packages for those modules, i.e. on Debian you would install the package `libdbd-mysql-perl`.

4.7.2.2 Creating the database and database user

Next, you need to create an (empty) database for Big Sister. You do so by running the `mysqladmin` command:

```
# mysqladmin create bigsister
```

Note

Your MySQL installation might require you to login with a privileged user and password in order to create databases and/or create users.

Now, a user must be created, that has read and write access to tables, and can create tables in our newly-created `bigsister` database. Do this using SQL statements executed via the `mysql` command:

```
# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 4.0.24_Debian-5-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> GRANT ALL ON bigsister.* TO bigsister@localhost IDENTIFIED by 'abc123';
Query OK, 0 rows affected (0.15 sec)

mysql> GRANT ALL ON bigsister.* TO bigsister@'%' IDENTIFIED by 'abc123';
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)
```

This example creates a user called `bigsister` with the password `abc123` that is granted all permissions in the `bigsister` database and may access the database from the local machine as well as via the network.

4.7.2.3 Configure Big Sister

Now, find the `etc/db.cfg` file and copy it over into the `adm` directory. Find the (commented out) section where `mysql` is mentioned. Uncomment this section. Also, find the lines containing `Null` and comment them out or remove them. Then, replace the username, password, database and database server settings. Your `db.cfg` should now look something like this:

```
db=host driver=DBI table=host datasource=DBI:mysql:database=bigsister \
  user=bigsister password=abc123
db=variable driver=DBI table=variable datasource=DBI:mysql:database=bigsister \
  user=bigsister password=abc123
db=perf driver=DBI table=perf datasource=DBI:mysql:database=bigsister \
  user=bigsister password=abc123
```

```
db=perfinfo driver=DBI table=perfinfo datasource=DBI:mysql:database=bigsisiter \
  user=bigsisiter password=abc123
...
```

Re-start the Big Sister server. The database tables will automatically be created as soon as there is data to store in them. Make sure, an agent is reporting performance data to your server. After a moment, have a look at your database:

```
# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 296 to server version: 4.0.24_Debian-5-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use bigsisiter;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_bigsisiter |
+-----+
| host                  |
| perf                  |
| perfinfo              |
| variable              |
+-----+
4 rows in set (0.01 sec)
```

Our database is working. If you see no tables, have a look at your syslog. A message should have been logged if Big Sister cannot access the database for some reason. Make sure the database user actually has permission to access the database, i.e. like that:

```
# mysql -u bigsisiter -p bigsisiter
Enter password:
Reading table information for completion of table and column names
...
```

4.7.3 Data logging without a database server: the file database

Setting up a database server may be prevented by your IT policy or just too much work for a small installation. Big Sister explicitly supports maintaining its data storage in a file based pseudo-database, mainly consisting of CSV files for non-critical tables and a special binary file storage for performance-critical parts of the database.

In order to enable this file database, please install the DBD::CSV Perl module and follow the [general setup](#) above. Find the commented out section in db.cfg with the title *File Data Store*, uncomment it and replace the file paths if necessary.

4.7.4 Database Structure

Big Sister 1.01 so far only stores performance data in the database. The main table is the *perf* table, where each row corresponds with an incoming numerical perf data report. Textual data is stored in the *perfinfo* table.

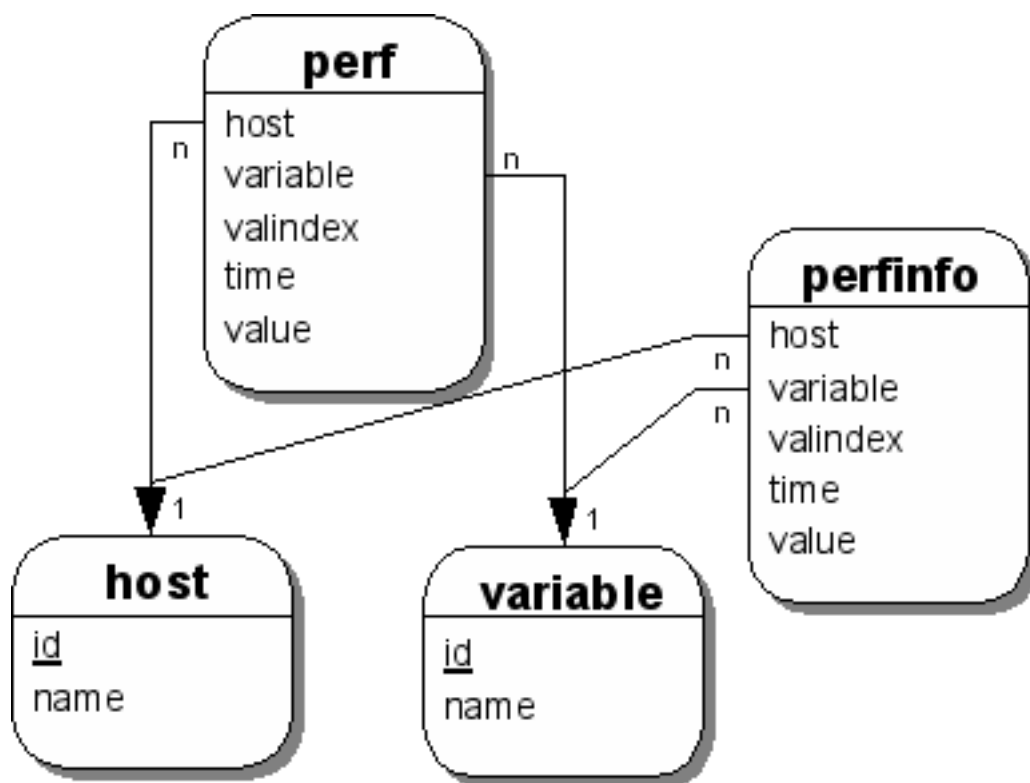


Figure 4.3: Database Structure of Big Sister 1.01

Chapter 5

Appendices

5.1 Question and Answers : Q+A

5.1.1 Trend graphs do not appear though I followed the instructions on setting them up

The most common problem with graphing is that Big Sister just cannot find the "rrdtool" command. Please check your syslog (/var/log/messages, /var/adm/messages, whatever) to see if Big Sister logged something like

```
Jun 19 18:30:02 susix bsmon: RRD creation failed for graph
```

In this case either change Big Sister's path settings (in the adm/resources) or copy rrdtool to some place Big Sister finds it by default, e.g. /usr/bin.

5.1.2 Can I have some graphics showing disk/memory/cpu/... usage during the last few days?

Big Sister 0.38 introduced agent side performance data and finally since 0.94 the server module supporting storing this data in a database and creating nice trend graphics is working. See the section on performance data collection for more details.

5.1.3 No links to my performance data trend graphics are appearing on the web pages

Check if the directories var/graphs and www/graphs exist and contain files. If not you have not successfully enabled performance data collection (see above). If yes, then it might just take some time for the links to appear on your status web pages. Note that the status pages will not be updated immediately if a graph is created - it will be included when the status page is updated next time because of status changes.

5.1.4 Links to trend graphics appear but when I click on it no graph is displayed

Check if you really have RRDTOOL installed. Note that the 'rrdtool' command must be in the path of both the Big Sister server and your web server.

5.1.5 After installing Big Sister my display works but I do not see any status from my clients

The Big Sister client "uxmon" will read its config file adm/uxmon-net on startup. It expects the Big Sister status collector (display server) listed there. After a new installation you will find a line:

```
bsdisplay bsdisplay
```

there. Please change the left "bsdisplay" to the name or IP address of your display server.

5.1.6 Some of my web pages look quite poor (no background, no lamps) while others look fine

Big Sister needs to know the url of its web pages. If you did not tell it during installation using e.g.

```
make install WEBROOT=/monitors/bs ...
```

or you moved your web pages to a new location then you should try altering the file

```
www/skins/default/WEBROOT.inc
```

BTW: It is not necessary to list a full url (The absolute path within your server is sufficient.

5.1.7 The History and Alarms menu entries do not work

Some features of Big Sister rely on CGIs. Please remember to do the following steps:

- where to put your Big Sister's CGIs. It might be a good idea to separate them from your other CGIs.
- Set up your web server so that it will execute CGIs in the respective locations and follow symbolic links.
- Create the physical directory where you want your Big Sister CGIs to be in and place symbolic links pointing to the Big Sister CGIs (bin/bshistory, bin/bswebalarm, etc., see)

During installation of Big Sister point Big Sister to your CGI url using e.g.

```
./configure ... --with-cgi=/cgi
```

5.1.8 CGIs are so slow

While sharing code and using the skin mechanism is a fine thing for program developers it makes startup of scripts a little bit slow. While there is no way around this you can reduce the number of script startup by using Fast CGI. You need to do two steps to enable running Big Sister CGIs in Fast CGI mode: - enable your web server executing scripts in Fast CGI mode (e.g. for apache users: install the mod_fastcgi module available from www.fastcgi.com and do a few configuration changes) - install the FCGI perl module (available from CPAN sites)

5.1.9 The .conn (ping) test does always report failure

By default the ping test will use UDP protocol. On most machines the ICMP echo service is built in the network driver at a very low level and will just work (I've even saw some SPARC boxes answering ICMP echo requests after being brought down to PROM mode!). The UDP and TCP echo services are only working if configured. On a Unix system they will have to be enabled in `/etc/inetd.conf`. On NT systems the "Simple TCP/IP" service will have to run.

If you rather like to use ICMP pings than to tell your boxes to answer UDP echo requests then you will have to use the ping test as:

```
mybox proto=icmp ping
```

Since usually processes running with user priviledges will not get access to the ICMP protocol layer uxmon has to be run with root priviledges in this case. Do a

```
touch adm/uxmon-asroot
```

in your Big Sister directory and restart uxmon. This will run uxmon with root priviledges.

Decide yourself which does suit your security policies best: running uxmon as root or enabling Echo services on all the machines you want to monitor.

Note: Older versions of uxmon did not implement the "proto=icmp" option of ping correctly. This has been fixed in release 0.30.

5.1.10 My display.history file is growing larger and larger

Since version 0.32 bbd implements a method for rotating its log files. However you need to tell bbd manually to execute it using

```
bsadmin -d mybbdhost savelogs
```

(see the How To on log file rotation too)

5.2 SNMP Agent

5.2.1 Big Sister as an SNMP trap source

Since version 0.38 Big Sister includes support for sending SNMP traps.

Traps may be sent on the following occasions:

- alarm raised by `bb_event_generator`
- alarm cleared
- reminder sent by `bb_event_generator`
- alarm acknowledgement received

Big Sister will (try) to act as an SNMP proxy, the agent reported is usually the IP address of the host associated with the service. If this IP address is not determinable then the agent is reported to be the system on which the SNMP trap is generated.

As of rev 0.99c3 Big Sister also supports sending traps on status changes. Please have a look at the [StatusTrap](#) configuration option in `bb-display.cfg`.

5.2.2 MIB

```
BIGSISTER-MIB DEFINITIONS ::= BEGIN

IMPORTS
    enterprises
        FROM RFC1155-SMI

    TRAP-TYPE
        FROM RFC-1215

    OBJECT-TYPE
        FROM RFC-1212;

-- Graeffet Communications and Big Sister enterprise OID
graeff      OBJECT IDENTIFIER ::= { enterprises 5281 }

bigsisiter  OBJECT IDENTIFIER ::= { graeff 1 }

-- object types used in Big Sister traps
-- note that Big Sister will always sort bindings
-- by object type, variables with lower object
-- type will always appear first in the bindings

bstrapinfo  OBJECT IDENTIFIER ::= { bigsisiter 1 }

bshost      OBJECT-TYPE
```

```
SYNTAX          OCTET STRING
ACCESS          not-accessible
STATUS          mandatory
DESCRIPTION
    "The name of the monitored host as
    reported by the Big Sister agent"
::= { bstrapinfo 1 }
```

bsservice OBJECT-TYPE

```
SYNTAX          OCTET STRING
ACCESS          not-accessible
STATUS          mandatory
DESCRIPTION
    "The name of the monitored service"
::= { bstrapinfo 2 }
```

bsstatus OBJECT-TYPE

```
SYNTAX          OCTET-STRING
ACCESS          not-accessible
STATUS          optional
DESCRIPTION
    "The service's status code"
::= { bstrapinfo 3 }
```

bscomment OBJECT-TYPE

```
SYNTAX          OCTET-STRING
ACCESS          not-accessible
STATUS          optional
DESCRIPTION
    "The text reported by the Big Sister
    agent associated with this event"
::= { bstrapinfo 4 }
```

bsseverity OBJECT-TYPE

```
SYNTAX          INTEGER (0..100)
ACCESS          not-accessible
STATUS          optional
DESCRIPTION
    "This event's severity, 100 for
    maximum severity"
::= { bstrapinfo 5 }
```

bsackmode OBJECT-TYPE

```
SYNTAX          OCTET-STRING
ACCESS          not-accessible
STATUS          optional
DESCRIPTION
    "In alarm acknowledge events this represents
    the type of acknowledgement"
::= { bstrapinfo 6 }
```

bsackuser OBJECT-TYPE

```
SYNTAX          OCTET-STRING
ACCESS          not-accessible
STATUS          optional
DESCRIPTION
    "In alarm acknowledge events this represents
    the name of the acknowledging user"
::= { bstrapinfo 7 }
```

```
-- Trap types

bsstatuschange TRAP-TYPE
    ENTERPRISE      graeff
    VARIABLES
        {
            bshost, bsservice,
            bsstatus, bscomment
        }
    DESCRIPTION
        "This trap will be generated on status changes
        of any monitored service"
    ::= 1

bsserviceup TRAP-TYPE
    ENTERPRISE      graeff
    VARIABLES
        {
            bshost, bsservice,
            bsstatus, bscomment,
            bsseverity
        }
    DESCRIPTION
        "This trap will be generated by bb_event_generator
        whenever a service up alarm is raised"
    ::= 2

bsservicedown TRAP-TYPE
    ENTERPRISE      graeff
    VARIABLES
        {
            bshost, bsservice,
            bsstatus, bscomment,
            bsseverity
        }
    DESCRIPTION
        "This trap will be generated by bb_event_generator
        whenever a service down alarm is raised"
    ::= 3

bsalarmack TRAP-TYPE
    ENTERPRISE      graeff
    VARIABLES
        {
            bshost, bsservice,
            bscomment,
            bsackmode, bsackuser
        }
    DESCRIPTION
        "This trap will be generated by bb_event_generator
        whenever an alarm acknowledgement is seen"
    ::= 4

bsremind TRAP-TYPE
    ENTERPRISE      graeff
    VARIABLES
        {
            bshost, bsservice,
            bsstatus, bscomment,
            bsseverity
        }
    DESCRIPTION
```

```
        "This trap will be generated by bb_event_generator  
        whenever a repeated alarm - aka reminder - is raised"  
    ::= 5
```

```
END
```

Chapter 6

Command Reference

6.1 testers

Name

testers – query healthcheck for available features, produce XML and static manpage

Synopsis

```
testers [-f feature1 ...] [-d device] [-t name of healthcheck/test] [-F xml|man] [known|tests|man]
```

Description

The `testers` command is available as of Big Sister rev. 0.98. The `testers` command can be used to query healthchecks for the syntax and possibilities available on the specified device. The standart output format is ascii. Depending on the software installed on your system, `testers` can also produce xml-formated output or preformatted manpages. This is useful for the Big Sister Documentation Project.

The bells and whistles coming with every healthcheck are mainly dependant on the *device -d* which is beeing tested, the operation system (the *feature* option *-f* currently refers to the OS) and whether a healthcheck is carried out using a *local* installet uxmon agent or *remote*.

Example 6.1 disk healthcheck: local and via the network (remote)

The `disk` healthcheck monitors the free disk space.

The `disk` healthcheck is a good example of featuresets differing with local and remote healthchecks: the `df` command can be used if the healthcheck is carried out by a local installed uxmon agent. If the free diskspace needs to be checked over the network by a remote uxmon agent, SNMP and the Host-MIB are used.

The `fstype` option is only available with local installed agents; -not via SNMP and not with every operation system. Therefore the output of:

```
./testers -f local -d computer -t disk
```

and

```
./testers -f remote -d computer -t disk
```

are different.

Known devices, features and healthchecks

This list might be outdated! You can get a more recent list when issueing the `./testers` command on your system without any options.

Device types [-d] computer

Feature Names [-f] aix

- bsd
- hpux
- linux
- local
- macos
- netware
- remote
- solaris
- sysv
- unix

Healthchecks [-t] diskfree

- etherport
- ftp
- http
- ica
- ident
- imap
- load
- network
- nntp
- ntp
- pop3
- procs
- procsname
- realhttp
- smtp
- snmpvar
- ssh
- tcp
- telnet
- users

Producing xml and manpages

By using the `-F` option you can either generate an xml formatted manpage

```
testers [your arguments go here] -F xml man
```

or complete preformatted manpages which can be copied into your system's manpath

```
testers [your arguments go here] -F man man
```

:

6.2 bsadmin

Name

bsadmin – send server commands to a Big Sister server

Synopsis

```
bsadmin [-D debuglevel] [-d server[:port]] [-t timeout] [-f commandfile] [command] [command-args ...]
bsadmin [[options]] [join group group1 ...]
bsadmin [[options]] [leave group group1 ...]
bsadmin [[options]] [remove host.check ...]
bsadmin [[options]] [displayname group ...]
bsadmin [[options]] [savelogs [tag]]
bsadmin [[options]] [archive logs directory [days]]
bsadmin [[options]] [status host.check color text]
bsadmin [[options]] [disable host.check duration|forever text]
bsadmin [[options]] [enable host.check text]
bsadmin [[options]] [perf time host:variable value]
```

Description

The bsadmin sends a **server command** to a Big Sister server. Also, multiple commands can be put into a text file, one command per line, and be sent to the server using the *-f* option. The target server can be specified with the *-d* option, it defaults to *localhost:1984*.

join group group1 ...

Makes the group or host named *group* join one or multiple groups. Without **KeepGroups** option set on the server-side, group joins done via the join command will only persist until the server is re-started or the server configuration file is changed.

leave group group1 ...

Makes the group or host named *group* leave one or multiple groups. If one of the group arguments is an asterisk (*) rather than a group name, the host/group leaves all groups. Without **KeepGroups** option set on the server-side, group leaves done via the leave command will only persist until the server is re-started or the server configuration file is changed.

remove host.check ...

Makes the server forget the status for the given host (row) and check (column) pairs. Check may be an asterisk (*), in which case all the checks for this host are dropped.

Note

This drops only the remembered status color and text on the server side. If there is still an agent reporting for the host/check, then its next report will make the host/check reappear.

displayname group name

Sets the displayed name of a group or host. The display name is the name used in the first column of status tables. Configured display names in the **groups section** of the server configuration will override the displayname on the next server re-start or configuration re-load unless the **KeepGroups** option is active.

savelogs [tag]

Forces a log-rotate of the growing Big Sister logs (mainly the display.history file). The logs are stored with a file extension passed via the tag option. If no tag is given, the current date and time are used to generate one.

archivelogs directory [days]

Like savelogs, archivelogs forces a log-rotate, but as well stores the saved logs in the local directory specified on the command line.

status host.check color text

Sets the status of a host/check to the given color and text.

disable host.check duration|forever text

Sets a given host/check into maintenance mode for the given duration. The duration must either be a time in *hours:minutes* notation or the string *forever*. In place of an existing check (row) the disable command also accepts an asterisk (*) and will then disable all checks of the given host.

enable host.check text

Resets a given host/check that was set into maintenance mode and re-starts normal status reporting for it. As with the disable command, the check may also be an asterisk meaning that all the checks of the given host should be reset.

perf time host:variable value

Sends performance data to the server. The time must be a Unix time value (integer, seconds from January 1st, 1970) or 0. In the latter case the current time is associated with the value.

Chapter 7

New-style Health Checks

7.1 accesstime

Name

accesstime – Monitors file access time

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] accesstime
```

Description

Monitors file access time

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under (default: file)

path [**string**] network path of the testfile

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

red [**integer number**] Minimum file access time or data transfer bandwidth unless red status is reported

size [**integer number**] size of the testfile (e.g. size=10M or size=100kB). Default: 100kB

yellow [**integer number**] Minimum file access time or data transfer bandwidth unless yellow status is reported

7.2 disk

Name

disk – Monitor a disks free space using the 'df' command

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] disk
```

Description

Monitor a disks free space using the 'df' command

Availability

- local / unix / bsd computer
- local / unix / sysv / aix computer
- local / unix / sysv / hpux computer
- local / unix / sysv / solaris computer
- local / unix / sysv / true64 computer
- remote
- remote / unix
- unix / local computer
- windows computer

Options

Global Options

community [string] The SNMP get community

fail [size in kBytes OR percentage] Fail level - Big Sister panics (red) if free space goes below this level

item [string] Item as which to report.

partition [list of regular expression/string] A list of resource (partition/volume/drive) names

perf [integer number] report perf data every perf minutes

storagetypes [list of oid] The object IDs of the resource(s) to be monitored

warn [size in kBytes OR percentage] Warn level - Big Sister warns (yellow) if free space goes below this level

local / unix / sysv / true64 computer

Monitor a disks free space using the 'df' command

fstype [list of regular expression/string] FS type (works only if OS supports it)

df [string] The 'df' command to be used

local / unix / sysv / solaris computer

Monitor a disks free space using the 'df' command

fstype [list of regular expression/string] FS type (works only if OS supports it)

df [string] The 'df' command to be used

unix / local computer

Monitor a disks free space using the 'df' command

fstype [list of regular expression/string] FS type (works only if OS supports it)

df [string] The 'df' command to be used

local / unix / bsd computer

Monitor a disks free space using the 'df' command

fstype [list of regular expression/string] FS type (works only if OS supports it)

df [string] The 'df' command to be used

local / unix / sysv / hpux computer

Monitor a disks free space using the 'df' command

fstype [list of regular expression/string] FS type (works only if OS supports it)

df [string] The 'df' command to be used

local / unix / sysv / aix computer

Monitor a disks free space using the 'df' command

fstype [list of regular expression/string] FS type (works only if OS supports it)

df [string] The 'df' command to be used

7.3 diskload

Name

diskload – Monitor the disk load

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] diskload
```

Description

Monitor the disk load

Availability

- local / unix / sysv
- local / windows
- unix / local / sysv / true64

Options

Global Options

fail [**integer number**] The test goes red if the disk load goes above this level (default: 95)

item [**string**] column name we should report the results under (default: cpu)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

warn [**integer number**] The test goes yellow if the disk load in percent goes above this level (default: 80)

local / unix / sysv

Monitor the disk load

sarcommand [**string**] sar command to be used (defaults to sar -A %d 1)

unix / local / sysv / true64

Monitor the disk load

sarcommand [**string**] sar command to be used (defaults to sar -A %d 1)

7.4 etherport

Name

etherport – Monitor interface operating status

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] etherport
```

Description

Monitor interface operating status

Availability

- remote
-

Options

Global Options

admindown [**string**] status color to report if admin shut the port down (default:blue)

community [**string**] The SNMP get community

ignoreadmin [**boolean (1: true, 0: false)**] If true also monitor interfaces that are disabled by configuration (default: 1)

interface [**list of regular expression/string**] A list of interface names

item [**string**] column name we should report the results under (default: port)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

7.5 ftp

Name

ftp – Monitor FTP service. Shortcut for: service=ftp tcp

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] ftp
```

Description

Monitor FTP service. Shortcut for: service=ftp tcp

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.6 http

Name

http – Monitors a http server

Synopsis

[parameter1=value1] [parameter2=value2] [...] http

Description

Monitors a http server

Availability

- unlimited

Options

Global Options

item [string] column name we should report the results under (default: cpu)

perf [integer number] time in minutes between sending performance data (0 for no performance data)

url [string] URL uxmon tries to load in order to determine if the server is up

7.7 ica

Name

ica – Monitor Citrix ICA service. Shortcut for: service=ica tcp

Synopsis

[parameter1=value1] [parameter2=value2] [...] ica

Description

Monitor Citrix ICA service. Shortcut for: service=ica tcp

Availability

- unlimited

Options

Global Options

item [string] column name we should report the results under

perf [integer number] time in minutes between sending performance data (0 for no performance data)

port [integer number OR string] port number or port name uxmon will try to connect to

service [string] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [integer number] time in seconds until which a response must be received (red)

7.8 icmp_ping

Name

icmp_ping – Monitors network connectivity via icmp echo requests

Synopsis

```
[parameter1=value1][parameter2=value2][...] icmp_ping
```

Description

Monitors network connectivity via icmp echo requests

Availability

- unlimited

Options

Global Options

count [**integer number**] number of echo requests to send

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

7.9 ident

Name

ident – Monitors ident server

Synopsis

```
[parameter1=value1][parameter2=value2][...] ident
```

Description

Monitors ident server

Availability

- unlimited
-

7.10 imap

Name

imap – Monitor IMAP service. Shortcut for: service=imap tcp

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ]  imap
```

Description

Monitor IMAP service. Shortcut for: service=imap tcp

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.11 ldap

Name

ldap – Monitors a LDAP(S) server specified via an URL

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ]  ldap
```

Description

Monitors a LDAP(S) server specified via an URL

Availability

- unlimited
-

Options

Global Options

base [**string**] LDAP base

dn [**string**] authentication dn (if not set, an anonymous bind is used)

filter [**string**] query filter

item [**string**] column name we should report the results under

password [**string**] password

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

timeout [**integer number**] Timeout in seconds (default: 30)

url [**string**] Service URL

version [**integer number**] LDAP version (default: 3)

7.12 load

Name

load – Monitors CPU load

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] load
```

Description

Monitors CPU load

Availability

- local / unix
- local / unix / sysv
- unix / local / sysv / true64
- windows

Options

Global Options

idle_red [**integer number**] Minimum % the CPU must be idle unless red status is reported (default: 10)

idle_yellow [**integer number**] Minimum % the CPU must be idle unless yellow status is reported (default: 15)

item [**string**] column name we should report the results under (default: cpu)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

local / unix / sysv

Monitors CPU load

sarcommand [**string**] sar command to be used (defaults to sar -A %d 1)

unix / local / sysv / true64

Monitors CPU load

sarcommand [**string**] sar command to be used (defaults to sar -A %d 1)

7.13 mailq

Name

mailq – Monitors a mail queue

Synopsis

[parameter1=value1][parameter2=value2][...] mailq

Description

Monitors a mail queue

Availability

- unlimited

Options**Global Options**

fail [**integer number**] number of queued messages to issue a failure message (default 750)

item [**string**] column name we should report the results under

mqcmd [**string**] mailq command to run (default mailq)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

warn [**integer number**] number of queued messages to issue a warning (default 500)

7.14 memory

Name

memory – Monitor virtual memory usage using Linux /proc/meminfo file

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] memory
```

Description

Monitor virtual memory usage using Linux /proc/meminfo file

Availability

- local / unix / linux
- remote
- unix / local / sysv
- unix / local / sysv / solaris
- unix / local / sysv / true64
- windows

Options

Global Options

community [**string**] The SNMP get community

fail [**size in kBytes OR percentage**] Fail level - Big Sister panics (red) if free space goes below this level

item [**string**] Item as which to report.

partition [**list of regular expression/string**] A list of resource (partition/volume/drive) names

perf [**integer number**] report perf data every perf minutes

storagetypes [**list of oid**] The object IDs of the resource(s) to be monitored

warn [**size in kBytes OR percentage**] Warn level - Big Sister warns (yellow) if free space goes below this level

7.15 myexpedap

Name

myexpedap – My External Performance Data Provider

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] myexpedap
```

Description

My External Performance Data Provider

Availability

- unlimited

Options

Global Options

input [**integer number**] a sample input value(default: 1)

item [**string**] column name we should report the results under (default: expedap)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

red [**integer number**] a sample threshold value for level red (default:2)

yellow [**integer number**] a sample threshold value for level yellow (default: 1)

7.16 mysql

Name

mysql – Monitors a mysql server

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] mysql
```

Description

Monitors a mysql server

Availability

- unlimited

Options

Global Options

fail [**integer number**] number of threads connected to issue a failure message (default 150), set this to 0 if you don't want the threads monitored

item [**string**] column name we should report the results under

password [**string**] MySQL user password

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

user [**string**] MySQL user

warn [**integer number**] number of threads connected to issue a warning (default 100)

7.17 network

Name

network – Monitor per interface network load and error rates

Synopsis

```
[parameter1=value1][parameter2=value2][ ... ] network
```

Description

Monitor per interface network load and error rates

Availability

- local / linux
- remote
- windows

Options

Global Options

interface [**list of regular expression/string**] A list of interface names

item [**string**] column name we should report the results under (default: cpu)

maxerror [**rate**] Test fails if error rate goes above this value (default:4)

maxload [**size in kBytes OR percentage**] Fail if interface load goes above this percentage (default:60%)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

speed [**rate in kbit/s**] Override speed reported by the interface with this value

warnerror [**rate**] Test shows yellow if error rate goes above this value (default: 1)

warnload [**size in kBytes OR percentage**] Warn if interface load goes above this percentage (default:30%)

remote

Monitor per interface network load and error rates

unavail [**string**] status color to report if a port is down, unavail=blue might be useful (default:red)

emulateadmin [**boolean (1: true, 0: false)**] if set, the AdminStatus of each interface is ignored and OperStatus is used instead (default:0)

community [**string**] The SNMP get community

windows

Monitor network interfaces using Windows PerfLib

unavail [**string**] status color to report if a port is down, unavail=blue might be useful (default:red)

emulateadmin [**boolean (1: true, 0: false)**] if set, the AdminStatus of each interface is ignored and OperStatus is used instead (default:0)

community [**string**] The SNMP get community

7.18 nntp

Name

nntp – Monitor NNTP Service. Shortcut for: service=nntp tcp

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] nntp
```

Description

Monitor NNTP Service. Shortcut for: service=nntp tcp

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.19 ntp

Name

ntp – Monitors NTP server via ntptrace command

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] ntp
```

Description

Monitors NTP server via ntptrace command

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

7.20 nutups

Name

nutups – Monitors uninterruptible power supplies attached via the Network UPS Tool

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] nutups
```

Description

Monitors uninterruptible power supplies attached via the Network UPS Tool

Availability

- unlimited

Options

Global Options

chargefail [**percentage**] Battery Charge Fail level - Big Sister panics (red) if battery charge goes below this level (default: 30%)

chargewarn [**percentage**] Battery Charge Warn level - Big Sister warns (yellow) if battery charge goes below this level (default: 90%)

item [**string**] Item as which to report.

loadfail [percentage] Battery Load Fail level - Big Sister panics (red) if UPS load goes above this level (default: 95%)

loadwarn [percentage] Battery Load Warn level - Big Sister warns (yellow) if UPS load goes above this level (default: 85%)

perf [integer number] report perf data every perf minutes

runtimefail [percentage] Battery Runtime Fail level - Big Sister panics (red) if UPS runtime goes below this level (default: 360)

runtimewarn [percentage] Battery Runtime Warn level - Big Sister warns (yellow) if UPS runtime goes below this level (default: 600)

status [string] Normal Status - Big Sister will warn (yellow) if status of the UPS does not equal this status (default: OL)

ups [list of regular expression/string] A list of UPS names

7.21 oracle

Name

oracle – Monitors an Oracle server using sqlplus

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] oracle
```

Description

Monitors an Oracle server using sqlplus

Availability

- unlimited

Options

Global Options

database [string] database connection string as defined in tnsnames.ora

item [string] column name we should report the results under

password [string] password

perf [integer number] time in minutes between sending performance data (0 for no performance data)

user [string] user name

7.22 pop3

Name

pop3 – Monitor POP3 service. Shortcut for: service=pop3 tcp

Synopsis

[parameter1=value1] [parameter2=value2] [...] pop3

Description

Monitor POP3 service. Shortcut for: service=pop3 tcp

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.23 procs

Name

procs – Monitors running processes

Synopsis

[parameter1=value1] [parameter2=value2] [...] procs

Description

Monitors running processes

Availability

- local / unix / linux
 - local / unix / sysv
 - local / unix / sysv / hpux
 - local / unix / sysv / solaris
 - local / unix / sysv / tru64
-

- macos / local
- remote
- unix / bsd / local
- unix / local
- windows

Options

Global Options

item [**string**] column name we should report the results under

max [**integer number**] maximum number of processes of that name that may run (default: infinite)

min [**integer number**] minimum number of processes of that name that must run (default: 1)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

proc [**string**] name of a monitored process

pscomm [**string**] ps command to use (default: system dependent)

remote

Monitor remote processes using SNMP HOST-MIB - monitor running processes paths

community [**string**] The SNMP get community

7.24 procsname

Name

procsname – Monitor remote processes using SNMP HOST-MIB - monitor running processes names

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] procsname
```

Description

Monitor remote processes using SNMP HOST-MIB - monitor running processes names

Availability

- remote
-

Options

Global Options

community [**string**] The SNMP get community

item [**string**] column name we should report the results under

max [**integer number**] maximum number of processes of that name that may run (default: infinite)

min [**integer number**] minimum number of processes of that name that must run (default: 1)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

proc [**string**] name of a monitored process

pscomm [**string**] ps command to use (default: system dependent)

7.25 readbandwidth

Name

readbandwidth – Monitors data transfer bandwidth (reading mode)

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] readbandwidth
```

Description

Monitors data transfer bandwidth (reading mode)

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under (default: file)

path [**string**] network path of the testfile

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

red [**integer number**] Minimum file access time or data transfer bandwidth unless red status is reported

size [**integer number**] size of the testfile (e.g. size=10M or size=100kB). Default: 100kB

yellow [**integer number**] Minimum file access time or data transfer bandwidth unless yellow status is reported

7.26 realhttp

Name

realhttp – Monitors a service specified via an URL

Synopsis

```
[parameter1=value1][parameter2=value2][ ... ] realhttp
```

Description

Monitors a service specified via an URL

Availability

- unlimited

Options

Global Options

extraheaders [**string**] custom header that should be sent to the server

extraheadersvalue [**string**] data that should go with extraheader

item [**string**] column name we should report the results under

match [**string**] regular expression a retrieved file must match (default: none)

maxsize [**integer number**] maximum file size (default: 1024)

nomatch [**string**] regular expression a retrieved file must not match (default: none)

noredirect [**boolean**] if true realhttp will not follow redirects (default: true)

password [**string**] password (for pages needing authentication only)

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

proxy [**string**] http proxy url, if set the requests go through the proxy

timeout [**integer number**] Timeout in seconds (default: 8)

url [**string**] Service URL

user [**string**] user name (for pages needing authentication, only)

7.27 service

Name

service – Monitors running services

Synopsis

[parameter1=value1] [parameter2=value2] [...] service

Description

Monitors running services

Availability

- windows

Options

Global Options

item [string] column name we should report the results under (defaults to procs)

perf [integer number] time in minutes between sending performance data (0 for no performance data)

service [string] name of a monitored service (short or full name)

7.28 smtp

Name

smtp – Monitor SMTP service. Shortcut for: service=smtp tcp

Synopsis

[parameter1=value1] [parameter2=value2] [...] smtp

Description

Monitor SMTP service. Shortcut for: service=smtp tcp

Availability

- unlimited

Options

Global Options

item [string] column name we should report the results under

perf [integer number] time in minutes between sending performance data (0 for no performance data)

port [integer number OR string] port number or port name uxmon will try to connect to

service [string] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [integer number] time in seconds until which a response must be received (red)

7.29 snmpvar

Name

snmpvar – Monitor a given SNMP variable and create alerts if a given threshold is reached.

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] snmpvar
```

Description

Monitor a given SNMP variable and create alerts if a given threshold is reached.

Availability

- remote

Options

Global Options

community [string] The SNMP get community

item [string] Item as which to report.

lower-red [integer number] Red alarm if value drops below this value.

lower-yellow [integer number] Yellow alarm is raised if value raises below this value and is still above lower-red.

perf [integer number] report perf data every perf minutes

upper-red [integer number] Red alarm if value raises beyond this value.

upper-yellow [integer number] Yellow alarm is raised if value raises above this value and is still below upper-red.

variable [string] SNMP variable to query (cf. etc/mibs.txt file).

7.30 ssh

Name

ssh – Monitor SSH service. Shortcut for: service=ssh tcp

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] ssh
```

Description

Monitor SSH service. Shortcut for: service=ssh tcp

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.31 tcp

Name

tcp – Monitors a generic TCP based service

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] tcp
```

Description

Monitors a generic TCP based service

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.32 telnet

Name

telnet – Monitor telnet service. Shortcut for: service=telnet tcp

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] telnet
```

Description

Monitor telnet service. Shortcut for: service=telnet tcp

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

port [**integer number OR string**] port number or port name uxmon will try to connect to

service [**string**] name of the service uxmon expects to be run behind the port (e.g. 'ftp', 'pop3')

timeout [**integer number**] time in seconds until which a response must be received (red)

7.33 users

Name

users – Monitor the number of logged on users on a SNMP aware device (Host-MIB)

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] users
```

Description

Monitor the number of logged on users on a SNMP aware device (Host-MIB)

Availability

- local / unix
- remote
- remote / netware

Options

Global Options

item [string] Item as which to report.

perf [integer number] report perf data every perf minutes

remote / netware

Monitor the number of logged on users on a netware server

red [integer number] The test goes red if the number of logged in users goes above this level (default: #licenses-1)

yellow [integer number] The test goes yellow if the number of logged in users goes above this level (default: #licenses-10%)

local / unix

Monitor the number of logged on users on a Unix machine (who command)

yellow [integer number] The test goes yellow if the number of logged in users goes above this level (default: no limit)

red [integer number] The test goes red if the number of logged in users goes above this level (default: no limit)

remote

Monitor the number of logged on users on a SNMP aware device (Host-MIB)

yellow [integer number] The test goes yellow if the number of logged in users goes above this level (default: no limit)

red [integer number] The test goes red if the number of logged in users goes above this level (default: no limit)

7.34 writebandwidth

Name

writebandwidth – Monitors data transfer bandwidth (writing mode)

Synopsis

```
[parameter1=value1] [parameter2=value2] [ ... ] writebandwidth
```

Description

Monitors data transfer bandwidth (writing mode)

Availability

- unlimited

Options

Global Options

item [**string**] column name we should report the results under (default: file)

path [**string**] network path of the testfile

perf [**integer number**] time in minutes between sending performance data (0 for no performance data)

red [**integer number**] Minimum file access time or data transfer bandwidth unless red status is reported

size [**integer number**] size of the testfile (e.g. size=10M or size=100kB). Default: 100kB

yellow [**integer number**] Minimum file access time or data transfer bandwidth unless yellow status is reported

Chapter 8

Old-style Health Checks

8.1 cpuload

Name

cpuload – check the CPU load as reported by the 'uptime' command

Synopsis

```
[localhost cpu_yellow=10 cpu_red=20]cpuload
```

Description

check the CPU load as reported by the 'uptime' command. cpu_yellow defaults to 10, cpu_red to 20.

8.2 statusfile

Name

statusfile – read status information from a file and report it to the Status Collector

Synopsis

```
[localhost file=adm/mystatus]statusfile
```

Description

This monitor is thought to be useful for interfacing to external simple monitors - they can write their status to a file rather than caring about TCP connections to Status Collectors, e.g.:

```
echo "status myhost.mytest green wow - everything is ok" >adm/mystatus \  
echo "status myhost.oterh yellow something went wrong" >> adm/mystatus
```

would do ...

8.3 bbscript

Name

bbscript – use BB style monitor script

Synopsis

```
[ localhost env="LIMIT1=5;LIMIT2=10" file=adm/bb-oracle.sh]bbscript
```

Description

"file=" must point to the script to execute each 5 minutes, "env=..." lists optional environment variables to be set in the scripts environment. The common variables (such as BBHOME and the like) are automatically set.

8.4 http

Name

http – check http response

Synopsis

```
[server url=http://some.url.de]http
```

or simple

```
[server]http
```

Description

When used without arguments it will connect to port 80 and try to get the file "/". Other URLs/ports may be passed. The "check" column the check results will appear under on the status display can be specified via the "check" argument as in

```
server check=proxy
```

The http check is based on the "tcp" check explained in its own manpage. The same additional arguments (especially "timeout=") apply to http also.

8.5 tcp

Name

tcp – check if the host does respond to tcp connection request

Synopsis

```
[ service=pop3,smtp,printer tcp service=ftp timeout=20]tcp
```

Probing well known services on non-default ports:

```
[server1 check=tcp service=ftp(21),smtp(3325)]tcp
```

Content awareness:

```
[ server1 check=tcp service=custom(80) send="HEAD / HTTP/1.0\r\n\r\n" expect="HTTP"]tcp
```

Description

Some well known services (such as smtp, pop3, nntp, ica) will be recognized and not only connected to but also checked against some expect/send pairs (e.g. when checking SMTP uxmon will expect an answer starting with '22'). `service` is a variable set to a comma separated list of services 'tcp' should check. `timeout` is the maximum time the tcp check waits for a response (default: 8s).

Some well known services have their own aliases, so they can directly be listed without "service=... tcp", e.g.

```
server1 pop3 smtp printer
```

is ok.

You can apply a test for a well known service to a non-default port by using the syntax `service=service(port)`. Content awareness can be added. The semantics of above given example is: connect to port 80 of server1, send the string specified with the `send` argument, check if the stream server1 sends contains the regular expression specified with `expect`.

8.6 ping

Name

ping – does a ping

Synopsis

```
[server1]ping
```

```
[server1 proto=udp]ping
```

```
[proto=tcp]ping
```

```
[proto=icmp]ping
```

```
[server2 proto=fping]ping
```

special protocols:

```
[server1 proto=external pingcmd="ping -c1" ]ping
```

Description

ICMP pings are only possible when uxmon is running with root privileges. By default 'ping' will therefore use the UDP protocol. Ping supports icmp/udp/tcp though. Most of the IP stacks are implementing both ICMP and UDP echo services ...

For running uxmon with root privileges create file `uxmon-asroot` and `bb_start` will start up uxmon as root.

if `proto=external` then a operating system command will be executed instead of using the built-in ping methods.

`proto=karl` is another special implementation of using the operating system ping. It is limited to Solaris though.

8.7 rpc

Name

rpc – does an 'rpc ping

Synopsis

```
[server1 rpc=mount,nlm rpc rpc=nfs version=3]rpc
```

Description

The healthcheck sends a 'NULL' remote procedure call to the respective program and checks for a correct answer.

Note

Currently known programs are:

mount, nfs, nlm, yp, yppasswd (list can be extended in Monitor::rpc_ping)

Note: rpc does need a working portmapper on the remote system

8.8 procs

Name

procs – checks for running processes or services (Win32)

Synopsis

```
[localhost procs=nfsd(1-16),sendmail,lpd(1-40)]procs
```

("there must be 1 to 16 nfsd processes, at minimum one sendmail process and 1 to 40 lpd processes")

```
[localhost pscomm=ps cax procs=nfsd(1-16),sendmail,lpd(1-40)]procs
```

uses the command `ps cax` for finding the running processes

Description

The `ps` command used for determining the list of processes is guessed by the `procs` check. If available it preferably uses `ps -e`, on systems without a working `ps -e` it tries to use `ps cax`. Both `ps -e` and `ps cax` display the process name as being the same of argument 0 passed to a program. If you prefer checking against the full command line you can use

```
localhost alternate=yes procs=nfsd,oracleinst1 procs
```

This will make `procs` use the `ps -ef` command.

Note

On Win32 systems the `pscomm` argument is ignored.

On Win32 systems it is possible to monitor remote systems.

8.9 diskfree

Name

diskfree – checks file systems for free space

Synopsis

```
[ localhost type=ufs fs=/(1000-5000),/var(10000-20000)]diskfree
("status red if / is below 1MByte free, yellow if below 5MBytes,..., the file system type is ufs")
[localhost fs=all-ext2(10%-20%),all-fat(5%-15%),/boot(500k-1000k)]diskfree
("status red if /boot goes below 500k free, one of the 'fat' filesystems goes below 5% free or one of the ext2 filesystems (except /boot) go below 10% free, yellow if /boot goes below 1000k free or one of the fat ...., green otherwise")
```

Description

Note

The all... syntax should be used in preference to the "type=" argument

On Win32 systems only percentual limits are supported and Big Sister does not distinguish between NTFS or FAT filesystems.

If specifying specific filesystems the respective file system will only be checked if it is of the same type as specified in the "type=" argument or one "fs=all-type"-Argument!

8.10 diskload

Name

diskload – check the average disk load (4 minutes) as reported by 'sar'

Synopsis

```
[ localhost yellow=3 red=8]diskload
```

Description

Reports status yellow when load>3%, reports status red when load>8%

8.11 load

Name

load – check for CPU idle time, I/O-wait, freeswap as reported by sar (4 minutes period)

Synopsis

```
[ localhost idle=10 wio=50 freeswap_red=100000 freeswap_yellow=200000]load
```

Description

Default values

idle=15, wio=50, freeswap_red=20000, freeswap_yellow=60000

The above example reports 'yellow' when %idle is below 10%, or %wio is greater than 50% or freeswap is below 200000 blocks, report red when freeswap is below 100000 blocks

8.12 dumpdates

Name

dumpdates – check for last backup if using dump/ufsdump

Synopsis

```
[ localhost fs=.7(6-10),.0(30-40),/dev/rdisk/c0t1d0s0.7(1-2) ]dumpdates
```

Description

yellow: last level 7 backup older than 6 days, last level 0 backup older than 10 days, last level 7 backup of /dev/rdisk/c0t1d0s0 older than 1 day

red: last level 7 backup older than 10 days, last level 0 backup older than 40 days, last level 7 backup of /dev/rdisk/c0t1d0s0 older than 2 days

Note

The dumpdates check needs a working "mount" command to work correctly

8.13 syslog

Name

syslog – check system log files

Synopsis

```
[ localhost syslog cfg=/etc/bs_syslog.cfg ]syslog
```

Description

syslog does need its own configuration file. By default this is etc/syslog. See below for a description of the file format

On startup syslog will re-read the last 15 minutes of the log file but at most 30kBytes

8.14 eventlog

Name

eventlog – check event log on Win32 systems

Synopsis

```
[server1]eventlog
```

Description

eventlog does need its own configuration file. By default this is etc/eventlog. This config file is the same format as the one for the syslog monitor. See below for a description of the syslog config file.

8.15 bbdisplay

Name

bbdisplay – check the log file being generated by bstrapd (var/snmp_traplog)

Synopsis

```
[server1]snmp_trap
```

Description

This is only useful if bstrapd is running (usually bstrapd is started up if its configuration file adm/bstrapd.cfg exists)! snmp_trap has its own configuration file called etc/snmp_trap. See below for a description (same format as syslog config file).

8.16 snmp

Name

snmp – remotely monitor hosts running SNMP agents

Synopsis

```
[server1 novell hub2 hub server3 type=ping,net,storage,nwusers,cpu,novell snmp router-1 community=secret type=net,ping]snmp
```

Description

The default community (when not passwed with "community=...") is 'public'. "type" may be a list of checks out of the following:

ping report in host.conn if snmp poll was successfull

net check any network interface for InputErrors and OutputErrors and report a failure if device reports more than 2/s (yellow) or 6/s (red)

storage (currently only together with "novell" and "nt") use the 'hrStorage' oid for checking disk and memory usage

cpu uses 'hrProcessorLoad' to monitor CPU load. Does report "yellow" if load is>80%

nwusers (will only work with Netware servers) Does check 'nwMaxLogins' against 'nwLoginCount', report 'yellow' if only 10 users are left, 'red' if less than 2 are left

hub / nt / novell / caty / cds / notes / linux these do not include any check but will tell 'snmp' which types of machine are checked. Some snmp checks are depending on the type of host:

- `hub`: network hub
- `nt`: Windows NT
- `novell`: Novell Netware
- `caty`: Cisco Catalyst
- `cds`: Axis CD-Server
- `linux`: Linux (with UCD SNMP)

Note

`snmp` does read the file `etc/mibs.txt`. By default this contains only the Internet MIB-II (`ping/net`). You need to add more mibs from the `contrib/mibs` directory, e.g.

```
cat host.txt mib-2.txt nwhostx.txt nwserver.txt > mibs.txt
```

8.17 OV

Name

OV – monitor HP Openview trapd.log

Synopsis

```
[ localhost cfg=/etc/bs_OV.cfg ]syslog
```

Description

OV does need its own configuration file. By default this is `etc/OV`. See below for a description of the file format. On startup OV will re-read the last 24 hours of the log file

8.18 metastat

Name

metastat – monitor Solstice DiskSuite metadevices

Synopsis

```
[ localhost stat=/usr/opt/SUNWmd/sbin/metastat ]metastat
```

Description

The default value of `stat` is `/usr/opt/SUNWmd/sbin/metastat`, so `stat=` can be omitted in most cases.

8.19 ntp

Name

ntp – check if the machine is running an ntp server

Synopsis

```
[foo]ntp
```

8.20 mrtg

Name

mrtg – receive and visualize mrtg data

Synopsis

```
[foo.bar.com prefix=10.1.1.253/10.1.1.253.1 column=mrtg maxlev=10485760 bits=1]mrtg
```

Description

The MRTG monitor will retrieve data from a MRTG-style .log file.

It will check the last value of the data being monitored against given thresholds and act accordingly. In the example above, the log file signifies a router interface. The log file is found in mrtglog directory. (see below)

Available arguments:

`column` the column to which status messages are posted default: mrtg (i.e post to foo.bar.com.mrtg)

`prefix` the directory/file prefix of the logfile no default, required.

`bits` determines if we are graphings bits/sec or bytes/sec (used w/ network interfaces) default: bytes

`maxlev` the maximum value the data may take on Only used to calculate the percentage reported and to calculate yellow/red levels. default: 10mb (bits=1), 1.25mB (bits=0)

`warnlev` These two options determine at which point the `paniclev` monitor should report yellow and red statuses. (0 -> warnlev is green, warnlev-> paniclev is yellow,>paniclev is red) You may specify a value to compare to the counter directly or a percentage of the maximum value. default: warnlev= 50%, paniclev= 75%

`units` This determines the units of the value. Asthetic only. default: b/s (bits=1), Bytes/s (bits=0)

`intext` These are the labels for the data being graphed.

`outtext` For network interfaces, In and Out are the default and should be sufficient.

Resources variables:

`mrtglog` the directory with mrtg logs in it

`mrtgweb` the path for the url to link to the graph

`mrtgloghtml` 1 to post html status, 0 to do text only So, it reads the file `$mrtglog/$prefix.log`, posts the status as a line of text, and optionally links to and to show the graph and get more info.

8.21 atmport or etherport

Name

atmport or etherport – check operating status of specific ports

Synopsis

```
[ fooswitch switch=192.168.0.4 port=3A2 vpi=0]atmport
```

```
[fooswitch switch=192.168.0.4 port=5]etherport
```

Description

The atmport monitor is configured to work with Marconi ATM gear. It has been tested with the ASX200BX, ASX1000, LE25, LE155, and ESX3000. It will use SNMP to query the switch specified for the status of the VPT given. It checks that signalling is up. The etherport module queries an ethernet switch for the link state of that port and reports accordingly.

`switch` the ip address of the host to SNMP query

`community` the SNMP community to use default: public

`port` (atmport) Symbolic name for port (Marconi convention) or the number used as the SNMP index (etherport) SNMP index number for the port.

`vpi` (atmport only) the Path number to monitor

To find numeric port numbers for the etherport module, use `snmpwalk`:

```
snmpwalk 10.100.101.61 public ifDescr
interfaces.ifTable.ifEntry.ifDescr.2 = FastEthernet0/1
interfaces.ifTable.ifEntry.ifDescr.3 = FastEthernet0/2
```

For FastEthernet0/1, I would use port 2. `ifName` or `ifAlias` may yield the desired result depending on the switch manufacturer.

8.22 software

Name

`software` – get firmware release via SNMP and check against an expected version

Synopsis

```
[ foo type=asn9000 expected=ForeThought.*6\.0\.1]software [foo2 expected=Linux.*2\.0\.36] software
```

Description

Reports a yellow status if the reported and expected versions do not match.

The `expected` argument is a regular expression tested against the value of `system.sysDescr.0`.

Some special type values exist:

```
type=asx for ASX ATM Switches checks ASXSoftware.0 instead of sysDescr
type=asn9000 for ASN9000 PowerHub checks ASNSoftware.0 instead of sysDescr
type=es2810 for ES2810 Ethernet Switch
```

8.23 realhttp

Name

`realhttp` – execute an HTTP(S) request for a specified url

Synopsis

```
[localhost url= realhttp url= item=cgi]realhttp [url=https://secure.mybc.com:447/index.html item=secure] r-  
ealhttp [url= greppos=success] realhttp [url= grepneg=test] realhttp
```

Description

This module is used to execute an HTTP request for a specified url. It utilizes the LWP libraries so it can support HTTPS if the Crypt/SSLay libraries have been installed on your system.

A query is considered successfull if LWP reports success. There is no method currently implemented to check that the page returned is what would be expected.

This check will only work if you have the LWP::UserAgent perl module installed (see CPAN)

Arguments:

`url` The url to retrieve.

`item` The item to report this test as. Defaults to 'http'.

`greppos` A regular expression which should be found in the contents returned from the server.

`grepneg` A regular expression which should *NOT* be found in the contents returned from the server.

`user` A username to supply in basic authentication

`pass` A password to supply in basic authentication

8.24 memory

Name

memory – report memory usage on NT and Linux systems

Synopsis

```
[localhost]memory
```

8.25 oracle

Name

oracle – tests an oracle database server

Synopsis

```
[ localhost ORACLE_CONNECT=... ORACLE_HOME=... ORACLE_BASE=... ORA_USER=... ORA_P-  
ASS=... ORACLE_TNSADMIN=... ORACLE_NLS_LANG=... ORACLE_ORA_NLS33=...]oracle
```

Requirements

Perl Requirements:

- DBI
- DBD Oracle

Oracle Requirements

- Installed Oracle (tested with 8.0.[56] and 8.1.[56])
- Spezial Oracle User and View

Oracle preparation

Create Special Oracle User and special View to connect reduce security problems and enable Big Sister to fetch state. Of course you can also use a dba user but we don't prefer username and password from highly privileged users in config files.

Edit PLSQL Script (see sample/oracle/bs-user-view.sql in your distribution) with password for sys and change username and password according your needs:

Default:

```
[username]/[password] sys/change_on_install check_db/check_db
```

Arguments

ORACLE_CONNECT: Connect String constisting looking like
[HOSTNAME]:[ORACLE_SID]:[BIG_SISTER_COLON_NAME]

Eg. Your database runs on "Asterix" and the Oracle SID you want to surveille is called ↔
TVD806. You want to have the collon beeing called "ORACLE DEMO". So your ORACLE_CONNECT ↔
string looks like: "Asterix:TVD806:ORACLE DEMO".

ORACLE_HOME: Home directory of the oracle product (where you have installed oracle version)

ORACLE_BASE: Home of the oracle user.

ORA_USER:User to connect to oracle db. (with our prepared PLSQL default is check_db)

ORA_PASS: Password for oracle user ORA_USER (with ourprepared PLSQL default is check_db)

ORACLE_TNSADMIN: Path where oracle will find the tnsnames.ora

optional fields: ORACLE_NLS_LANG, ORACLE_ORA_NLS33

eg:

```
localhost ORACLE_CONNECT=ds1skeys:DKMS2:oAPP2\  
ORACLE_HOME=/u00/app/oracle/product/8.1.6\  
ORACLE_BASE=/u00/app/oracle\  
ORACLE_NLS_LANG=american_america.WE8ISO8859P1\  
ORACLE_ORA_NLS33=/u00/app/oracle/product/8.1.6/ocommon/nls/admin/data\  
ORA_USER=TSMW ORA_PASS=TSMW ORACLE_TNSADMIN=/u00/app/oracle/network\  
oracle
```

Reporting

Big Sister reports three states for oracle:

- red DB is not available
- yellow DB is up, but in restrict mode
- green DB is ready

Philip Markwalder Philip.Markwalder@trivadis.com

8.26 tripwire

Name

tripwire – check system consistency via tripwire

Synopsis

```
[localhost_]tripwire
```

Description

Has been tested with RedHat 7.0 and Solaris 7.0 with commercial/free version of Tripwire 2.3.

Tripwire 2.3 Portions copyright 2000 Tripwire, Inc. Tripwire is a registered trademark of Tripwire, Inc. This software comes with ABSOLUTELY NO WARRANTY; for details use --version. This is free software which may be redistributed or modified only under certain conditions; see COPYING for details. All rights reserved.

8.27 ldap

Name

ldap – query an ldap server and attempt to retrieve a register

Synopsis

```
ldap_mozilla
```

Description

The `ldap` check uses `Net::ldap` module which is available at CPAN while `ldap_mozilla` uses `Mozilla::LDAP` available from Mozilla.org.

Parameters

`dn` DN to check

`pass` UserPassword

`port` ldap tcp port (default: 389)

`item` The item to report (default: ldap)

`time` The number of seconds to wait for a response before timing out. (default: 5)

```
myhost dn= cn=whoever,o=wherever,c=us\
pass=test ldap
```

Manuel de Vega Barreiro mbarreiro@red.madritel.es (Madrid, Spain)

Based in Kevin O'Donnell's kevin_odonnell@telus.net Radius Monitor

8.28 command

Name

command – execute an arbitrary command

Synopsis

```
[ localhost exec=/opt/bs/otros/test.pl test1 time=10 item=test]command
```

```
[ localhost exec=ntpdate -q localhost results=0:green:ntp OK;stratum 16:yellow:ntp running but not synchronized;timeout:red:ntp not running;*:red:ntp FAILURE]command
```

Note

commands are executed with "/" base directory. uxmon-net

Description

This executed command must return status 0 if test is ok, and status != 0 if it is wrong.

In order to allow interfacing to commands not explicitly written for use with Big Sister the default behaviour regarding exit codes and other error conditions is configurable (see "results" parameter below).

All text messages send to standard output/error is send as status messages to bsdisplay.

Parameters

`exec` command to execute

`item` The item to report (default: command)

`time` The number of seconds to wait for a response before timing out. (default:5)

`results` patterns and resulting status to report when a pattern matches. Patterns are separated by semicolons and the results string looks like

where pattern may be one of

: matches the commands exit code
 timeout : matches if command timed out
 failed : matches if uxmon was not able to even start the command
 : matched against the stdout/ stderr output of the command

The first matching pattern decides which rule will apply.

8.29 ups

Name

ups – query an uninterruptible power supply

Synopsis

[myups]ups

Description

The probed power supply must support SNMP queries via the UPS MIB.

This test is very limited. It reports green unless

- the number of "Line Bads" increased during the last hour => yellow
- the reported power source is something different from "normal" => red

In addition to this it collects some useful performance data like the Battery voltage (indicating battery quality), the output power and output load. The standard graphtemplates file includes graph definitions for these values.

8.30 qmqueue

Name

qmqueue – monitor qmail's mail queue

Synopsis

[localhost queue_dir=/var/qmail/queue/ queue_yellow=2000 queue_red=3500]qmqueue

Description

Checks the number of messages in qmail's queue, raises a warning if message number passes the queue_yellow limit, an error if it passes the queue_red limit. Queue_yellow defaults to 2000, queue_red to 3500, queue_dir to "/var/qmail/queue".

Chapter 9

Server Command Reference

9.1 join

Name

join – join a group of network objects

Synopsis

```
join [ group group1 ... groupn ]
```

Description

The host or group called `group` joins the groups `group1` through `groupn`. Non-existent groups are automatically set up when the first member joins them. note: If the `%Autojoin` feature of `bbd` is active, any host appearing automatically joins one or more groups.

9.2 leave

Name

leave – leave a group of network objects

Synopsis

```
leave [ group group1 ... groupn ] leave [group *]
```

Description

The host or group called `group` leaves the groups `group1` through `groupn`

The host or group called `group` leaves any group it was in, any cached information for this host is deleted.

Groups are dropped when their last member leaves.

9.3 displayname

Name

displayname – set text appearing on web pages

Synopsis

```
displayname [ grouptext]
```

Description

Set the Text appearing on the Web pages for the host or group called *group* to *text*. All the text up to the end of the line is treated as one argument - the text may contain spaces.

9.4 status

Name

status – force status of a specified check

Synopsis

```
status [hostname.check color comment]
```

Description

Set the status of the check called *check* of the host called *hostname* to *color*. Allowed colors are red, yellow, purple and green. The comment may be an arbitrary (multi-lined) text. By convention it starts with

(seconds since January 1 1970) time-and-date-in-human-readable-format

though. A valid status command therefore is for instance: `status myhost.bak red (926008681) Thu May 6 18:38:01 1999 backup failed` Status messages exchanged between Big Sister status collectors usually will have "(proxy delay: XXs)" added after the human readable time, e.g.

```
status myhost.bak red (926008681) Thu May 6 18:38:01 1999 (proxy
delay: 126s) backup failed
```

9.5 page

Name

page – implemented for backward compatibility with big brother

Synopsis

```
page [text]
```

Description

Send a paged alarm.

9.6 savelogs

Name

savelogs – rotate big sister logfiles

Synopsis

```
savelogs [tag]
```

Description

The optional tag (alphanumeric string) does associate an identifier with savelogs e.g. used with sendlogs. Savelogs will not be executed if savelogs has already been executed for this tag.

9.7 sendlogs

Name

sendlogs – send history logfile

Synopsis

```
sendlogs [tag]
```

Description

Sends the history log file called "tag" to the client. The transmission ends with a line containing only "--END". If the tag is not known (savelogs must have been executed for this tag) no file transmission will take place.

This command is mainly used in "bsadmin archivelogs" (see).

9.8 perf

Name

perf – send history logfile

Synopsis

```
perf [time host:variable value]
```

Description

Log performance data. The sample for host 'host', variable 'variable' at time 'time' was of value 'value'.

9.9 remove**Name**

remove – discard all collected data of specified host.check

Synopsis

```
remove [host.check]
```

Description

Discards in-memory tracks of host.check on the server. The status light for host.check disappears.

9.10 event**Name**

event –

Synopsis

```
event [time host.item priority text]
```

Description

Chapter 10

Server configuration reference: bb-display.cfg

10.1 %Option

Name

%Option – specify various not necessarily display related parameters

Synopsis

```
%Option [ ImmediateHTML KeepGroups BBLog StartOK ]
```

Description

If arguments are prefixed with a '-' switch the option off, if prefixed with '+' switch it on. If no prefix is given '+' is assumed.

ImmediateHTML

(default: off)

write HTML status files immediately on status receipt no matter if the status text has changed or not.

KeepGroups

(default: off)

read saved grouping information on startup and whenever the configuration changes - do not lose dynamic group information

BBLog

(default: on)

Log incoming status messages to www/logs/*.*** files ("BB compatible logging")

StartOK

(default: off)

When the display server receives the first status message for a new host/check, assume the status was green before. This has effects on alarming: If StartOK is off then no alarm will be generated if a newly added check already reports "down". If StartOK is on then the alarm generator will handle the status as if it had changed from green to the reported status.

DNS

(default: off)

if disabled, bbd and bsmon will never try to use name resolution. The effect of this is that you may not use host or domain names in the permissions file. Enabling name resolution on the other hand may result in serious performance degradation if the name resolution service is hit by a problem - however, it is essential that Big Sister works well in case of troubles, so making Big sister depend on the network/systems it should monitor is never a good idea.

10.2 %Autoconn

Name

%Autoconn – Tell bbd to automatically set the status of host.conn to 'green' each time a connection is coming in from this host

Synopsis

```
%Autoconn [host1 ... hostn]
```

Description

Tell bbd to automatically set the status of host.conn to 'green' each time a connection is coming in from this host. Set status to 'red' (not to 'purple') if no report for >15 min.

Note :

note: the host name must be resolvable!

10.3 %Autojoin

Name

%Autojoin – automatically put any newly appearing host into the specified group(s)

Synopsis

```
%Autojoin [new | all | all_hosts] [groupname]
```

Description

```
%Autojoin new GROUP
```

tells bbd to automatically put any newly appearing host into the group 'GROUP' ("newly appearing" means that bbd is receiving status messages for a host not yet known)

```
%Autojoin all_hosts GROUP
```

tells bbd to automatically put every known host into group 'GROUP' (where "host" means every object bbd is receiving status messages for)

```
%Autojoin all GROUP
```

tells bbd to automatically put every known object - hosts and groups - into group 'GROUP'

10.4 %Autojoin pattern

Name

%Autojoin pattern – filter newly appearing hosts by host name and put them automatically into the specified group(s)

Synopsis

```
%Autojoin pattern [pattern] [groupname]
```

Description

This is similar to **%Autojoin** but only hosts selected by pattern matching their host name are joined.

```
%Autojoin pattern a.* GROUP
```

tells bbd to automatically put any newly appearing host with a name starting with "a" into the group 'GROUP' ("newly appearing" means that bbd is receiving status messages for a host not yet known)

10.5 %Port

Name

%Port – specify the port the Big Sister server should listen for agent connections

Synopsis

```
%Port [port_number]
```

Description

```
%Port port_number
```

Will make the Big Sister server listen on port "port_number" for incoming agent connections.

10.6 %Pager

Name

%Pager – provided for compatibility with Big Brother

Synopsis

```
%Pager [command]
```


Description

Big Brother clients use to send pages directly via their Display Server. Usually - in a Big Brother environment - you will use %Pager like:

```
%Pager /usr/where-bb-is/bin/bb-pager.sh
```

10.7 %Groups

Name

%Groups – determine group memberships

Synopsis

```
%Groups
```

Description

The %Group statement is useful to determine alias names and group memberships. The status of single hosts as well as the status of predefined groups can be displayed in your html output. The color of a group status light reflects the status of the member with the highest severity level. E.g. if a group consists of few subgroups and several hosts, the status of the whole group is set to red of any single host (no matter if it is member of a subgroup or not) switches to red.

The %Groups statement is followed by a number of lines with the syntax:

```
name(Display name) GROUP1 ... GROUPN
```

The meaning is: 'name' will be printed as 'Display name' when appearing on a html page and belongs to the groups GROUP1 ... GROUPN.

The group definitions are recursive. This means that e.g.:

```
host1(Computer 1) GROUP1 GROUP2
```

```
GROUP1(Group of computers) GROUP3
```

is valid and means that the reported status values for host1 will influence not only GROUP1 and GROUP2 but also GROUP3.

10.8 %StatusTrap

Name

%StatusTrap – send status change SNMP traps to a target host

Synopsis

```
%StatusTrap [ target ] [ options ]
```

Description

Specifies a destination for status change traps. On every status change of a host/test, Big Sister will send an SNMP trap to this destination. The target argument must be defined as *community@host*. Currently, only one option, *agent*, is known. If set, the agent field in the trap will be set to the name of the monitored host, thus the trap receiver will look at Big Sister as a SNMP trap proxy forwarding traps from the monitored host.

```
%StatusTrap public@1.2.3.4 agent
```

will make Big Sister send status change traps to host 1.2.3.4 with community public.

10.9 %Page

Name

%Page – describing the content of a webpage

Synopsis

```
%Page [ name ] [ title ]
```

name: the file name of the created page. .html is automatically appended if the name is not ending in .htm or .html. The file is created in the directory *www*

title: the title of the created page

Description

The %Page statement is usually followed by one or more statements describing the contents of the page.

See also: %skin, %Logskin, %title, %refto, %itemref, %table, %image, %ref, %frameset and %-include

10.10 %Section

Name

%Section – add a section in the menu bar of the server display

Synopsis

```
%Section [ title ]
```

Description

Adds a section name in the menu bar.

10.11 %title

Name

%title – specify names for generated tables

Synopsis

`%title [title]`

Description

This statement should only be used within `%Page` statements! It tells 'bbd' which title should be used for the tables generated by following `%table` statements. If `title` is set to `auto` the display names as defined in the `%Groups` section are used.

10.12 `%refto`

Name

`%refto` – set individual hypertext links for specific groups or hosts

Synopsis

`%refto [url]`

`%refto [name url]`

Description

This statement should only be used within `%Page` statements! When creating tables 'bbd' will usually create hypertext links for hosts or groups appearing in the table (left row). This is done by appending a "#" character plus the name of the host or group to the url given.

With the `name` argument, setting individual hypertext links for specific groups or hosts is also possible. If a group or host called `name` appears in a table the `%refto url` is overridden by the url given in the respective `%refto name url` command. You can set up a table of name/url pairs by using multiple `%refto` statements. The table is truncated with the statement `%refto clear`.

Note

Special pseudo-urls:

none - omit creating hypertext links self - the host/group will be found on the same page

clear - clear table with individual urls (see above)

10.13 `%itemref`

Name

`%itemref` – influence hypertextlinks of status lamps

Synopsis

`%itemref [url]`

Description

This statement should only be used within `%Page` statements. When creating tables 'bbd' will usually create hypertext links for each status lamp in the table. This is done by appending a '/' character, the name of the host/group, a '.' character plus the name of the status item to the url given.

If a file with the prefix '.html' exists, then this one is used...

Note

Special pseudo-urls:

none - omit creating hypertext links

logs - point to the status message collected by the Status Collector

html - point to the HTML version of the status message

10.14 %table

Name

`%table` – create a table with the specified hosts and groups

Synopsis

```
%table [GROUP1 ... GROUPN] [HOST1 ... HOSTN]
```

Description

This statement should only be used within `%Page` statements. It tells 'bbd' to create a table for each of the arguments containing all the hosts/groups contained in the respective group.

Since 0.22beta each group may be prefixed by one or more "+" characters. In this case the table will not contain the groups/hosts in the respective group but all the groups/hosts found when descending the group tree.

10.15 %sort

Name

`%sort` – sort order for hosts in subsequent tables

Synopsis

```
%sort [criteria]
```

Description

This statement should only be used within `%Page` statements. It specifies the order in which rows in subsequent tables should be ordered. The criteria must be one of "severity" (hosts with most severe status first) and "name" (sort by name).

10.16 %select

Name

`%select` – filter rows that should be listed in subsequent tables

Synopsis

```
%select [comparison] [color]
```

Description

This statement should only be used within `%Page` statements. It selects the host that should appear in subsequent tables. "comparison" must be one out of "<", ">", "<=" or ">=", color must be one of the known status colors. Only hosts with the selected status will be displayed. For instance,

```
%select <green  
%table ALL
```

will display a table containing all hosts that have at least one test with a status worse than green.

10.17 %select_names

Name

`%select_names` – filter rows that should be listed in subsequent tables by entry name

Synopsis

```
%select [pattern]
```

Description

This statement should only be used within `%Page` statements. It selects the entries that should be displayed in subsequently specified tables by entry name (host name). The pattern is a perl regular expression. Multiple patterns may be listed. For instance,

```
%select_names s.* a.*  
%table ALL
```

will display a table containing all hosts with names starting with "s" or "a".

10.18 %select_items

Name

`%select_items` – filter columns that should be displayed in subsequent tables

Synopsis

```
%select [item/group] [ ...]
```

Description

This statement should only be used within %Page statements. It selects the columns that are to be displayed in subsequently specified tables either by their column name (item) or by item groups defined with the %Itemgroup statement.

```
%Itemgroup SYSTEMTESTS cpu disk
%select_items SYSTEMTESTS msgs conn
%table ALL
```

will display a table with the rows cpu, disk, msgs and conn.

10.19 %Itemgroup

Name

%Itemgroup – define a group of table columns

Synopsis

```
%Itemgroup [groupname] [ groupmember] [ ...]
```

Description

Define a group of columns with the name "groupname" and the members "groupmember". Only single columns may be group members, not other item groups. Item groups are useful together with the %select_items statement.

```
%Itemgroup SYSTEMTESTS cpu disk
```

will define an item group called SYSTEMTESTS containing the columns cpu and disk.

10.20 %column_wrap

Name

%column_wrap – wrap a table after a specified number of columns

Synopsis

```
%column_wrap [column_count]
```

Description

This statement must be placed within `%Page` statements. If a `column_count > 0` is defined, status tables will contain at most the specified number of columns. Rows that are going to use more than this count of columns are wrapped into multiple rows.

```
%Page ....
%column_wrap 17
%table TABLE_WITH_67_COLUMNS
# reset to normal for the remainder of the page:
%column_wrap 0
%table NEXT_TABLE
```

10.21 %image

Name

`%image` – insert an image or HTML image map

Synopsis

```
%image [cfg-file]
```

Description

Please read the section [Using Imagemaps](#) for getting more help on image maps.

Note on %image

when using `%image` you must have installed the GD perl module.

10.22 %ref

Name

`%ref` – create an HTML label / anchor at the current position

Synopsis

```
%ref [name]
```

Description

This statement should only be used within `%Page` statements! It tells 'bbd' to create a HTML label at the current position in the html code. 'bbd' does automatically generate labels for any table appearing within the page, so this statement is not commonly used.

10.23 %skin

Name

%skin – use skinset to describe the look of the created pages

Synopsis

```
%skin [skin1 ... skinN]
```

Description

Use the skinset skin1 thru skinN to describe the look of created pages. See the `www/skins` directory for valid skin names and `www/skins/*/README` files for what the respective skins are meant to introduce.

Most of the skins are incomplete - means they add certain details to another skin. Therefore you can list more than one skin in one %skin statement. The skins are treated in order - later listed skins taking precedence over earlier listed ones. The 'default' skin is always the first skin included.

Example 10.1 Usage of the %skin statement

```
%skin default white_bg static_lamps
```

will force Big Sister to create pages with the default look but with a white background rather than the variable one and static instead of blinking lights. Since the default skin is always implicitly included it's exactly the same as:

```
%skin white_bg static_lamps
```

10.24 %Logskin

Name

%Logskin – use skinset to describe the look of the created log pages

Synopsis

```
%Logskin [skin1 ... skinN]
```

Description

%Logskin is similiar to %skin, but it does define the look of log pages (the pages where the current status of a monitor is shown)

10.25 %rsync

Name

%rsync – report the stati to a remote Big Sister server

Synopsis

```
%rsync [host:port prefix GROUP1 ... GROUPN]
```

Description

This statement tells 'bbd' to regularly (each 5 minutes cycle) build a list of stati known to it and report them to a remote status collector. Each host name is prepended by a prefix (prefix "none" means no prefix). Currently only host stati can be synced. No group information is exchanged.

Let's say your bbdisplay machine is A and another bbdisplay machine is called B and you want stati arriving on B be transferred to A; -so you could do an rsync on B via:

```
%Rsync A:1984 prefix GROUP
```

Note

Using [ssh-tunnels](#) for exchanging status information and getting around firewall-"problems" is always a good idea!

10.26 %frameset

Name

%frameset – creating a frameset around the Big Sister status pages

Synopsis

```
%frameset [output-file initial-page title]
```

output-file the name of the generated html file

initial-page the name of the html file initially displayed

title the title of the page

Description

The frameset is only generated when bb-display.cfg is re-read (e.g. after a configuration change or on startup).

For %Frameset to work correctly a skinset must be used that includes frame definitions (e.g. "frames")

10.27 %include

Name

%include – include a file

Synopsis

```
%include [file1 ... fileN]
```

Description

This statement only works when used within `%Page` statements! It tells bbd to include a file whenever the respective page is rebuild. The file is included at the 'current' position, so e.g.

```
%Page ... %table TEST %include file
```

will include the file 'file' after the table 'TEST'

If no absolute path is given, the path is supposed to be relative to the Big Sister root directory
