Integrated Server Monitoring

Description of the Agents

© Wilhelm Buchholz, Im Bruche 6, 31275 Lehrte

http://www.monitor-site.de

mailto:new-monitoring@t-online.de

Contents

1. Introduction

The agents offer the following groups of functions:

- Standard Monitoring: File systems, processes, syslog respectively event logs, performance, listenports, Lifecheck/Heartbeat
- Log file analysis: Application monitoring, system monitoring
- Any monitoring scripts: Application monitoring, system monitoring
- Send direct messages to the Event Browser of the Management Station

The program for standard monitoring has finished functions with a default output display for the central Management Station.

Similar it is with the agents for log file analysis. In the configuration files - ready, new or modified - one defines file names, filters, and Severities.

The different agents do not need to be installed! The program files and associated configuration files are to be placed on the servers, which are to be monitored, each under a user ID of its choice to be operated there. The registration to the Management Station is done automatically on first use. It can also be more than one instance of an agent to be operated, if necessary, which differ in the configuration files and access rights.

Each agent can be operated either as a batch program that terminates after each call and execution, or as a background process (daemon). In the first case a scheduler calls it periodically. In the second case it is started at boot time from an rc-routine (e.g. rc.local) and determines the polling interval of an entry in the configuration file. When starting an informational message goes with the PID of the process, the name of the agent program with full path name of the configuration file to the Management Station. When you stop the background process, a message with the same information but the Severity "major" is sent to the Management Station.

The operation of the agents and the structure of the configuration files are described below with reference to examples.

The configuration files are designed on the principle of minimum input.

2. General Definitions

The configuration files for the agents consist of keywords followed by implementation details of the monitoring function such as threshold values. A comment begins with a ‘#’ to end of line. If you want to use the character '#' for another purpose, for example in a search pattern, it must be escaped with a preceding backslash ("\#")! Blank lines are ignored. The order of the keywords does not matter. The special meaning of the delimiter "::" can be turned off with a preceding backslash (\::).

A keyword must begin in the first column of the configuration file!

2.1 Severities

There are the Severities:

1) info: inform
2) min: minor
3) warn: warning
4) maj: major
5) crit: critical

The Severities "minor" and "warning" may indicate impending problems. The Severities "major" and "critical" mean the need to intervene.

2.2 Filters (Log Files, Monitoring Scripts)

A filter for the analysis of log files and the output of monitoring scripts consists of a search pattern (*extended regular expression*, POSIX standard), a format string (optional) and an indication of the Severity and the frequency of matching entries. The search pattern and the optional format string are enclosed with a quote character ‘”’ at the beginning and a quote followed by a hyphen ‘-‘ at the end and are separated by the rightmost double semicolon ";;".

If more quotes to come between the delimiting quotation marks, a backslash (\”) must be preceded him!

A filter has the form:

1) “[!]<reg.expr.>[/i|v|!][;;<format string>]“-severity: If a number in square brackets after the Severity is missing, a maximum of **five** lines found are sent to the Management Station; however, all the lines found are counted;

for negative filter the default value is **one** (at least one line has to be found)

2) “<reg.expr.>[/i|v|!][;;<format string>]“-severity[0]: Zero in brackets means that **all** lines found are sent to the Management Station
3) “<reg.expr.>[/i|v|!][;;<format string>]“-severity[N]: Maximum of N lines found are sent to the Management Station; the total number of lines found and the associated time period is sent in binary form to the Management Station and displayed there
4) “<reg.expr.>[/i|v|!][;;<format string>]“-severity[+N]: A plus sign '+' in front of N means that at most N lines found are sent to the Management Station; the total number of lines found is **not** sent to the Management Station
5) “<reg.expr.>[/i|v|!][;;<format string>]“-severity[-N]: Minus sign ‘-‘ before N means the upper threshold (maximum value) for entries found, i.e. it will transfer **one** line when more than N lines are found; the total number and the time duration are transmitted
6) “<reg.expr.>[/i|v|!][;;<format string>]“-severity[>N]: The character '>' before N also means the maximum value for found lines; however, there is only one message about the number of rows found (N >= 0)
7) “<reg.expr.>[/i|v|!][;;<format string>]“-severity[<N]: The character '<' before N means negative filter for the required minimum value for matched lines, i.e. it must be found N entries
8) “!<reg.expr.>[/i|v|!][;;<format string>]“-severity[N]: Negative filter (like item 7), N is the lower threshold (minimum) value for the number of matched lines, i.e. at least N entries must be found, otherwise a message is generated
9) “<reg.expr.>[/i|v|!]“-null: null instead of Severity means: found lines are suppressed (Suppression Filter)

N being a natural number greater than zero or greater equal zero (item 6),
severity :=info|min|warn|maj|crit

The string to the left of ";;" is the search pattern, the right of ";;" the format instruction for detected lines. If ";;" is missing, a line found in the normal case is transferred as a whole. The special meaning of the double semicolons can be turned off by a preceding backslash (\;;), if more than a “;;” occurs or it is part of the search pattern.

A leading exclamation mark '!' before the search pattern means a negative filter, that generates a message when an evaluation of the overall search procedure was unsuccessful or have been found too few entries or lines (search for absence). For negative filter the order in the list of filters does not matter for the other filters very well! The special meaning of the leading exclamation mark can be es-

caped by a preceding backslash (“\!”). The other possibility is to use the operator '<' in square brackets (item 7).

The option “/i” at the end of the search pattern means that a case insensitive matching is performed. The option “/v” means that search result is reversed and case sensitive matching is performed. The option “/!” means that search result is reversed and a case **insensitive** matching is performed.

Example: The search pattern “error/i” will match lines containing “ERROR”, “Error”, “error”. The search pattern “normal/!” will match lines **not** containing “NORMAL”, “Normal”, “normal”. The search pattern “*” has a special meaning and will match any line.

For evaluation of multiple lines, the regular expression is divided by the special character '\n'.

Example: “<pattern-1\npattern-2\npattern-3>“-warn

There is a message when the three search patterns meet in three consecutive lines.

It is a common search for a maximum of twenty lines possible. The special meaning of "\n" can be escaped by a preceding backslash ("\\n").

The lines found by the search patterns are transferred to the Management Station along with the name of the log file or the command and preceded by a prompt. If a format string is specified in the filter, instead of the whole line the result of the transformation is transmitted.

Example filter:

A sequence of three filters for the command "netstat -an" (Linux)

“ESTABLISHED“-min[-90] “[ ]LISTEN[ ]/v“-null “:(22|25|110|111|2049)[ ]/v“-warn[3]

The first filter searches for lines containing "ESTABLISHED". If more than 90 lines occur, there is exactly one message and the total number is also transmitted.

The second filter suppresses any lines that do **not** contain " LISTEN “. The third filter detects the remaining lines, which do **not** contain the (allowable) port numbers 22, 25, 110, 111, 2049. It will transfer a maximum of three lines found, the total number of items is also transferred. (for the agent program asyncmonagent , there is another representation, see below).

In the same way, the analysis of log files is done.

2.3 Format Strings (Log File Analysis, Monitoring Scripts)

A format string is used to convert a word or line of text obtained by the analysis of a log file or by evaluating a script already on agents to increase the readability and expressiveness. Alternatively or in addition, one may use the same mechanism to the Management Station to the local filters. A format string consists of various special characters that are associated with certain operations. The special meaning of the characters can be switched off, preceded by a backslash '\'.

There are the special characters:

1) $n or ${n}: n is a number [1..99]. Outputs the <n>th word of the input text
2) %n or %{n}: shift to the left, outputs the <n> columns to the left shifted input text, the input line remains unchanged
3) &n or &{n}: Shift of <n> characters to the left of the input text, there is no immediate output, the new beginning of the text input field is automatically set to the beginning of a word or column
4) &{n,m}: Outputs <m> characters from the <n>th character of the input line
5) &{n[|#]substring}: Search for a sub-string in a word. Outputs from the <n>th character to the sub-string substring in the same word. If substring is not found, the output is to the end of the word (special character is either '|' or '#')
6) @n or @{n}: Shift to the left by <n> columns in the input line, the original column <n+1> is then the beginning of the input line, there is no direct output
7) %<[n|]substring>: Search for a sub-string in the whole line or optionally after the <n>th occurrence of a sub-string in the line (n > 0). Then shift left until substring in the input line. The sub-string found is the new beginning of the input line, there is no direct output
8) ?<[n|]substring>: Outputs the sub-string shifted to the left of the text input line. If sub-string is found, terminates the formatting, otherwise continue with the following special characters; optional search for multiple occurrences
9) -<[n|]substring>: Outputs the at the point of occurrence of sub-string substring truncated input text, the found sub-string is cut off, the input line remains unchanged; optional search for multiple occurrences
10) $*: Outputs the whole line
11) $$: Outputs the last column/word of the input text
12) $0: Outputs the search pattern, that has filtered the input line

If the format string does not contain any special characters, the line found or the found entry is mapped to the string constant. The search for a sub string happens from left to right. If the sub string is not found, no operation is done.

Example 1: “Fatal|FATAL|fatal;;Line with \“Fatal\“ occurred: $1 %5“-maj

The search string as a regular expression to find a line is to the left of ";;", the format string to edit a found line starts to the right. The output line is preceded by "Line with \”Fatal\” occurred: ". Then comes the first column respectively word of the input line followed by five columns of the left-shifted input line.

Equivalently:

“fatal/i;;Line with \“Fatal\“ occurred: $1 @5$*“-maj

The operator "@5" causes the shift to five columns to the left of the input line, but no output. The issue happens with the operator "$*", which outputs the changed input line total.

Example 2:

The following line should be formatted:

```
Oct 27 11:19:57 NEPTUN kernel: [ 1739.435161] oracle[3065]: segfault at 0 ip 00007f34d47f7957 sp 00007fffb3733628 error 6 in libc-2.11.3.so[7f34d4778000+159000]
```

Filter: “segfault at;;Segmentation violation of: %<] >@1&{1|[}, %1“-crit
Output for the event text on the Management Station:

```
Segmentation violation of: oracle, segfault at 0 ip 00007f34d47f7957 sp 00007fffb3733628 error 6 in libc-2.11.3.so[7f34d4778000+159000]
```

The filter consists of the part left of ";;" (search pattern for a line) and on the right of it the treatment of the found line. The expression "%<] >" (or "%<1|] >") shifts the input line to the left until the first occurrence of the sub-string "] ". The following "@1" shifts the input line again one column to the left so that the sub-string disappears. By the expression "&{1|[}” (or "&{1\#[}”) the column "oracle[3065]" is cut off at the point "[" and becomes "oracle". Finally, by the operator "%1", the line is output by one column shifted to the left. The results of the different operations are chained together in the output.

2.4 Encryption And Network Connection

For encrypted data transfer from the agent to the Management Station there is the file with the fixed name "monitorkeys.sig". It contains a signature that is composed of 32 bytes with the ranges of values [0..ff]. Thus, the secret keys can be generated at run time, which are the basis for the dynamic encryption.

The file is first searched for in the same directory as the agent program, then in the user's home directory, under which rights to run the programs, and then in the directory "/etc" (Windows: “C:\”). In the latter case the scope of the file extends over the entire server, if it does not exist in the other places. On the opposite side of the communication the same file with the same signature must exist and be valid. Without the file, an agent cannot be started

The signature of the file is determined at the beginning of the operation and not changed in the following. The agents are distributed along with this file. Two distributed systems with different signatures are not compatible.

In the same file you can optionally and at a central point specify the information for the network connection to the Management Station, which then applies to all agents and should not to be changed.

There are the keywords:

- destination:: <management station> [<alternate management station>]
- portno:: <port number>

One can use a DNS name or an IP address.

Example:

```
# signature for encryption in hex digits separated by ‘:’
07:0a:f7:e1:05:21:60:a4:b3:c9:03:28:1a:9c:25:d2:1e:f0:11:06:f5:7a:b3:c9:1b:03:01:04:a5:09:a1:b7
# network connection to the Management Station
destination::192.168.20.10
portno::55555
```

Note: The entries for the network connection may be overwritten by settings directly at the agent. The comment character '#' must be in this file only at the beginning of the line.

2.5 Options

All agents can be invoked with the following options:

- -d: Address or name of the Management Station
- -e: ditto secondary (alternate) Management Station
- -p: Port number (TCP)
- -x: Definitely start as background process
- -f: Do not fork() from the calling shell
- -t: Test mode

The information through options override any entries in the configuration files and are not changeable.

2.6 Keywords

There are in the configuration files, the following common keywords:

- destination:: <management station> [<alternate management station>]
- portno::<port number>
- attribute::<group> [<object>]
- pollingsecs::<seconds>
- suppressionsecs::<seconds>
- nodename::<alias>
- disabled::
- charset::UTF-8|ISO-8859-1..ISO-8859-10,ISO-8859-13..ISO-8859-16; if the entry is missing, the text is parsed and either UTF-8 or ISO-8859-1 (Latin1) is used (only for Unix)

The two-time colon is part of the keyword. This is followed by the arguments as positional parameters, separated by spaces, if there are several.

For destination:: you can specify the DNS name or IP address of the Management Station and optionally the alternate station, both ipv4 and ipv6. For two data they must be separated by at least one space. The specification portno:: defines the port number (tcp). The entries for the network connection can also be made in the file "monitorkeys.sig" (see above). Then they apply to all agents in the scope.

The name for <group> and <object> must each form a contiguous string, separated by at least one space from each other, and can be up to thirty characters long. The keyword disabled:: inactivates the agent for this configuration file. The keyword pollingsecs:: causes the agent program runs after starting a back-

ground process that performs its functions periodically with the specified number of seconds as the polling interval. For this mode, the agent is to be set in a rc startup routine. If the keyword pollingsecs:: is missing, the agent terminates after each call (exception: “asyncmonagent”, see below).

The keyword suppressionsecs:: specifies the period in seconds in which the repetition of the same identical, **outgoing** messages will be suppressed. Within this period, only different messages are sent to the Management Station, whose significance may be affected by a transformation with a format string. The default setting for this value is zero.

With the keyword nodename:: you can define another name (“Alias”) as the network name agreed in the operating system.

The keywords listed above must be on one line with their arguments and may occur only once.

In addition, there are keywords that must be unique as well, but their arguments may occur in several successive lines. Furthermore, there are keywords that may appear more than once and their arguments can be in several successive lines. The following lines terminate when a row with a keyword occurs or a blank line or a line with a comment or the end of configuration file is reached. The marking of a line as a continuation line is not necessary.

Unique keywords with following lines: filesystem::, fstypes::, procs::, procs_ef::, drives::, system::, application::, security::, tasks::, services::, logfiledir::, files::, filter::

Repeated keywords with following lines: cmd::, syslog::, logfile::, incremental::, total::, restart::

In the following, the keywords will be explained in connection with the various agents.

3. Program “basemonagent“

The program implements the standard monitor for Unix.

It is determined:

- Utilization of file systems (extent of utilization)
- Utilization inodes of file systems (extent of utilization)
- Existence of processes in the process list
- Listenports
- Restart of processes
- Occur zombie processes
- CPU-Utilization and Memory
- Swap + Memory
- Load average and occurrence of reboots
- Analysis of syslog files and other log files
- Evaluating the return of monitoring scripts
- Lifecheck (Heartbeat)

1) Program call

basemonagent [-c <configfilename>] [...]

If the option "-c" is missing, then the configuration file is named "basemonagent.conf" and is located in the same directory. Also specified with "-c" the file must be in the same directory as the program file.

The option "-t" switches to test mode. Messages appear on *stdout*, do not go over the net.

2) Configuration File

There are the keywords:

- filesystem:: List of file system names, thresholds and severities
- fstypes:: List of file system types that are to be monitored
- procs:: List of processes according to "ps -e"
- procs_ef:: List of processes according to "ps -ef"
- restart:: Automatic start of a background process
- defuncts:: two threshold values for Severity and number of zombies
- listenports:: List of port numbers (tcp) that should be in the state of "listening" (ipv4), runs in parallel

- listenports6:: ditto for ipv6
- cpu:: two percentages and Severity of exceeding cpu load
- load:: two threshold values for Severity and load average 15 minutes, further signaling to reboot
- swap:: two percentages and Severity of swap and memory allocation
- syslog:: Analysis of syslog log files and other files
- cmd:: Parallel (concurrent) execution of monitoring scripts and filtering of the output, see also "scriptmonagent"
- lifecheck:: Dynamic registration at the Management Station

Threshold Inodes:

The threshold for the utilization rate of the number of inodes is constant at 95%.

File System Monitoring:

The agreement for the file system monitor has the following form:

Form 1:

filesystem::[<mins>::]-d <percent> [-D <fs1>[[<percent>[,<sev>]]] ...] [-E <fse1> <fse2> ... <fseN>]

Form 2:

filesystem::[<mins>::]-D <fs1>[[<percent_1>[,<sev>]]] ... <fsN>[[<percent_N>[,<sev>]]]

fstypes::fstype_1 fstype_2 ... fstype_N

If the line with fstypes:: is missing, **each** mounted file system is detected. If the line is present, only the file systems are detected, their types are included in the list. The number <mins> has the meaning of time in minutes for the repetition of the message when exceeding the threshold. In between, there is only one message if the change is greater than 1 MB. After the "-d" option follows a number with the meaning <percent> utilization rate in % for **all** detected file systems. After the "-D" option there is a list of file system names that need to be mounted and can have specific thresholds <percent>. After the "-E" option there is a **list** of file system names that should be excluded from monitoring. The list elements are separated by at least one space from another. If the option "-d" is missing, only the file systems specified in "-D" are monitored. File system names can be specified as search pattern with the special characters '*', '?' and [...].
If not specified for <sev>, the default value is "minor". If the utilization rate for a file system is greater than 98%, the Severity is changed to "critical". This be-

havior can be changed by the filters on the Management Station when needed. There you can bind the threshold and/or the Severity of the value of the remaining available memory, which is included in each message.

The shortest form of the file system monitoring is:

filesystem::-d 93 # first threshold for all existing file systems 93%

A file system name can be enclosed in quotes if it contains spaces in them. Example: -D "Volume 1[95 maj]"

The percentage value is the utilization rate (allocated/total). Of 93% seven percent are still free respectively available.

Process Monitoring:

The agreement for process monitoring has the form:

procs::procname-1[OPTIONS] procname-2[OPTIONS] ... procname-N[OPTIONS]
procs_ef::procname-1[OPTIONS] procname-2[OPTIONS] ... procname-N[OPTIONS]

After entering the keyword a list of process names plus possibly OPTIONS is being done. The list elements are separated by at least one space. In the keyword procs:: the process name must exactly match the output of "ps -e" (MacOS: ps -acx). In the keyword procs_ef:: the process name is a search pattern (regular expression) for the output of "ps -ef". The specified process names including the options in [...] can be put in quotation marks if the name contains blanks (Example: procs_ef::"httpd -D FOREGROUND[1-5]").

Form 1:

procs::procname[[<sev>]] ...

The processes specified must exist in the process list at least once. The default value for <sev> is "critical".

Form 2:

procs::procname[COUNT[,<sev>]] …

The number of instances of the processes must be exactly COUNT. When COUNT is equal to zero there is signaled when the process is active ("should not be running")!

Form 3:

procs::procname[MIN-MAX[,<sev>]] ...

Where MAX > MIN. Number of instances of a process in the range of MIN to MAX. For MIN == 0 and MAX > 0 the number of instances may not be greater than MAX, but can also be zero.

Form 4:

procs::procname[+MAX[,<sev>]] ...

Number of instances may not be greater than MAX, but must be at least one.

Form 5:

procs::procname[-MIN[,<sev>]] ...

Number of instances must not be less than MIN.

The same rules also apply to the keyword procs_ef::. It should be noted that is signaled continuously in the event of an error until the fault is corrected.

Process Monitoring with Restart:

The declaration has the form:

restart::[COUNT::]procname::startprocedure

The facility offers the opportunity to start a background process again immediately after the detection of its failure. It is all done automatically, so that downtime is minimized. There may be more than one line of this type in a configuration file.

The process name is a pattern (*regular expression*) for the output of "ps -ef". This is followed by the name of the start procedure with full path, which starts the required process (again), when it is inactive. The passing of parameters to the procedure is permitted. The procedure must terminate within 5 seconds otherwise there is a timeout error. COUNT is a number representing the maximum number of unsuccessful attempts. If the number is exceeded, there are no further attempts to start the process, but a signaling. The default value for COUNT is 5. The starting procedure must terminate with an exit code of zero otherwise there is a message.

If there is more than one process, the processing of the corresponding lines takes place in the order from top to bottom. The failure of the process and call the start procedure is indicated with a critical message to the Management Station. The operation is not complete until the next polling interval the proper running of the process is determined. Then there is an informative message about the duration of the outage. Otherwise, the operation is repeated until COUNT is exceeded.

If special privileges are needed for starting the program basemonagent must operated for this purpose with root privileges. Also possible is the operation of an instance basemonagent only for this purpose that runs with root privileges.

Example:

restart::3::^/usr/sbin/snmpd$::/usr/sbin/snmpd

Process "snmpd" is started or restarted if it is not active. The agent must be running with root privileges.

Log File Analysis:

The declaration has the following form:

syslog::[<name>::][nMB-Size[<sev>]::][!-]<full_pathname_logfile>::filter_1 filter_2 filter_n

After the keyword syslog:: optionally follows a logical name (label) and optionally a number with the threshold for the size of the target file in number of megabytes (MB). The logical name can be up to 30 characters long and must begin with a letter. Then comes the full path name of the syslog file (or any other log file). Then follows the list of filters, whose elements are separated from each other by at least one space. Before the file name can optionally the special characters '-' and/or are '!'. The character '!' causes no message comes when the file does not exist. The character ‘-‘ specifies the starting condition at the beginning of monitoring. Without this option, the agent will send at the first run messages on the frequency of pattern matching. With the option it does not occur. It is only determined the initial value for the incremental monitoring. Thereafter, the normal operation is the evaluation of lines, which have been added between two calls. If the destination file is truncated or emptied, there is a new initial run with information on the frequency of entries found.

The name of the destination file may be in its base name include wildcards '*', '?' "[...]". All files in the directory whose names match this pattern are then monitored with a common list of filters (example: /var/log/*.log).

3) Examples:

Example 1:

```
destination::NEPTUN PLUTO
portno::55555
attribute::OS Unix
pollingsecs::300
fstypes::ext3 nfs
filesystem::-d 93 -D /[85,maj] /home[90,warn] -E /dev /net
#filesystem::60::-d 93 -D /[85,maj]
#end of configuration file
```

The name of the Management Station is NEPTUNE, PLUTO the alternate station. The transmission port is 55555/tcp. There are two attributes "OS" for "Group" and "Unix" for "Object" in the Browser of the Management Station. The polling interval is five minutes (300 seconds).

The general threshold for the utilization rate of file systems is 93%. When exceeding a minor message is sent. The threshold for the root file system is 85% and there is a major message. The File System "/home" has a threshold of 90% with the Severity "warning". The file systems "/dev" and "/net" are excluded from monitoring.

It will cover all file systems of type "ext3" and "nfs". If not specified fstypes::, all existing file systems are detected.

With the statement "-E /fsname_1 /fsname_2 /fsname_n" you can exclude file systems. For the names of the mount points search patterns for file names are allowed.

Missing the option "-d", only the file systems are monitored, which are listed after the "-D" option.

A number after the keyword filesystem:: is the repetition time in minutes. The number "60" means that after 60 minutes the message is repeated, unless a change has happened before. The default value for this entry is 120.

Note: If the utilization rate of a file system is greater or equal 98%, there is a message with the Severity "critical".

Example 2:

```
destination::192.168.178.21
```

```
portno::55555
attribute::OS Unix
pollingsecs::90
procs::java[-2,crit] webscan ascWrapper[2-5,maj] lpd[+5,warn] ftpd[0,warn]
procs_ef::oracleapp.*LOCAL=yes[+5,min] ora_.*_app[4]
# end of configuration file
```

procs::
The process "java" must exist at least 2 times in the process list, or critical message.
The process "webscan" must be active (no matter how many instances), or critical.
The process "lpd" may be at most 5 times, or warning. The number of instances of "ascWrapper" should be between 2 and 5, otherwise major message. If "ftpd" is running, message will come with the Severity "warning".

procs_ef:: (command ps -ef) The search pattern can be specified as regular expressions. They relate to the program's name and a list of additional options or arguments, as they correspond to the output of the command "ps -ef".

In Example 2 should also be recognized when the process "oracleapp.*LOCAL=YES" exists more than five times in the process list. The message has the Severity "minor". Processes that match the pattern "ora_.*_app" must appear exactly four times in the process list, otherwise there is a major message.

If the required number of instances does not coincide with the actual, the default Severity is "major". If the process does not exist in the process list, the default Severity is "critical".

Example 3:

```
destination::NEPTUN
portno::55555
attribute::OS Unix
procs::inetd snmpd
swap::90 98[crit] # percentage allocation swap:90%->warning,98%->critical
load::20[maj] 50[crit] # run queue respectively load average
cpu::95[maj] 99[crit] 120
# average over 120 minutes; first threshold 95%, second threshold 99%
#cpu::95[maj] 99[crit] # alternative: current load
defuncts::100[maj] 200[crit] # Zombies; first threshold 100, second threshold 200
#defuncts::200[crit] # alternative: just one threshold
lifecheck:: # lifecheck on
#end of configuration file
```

Additionally happens monitoring the performance and Lifecheck is turned on. Lifecheck means that at each call control information is sent to the Management Station. In the absence of this information, an appropriate signaling on the Management Station takes place.

Example 4:

```
destination::NEPTUN
portno::55555
attribute::OS Solaris
lifecheck::
syslog::10[maj]::/var/adm/messages:“insignificant line“-null
“suncluster is down;;Suncluster is down: %4“-crit[3] “cluster event xyz“-min[-3]
“warning/i”-warn “error|fatal/i”-maj
# drift net
#syslog::… more log files
# end of configuration file
```

It is additionally evaluated the syslog file in Solaris (filename: /var/adm/messages). In case of "insignificant line" there is no message; this is suppressed. The entry "suncluster is down" is as critical message with maximum of three times. The found line is shifted by the format string after “;;” by four columns to the left and the text "Sun Cluster is down:" is prefix.

The entry "cluster event xyz" must occur at least four times, so that it appears as a minor message, but then only once. The last line ("warning/i") causes unexpected lines, in which suspicious entries such as "warning", "error" or "fatal" occur, are recognized even when a corresponding event is unknown or never happened.

The entry after syslog::10[maj] causes the size of the syslog file is monitored. In the case over 10 MB, there is a major message.

The list of filters can be arbitrarily long. If a filter hits the list run is finished. So it depends on the order of list elements. Found lines from the syslog file will be sent prefixed with the prompt "SyslogEntry::" to the Management Station, the associated Event Type is "Syslog". When a substitution occurs by a format string, the prompt is "SyslogEntryFmt::", the Event Type is "SyslogFormat".

For the specification of the file name one can use Meta characters '*', '?' (search pattern for file names). There are then all files matching the search pattern monitored. This also applies to files, which are newly created on the fly.

Example 5:

```
destination::NEPTUN
portno::55555
attribute::OS Unix
restart::3::lpsched::startlpsched
listenports::21 22 25 80 111
listenports6::22 80
cmd::netstat -an::"^udp4.*\*\.161 ;;Port 161/udp4 (snmp) unreachable"-crit[<1]
"^udp4.*\*\.123 ;;Port 123/udp4 (ntp) unreachable"-maj[<1]
# end of configuration file
```

By Keyword restart::, followed by the name of a process in accordance with "ps -ef", the existence of the process in the process list checked. If the specified process is not active, the restart procedure "startlpsched" is called. For the specification of the process name, you can use regular expressions. The operation is repeated not more than three times when the restart was unsuccessful.

If several process to be started or restarted, there may be more than one line of this type.

The keyword listenports:: with the following port numbers causes examination of the ports tcp4 against the loopback interface. This keyword listenports6:: applies to ipv6.

With the keyword cmd:: you declared a program/script, whose output is evaluated. In the example it is the netstat command with which listening UDP ports are checked. Note the negative filters and format strings for evaluation.

4) Scheduling

If the keyword pollingsecs:: is missing in the configuration file, you can call the program periodically by crontab.

Example: 1,21,41 * * * * /home/monitor/basemonagent >> /tmp/basemonagent.log 2>&1

If one runs the agent as a background process, a startup script to start when booting the operating system is needed. The easiest way is to use the start script "rc.local", which is available on many Unix derivatives under "/etc". In it you can enter a line type:

```
su - monitor -c /home/monitor/basemonagent
logger “Agent for standard monitoring has started“
```

When booting the agent program is "basemonagent" is started, that belongs to the user "monitor". It is indicated on the Management Station.

The agent will be terminated by the signal SIGTERM or SIGINT, which also causes a message to the Management Station.

The minimum value for pollingsecs:: is 60 (60 seconds, 1 minute).

Important: After changing the configuration file, whether by an editor or from the Management Station, the agent reads the new configuration automatically. One need not stop the agent and then start again.

4. Program “logmonagent“

Agent program for log file analysis. With one configuration file you can evaluate several log files. The program can be provided with different configuration files.

1) Program call

logmonagent [-c <configdatei>] [...]

If no configuration file is specified with the option "-c", the system uses the default name "logmonagent.conf". If this is not present or cannot be read, an error message is output to *stdout*. Program and configuration file must be in the same directory.

Found lines of a log file are sent with prefixed "LogfileEntry" respectively "LogfileEntyFmt", if a format string is specified, to the Management Station. The message text consists of the contents of the found line, to which the name of the log file is appended in brackets. This is the last column of each row found. This can additionally determine the message source. For the collection and transmission of a line 1024 characters are available.

2) Configurations File

There are the keywords:

- logfile:: Mode "growth since the last call to evaluate"
- incremental:: identical to logfile::
- total:: Mode "Evaluate entire file if it has changed"

The lines for the log files have the general form:

logfile::[<name>::][nMB-Size[<sev>]::][!-]<full_pathname_logfile>::filter_1 filter_2 filter_n

or

total::[<name>::][nMB-Size[<sev>]::][!-]<full_pathname_logfile>::filter_1 filter_2 filter_n

After the keyword logfile:: or incremental:: follows optionally a logical name (label) and optionally a number as a threshold for the size of the target file in megabytes (MB). The logical name can be up to 30 characters long and must begin with a letter. Then comes the full path name of the log file. A preceding,

optional exclamation mark '!' causes, that it is not reported if the target file should not exist. Then follows the list of filters.

Regarding the incremental log file analysis the character '-' immediately before the file name defines the start condition. It suppresses signaling at the first call of the function. Without this option, in the first run, the number of lines that match a search pattern appears. Then happens normal evaluation of newly added lines.

With the key word total:: found lines are indicated without the character '-' at the first run, all the same how old these entries are. With the option of '-' found lines are indicated only when **after** the first run the file has changed or has renewed.

The file names can contain in both cases in their base names search pattern for file names ('*', '?', "[...]"). Then all files in the directory, which correspond to the pattern, are monitored. It is important to note that the file names that have been determined by a search pattern, and file names without the search pattern are managed separately. The name of a log file or a search pattern for it per mode of evaluation (incremental, total) must not appear more than once in a configuration file! But it is possible to use two different patterns for the same file.

With the next list of the filters the real, content evaluation of the protocol file takes place. The list will run through sequential; if the respective searching pattern with a line agrees, this is transferred and shown as an event text and the run is ended. It depends therefore on the order of the filters. The list run also ends if it concerns a suppression filter.

Notice: The search pattern “*” of the filter has the special meaning "anything".

3) Examples:

Example 1:

It is the online log file of DBMS Informix to be monitored:

```
destination::NEPTUN
portno::55555
attribute::Informix
pollingsecs::20
logfile::/home/informix/online.log::"transaction aborted"-warn
"failed transaction"-warn "cannot accept a connection"-maj
"ABORTED"-maj "Archive completed"-null "Archive on.*Completed"-null
"Archive on"-maj "Backup.*(Started|Completed|completed)"-null
"Logical [Ll]og.*Backup"-maj #  backup not successful, order of filters!
"Assert Failed"-maj "Lock table overflow"-maj
"log files are almost full"-maj
"no more extents"-crit # lack of resources
"(password is not correct|Incorrect password)"-warn
"Read error"-warn "25580.*System error"-null
"System error"-warn "[Oo]verflow"-warn
"error|errstr/i"-maj "inconsistencies/i"-maj
# more log files
# logfile::...
# total::...
# end of configuration file
```

The minimum value for pollingsecs:: is 2 (2 seconds).

The name of the log file is "/home/informix/online.log", it should be evaluated incrementally. The Management Station is NEPTUNE. The transfer port is 55555. There is only one attribute - "Informix" for group - specified. As the "Object" the base name of the log file will appear in the Event Browser.

Example 2:

It is the log file "/oracle/PRX/saptrace/background/alert_PRX.log" to be monitored for the application PROJECTX.

```
destination::192.168.178.20
portno::55555
attribute::PROJECTX Oracle/Sap
pollingsecs::15
logfile::512[crit]::/oracle/PRX/saptrace/background/alert_PRX.log::
“Errors in file.*background.*smon_28706.trc“-min
“ORA-01575.*timeout waiting“-min “ORA-01595.*error freeing extent“-min
“ORA-1552“-min “ORA-1652“-min “ORA-1661.*PSAPTEMP“-min
“ORA-1631.*(TST03|MCSI|SNAP)“-min
“ORA-1631“-crit # vb error
“ORA-16014”-crit “ORA-165[34]”-crit
“ORA-“-maj # unknown entries with ORA-; Severity major
“error|fatal||fail/i“-maj # drift net for ERROR, …
“[sS]hutting down instance;;Instance is shut down: $*“-warn
# after “;;” format statement (format string)
# end of configuration file
```

The entry after the keyword logfile::512[crit] causes the (growing) size of the log file is monitored. In excess of 512 MB there is a critical message.

Note: If the log file does not exist or is not readable, a system error message is sent to the Management Station. This can be suppressed by the file name prefixing the character '!'.

Example 3:

```
destination::192.168.178.20
portno::55555
attribute::Security
pollingsecs::30
total::!/etc/hosts.equiv::“*;;File not allowed here“-crit[1]
“!*;;File not allowed here (file empty)“-maj
#end of configuration file
```

A message appears if the file "/etc/hosts.equiv" is created or already exists. The exclamation mark in front of the file name ensures that there is no message if the file does not exist.

5. Program “logdiragent“

Incremental analysis of log files in directories. The program evaluates the growth of files between two calls. You can specify multiple directories and a list of search strings for file names. The search pattern for file names must conform to fnmatch().

1) Program call

logdiragent [-c <configfilename>] [-n] [...]

The default name for the configuration file is: logdiragent.conf. The configuration file must be in the same directory as the program. With the option "-c" <configfilename> you can choose a different name. The option "-n" causes the agent signaled even if between two calls is not the size but the date for the target file has renewed itself.

Found lines of a log file are prefixed with "LogfileEntryDir" sent to the display. The message text consists of the contents of the found line, to which the name of the log file is appended in brackets. This can additionally determine the message source.

The minimum value for pollingsecs:: is 10 (10 seconds).

2) Configuration File

There are the keywords:

files::[<nMB-Size>[severity]::]filespec_1 filespec_2 .. filespec_n
logfiledir::logfiledir_1 logfiledir_2 .. logfiledir_n
filter::filter_1 filter_2 .. filter_n

3) Examples:

Example 1:

```
portno::55555
destination::192.168.178.20
attribute::OS Logfiles
pollingsecs::20
# if the second column is missing, base name of log files as “object“ is taken
files::10[maj]::*.log *.error # specification of file name according to pattern
logfiledir::/var/adm/mail /var/adm/syslog # specification of directories
filter::”warning/i”-warn[3] ”error/i”-maj “fatal/i”-crit
# list of filters for all files
#end of configuration file
```

The directories "/var/adm/mail" and "/var/adm/syslog" are evaluated. All files which correspond to the pattern "*.log" and "*.error" are taken. If one of the files is larger than 10 MB, there is a message with the Severity "major".

Note: If a log file to be grasped is not to be opened reading, a system error message occurs at the Management Station.

Example 2:

There are the trace files to be monitored in the work directory of SAP/R3.

```
portno::55555
attribute::SAP/R3
destination::NEPTUN
pollingsecs::15
files::4[crit]::dev_w? dev_w?? dev_rfc? dev_rfc?? dev_disp dev_?? stderr*
# specification of in directory; pattern according to fnmatch()
logfiledir::/usr/sap/PC3/DVEBMGS11/work  # work directory with log files
 filter::"update deactivated"-crit "vb error"-crit  # vb error
"^Disconnecterror:"-crit "^Sqlerror:"-warn "^Sevdberror:"-maj
"^Profileerror:"-warn "Sharedmemoryerror"-maj "^Stackerror:"-warn
"^Mallocerror:"-maj "^Applicationerror:"-maj "^Inputbuffererror"-maj
"^Speichermangel:"-warn "^Shared Memory"-warn
"update activated"-info "Error Code"-min # job termination
"error in ABAP statement"-maj
"Memory exhausted:"-maj
"Error.*in.*application.*program"-min
"CPIC-Error"-null “ERROR.*shmctl“-maj “ERROR.*shmget”-maj
“error/i”-warn[-1] “warning/i”-min[-1]
“fatal/i”-maj[3] # drift net for “FATAL”, “fatal“, ...
# end of configuration file
```

The list of filters applies to all log files to be detected.

6. Program “logrecagent“

Incremental analyses of log files in directories and subdirectories. The program evaluates the growth of files between two calls. You can enter multiple directories and multiple search patterns for file names. It also considers subdirectories of the specified directories.

1) Program call

logrecagent [-c <configfilename>] [-n] [...]

The default name for the configuration file is: logrecagent.conf. The configuration file must be in the same directory as the program. With the option "-c" <configfilename> you can choose a different name. The option "-n” causes the agent signaled even if between two calls is not the size but the date for the target file has renewed itself.

Found lines of a log file are prefixed with "LogfileEntryDirRec" sent to the display. The message text consists of the contents of the found line where the log file name in square brackets is appended. This can additionally determine the message source.

The minimum value for pollingsecs:: is 30 (30 seconds).

2) Configuration File

There are the keywords:

files::[<nMB-Size>[severity]::]filespec_1 filespec_2 .. filespec_n
logfiledir::logfiledir_1 logfiledir_2 .. logfiledir_n
filter:: filter_1 filter_2 .. filter_n

3) Example:

```
portno::55555
destination::X8PSR Y8PSS # Y8PSS optional alternate server
attribute::OS Logfiles # if the second column is missing, base name of log files as “Object“ is taken
files::100[crit]::*.log *.error # specification of file names
logfiledir::/var/adm # specification of the directories
filter::”warning/i”-warn[3] ”error/i”-maj
“fatal/i”-crit # list of filters for all files
```

Evaluates the directory "/var/adm" with the underlying subdirectories. Be taken all the files, which match the pattern “*.log” and “*.error”. If one of the files is larger than 100 MB, there is a message with Severity "critical".

Note: If a log file to be grasped is not to be opened reading, a system error message occurs at the Management Station.

7. Program “scriptmonagent“

The agent program calls executable programs, filtering its text output and sends the result to the Management Station when needed. With one configuration file you can define multiple monitoring programs. The program can be parameterized with different configuration files. It can be run as a batch program or as a background process.

A program for monitoring is carried out internally by the system call *fork()* followed by the function *execvp()*. That is the normal cases. However, if the declaration of the program contains special *Shell* characters such as '|' or ';', *execl()* is used in conjunction with "/bin/sh -c". In this way, a further processing of the output with the pipe '|' and maybe *awk* or *perl* is possible. This rule also applies to the agent basemonagent and asyncmonagent!

The programs defined in the configuration file are called periodically; each program is executed in parallel in its own *thread*. There is an adjustable timeout for each program. If the limit is exceeded, the program is stopped with SIGTERM and a critical message is sent to the Management Station. **IMPORTANT:** The output takes place immediately after the end of the program but not during the program runtime. A program in a *thread* can only be called again when the previous program run has ended. The runtime can be longer than the polling interval.

1) Program call

scriptmonagent [-c <configdatei>] [...]

If no configuration file is specified with the option "-c", the system uses the default name "scriptmonagent.conf". Program and configuration file must be in the same directory!

The minimum value for pollingsecs:: is 2 (two seconds).

The Path variable "PATH" is set to:

PATH="/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:<agent-path>:<agent-path>/bin

2) Configuration File

There's the key word:

- cmd:: Declaration of an executable program with name and filters for the output

The lines for the scripts have the form:

cmd::[<name>::][<tos>::]<program name>::filter_1 filter_2 filter_n-1 ... filter_n

tos: Timeout in seconds, default: 30 (> 0)
name: Logical name (label)

The program must have an output to *stdout* and/or *stderr*. It should be noted that a call to the monitoring program might generate more than one message when the output has more than one line, each of which corresponds to the pattern of the filter.

The analysis is done through the filter list whose elements are separated by at least one space, and if necessary by additional commands ("grep", "awk") and command chaining with the pipe '|'. For more complex analyzes, such as for database management systems such as Oracle or Informix, the program should be outsourced to a separate shell script whose name is then declared.

From the command return filtered lines without format specification will be sent prefixed with "ScriptMonitor::" to the Management Station; with format specification the prompt is "ScriptMonitorFmt::".

3) Example:

It is the memory usage of a Linux server to be monitored by the analysis of a command. To facilitate the numerical operations the report generator "awk" is taken. In the awk program two thresholds values for the utilization rate are included serving two filters with different Severity in the filter list.

```
Destination::192.168.178.20
portno::55555
attribute::Linux Performance-Extra
pollingsecs::120

# Analysis of listening sockets
cmd::NETSTAT-TULPN::netstat -tulpn::":[0-9]{1,5}[ ]/v"-null
":80[ ];;Port 80 not listening"-crit[<1] # ...
":(22|25|80|443)[ ]/v;;Unknown port: $4/$1 PID: $$"-maj

# evaluate command
cmd::free -m | awk 'BEGIN {lim1=90;lim2=98} {if(NR == 2){
val = ($3*100)/$2; if(val >= lim2){
printf("Memory-Utilization2 (Threshold: %d%): %d% total: %d MB, free %d MB",lim2,val,$2,$4)
} else if( val >= lim1 ){
printf("Memory-Utilization1 (Threshold: %d%): %d% total: %d MB, free %d MB",lim1,val,$2,$4)
}}}'::"^Memory-Utilization1"-min "^Memory-Utilization2"-maj "*;;System error: $*"-crit[1]

cmd::PING-GOOGLE::ping -c 3 www.google.com::" time=[4-9][0-9]{1,}[.,];;TIME: %<time=>$*"-warn[1]
"([1-9][0-9]*|100)% packet loss"-maj
# more lines with programs and filters
# cmd::..…
# end of configuration file
```

The transmission port is 55555. There are the attributes "Linux" for "Group" and "Performance-Extra" for "Object" in the Event Browser. If the second attribute for "Object" is omitted, the base name of the specific program name is taken.

Note: The monitoring program must be executable and provide a return code of zero. Otherwise, there is a separate system error message to the Management Station, indicating the return code.

8. Program “asyncmonagent”

The program is an extension of the current agents basemonagent, scriptmonagent and logmonagent. Each script and each log file that is declared in the configuration file can be provided with its own polling interval or can be defined as a cron job. The execution of the individual monitoring functions is done concurrently in the form of threads. The agent is operated solely as a background process!

The periodic execution of a function is designed so that a call in one thread can take place only if the previous has been completed, regardless of the other threads. There is no danger of overlapping calls, which occurs when scripts modify their run time or even block ("duplicate cron jobs"). In contrast to the other agents, there is **no default timeout** for the execution of scripts and programs. This way you can also use special monitoring programs which are either not at all or only terminate after an indeterminate time (e.g. journalctl -f, tcpdump, nmap, snmpdelta, mpstat, sar). Their output is sent immediately after the filtering in real time to the Management Station.

This behavior can be changed with an option (see below), so that the result is only delivered after the monitoring program has been processed and all data are available. In this case, the total number of entries found per filter is also transferred. The entire running time is limited by a timeout. If the limit is exceeded, the function is aborted and an error message is displayed.

1) Program call

asyncmonagent [-c <configdatei>] […]

If no configuration file is specified with the option "-c", the system uses the default name "asyncmonagent.conf". Program and configuration file must be in the same directory!

2) Configuration File

There are the key words:

- cmd:: Declaration of an executable program with name and filters for the text output
- logfile:: Mode “growth since the last call to evaluate“
- incremental:: identical to logfile::
- total:: Mode “evaluate entire file if it has changed“
- lifecheck:: Dynamic registration at the Management Station

Line with lifecheck::

lifecheck::[<number of seconds>]
The minimum value for <number of seconds> is 30. If the specification for <number of seconds> is missing, the default is 300. It should be noted that the use of lifecheck:: of this agent and the agent basemonagent is mutually exclusive; it can be used only one or the other at the same time using the same computer name. This can be avoided by placing an alias name for the server with the keyword nodename::.

8.1 Individual Polling Intervals

The lines for the scripts have the form:

cmd::[<name>::]<polling interval>::[%]<program name>::filter_1 filter_2 filter_n-1 ... filter_n

The option with the special character '%' in front of <program name> means that there is a timeout of 30 seconds. Without this addition there is no timeout.

The lines fort the log files have the form:

logfile::[<name>::]<polling interval>[/<mbsize>[<sev>]]::[!-]<full_pathname_logfile>::filter_1 ... filter_n
total::[<name>::]<polling interval>[/<mbsize>[<sev>]]::[!-]<full_pathname_logfile>::filter_1 ... filter_n

The optional statement <name> indicates the logical name of the instance if you want to evaluate one and the same log file more than once with different parameters. The name can be up to 30 characters, must begin with a letter and must be unique.

The value for <polling interval> can be specified in the following form:

- A number as the number of seconds, e.g., “90” (90 seconds)
- <minutes>.<seconds>: Number of minutes and number seconds separated by a dot ‘.’, e.g. “10.0” (10 minutes plus zero seconds)
- Number of hours, number of minutes, number of seconds separated by dots, e.g. “1.30.0” (1 hour + 30 minutes + zero seconds)

The minimum value for <polling interval> is 2 seconds.

8.2 Running as a Cron Job (Scheduler)

This allows functions at discrete times call.

The lines for the scripts or commands have the form:

cmd::[<name>::]<crontabspec>::[%]<program name>::filter_1 filter_2 filter_n-1 ... filter_n

The option with the special character '%' in front of <program name> means that there is a timeout of 30 seconds. Without this addition there is no timeout.

The lines for the log file analysis have the form:

logfile::[<name>::]<crontabspec>::[!-]<full_pathname_logfile>::filter_1 ... filter_n
total::[<name>::]<crontabspec>::[!-]<full_pathname_logfile>::filter_1 ... filter_n

The data for <crontabspec> is based on the facility crontab() of Unix. For the times a cron job, there are the following characters:

- A number for the minute [0-59], hour [0-23], day of month [1-31], month [1-12], day of week [0-7]; 0 or 7 is Sun; symbolic names are not allowed
- The asterisk ‘*’ stands for every minute, every hour, every day, every month and every day of the week
- Commas ‘,’ for a list of times
- Hyphen ‘-‘ for a range
- Slash ‘/’ for a period
- Blank ‘ ‘ as delimiter for the time fields

Examples for <crontabspec>:

“* * * * *”: Call every minute
“*/5 * * * *”: Call every 5 minutes
“15,45 10,11,12 * * *”: Call at 10, 11, 12 clock 15 and 45
“*/10 6-18 * * 1-5”: Call every 10 minutes 6-19 clock from Mo to Fr

For programs that do not terminate (daemons), please note the following: The agent starts the program at the end of the first polling interval (e.g. 10 seconds). The Polling Interval has the function to set the filters and their associated counter after each time period back in this case. In a cron job, this value is the constant 60 (1 minute). When changing the configuration during operation the daemon is stopped with SIGTERM and then started again.

Example:

```
cmd::TCPDUMP::10::tcpdump -t -l -q::“ICMP echo request“-min # ...
# will start after 10 seconds and runs infinitely
cmd::JOURNALCTL-CRIT::10::journalctl -f -q -p crit::”*”-crit
# runs infinitely
cmd::SNMPTRAP::5::snmptrapd -f -Lo -Oq::“[ ]\[UDP:[ ]\n*“-warn[0]
# direct evaluation of snmptrapd
cmd::MPSTAT::3::mpstat -u -P ALL 3600 1::“^[A-Za-z]+:[ ]/v“-null “CPU|all“-null
“[ ][0-4]([.,][0-9]+)?$;;High average cpu usage, cpu-no: $2, %2“-warn
# runs one hour (3600 s) and then starts again
```

The program mpstat will start after 3 seconds, runs for an hour and starts after 3 seconds again, etc.

Otherwise, the description for the agent logmonagent and scriptmonagent applies.

Note: If there is no timeout for monitoring programs, the quantitative evaluation of found items is done in a different way. If for an analysis with the corresponding setting, the number of occurrences found is greater than that declared in the filter, there is a separate message with the Event Type "Frequency" with otherwise identical attributes. The text of the message displays the search pattern of the filter and the number of lines found. Specifying the quantifier of a filter can influence this mechanism. One can choose a high number or the value "0" (e.g. “[ ]error[ ]/i“-warn[0]). The value "0" disables the mechanism.

These considerations do not apply if you choose the option '%' in front of the program name for programs with a short run time. In this case, the output only occurs after the program has been completed, and the total number of entries found per filter is transmitted with each message in binary form.

More Examples:

```
destination::192.168.178.20
portno::55555
attribute::Linux AsyncMonitorCollection
lifecheck::120
# will send every 120 seconds a heartbeat signal to the Management Station

suppressionsecs::300
# will suppress the repetition of identical messages within 300 seconds

# polling interval 5 seconds
logfile::5::/var/log/apache2/access.log::"\{.*;.*\};;SHELLSHOCK?: $*"-maj
"(sh[ ]+\-c)|(/bin/bash)|(/bin/sh);;Why bash?: $*"-warn

# polling interval 20 seconds; if larger than 5 MB there is a minor message
logfile::20/5[min]::/var/log/daemon.log::"error/I;;%4"-warn # …

# polling interval 10 seconds (it will be sent a maximum of 3 lines found)
logfile::10::/var/log/messages::"POSSIBLE BREAK-IN ATTEMPT;;%5"-maj[+3]

# polling interval 5 minutes (one message when more than 100 in 5 minutes)
logfile::ssh-login::5.0::/var/log/messages::"sshd.*[Ii]nvalid user"-maj[>100]
"Failed keyboard-interactive"-maj[>100]

# polling interval 1 hour; evaluate entire file, if it has changed
total::1.0.0::!/var/log/boot.log::"error/i"-warn # …

# analysis of current network connections (tcp/udp, Linux),  polling interval 4 seconds
cmd::NETSTAT-TUNP::4::%netstat -tunp::":[0-9]{1,5}[ ]+(127.0.0\.1|\::1):"-null
":[0-9]{1,5}[ ]+192\.168\."-null # suppress your own addresses
":[0-9]{1,5}[ ].*:[0-9]{1,5}[ ];;Unknown foreign addr: $5/$1 PID: $$"-warn[8] # show unknown

# CRONJOBS
cmd::0 10 * * 1-5::echo "It is ten o'clock a.m. (mo-fr): `date`"::"*"-info
# check disk space
cmd::15,45 6-20 * * *::%df -k::"[ ]100%[ ];;FS full: $*"-crit "[ ]9[5-9]%[ ]"-maj
"[ ]9[0-4]%[ ]"-warn "[ ]8[7-9]%[ ]"-min
# … more functions
```

9. Program “secmonagent”

Agent for the purpose of security. The program monitors system files and/or system directories and other directories with the files contained in them on the change of its attributes. There will also be detected and reported if files have been added to a directory! The examined attributes are: *inode number*, *size*, *modification time*, *mode/permission*, *status time*, *user id of owner*, *group id of owner*.

1) Program call

secmonagent [-c <configdatei>] [...]

If no configuration file is specified with the option "-c", the system uses the default name "secmonagent.conf". Program and configuration file must be in the same directory! You can provide the program with different configuration files. It can be operated as a batch program or as a background process (daemon).

2) Configuration File

There are the key words:

- files:: A list of directories and files with full path name that can be optionally provided with a Severity (default: "warning")
- exclude:: A list of files and/or subdirectories with full path name that should be excluded from the monitoring

The names can be enclosed in quotes ‘”’ if there are spaces in them. If after the name of a directory a slash '/' occurs, only the contents of the directory is monitored, but not changes to the directory itself.

Using the agent itself installations can be observed. This applies to intentional as for unintended changes (e.g. ***rootkits***). If desired installations are made, you can make comparisons between announced and actual changes. If the program has detected one or more amendments, there is a message per directory with file names and kind of change.

Example 1:

```
destination::192.168.178.20
portno::55555
attribute::Debian Security
pollingsecs::30 # run as daemon, evaluation every 30 seconds

files::/bin[crit] /sbin[crit] /usr/bin[maj] /usr/sbin[maj] /etc/init.d[crit] /etc/[maj]
/usr/lib /boot[crit] /boot/grub[crit] /lib/modules[maj] /lib # ...
exclude::/etc/mtab /etc/resolv /etc/adjtime # ...
```

There are the directories listed checked for changes. The files "/etc/mtab", "/etc/resolv" and "/etc/adjtime" are excluded from monitoring.

The minimum value for pollingsecs:: is 5 (5 seconds)! The notifications from the agent have the Event Type "Security".

Example 2: Different Severities for files in the same directory

```
destination::192.168.178.20
portno::55555
attribute::Debian Security
pollingsecs::30

files::/bin/sh[crit] /bin/ls[crit] /bin/ps[crit] /bin/netstat[crit] /bin[maj] # ...
```

In the case of a change the files “/bin/sh”, “/bin/ls”, “/bin/ps” and “/bin/netstat” generate a message with severity "critical" because they stand **in front** of the directory “/bin” and are not directories. All other files in the directory "/bin" generate a message with severity "major" on change.

10. Program “rsendmsg“

The program sends messages directly to the Management Station. There you can filter or transpose them as needed. Used in scripts you can implement any monitoring functions.

Program call:

rsendmsg [-p <port>] [-d <server>] [-e <alternate server>] -g <group> -o <object> [-n <node name>] [-c <charset>] [-6] -s <severity> -m <message text>

Parameter:

- -p: Port number tcp; optional when entry in “monitorkeys.sig”
- -d: Management Station; optional when entry in “monitorkeys.sig
- -e: Alternate Management Station; optional when entry in “monitorkeys.sig
- -g: Message group
- -o: Object
- -n: Node name (optional)
- -s: Severity [inform|minor|warning|major|critical]
- -m: Message text/Event text
- -c: Character set: UTF-8|ISO-8859-1..ISO-8859-10, ISO-8859-13..ISO-8859-16
- -6: only ipv6

Return code:

0: successful
1: input error
2: communication error

11. Program “remoteconfd”

Background process for central processing of the configuration files from the Management Station. The continuous data over the network are encrypted using the AES method with block chaining mode (CBC). The encryption is done dynamically; one and the same plaintext is encrypted subsequently always different. The socket type for the configuration files is UDP.

Program call:

remoteconfd [-p <portno>] [-i <client1>] [-j <client2>] [-a <authfile>] [-F <root-fs>]
[-L <logfilename>] [-r] [-m <portno>] [-g <group>] [-4] [-f]

Parameter:

- -p: Port number udp; optional when entry in “monitorkeys.sig
- -i: IP Address or name of Management Station as client; optional when entry in “monitorkeys.sig
- -j: Address or name of the alternate Management Station; optional when entry in “monitorkeys.sig
- -a: File name for the agreement of a string for authentication. The string must not be less than eight and no more than 30 characters
- -F: root-fs for restricting access to files located in this directory or subdirectory
- -r: Switch off command interface, it can no commands are executed by the clients
- -L: Name of the log file (default: remoteconfd.logfile)
- -m: Port for message (start/stop) to the Management Station (TCP); missing the option there is taken the value of the option “-p“
- -g: Message group for start/stop message (default: ADMIN_)
- -4: only ipv4
- -f: Do not fork() from the calling shell

The options "-i" and "-j" with host names or IP addresses ensure that requests can only be made by those clients. If the options are omitted, requests can be made from anywhere.

The default value for the option "-a" is "remoteconfd.auth" in the same directory.

12. Program “winmonagent.exe“

The program provides the standard monitoring for Windows. This and the subsequent programs use the character sets CP1250 to CP1258 for transmission to the Management Station. If other character sets are set on the server, a conversion to UTF-8 is performed.

It is determined:

- Utilization of all local drives
- Existence of tasks in the task list
- Existence of services
- Evaluating the text output of monitoring scripts, see also “scriptmonagent”
- Event logs according Eventides’ (System, Application, Security)
- Listenports
- CPU load on all CPUs
- Threshold Memory
- Threshold Page file
- Occurrence of reboots
- Lifecheck (Heartbeat)

1) Program call

winmonagent [...]
startwinmonagent [...]

2) Configuration File

The program "winmonagent.exe" needs a configuration file named "winmonagent.conf" in the same directory.

The minimum value for pollingsecs:: is 60 (60 seconds, 1 minute).

There are the keywords:

- drives:: Threshold values in percent for Windows file systems
- tasks:: Name and number of instances that must be active
- services:: Names of services that need to be active
- cmd:: Call monitoring scripts and avaluating the text output
- system:: List of to be notified System Event Ids and list of Ids to be suppressed

- application:: List of to be notified Application Event Ids and list of Ids to be suppressed
- security:: List of to be notified Security Event Ids and list of Ids to be suppressed
- cpu:: two threshold values in percent for utilization of all CPUs
- memory:: two threshold in percentage for memory usage
- page:: two threshold values in percent for utilization page file
- listenports:: List of port numbers (tcp) that should be in the state of “listening“ (ipv4)
- listenports6:: ditto for ipv6
- lifecheck:: Turn on registration mechanism for the Management Station

The declaration for the monitoring of drives has the following form:

Form 1:

drives::[<mins>::] -d <percent> [-D <dr1>[[<percent>[,<sev>]]] ...] [-E <dre1> <dre2> ... <dreN>]

Form 2:

drives::[<mins>::]-D <dr1>[[<percent>[,<sev>]]] ...

The number <mins> has the meaning of time in minutes for the repetition of the message when exceeding the threshold. In between, there is only one message if the change is greater than 1 MB. The default value for this is 120 (2 hours). After the option "-d" follows a number with the meaning <percent> utilization rate in % for all detected drives. After the option "-D” there is a list of disks that must be mounted and can have specific thresholds <percent>. After the option "-E" there is a list of drives that are to be excluded from monitoring. The list elements are separated by at least one space from another. If the option "-d" is missing, only the drives listed under "-D" are monitored. The drives are specified in the notation "C:\", "D:\", etc.

If not specified for <sev>, the default value is "minor". If the utilization rate for a drive is greater than 98%, the Severity is changed to "critical". This behavior can be changed by the filter on the Management Station when needed. There you can bind the threshold and/or the Severity to the value of the remaining available memory, which is included in each message.

Monitoring of Tasks:

tasks::taskname-1[OPTIONS] taskname-2[OPTIONS] ... taskname-N[OPTIONS]

After the keyword follows a list of task names plus possibly OPTIONS. Upper/ lower case is not significant, the name must not end with ".exe". The list elements are separated by at least one space.

Form 1:

tasks::taskname[[<sev>]] ...

The tasks specified must exist in the task list at least once. The default value for <sev> is "critical".

Form 2:

tasks::taskname[COUNT[,<sev>]] ...

The number of instances of the tasks must be exactly COUNT. When COUNT is equal to zero it is signaled when the task is active ("should not be running!")

Form 3:

tasks::taskname[MIN-MAX[,<sev>]] ...

Where MAX > MIN. Number of instances of a task in a range from MIN to MAX. For MIN == 0 and MAX > 0 the number of instances may not be greater than MAX, but can also be zero.

Form 4:

tasks::taskname[+MAX[,<sev>]] ...

Number of instances may not be greater than MAX, but must be at least one.

Form 5:

tasks::taskname[-MIN[,<sev>]] ...

Number of instances must not be less than MIN.

Monitoring of Services:

services::servicename-1[OPTION] servicename-2[OPTION] ... servicename-N[OPTION]

Input a list of service names plus possibly OPTION, which are separated by at least one space. The particular name is the "service name", not "display name". Upper/lower case in the name is not significant.

Form 1:

services::servicename ...

It is reported when a service is **not registered**. The Severity is "warning".

Form 2:

services::servicename[<sev>] ...

It is reported with the Severity <sev> if a service is **not running**.

Form 3:

services::servicename[null] ...

Determination of the presence. If the service is **active**, a message is displayed with the Severity "major".

Monitoring Event Logs

The declaration for monitoring the Event Logs has the following form:

system::[error] [warning] [any] <evid1>[[<sev1>]] <evid2>[[sev2]] ... -S <evid_s1> <evid_s2> ...
application::[error] [warning] [any] <evid1>[[<sev1>]] <evid2>[[sev2]] ... –S <evid_s1> <evid_s2> ...
security::[error] [warning] [any] <evid1>[[<sev1>]] <evid2>[[sev2]] ... -S <evid_s1> <evid_s2> ...

For the monitoring of System, Application, and Security Event Log, there is a **list** of Event Ids plus optional Severity, when it occurs, a message is to take place. The list elements are separated by at least one space from another.

By setting the optional error and/or warning the occurrence of an event with the Windows Event type (= Windows Severity) "error" or "warning" is reported. By optionally setting any each new event is reported. Restrictions to this you can make through the option "-S" followed by a list of Event Ids, which are to be suppressed. With any and the option "-S" one can formulate a negative list of known Event Ids classified as insignificant, so only previously unknown Event Ids are reported. Another strategy is to establish a positive list to signal Event Ids in combination with error and/or warning.

If for an Event Id the specification of the Severity is absent, becomes the Windows-Severity "ERROR_TYPE" the Severity "major", "WARNING_TYPE" and "AUDIT_FAILURE" to "warning" and remaining to "inform".

A message contains the Log Type, Event Id, Event Type, Source and Message Strings, if they are available. These data can be filtered again according to content criteria on the Management Station.

3) Examples:

Example 1:

```
destination::NEPTUN
portno::55555
attribute::Windows Standard-Monitoring
pollingsecs::300
drives::-d 93 -D D:\[95,maj]
#end of configuration file
```

The Management Station is NEPTUN. The transmission port is 55555/tcp. There are two attributes "Windows" for "Group" and "Standard-Monitor" for "Object" in the Browser of the Management Station. The agent operates as a background process with a polling interval of five minutes.

The general threshold for the utilization rate of the drives is 93%. When exceeding a minor is sent. The threshold for drive “D:\” is 95% and there is a major message. If all drives are recognized, the only question is whether they are specified with individual thresholds or not. If a drive is specified but does not exist, there is a critical message.

By specifying -E G:\ X:\ ... you can exclude drives!

Note: If the utilization rate of a drive is greater than 98%, there is a critical message

Example 2:

```
destination::NEPTUN
portno::55555
attribute::OS Windows
tasks::jrun[-2,crit] webscan inetinfo[1,maj] cmd[+5,warn] ascWrapper[2-5,maj]
services::spooler[maj] snmp[crit] ftpsvc[null]
#end of configuration file
```

The task "jrun" must exist at least twice in the task list, or critical message. The task "webscan" must be active (no matter how many instances), or critical. The task "inetinfo" should exist exactly once, otherwise major message. The task "cmd" shall not exceed five times otherwise "warning" will happen. The number of instances of "ascWrapper" should be between two and five, otherwise major message.

The service "spooler" and "snmp" must be active, otherwise there is a major or critical message. The search argument for a service is the service name, not display name! By name upper/lower case are not distinguished. If the "ftpsvc" is active, a message is displayed with the Severity "major".

The data for tasks and services can be enclosed in quotes if there are spaces in them.

Note: Missing Severity in square brackets for a service there is only checked the registration on the server, but not whether it is active. To test it a Severity must be specified. Replaced by "zero" in the square bracket there is carried out a reverse test, reports that a service is active.

Example 3:

```
destination::NEPTUN
portno::55555
attribute::OS Windows
security::error 529[warn] 578[min]
system::error warning 4319[crit] 100[min] -S 10001 36871 9001 8032  500
770 104 257
application::error 4034[warn] 5000[min] 1000 1001[crit] -S 300 301 302 1066
1003 4625 900
#end of configuration file
```

Monitoring the Event Logs of Windows:

Searching argument is the Event Id and/or the Event Type "error", "warning". A list can be defined by Event Ids, which should be suppressed.

For the Security Event Log Id 529 (wrong logging in) and 578 is (shutdown of the system) are monitored. In addition, all events with the Windows Severity "ERROR_TYPE". In the System Event Log displays all events with the Windows Severity "ERROR_TYPE" and "WARNING_TYPE". In addition, the occurrence of the Event Id 4319 (duplicate IP address) brings about a critical mes-

sage and Event Id 100 (bad FTP login) causes a minor message. The events with the Event Id 10001, 36871, 9001, 8032, 500, 770, 104, 257 are to be suppressed.

The "-S" option with the following list of Event Ids to be suppressed must be at the end of the line.

Note: The search arguments Event Id and Event Type can be understood as a pre-selection, because the attributes "Source" and “Message Strings” in an appropriate message are included. These can additionally be filtered on the Management Station.

Example 4:

```
destination::192.168.178.21
portno::55555
attribute::OS Windows
security::529[warn] 578[min]
application::4034[warn] 5000[min]
lifecheck::
#end of configuration file
```

Lifecheck means that at each call a separate control information is sent to the Management Station. In the absence of a corresponding message signaling occurs.

Example 5:

```
destination::192.168.178.21
portno::55555
attribute::Windows Performance
security::529[warn] 578[min]
application::4034[warn] 5000[min]
cpu::96[min] 99[crit] 120 # average over 120 minutes, 96% minor 99% critical
#cpu::95[min] 100[crit] # thresholds: 95% minor, 100 critical
memory::90[warn] 99[crit] # thresholds 90% and 99%, Severity warning and critical
page::95[maj] 99[crit] # thresholds 95% and 99%, Severity major and critical
lifecheck::
#end of configuration file
```

Performance monitoring: CPU load, memory utilization and utilization rate page file.

By means of the program startwinmonagent.exe you can call the agent as a background task. It must be in the same directory as the agent program and has

the same parameters. By using the establishment "Scheduled Tasks" one can activate the monitor at startup or when you reboot the operating system.

4) Scheduling

If the program is not operated by the keyword pollingsecs:: as a background process, it can be called with the establishment of "Scheduled Tasks" periodically. The interval should be no greater than 30 minutes. Recommended is 5 to 20 minutes.

13. Program “logmonagent.exe“

Analyses of log files for Windows. In the configuration file multiple log files can be declared.

1) Program call

logmonagent [-c <configdatei>] [...]
startlogmonagent [-c <configdatei>] [...]

2) Configuration File

The configuration file is named "logmonagent.conf" and is located in the same directory as the agent program. You can also monitor the (growing) size of the target files.

The minimum value for pollingsecs:: is 2 (2 seconds).

There are the keywords:

- logfile:: Analysis of the increase since the last evaluation
- incremental:: Identical with logfile::
- total:: Evaluation of the entire file after change

The structure of the configuration file is as in the agent program logmonagent for Unix. When specifying file names, one can use Meta characters '*' and '?'.

3) Example:

It is the Oracle alert log to be monitored.

```
destination::NEPTUN
portno::55555
attribute::ORACLE ORALOG
incremental::512[crit]::D:\opt\alertlog\ora_alert_log::”^ORA-1631“-crit[3] ”^ORA-”-maj
”warning/i”-warn[-2] “error/i”-maj
“fatal/i”-crit # drift net for lines with “FATAL”, “fatal”, …
“shutdown”-warn
#end of configuration file
```

If the file is larger than 512 MB, there is a critical message. This monitoring is optional. The Filter “^ORA-1631“-crit[3] limits the display to three, even if more lines that match this search patterns are added between two calls. Each line starting with "ORA" is displayed with the severity "major". Lines containing

“warning”, “error”, “fatal” appears with the Severity "warning", "major", "critical". A line with "shutdown" appears as a "warning".

When specifying the file name, you can also use the Meta character '*' and '?' in the base name. Then all files matching the search pattern are monitored.

To start a task in the background there is the program startlogmonagent.exe, which must be in the same directory as logmonagent.exe and has the same parameters.

14. Program “scriptmonagent.exe”

The agent program calls per configuration file executable programs, filtering its text output and sends the result to the Management Station when needed. With one configuration file you can define multiple monitoring programs. The program can be parameterized with different configuration files. It can be run as a batch program or as a background process.

The programs defined in the configuration file are called periodically; each program is executed in parallel in its own *thread*. There is an adjustable timeout for each program. If the limit is exceeded, the program is stopped by the system function *TerminateProcess()* and a critical message is sent to the Management Station. **IMPORTANT:** The output takes place immediately after the end of the program but not during the program runtime. A program in a *thread* can only be called again when the previous program run has ended. The runtime can be longer than the polling interval.

1) Program call

scriptmonagent [-c <configdatei>] [...]

If no configuration file is specified with the option "-c", the system uses the default name "scriptmonagent.conf". Program and configuration file must be in the same directory!

The minimum value for pollingsecs:: is 2 (two seconds).

2) Configuration File

There's the key word:

- cmd:: Declaration of an executable program with name and filters for the output

The lines for the scripts have the form:

cmd::[<name>::][<tos>::]<full_programname>::filter_1 filter_2 ... filter_n

tos: Timeout in seconds, default: 30 (> 0)
name: Logical name (label)

The program must have an output to *stdout* and/or *stderr*. It should be noted that a call to the monitoring program might generate more than one message when

the output has more than one line, each of which corresponds to the pattern of the filter.

Note: The monitoring program must be executable and provide a return code of zero. Otherwise, there is a separate system error message to the Management Station. Similarly, there is a message when the timeout for the script is exceeded.

To start a task in the background there is the program startscriptmonagent.exe, which must be in the same directory as scriptmonagent.exe and has the same parameters.
Example: Evaluation of the command “netstat -an“

```
destination::192.168.178.21
portno::55555
attribute::Windows
pollingsecs::30

# analysis of current network connections
cmd::NETSTAT-ANO::netstat -ano::":[0-9]{1,5}[ ]+(127.0.0\.1|\[\::1\]):"-null
":[0-9]{1,5}[ ]+192\.168\."-null # suppress your own addresses
":[0-9]{1,5}[ ].*:[1-9][0-9]{,4}[ ];;Unknown foreign addr: $3/$1 PID: $$"-warn[+8] # show unknown

# negative filter
cmd::NETSTAT-AN::netstat -an::"ESTABLISHED;;Should not be more than 100: $0"-warn[-100]
"ESTABLISHED;;Should not be less than five: $0"-min[<5]
```

The first filter reports, if there are more than 100 lines found. The second filter is a negative filter and signals when there are less than five.

It should be noted that for negative filters the sequence in the list of filters is irrelevant.

15. Program “asyncmonagent.exe”

The program is an extension of the agent programs winmonagent.exe, scriptmonagent.exe **and** logmonagent.exe. Each script and each log file that is declared in the configuration file can be provided with its own polling interval or can be defined as a cron job. The execution of the individual monitoring functions is done concurrently in the form of threads. The agent is operated solely as a background process. There is no default timeout for the execution of programs and scripts. Their output is sent immediately after filtering in real time to the Management Station. Moreover, it applies the description of asyncmonagent as on Unix.

1) Program call

asyncmonagent [-c <configdatei>] [...]

If no configuration file is specified with the option "-c", the system uses the default name "asyncmonagent.conf". Program and configuration file must be in the same directory!

2) Configuration File

There are the key words:

- cmd:: Declaration of an executable program with name and filters for the text output, identification of the program either in short form (for example, "netstat -ano" or “%netstat -ano”) or with full path, including extension (.exe, .bat)
- logfile:: Mode “growth since the last call to evaluate“
- incremental:: identical to logfile::
- total:: Mode “evaluate entire file if it has changed“
- lifecheck:: Dynamic registration at the Management Station

The indication of the executable program in short form causes the command to be executed internally with "cmd.exe /c ...". Specifying a program with full path (containing backslash character ‘\’) causes the direct execution. In this case, the file type extension (.exe, .bat) cannot be missing and the base name of the file cannot contain spaces! Possible parameters follow the program name separated by at least one space.

Example:

```
# Programm “wmic.exe“ will start after 15 seconds and runs infinitely, direct execution
cmd::CPULOAD-WMI::15::C:\Windows\system32\wbem\WMIC.exe
cpu get loadpercentage /every:20::”[0-9]+”-info
# “wmic ...“ is internally executed with “cmd.exe /c“
cmd::PROCESS-WMI::30::wmic process list brief::“[ ]([7-9][0-9]|[0-9]{3,})[ ]“-warn
```

To start a task in the background there is the program startasyncmonagent.exe, which must be in the same directory as asyncmonagent.exe and has the same parameters.

Examples:

```
destination::192.168.178.20
portno::55555
attribute::Windows AsyncMonitorCollection
lifecheck::120
# will send every 120 seconds a heartbeat signal to the Management Station

suppressionsecs::3600
# will suppress the repetition of identical messages within one hour

# CRONJOBS
cmd::0 10 * * 1-5::echo It is ten o’clock a.m. (mo-fr)::”*”-info

# evaluate “netstat -e”, every 15 minutes from 6 - 21 (excluding 21)
cmd::*/15 6-20 * * *::netstat -e::
”^Errors[ ]+0[ ]+0”-null “^Errors;;Errors received: $2, errors sent: $3”-min

# wmic for processes, polling interval 5 minutes
cmd::THREADS-PROCESSES::5.0::wmic process get
Name,ProcessID,ThreadCount::”[ ][0-9]+[ ]+[0-9]{3,};;$1 id: $2 thrc: $3”-min

# analysis of current network connections,  polling interval 4 seconds
cmd::NETSTAT-ANO::4::%netstat -ano::”:[0-9]{1,5}[ ]+(127.0.0\.1|\[\::1\]):”-null
“:[0-9]{1,5}[ ]+192\.168\.”-null # suppress your own addresses
“:[0-9]{1,5}[ ].*:[1-9][0-9]{,4}[ ];;Unknown foreign addr: $3/$1 PID: $$”-warn[8] # show unknown

# … more functions
```

16. Program “rsendmsg.exe“

The program sends messages directly to the Management Station. There you can filter or transpose them as needed. Used in scripts you can implement any monitoring functions. The associated Event Type is "Rsendmsg".

Program call

rsendmsg.exe -p <port> -d <server> [-e <alternate server>] -g <group> -o <object> [-n <nodename>] -s <severity> -m <message text>

Parameter:

- -p: Port number tcp; optional when entry in “monitorkeys.sig
- -d: Management Station; optional when entry in “monitorkeys.sig
- -e: Alternate Management Station; optional when entry in “monitorkeys.sig
- -g: Message group
- -o: Object
- -n: Nodename (optional)
- -s: Severity (inform|minor|warning|major|critical)
- -m: Message Text/Event Text

Returncode:

0: successful
1: input error
2: communication error

17. Program “remoteconfd.exe“

Background process for central processing of the configuration files from the Management Station. The data over the network are encrypted using the AES method with block chaining mode (CBC). The encryption is performed dynamically; the same plaintext is encrypted subsequently always different. The Socket Type for the communication is *udp* or *tcp*.

Program call:

remoteconfd.exe [-p <portno>] [-t] [-i <client1>] [-j <client2>] [-u] [-a <authfile>]
[-F <root-fs>] [-L <logfilename>] [-r] [-m <portno>] [-g <group>] [-6] [-h]

Parameter:

- -p: Port number; optional when entry in “monitorkeys.sig
- -t: Listening on *tcp*, otherwise *udp*
- -i: IP Address or name of the Management Station as client; optional when entry in “monitorkeys.sig
- -j: Address or name of the alternate Management Station; optional when entry in “monitorkeys.sig
- -u: UTF-8 output for the command interface; default: current *code page* (e.g. CP1252)
- -a: File name for the agreement of a string for authentication. The string must not be less than eight and no more than 30 characters
- -F: root-fs for access to files
- -r: Shutdown command interface, no commands can be executed by the clients
- -L: Name of the log file
- -m: Port for message to the Management Station (TCP)
- -g: Message group for start-/stop message (default: ADMIN_)
- -6: ipv6, otherwise ipv4
- -h: help text

The options "-i" and "-j" with the host name or IP address ensure that requests can only be made by those clients. Missing, the options can be made from any inquiries. The default value for the option "-a" is "remoteconfd.auth" in the same directory.

To start the task in the background there is the program startremoteconfd.exe, which must be in the same directory and has the same parameters as remoteconfd.exe. With the establishment of "Scheduled Tasks" you can activate so monitoring when booting the operating system.

Example:

startremoteconfd -p 55555

The program remoteconfd.exe in the same directory is called and pushed into the background. It listens on port 55555/udp.