Integrated Server Monitoring

Beschreibung der Agenten

© Wilhelm Buchholz, Im Bruche 6, 31275 Lehrte

http://www.monitor-site.de

mailto:new-monitoring@t-online.de

Inhaltsverzeichnis

1. Einleitung

Die Agenten bieten die folgenden Gruppen von Funktionen an:

- Standardüberwachung: Filesysteme, Prozesse, Syslog bzw. Eventlogs, Performance, Listenports, Lifecheck/Heartbeat
- Logfileauswertung: Anwendungsüberwachung, Systemüberwachung
- Beliebige Überwachungsscripte: Anwendungsüberwachung, Systemüberwachung
- Direktes Schicken von Meldungen zum Browser der Management-Station

Das Agentenprogramm für Basismonitoring verfügt über fertige Funktionen mit einer standardmäßige Ausgabe für die zentrale Anzeige der Management-Station.

Ebenso ist es mit den Agenten für die Logfileauswertung. In den Konfigurationsdateien – fertig, neu oder modifiziert – hinterlegt man Dateinamen, Filter und Meldungsschweren.

Die verschiedenen Agenten brauchen nicht installiert zu werden! Die Programmdatei und die zugehörige Konfigurationsdatei(en) werden jeweils auf die zu überwachenden Server gebracht und dort unter einer User-ID seiner Wahl betrieben. Die Registrierung an der Management-Station geschieht automatisch bei der ersten Benutzung. Es kann bei Bedarf auch mehr als eine Instanz eines Agenten betrieben werden, die sich in den Konfigurationsdateien und Zugriffsrechten unterscheiden.

Jeder Agent kann entweder als Batch-Programm, das nach jedem Aufruf und Abarbeitung terminiert, oder als Hintergrundprozess (daemon) betrieben werden. Im ersten Fall wird er von einem Scheduler periodisch aufgerufen. Im zweiten Fall wird er beim Systemstart von einer rc-Routine (z.B. rc.local) gestartet und das Polling Intervall von einem Eintrag in der Konfigurationsdatei bestimmt. Beim Start geht eine informative Meldung mit der PID des Prozesses, dem Namen des Agentenprogramms mit vollständiger Pfadangabe und dem Namen der Konfigurationsdatei an die Management-Station. Beim Stoppen des Hintergrundprozesses erfolgt eine Meldung mit den gleichen Angaben aber der Severity “major“ an die Management-Station.

Die Funktionsweise der Agenten und der Aufbau der Konfigurationsdateien werden im Folgenden anhand von Beispielen erläutert.

Die Konfigurationsdateien sind nach dem Prinzip der minimalen Eingabe gestaltet.

2. Allgemeine Vereinbarungen

Die Konfigurationsdateien zu den Agenten bestehen aus Schlüsselwörtern gefolgt von Angaben zur Ausführung der Überwachungsfunktion wie zum Beispiel Schwellwerten. Ein Kommentar beginnt mit einem ’#’ bis zum Zeilenende. Soll das Zeichen ’#’ für einen anderen Zweck, zum Beispiel in einem Suchmuster, verwendet werden, muss es mit einem vorangestellten Backslash (“\#“) versehen werden! Leerzeilen werden ignoriert. Die Reihenfolge der Schlüsselwörter ist egal. Die Sonderbedeutung des Trennsymbols “::“ lässt sich mit einem vorangestellten Backslash ausschalten (\::)!

Ein Schlüsselwort muss in der ersten Spalte der Konfigurationsdatei beginnen!

2.1 Meldungsschwere (Severity)

Es gibt die Meldungsschweren (Severities):

1) info: inform, informativ
2) min: minor, geringfügig
3) warn: warning, warnung
4) maj: major, schwerwiegend
5) crit: critical, kritisch

Die Meldungsschweren “minor“ und “warning“ weisen auf eventuell bevorstehende Probleme hin. Die Meldungsschweren “major“ und “critical“ bedeuten die Notwendigkeit zum Eingreifen.

2.2 Filter (Protokolldateien, Überwachungsskripte)

Ein Filter zur Auswertung von Protokolldateien und Überwachungsprogrammen besteht aus einem Suchmuster (*extended regular expression*, POSIX Standard), einem Formatstring (optional), der Meldungsschwere (Severity) und einer Angabe zur Häufigkeit von gefundenen Einträgen. Das Suchmuster und der optionale Formatstring werden mit einem Anführungszeichen ’“’ am Anfang und einem Anführungszeichen gefolgt von einem Bindestrich ’-’ am Ende eingeschlossen und sind getrennt durch das doppeltes Semikolon “;;“, das am weitesten rechts steht.

Wenn zwischen den begrenzenden Anführungszeichen ein Anführungszeichen kommen soll, muss es mit einem vorangestellten Backslash (\“) versehen werden!

Ein Filter hat die Form:

1) “[!]<reg.expr.>[/i|v|!][;;<formatstring>]“-severity: Wenn nach der Severity eine Zahl in eckigen Klammern fehlt, werden maximal **fünf** der mit <reg.exp.> gefundene Zeilen zur Management-Station geschickt; jedoch werden alle gefundenen Zeilen gezählt; für Negativ-Filter ist der voreingestellte Wert Eins (mindestens eine Zeile muss gefunden werden)
2) “<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[0]: Null in eckigen Klammern bedeutet, dass **alle** gefundenen Zeilen zur Management-Station geschickt werden
3) “<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[N]: Maximal N gefundene Zeilen werden zur Management-Station geschickt; die Gesamtzahl der gefundenen Zeilen und der damit verbundene Zeitraum wird binär zur Management-Station geschickt und dort zur Anzeige gebracht
4) “<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[+N]: Ein Pluszeichen ‘+’ vor N bedeutet, dass maximal N gefundene Zeilen zur Management-Station geschickt werden; die Gesamtzahl der gefundenen Zeilen wird **nicht** zur Management-Station geschickt
5) “<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[-N]: Ein Minuszeichen ’-’ vor N bedeutet den oberen Schwellwert (Maximalwert) für gefundene Einträge, d.h. es wird **eine** gefundene Zeile geschickt, wenn mehr als N Zeilen gefunden werden; Gesamtzahl und Dauer wird übertragen
6) “<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[>N]: Das Zeichen ‘>’ vor N bedeutet ebenfalls den Maximalwert für gefundene Einträge; jedoch gibt es nur eine Meldung über die Anzahl der gefundenen Zeilen (N >= 0)
7) “<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[<N]: Das Zeichen ‘<’ vor N bedeutet Negativ-Filter für den erforderlichen Minimalwert der gefundene Zeilen, d.h. es müssen mindestens N Einträge gefunden werden
8) “!<reg.expr.>[/i|v|!][;;<formatstring>]“-severity[N]: Negativ-Filter (wie Punkt 7), N ist der untere Schwellwert (Minimalwert) für die Anzahl gefundener Zeilen; d.h. mindestens N Einträge müssen gefunden werden, sonst wird eine Meldung erzeugt
9) “<reg.expr.>[/i|v|!]“-null: null anstelle von severity bedeutet: Gefundene Zeilen werden unterdrückt (Suppressionfilter)

N ist eine natürliche Zahl größer als Null oder größer gleich Null (Fall 6),
severity := info|min|warn|maj|crit

Die Zeichenkette links von “;;“ bildet das Suchmuster, rechts von “;;“ die Formatanweisung für gefundene Zeilen. Wenn “;;“ fehlt, wird eine gefundene Zeile im Normalfall insgesamt übertragen. Die Sonderbedeutung des doppelten Semikolons kann durch ein vorausgehendes Backslash “\;;“ ausgeschaltet werden, wenn mehr als ein “;;“ vorkommt oder es Teil des Suchmusters ist.

Ein vorangestelltes Ausrufungszeichen ’!’ vor dem Suchmuster bedeutet Negativ-Filter, der eine Meldung dann erzeugt, wenn für eine Auswertung der Suchvorgang insgesamt erfolglos war oder zu wenig Einträge bzw. Zeilen gefunden wurden (Suche nach Abwesenheit). Für Negativ-Filter spielt die Reihenfolge in der Liste der Filter keine Rolle für die anderen Filter sehr wohl! Die Sonderbedeutung von ’!’ kann durch ein vorausgehendes Backslash (“\!“) ausgeschaltet werden. Die andere Möglichkeit besteht darin, den Operator ’<’ in eckigen Klammern zu verwenden (Fall 7).

Die Option “/i“ am Ende des Suchmusters bewirkt, dass Groß- und Kleinbuchstaben **nicht** unterschieden werden. Die Option “/v“ kehrt das Suchergebnis bei Beachtung der Groß/Kleinschreibung um. Die Option “/!“ kehrt das Suchergebnis um, Groß/Kleinschreibung wird **nicht** unterschieden.

Beispiel: Das Suchmuster “error/i“ trifft Zeilen mit “ERROR“, “Error“ und “error“. Das Suchmuster “normal/!“ trifft Zeilen, die “NORMAL“, “Normal“, “normal“ **nicht** enthalten. Das Suchmuster “*“ hat eine Sonderbedeutung und trifft jede Zeile.

Bei mehrzeiliger Auswertung teilt sich der reguläre Ausdruck durch das Sonderzeichen ’\n’.

Beispiel: “<muster-1\nmuster-2\nmuster-3>“-warn

Es gibt eine Meldung, wenn die drei Suchmuster in drei aufeinander folgenden Zeilen treffen.

Es ist eine gemeinsame Suche über maximal zwanzig Zeilen möglich. Die Sonderbedeutung von “\n“ kann durch ein vorausgehendes Backslash (“\\n“) ausgeschaltet werden.

Die von den Suchmustern gefundenen Zeilen werden zusammen mit dem Namen der Protokolldatei oder des Kommandos und einem vorangestellten Prompt zur Management-Station übertragen. Wenn in dem Filter ein Formatstring angegeben ist, wird anstelle der ganzen Zeile das Ergebnis der Transformation übertragen.

Beispiel Filter:

Eine Sequenz von drei Filtern für den Befehl “netstat -an“ (Linux)

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

Der erste Filter sucht nach Zeilen, die “ESTABLISHED“ enthalten. Wenn mehr als 90 Zeilen auftreten, gibt es genau eine Meldung, die Gesamtzahl der gefundenen Zeilen wird binär übertragen und an der Management-Station angezeigt.

Der zweite Filter unterdrückt alle Zeilen, die **nicht** “ LISTEN “ enthalten. Der dritte Filter erfasst die verbleibenden Zeilen, die **nicht** die (erlaubten) Portnummern 22, 25, 110, 111, 2049 enthalten. Es werden maximal drei gefundene Zeilen übertragen, die Gesamtzahl wird übertragen (für das Agentenprogramm asyncmonagent gibt es eine andere Darstellung, siehe unten).

In der gleiche Art und Weise geschieht die Auswertung von Protokolldateien.

2.3 Formatstrings (Logfileauswertung, Überwachungsskripte)

Ein Formatstring dient dazu, ein durch die Auswertung einer Protokolldatei oder durch die Auswertung eines Skriptes gewonnenen Text bzw. Textzeile schon am Agenten umzusetzen, um die Lesbarkeit und die Aussagekraft zu erhöhen. Alternativ oder als Ergänzung dazu kann man den gleichen Mechanismus an der Management-Station bei den dortigen Filtern benutzen. Ein Formatstring besteht aus verschiedenen Sonderzeichen, die mit bestimmten Operationen verbunden sind. Die Sonderbedeutung der Zeichen lässt sich mit vorangestelltem Backslash ’\’ ausschalten.Es gibt die Sonderzeichen:

1) $n oder ${n}: n ist eine Zahl [1..99]. Gibt aus das <n>. Wort des Eingangstextes
2) %n oder %{n}: linksshift, gibt aus den um <n> Spalten nach links verschobenen Eingangstext, die Eingangszeile selbst bleibt unverändert
3) &n oder &{n}: Verschiebung um <n> Zeichen (*character*) nach links des Eingangstextes, es gibt keine unmittelbare Ausgabe, der neue Textanfang der Eingangszeile wird automatisch auf den Anfang eines Wortes bzw. Spalte gelegt
4) &{n,m}: Gibt aus <m> Zeichen ab dem <n>. Zeichen der Eingangszeile
5) &{n[|#]substring}: Suche nach Substring in einem Wort. Gibt aus ab dem <n>. Zeichen bis zum Substring substring im gleichen Wort. Wenn substring nicht gefunden wird, erfolgt die Ausgabe bis zum Ende des Wortes (Sonderzeichen ist entweder ‘|‘ oder ‘#‘)
6) @n oder @{n}: Verschieben der Eingangzeile nach links um <n> Spalten, die ursprüngliche Spalte <n+1> steht danach am Anfang der Eingangzeile, es gibt keine unmittelbare Ausgabe
7) %<[n|]substring>: Suche nach einem Substring in der ganzen Zeile oder optional nach dem <n>. Auftreten eines Substrings in der Zeile (n > 0). Shift nach links in der Eingangszeile zu dem Unterstrings substring. Der

gefundene Substring bildet den neuen Anfang der Eingangszeile, es erfolgt keine unmittelbare Ausgabe

8) ?<[n|]substring>: Gibt aus den bis substring nach links verschobenen Text der Eingangszeile. Wenn substring gefunden wird, terminiert die Formatierung, sonst wird mit dem folgenden Sonderzeichen fortgefahren; optional mehrmaliges Auftreten
9) -<[n|]substring>: Gibt aus den an der Stelle des Auftretens von Substring substring abgeschnittenen Eingangstext, der gefundene Substring wird mit abgeschnitten, die Eingangszeile bleibt unverändert; optional mehrmaliges Auftreten
10) $*: Gibt aus die ganze Eingangszeile
11) $$: Gibt aus die letzte Spalte/letztes Wort des Eingangstextes
12) $0: Gibt aus das Suchmuster, das die Eingangzeile herausgefiltert hat

Enthält der Formatstring keine Sonderzeichen, wird die gefundene Zeile bzw. der gefundene Eintrag durch die Stringkonstante ersetzt. Die Suche nach einem Unterstring erfolgt von links nach rechts. Wird bei der mehrmaligen Suche die Anzahl <n> nicht erreicht, wird der am weitesten rechts stehende Unterstring genommen. Wird der Unterstring nicht gefunden, erfolgt keine Operation.

Beispiel 1: “Fatal|FATAL|fatal;;Zeile mit \“Fatal\“ aufgetreten: $1 %5“-maj

Das Suchmuster als regulärer Ausdruck zum Finden einer Zeile befindet sich links von “;;“, der Formatstring zum Bearbeiten einer gefundenen Zeile beginnt rechts davon. Der Ausgangszeile geht voran “Zeile mit “Fatal“ aufgetreten: “. Dann kommt die erste Spalte bzw. Wort der Eingangszeile gefolgt von der um fünf Spalten nach links verschobenen Eingangszeile.

Dazu äquivalent:

“fatal/i;;Zeile mit \“Fatal\“ aufgetreten: $1 @5$*“-maj

Der Operator “@5“ bewirkt die Verschiebung um fünf Spalten nach links der Eingangszeile, aber keine Ausgabe. Die erfolgt mit dem Operator “$*“, der die veränderte Eingangszeile insgesamt ausgibt.

Beispiel 2:

Die folgende Zeile soll formatiert werden:

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

Ausgabe für den Event-Text an der Management-Station:

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

Der Filter besteht aus dem Teil links von “;;“ (Suchmuster für die Zeile) und rechts davon die Bearbeitung der gefundenen Zeile. Der Ausdruck “%<] >“ (oder “%<1|] >“) verschiebt die Eingangszeile bis zum ersten Auftreten von Unterstring “] “ nach links. Das nachfolgende “@1“ verschiebt die Eingangszeile nochmals um eine Spalte nach links, so dass der Unterstring verschwindet. Durch den Ausdruck “&{1|[}“ (oder “&{1\#[}“) wird die Spalte “oracle[3065]“ an der Stelle “[“ abgeschnitten und zu “oracle“. Schließlich wird durch den Operator “%1“ die um eine Spalte nach links verschobene Eingangszeile ausgegeben. Die Ergebnisse der einzelnen Operationen werden in der Ausgabe miteinander verkettet.

2.4 Verschlüsselung und Netzverbindung

Für die verschlüsselte Datenübertragung von den Agenten zur Management-Station gibt es die Datei mit dem festen Namen “monitorkeys.sig“. Sie enthält eine Signatur, die aus 32 Bytes mit den Wertebereichen jeweils [0..ff] besteht. Daraus werden zur Laufzeit die geheimen Schlüssel generiert, die die Basis für eine dynamische Verschlüsselung bilden.
Die Datei wird zuerst im gleichen Verzeichnis wie das Agentenprogramm gesucht, dann im Home-Verzeichnis des Benutzers, unter dessen Rechte die Programme laufen und dann im Verzeichnis “/etc“ (Windows: “C:\“). Im letzteren Fall erstreckt sich der Gültigkeitsbereich der Datei über den ganzen Server, wenn sie an den anderen Stellen nicht existiert. Auf der Gegenseite muss die gleiche Datei mit der gleichen Signatur existieren und gültig sein. Ohne die Datei kann ein Agent nicht gestartet werden.

Die Signatur der Datei wird zu Beginn des Betriebes festgelegt und im Weiteren nicht mehr verändert. Die Agenten werden zusammen mit dieser Datei verteilt. Zwei Systeme mit unterschiedlichen Signaturen sind nicht miteinander kompatibel.

In der gleichen Datei kann man optional und an zentraler Stelle die Angaben für die Netzverbindung zur Management-Station festlegen, die dann für alle Agenten gilt und nicht verändert werden soll.

Dafür gibt es die Schlüsselwörter:

- destination:: <management-station> [<ausweich-management-station>]
- portno:: <programmnummer>

Man kann einen DNS-Namen oder eine IP-Adresse verwenden.

Beispiel:

```
# Signatur zur Verschlüsselung in Hexziffern, 32 getrennt durch ':'
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
# Netzverbindung zur Management-Station
destination::192.168.20.10
portno::55555
```

Man beachte: Die Einträge zur Netzverbindung werden durch Vereinbarungen unmittelbar am Agenten überschrieben. Das Kommentarzeichen '#' darf in dieser Datei nur am Zeilenanfang stehen.Optionen

Alle Agenten können mit den folgenden Optionen aufgerufen werden:

- -d: Adresse oder Name der Management-Station
- -e: dito Ausweichstation
- -p: Portnummer (tcp)
- -x: Auf jeden Fall als Hintergrundprozess starten
- -f: Es erfolgt kein fork() von der aufrufenden Shell
- -t: Testmodus

Die Angaben durch Optionen überschreiben eventuelle Angaben in den Konfigurationsdateien und sind nicht veränderbar.

2.5 Schlüsselwörter

Es gibt in den Konfigurationsdateien die folgenden gemeinsamen Schlüsselwörter:

- destination:: <managementserver> [<ausweich-managementserver>]
- portno::<programmnummer>
- attribute::<gruppe> [<objekt>]
- pollingsecs::<sekunden>
- suppressionsecs::<sekunden>
- nodename::<aliasname>
- disabled::
- charset::UTF-8|ISO-8859-1..ISO-8859-10, ISO-8859-13..ISO-8859-16; fehlt die Angabe, wird der Text geparst und entweder UTF-8 oder ISO-8859-1 (Latin1) genommen (gilt nur für Unix)

Der zweimalige Doppelpunkt gehört zum Schlüsselwort. Dahinter folgen die Argumente als Stellungsparameter, durch Leerzeichen getrennt, wenn es mehrere sind.

Für destination:: kann man den DNS-Namen oder die IP-Adressen der Management-Station und optional des Ausweichservers angeben, sowohl ipv4 als auch ipv6. Bei zwei Angaben müssen sie durch mindestens ein Leerzeichen getrennt sein. Die Angabe portno:: definiert die Programmnummer (tcp). Die Einträge zur Netzverbindung können auch in der Datei "monitorkeys.sig" gemacht werden (siehe oben). Dann gelten sie für alle Agentenprogramme in dem Gültigkeitsbereich.

Die Namen für <gruppe> und <objekt> müssen jeweils einen zusammen hängenden String bilden, die durch mindestens ein Leerzeichen von einander getrennt sind, und können bis zu dreißig Zeichen lang sein. Umlaute sind nicht zulässig. Das Schlüsselwort disabled:: inaktiviert den Agenten für diese Konfigurationsdatei. Das Schlüsselwort pollingsecs:: bewirkt, dass das Agentenprogramm nach dem Start als Hintergrundprozess läuft, der seine Funktionen periodisch mit der angegebenen Anzahl von Sekunden als Polling Intervall ausführt. Für diese Betriebsart ist der Agent beim Systemstart in einer rc-Routine zu vereinbaren. Fehlt das Schlüsselwort pollingsecs::, terminiert der Agent nach jedem Aufruf (Ausnahme: "asyncmonagent", siehe unten).

Das Schlüsselwort suppressionsecs:: gibt einen Zeitraum in Sekunden an, in dem die Wiederholung identisch gleicher, **ausgehender** Meldungen unterdrückt werden soll. Innerhalb dieses Zeitraumes werden nur unterschiedliche Meldungen zur Management-Station geschickt, deren Signifikanz durch eine Transformation mit einem Formatstring beeinflusst werden kann. Die Voreinstellung für diesen Wert ist Null.

Mit dem Schlüsselwort nodename:: lässt sich ein anderer Name („Aliasname") als der im Betriebssystem vereinbarte Netzwerk-Name definieren.

Die oben aufgeführten Schlüsselwörter müssen mit ihren Argumenten in einer Zeile stehen und dürfen nur einmal auftreten.

Darüber hinaus gibt es Schlüsselwörter, die ebenfalls nur einmal vorkommen dürfen, deren Argumente aber in mehreren aufeinander folgenden Zeilen auftreten dürfen. Ferner gibt es Schlüsselwörter, die mehrmals vorkommen dürfen und deren Argumente in mehreren aufeinander folgenden Zeilen stehen können. Die Folgezeilen terminieren, wenn eine Zeile mit einem Schlüsselwort auftritt oder eine Leerzeile oder eine Zeile mit einem Kommentar oder das Ende der Konfigurationsdatei erreicht ist. Die Kennzeichnung einer Zeile als Folgezeile ist nicht nötig.

Einmalige Schlüsselwörter mit Folgezeilen: filesystem::, fstypes::, procs::, procs_ef::, drives::, system::, application::, security::, tasks::, services::, logfiledir::, files::, filter::, exclude::

Mehrmalige Schlüsselwörter mit Folgezeilen: cmd::, syslog::, logfile::, incremental::, total::, restart::

Im Folgenden werden die Schlüsselwörter in Zusammenhang mit den einzelnen Agenten erläutert.

3. Programm “basemonagent“

Das Programm realisiert die Standardüberwachung für Unix.

Es wird ermittelt:

- Auslastung der Filesysteme (Belegungsgrad)
- Auslastung Inodes von Filesystemen (Belegungsgrad)
- Existenz von Prozessen in der Prozessliste
- Listenports
- Nachstarten von Prozessen
- Auftreten Zombie-Prozesse
- CPU-Belastung und Memory
- Swap + Memory
- Load Average und Auftreten von Reboots
- Auswertung von Syslogdateien und anderen Logfiles
- Auswertung der Rückgabe von Überwachungsskripten
- Lifecheck (Heartbeat)

1) Aufruf

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

Fehlt die Option “-c“ heißt die Konfigurationsdatei „basemonagent.conf“ und liegt im gleichen Verzeichnis. Auch die mit “-c“ angegebene Datei muss im gleichen Verzeichnis liegen wie die Programmdatei.

Die Option “-t“ schaltet den Testmodus ein. Meldungen erscheinen auf *stdout*, gehen nicht über das Netz.

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

- filesystem:: Liste von Filesystemnamen, Schwellwerte und Severities
- fstypes:: Liste von Filesystemtypen, die überwacht werden sollen
- procs:: Liste von Prozessen entsprechend „ps -e“
- procs_ef:: Liste von Prozessen entsprechend „ps -ef“
- restart:: Automatisches Anstarten eines Hintergrundprozesses
- defuncts:: zwei Schwellwerte und Severity für Anzahl Zombies
- listenports:: Liste von Portnummern (tcp), die im Zustand “listen“ sein sollen (ipv4), wird parallel ausgeführt
- listenports6:: dito für ipv6

- cpu:: zwei Prozentwerte und Severity für Überschreitung cpu-Belastung
- load:: zwei Schwellwerte und Severity für load average 15 Minuten, ferner Signalisierung bei Reboot
- swap:: zwei Prozentwerte und Severity für Belegung Swap und Memory
- syslog:: Auswertung von Syslogdateien und anderen Protokolldateien
- cmd:: Parallele (nebenläufige) Ausführung von Überwachungsskripten und Filterung der Ausgabe, siehe auch Kapitel "scriptmonagent"
- lifecheck:: Dynamische Registrierung an der Management-Station

Schwellwerte Inodes:

Der Schwellwert des Belegungsgrades für die Anzahl der Inodes liegt konstant bei 95%.

Filesystemüberwachung:

Die Vereinbarung für die Filesystemüberwachung hat die folgende Form:

Form 1:

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

Form 2:

filesystem::[<mins>::]-D <fs1>[[<prozent_1>[,<sev>]]] ... <fsN>[[<prozent_N>[,<sev>]]]

fstypes::fstype_1 fstype_2 ... fstype_N

Wenn die Zeile mit fstypes:: fehlt, wird **jedes** gemountete Dateisystem erfasst. Ist die Zeile vorhanden, werden nur die Dateisysteme erfasst, deren Typen in der Liste enthalten sind. Die Zahl <mins> hat die Bedeutung Zeit in Minuten für die Wiederholung der Meldung bei Überschreitung des Schwellwertes. Zwischendurch gibt es nur dann eine Meldung, wenn die Veränderung größer als 1 MB ist. Nach der Option "-d" folgt eine Zahl <prozent> mit der Bedeutung Belegungsgrad in % für **alle** erfassten Filesysteme. Nach der Option "-D" kommt eine **Liste** von Filesystemnamen, die gemountet sein müssen und spezielle Schwellwerte <prozent> haben können. Nach der Option "-E" kommt eine **Liste** von Filesystemnamen, die von der Überwachung ausgeschlossen werden sollen. Die Listenelemente werden durch mindestens ein Leerzeichen von einander getrennt. Fehlt die Option "-d", werden nur die unter "-D" angegebenen Filesysteme überwacht. Filesystemnamen können als Suchmuster mit den Sonderzeichen '*', '?' und [...] ausgeführt werden.

Fehlt die Angabe für <sev>, ist der Standardwert “minor“. Wenn der Belegungsgrad für ein Filesystem größer als 98% ist, wird die Severity automatisch zu “critical“. Dieses Verhalten lässt sich durch die Filter an der Management-Station bei Bedarf ändern. Dort kann man den Schwellwert und/oder die Severity an den Wert des noch verfügbaren Speichers binden, der bei jeder Meldung mitgeliefert wird.

Die kürzeste Form der Filesystemüberwachung ist:

filesystem::-d 93 # erster Schwellwert für alle vorhandenen Filesysteme 93%

Filesystemnamen können in Anführungszeichen gesetzt werden, wenn in ihnen Leerzeichen enthalten sind. Beispiel: -D “Volume 1[95,maj]“

Der prozentuale Wert ist der **Belegungsgrad** (belegt/total). Bei 93% sind noch sieben Prozent frei.

Prozessüberwachung:

Die Vereinbarung für die Prozessüberwachung hat die Form:

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

Nach dem Schlüsselwort die Eingabe einer Liste von Prozessnamen plus eventuell OPTIONEN. Die Listenelemente werden getrennt durch mindestens ein Leerzeichen. Bei dem Schlüsselwort procs:: muss der Prozessname genau der Ausgabe von “ps -e“ (MacOS: ps -acx) entsprechen. Bei dem Schlüsselwort procs_ef:: ist der Prozessname ein Suchmuster (*regular expression*) für die Ausgabe von “ps -ef“. Die spezifizierten Prozessnamen einschließlich der OPTIONEN in [...] können in Anführungszeichen gesetzt werden, wenn in den Namen Leerzeichen vorkommen (Beispiel: procs_ef::"httpd -D FOREGROUND[1-5]").

Form 1:

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

Die angegebenen Prozesse müssen mindestens einmal in der Prozessliste existieren. Der Default-Wert für <sev> ist “critical“.

Form 2:

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

Die Anzahl der Instanzen der Prozesse muss genau COUNT sein. Bei COUNT gleich Null wird signalisiert, wenn der Prozess aktiv ist („should **not** be running“)!

Form 3:

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

Wobei MAX > MIN ist. Anzahl der Instanzen eines Prozesses in einem Bereich von MIN bis MAX. Bei MIN == 0 und MAX > 0 darf die Anzahl der Instanzen nicht größer als MAX sein, kann aber auch Null sein.

Form 4:

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

Anzahl der Instanzen darf nicht größer als MAX sein, muss aber mindestens eins sein.

Form 5:

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

Anzahl der Instanzen darf nicht kleiner als MIN sein.

Die gleichen Regeln gelten auch für das Schlüsselwort procs_ef::. Es ist zu beachten, dass im Fehlerfall fortlaufend signalisiert wird bis die Störung behoben ist.

Prozessüberwachung mit Restart:

Die Vereinbarung hat die Form:

restart::[COUNT::]procname::startprozedur

Die Einrichtung bietet die Möglichkeit, nach der Feststellung des Ausfalls eines Hintergrundprozesses diesen unmittelbar danach wieder zu starten. Das ganze geschieht automatisch, so dass die Ausfallzeit möglichst gering ist. Es kann in einer Konfigurationsdatei mehr als eine Zeile dieser Art geben.

Der Prozessname ist ein Suchmuster (*regular expression*) für die Ausgabe von “ps -ef“. Danach folgt der Name der Startprozedur mit vollständiger Pfadangabe, die den geforderten Prozess (wieder) anstartet, wenn er nicht aktiv ist. Die Übergabe von Parametern an die Prozedur ist zulässig. Die Prozedur muss nach

spätestens 5 Sekunden terminieren, sonst gibt es einen Timeout-Fehler. COUNT ist eine Zahl, die die maximale Anzahl von Fehlversuchen darstellt. Wenn die Zahl überschritten wird, gibt es keine weiteren Versuche mehr, den Prozess zu starten, wohl aber eine Signalisierung. Der Default-Wert für COUNT ist 5. Die Startprozedur muss mit dem Exit-Code Null terminieren, sonst gibt es eine entsprechende Meldung.

Bei mehr als einem Prozess erfolgt die Abarbeitung der entsprechenden Zeilen in der Reihenfolge von oben nach unten. Der Ausfall des Prozesses und der Aufruf der Startprozedur wird mit einer kritischen Meldung der Management-Station signalisiert. Der Vorgang ist erst abgeschlossen, wenn beim nächsten Polling Intervall das ordnungsgemäße Laufen des Prozesses festgestellt wird. Dann gibt es eine informative Meldung über die Dauer des Ausfalls. Ansonsten wiederholt sich der Vorgang, bis COUNT überschritten ist.

Wenn für das Anstarten besondere Rechte notwendig sind, muss das Programm basemonagent für diesen Zweck als Root betrieben werden. Möglich ist auch eine nur für diesen Zweck betriebene Instanz von basemonagent, die mit Root-Rechten läuft.

Beispiel:

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

Prozess “snmpd“ wird gestartet oder neu gestartet, wenn er nicht aktiv ist. Der Agent muss mit Root-Rechten laufen.

Logfileauswertung:

Die Vereinbarung hat die folgende Form:

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

Nach dem Schlüsselwort syslog:: folgt optional ein logischer Name (Label) und ebenfalls optional eine Zahl mit dem Schwellwert für die Größe der Zieldatei in Anzahl Megabyte (MB). Der logische Name kann bis zu 30 Zeichen lang sein und muss mit einem Buchstaben beginnen. Dann kommt der volle Pfadname der Syslog-Datei (oder einer anderen Protokolldatei). Danach folgt die Liste der Filter, deren Elemente mit mindestens einem Leerzeichen voneinander getrennt sind. Vor dem Dateinamen können optional die Sonderzeichen ’-’ und/oder ’!’ stehen. Das Zeichen ’!’ bewirkt, dass **keine** Meldung kommt, wenn die Datei nicht existiert. Das Zeichen ’-’ legt die Startbedingung zu Anfang der Überwachung fest. Ohne diese Option kommen beim ersten Lauf Meldungen über die Häufigkeit der gefundenen Suchmuster. Mit der Option unterbleibt das, es wird lediglich der Anfangswert für die inkrementelle Überwachung bestimmt. Da-

nach folgt der Regelbetrieb, bei dem gefundene Einträge bzw. Zeilen, die zwischen zwei Aufrufen hinzugekommen sind, übertragen und dargestellt werden. Wenn sich die Zieldatei verkürzt oder geleert wird, gibt es einen neuen initialen Lauf mit Angabe der Häufigkeit bei gefundenen Einträgen.

Der Name der Zieldatei kann in seinem Basisteil Suchmuster '*', '?', "[...]" für Dateinamen enthalten. Alle Dateien, deren Name in dem Verzeichnis diesem Suchmuster entsprechen, werden dann mit einer gemeinsamen Liste von Filtern überwacht (Beispiel: /var/log/*.log).

3) **Beispiele:**

Beispiel 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
```

Der Name der Management-Station lautet NEPTUN, der Ausweichstation PLUTO. Der Übertragungsport ist 55555/tcp. Es gibt zwei Attribute "OS" für "Group" und "Unix" für "Object" im Browser der Management-Station. Das Polling Intervall ist fünf Minuten (300 Sekunden).

Der allgemeine Schwellwert für den Füllgrad von Filesystemen ist 93%. Bei Überschreitung erfolgt eine Minor-Meldung. Der Schwellwert für das Root-Filesystem ist 85% und es kommt eine Majormeldung, das Filesystem "/home" hat einen Schwellwert von 90% mit der Severity "warning". Die Dateisysteme "/dev" und "/net" sind aus der Überwachung ausgeschlossen.

Es werden alle Filesysteme vom Typ "ext3" und "nfs" erfasst. Fehlt die Angabe fstypes::, werden alle vorhandenen Filesysteme erfasst.

Mit der Angabe "-E /fsname1 /fsname2 /fsname_n" kann man Filesysteme ausschließen. Für die Namen der Mountpoints sind Suchmuster für Dateinamen zulässig.

Fehlt die Option "-d", werden nur die Filesysteme überwacht, die nach der Option "-D" aufgeführt sind.

Eine Zahl nach dem Schlüsselwort filesystem:: gibt die Wiederholungszeit in Minuten an. Die Zahl “60“ bedeutet, dass erst nach 60 Minuten die Meldung wiederholt wird, wenn nicht vorher eine Änderung passiert ist. Der Default-Wert für diese Angabe ist 120.

Achtung: Wenn der Füllgrad eines Filesystems größer als 98% wird, gibt es eine Meldung mit der Severity “critical“.

Beispiel 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::
Der Prozess "java" muss mindestens 2 mal in der Prozessliste existieren, sonst kritische Meldung.
Der Prozess "webscan" muss aktiv sein (egal wie viele Instanzen), sonst kritisch.
Der Prozess "lpd" darf höchstens 5 mal vorkommen, sonst Warning. Die Anzahl der Instanzen von "ascWrapper" soll zwischen 2 und 5 liegen, sonst Majormeldung. Bei Auftreten von “ftpd“ kommt eine Meldung mit der Severity “warning“.

procs_ef:: (Befehl ps -ef) Die Suchmuster können als reg. Expressions angegeben werden. Sie beziehen sich auf die Programmnamen und zusätzlich die Liste der Optionen bzw. Argumente, wie sie der Ausgabe des Befehls “ps -ef“ entsprechen.

In Beispiel 2 soll auch erkannt werden, wenn der Prozess “oracleapp...LOCAL=YES“ mehr als fünfmal in der Prozessliste auftaucht. Die Meldung hat die Severity “minor“. Prozesse, die dem Muster “ora_.*_app“ müssen genau viermal in der Prozessliste vorkommen, sonst gibt es eine Majormeldung.

Wenn die geforderte Anzahl der Instanzen mit der tatsächlichen nicht übereinstimmt, ist die defaultmässige Severity “major“. Wenn der Prozess nicht in der Prozessliste existiert, ist die defaultmässige Severity “critical“.

Beispiel 3:

```
destination::NEPTUN
portno::55555
attribute::OS Unix
procs::inetd snmpd
swap::90 98[crit] # prozentuale Belegung swap:90%->warning,98%->critical
load::20[maj] 50[crit] # run queue bzw. load average
cpu::95[maj] 99[crit] 120
#Mittelwert über 120 Minuten; 1. Schwellwert 95%, 2. Schwellwert 99%
#cpu::95[maj] 99[crit] # alternativ: momentane Belastung, nicht gemittelt
defuncts::100[maj] 200[crit] # Auftreten Zombies; 1. Schwellwert 100, 2. Schwellwert 200
#defuncts::200[crit] # alternativ: nur ein Schwellwert für Zombies
lifecheck:: # lifecheck on
#end of configuration file
```

Zusätzlich geschieht die Überwachung der Performance und der Lifecheck ist eingeschaltet. Lifecheck bedeutet, dass bei jedem Aufruf eine Steuerinformation zur Management-Station geschickt wird. Bei Ausbleiben dieser Information erfolgt eine entsprechende Signalisierung auf der Management-Station.

Beispiel 4:

```
destination::NEPTUN
portno::55555
attribute::OS Solaris
lifecheck::
syslog::10[maj]::/var/adm/messages:“unbedeutende Zeile“-null
“suncluster is down;;Suncluster is down: %4“-crit[3] “clustermeldung xyz“-min[-3]
“warning/i”-warn “error|fatal/i”-maj #Schleppnetz
#syslog::… weitere Protokolldateien
# end of configuration file
```

Es wird zusätzlich die Syslog-Datei von Solaris (Dateiname:/var/adm/messages) ausgewertet. Bei Auftreten von “unbedeutende Zeile“ gibt es keine Meldung, diese wird unterdrückt. Der Eintrag “suncluster is down“ kommt als kritische Meldung maximal dreimal. Die gefundene Zeile wird durch den Formatstring nach “;;“ um vier Spalten nach links verschoben und der Text “Suncluster is down: “ vorangestellt.

Der Eintrag „clustermeldung xyz“ muss mindestens viermal auftreten, damit er als Minor-Meldung signalisiert wird, dann aber nur einmal. Die letzte Zeile (“warning/i“) bewirkt, dass unverhoffte Zeilen, in denen verdächtige Einträge wie “warning“, “error“ oder “fatal“ auftreten, erkannt werden, auch wenn ein entsprechendes Ereignis unbekannt bzw. noch nie passiert ist.

Der Eintrag nach syslog::10[maj] bewirkt, dass die Größe der Syslogdatei überwacht wird. Bei Überschreiten von 10 MB gibt es eine Major-Meldung.

Die Liste der Filter kann beliebig lang sein. Wenn ein Filter trifft, wird der Listendurchlauf beendet. Es kommt also auf die Reihenfolge der Einträge an. Gefundene Zeilen aus der Syslogdatei werden mit dem vorangestellten Prompt “SyslogEntry:: “ zur Management-Station geschickt, der zugehörige Event-Typ ist “Syslog“. Findet eine Ersetzung durch einen Formatstring statt, ist der Prompt “SyslogEntryFmt:: “, der Event-Typ ist “SyslogFormat“.

Für die Spezifikation des Dateinamens kann man Metazeichen ’*’, ’?’ (Suchmuster für Dateinamen) verwenden. Es werden dann alle Dateien, die dem Suchmuster entsprechen, überwacht. Das gilt auch für Dateien die im laufenden Betrieb neu angelegt werden.

Beispiel 5:

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

Durch das Schlüsselwort restart::, gefolgt von dem Namen eines Prozesses entsprechend “ps -ef“, wird die Existenz des Prozesses in der Prozessliste überprüft. Ist der angegebene Prozess nicht aktiv, wird die Restartprozedur „startlpsched“ aufgerufen. Für die Spezifikation des Prozessnamens kann man reguläre Ausdrücke verwenden. Der Vorgang wird höchstens drei Mal wiederholt, wenn das Neustarten erfolglos war.

Bei mehreren nachzustartenden Prozessen kann es mehr als eine Zeile dieser Art geben.

Das Schlüsselwort listenports:: mit nachfolgenden Portnummern bewirkt die Prüfung der Ports tcp4 gegen das Loopback-Interface. Das Schlüsselwort listenports6:: gilt für ipv6.

Mit dem Schlüsselwort cmd:: vereinbart man ein Programm/Skript, dessen Ausgabe ausgewertet wird. In dem Beispiel handelt es sich um den netstat-Befehl, mit dessen Hilfe horchende udp-Ports überprüft werden. Man beachte die Negativ-Filter und die Formatstrings für die Auswertung.

4) Scheduling

Wenn in der Konfigurationsdatei das Schlüsselwort “pollingsecs::“ fehlt, kann man das Programm durch die crontab periodisch aufrufen.

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

Wenn man den Agenten als Hintergrundprozess betreibt, ist ein Startskript für das Anstarten beim Hochfahren des Betriebssystems nötig. Die einfachste Art ist die Verwendung des Startskriptes “rc.local“, das es auf vielen Unix-Derivaten unter “/etc“ gibt. Darin kann man eine Zeile der folgenden Art eintragen:

```
su - monitor -c /home/monitor/basemonagent
logger “Agent fuer Standardueberwachung gestartet“
```

Beim Booten wird das Agenten-Programm “basemonagent“ gestartet, das dem Benutzer “monitor“ gehört. Der Vorgang wird an der Management-Station signalisiert.

Der Agent wird beendet durch das Signal SIGTERM oder SIGINT, was ebenfalls eine Meldung an der Management-Station hervorruft.

Der kleinste Wert für pollingsecs:: beträgt 60 (60 Sekunden, 1 Minute).

Wichtig: Nach Änderung der Konfigurationsdatei, egal ob mit Editor oder von der Management-Station aus, liest der Agent die neue Konfiguration automatisch ein. Man braucht den Agenten also nicht stoppen und dann wieder starten.

4. Programm “logmonagent“

Agentenprogramm zur Logfileauswertung. Mit einer Konfigurationsdatei lassen sich mehrere Protokolldateien auswerten. Das Programm kann man mit verschiedenen Konfigurationsdateien versehen.

1) Aufruf

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

Wird mit der Option “-c“ keine Konfigurationsdatei angegeben, nimmt das System den Defaultnamen “logmonagent.conf“. Ist diese nicht vorhanden oder nicht lesbar, erfolgt eine Fehlermeldung auf *stdout*. Programm und Konfigurationsdatei müssen im gleichen Verzeichnis sein.

Gefundene Zeilen einer Logdatei werden mit vorangestelltem “LogfileEntry:: “ bzw. “LogfileEntyFmt:: “ bei einer Formatangabe zur Management-Station geschickt. Der Meldungstext besteht aus dem Inhalt der gefundenen Zeile, an der der Name der Logdatei in eckigen Klammern angehängt wird. Dies ist die letzte Spalte einer jeden gefundenen Zeile. Dadurch lässt sich zusätzlich die Meldungsquelle bestimmen. Für die Erfassung und Übertragung einer Zeile stehen 1024 Zeichen zur Verfügung.

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

- logfile:: Modus „Zuwachs seit dem letzten Aufruf auswerten“
- incremental:: identisch mit logfile::
- total:: Modus „ganze Datei auswerten, wenn sie sich geändert hat“

Die Zeile(n) für die Logdatei(en) haben die allgemeine Form:

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

oder

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

Nach dem Schlüsselwort logfile:: oder incremental:: folgt optional ein logischer Name (Label) und ebenfalls optional eine Zahl als Schwellwert für die Größe der Zieldatei in Megabyte (MB). Der logische Name kann bis zu 30 Zeichen lang sein und muss mit einem Buchstaben beginnen. Dann kommt der volle

Pfadname der Logdatei. Ein vorangestelltes, optionales Ausrufungszeichen ’!’ bewirkt, dass nicht gemeldet wird, wenn die Zieldatei nicht existieren sollte. Danach folgt die Liste der Filter.

Bei der inkrementellen Logfileauswertung legt das Zeichen ’-’ unmittelbar vor dem Dateinamen die Startbedingung fest. Es unterdrückt eine Signalisierung bei dem ersten Aufruf der Funktion. Ohne diese Option wird bei dem ersten Lauf die Anzahl der Zeilen, die einem Suchmuster entsprechen, angezeigt. Danach werden bei gefundenen Einträgen die zwischen zwei Aufrufen neu hinzugekommenen Zeilen übertragen und angezeigt.

Bei den Schlüsselwort total:: werden ohne das Zeichen ’-’ gefundene Zeilen beim ersten Lauf angezeigt, egal wie alt diese Einträge sind. Mit der Option ’-’ werden gefundene Zeilen erst dann angezeigt, wenn **nach** dem ersten Lauf die Datei sich verändert oder erneuert hat.

Die Dateinamen können in beiden Fällen in ihrem Basisteil Suchmuster für Dateinamen enthalten (’*’, ’?’, “[...]“). Es werden dann alle Dateien in dem Verzeichnis mit dem Namen, die dem Suchmuster entsprechen, überwacht. Es ist zu beachten, dass die Dateinamen, die durch ein Suchmuster ermittelt wurden, und Dateinamen ohne Suchmuster getrennt verwaltet werden. Der Name einer Protokolldatei oder ein Suchmuster dafür dürfen pro Art der Auswertung (incremental, total) nicht mehr als einmal in einer Konfigurationsdatei auftauchen! Es ist aber möglich, zwei verschiedene Muster für dieselbe Datei zu verwenden.

Mit der anschließenden Liste der Filter findet die eigentliche, inhaltliche Auswertung der Protokolldatei statt. Die Liste wird für jede Zeile sequentiell durchlaufen; wenn das jeweilige Suchmuster mit einer Zeile übereinstimmt, wird diese übertragen und als Event-Text dargestellt und der Durchlauf terminiert. Es kommt somit auf die Reihenfolge der Filter an. Der Listendurchlauf endet auch, wenn es sich um einen Suppression-Filter handelt.

Achtung: Das Suchmuster “*“ des Filters hat die Sonderbedeutung “alles“.

3) **Beispiele:**

Beispiel 1:

Es soll das Online-Logfile von DBMS Informix überwacht werden:

```
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 # erfolgreiches Backup
unterdrücken
"Logical [Ll]og.*Backup"-maj #  Backup nicht erfolgreich, Reihenfolge der Filter!
"Assert Failed"-maj "Lock table overflow"-maj
"log files are almost full"-maj
"no more extents"-crit # Ressourcenmangel
"(password is not correct|Incorrect password)"-warn
"Read error"-warn "25580.*System error"-null
"System error"-warn "[Oo]verflow"-warn
"Error|ERROR|error|errstr"-maj "inconsistencies"-maj
# Weitere Logfiles
# logfile::...
# total::...
# end of configuration file
```

Der kleinste Wert für pollingsecs:: ist 2 (2 Sekunden).

Der Name der Logdatei ist „/home/informix/online.log“, sie soll inkrementell ausgewertet werden. Die Management-Station ist NEPTUN. Der Übertragungsport ist 55555. Es ist nur ein Attribut - “Informix“ für Gruppe - angegeben. Als “Object“ im Event Browser erscheint dann der Basename der Logdatei.

Beispiel 2:

Es soll die Logdatei “/oracle/PRX/saptrace/background/alert_PRX.log“ für die Anwendung PROJECTX überwacht werden.

```
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  # evt. Verbucherfehler
“ORA-16014”-crit “ORA-165[34]”-crit
“ORA-“-maj  # alle unbekannten ORA-Einträge haben Severity major
“error|fatal||fail/i“-maj  # Schleppnetz für verdächtige Einträge
“[sS]hutting down instance;;Achtung Instanz wird heruntergefahren: $*“-warn
# nach ;; Formatanweisung (Formatstring)
# end of configuration file
```

Der Eintrag nach dem Schlüsselwort “logfile::“ 512[crit] bewirkt, dass auch die (anwachsende) Größe der Logdatei überwacht wird. Bei Überschreitung von 512 MB erfolgt eine kritische Meldung.

Achtung: Wenn das Logfile nicht existiert oder nicht lesbar ist, erfolgt eine Systemfehlermeldung an der Management-Station. Das lässt sich unterdrücken, indem man dem Dateinamen das Zeichen ’!’ voranstellt.

Beispiel 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
```

Es erfolgt eine Meldung, wenn die Datei “/etc/hosts.equiv“ angelegt wird oder schon existiert. Das Ausrufungszeichen vor dem Dateinamen sorgt dafür, dass es keine Meldung gibt, wenn die Datei nicht existiert.

5. Programm “logdiragent“

Inkrementelle Auswertung von Protokolldateien in Verzeichnissen. Das Programm wertet den Zuwachs der Dateien zwischen zwei Aufrufen aus. Man kann mehrere Verzeichnisse und eine Liste von Suchmustern für Dateinamen angeben. Die Suchmuster für Dateinamen müssen fnmatch() entsprechen.

1) Aufruf

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

Der Defaultname für die Konfigurationsdatei lautet: logdiragent.conf. Die Konfigurationsdatei muss im gleichen Verzeichnis liegen wie das Programm. Mit der Option “-c“ <configfilename> kann man einen anderen Name wählen. Die Option “-n“ bewirkt, dass der Agent auch dann signalisiert, wenn zwischen zwei Aufrufen nicht die Größe aber dafür das Datum der Zieldatei sich erneuert hat.

Gefundene Zeilen einer Logdatei werden mit vorangestelltem “LogfileEntry-Dir:“ zur Anzeige geschickt. Der Meldungstext besteht aus dem Inhalt der gefundenen Zeile, an der der Name der Logdatei in eckigen Klammern angehängt ist. Dadurch läßt sich zusätzlich die Meldungsquelle bestimmen.

Der kleinste Wert für pollingsecs:: ist 10 (10 Sekunden).

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

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) **Beispiele:**

Beispiel 1:

```
portno::55555
destination::192.168.178.20
attribute::OS Logfiles
pollingsecs::20
# Wenn die zweite Spalte fehlt, erscheint als „Object“ der Basename des Logfiles
files::10[maj]::*.log *.error # Spezifikation der Dateinamen, entsprechend Suchmuster
logfiledir::/var/adm/mail /var/adm/syslog # Spezifikation der Verzeichnisse
filter::”warning/i”-warn[3] ”error/i”-maj “fatal/i”-crit
# Liste der Filter ist allen Dateien gemeinsam
#end of configuration file
```

Ausgewertet werden die Verzeichnisse “/var/adm/mail” und “/var/adm/syslog”. Genommen werden alle Dateien, die dem Muster “*.log” und “*.error” entsprechen. Ist eine der Dateien größer als 10 MB, gibt es eine Meldung mit der Severity “major”.

Achtung: Wenn ein zu erfassendes Logfile nicht lesend zu öffnen ist, erfolgt eine Systemfehlermeldung an der Management-Station.

Beispiel 2:

Es sollen die Trace-Dateien im Workdirectory von SAP/R3 überwacht werden.

```
portno::55555
attribute::SAP/R3
destination::NEPTUN
pollingsecs::15
files::4[crit]::dev_w? dev_w?? dev_rfc? dev_rfc?? dev_disp dev_?? stderr*
# Spezifikation der Dateien im Verzeichnis; Suchmuster entsprechend fnmatch()
logfiledir::/usr/sap/PC3/DVEBMGS11/work  # Workdirectory mit Logfiles
 filter::"update deactivated"-crit "vb error"-crit # Verbucherabbruch
"^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 # Jobabbrueche
"Fehler in einer ABAP-Anweisung"-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] # Schleppnetz mit unbekannten Zeilen “FATAL, fatal …”
# end of configuration file
```

Die Liste der Filter bezieht sich auf alle zu erfassenden Protokolldateien.

6. Programm “logrecagent“

Inkrementelle Auswertung von Protokolldateien in Verzeichnissen. Das Programm wertet den Zuwachs der Dateien zwischen zwei Aufrufen aus. Man kann mehrere Verzeichnisse und mehrere Suchmuster für Dateinamen eingeben. Es werden auch Unterverzeichnisse der angegebenen Verzeichnisse berücksichtigt.

1) Aufruf

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

Der Defaultname für die Konfigurationsdatei lautet: logrecagent.conf. Die Konfigurationsdatei muss im gleichen Verzeichnis liegen wie das Programm. Mit der Option “-c“ <configfilename> kann man einen anderen Name wählen. Die Option “-n“ bewirkt, dass der Agent auch dann signalisiert, wenn zwischen zwei Aufrufen nicht die Größe aber dafür das Datum der Zieldatei sich erneuert hat.

Gefundene Zeilen einer Logdatei werden mit vorangestelltem “LogfileEntryDirRec: “ zur Anzeige geschickt. Der Meldungstext besteht aus dem Inhalt der gefundenen Zeile, an der der Name der Logdatei in eckigen Klammern angehängt ist. Dadurch läßt sich zusätzlich die Meldungsquelle bestimmen.

Der kleinste Wert für pollingsecs:: ist 30 (30 Sekunden).

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

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) Beispiel:

```
portno::55555
destination::X8PSR Y8PSS # Y8PSS optional Ausweichserver
attribute::OS Logfiles # Wenn die zweite Spalte fehlt, erscheint als „Object“ der Basename des Logfiles
files::100[crit]::*.log *.error # Spezifikation der Dateinamen
logfiledir::/var/adm # Spezifikation des/der Verzeichnisse
filter::”warning/i”-warn[3] ”error/i”-maj
“fatal/i”-crit # Liste der Filter gemeinsam für alle Dateien
```

Ausgewertet wird das Verzeichnis “/var/adm” mit den darunterliegenden Unterverzeichnissen. Genommen werden alle Dateien, die dem Muster “*.log” und “*.error” entsprechen. Ist eine der Dateien größer als 100 MB, gibt es eine Meldung mit der Severity “critical”.

Achtung: Wenn ein zu erfassendes Logfile nicht lesend zu öffnen ist, erfolgt eine Systemfehlermeldung an der Management-Station.

7. Programm “scriptmonagent“

Das Agentenprogramm ruft ausführbare Programme auf, filtert deren Textausgabe und schickt das Ergebnis bei Bedarf zur Management-Station. Mit einer Konfigurationsdatei lassen sich mehrere Überwachungsprogramme definieren. Das Programm kann mit verschiedenen Konfigurationsdateien parametrisiert werden. Es kann als Batch-Programm oder als Hintergrundprozess betrieben werden.

Ein Programm zur Überwachung wird intern durch den Systemaufruf *fork()* gefolgt von der Funktion *execvp()* ausgeführt. Das ist der Normalfall. Wenn die Deklaration des Programms jedoch Sonderzeichen der *Shell* wie z.B. ’|’ oder ’;’ enthält, wird *execl()* in Verbindung mit “/bin/sh -c“ benutzt. Auf diese Weise ist eine Weiterverarbeitung der Ausgabe mit der Pipe ’|’ und eventuell *awk* oder *perl* möglich. Diese Regelung gilt auch für die Agenten basemonagent und asyncmonagent!

Die in der Konfigurationsdatei definierten Programme werden periodisch aufgerufen; die Ausführung eines jeden Programms erfolgt **parallel** in einem eigenen *Thread*. Es gibt für jedes Programm eine einstellbare Auszeit (*Timeout*). Bei Überschreitung wird das Programm mit SIGTERM gestoppt und es folgt eine kritische Meldung an der Management-Station. **WICHTIG**: Die Ausgabe erfolgt unmittelbar nach Beendigung des Programms aber nicht währende der Programmlaufzeit. Ein Programm in einem *Thread* kann nur dann neu aufgerufen werden, wenn der vorherige Programmlauf beendet ist. Die Laufzeit kann größer sein als das Polling Intervall.

1) Aufruf

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

Wird mit der Option “-c“ keine Konfigurationsdatei angegeben, nimmt das System den Defaultnamen “scriptmonagent.conf“. Programm und Konfigurationsdatei müssen im gleichen Verzeichnis sein!

Der kleinste Wert für pollingsecs:: ist 2 (zwei Sekunden).

Die Path-Variable “PATH“ ist gesetzt auf:

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

2) Konfigurationsdatei

Es gibt das Schlüsselwort:

- cmd:: Vereinbarung eines ausführbaren Programms mit Name und Filtern für die Ausgabe

Die Zeile(n) für die Skripte haben die Form:

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

tos: Timeout in Sekunden, default: 30 (> 0)
name: logischer Name (Label) bis zu 30 Zeichen lang

Das Programm muss eine Ausgabe nach *stdout* und/oder *stderr* haben. Es ist zu beachten, dass bei einem Aufruf des Überwachungsprogramms mehr als eine Meldung erzeugt werden kann, wenn die Ausgabe mehr als eine Zeile hat, die jeweils dem Suchmuster eines der Filter entspricht.

Die Auswertung geschieht durch die Filterliste, deren Elemente durch mindestens ein Leerzeichen getrennt sind, und bei Bedarf durch zusätzliche Kommandos (“grep“, “awk“) und Kommandoverkettung mit der Pipe ’|’. Bei komplexeren Auswertungen, zum Beispiel für Datenbankverwaltungssysteme wie Oracle oder Informix, sollte das Programm in eine eigene Shell-Prozedur ausgelagert werden, deren Namen dann vereinbart wird.

Aus der Befehlsrückgabe gefilterte Zeilen ohne Formatangabe werden mit vorangestelltem “ScriptMonitor:: “ zur Management-Station geschickt, mit Formatangabe ist der Prompt “ScriptMonitorFmt:: “.

3) **Beispiel:**

Es soll die Hauptspeicherbelegung eines Linux-Servers mit Hilfe eines Kommandos überwacht werden. Zur Erleichterung der numerischen Operationen wird der Reportgenerator “awk“ genommen. In dem Awk-Programm sind zwei Schwellwerte für den Belegungsgrad enthalten, die in der anschließenden Filterliste zwei Filter mit verschiedener Severity bedienen.

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

# Analyse der aktuellen Netzwerk Verbindungen (tcp/udp, Linux)
cmd::NETSTAT-TUNP::netstat -tunp::”:[0-9]{1,5}[ ]+(127.0.0\.1|\::1):”-null
“:[0-9]{1,5}[ ]+192\.168\.”-null # unterdrueckt eigene adressen
“:[0-9]{1,5}[ ].*:[0-9]{1,5}[ ];;Unbekannte foreign adr: $5/$1 PID: $$”-warn[+8] # zeigt unbekannt

# Analyse von 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;;Unbekannter Port: $4/$1 PID: $$”-maj

# Kommando-Auswertung, Programme werden parallel ausgefuehrt!
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 "*;;Systemfehler: $*"-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
# weitere Zeilen mit Programmen und Filtern …
# cmd::.....
# end of configuration file
```

Der Übertragungsport ist 55555. Es gibt die Attribute “Linux“ für “Group“ und “Performance-Extra“ für “Object“ im Event Browser. Wenn das zweite Attribut für “Object“ fehlt, wird der Basename des jeweiligen Programmnamens genommen.

Achtung: Das Überwachungsprogramm muss ausführbar sein und einen Returncode gleich Null liefern. Sonst gibt es eine gesonderte Systemfehlermeldung an der Management-Station unter Angabe des Returncodes.

8. Programm “asyncmonagent"

Das Programm stellt eine Erweiterung der bisherigen Agenten basemonagent, scriptmonagent und logmonagent dar. Jedes in der Konfigurationsdatei vereinbarte Skript bzw. jede Protokolldatei kann mit einem eigenen Polling Intervall versehen werden oder als Cronjob definiert werden. Die Ausführung der einzelnen Überwachungsfunktionen geschieht nebenläufig in Form von Threads. Der Agent wird ausschließlich als Hintergrundprozess betrieben!

Die periodische Ausführung einer Funktion ist so gestaltet, dass ein Aufruf in einem Thread nur dann erfolgen kann, wenn der vorherige beendet worden ist, unabhängig von den anderen Threads. Es besteht nicht die Gefahr von sich überschneidenden Aufrufen, die auftritt, wenn Skripte ihre Laufzeit ändern oder gar blockieren (“duplicate cron jobs“). Im Gegensatz zu den anderen Agenten gibt es hier standardmäßig für die Ausführung von Skripten und Programmen **keine** Auszeit (Timeout). Dadurch kann man auch spezielle Überwachungsprogramme einsetzen, die entweder gar nicht oder nur nach einer unbestimmten Zeit terminieren (z.B. journalctl -f, tcpdump, nmap, snmpdelta, mpstat, sar). Deren Ausgabe wird nach der Filterung während des Programmlaufs in Echtzeit zur Management-Station geschickt.

Mit einer Option (siehe unten) kann man dieses Verhalten ändern, so dass das Ergebnis erst **nach** Abarbeiten des Überwachungsprogramms und bei Vorliegen aller Daten geliefert wird. In diesem Fall wird die Gesamtzahl der gefundenen Einträge pro Filter mit übertragen. Die gesamte Laufzeit wird durch eine Auszeit (*Timeout*) begrenzt. Bei Überschreitung wird die Funktion abgebrochen und es gibt eine Fehlermeldung.

1) Aufruf

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

Wird mit der Option “-c“ keine Konfigurationsdatei angegeben, nimmt das System den Defaultnamen “asyncmonagent.conf“. Programm und Konfigurationsdatei müssen im gleichen Verzeichnis liegen!

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

- cmd:: Vereinbarung eines ausführbaren Programms mit Name und Filtern für die Textausgabe
- logfile:: Modus „Zuwachs seit dem letzten Aufruf auswerten“
- incremental:: identisch mit logfile::

- total:: Modus „ganze Datei auswerten, wenn sie sich geändert hat“
- lifecheck:: Dynamische Registrierung an der Management-Station

Zeile mit lifecheck::

lifecheck::[<anzahl_sekunden>]

Der kleinste Wert für <anzahl sekunden> ist 30. Wenn die Angabe für <anzahl_sekunden> fehlt, ist die Voreinstellung 300. Es ist zu beachten, dass sich der Gebrauch von lifecheck:: von diesem Agenten und vom Agenten base-monagent gegenseitig ausschließt; es kann zur gleichen Zeit mit dem gleichen Rechnernamen nur der eine oder der andere benutzt werden. Das lässt sich vermeiden, indem man mit dem Schlüsselwort nodename:: einen Alias-Namen für den Server vergibt.

8.1 Individuelle Polling Intervalle

Die Zeilen für die Skripte haben die Form:

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

Die Option mit dem Sonderzeichen ‘%‘ vor <programmname> bedeutet, dass es eine Auszeit (Timeout) von 30 Sekunden gibt. Ohne diesen Zusatz gibt es **kein** Timeout.

Die Zeilen für die Logfileauswertung haben die Form:

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

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

Die optionale Angabe <name> bedeutet den logischen Namen einer Instanz, wenn man ein und dieselbe Protokolldatei mehr als einmal mit verschiedenen Parametern auswerten will. Der Name kann bis zu 30 Zeichen lang sein, muss mit einem Buchstaben beginnen und muss eindeutig sein.

Der Wert für <polling interval> kann in folgender Form angegeben werden:

- Eine Zahl als die Anzahl von Sekunden, z.B. “90“ (90 Sekunden)
- <minuten>.<sekunden>: Anzahl Minuten und Anzahl Sekunden getrennt durch einen Punkt ’.’, z.B. “10.0“ (10 Minuten + Null Sekunden)
- <stunden>.<minuten>.<sekunden>: Anzahl Stunden, Anzahl Minuten, Anzahl Sekunden getrennt durch Punkte, z.B. “1.30.0“ (1 Stunde + 30 Minuten + Null Sekunden)

Der kleinste Wert für <polling interval> beträgt 2 Sekunden.

8.2 Ausführung als Cronjob (Scheduler)

Hiermit lassen sich Funktionen zu diskreten Zeitpunkten aufrufen.

Die Zeilen für die Skripte haben die Form:

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

Die Option mit dem Sonderzeichen ‘%‘ vor <programmname> bedeutet, dass es eine Auszeit (Timeout) von 30 Sekunden gibt. Ohne diesen Zusatz gibt es **kein** Timeout.

Die Zeilen für die Logfileauswertung haben die Form:

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

Die Angaben für <crontabspec> orientiert sich an der Einrichtung crontab() von Unix. Für die Zeitangaben eines Cronjobs gibt es die folgenden Zeichen:

- Eine Zahl für Minute [0-59], Stunde [0-23], Tag des Monats [1-31], Monat [1-12] und Wochentag [0-7]; 0 oder 7 ist Sonntag; symbolische Namen sind nicht erlaubt
- Das Zeichen ’*’ steht für jede Minute, jede Stunde, jeden Tag, jeden Monat und jeden Wochentag
- Kommata ’,’ für mehrere Zeitangaben
- Bindestrich ’-’ für einen Zeitraum
- Schrägstrich ’/’ für eine Periode
- Leerzeichen (Blank) ’ ’ zum Trennen der Zeitfelder

Beispiele für <crontabspec>:

“* * * * *“: Aufruf jede Minute
“*/5 * * * *“: Aufruf alle fünf Minuten
“15,45 10,11,12 * * *“: Aufruf 10, 11, 12 Uhr jeweils 15 und 45 Minuten
“*/10 6-18 * * 1-5“: Aufruf alle 10 Minuten von 6-19 Uhr von Mo bis Fr

Für nicht terminierende Programme (Daemons) ist folgendes zu beachten: Der Agent startet das Programm am Ende des ersten Polling Intervalls (z.B. nach 10 Sekunden). Das Polling Intervall hat in diesem Fall die Funktion, die zugehörigen Filter und deren Zähler nach jedem Ablauf zurück zu setzen. Bei einem Cronjob ist dieser Wert konstant 60 (1 Minute). Bei Änderung der Konfiguration im laufenden Betrieb wird der Daemon mit SIGTERM beendet und dann wieder gestartet.

Beispiel:

```
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::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 after 3 seconds
cmd::SNMPTRAP::5::snmptrapd -f -Lo -Oq::“[ ]\[UDP:[ ]\n*“-warn[0]
# direkte Auswertung von snmptrapd
```

Das Programm mpstat startet nach 3 Sekunden, läuft eine Stunde und startet nach 3 Sekunden erneut usw.

Ansonsten gilt die Beschreibung für die Agenten logmonagent und scriptmonagent.

Hinweis: Wenn es für Überwachungsprogramme kein Timeout gibt, erfolgt die quantitative Auswertung gefundener Einträge in einer anderen Weise. Wenn die Anzahl der gefundenen Vorkommnisse größer ist als die in dem Filter vereinbarte, gibt es eine gesonderte Meldung mit dem Event-Typ “Frequency“ bei sonst gleichen Attributen. Der Text dieser Meldung zeigt das Suchmuster des Filters und die Anzahl der gefundenen Zeilen an. Dieser Mechanismus lässt sich durch die Angabe des Quantifizierers eines Filters beeinflussen. Man kann eine hohe Zahl oder den Wert “0“ wählen (z.B. “[ ]error[ ]/i“-warn[0]). Der Wert “0“ schaltet den Mechanismus aus.

Diese Überlegungen entfallen, wenn man für Programme mit kurzer Laufzeit die Option ‘%‘ vor dem Programmnamen wählt. Dann erfolgt die Ausgabe erst nach Abschluss des Programms, und die Gesamtzahl der gefundenen Einträge pro Filter wird mit jeder Meldung in binärer Form mit übertragen.

Weitere Beispiele:

```
destination::192.168.178.20
portno::55555
attribute::Linux AsyncMonitorCollection
lifecheck::120 # alle 120 Sekunden heartbeat signal zur Management-Station

suppressionsecs::300
# Wiederholung von identisch gleichen Meldungen für 300 Sekunden unterdrücken

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

# Polling Intervall 20 Sekunden; wenn größer als 5 MB Minor Meldung
logfile::20/5[min]::/var/log/daemon.log::"error/i;;%4"-warn # …

# Polling Intervall 10 Sekunden (max. 3 gefundene Zeilen schicken)
logfile::10::/var/log/messages::"POSSIBLE BREAK-IN ATTEMPT;;%5"-maj[+3]

# Polling Intervall 5 Minuten (eine Meldung wenn mehr als 100 in 5 Minuten)
logfile::ssh-login::5.0::/var/log/messages::"sshd.*[Ii]nvalid user"-maj[>100]
"Failed keyboard-interactive"-maj[>100]

# Polling Intervall 1 Stunde; Auswertung der ganzen Datei bei Änderung
total::1.0.0::!/var/log/boot.log::"error/i"-warn # …

# Analyse der aktuellen Netzwerk Verbindungen (tcp/udp, Linux),  Polling Intervall 4 Sekunden
cmd::NETSTAT-TUNP::4::%netstat -tunp::":[0-9]{1,5}[ ]+(127.0.0\.1|\::1):"-null
":[0-9]{1,5}[ ]+192\.168\."-null # unterdrueckt eigene adressen
":[0-9]{1,5}[ ].*:[0-9]{1,5}[ ];;Unbekannte foreign adr: $5/$1 PID: $$"-warn[8] # zeigt unbekannt

# CRONJOB, jeden Tag von Mo - Fr, 10.00
cmd::0 10 * * 1-5::echo "Es ist 10 Uhr (Wochentag): `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
# … weitere Protokolldateien, Befehle, Cronjobs
```

9. Programm “secmonagent“

Agent zum Zweck der Security. Das Programm überwacht Systemdateien und/ oder Systemverzeichnisse und andere Verzeichnisse mit den in ihnen enthaltenen Dateien bezüglich der Änderung ihrer Attribute. Es wird auch ermittelt und gemeldet, wenn Dateien in einem Verzeichnis neu hinzugekommen sind! Die untersuchten Attribute sind: *inode number*, *size*, *modification time*, *mode/permission*, *status time*, *user id of owner*, *group id of owner*.

1) Aufruf

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

Wird mit der Option “-c“ keine Konfigurationsdatei angegeben, nimmt das System den Defaultnamen “secmonagent.conf“. Programm und Konfigurationsdatei müssen im gleichen Verzeichnis liegen! Man kann das Programm mit verschiedenen Konfigurationsdateien versehen. Es kann als Batch-Programm oder als Hintergrundprozess (daemon) betrieben werden.

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

- files:: Eine Liste von Verzeichnissen und/oder Dateien mit voller Pfadangabe, die optional mit einer Severity versehen werden können (default: “warning“)
- exclude:: Eine Liste von Dateien und/oder Unterverzeichnissen mit voller Pfadangabe, die von der Überwachung ausgeschlossen werden sollen

Die Namen können in Anführungszeichen ’“’ gesetzt werden, wenn in ihnen Leerzeichen vorkommen. Wenn hinter dem Namen eines Verzeichnisses ein Schrägstrich ’/’ kommt, wird nur der Inhalt des Verzeichnisses überwacht, nicht aber Änderungen des Verzeichnis selbst.

Mit Hilfe des Agenten lassen sich Installationsvorgänge, gewollte und nicht gewollte (z.B. ***rootkits***), feststellen. Bei gewollten Installationen kann man Vergleiche zwischen angekündigten und tatsächlichen Änderungen anstellen. Wenn das Programm eine oder mehrere Änderungen festgestellt hat, gibt es pro Verzeichnis eine Meldung mit den Namen der Dateien und der Art der Änderung.

Beispiel 1:

```
destination::192.168.178.20
portno::55555
attribute::Debian Security
pollingsecs::30 # Betrieb als Daemon, Auswertung alle 30 Sekunden

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[maj] # ...
exclude::/etc/mtab /etc/resolv /etc/adjtime # ...
```

Es werden die aufgeführten Verzeichnisse auf Änderungen geprüft. Die Dateien "/etc/mtab", "/etc/resolv" und "/etc/adjtime" werden von der Überwachung ausgenommen.

Der kleinste Wert für pollingsecs:: beträgt 5 (5 Sekunden)! Die Meldungen von dem Agenten haben den Event-Typ "Security".

Beispiel 2: Unterschiedliche Severities für Dateien in einem Verzeichnis

```
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] # ...
```

Die Dateien "/bin/sh", "/bin/ls", "/bin/ps" und "/bin/netstat" erzeugen bei Änderung eine Meldung mit der Severity "critical", weil sie **vor** dem Verzeichnis "/bin" stehen und keine Verzeichnisse sind. Alle anderen Dateien im Verzeichnis "/bin" erzeugen bei Änderung eine Meldung mit der Severity "major".

10. Programm “rsendmsg“

Das Programm schickt Meldungen direkt zur Management-Station. Dort kann man sie bei Bedarf filtern oder umsetzen. In Scripten verwendet kann man beliebige (Überwachungs-)Funktionen realisieren.

Aufruf:

rsendmsg [-p <port>] [-d <server>] [-e <ausweichserver>] -g <gruppe> -o <object> [-n <nodename>] [-c <charset>] [-6] -s <severity> -m <meldungstext>

Parameter:

- -p: Portnummer tcp; optional wenn Eintrag in “monitorkeys.sig”
- -d: Managementserver; optional wenn Eintrag in “monitorkeys.sig”
- -e: Ausweichserver; optional wenn Eintrag in “monitorkeys.sig”
- -g: Meldungsgruppe
- -o: Object
- -n: Nodename (optional)
- -s: Severity [inform|minor|warning|major|critical]
- -m: Meldungstext/Event-Text
- -c: Zeichensatz (*character set*): UTF-8|ISO-8859-1..ISO-8859-10, ISO-8859-13..ISO-8859-16
- -6: nur ipv6

Returncode:

0: erfolgreich
1: Eingabefehler
2: Kommunikationsfehler

11. Programm “remoteconfd”

Hintergrundprozess zur zentralen Bearbeitung der Konfigurationsdateien von der Management-Station aus. Die über das Netz gehenden Daten werden nach der Methode AES mit Blockverkettungsmodus (CBC) verschlüsselt. Die Verschlüsselung erfolgt dynamisch, ein und derselbe Klartext wird nachfolgend immer verschieden chiffriert. Der Socket-Typ für die Konfigurationsdateien ist udp.

Aufruf:

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

Parameter:

- -p: Portnummer udp; optional wenn Eintrag in “monitorkeys.sig”
- -i: IP-Adresse oder Name der Management-Station als Client; optional wenn Eintrag in “monitorkeys.sig”
- -j: Adresse oder Name der Ausweichstation als Client; optional wenn Eintrag in “monitorkeys.sig”
- -a: Dateiname für die Vereinbarung einer Zeichenkette zur Authentifizierung. Der String darf nicht kürzer als acht und nicht länger als 30 Zeichen sein
- -F: root-fs für Beschränkung des Zugriffs auf Dateien, die in diesem Verzeichnis oder Unterverzeichnis liegen
- -r: Abschalten Kommando-Interface; es können keine Kommandos von den Clients ausgeführt werden
- -L: Name der Protokolldatei (default: remoteconfd.logfile)
- -m: Port für Meldung (Start/Stop) zur Management-Station (tcp); fehlt die Option, wird der unter -p angegebene Port genommen
- -g: Meldingsgruppe für Start/Stop-Meldungen (default: ADMIN_)
- -4: nur ipv4
- -f: Kein fork() nach Aufruf

Die Optionen “-i“ und “-j“ mit den Hostnamen oder IP-Adressen sorgen dafür, dass Anfragen nur von diesen Clients gemacht werden können. Fehlen die Optionen, können von überall Anfragen gemacht werden.

Der Default-Wert für die Option “-a“ ist “remoteconfd.auth“ im gleichen Verzeichnis.

12. Programm “winmonagent.exe“

Das Programm liefert die Standardüberwachung für Windows. Dieses und die nachfolgenden Programme verwenden für die Übertragung zur Management-Station die Zeichensätze (*character sets)* CP1250 bis CP1258. Wenn auf dem Server andere Zeichensätze eingestellt sind erfolgt eine Konvertierung nach UTF-8.

Es wird ermittelt:

- Auslastung aller lokalen Drives
- Existenz von Tasks in der Taskliste
- Existenz von Services
- Eventlogs nach Eventid’s (System, Application, Security)
- Auswertung der Textausgabe von einem oder mehreren Überwachungsskripten
- Listenports
- CPU-Belastung aller CPU’s
- Schwellwert Memory
- Schwellwert Pagefile
- Auftreten von Reboots
- Lifecheck (Heartbeat)

1) Aufruf

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

2) Konfigurationsdatei

Das Programm "winmonagent.exe" benötigt eine Konfigurationsdatei mit dem Namen "winmonagent.conf" im gleichen Verzeichnis.

Der kleinste Wert für pollingsecs:: beträgt 60 (60 Sekunden, 1 Minute).

Es gibt die Schlüsselwörter:

- drives:: Schwellwerte in Prozent für Windows-Filesysteme
- tasks:: Name und Anzahl Instanzen, die aktiv sein müssen
- services:: Namen von Services, die aktiv sein müssen
- cmd:: Ausführung eines Überwachungsprogrammes und Filterung der Ausgabe
- system:: Überwachung System Event Log

- application:: Überwachung Application Event Log
- security:: Überwachung Security Event Log
- cpu:: zwei Schwellwerte in Prozent für Auslastung aller CPU's
- memory:: zwei Schwellwerte in Prozent für Auslastung Memory
- page:: zwei Schwellwerte in Prozent für Auslastung Pagefile
- listenports:: Liste von Portnummern (tcp), die im Zustand "listen" sein sollen (ipv4)
- listenports6:: dito für ipv6
- lifecheck:: Einschalten Registrierungsmechanismus für die Management-Station

Die Vereinbarung für die Überwachung von Laufwerken hat die folgende Form:

Form 1:

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

Form 2:

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

Die Zahl <mins> hat die Bedeutung Zeit in Minuten für die Wiederholung der Meldung bei Überschreitung des Schwellwertes. Zwischendurch gibt es nur dann eine Meldung, wenn die Veränderung größer als 1 MB ist. Der voreingestellte Wert hierfür ist 120 (2 Stunden). Nach der Option "-d" folgt eine Zahl <prozent> mit der Bedeutung Belegungsgrad in % für **alle** erfassten Drives. Nach der Option "-D" kommt eine **Liste** von Laufwerken, die gemountet sein müssen und spezielle Schwellwerte <prozent> haben können. Nach der Option "-E" kommt eine **Liste** von Drives, die von der Überwachung ausgeschlossen werden sollen. Die Listenelemente werden durch mindestens ein Leerzeichen von einander getrennt. Fehlt die Option "-d", werden nur die unter "-D" angegebenen Laufwerke überwacht. Die Laufwerke werden in der Notation "C:\", "D:\" usw. angegeben.

Fehlt die Angabe für <sev>, ist der Standardwert "minor". Wenn der Belegungsgrad für ein Laufwerk größer als 98% ist, wird die Severity automatisch zu "critical". Dieses Verhalten lässt sich durch die Filter an der Management-Station bei Bedarf ändern. Dort kann man den Schwellwert und/oder die Severity an den Wert des noch verfügbaren Speichers binden, der bei jeder Meldung mitgeliefert wird.

Überwachung von Tasks:

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

Nach dem Schlüsselwort die Eingabe einer Liste von Tasknamen plus eventuell OPTIONEN. Groß/Kleinschreibung ist nicht signifikant, die Name dürfen nicht mit „.exe“ enden. Die Listenelemente werden getrennt durch mindestens ein Leerzeichen.

Form 1:

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

Die angegebenen Tasks müssen mindestens einmal in der Taskliste existieren. Der Default-Wert für sev ist “critical“.

Form 2:

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

Die Anzahl der Instanzen der Tasks muss genau COUNT sein. Bei COUNT gleich Null wird signalisiert, wenn der Task aktiv ist („should **not** be running“)!

Form 3:

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

Wobei MAX > MIN ist. Anzahl der Instanzen einer Task in einem Bereich von MIN bis MAX. Bei MIN == 0 und MAX > 0 darf die Anzahl der Instanzen nicht größer als MAX sein, kann aber auch Null sein.

Form 4:

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

Anzahl der Instanzen darf nicht größer als MAX sein, muss aber mindestens eins sein.

Form 5:

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

Anzahl der Instanzen darf nicht kleiner als MIN sein.

Überwachung von Diensten (Services):

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

Eingabe eine Liste von Service-Namen plus ggf. OPTION, die durch mindestens ein Leerzeichen getrennt sind. Der jeweilige Name ist der „service name“, nicht „display name“. Groß/Kleinschreibung bei den Namen ist nicht signifikant.

Form 1:

services::servicename ...

Es wird gemeldet, wenn ein Dienst **nicht registriert** ist. Die Severity ist “warning“.

Form 2:

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

Es wird mit der Severity <sev> gemeldet, wenn ein Dienst **nicht aktiv** ist.

Form 3:

services::servicename[null] ...

Prüfung auf Anwesenheit. Ist der Dienst **aktiv**, erfolgt eine Meldung mit der Severity “major“.

Überwachung Event-Logs

Die Vereinbarung für die Überwachung der Eventlogs hat die folgende 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> ...

Für die Überwachung der System-, Application-, und Security-Eventlog Eingabe einer **Liste** von Event-Ids plus optional Severity, bei deren Auftreten eine Meldung erfolgen soll. Die Listenelemente werden durch mindestens ein Leerzeichen von einander getrennt.

Durch die optionale Angabe error und/oder warning wird das Auftreten eines Events mit dem Windows-Event-Typ (= Windows-Severity) “error“ bzw. “warning“ gemeldet. Durch die optionale Angabe any wird **jedes** neue Event gemeldet. Einschränkungen hiervon kann man durch die Option “-S“ gefolgt von einer

Liste von zu unterdrückenden Event-Ids vornehmen. Mit der Option any und der Option "-S" lässt sich eine Negativliste von bekannten Event-Ids formulieren, die als unwesentlich eingestuft werden, so dass nur bisher unbekannte Event-Ids gemeldet werden. Eine andere Strategie besteht darin, eine Positivliste mit zu signalisierenden Event-Ids aufzustellen, in Kombination mit error und/oder warning.

Wenn für eine Event-Id die Spezifikation der Severity fehlt, wird die Windows-Severity "ERROR_TYPE" zur Severity "major", "WARNING_TYPE" und "AUDIT_FAILURE" zu "warning" und die übrigen zu "inform".

In einer Meldung ist enthalten der Log-Typ, die Event-Id, der Event-Typ, Source und die Message-Strings, sofern sie verfügbar sind. Diese Daten lassen sich an der Management-Station noch einmal nach inhaltlichen Kriterien filtern.

3) **Beispiele:**

Beispiel 1:

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

Die Management-Station ist NEPTUN. Der Übertragungsport ist 55555/tcp. Es gibt zwei Attribute "Windows" für "Group" und "Standardmonitor" für "Object" im Browser der Management-Station. Der Agent wird als Hintergrundprozess mit einem Polling Intervall von fünf Minuten betrieben.

Der allgemeine Schwellwert für Füllgrad von Filesystemen ist 93%. Bei Überschreitung erfolgt eine Minor-Meldung. Der Schwellwert für Laufwerk D:\ ist 95% und es kommt eine Major-Meldung. Es werden alle Laufwerke erfasst, die Frage ist nur, ob man sie mit individuellen Schwellwerten spezifiziert oder nicht. Wenn ein Laufwerk angegeben ist, aber nicht existiert, erfolgt eine kritische Meldung.

Mit der Angabe -E G:\ X:\ ... kann man Laufwerke ausschließen!

Achtung: Wenn der Füllgrad eines Filesystems größer wird als 98%, gibt es eine kritische Meldung!

Beispiel 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
```

Die Task "jrun" muss mindestens zweimal in der Taskliste existieren, sonst kritische Meldung. Die Task "webscan" muss aktiv sein (egal wie viele Instanzen), sonst kritisch. Die Task "inetinfo" soll genau einmal existieren, sonst Major-Meldung. Die Task "cmd" darf höchstens fünfmal vorkommen, sonst “warning“. Die Anzahl der Instanzen von "ascWrapper" soll zwischen zwei und fünf liegen, sonst Majormeldung.

Die Dienste (Services) “spooler“ und “snmp“ müssen aktiv sein, sonst erfolgt eine Major- bzw. Critical-Meldung. Das Suchargument für einen Dienst ist der *service name*, nicht *display name*! Beim Namen wird Groß/Kleinschreibung nicht unterschieden. Wenn der Dienst “ftpsvc“ aktiv ist, erfolgt eine Meldung mit der Severity “major“.

Die Angaben für Tasks und Services können in Anführungszeichen gesetzt werden, wenn in ihnen Leerzeichen vorkommen.

Achtung: Fehlt bei Diensten die Angabe einer Severity in eckigen Klammern, wird lediglich die Existenz bzw. Registrierung auf dem Server geprüft, nicht aber, ob er aktiv ist. Um das zu prüfen muss eine Severity angegeben werden. Durch die Angabe “null“ in den eckigen Klammern führt man eine umgekehrte Prüfung durch, die meldet, dass ein Service aktiv ist.

Beispiel 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
```

Überwachung der Eventlogs von Windows:

Suchargument ist die Event-Id und/oder der Event-Typ “error“, “warning“. Es kann eine Liste von Event-Ids, die unterdrückt werden sollen, definiert werden.

Für das Security-Eventlog wird Id 529 (falsches Einloggen) und 578 (Herunterfahren des Systems) überwacht. Außerdem alle Events mit der Windows-Severity “ERROR_TYPE“. Beim System-Eventlog werden alle Events mit der Windows-Severity “ERROR_TYPE“ und “WARNING_TYPE“ angezeigt. Außerdem soll das Auftreten der Event-Id 4319 (doppelte IP-Adresse) eine kritische Meldung erzeugen und die Event-Id 100 (falsches FTP login) eine Minor-Meldung hervorrufen. Die Events mit der Event-Id 10001, 36871, 9001, 8032, 500, 770, 104, 257 sollen unterdrückt werden.

Die Option “-S“ mit der anschließenden Liste der zu unterdrückenden Event-Ids muss am Ende der Zeile stehen.

Hinweis: Die Suchargumente Event-Id und Event-Typ kann man auch als Vorauswahl verstehen, weil bei einer entsprechenden Meldung die Attribute „Source“ und die Message-Strings mitgeliefert werden. Diese lassen sich an der Management-Station zusätzlich filtern.

Beispiel 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 bedeutet, dass bei jedem Aufruf eine gesonderte Steuerinformation zur Management-Station geschickt wird. Bei Ausbleiben dieser Meldung erfolgt eine entsprechende Signalisierung.

Beispiel 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 # Schwellwert über 120 Minuten gemittelt, 96% minor 99% critical
#cpu::95[min] 100[crit] # Schwellwert für kurzzeitige Belastung: 95% minor, 100 critical
memory::90[warn] 99[crit] # Schwellwerte 90% und 99%, Severity warning und critical
page::95[maj] 99[crit] # Schwellwerte 95% und 99%, Severity major und critical
lifecheck::
#end of configuration file
```

Überwachung Performance: CPU-Belastung, Auslastung Memory, Belegung Pagefile.

Mit dem Programm startwinmonagent.exe kann man den Agenten als Task im Hintergrund aufrufen. Es muss im gleichen Verzeichnis liegen wie das Agentenprogramm und hat die gleichen Parameter. Durch Verwendung der Einrichtung „Geplante Tasks“ aktiviert man die Überwachung beim Hochfahren bzw. beim Reboot des Betriebssystems.

4) Scheduling

Wenn das Programm nicht durch das Schlüsselwort pollingsecs:: als Hintergrundprozess betrieben wird, kann es periodisch mit der Einrichtung „Geplante Tasks“ aufgerufen werden. Das Intervall sollte nicht größer sein als 30 Minuten. Empfohlen wird 5 bis 20 Minuten.

13. Programm “logmonagent.exe“

Auswertung von Logfiles für Windows. In der Konfigurationsdatei können mehrere Protokolldateien vereinbart werden.

1) Aufruf

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

2) Konfigurationsdatei

Die Konfigurationsdatei hat den Name „logmonagent.conf“ und liegt im gleichen Verzeichnis wie das Agentenprogramm. Außerdem kann man die (anwachsende) Größe der betreffenden Dateien überwachen.

Der kleinste Wert für pollingsecs:: ist 2 (2 Sekunden).

Es gibt die Schlüsselwörter:

- logfile:: Auswertung des Zuwachses seit der letzten Auswertung
- incremental:: Identisch mit logfile::
- total:: Auswertung der ganzen Datei nach Veränderung

Der Aufbau der Konfigurationsdatei ist so wie bei dem Agentenprogramm logmonagent für Unix. Bei der Spezifikation von Dateinamen kann man als Metazeichen ’*’ und ’?’ verwenden.

3) Beispiel:

Es soll die Alertlog von Oracle überwacht werden.

```
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 # Schleppnetz für Zeilen mit “FATAL”, “fatal” …
“shutdown”-warn
#end of configuration file
```

Wenn die Datei größer als 512 MB ist, erfolgt eine kritische Meldung. Diese Überwachung ist optional. Der Filter “^ORA-1631“-crit[3] begrenzt die Anzeige auf drei, auch wenn mehr Zeilen, die diesem Suchmuster entsprechen, zwischen zwei Aufrufen hinzugekommen sind. Jede Zeile mit dem Anfang “ORA-” wird

mit der Severity “major“ angezeigt. Zeilen mit “warning“, “error“, “fatal“ erscheinen mit der Severity “warning“, “major“, “critical“. Eine Zeile mit “shutdown” erscheint als “warning“.

Bei der Angabe des Dateinamens kann man auch die Metazeichen ’*’ und ’?’ für den Basisnamen benutzen. Dann werden alle die dem Suchmuster entsprechenden Dateien überwacht.

Zum Anstarten als Task im Hintergrund gibt es das Programm startlogmonagent.exe, das im gleichen Verzeichnis wie logmonagent.exe liegen muss und die gleichen Parameter hat.

14. Programm “scriptmonagent.exe“

Das Agentenprogramm ruft per Konfigurationsdatei ausführbare Programme auf, filtert deren Textausgabe und schickt das Ergebnis bei Bedarf zur Management-Station. Mit einer Konfigurationsdatei lassen sich mehrere Überwachungsprogramme definieren. Das Programm kann mit verschiedenen Konfigurationsdateien parametrisiert werden. Es kann als Batch-Programm oder als Hintergrundprozess betrieben werden.

Die in der Konfigurationsdatei definierten Programme werden periodisch aufgerufen; die Ausführung eines jeden Programms erfolgt **parallel** in einem eigenen *Thread*. Es gibt für jedes Programm eine einstellbare Auszeit (*Timeout*). Bei Überschreitung wird das Programm mit der Systemfunktion *TerminateProcess()* gestoppt und es folgt eine kritische Meldung an der Management-Station. **WICHTIG**: Die Ausgabe erfolgt unmittelbar nach Beendigung des Programms aber nicht während der Programmlaufzeit. Ein Programm in einem *Thread* kann nur dann neu aufgerufen werden, wenn der vorherige Programmlauf beendet ist. Die Laufzeit kann größer sein als das Polling Intervall.

1) Aufruf

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

Wird mit der Option “-c“ keine Konfigurationsdatei angegeben, nimmt das System den Defaultnamen “scriptmonagent.conf“. Programm und Konfigurationsdatei müssen im gleichen Verzeichnis sein!

Der kleinste Wert für pollingsecs:: beträgt 2 (zwei Sekunden).

2) Konfigurationsdatei

Es gibt das Schlüsselwort:

- cmd:: Vereinbarung eines ausführbaren Programms mit Name und Filtern für die Textausgabe

Die Zeile(n) für die Skripte haben die Form:

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

tos: Timeout in Sekunden, default: 30 (> 0)
name: logischer Name (Label) bis zu 30 Zeichen lang

Zum Anstarten als Task im Hintergrund gibt es das Programm startscriptmonagent.exe, das im gleichen Verzeichnis wie scriptmonagent.exe liegen muss und die gleichen Parameter hat.

Beispiel: Auswertung des Befehls “netstat -an“

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

# Analyse der aktuellen Netzwerk Verbindungen
cmd::NETSTAT-ANO::netstat -ano::":[0-9]{1,5}[ ]+(127.0.0\.1|\[\::1\]):"-null
":[0-9]{1,5}[ ]+192\.168\."-null # unterdrueckt eigene adressen
":[0-9]{1,5}[ ].*:[1-9][0-9]{,4}[ ];;Unbekannte fremde adr: $3/$1 PID: $$"-warn[+8] # zeigt unbekannt

# Negativ-Filter
cmd::netstat -an::"ESTABLISHED|HERGESTELLT;;Should not be more than 100: $0"-warn[-100]
"ESTABLISHED|HERGESTELLT;;Should not be less than five: $0"-min[<5]
```

Der erste Filter meldet, wenn es mehr als 100 gefundene Zeilen gibt. Der zweite Filter ist ein Negativ-Filter und meldet, wenn es weniger als fünf gibt.

Man beachte, dass für Negativ-Filter die Reihenfolge in der Liste der Filter keine Rolle spielt.

15. Programm “asyncmonagent.exe“

Das Programm stellt eine Erweiterung der Agenten winmonagent.exe, scriptmonagent.exe **und** logmonagent.exe dar. Jedes in der Konfigurationsdatei vereinbarte Skript bzw. jede Protokolldatei kann mit einem eigenen Polling Intervall versehen werden oder als Cronjob definiert werden. Die Ausführung der einzelnen Überwachungsfunktionen geschieht nebenläufig in Form von Threads. Der Agent wird ausschließlich als Hintergrundprozess betrieben. Es gibt für die Ausführung von Programmen und Skripten standardmäßig keine Auszeit (*Timeout*); deren Ausgabe wird nach der Filterung in Echtzeit zur Management-Station geschickt. Ansonsten gilt die Beschreibung von asyncmonagent unter Unix.

1) Aufruf

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

Wird mit der Option “-c“ keine Konfigurationsdatei angegeben, nimmt das System den Defaultnamen “asyncmonagent.conf“. Programm und Konfigurationsdatei müssen im gleichen Verzeichnis liegen!

2) Konfigurationsdatei

Es gibt die Schlüsselwörter:

- cmd:: Vereinbarung eines ausführbaren Programms mit Name und Filtern für die Textausgabe, Angabe des Programms entweder in Kurzform (z.B. “netstat -ano“ oder “%netstat -ano“) oder mit vollständiger Pfadangabe einschließlich *extension* (.exe, .bat)
- logfile:: Modus „Zuwachs seit dem letzten Aufruf auswerten“
- incremental:: identisch mit logfile::
- total:: Modus „ganze Datei auswerten, wenn sie sich geändert hat“
- lifecheck:: Dynamische Registrierung an der Management-Station

Die Angabe des ausführbaren Programms in Kurzform bewirkt, dass der Befehl intern mit “cmd.exe /c ...“ ausgeführt wird. Die Angabe eines Programms mit vollständiger Pfadangabe (erkennbar durch ‘\’) bewirkt die direkte Ausführung. In diesem Fall darf der Dateityp (.exe, .bat, etc.) **nicht** fehlen und der *Basename* der Datei darf keine Leerzeichen enthalten! Mögliche Parameter folgen dem Programmnamen getrennt durch mindestens ein Leerzeichen.

Beispiel:

```
# Programm “wmic.exe“ startet nach 15 Sekunden und terminiert nicht, wird direkt ausgeführt
cmd::CPULOAD-WMI::15::C:\Windows\system32\wbem\WMIC.exe
cpu get loadpercentage /every:10::”[0-9]+”-info
# “wmic“ wird intern ausgeführt mit “cmd.exe /c“
cmd::PROCESS-WMI::30::wmic process list brief::“[ ]([7-9][0-9]|[0-9]{3,})[ ]“-warn
```

Zum Anstarten als Task im Hintergrund gibt es das Programm startasyncmon-agent.exe, das im gleichen Verzeichnis wie asyncmonagent.exe liegen muss und die gleichen Parameter hat.

Beispiele:

```
destination::192.168.178.20
portno::55555
attribute::Windows AsyncMonitorCollection
lifecheck::120 # alle 120 Sekunden heartbeat signal zur Management-Station

suppressionsecs::3600
# Wiederholung von identisch gleichen Meldungen für eine Stunde
# unterdrücken

# CRONJOB, jeden Tag von Mo - Fr, 10.00
cmd::0 10 * * 1-5::echo Es ist 10 Uhr (Wochentag)::”*”-info

# auswerten “netstat -e”, alle 15 Minuten von 6-21 Uhr (ausschließlich 21)
cmd::NETSTAT-E::*/15 6-20 * * *::netstat -e::
”^(Errors|Fehler)[ ]+0[ ]+0”-null “^(Errors|Fehler);;Errors received: $2, errors sent: $3”-min

# wmic fuer Prozesse, Polling Intervall 5 Minuten
cmd::THREAD-PROCESSES::5.0::wmic process
get Name,ProcessID, ThreadCount::”[ ][0-9]+[ ]+[0-9]{3,};;$1 id: $2 thrcnt: $3”-min

# Analyse der aktuellen Netzwerk Verbindungen,  Polling Intervall 4 Sekunden
cmd::NETSTAT-ANO::4::%netstat -ano::”:[0-9]{1,5}[ ]+(127.0.0\.1|\[\::1\]):”-null
“:[0-9]{1,5}[ ]+192\.168\.”-null # unterdrueckt eigene adressen
“:[0-9]{1,5}[ ].*:[1-9][0-9]{,4}[ ];;Unbekannte fremde adr: $3/$1 PID: $$”-warn[8] # zeigt unbekannt
# … weitere Protokolldateien, Befehle, Cronjobs
```

16. Programm “rsendmsg.exe“

Das Programm schickt Meldungen direkt zur Management-Station. Dort kann man sie bei Bedarf filtern oder umsetzen. In Scripten verwendet kann man beliebige (Überwachungs-)Funktionen realisieren. Der zugehörige Event-Typ ist “Rsendmsg“.

Aufruf

rsendmsg.exe [-p <port>] [-d <server>] [-e <ausweichserver>] -g <gruppe> -o <object> [-n <nodename>] -s <severity> -m <meldungstext>

Parameter:

- -p: Portnummer tcp; optional wenn Eintrag in “monitorkeys.sig”
- -d: Managementserver; optional wenn Eintrag in “monitorkeys.sig”
- -e: Ausweichserver; optional wenn Eintrag in “monitorkeys.sig”
- -g: Meldungsgruppe
- -o: Object
- -n: Nodename (optional)
- -s: Severity (inform|minor|warning|major|critical)
- -m: Meldungstext/Event-Text

Returncode:

0: erfolgreich
1: Eingabefehler
2: Kommunikationsfehler

17. Programm “remoteconfd.exe“

Hintergrundprozess zur zentralen Bearbeitung der Konfigurationsdateien von der Management-Station aus. Die über das Netz gehenden Daten werden nach der Methode AES mit Blockverkettungsmodus (CBC) verschlüsselt. Die Verschlüsselung erfolgt dynamisch, ein und derselbe Klartext wird nachfolgend immer verschieden chiffriert. Der Socket-Typ für die Konfigurationsdateien ist *udp* oder *tcp*.

Aufruf:

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: Portnummer udp; optional wenn Eintrag in “monitorkeys.sig”
- -t: Horcht auf einen Port *tcp*, sonst *udp*
- -i: IP-Adresse oder Name der Management-Station als Client; optional wenn Eintrag in “monitorkeys.sig”
- -j: Adresse oder Name der Ausweichstation als Client; optional wenn Eintrag in “monitorkeys.sig”
- -u: Die Ausgabe von “remote command” soll immer UTF-8 sein; Voreinstellung: Die aktuelle *code page* (z.B. CP1252)
- -a: Dateiname für die Vereinbarung einer Zeichenkette zur Authentifizierung. Der String darf nicht kürzer als acht und nicht länger als 30 Zeichen sein
- -F: root-fs für Zugriff auf Dateien
- -r: Abschalten Kommando-Interface (remote command); es dürfen keine Kommandos von den Clients ausgeführt werden
- -L: Name der Protokolldatei
- -m: Port für Meldung zur Management-Station (tcp)
- -g: Meldungsgruppe für Start-/Stop-Meldung (default: ADMIN_)
- -6: ipv6, sonst ipv4
- -h: Hilfetext

Die Optionen “-i“ und “-j“ mit den Hostnamen oder IP-Adressen sorgen dafür, dass Anfragen nur von diesen Clients gemacht werden können. Fehlen die Optionen, können von überall Anfragen gemacht werden. Der Default-Wert für die Option “-a“ ist “remoteconfd.auth“ im gleichen Verzeichnis.

Zum Anstarten als Task im Hintergrund gibt es das Programm startremoteconfd.exe, das im gleichen Verzeichnis wie remoteconfd.exe liegen muss und die gleichen Parameter wie remoteconfd.exe hat. Mit der Einrichtung „Geplante

Tasks“ kann man so beim Hochfahren des Betriebssystems die Überwachung aktivieren.

Beispiel:

startremoteconfd -p 55555

Das Programm remoteconfd.exe im gleichen Verzeichnis wird aufgerufen und in den Hintergrund geschickt. Es horcht auf den Port 55555/udp.