Integrated Server Monitoring

System zur automatisierten Beaufsichtigung von Computer-Servern

Monitoring der nächsten Generation

© Dipl.-Inform. Wilhelm Buchholz

http://www.monitor-site.de

Das System bietet die folgenden Funktionen:

- Autonome Agenten für Linux, Windows, AIX, Solaris, HP-UX, Mac OS X
- Verschiedene Formen der Logfileauswertung, volle Integration von journalctl (systemd)
- Überwachungs- und Korrekturskripte
- Filterung und Darstellung von SNMP-Traps
- Aktive Überwachung von Tcp-Ports
- Zentrale Management-Station mit Graphischer (X11) Bedienoberfläche und Weboberfläche
- Multiuserfähigkeit
- Integrierte Datenhaltung mit dynamischer Speicherverwaltung und Ringspeicher
- Parallelität durch Multithreading, asynchrone Verarbeitung
- Verlaufsanzeige und Statusanzeige an der Management-Station
- Dynamische Registrierung und Heartbeat
- Filterfunktionen zur Differenzenanzeige
- Weiterleitung von Meldungen, automatische Aktionen
- Zentrale Verwaltung der Konfigurationsdateien und administrativer Zugriff auf die Nodes
- Verschlüsselte Kommunikation mit den Agenten über einen Port (tcp/udp)
- Sowohl ipv4 als auch ipv6
- Programm vollständig in C++, keine Metasprachen
- Management-Station für Linux, vorzugsweise 64 Bit
- Komfortable Bedienung

Das vorliegende System, aus langjähriger betrieblicher Erfahrung entstanden, ist überschaubar, mühelos zu installieren und zu betreiben. Es hat eine Bandbreite von einigen Dutzend bis zu einigen Tausend Servern (Nodes), ist universell einsetzbar und hat somit die besten Voraussetzungen für ein breites Publikum. Durch die Mehrbenutzerfähigkeit ist es auch für große, arbeitsteilige Unternehmen geeignet.

Inhaltsverzeichnis

1. Einleitung

Der Begriff Überwachung oder Monitoring bedeutet das zeitnahe Erkennen kundenrelevanter Störungen auf Computer-Servern von einer zentralen Warte aus. Dadurch lässt sich die Verfügbarkeit von Rechenanlagen erhöhen. Das vorliegende System zur Echtzeitüberwachung ist ein Gegenentwurf zu den kommerziellen Frameworks namhafter Anbieter, die aus den neunziger Jahren des vorigen Jahrhunderts stammen.

Die damals entwickelten Systeme, die es schaffen, dass man für das Installieren schon Schulungen (oder fremde Hilfe) braucht, haben eine aufgeblähte Komplexität, die in keinem Verhältnis zu den betrieblichen Anforderungen steht und sich auch negativ auf den Aufwand für die Vernetzung auswirkt. Die Kosten für Anschaffung, Betrieb, Lernaufwand, „Berater“ kann man nur als abenteuerlich bezeichnen. Der Umgang mit Werkzeugen dieser Art stellt selbst ein nicht unerhebliches Problem dar.

Parallel - und auch als Reaktion auf die Schwierigkeiten mit den großen kommerziellen Systemen - sind dann eine Reihe von kleineren Tools (auch Open Source) entstanden, die vielleicht von der Anschaffung und den Betriebskosten günstiger sind, denen es aber an Funktionalität mangelt. Das gilt zum Beispiel für eine praxistaugliche, inhaltliche Logfileauswertung, bei der es weniger um visuelle Effekte geht als vielmehr um die Gewinnung weiterführender (proaktiver) Informationen über einen Server; es gilt aber auch für eine sinnvolle Ergebnisdarstellung, die mit einer persistenten Datenhaltung gekoppelt ist. Das Kapazitätsproblem, also die Aufnahmefähigkeit **einer** zentralen Management-Station für eine bestimmte Anzahl von Clients, ist meistens nicht geklärt. Es kommt nicht nur auf einen, sondern auf alle in der Überwachung befindlichen Server an.

Die derzeitigen, real existierenden Systeme tendieren dazu, Probleme, die eigentlich der Anbieter lösen müsste, auf den Benutzer abzuwälzen. Dieser sieht sich dann unvermittelt mit Entwicklungs- und/oder Entwurfsaufgaben konfrontiert, die den beabsichtigten Nutzen in das Gegenteil verkehren (bei den großen Systemen sind Entwicklungs- und Testumgebungen obligatorisch). Die Aussicht auf eine permanente „Baustelle“ ohne realen Zugewinn ist groß. Der Bedienaufwand ist entschieden zu hoch (was auch dem Grundgedanken der Automatisierung widerspricht).

In diesem System ist berücksichtigt, dass heute weitgehend bekannt ist, was man bei Computerservern zu überwachen hat. Dazu gehört die automatische und zeitnahe Auswertung einer Reihe von Protokolldateien (Logfiles,

auch für individuelle Anwendungen) auf einem Server, bei der von Suchmustern gefundene Zeilen übertragen und zentral dargestellt werden. Weiter gibt es Überwachungsskripte und eine Standardüberwachung als Bündelung der wichtigsten, problembezogenen Überwachungsfunktionen. Ferner sind Funktionen zur Eskalation von Störungsmeldungen und zum Versenden von E-Mails integriert.

Des weiteren ist berücksichtigt, dass die Anzahl der Server seit der Anfangszeit sprunghaft gestiegen ist und es Restriktionen im Netzwerk wie Firewalls, (doppelte) *address translation*, ipv6 sowie neue Betriebsarten (Cloud-Computing, Grid-Computing) gibt. Auch langsame Netzverbindungen stellen kein Hindernis dar. Zum Datentransfer wird lediglich ein Port tcp/udp benötigt. Durch eine dynamische Verschlüsselung (AES/CBC, MRC4) ist auch das Internet als Übertragungsmedium nutzbar. Als zentrale Ausgabe und als Front-End dient auf der Basis von Linux eine Verlaufsanzeige für Störungsmeldungen und eine Statusanzeige für alle in der Überwachung befindlichen Server, jeweils als graphische Oberfläche und als Weboberfläche. Die Verlaufsanzeige ist in der Lage, eine Störung als dynamischen Vorgang mit Anfang, Ende und Dauer darzustellen und zu dokumentieren.

Das System hat einen definierten Funktionsumfang, so dass irgendwelche „plugins“ oder „smart plugins“ nicht notwendig sind.

2. Agenten und Ergebnisanzeige

Das Schwergewicht der Überwachung liegt auf den systemeigenen Agenten.

Agenten sind Programme, die auf den zu überwachenden Servern (=Nodes) periodisch Abfragen durchführen. Das Ergebnis wird in Form eines Events (=Meldung) über das Netz zu einer zentralen Management-Station geschickt und dort gut sichtbar zur Anzeige gebracht. Die Events werden mit einem geheimen Schlüssel und *session keys* verschlüsselt (*session keys* bewirken, dass jede Nachricht anders chiffriert wird). Darüber hinaus findet eine Integritätsprüfung durch Checksummen statt.

Die Übertragung zur Management-Station erfolgt in eine Richtung über einen frei wählbaren Port tcp. Jeder Agent ist mit einer Konfigurationsdatei parametrisierbar, in die man neben den Parametern zur Überwachung den Port und die Adresse der Management-Station vereinbart. Das kann auch eine numerische IP-Adresse sein, weil die Kommunikation unabhängig von DNS (*domain name service*) stattfindet. Eine Namensauflösung auf der Management-Station ist nicht nötig.

Optional kann zusätzlich die Adresse einer Ausweich-Managementstation angegeben werden, für den Fall, dass die erste ausfällt.

Die Verwendung von autonomen Agenten hat die folgenden Vorteile:

- Geringe Netzbelastung: Es wird nur im Störungsfall gemeldet. Man muss sich klar machen, dass weit über 90% der lokalen Abfragen auf den Nodes negativ verlaufen (was auch erwünscht ist) und das Netz nicht in Anspruch genommen wird.
- Entlastung der Management-Station von Abfragen. Bei zunehmender Anzahl von zu überwachenden Servern steigt die Belastung allenfalls linear aber nicht progressiv.
- Security: Meldungen werden geschickt, sie können nicht von außen abgefragt werden. Durch die Agenten sind die Nodes nicht einmal potenziell angreifbar.
- Kein Informationsverlust bei (kurzzeitigen) Netzstörungen. Die Agenten speichern die Meldungen lokal und liefern sie am Ende der Störung nach.
- Zugänglichkeit von Daten. Es liegt in der Natur der Sache, dass den Server betreffende Daten auf dem Server selbst am besten zu gewinnen sind.

Der Einsatz von Agenten ist von der Ressourcenbelastung her weitaus günstiger als das ständige Abfragen über das Netz von einer Management-Station aus. Die Netzverbindung zur Management-Station wird bei Bedarf geöffnet und wieder geschlossen. Es gibt keine permanente Tcp-Verbindung.

Es gibt Agenten für die Aufgaben:

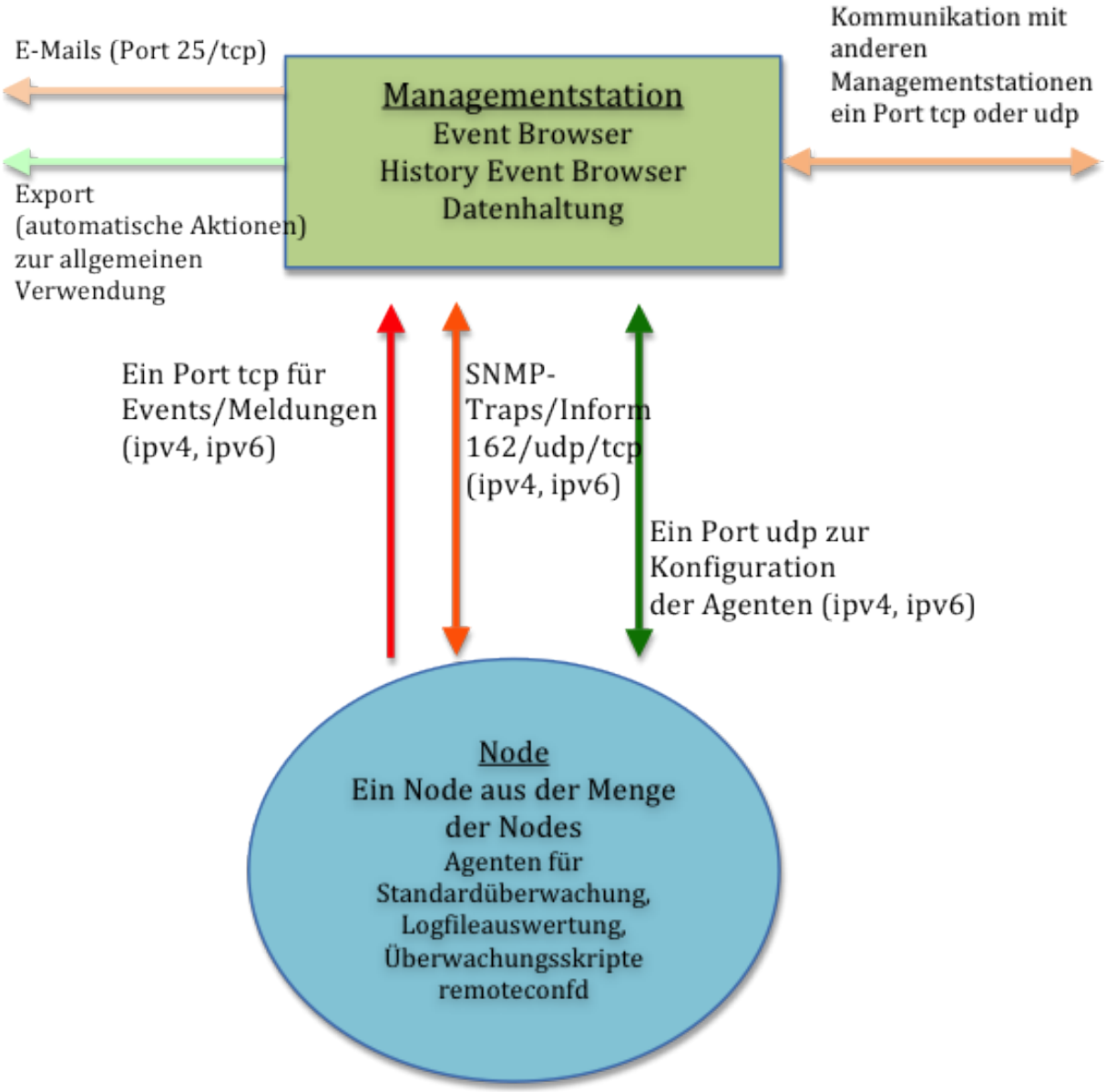
- Standardüberwachung (Unix, Windows)
- Überwachungsskripte (Unix, Windows)
- Allgemeine Logfileauswertung (Unix, Windows)
- Multiple Logfileauswertung (Unix)
- Multiple rekursive Logfileauswertung (Unix)
- Direktmeldung eines Events (Unix, Windows)

Die Agenten können wahlweise als Hintergrundprozess oder als Batchprogramm betrieben werden (Ausnahme: “asyncmonagent“, siehe unten). Bei Hintergrundprozessen vereinbart man das Polling Intervall in der Konfigurationsdatei mit der Zeiteinheit Sekunden(z.B. 300 für fünf Minuten). Bei einer späteren Änderung der Konfigurationsdatei wird diese automatisch eingelesen und der Vorgang an der Management-Station signalisiert.

Ohne den Eintrag für das Polling Intervall terminiert der Agent nach jedem Aufruf. Er wird dann von einem lokalen Scheduler (zum Beispiel crontab für Unix, Aufgabenplanung für Windows) periodisch aufgerufen. Die Agenten benötigen keine root- oder Administratorenrechte.

Auf der Management-Station erscheinen die Events in chronologischer Reihenfolge als **Verlaufsanzeige** in dem „Event Browser“, der zum einen als X11-Programm, zum anderen als Web-Applikation vorliegt**.** Im Gegensatz zu einer reinen Statusanzeige bleiben die Meldungen erhalten, gehen also nicht durch Überschreiben mit einem neuen Status verloren. Damit lassen sich Störungen als dynamische Vorgänge erkennen und auch später noch rekonstruieren. Darüber hinaus lassen sich auch Mehrfachstörungen darstellen. Zum Beispiel kann nicht nur ein Filesystem volllaufen sondern mehrere, was auch mit mehreren Meldungen angezeigt wird (bei großen Unix-Servern kann es Dutzende von Filesystemen geben). Das gleiche gilt auch für das umfangreiche Gebiet der Logfileauswertung, bei der gefundene Zeilen der Protokolldatei als Meldungstext dargestellt werden. Das System verwendet die Zeichensätze UTF-8 (Unicode), ISO-8859-1 bis ISO-8859-10, ISO-8859-13 bis ISO-8859-16 sowie CP1250 bis CP1258 (Windows).

Datenflüsse



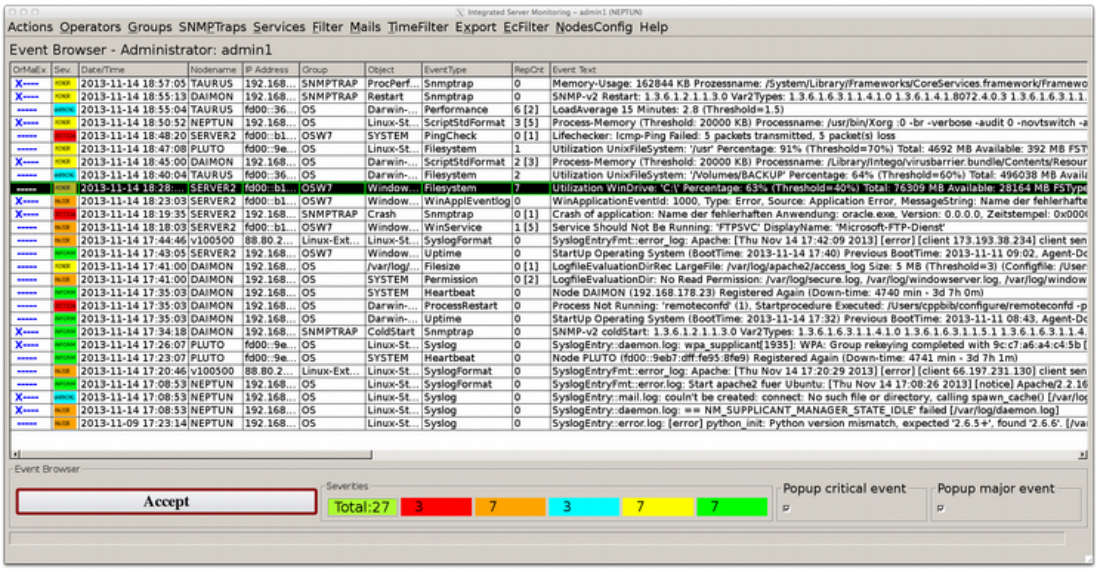
Die Prinzipskizze zeigt die Kommunikation der Agenten mit der Management-Station. Die Konfigurationsdateien für die Agenten sind Textdateien, die auch mit einem lokalen Editor zu bearbeiten sind.

Es gibt für das System fünf verschiedene Meldungsschweren (Severities):

1. inform (informativ)
2. minor (geringfügig)
3. warning (warnend)
4. major (schwerwiegend)
5. critical (kritisch)

Eine Meldung besteht aus den Attributen Meldungsschwere bzw. Severity (farbig unterlegt), Empfangszeit (ISO Datumsformat), Nodename als Netzwerkname der Meldungsquelle (Stringkonstante), IP-Adresse (ipv4 oder ipv6), Gruppe (Stringkonstante), Objekt (Stringkonstante), EventType (Stringkonstante), RepCnt (Zahl) als Wiederholungszähler und ein Zähler für unterdrückte, gleiche Meldungen; zum Schluss der Meldungstext, der eine Länge von 1024 Byte haben kann. Ferner gibt es als verstecktes Attribut unter anderem die Entstehungszeit auf dem Node (UTC-Zeitstempel), den Namen des Operators, der ein Event quittiert hat, und die Zeit des Quittierens. Das Attribut „Group“ (Meldungsgruppe) bestimmt die Zuordnung zu den Operatoren.

X11-Oberfläche (Verlaufsanzeige)



Die Abbildung zeigt den Event Browser der zentralen Management-Station mit einer Abfolge von aktuellen Meldungen. Jede Zeile stellt ein Event bzw. Meldung dar. Die jüngste Meldung ist ganz oben, die älteste am unteren Ende. Die zweite Spalte links ist die Meldungsschwere (Severity), gefolgt von Datum und Uhrzeit, die letzte ganz rechts der Event-Text, der eine eindeutige Beschreibung zu dem Ereignis liefert. Der Meldungstext hat eine herausragende Bedeutung, weil aus ihm in Kombination mit der Severity die Fehlerbehandlung hervorgeht.

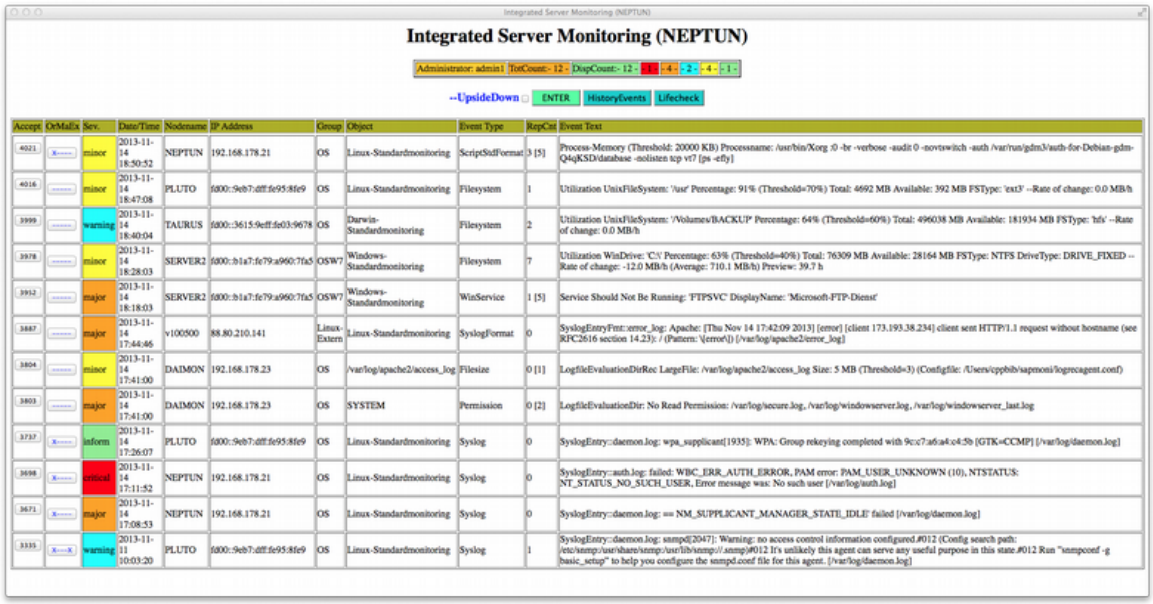
Mit dem Pushbutton “*Accep*t“ (links unten) wird eine Meldung nach der Bearbeitung entfernt, vom Zustand „aktuell“ in den Zustand „archiviert“ gebracht. Sie ist dann auch für andere Benutzer nicht mehr sichtbar, kann aber in einem anderen Teil der Bedienoberfläche, dem „History Event Browser“, wieder zum Vorschein gebracht werden.

In der oberen Leiste befinden sich die Pulldownmenues, mit denen man zu den Eingabemasken für die verschiedenen Einstellungen des Systems gelangt. Sie unterscheiden sich je nachdem, ob man sich als Administrator oder Operator anmeldet. Ein Administrator, von denen es mehrere geben kann, hat die Berechtigung, alle Einstellungen vorzunehmen, und er sieht grundsätzlich alle Meldungen. Ein Operator sieht nur die Meldungen, deren Gruppen ihm zugewiesen sind. Er ist berechtigt, die entsprechenden Nodes mit dem Menuepunkt „*NodesConfig*“ zu konfigurieren.

Alle Benutzer des Systems müssen sich mit Benutzerkennung und Passwort anmelden. Ein Benutzer kann sich auch mehrmals einloggen.

Die optische Erscheinungsform der X11-Oberfläche (Farben, Schriftgrößen, Rahmen, etc) kann man mit Hilfe von *stylesheets* individuell gestalten.

Web-Oberfläche (Verlaufsanzeige)



Die Abbildung zeigt die gleiche Sicht als Web-Oberfläche für aktive (unbearbeitete) Meldungen. Eine Meldung wird nach der Bearbeitung durch einen Pushbutton auf der linken Seite quittiert. Sie ist dann hier nicht mehr zu sehen. Die Zahl in eckigen Klammern in der Spalte “RepCnt“ zeigt die Anzahl der von dem entsprechenden Filter in einem Zeitraum unterdrückten Meldungen an, so dass man eine Kontrolle über die Häufigkeit hat.

Die Form der Darstellung als zeitlicher Ablauf von Ereignissen ist eine grundlegende betriebliche Anforderung an ein Überwachungswerkzeug.

3. Ersetzungsmechanismus (Formatanweisung)

Der Ersetzungsmechanismus ist bestimmt für die benutzergerechte Gestaltung des Meldungstextes eines Events. Er wird auf der Management-Station für die Konfiguration von SNMP-Traps und für die Filter eingesetzt, bei den Agenten für die Logfile-Auswertung und für die Auswertung von Überwachungsskripten.

Der Gebrauch des Ersetzungsmechanismus erlaubt es, einen eingehenden Text, der aus einer Reihe von Wörtern bzw. Spalten besteht, umzuordnen, zu verkürzen (also unerwünschte Teile des Textes zu entfernen) und neue Informationen hinzuzufügen. Die hinzugefügten Informationen können auch Anweisungen zur Fehlerbehebung sein.

Der Mechanismus wird realisiert durch einen Formatstring. Darin enthalten sind die Operatoren '$', '%', '&', @, ? und '-´, denen eine Zahl oder ein Substring folgt.

Auf diese Weise kann man durch den Formatstring (= Transformationsanweisung) eine eingehende Zeile in einen ausgehenden Text umwandeln.

- $n oder ${n}: n ist eine Zahl [1..99]. Gibt aus das <n>. Wort des Eingangstextes
- %n oder %{n}: linksshift, gibt aus den um <n> Spalten nach links verschobenen Eingangstext, die Eingangszeile selbst bleibt unverändert
- &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
- &{n,m}: Gibt aus <m> Zeichen ab dem <n>. Zeichen der Eingangszeile
- &{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
- @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
- %<[n|]substring>: Suche nach einem Substring in der ganzen Zeile oder optional nach dem <n>. Auftreten eines Substrings in der Zeile (n > 0). Dann Shift nach links in der Eingangszeile zu dem Unterstrings substring. Der gefundene Substring bildet den neuen Anfang der Eingangszeile, es erfolgt keine unmittelbare Ausgabe

- ?<[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
- -<[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
- $*: Gibt aus die ganze Eingangszeile
- $$: Gibt aus die letzte Spalte des Eingangstextes

Die Suche nach Unterstrings erfolgt von links nach rechts, so wie auch die Formatanweisung von links nach rechts abgearbeitet wird. Die Operatoren lassen sich kombinieren. Man kann zum Beispiel das erste Wort mit dem zweiten im Meldungstext vertauschen und dazwischen einen frei wählbaren String setzen. Fehlen die Operatoren im Formattext, dann wird der eingehende Text komplett durch die Stringkonstante ersetzt.

Beispiel:

Bei der Auswertung der Syslog-Datei eines Linux-Servers ist die folgende Zeile ermittelt worden, die den Eingangstext für die Formatierung bildet:

Oct 20 16:06:45 v100500 sshd[3992]: Address 69.65.49.82 maps to guryat.com, but this does not map back to the address - POSSIBLE BREAK-IN ATTEMPT!

Formatstring: “Einbruchsversuch per ssh (betroffener Server: $4): %5“

Ausgabe für den Event-Text:

Einbruchsversuch per ssh (betroffener Server: v10050): Address 69.65.49.82 maps to guryat.com, but this does not map back to the address - POSSIBLE BREAK-IN ATTEMPT!

Der Operator “%5“ bewirkt die Ausgabe der um fünf Spalten/Wörter nach links verschobene Eingangszeile. “$4“ setzt das vierte Wort im Eingangstext an die gewünschte Stelle des Ausgangstextes. Die formatierte, eigentliche Textinformation kann jetzt auf Gleichheit geprüft werden, damit sie innerhalb eines wählbaren Zeitraumes (z.B. 10 Minuten) nur einmal angezeigt wird (bei einem Eindringversuch kann es innerhalb weniger Minuten hunderte von Einträgen dieser Art geben).

Wenn die Formatanweisung fehlt, wird der Text in voller Länge ausgegeben.

4. Regular Expressions (Suchmuster)

Das System benutzt *extended regular expressions* nach dem POSIX Standard als Suchmuster. Die Eigenschaften sind in den *manual pages* von Unix nachzulesen. Für die speziellen Anforderungen dieses Systems gibt es optionale Zusätze, die an das Ende des Suchmusters nach einem Schrägstrich '/' angehängt werden.

Die Syntax ist: <RegExp>[/i|v|!]

Das eigentliche Suchmuster gefolgt von '/' und 'i' oder 'v' oder '!'.

Die Bedeutung der Zeichen ist:

- 'i': Kein Unterschied bei Groß/Kleinschreibung
- 'v': Das Suchergebnis wird umgekehrt, Groß/Kleinschreibung wird unterschieden
- '!': Das Suchergebnis wird umgekehrt, Groß/Kleinschreibung wird **nicht** unterschieden

Die Sonderbedeutung kann man mit einem vorangestellten Backslash '\' aufheben.

Beispiele:

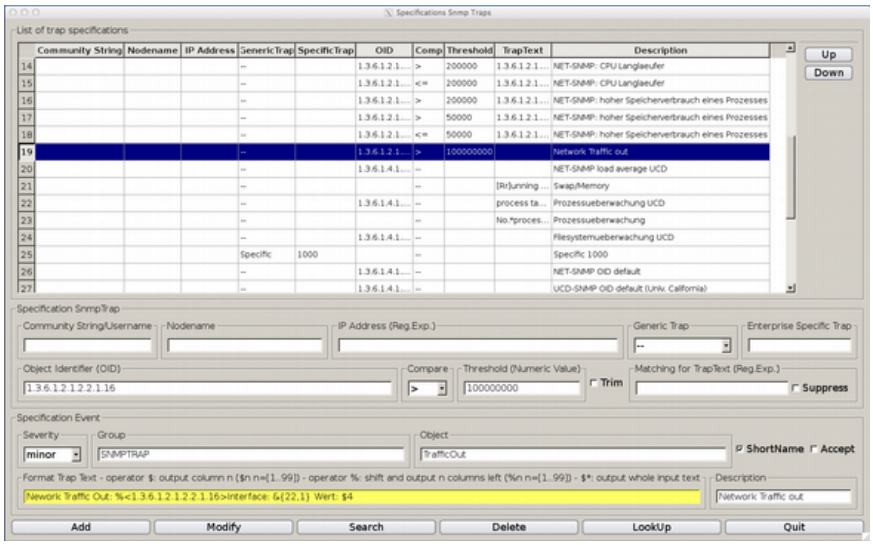
"^os$/i" trifft die Zeichenkette "OS", "Os", "oS", "os"
"fatal/i" trifft Zeilen mit "Fatal", "FATAL", "fatal", ...
"[0-9]/v" trifft Zeilen, die keine Ziffern enthalten
"ABC/v" trifft Zeilen, die **nicht** "ABC" enthalten
"ABC/!" trifft Zeilen, die **nicht** "Abc", "ABC", "abc", ... enthalten
"[ ][1-9][0-9]{1,2}[ ]/v" trifft eine Zahl, die mehr als drei Stellen hat
"[ ]3\.14[0-9]*[ ]" trifft die Zahl 3.14...
"[ ]([3-9][0-9]{5})|([1-9][0-9]{6,})[ ]" findet eine Zahl, die größer oder gleich 300000 ist

5. SNMP-Traps (Trap Receiver)

SNMP (*simple network management protocol*) ist eine standardmäßige Einrichtung für das System Management, die plattformunabhängig ist. Traps bzw. *Notifications* werden eigenständig von den Servern über den Port 162/udp geschickt (nicht zu verwechseln mit Port 161/udp, über den Abfragen von außen **an** einen Server möglich sind). Auf den Servern muss der Hintergrundprozess „snmpd“ aktiv sein und in der Konfigurationsdatei die Management-Station als Trap Destination vereinbart sein. Für Windows ist es der SNMP-Dienst.

Das System ist in der Lage, SNMP-Traps der Version 1, 2c und 3 zu empfangen, zu filtern und darzustellen. (Traps der Version 3 nur dann, wenn sie **nicht** verschlüsselt sind, mehr siehe unten). Die Filterung geschieht mit einer geordneten Liste von Spezifikationen, die durch konjunktive Vergleiche der Komponenten darüber entscheidet, ob und wie ein ankommender Trap dargestellt wird. Die empfangenen Trap-Meldungen können auch von anderen SNMP-fähigen Geräten wie Router oder Netzwerkdruckern stammen.

Die Einstellungen für SNMP-Traps nimmt ein Administrator mit Hilfe der graphischen Bedienoberfläche vor:

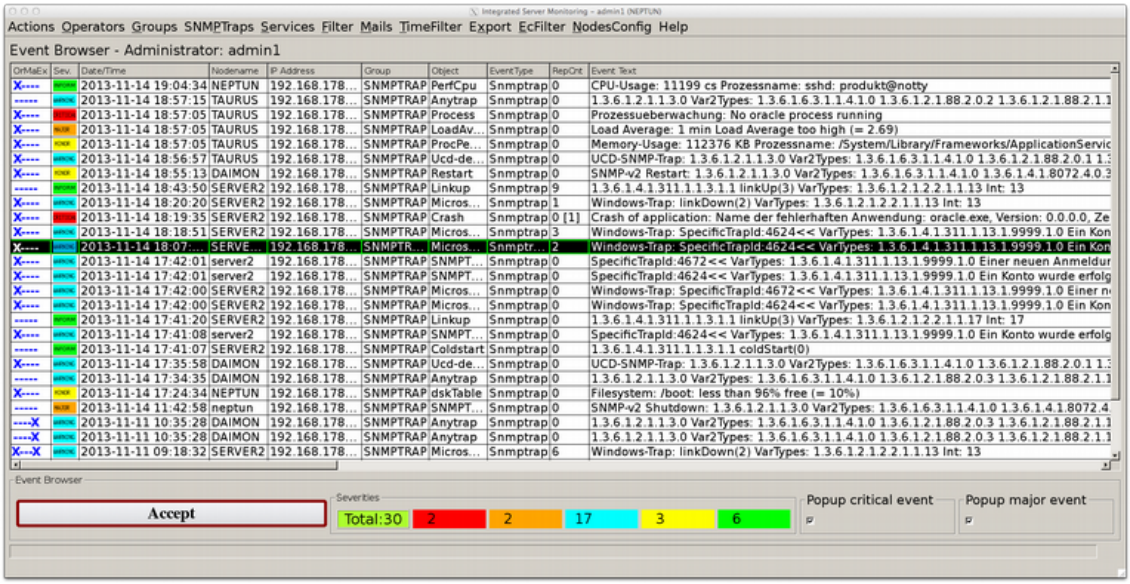


Die Abbildung zeigt die Eingabemaske zur Spezifikation der Trap Meldungen. Im oberen Teil ist die Tabelle der schon existierenden Beschreibungen. Man kann Datensätze hinzufügen, ändern, suchen und löschen. Die Länge der Tabelle ist nicht limitiert.

Ein Trap wird spezifiziert durch die Felder „*Community String*“, „*Generic Trap*“ (V1), „*Enterprise Specific Trap*“ (V1), „*Object Identifier* (*OID*)“ und „Matching for Trap Text“. Man kann gezielt einzelne OID's innerhalb eines Traps ansprechen und dessen zugehörigen Zahlenwert mit einem Schwellwert vergleichen. Ebenso kann man den gleichen Zahlenwert mit unterschiedlichen Schwellwerten und Severities versehen. Die Felder „*Nodename*“ und „*IP Address*“ sind für den Fall vorgesehen, dass es für eine ankommende IP-Adresse keine Namensauflösung gibt oder man einen Aliasnamen seiner Wahl vergeben will (zum Beispiel für Router). Ist ein Feld leer, fällt der Vergleich positiv aus. Sind alle Felder leer oder inaktiv, bedeutet dies: Jeder Trap.

Im unteren Eingabebereich gibt man die Attribute „*Severity*“, „*Group*“ und „*Object*“ für eine Meldung an. Mit dem Eingabefeld „*Format Trap*“ lässt sich optional ein Formatstring definieren, der den eingehenden Meldungstext in den Ausgabetext umwandelt.

Bei einem ankommenden Trap wird die Tabelle von oben beginnend zeilenweise bzw. satzweise durchlaufen. Wenn die Spezifikation übereinstimmt, kommt es zur Ausgabe gemäß Vereinbarungen und der Vorgang terminiert. Mit der Checkbox „Suppress“ lassen sich Traps unterdrücken. Die Reihenfolge der Abarbeitung lässt sich dafür nutzen, auch Traps bisher unbekannter Art zu verarbeiten.



Die Abbildung zeigt eine Auswahl von Trap Meldungen. Ein Trap wird standardmäßig als Meldungstext in einer Abfolge von numerischen OID's und zugehörigem Wert dargestellt (die numerische OID kommt über das Netz und ist verbindlich). Eine besondere Rolle spielen SNMP-Zeichenket-

ten („octet strings“), die entweder als Beschriftung für numerische Werte dienen (zum Beispiel Prozessnamen) oder eigenständige Informationen enthalten. So kann eine Anwendung oder ein Subsystem eine ganze Textzeile als Protokollinformation schicken, die dann als Meldungstext dargestellt wird.

Dieses System ist darauf eingestellt, auch auf Port 162/tcp (oder auch andere Portnummer) zu horchen. Allerdings ist der Gebrauch von tcp bei SNMP eher die Ausnahme.

SNMP-v3:

Für den Empfang verschlüsselter Traps (*secLevel*: *authPriv*) kann man den Trap Receiver snmptrapd benutzen, der für Linux standardmäßig zur Verfügung steht. Mit dem Aufruf “snmptrapd –Lsd -Oq“ schreibt das Programm die empfangenen Notifications mit symbolischen OID’s in die System-Logdatei “/var/log/daemon.log“, die man dann für eine Logfileauswertung verwendet; oder man nimmt den Befehl “journalctl -f -u snmptrapd“ bzw. “journalctl -u snmptraps“.

Hinweis: Für Windows gibt es die Einrichtung „Ereignis-nach-Trap-Konvertierung“ (evntwin.exe). Sie verschickt Einträge in die verschiedenen Windows-Eventlogs als SNMP-Trap der Version 1. Dabei wird die Eventid von Windows als Enterprise Specific Trapid dargestellt.

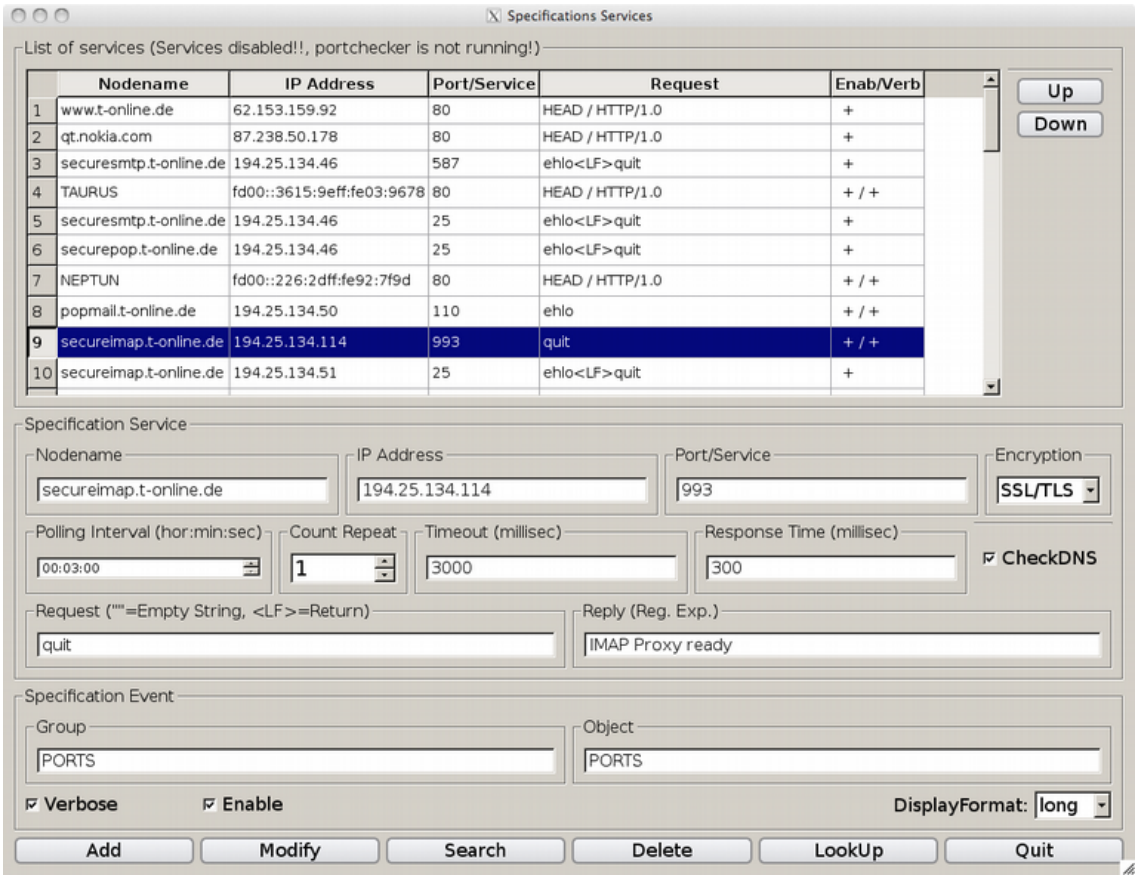
Für SNMP-Abfragen (Port 161) siehe Kapitel „Überwachungsskripte“.

6. Aktive Überwachung

Aktive Überwachung bedeutet das Prüfen von Ports (tcp) entfernter Server. Die Prüfung erfolgt von der Management-Station aus. Es gibt zwei verschiedene Möglichkeiten:

1. Prüfen Erreichbarkeit des Ports durch Öffnen der Netzverbindung
2. Wie Punkt 1, zusätzlich Senden einer Anfrage und Auswertung der Rückgabe

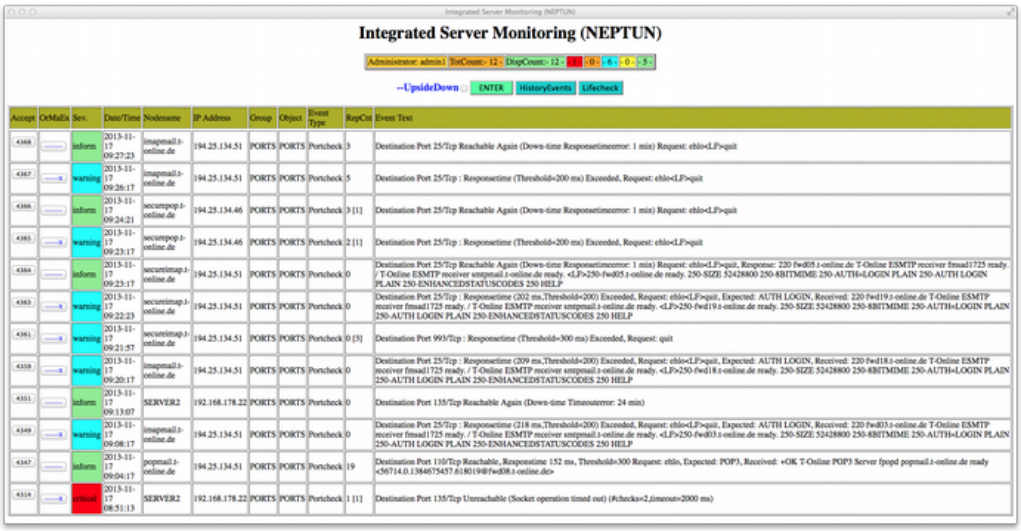
In beiden Fällen wird die Antwortzeit ermittelt und mit einem Schwellwert verglichen. Man kann außerdem bestimmen, ob und wie oft die Prüfung wiederholt werden soll. Es lassen sich auch Ports bzw. Services ansprechen, die mit SSL-Verschlüsselung arbeiten (zum Beispiel Port 443 für https).



Die Abbildung zeigt die Einstellungen, wie sie an der Management-Station von einem Administrator vorgenommen werden können. Die Checkbox „CheckDNS“ bewirkt, dass der eingegebene Nodename als Funktionsargument benutzt wird. Dadurch wird zusätzlich die DNS-Auflösung des Namens

überprüft. Ansonsten wird die IP-Adresse als Funktionsargument genommen.

Die Ausgabe bzw. das Format der Ausgabe nimmt das System vor. Sie lässt sich mit dem Filtermechanismus anpassen.



Die Abbildung zeigt Meldungen von der Portüberwachung in der Web-Anzeige. Nach einer Störung wird mit einer grünen Meldung die Dauer des Ausfalls (Down-time) angezeigt.

7. Massenproblem und Datenhaltung

Einer der wichtigsten Anforderungen ist die zentrale Anzeige und die Verwaltung. Damit ist gemeint, dass es möglichst eine Management-Station gibt und nicht mehrere, auch wenn die Anzahl der zu überwachenden Server nennenswert ist, unabhängig von der Anzahl der Überwachungsfunktionen pro Server. Das muss auch gelten, wenn die Netzwerkumgebung durch Firewalls, (doppelte) *address translation* und andere Besonderheiten schwierig ist.

Für das vorliegende System liegt die maximale Anzahl von Nodes pro Management-Station bei **8192**. Wenn mehrere Stationen im Einsatz sind, können sie Meldungen untereinander austauschen.

Wegen der daraus resultierenden strengen Anforderungen an die Laufzeiteffizienz verwendet man kein (relationales) Datenbankverwaltungssystem, sondern eine extra für dieses Problem entwickelte Lösung, die aus einer Kombination aus *shared memory* und indexsequentiellen Binärdateien besteht.

Bei der Verwaltung der Meldungen bzw. Events muss man zwischen den aktuellen, noch nicht bearbeiteten Events und den alten Meldungen – den History-Daten – unterscheiden. Die aktuellen Meldungen werden in einem *shared-memory*-Bereich, der als **Ringspeicher** organisiert ist und eine Kapazität von 100000 Events hat, gehalten. Hier geschehen auch die Vergleichs- und Ersetzungsoperationen, wenn neue Meldungen eingetroffen sind. Parallel dazu gibt es für die langfristige Speicherung der alten Meldungen die Archiv-Datei als Binärdatei, deren Kapazität bei einem 64-Bit-System praktisch nur von der Größe des Filesystems begrenzt wird. Es lassen sich so alte Meldungen auch bei größeren Umgebungen noch jahrelang zurückverfolgen.

Wichtig ist, dass sowohl der Teil im *shared memory* als auch die Binärdatei unmittelbar, ohne irgendwelche Transformationen zum Zugriff für die X11-Oberfläche und die Web-Oberfläche bereit stehen. Die Zugriffszeiten werden durch binäres Suchen beschleunigt, so dass auch eine große Anzahl (einige Millionen) archivierter Meldungen effizient bewältigt werden können.

Der Empfang der Daten von den Nodes ist so gestaltet, dass die Annahme und die dann folgende Verarbeitung entkoppelt sind, mit einem internen Zwischenspeicher als Schnittstelle. Beides erfolgt nebenläufig in Form von Threads. Durch die asynchrone Verarbeitung und Abspeicherung werden

Belastungsspitzen ausgeglichen, wie sie beim Betrieb von einigen tausend Clients auftreten können.

Die Datenhaltung ist vollständig in das System integriert, es fallen keine administrativen Aufwendungen an, wie sie bei einem regulären Datenbankverwaltungssystem nötig wären. Die Datenbereiche werden im laufenden Betrieb dynamisch erweitert. Alte Meldungen können über die X11- und Web-Oberfläche abgefragt und heruntergeladen werden.

8. Agenten zur Überwachung von Protokolldateien

Protokolldateien („Logfiles“) sind Textdateien, die von System- und Anwendungsprogrammen fortlaufend mit Informationen über ihren Zustand beschrieben werden. Logfiles sind zumindest auf Unix-Systemen allgegenwärtig. Es gibt nicht nur die System-Logdatei(en) (Syslog) sondern auch Logfiles für Datenbankverwaltungssysteme, Web-Server, Firewalls, Backup-Server, etc. Jede Anwendung, für die ein Server letztendlich betrieben wird, hat in der Regel eine (oder mehrere) Protokolldateien.

Dieses System bietet die Möglichkeit, eine Vielzahl von Logdateien auf einem Server auszuwerten, indem man durch geeignete Suchmuster solche Einträge herausfiltert, die auf eine akute oder bevorstehende Störung hinweisen. Als Suchmuster dienen reguläre Ausdrücke (*extended regular expression*, POSIX Standard). Gefundene Einträge bzw. Zeilen werden zusammen mit dem Dateinamen zur Management-Station geschickt und dort als Meldungstext dargestellt. Bei Bedarf kann der Meldungstext am Agenten und/oder an der Management-Station durch einen Formatstring verändert werden.

Large files: Optional kann die Größe einer Protokolldatei überwacht werden. Als Schwellwert gibt man die maximale Größe in Megabyte (MB) und eine Severity an. Bei Überschreitung gibt es eine Meldung mit einem vorformulierten Meldungstext.

Ausnahmebehandlung: Wenn die Zieldatei nicht existiert oder nicht lesend geöffnet werden kann, gibt es eine Meldung mit einem vorformulierten Meldungstext. Dieses Verhalten lässt sich durch ein Sonderzeichen vor der Vereinbarung des Dateinamens ausschalten.

Die Agenten dieses Systems verwenden eine Liste von Filtern, die in den zugehörigen Konfigurationsdateien zusammen mit den Dateinamen vereinbart werden. Ein Filter besteht aus Suchmuster, optional Formatstring, Severity und optional Zähler für die Häufigkeit der gefundenen Einträge.

Es gibt drei verschiedene Filter:

1. Positiv-Filter: Zeile, die dem Suchmuster entspricht, wird angezeigt
2. Suppression-Filter: Zeile, die dem Suchmuster entspricht, wird unterdrückt
3. Negativ-Filter: Es wird angezeigt, wenn beim gesamten Suchvorgang keine Zeile trifft (Suche nach Abwesenheit)

Durch die Liste der Filter am Agenten, die beliebig lang sein kann, findet eine Selektion (Suchmuster) und eine Bewertung (Severity) statt. Der Vorgang spielt sich an der Stelle ab, wo die Daten entstehen. Daten, die nicht selektiert werden, treten nicht in Erscheinung und belasten weder das Netz noch die Management-Station.

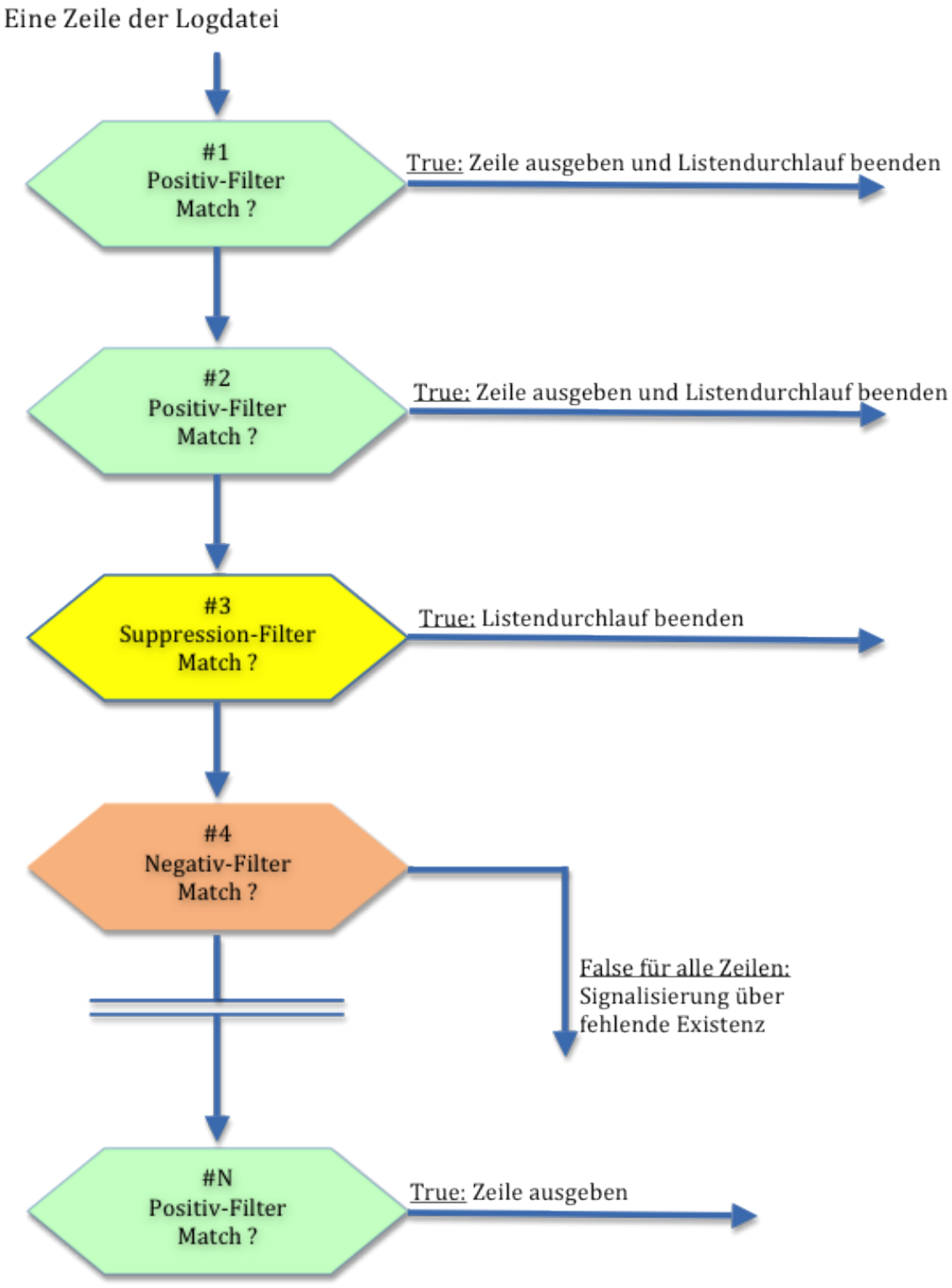
Die Liste der Filter wird bei jeder Zeile bzw. neu hinzugekommenen Zeile der Zieldatei vom ersten bis zum letzten Element durchlaufen. Wenn ein Suchmuster trifft, endet der Durchlauf. Durch diese Form der Abarbeitung wird Eindeutigkeit hergestellt; das heißt, es spielt keine Rolle (es kann sogar beabsichtigt sein), ob ein Suchmuster aus der Liste in einem anderen enthalten ist. So ist z.B. das Suchmuster “error“ in dem Suchmuster “noerror“ enthalten.

Indem man „noerror“ **vor** „error“ platziert und mit einem Suppressionfilter versieht, verhindert man das Erscheinen unerwünschter Meldungen. Die Kombination aus Positiv- und Suppressionfilter in **einer** geordneten Liste ermöglicht die Verwendung von allgemeinen Suchmustern wie “fatal“, “emergency“, “panic“, “sql-error“, “segfault“, “inconsistencies”, “deadlock“, etc.

Bei der Spezifikation der Dateinamen sind Metazeichen wie ’*’ und ’?’(„Wildcards“) im Basisnamen erlaubt. Dadurch lassen sich Protokolldateien in einem gemeinsamen Verzeichnis bündeln. Außerdem werden neu hinzukommende Dateien, die dem Muster entsprechen, dynamisch erfasst.

In der nachfolgenden Skizze wird das Prinzip der Auswertung noch einmal mit einer Liste der Länge N dargestellt. Es ist zu beachten, dass die Reihenfolge von Positiv-Filtern und Suppression-Filtern signifikant ist. Bei Negativ-Filtern spielt die Reihenfolge keine Rolle, da hier die Abwesenheit für den Gesamtvorgang entscheidend ist.

Der gleiche Mechanismus wird auch bei der Auswertung von Überwachungsskripten angewandt.



Auswertungskaskade für Protokolldateien

Die anderen Attribute eines Filters sind neben der Severity optionale Angaben über die maximalen und minimalen Vorkommnisse. Damit werden

„Meldungsfluten“ verhindert. Darüber hinaus besteht sowohl bei den Agenten als auch an der Management-Station die Möglichkeit, eine Transformation des Suchergebnisses durch den Formatstring vorzunehmen.

Das Polling Intervall für die Agenten liegt normalerweise bei 1 bis 5 Minuten. In besonderen Fällen kann man bis auf zwei Sekunden heruntergehen (Programm “asyncmonagent“, siege unten). Die Auswertung über mehr als eine Zeile ist möglich. Man verwendet dazu das Sonderzeichen ‚\n’ in dem Suchmuster.

Beispiel für Protokolldateien sind die Systemlogdateien von Unix (“/var/log/messages“, “/var/log/system.log“, “/var/log/firewall.log“, usw). Der Web-Server Apache hat die Protokolldateien “error_log“ und “access_log“.

8.1 Allgemeine Logfileauswertung

Für die verschiedenen Unix-Derivate gibt es das Programm logmonagent.
Für Windows logmonagent.exe.

Die Agenten sind mit verschiedenen Konfigurationsdateien parametrisierbar. In einer Konfigurationsdatei können mehrere Protokolldateien zusammen mit der Liste von Filtern vereinbart werden.

Inkrementelle Auswertung:

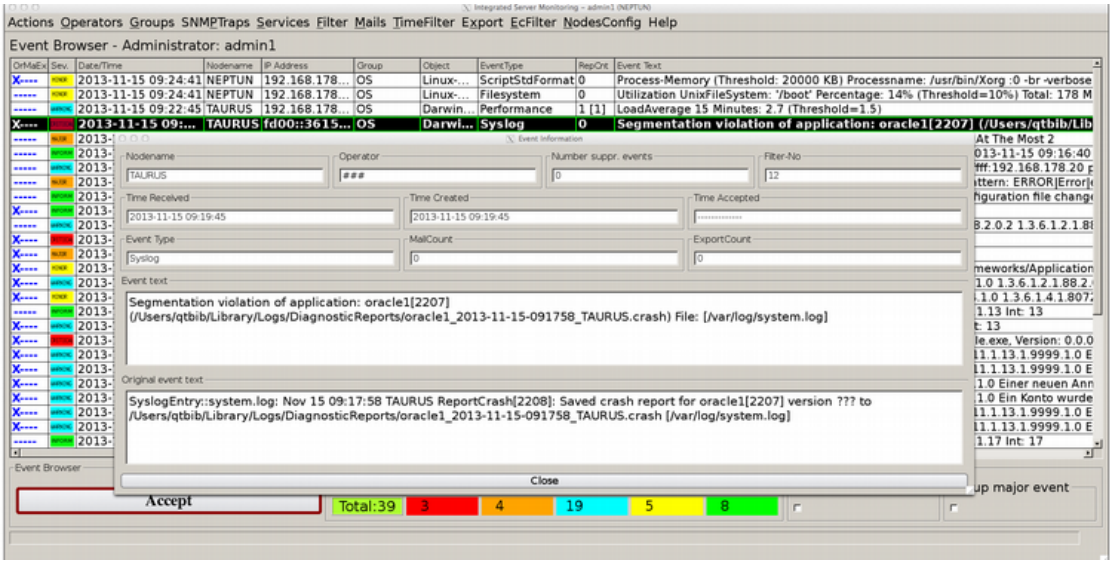
Es findet die sukzessive Auswertung des Zuwachses der Zieldatei während des Polling Intervalls statt. Dieses ist der Normalfall der Logfileauswertung.

Totale Auswertung:

Die Protokolldatei wird immer von Anfang bis Ende betrachtet. Die Auswertung erfolgt, wenn sich die Modifikationsszeit der Datei geändert hat.

Mit dieser Art der Auswertung lassen sich zum Beispiel System- und Konfigurationsdateien wie “/etc/hosts“, “/etc/exports“, “/etc/sshd_config“ usw. auf unerwünschte oder verdächtige Einträge untersuchen. Gleichzeitig bekommt man mit, wenn die Dateien geändert worden sind. Darüber hinaus lässt sich die gewollte Abwesenheit von bestimmten Systemdateien (zum Beispiel “/etc/hosts.equiv“) überwachen; d.h. wenn jemand die Datei anlegt, erfolgt eine Signalisierung.

Analoges gilt für Windows.



Das Bild zeigt eine Meldung aus der Überwachung der System-Protokolldatei "/var/log/system.log" von Mac OS X. Das Suchmuster (Suchargument), das die Meldung hervorrief, ist "[Ss]aved crash report". Die gefundene Zeile, also das Suchergebnis, ist durch einen Filter mit einem Formatstring an der Management-Station in eine kurze, lesbare Form gebracht worden. Durch Doppelklick auf die Meldung bekommt man das Widget „Event Information" mit dem ursprünglichen (originalen) Meldungstext.

Es handelt sich hier um die inkrementelle Logfileauswertung. Siehe auch Beispiel im Anhang.

8.2 Multiple Logfileauswertung (Unix)

Ein weiterer Agent erlaubt es, nicht nur eine sondern mehrere Protokolldateien, die sich in einem Verzeichnis befinden, zu überwachen. Die Liste der Filter bezieht sich auf alle gefundenen Dateien. Im Gegensatz zum vorherigen Agenten beschreibt man in der Konfigurationsdatei die auszuwertenden Dateien mit einer Liste von Suchmustern („Wildcards") für Dateinamen.

Der Name des Agenten lautet: logdiragent und ist mit verschiedenen Konfigurationsdateien parametrisierbar.

Beispiel einer Konfigurationsdatei für multiple Logfileauswertung: SAP/R3-Tracefiles:

```
destination::192.168.178.21 # Vereinbarung Management-Station
portno::55555 # Port tcp zur Übertragung
attribute::SAPGROUP SAP/R3 # Attribute Group, Object einer Meldung
pollingsecs::30  # run as daemon
logfiledir::/usr/sap/PD3/PWEBMGS11/work  # Verzeichnis der Protokolldateien
files::4[maj]::dev_w? dev_w?? dev_rfc? dev_rfc?? dev_disp dev_?? stderr*
# Spezifikation der Dateinamen im Verzeichnis, „4[maj]“: large files für Dateien > 4 MB
# nachfolgend Liste der Filter für die Auswertung
 filter::"update deactivated;;Verbucherabbruch: $*"-crit # nach “;;” Formatstring
"vb error;;Verbucherabbruch: $*"-crit # Verbucherabbruch
"^Disconnecterror:"-crit "^Sqlerror:"-warn
"^Sevdberror:;;Achtung Datenbankfehler: $*"-maj
"^Profileerror:"-warn "Sharedmemoryerror"-maj "^Stackerror:"-warn
"^Mallocerror:"-maj "^Applicationerror:"-maj "^Inputbuffererror"-maj
"^Speichermangel:"-warn "^Shared Memory"-warn
"update activated"-null # Suppression filter
"Error Code"-min # Jobabbrueche
"Fehler in einer ABAP-Anweisung"-maj
"[Mm]emory exhausted:"-maj
"Error.*in.*application.*program"-min
"CPIC-Error"-null # Suppression filter
“ERROR.*shmctl“-maj “ERROR.*shmget”-maj
“ERROR|Error|error”-warn[-1] “WARNING|Warning|warning”-min[-1]
“FATAL|Fatal|fatal”-maj[3] # Schleppnetz mit unbekannten Zeilen “FATAL…”
# end of configuration file
```

Das Suchmuster (*regular expression*) beginnt mit einem Anführungszeichen ’“’ und endet entweder mit “;;“ oder ebenfalls mit einem Anführungszeichen.

8.3 Multiple rekursive Logfileauswertung (Unix)

Es werden Protokolldateien nicht nur in einem Verzeichnis sondern auch in den darunter liegenden Verzeichnissen überwacht.

Der Name des Agenten lautet: logrecagent und ist mit verschiedenen Konfigurationsdateien parametrisierbar.

9. Agenten für Standardüberwachung

Die Standardüberwachung bietet eine Reihe von fertigen, problemorientierten Funktionen an, die so ausgewählt wurden, dass sie für die ganz überwiegende Anzahl der Nodes relevant und ausreichend sind. Es ist an dieser Stelle weder eine Programmierung noch ein übertriebener Konfigurationsaufwand notwendig. Die Gestaltung der Meldungstexte nimmt der Agent vor, sie können durch die Filter an der Management-Station bei Bedarf angepasst werden. Die vorformulierten Meldungstexte enthalten nicht nur die eingestellten Schwellwerte, sondern auch die ermittelten aktuellen Werte, so dass man Entwicklungen erkennen kann. Die Schwellwerte selbst sind, soweit dies möglich ist, bezogene (prozentuale) Werte, um eine Vergleichbarkeit und Allgemeingültigkeit zu erzielen. Siehe auch Beispiele im Anhang.

9.1 Unix

Die folgenden Funktionen sind in dem Agenten für die Standardüberwachung enthalten:

- **Schwellwertüberwachung Filesysteme:** Man gibt einen allgemeinen Schwellwert für den Belegungsgrad (Quotient aus genutztem zu verfügbarem Speicher) an, der für **alle** aktuellen und neu hinzukommenden Filesysteme gilt. Darüber hinaus kann man für jedes einzelne Filesystem individuelle Schwellwerte festlegen oder es ganz aus der Überwachung herausnehmen. Bei einem individuellen Schwellwert wird auch gleichzeitig die Existenz des Mountpoints geprüft. Bei Überschreitung eines Schwellwertes gibt es für jedes Filesystem eine eigene Meldung, die im Meldungstext die folgenden Informationen enthält: Mountpoint, Schwellwert, Speicher gesamt, noch freier Speicher, Filesystemtyp und Änderungsgeschwindigkeit des Verbrauchs in MB/h. Damit lässt sich verbleibende Zeit bis zur vollständigen Belegung des Datenträgers abschätzen.
- Schwellwertüberwachung Inodes von Filesystemen: Der prozentuale Nutzungsgrad als Quotient von Anzahl gebrauchter Inodes und Gesamtanzahl von Inodes wird für jedes Filesystem überwacht. Der Schwellwert ist konstant 95%.
- **Prozessüberwachung:** Prüfen, ob Hintergrundprozesse aktiv sind. Die Prüfung bezieht sich auf die Existenz als boolschen Wert oder auf die Anzahl von Instanzen. Einzugeben ist eine Liste von Prozessnamen, Suchmuster (*regular expression*) sind erlaubt. Die Suche bezieht sich auch auf die Prozessparameter. Nach Ausfall und anschließendem

Neustart eines Hintergrundprozesses gibt es automatisch eine Meldung, die im Meldungstext die Ausfallzeit (Down time) explizit angibt.

- **Restartfunktion:** Erweiterung der Prozessüberwachung; bei Ausfall eines Hintergrundprozesses (oder mehrerer) wird dieser durch eine Restartprozedur automatisch neu gestartet. Der Vorgang wird an der Management-Station signalisiert.
- Schwellwertüberwachung Anzahl Zombieprozessen. Es sind zwei Schwellwerte + Severity einstellbar (z.B. >=100 „warn“ und >=200 „crit“).
- **Überwachung von Syslogdateien:** Inkrementelle Auswertung einer oder mehrerer Protokolldateien. Es kann neben der offiziellen Syslog-Datei noch andere geben wie z.B. „secure.log“, „daemon.log“, „kernel.log“. Zulässig sind auch andere Protokolldateien wie die von Web-Servern, Datenbankverwaltungssystemen oder Anwendungen.
- Schwellwertüberwachung Größe Syslogdatei: Wenn die Größe der Syslogdatei einen bestimmten Wert in MB überschreitet, gibt es eine Meldung. Einstellbar ist der Wert + Severity. Allgemeine Überwachung von Dateigrößen („large files“) erfolgt bei der Logfileauswertung.
- **Überwachungsprogramme:** Aufruf und Auswertung von einem oder mehrerer Überwachungsskripte bzw. ausführbarer Programme. Die Auswertung bezieht sich auf die Ausgabe des Programms, die genauso gefiltert wird wie der Inhalt einer Protokolldatei (siehe auch Überwachungsskripte).
- **Listenports:** Prüfen von lokalen Tcp-Ports gegen das loopback-interface. Eingabe einer Liste von Portnummern (z.B. 22, 80, 443). Bei dieser Prüfung entfällt die Zweideutigkeit (Netz oder Server), die bei einer Prüfung über das Netz zwangsläufig auftritt. Nach Ausfall und wieder Erreichbarkeit gibt es eine informative Meldung mit Angabe der Ausfallzeit.
- Schwellwertüberwachung CPU-Auslastung: Einstellbar sind zwei prozentuale Schwellwerte + Severity für kurzzeitige Belastung sowie durchschnittliche Belastung über einen ebenfalls konfigurierbaren Zeitraum, z.B. 120 Minuten.
- Schwellwertüberwachung Swap: Quotient aus belegtem zu maximal verfügbarem Auslagerungsspeicher in Prozent. Einstellbar sind zwei prozentuale Schwellwerte + Severity.
- **Schwellwertüberwachung Ladefaktor (load average) 15 min:** Man kann zwei Schwellwerte + Severity einstellen (z.B. >10 „warn“, >50 „crit“).
- Reboot und Startup: Meldung an der Management-Station bei diesem Ereignis.

- **Lifecheck (Heartbeat):** Dynamische Registrierung des Nodes an der Management-Station. Nach Ausfall des Servers, etwa durch Wartungsarbeiten oder durch eine größere Störung, wird die Ausfallzeit an der Management-Station explizit angegeben.

Anzumerken ist noch, dass bei der wichtigen Filesystemüberwachung nicht jeder Filesystemname einzeln eingegeben werden muss, was bei einer Anzahl von ein paar Dutzend (oder mehr) ein erhebliches Problem und bei neu hinzukommenden unmöglich wäre.

Der Name des Agentenprogrammes lautet für alle Unix-Derivate: basemonagent. Es ist mit verschiedenen Konfigurationsdateien parametrisierbar.

9.2 Windows

Die folgenden Funktionen sind in den Agenten für die Standardüberwachung enthalten:

- **Schwellwertüberwachung Filesysteme (Drives):** Wie bei Unix die Filesysteme. Eingabe eines allgemeinen Schwellwertes, ab dem gemeldet wird. Betrachtet werden die lokalen Laufwerke. Ergänzend kann jedes Drive mit einem individuellen Schwellwert + Severity versehen werden. In diesem Fall gibt es für die Nicht-Existenz des/der Laufwerke eine kritische Meldung. Der zweite Schwellwert von 98% verursacht eine kritische Meldung. Ausgabe erfolgt analog zu Filesystemen bei Unix.
- **Überwachung Tasks**: Überprüfung der Existenz von Tasks. Eingabe einer Liste von Task-Namen. Überprüfbar sind auch die Anzahl Instanzen einer Task.
- **Überwachung Services (Dienste):** Feststellen der Existenz von aktiven und registrierten Services. Eingabe einer Liste von Service-Namen.
- **Überwachungsprogramme:** Aufruf und Auswertung der Textausgabe von einem oder mehreren Überwachungsskripten (siehe auch Überwachungsskripte)
- **System Event Log:** Logfileauswertung für Störungen des Betriebssystems. Suchargument ist die Event-Id und/oder die Microsoft-Eventtyp „error“ und/oder „warning“. Jede Event-Id kann mit einer individuellen Severity versehen werden. Ein oder mehrere Event-Ids können

unterdrückt werden. Im Meldungstext enthalten sind auch die Message-Strings.

- **Application Event Log:** Logfileauswertung für Ereignisse bei Anwendungen.
- **Security Event Log:** Logfileauswertung für Vorkommnisse bei der Systemsicherheit.
- **Listenports:** Prüfen von lokalen Tcp-Ports gegen das loopback-interface. Eingabe einer Liste von Portnummern (z.B. 22, 80, 135, 443). Nach Ausfall und wieder Erreichbarkeit gibt es eine informative Meldung mit Angabe der Ausfallzeit.
- Schwellwertüberwachung CPU-Auslastung: Einstellbar sind zwei Schwellwerte für eine kurzzeitige Belastung und für eine durchschnittliche Belastung über einen ebenfalls einstellbaren Zeitraum (z. B. 60 Minuten). Es werden alle CPUs einzeln überwacht.
- Schwellwertüberwachung Memory: Ist definiert als Quotient aus belegten zu gesamten Speicherplatz. Es gibt zwei einstellbare Schwellwerte + Severity (z.B. 90% + „warn“, 99% + „crit“). Bei Überschreiten der Schwellwerte wird im Meldungstext auch der Gesamtspeicherplatz in MB angezeigt.
- Schwellwertüberwachung Swap (Page): Ist definiert als Quotient aus belegter zur maximalen Größe des Pagefiles. Es gibt zwei einstellbare Schwellwerte + Severity (z.B. 95% + „maj“, 99% + „crit“). Bei Überschreiten der Schwellwerte wird auch die maximale Größe des Pagefiles an der Management-Station ausgegeben.
- Reboot und Startup des Betriebssystems werden der Management-Station gemeldet.
- **Lifecheck:** wie Unix

Der Name des Agentenprogrammes lautet: winmonagent.exe. Siehe auch Beispiel im Anhang.

10. Agenten für Überwachungsskripte

Agentenprogramme für die Ausführung und Auswertung von Überwachungsskripten zur Realisierung spezieller Anforderungen. In den Programmen können auch Anweisungen zur Fehlerbehandlung enthalten sein.

10.1 Unix

Zur Ausführung von Überwachungsskripten aller Art stehen die Programme scriptmonagent und asyncmonagent zur Verfügung, die jeweils mit verschiedenen Konfigurationsdateien parametrisierbar sind.

Es können beliebige ausführbare Programme bzw. Skripte eingesetzt werden, die eine Textausgabe nach *stdout* und/oder *stderr* haben. Die Ausgabe wird nach dem gleichen Muster wie bei der Logfileauswertung gefiltert. Pro gefilterte Textzeile wird ein Event erzeugt und zur Management-Station geschickt.

In einer Konfigurationsdatei können mehrere Programme plus Filter vereinbart werden. Die Skripte können mit Eingabeparametern versehen werden. Wenn der Exit-Code eines Skriptes ungleich Null ist, interpretiert der Agent das als fehlerhaftes Verhalten und schickt eine gesonderte, kritische Meldung zur Management-Station. Für den Fall einer Blockierung gibt es einen Timeout (Voreinstellung: 30 Sekunden) und ebenfalls eine Ausnahmemeldung. Ebenso gibt es eine Fehlermeldung, wenn das angegebene Skript nicht ausführbar ist.

Nachfolgend eine Konfigurationsdatei als Beispiel, es soll der Speicherverbrauch von Prozessen mit zwei Schwellwerten überwacht werden.

```
destination::192.168.178.21 # Management-Station
portno::55555 # Uebertragungsport
attribute::Linux Ueberwachungsskripte # Group Object einer Meldung
pollingsecs::600 # Betrieb als Hintergrundprozess
cmd::ps -efly | awk 'BEGIN {lim1=100000;lim2=500000} {if((NR > 1) && (NF >= 14)){
if($8 >= lim2){print "Lim2:",lim2,$0
} else if( $8 >= lim1 ){
print "Lim1:",lim1,$0
}}}'::"^Lim1:;;Process-Memory (Threshold: $2 KB) Usage: $10 KB, Processname: %15"-min[10]
"^Lim2:;;Process-Memory (Threshold: $2 KB) Usage: $10 KB, Processname: %15"-maj[10]
"*;;Systemfehler: $*"-crit[1]
#cmd:: ... weitere Befehle zum Auswerten
#end of configuration file
```

Nach dem Schlüsselwort cmd:: folgt der auszuführende Befehl (“ps -efly“, Linux), der über eine Pipe ’|’ mit dem Reportgenerator “awk“ bearbeitet wird.

Danach kommt die Liste mit den Filtern, die die Ausgabe pro Zeile formatiert und je nach überschrittenem Schwellwert zwei verschiedene Severities vergibt. Für jeden Prozess, der den ersten oder zweiten Schwellwert überschritten hat, gibt es eine eigene Meldung, die im Event-Text den Schwellwert, den aktuellen Wert und den Prozessnamen enthält. Natürlich ist es auch möglich, den Befehl und die (numerische) Auswertung in einem Programm zu realisieren, das dann in der Konfigurationsdatei vereinbart wird. (Die gleiche Funktion kann auch mit dem Agenten für die Standardüberwachung basemonagent realisiert werden.)

Weitere Beispiele für den Einsatz von Überwachungsskripten sind Ermittlung von Füllgrad der Tablespaces und Extents bei Datenbankverwaltungssystem Oracle. Bei Informix sind es die DB-Spaces und die logischen Logs.

Das Agentenprogramm asyncmonagent ist eine Erweiterung der Programme scriptmonagent **und** logmonagent. Jedes in der Konfigurationsdatei vereinbarte Skript und/oder jede Protokolldatei wird mit einem eigenen Polling Intervall versehen. Die Ausführung der Überwachungsfunktionen geschieht nebenläufig in Form von Threads. Der Agent wird ausschließlich als Hintergrundprozess betrieben.

CRONJOBS (SCHEDULER) UND DAEMONS:

Mit dem Agentenprogramm asyncmonagent können auch Cronjobs realisiert werden. Die Handhabung der Zeiten geschieht wie bei der Einrichtung crontab() von Unix.

Beispiel:

```
destination::168.178.20.21
portno::55555
attribute::Linux AsyncMonitorCollection

#logfile::[name::]<crontabspec::<filename>::filterliste
#total::[name::]<crontabspec>::<filename>::filterliste
#cmd::[name::]<crontabspec>::<command>::filterliste
cmd::0 10 * * 1-5::echo “It is ten o’clock in the morning (mo-fr) `date`“::“*“-info
# check disk space
cmd::15,45 6-20 * * *::df -k::“[ ]100%[ ];;FS full: $*“-crit “[ ]9[5-9]%[ ]“-maj
“[ ]9[0-4]%[ ]“-warn“[ ]8[7-9]%[ ]“-min “[ ]8[2-6]%[ ]“-info
cmd::5::journalctl -f -q --since=-2m -p crit::“*“-crit # schnellste Art der Signalisierung
# “journalctl -f“ startet nach 5 Sekunden und terminiert nicht
# ... weitere Funktionen und Instanzen
```

SNMP:

Mit den Skriptagenten lassen sich SNMP-Abfragen auf dem lokalen Server (Domainname: localhost) durchführen. Dazu nimmt man die Unix-Befehle snmpget, snmpwalk, snmpdelta, snmpbulk und filtert die Ausgabe. Das Ergebnis gelangt über den normalen Tcp-Port in den Event Browser. Dadurch erspart man sich dauernde Abfragen über das Netz.

Direktmeldungen:

Darüber hinaus gibt es den Befehl rsendmsg, mit dem man Meldungen direkt zur Management-Station schicken kann. Er kann für Skripte jeglicher Art (Shell, Perl, etc) verwendet werden.

10.2 Windows

Es gibt für diesen Zweck die Programme scriptmonagent.exe und asyncmonagent.exe, die die gleichen Funktionen wie bei Unix haben.

Beispiel: Auswertung des Befehls “netstat -an“

```
destination::192.168.178.21
portno::55555
attribute::Windows
pollingsecs::120
cmd::netstat -an::“^[ ]*UDP/v“-null  “127\.0\.0\.1|\[\::1\]“-null
“:(68|13[78]|500|1900|3702|4500|4919[23]|515[45][1-9]|5855[01])[ ]/v;;Udp-Port: %1“-warn[3]
#cmd::... weitere Befehle
```

Die Auswertung geschieht mit einer Positiv-Liste von zulässigen Ports, die in dem dritten Filter enthalten ist. Der erste Filter unterdrückt alle Zeilen, die nicht mit “UDP“ anfangen.

Direktmeldungen:

Analog zu Unix steht für Windows der Befehl rsendmsg.exe zur Verfügung. Er wird mit der Portnummer, der Adresse der Management-Station und den Komponenten der Meldung parametrisiert.

11. Agent für Security (Unix)

Agent zum Zweck der Security, der in kurzen Polling-Intervallen Änderungen am Filesystem meldet.

Der Agent meldet nach der anfänglichen Initialisierung Änderungen der Attribute von Systemverzeichnissen und Systemdateien. Es handelt sich vorzugsweise um solche Dateien und Verzeichnisse, die das Verhalten des Betriebssystems bestimmen und die sich im laufenden Betrieb nur unter besonderen Umständen ändern. Dazu zählen reguläre Installationen oder Updates, aber auch verdeckte Installationen durch ***rootkits*** und andere Versuche der Manipulation. Bei gewollten Installationen kann man Vergleiche zwischen den angekündigten und tatsächlichen Änderungen machen. Wenn in einem Verzeichnis Dateien jedweder Art neu hinzukommen, wird es ebenfalls gemeldet!

Die untersuchten, unterscheidbaren Attribute sind: *inode number*, *size*, *modification time*, *mode/permission, status time*, *user id of owner*, *group id of owner.*

Der Name des Agentenprogrammes lautet secmonagent und ist mit verschiedenen Konfigurationsdateien parametrisierbar.

Beispiel:

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

Vereinbart wird eine Liste von Verzeichnissen und/oder Dateien, die optional mit einer Severity versehen sind. Optional kann man mit dem Schlüsselwort exclude:: eine Liste von Dateinamen und/oder Unterverzeichnissen mit voller Pfadangabe vereinbaren, die von der Überwachung ausgenommen werden sollen.

Die Meldungen des Agenten enthalten im Event-Text die Namen der Verzeichnisse und die in ihnen liegenden Dateien sowie die Art der Änderung.

12. Lifecheck (Heartbeat), dynamische Registrierung

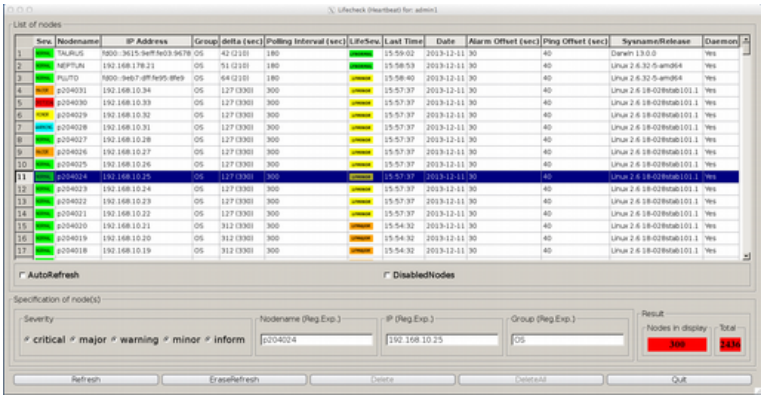
Für den Betrieb der Überwachungs-Agenten braucht man Daten wie Rechnername und IP-Adresse nicht zu vereinbaren, sondern sie werden an der Management-Station automatisch registriert und verwaltet. Die Informationen kommen von den zu überwachenden Servern und werden ständig aktualisiert.

Ein Node wird vom System erfasst, nachdem man die Standardüberwachung für den Server eingerichtet hat. Sie schickt bei jedem Aufruf eine spezielle, nicht sichtbare Steuermeldung, die auch gleichzeitig eine Authentifizierung ist, zur Management-Station und setzt oder erneuert einen Zeitstempel. Das Alter des Zeitstempels wird im Hintergrund laufend überwacht. Sowohl die (erstmalige) Anmeldung als auch das Ausbleiben der Steuermeldung wird nach einer bestimmten, einstellbaren Anzahl von Sekunden im Event Browser signalisiert.

Zusätzlich nimmt die Management-Station automatisch eine Prüfung mit Icmp-Ping vor, wenn sich ein Server nach einer bestimmten, ebenfalls einstellbaren Zeit nicht gemeldet hat. Nach insgesamt dreimaliger Prüfung mit negativem Ausgang und entsprechender Signalisierung geht der betreffende Server automatisch in den Zustand „disabled“ über. Es gibt dann keine Meldungen mehr. In diesem Zustand verbleibt er, bis er sich erneut anmeldet. Bei einer Wiederanmeldung erfolgt eine informative Meldung mit der Angabe der Ausfallzeit im Meldungstext.

Mit diesem Automatismus hat man eine Kontrolle über die Netzverbindung und den eventuellen Ausfall des ganzen Servers.

Die Menge der in Überwachung befindlichen Server kann man sich in der X11- und Web-Oberfläche auflisten lassen. Sie ist kombiniert mit einer **Statusanzeige**, die für jeden Node die höchste Severity einer Meldung im aktiven Browser anzeigt. Wenn sich ein Nodename mit einer anderen IP-Adresse als zuvor meldet, gibt es automatisch eine Warning im Event Browser. Das kann durchaus passieren bei Servern mit mehr als einer Netzwerkkarte. Ein störungsfrei laufender Server, der keine Meldungen hervorruft, ist in dieser Sicht durch den periodisch sich erneuernden Zeitstempel und eine grüne Statusmeldung zu sehen (linke Spalte des nachfolgenden Bildes).



Das Bild zeigt die Liste der Nodes, die mit den Agenten von dieser Management-Station überwacht werden. Es ist bei jedem Server zu erkennen, wann er sich das letzte Mal gemeldet hat, der Abstand in Sekunden von der aktuellen Zeit und das Polling Intervall. Die Eingabefelder im unteren Bereich sind Suchfelder, da die Liste sehr lang werden kann. Mit der Checkbox „DisabledNodes“ werden die betreffenden Server getrennt aufgeführt.

Hinweis: Bei herkömmlichen Systemen dieser Art muss der Administrator Hostname und IP-Adresse auf der Management-Station manuell eingeben. Sie sind von da ab quasi Systemkonstanten und bilden die Vorraussetzung für eine Überwachung. Das Problem ist weniger der Arbeitsaufwand (plus Möglichkeit zur Falscheingabe) als vielmehr die Tatsache, dass Hostname und IP-Adresse an anderer Stelle schon vereinbart sind, und zwar im Original. Es ist auch keine Seltenheit, dass ein Server mehr als eine Adresse hat. Hinzu kommt noch die Abhängigkeit von DNS-Einträgen. Zu einem ernsthaften Problem wird eine starre Vereinbarung bei bestimmten Serverarchitekturen (zum Beispiel Hochverfügbarkeitssysteme), die eine dynamische Zuordnung von Adressen und Servernamen vorsehen.

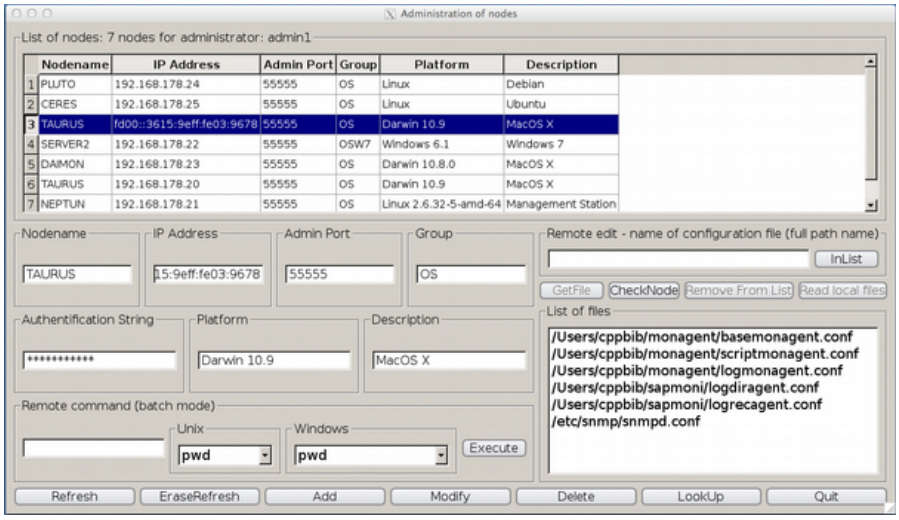
13. Konfiguration der Agenten, Kommando-Interface

Die graphische Bedienoberfläche bietet die Möglichkeit, die Konfigurationsdateien der Nodes zentral zu verwalten und zu editieren. Dazu kommuniziert sie über einen Port udp/tcp mit einem Hintergrundprozess des Servers. Die Namen der Konfigurationsdateien werden auf der Management-Station gehalten, deren Inhalt dann über das Netz editierbar ist. Es findet keine redundante Datenhaltung statt. Nach Bearbeitung wird die jeweilige Datei zurück gespeichert. Es gibt keine funktionale Abhängigkeit zu der Kommunikation der Agenten.

Zusätzlich gibt es die Möglichkeit, Konfigurationsdateien lokal auf der Management-Station abzuspeichern und existierende Dateien einzulesen, die dann auf einen beliebigen Node verteilt werden können.

Darüber hinaus besteht die Möglichkeit, über ein Kommando-Interface für die Operatoren Befehle zu administrativen Zwecke (zum Beispiel “ps -ef“, “df -k“, “netstat“, usw.) auf einen Node auszuführen. Der Text der Befehlsrückgabe erscheint in einem extra Widget. Siehe auch Beispiel im Anhang.

Die Rechte für die Konfiguration liegen beim Administrator sowie bei den Operatoren, denen ein Server zugewiesen ist.



Das Bild zeigt rechts unten die Namen der Konfigurationsdateien eines selektierten Servers. Mit Doppelklick auf die linke Spalte der oberen Liste kommt man direkt auf das Kommando-Interface des betreffenden Servers und kann Befehle absetzen.

Der Name des Hintergrundprozesses ist für Unix: remoteconfd, und Windows: remoteconfd.exe.

Sicherheit: Die Interaktion ist besonders geschützt

- Authentifizierung durch geheimen Schlüssel, IP-Adresse der Management-Station und/oder „Authentification String“; Schutz vor Replay-Angriffen gewährleistet
- Verschlüsselung durch AES mit Blockverkettungsmodus (CBC) und *session key*
- Integritätsprüfung durch Checksummen

14. Benutzerverwaltung

Um das System zu benutzen muss man sich mit einer Benutzerkennung und einem Passwort anmelden. Die Benutzerverwaltung ist Aufgabe eines Administrators. Dazu gibt es drei Eingabemasken an der graphischen Bedienoberfläche.

1. Eintrag der Benutzer, hierbei wird der Name und das Anfangspasswort eingetragen und bestimmt, ob es sich um einen Operator oder Administrator handelt
2. Eintrag der Gruppennamen
3. Zuweisung von Gruppen an einen Operator

Es können bis zu 255 Operatoren und/oder Administratoren angelegt werden. Die Anzahl der Gruppen ist nicht limitiert.

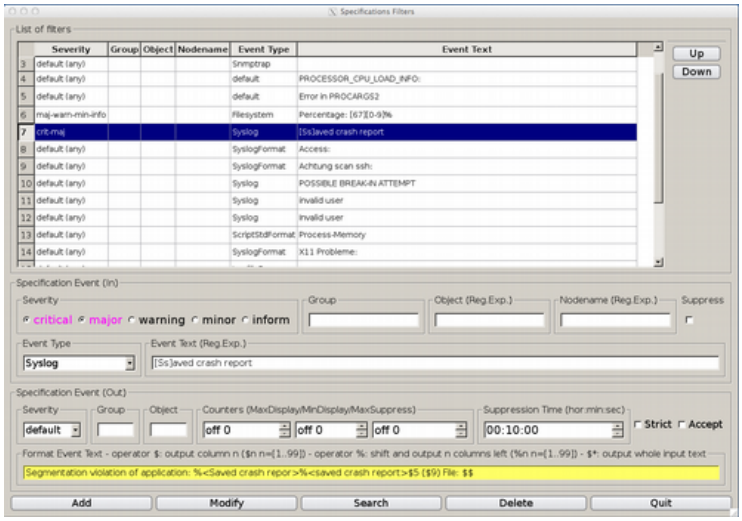
Ein Administrator sieht alle Meldungen, ein Operator nur die Meldungen der ihm zugewiesenen Gruppen.

15. Der Filtermechanismus (Management-Station)

Integraler Bestandteil dieses System ist die Filterung ankommender Meldungen an der Management-Station, die dazu dient, nicht relevante oder sich immer wiederholende Informationen vom System fernzuhalten und/oder nicht mehr aktuelle Meldungen automatisch zu archivieren. Die Einstellungen dafür nimmt ein Administrator an der graphischen Bedienoberfläche im laufenden Betrieb vor.

15.1 Vorfiltermechanismus

Der Vorfiltermechanismus ist vor der Anzeige und dem Abspeichern einer Meldung angesiedelt. Events, die unterdrückt werden, gelangen nicht in die Datenbasis. Wichtig ist der häufig auftretende Fall von ähnlichen Meldungen, die von der gleichen Meldungsquelle kommen und sich von den Attributen her nur im Meldungstext voneinander unterscheiden. Der Eingangstext variiert zum Beispiel bei Performance-Daten oft nur in Zahlenwerten. Den Grad der Ähnlichkeit kann man durch die Verwendung von Suchmustern (*regular expression*) bestimmen. Daneben lässt sich durch die Verwendung einer Formatanweisung der Event-Text verändern.



Die Abbildung zeigt die graphische Eingabemaske für die Filter. Im oberen Teil sind die schon vorhandenen Filter. Im Bereich darunter (*„Specification Event (In)“)* wird eine eingehende Meldung in ihren Attributen beschrieben. Freigelassene Eingabefelder (oder “default“) bewirken ein positives Vergleichsergebnis für dieses Attribut. Mit der Checkbox *„Suppress“* lassen sich Meldungen unbefristet unterdrücken.

Im unteren Teil der Maske (*„Specification Event (Out)“)* kann man optional die ausgehende Meldung bestimmen. Es lassen sich *„Severity“*, *„Group“*, *„Object“* und *„Event Text“* neu bestimmen bzw. umdefinieren. Mit dem Eingabefeld *„Suppression Time (hor:min:sec)“* gibt man die Dauer der zeitlichen Unterdrückung von gleichen Meldungen an. Beim Unterdrücken wird ein Zähler bei der betroffenen Meldung um eins erhöht, der in der Spalte „RepCnt“ des Event-Browsers in eckigen Klammern dargestellt wird.

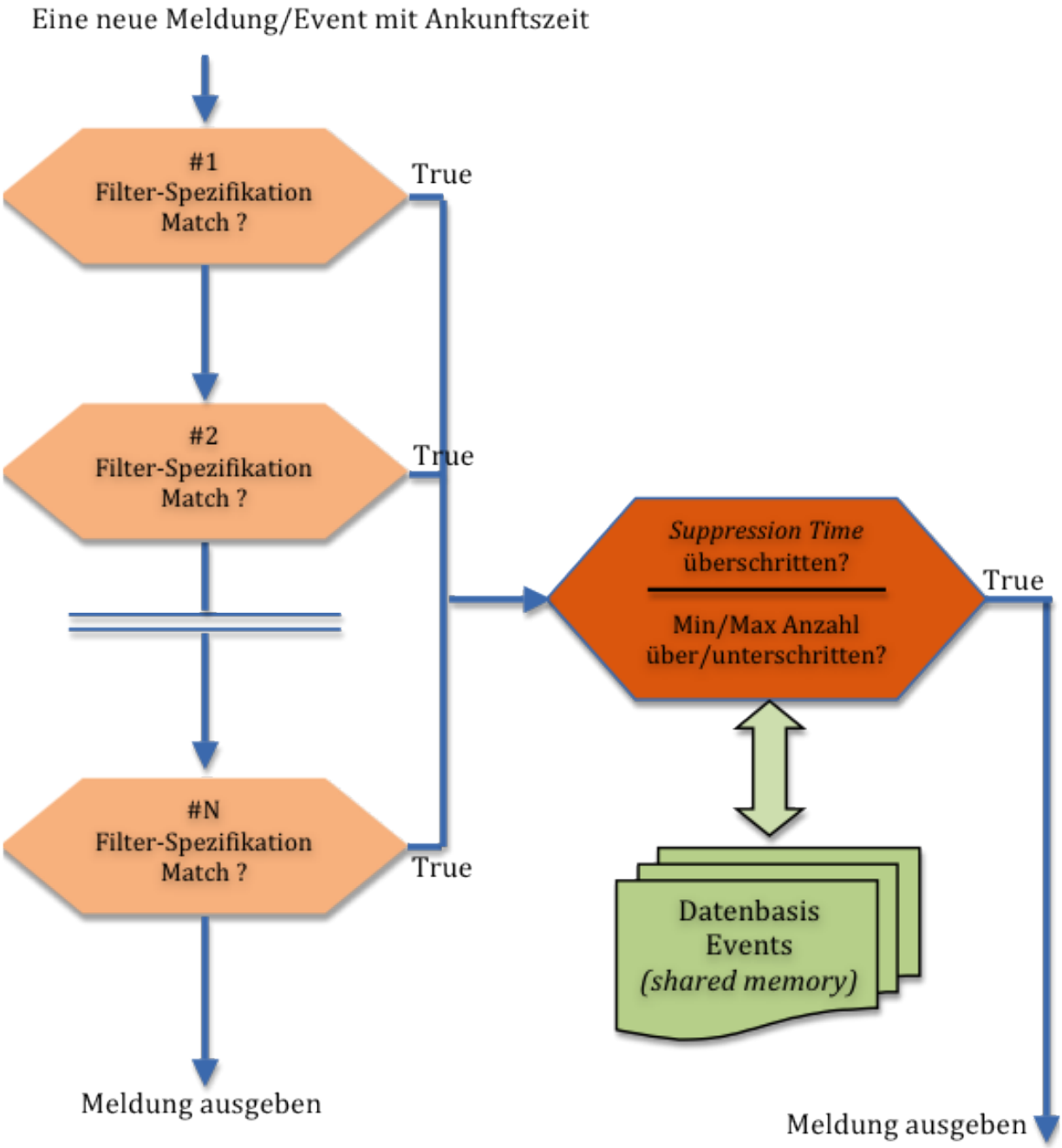
Wenn in der Beschreibung der Eingabe alle Felder leer bleiben oder “default“ sind und alle Severities gesetzt sind, hat das die folgende Wirkung: Jede Meldung, die auf diesen Filter trifft, wird bei erneutem Auftreten für den eingestellten Zeitraum unterdrückt. Diese allgemeine Beschreibung einer ankommenden Meldung kann man schrittweise verfeinern, indem man entsprechende Filter davor setzt. Es handelt sich hier um einen selbstregelnden Mechanismus, der auch bisher unbekannte Meldungen erfasst.

Wichtig: Betroffen sind nur Meldungen jeweils gleichen Inhalts, die in einem bestimmten Zeitraum mehrmals auftreten, während unterschiedliche Meldungen angezeigt werden (**Differenzenanzeige**).

Die Checkbox *„Strict“* schaltet den Vergleich der Meldungstexte aus. Es werden dann auch ähnliche Meldungen unterdrückt. Mit den Spinboxes in dem Feld *„Counters“* kann man innerhalb des eingestellten Zeitraumes die Unter-

drückung von gleichen oder ähnlichen Meldungen von der Anzahl her beschränken.

Bei einer eintreffenden Meldung wird die Liste der Filter (von oben beginnend) durchlaufen. Wenn sie mit der Beschreibung übereinstimmt, gibt es eine Behandlung und der Vorgang terminiert. Es ist zu beachten, dass das Ergebnis der Filterung von der Reihenfolge der Filter abhängt. Sie sind so anzuordnen, dass die speziellen vor den allgemeinen stehen.



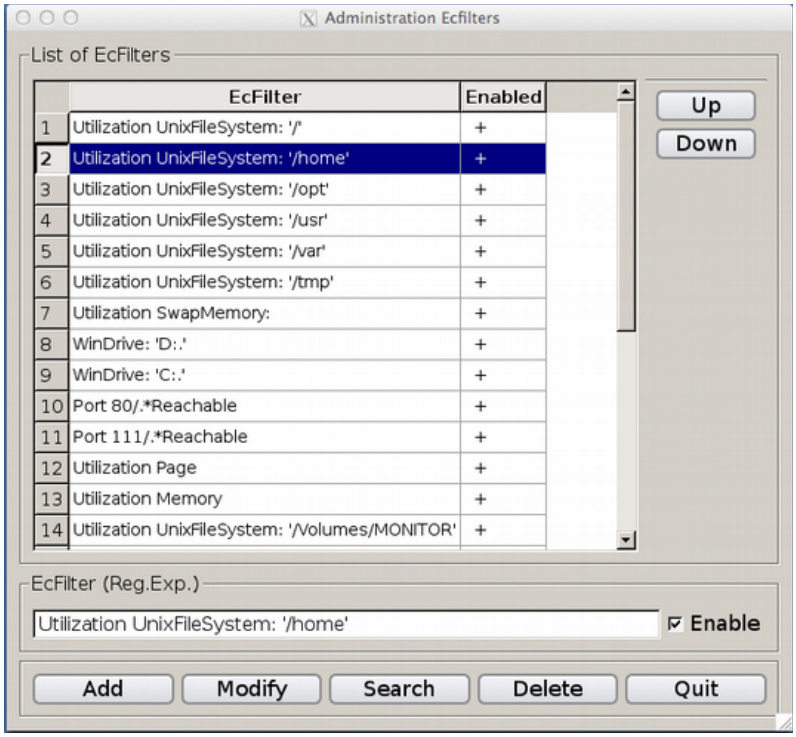
Das Diagramm zeigt noch einmal das Prinzip der Verarbeitung bei einer Anzahl von N Filtern. Es vollzieht sich in zwei Schritten: Feststellen des Filters durch Vergleich und danach anhand der schon erschienenen Meldungen die Entscheidung, ob angezeigt wird oder nicht. Auf diese Weise ist das System in der Lage, sich dem Meldungsaufkommen dynamisch anzupassen, ohne dass es zu einem Informationsverlust kommt.

Bei allen Transformationen bleibt der ursprüngliche (originale) Meldungstext erhalten und kann mit Doppelklick auf eine Meldung eingesehen werden.

Die Anzahl der Filter ist nicht limitiert. Ist die Liste der Filter leer, wird jede Meldung angezeigt.

15.2 Nachfiltermechanismus (EcFilter)

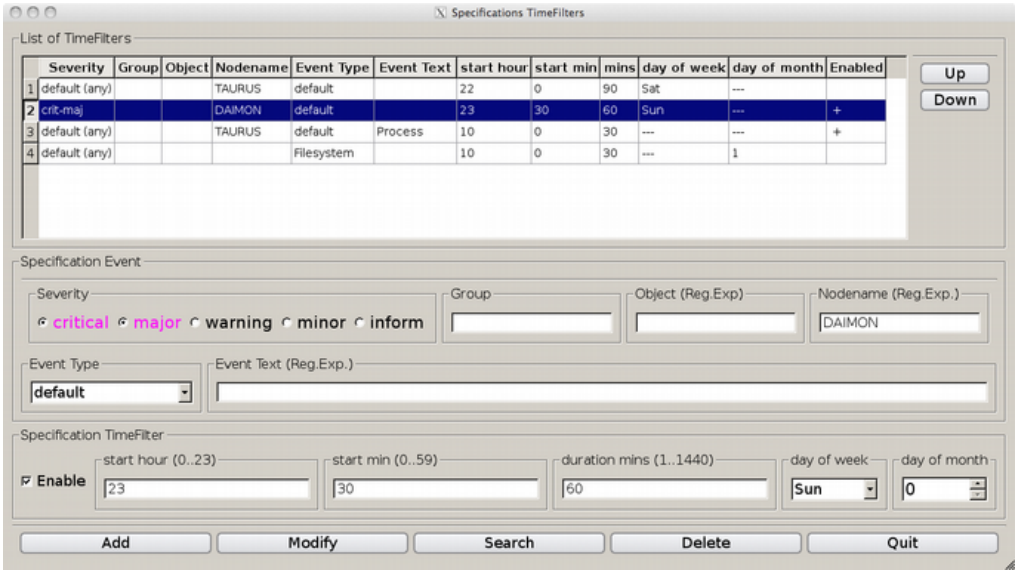
Hier besteht die Möglichkeit, durch eine Liste von Suchmustern ähnliche Meldungen, die noch aktuell sind, automatisch zu archivieren und durch die neu eintreffende Meldung zu ersetzten. Um diesen Vorgang nach außen hin sichtbar zu machen, wird der Wiederholungszähler (“RepCnt“) der entsprechenden Nachricht um eins erhöht. Identisch gleiche Meldungen werden immer so behandelt.



Die Abbildung zeigt die graphische Eingabemaske. Einzugeben ist eine Stringkonstante oder ein Suchmuster (*regular expression*). Damit werden der Reihe nach die Meldungstexte einer neuen Nachricht und bestehender Events verglichen und damit ein *event correlation* hergestellt.

15.3 Timefilter (Scheduler)

Hier findet eine Unterdrückung von Meldungen in einem festen Zeitraster statt. Das gilt zum Beispiel für Wartungszeiten, die an einem bestimmten Tag der Woche oder Monat in einem bestimmten Zeitraum vorgenommen werden. Ein anderes Beispiel sind Betriebszeiten nur an Wochentagen von 6 bis 20 Uhr. Grundsätzlich wird aber davon ausgegangen, dass ein Server rund um die Uhr durchläuft.



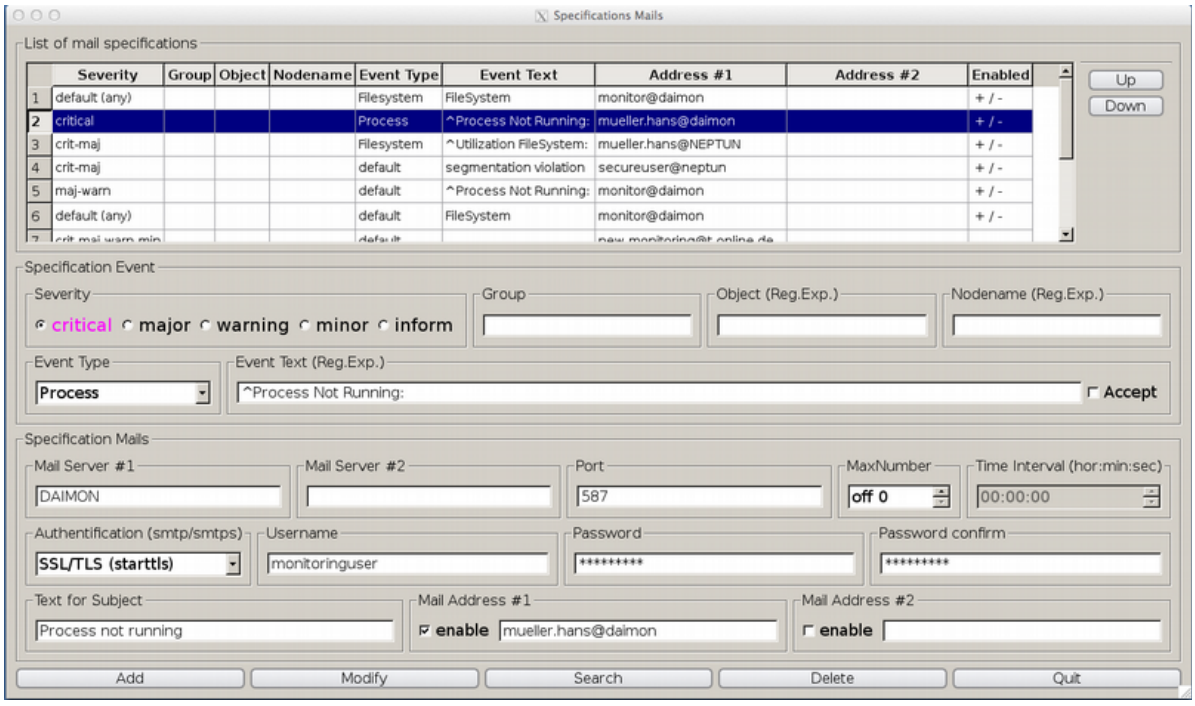
Das Bild zeigt die Eingabemaske der TimeFilter. Im oberen Teil sind die existierenden Filter gelistet. Im mittleren Teil spezifiziert man Meldungen in ihren Attributen („*Severity*“, „*Group*“, „*Object*“, „*Nodename*“, „Event Type“, „*Event Text*“). Im Eingabebereich darunter definiert man den Zeitraum, in dem man einen Zeitpunkt pro Woche oder Monat und die Dauer in Minuten festlegt.

16. Weiterleitung von Meldungen

Aufgelaufene Meldungen können durch eine spezielle und eine allgemeine Einrichtung in Realzeit weitergeleitet werden. Es sei darauf hingewiesen, dass die Anzeige und Speicherung von Meldungen und deren Eskalation zwei verschiedene Vorgänge sind (das eine kann das andere nicht ersetzen).

16.1 Weiterleitung durch E-Mails

Mails mittels SMTP oder SMTPS (*secure smtp*) können direkt und ohne Programmierung erzeugt werden. Man spezifiziert eine ankommende Meldung und fügt Mailadresse(n), Mailserver, optional Ausweichserver und Headertext hinzu. Im Fall von *secure smtp* gibt man noch Benutzerkennung und Passwort des Postfaches auf dem Mailserver an. Mit den Eingabefeldern „MaxNumber“ und „Time Interval“ lässt sich die Anzahl der abgehenden Mails in einem Zeitraum begrenzen. In der E-Mail wird die Meldung als Text komponentenweise dargestellt.

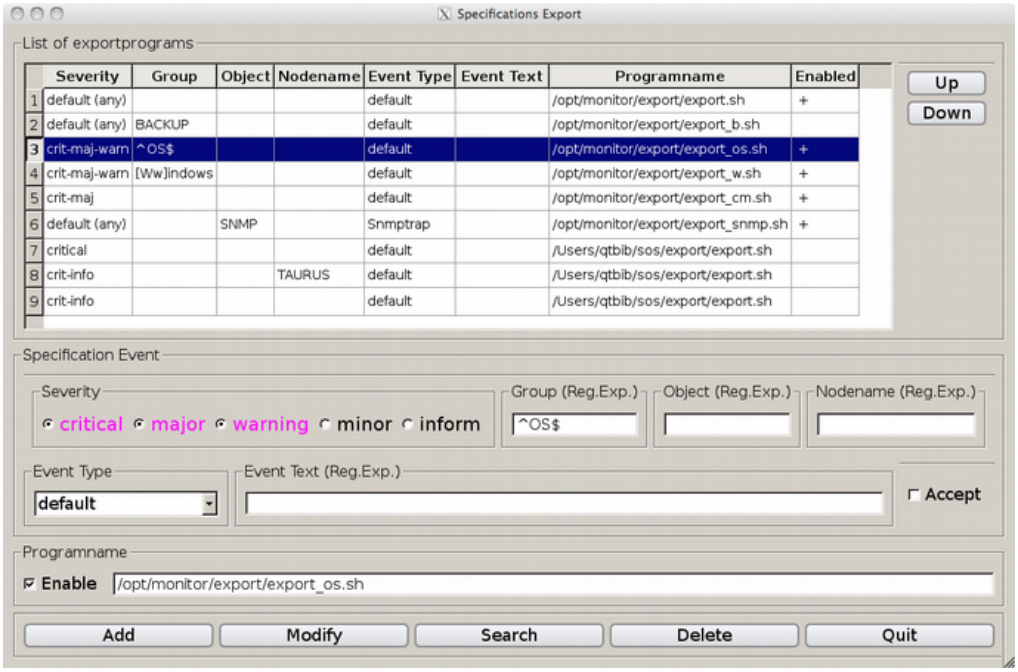


Die Abbildung zeigt die Spezifikationen für das Verschicken von Mails. Eine Meldung kann zu beliebig vielen Adressen geschickt werden.

16.2 Export (Automatische Aktionen)

Das System kann Meldungen, die in der graphischen Oberfläche spezifiziert sind, zu einer Programmschnittstelle herausreichen. Das verarbeitende Programm wird ebenfalls in der Bedienoberfläche angegeben. Die Meldung selbst wird komponentenweise als Stellungsparameter dem Programm übergeben. Diese Funktionalität wird oft auch als „automatische Aktion“ bezeichnet.

Durch diesen Mechanismus kann man auch Meldungen zu einer anderen Management-Station übertragen. Die Übertragung kann wahlweise mit tcp oder udp geschehen.



Das Bild zeigt die graphische Bedienoberfläche für den Export von Meldungen.

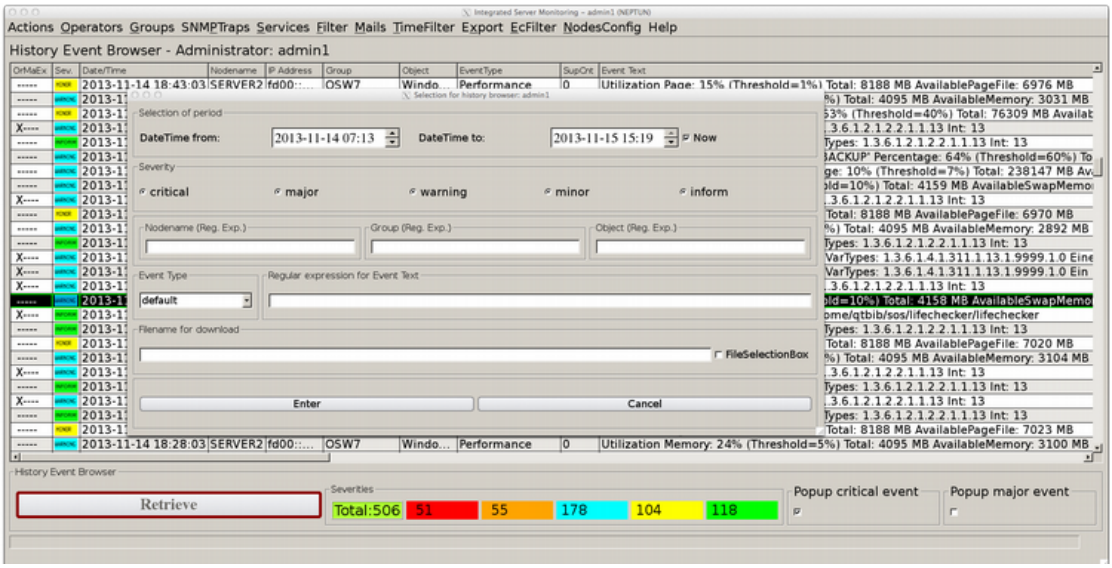
Wenn das vereinbarte Programm einen Return-Code ungleich Null zurück liefert, wird dies als Fehler angesehen. In der ersten Spalte des Event Browsers (“OrMaEx“) ist zu sehen, ob für eine Meldung eine Mail abgeschickt wurde und/oder ob ein Export erfolgte. Bei erfolgreicher Weiterleitung erscheint der Buchstabe ’X’, im Fehlerfall der Buchstabe ’E’ in roter Farbe.

17. History Daten (Reports)

Nach dem Erscheinen und der Bearbeitung einer Meldung im aktiven X11-Browser oder in der Web-Oberfläche betätigt der Operator oder Administrator den Push-Button “Accept“. Bei diesem Vorgang registriert das System die Zeit und den Name des Operators. Die Meldung ist dann in diesem Ausgabemedium nicht mehr sichtbar, auch nicht für andere Nutzer.

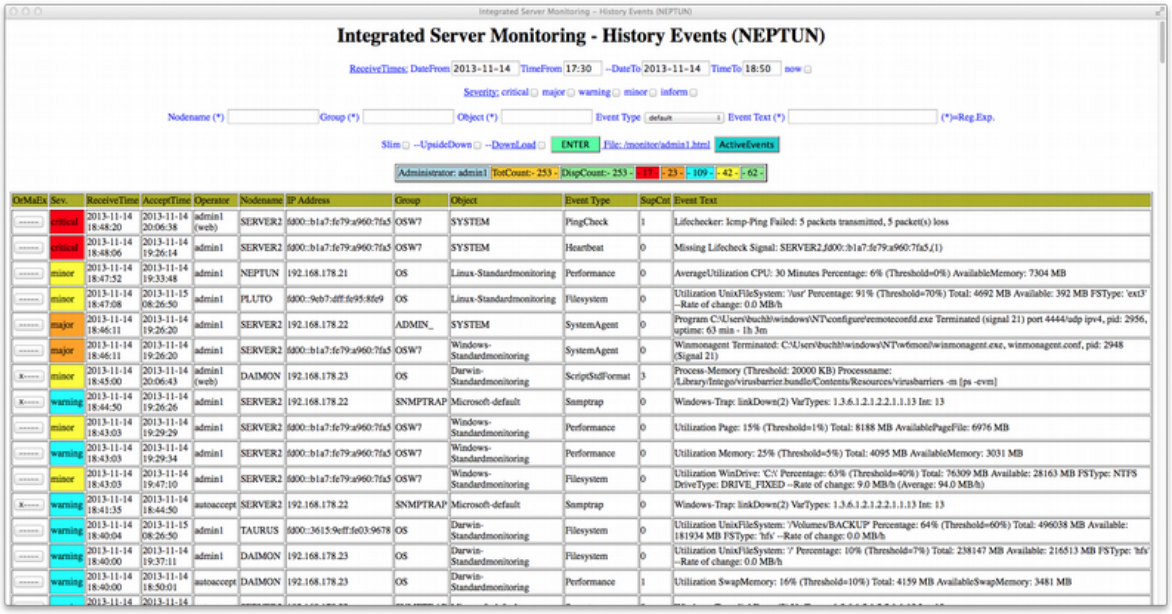
Man hat so eine Kontrolle darüber, wer welches Event zu welchem Zeitpunkt abgeschlossen (quittiert) hat.

Durch eine Auswahlmaske, in der man nach den Attributen eines Events suchen kann, werden die alten Meldungen wieder sichtbar gemacht.



Die Abbildung zeigt die Auswahlmaske und die Darstellung der gefundenen Events im History Event Browser. Zusätzlich kann man den Namen einer Datei angeben, in der die Meldungen zeilenweise abgespeichert werden. Dafür steht als Bedienungselement auch eine File Selection Box zur Verfügung.

Durch Doppelklick einer Zeile bekommt man ein Widget mit den Zeiten für Entstehung, Empfang und Quittierung der Meldung. Mit dem Pusch-Button “Retrieve“ kann eine Meldung wieder in den aktiven Browser gebracht werden.



Analog zur X11-Oberfläche gibt es die Web-Form für die Betrachtung der History-Daten. Der Operator kann den Inhalt direkt in eine Datei auf seinen PC herunterladen. Die Datei enthält ein Event pro Zeile, in der die Komponenten der Meldung als Text dargestellt und durch Semikolon getrennt sind.

Mit den History Daten kann man nicht nur statistische Auswertungen machen sondern man kann neue Störungen mit alten Vorkommnissen vergleichen, um die Fehlerbehandlung zu optimieren. Das gilt auch und gerade für große Datenmengen, wenn die Ereignisse mehrere Jahre zurückliegen.

18. Hintergrundprozesse der Management-Station

Es gibt die folgenden Hintergrundprozesse:

- monlistener: Empfängt die Meldungen von den Agenten, je eine Instanz pro Portnummer; empfängt ferner weitergeleitete Meldungen von anderen Management-Stationen; Empfang erfolgt nebenläufig und asynchron
- lifechecker: Bewerkstelligt das Heartbeat und signalisiert bei Ausfall eines Servers; Abarbeitung erfolgt nebenläufig
- snmptraplistener: Empfängt die SNMP-Traps, voreingestellt für 162/udp4+6; Empfang erfolgt nebenläufig und asynchron, ebenfalls asynchron/nebenläufig die Namensauflösung der IP-Adressen; der Hintergrundprozess ermöglicht eine unlimitierte Anzahl von Clients
- portchecker: Realisiert die aktive Überwachung; Abarbeitung erfolgt nebenläufig
- browserctl: Verwaltet Datenbasis und bewerkstelligt E-Mails und Export von Meldungen; Abarbeitung erfolgt nebenläufig
- udplistener: Empfang von Meldungen anderer Management-Stationen über udp

Die Nebenläufigkeit wird realisiert durch Multithreading. Die Koordinierung der Datenzugriffe erfolgt mit fcntl(), Mutex, Semaphoren.

Alle Programme aus Effizienzgründen in C++ (gilt auch für die Agenten).

19. Urheberrechte

Der Autor des hier vorgestellten Programms ist Wilhelm Buchholz, Im Bruche 6, 31275 Lehrte, bei dem auch die Urheberrechte liegen und in dessen Besitz sich die Quelltexte befinden. Der Autor hat zwanzig Jahre im Systemmanagement gearbeitet, zehn Jahre davon mit verschiedenen Tools auf diesem Gebiet.

Es gibt Agenten für die folgenden Plattformen: Linux, Windows, Mac OS X, AIX, HP-UX, Solaris. Andere Plattformen lassen sich einbeziehen.

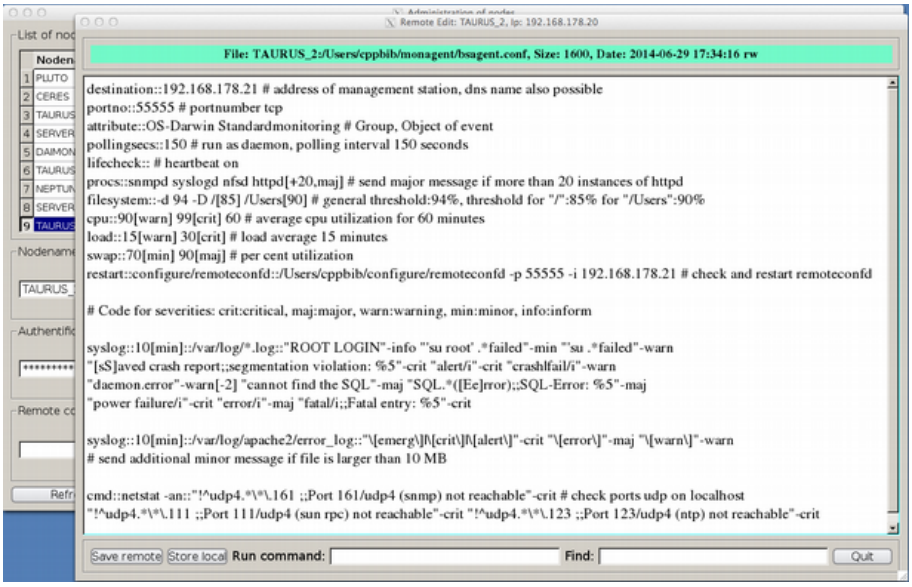
Auf dem Gebiet der Echtzeitüberwachung gibt es als Anbieter die Firma Hewlett-Packard (HP) mit dem hauptsächlichen Produkt HP OpenView Operations (früher OpC, dann ITO, VPO, OVO) und IBM mit dem Produkt Tivoli Monitoring in den verschiedensten Versionen. Erwähnen kann man noch die Firma BMC mit dem Produkt Patrol.

Andere Systeme auf diesem Gebiet, darunter auch Open Source, sind eher uninteressant, weil adäquate Funktionen für Ergebnisdarstellung, Datenhaltung, Mehrbenutzerfähigkeit, Logfileauswertung, Kapazitätsproblem fehlen oder nicht geklärt sind. Gravierend ist das Problem der Kapazität; also was passiert, wenn nicht mehr ein paar Hundert sondern mehrere Tausend Server in der Überwachung sind. Man kann nur vage Aussagen über den Nutzen und die Betriebskosten machen.

Die extrem aufwendigen Systeme von HP und IBM unterscheiden sich von der Masse der anderen im Wesentlichen dadurch, dass sie eine vollständige Logfile-Überwachung anbieten.

20. Abbildungen (Beispiele)

20.1 Beispiel Standardüberwachung Unix



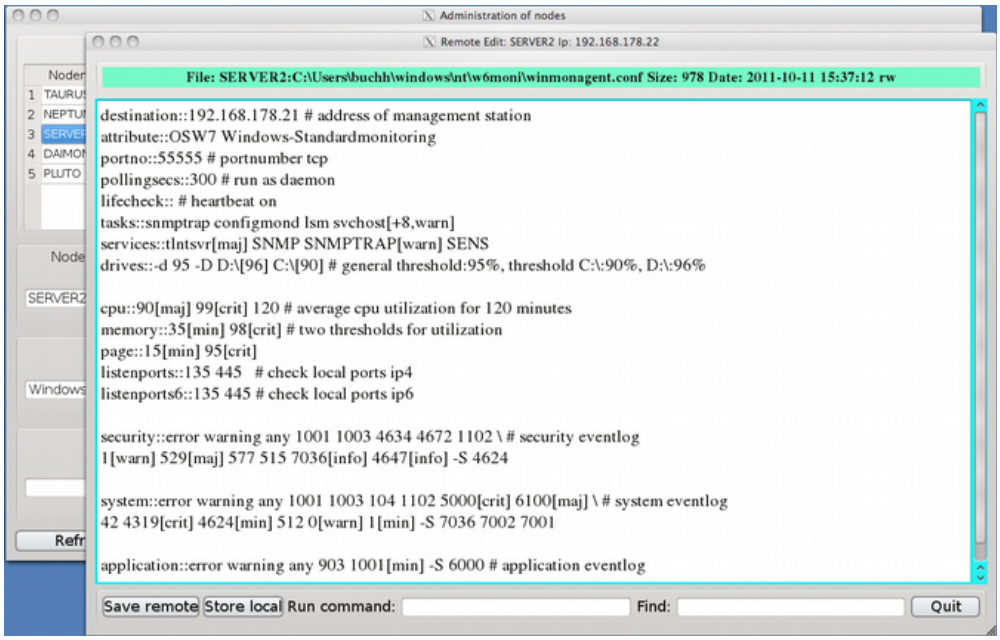
Die Abbildung zeigt eine Konfigurationsdatei zur Standardüberwachung im Remote Editor der Management-Station. Der Agent wird als Hintergrundprozess mit einem Polling Intervall von 150 Sekunden betrieben.

Es wird überwacht: Alle Filesysteme, eine Liste von vier Hintergrundprozessen, Swap/Memory, load average, durchschnittliche CPU-Belastung über 60 Minuten und alle Protokolldateien mit dem Muster "*.log" im Verzeichnis "/var/log" sowie die error_log von Apache.

Ferner wird der Hintergrundprozess "remoteconfd" überwacht, und sollte er nicht aktiv sein neu gestartet. Mit dem Befehl "netstat" werden die Udp-Ports 111, 123 und 161 überwacht. Die Auswertung des Kommandos geschieht mit Negativ-Filtern (unterer Bildrand).

Die Filesystemüberwachung bildet unabhängig von der Anzahl der angeschlossenen Dateisysteme eine Einheit. Das gleiche gilt auch für die Prozess- und Portüberwachung.

20.2 Beispiel Standardüberwachung Windows

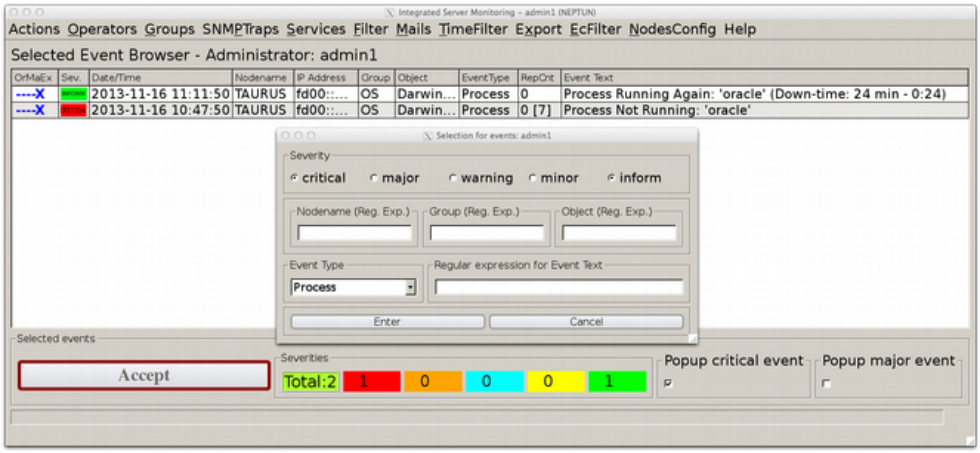


Die Abbildung zeigt die Bearbeitung einer Konfigurationsdatei mit dem Remote Editor an der Management-Station. Es ist die Konfigurationsdatei zur Standardüberwachung für Windows.

Es wird überwacht: Alle Laufwerke (drives), eine Liste von vier Tasks, fünf Dienste (Services), Memory, Pagefile, durchschnittliche CPU-Belastung für 120 Minuten, zwei Listenports ip4 und zwei Listenports ip6. Zusätzlich die System-Eventlog, die Security-Eventlog und Application-Eventlog. Als Suchargument für die Eventlogs dienen Event-Ids und/oder Event-Types. Die Option „-S“ gefolgt von einer Liste mit Event-Ids bewirkt den Ausschluss der entsprechenden Windows-Events. Die Schwellwerte für Pagefile und Memory sind Prozentwerte der Belegung.

Wie man sieht, sind die Konfigurationsdateien kompakt, gut zu lesen und mühelos zu administrieren, da sie auf Betriebssystemebene existieren. Durch den Austausch von Konfigurationsdateien lassen sich gute Synergieeffekte erzielen.

20.3 Beispiel Prozessüberwachung

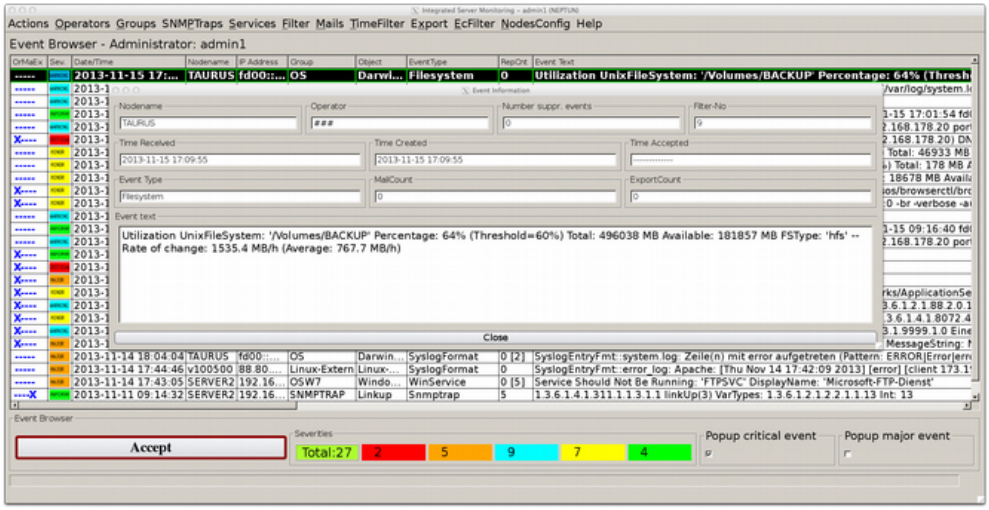


Die Abbildung zeigt Meldungen in dem Browser “Selected Event Browser“, mit dem man spezielle Meldungen im aktiven Browser auswählen kann. Die ausgewählten Events signalisieren den Anfang und das Ende einer Störung.

Zu Beginn der Störung kommt die kritische Meldung mit dem Text “Process Not Running: ’oracle’“. Der Zahlenwert in eckigen Klammern in der Spalte “RepCnt“ zeigt die Anzahl der von einem Filter wegen inhaltlicher Gleichheit unterdrückten Meldungen an. Wenn der Hintergrundprozess wieder läuft, kommt automatisch (also ohne Konfiguration) eine grüne (informative) Meldung mit der expliziten Angabe der Ausfallzeit im Meldungstext. Diese Information ist verwertbar für *service level agreements* und andere statistische Erhebungen.

Die Meldungen kommen von den Agenten und gehören zur Standardüberwachung (Prozessüberwachung). Die gleichen Funktionen gibt es in der Standardüberwachung für Windows bezogen auf Tasks und Services.

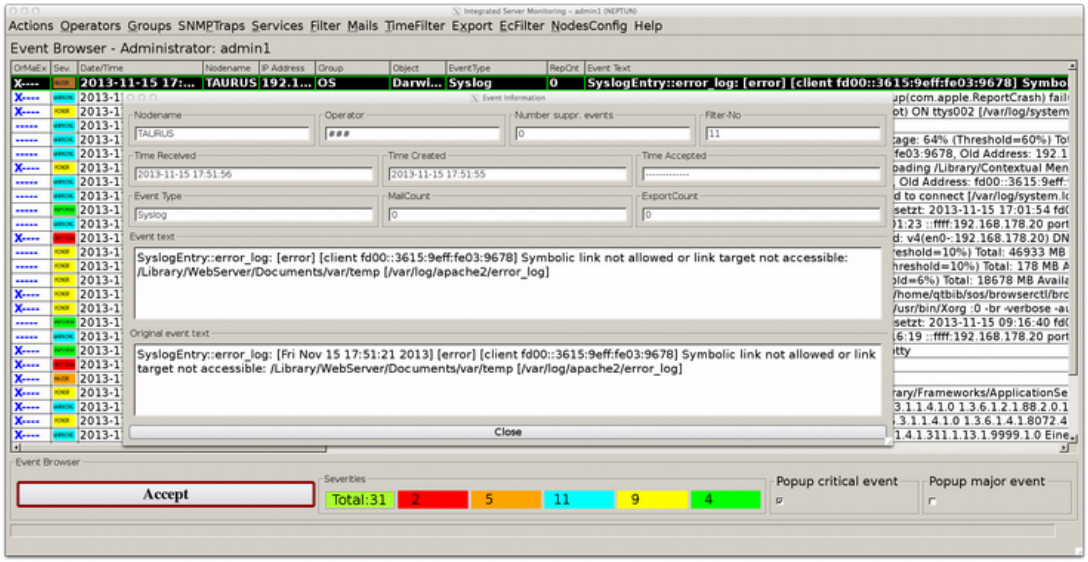
20.4 Beispiel Filesystemüberwachung



Die Abbildung zeigt die Standardmeldung aus der Filesystemüberwachung. Im Meldungstext sind die folgenden Informationen enthalten: Betroffener Mountpoint, prozentuale Belegung, überschrittener prozentualer Schwellwert, totale Belegung in MB, verfügbarer Speicher in MB, Filesystemtyp, Änderungsgeschwindigkeit des Verbrauchs in MB/h, Durchschnittsgeschwindigkeit in MB/h. Durch die Angabe der Änderungsgeschwindigkeit sind aufwendige Trendgrafiken überflüssig.

Es werden ohne zusätzlichen Aufwand alle vorhandenen Dateisysteme erfasst. Für jedes Filesystem gibt es bei Überschreitung eines Schwellwertes eine eigene Meldung.

20.5 Beispiel Logfileauswertung

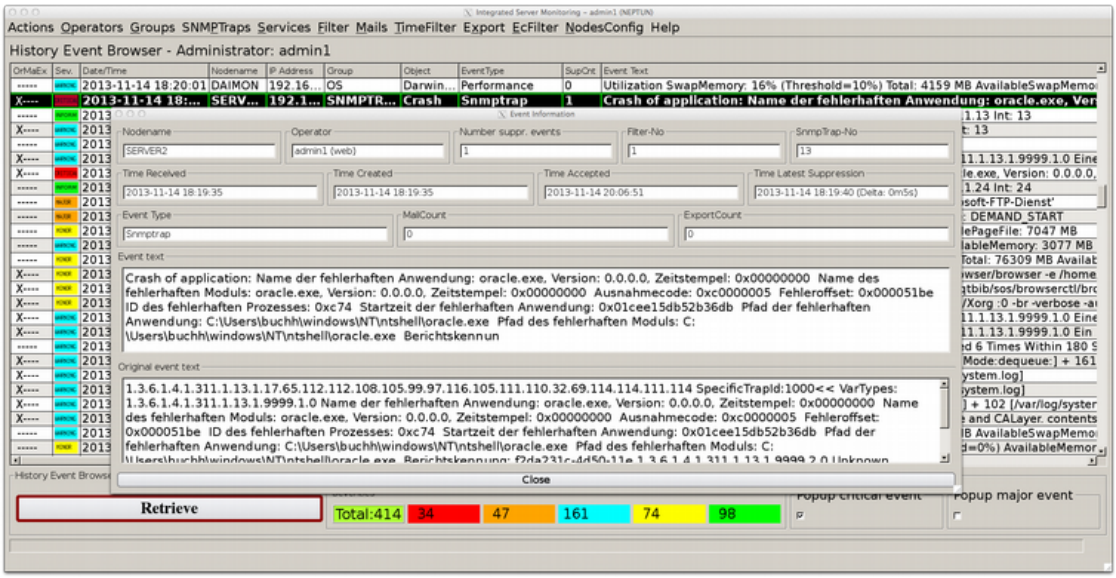


Die Abbildung zeigt eine Meldung von der Standardüberwachung, nämlich die Auswertung der error_log von Apache2. Das auslösende Suchmuster beim Agenten war „\[error\]“. Die gefundene Zeile ist an der Management-Station durch eine Kombination aus Substitution und Linksshift so verändert worden, dass die Datumsangabe fehlt. Man sieht im Bild den originalen und umgesetzten Meldungstext.

Diese Ersetzungsfunktion ist von großer Bedeutung, weil man ohne Datums- und Zeitangabe auf inhaltliche (wertmäßige) Gleichheit prüfen und zeitlich unterdrücken kann. Dadurch kommt es nicht zu „Meldungsfluten“, die bei herkömmlichen Systemen ein erhebliches Problem darstellen.

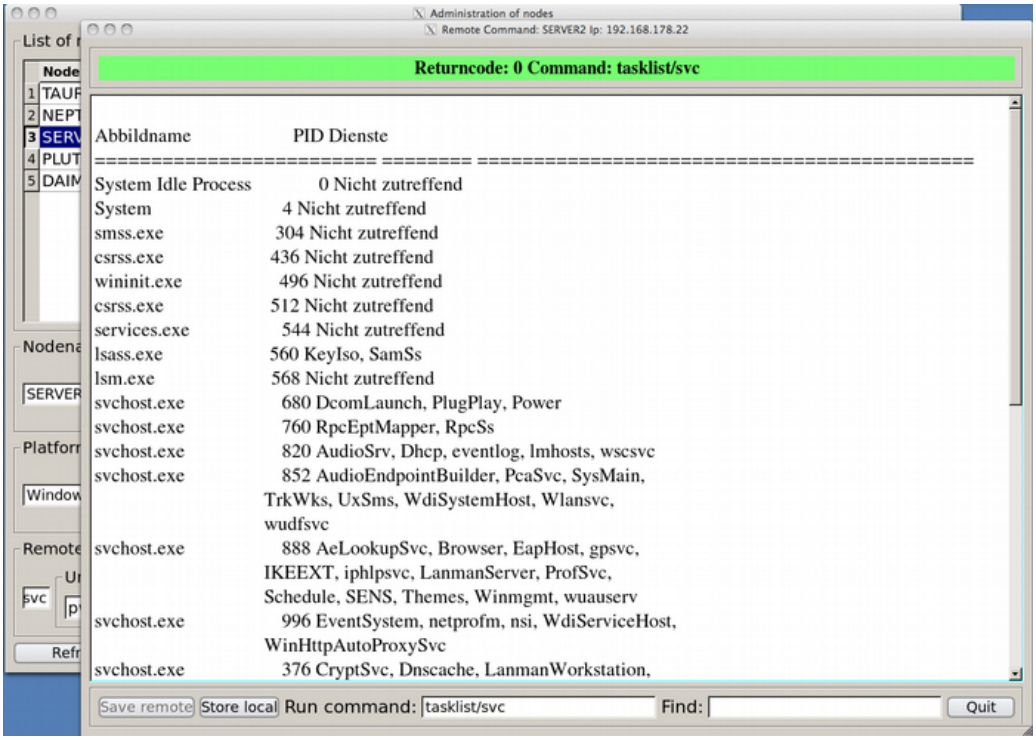
Es kommt vor, dass bei bestimmten Störungen (zum Beispiel Plattenfehler) tausende von Einträgen in relativ kurzer Zeit in die System-Protokolldatei geschrieben werden. Prinzipiell kann das bei anderen Logfiles ebenfalls geschehen. Daher braucht man eine wirksame Filterung sowohl bei den Agenten als auch an der Management-Station.

20.6 Beispiel SNMP-Traps



Die Abbildung zeigt eine von einem Programmabsturz verursachte Trap-Meldung, die durch einen Formatstring benutzergerecht aufbereitet wurde. Durch Doppelklick einer selektierten Meldung erhält man das Widget „Event Information“, in dem der veränderte sowie originale Meldungstext zu sehen ist.

20.7 Beispiel Kommando-Interface



Die Abbildung zeigt die Rückgabe eines Befehls an einen entfernten Windows-Server. Man kann so auch Dienste nachstarten, wenn der Hintergrundprozess remoteconfd.exe die entsprechenden Rechte hat. Die Kommunikation ist mit AES/CBC verschlüsselt. Das gleiche gilt für Unix-Server.