Deutsch

FUJITSU Software BS2000

SNMP Management (NET-SNMP) für BS2000

Benutzerhandbuch

Gültig für:

NET-SNMP V5.7
SNMP-AGENTS V1.0

Ausgabe Januar 2019

Kritik… Anregungen… Korrekturen…

Die Redaktion ist interessiert an Ihren Kommentaren zu diesem Handbuch. Ihre Rückmeldungen helfen uns, die Dokumentation zu optimieren und auf Ihre Wünsche und Bedürfnisse abzustimmen.

Sie können uns Ihre Kommentare per E-Mail an bs2000services@ts.fujitsu.com senden.

Nach DIN EN ISO 9001:2015 zertifizierte Dokumentationserstellung

Um eine gleichbleibend hohe Qualität und Anwenderfreundlichkeit zu gewährleisten, wurde diese Dokumentation nach den Vorgaben eines Qualitätsmanagementsystems erstellt, welches die Forderungen der DIN EN ISO 9001:2015 erfüllt.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright und Handelsmarken

Inhalt

1 Einleitung

Die Liefereinheit NET-SNMP V5.7 des BS2000-Betriebssystems und das Produkt SNMP-AGENTS V1.0 bieten die Basis-Funktionalität für BS2000-Systeme, um in SNMP-basierte Managementumgebungen eingebunden werden zu können. NET-SNMP V5.7 und SNMP-AGENTS V1.0 erlauben Netz-, System- und Anwendungsmanagement über SNMP.

Diese Komponenten werden ergänzt durch die produktspezifischen SNMP Agenten für BCAM, FTP (als mit TCP-IP-AP ausgelieferter Bestandteil von interNet Services) und SESAM/SQL.

1.1 Zielsetzung

Dieses Handbuch beschreibt die Einbettung von NET-SNMP und SNMP-AGENTS sowie der BCAM-, FTP- und SESAM/SQL-Agenten in BS2000-Systeme, die zum Betrieb notwendigen Installations- und Konfigurationsschritte sowie den Betrieb selbst. Die zur Überwachung notwendigen Agenten und ihre MIBs werden detailliert vorgestellt.

Zusätzlich werden ausführliche Hinweise zum sicheren Betrieb des SNMP-Managements gegeben.

1.2 Zielgruppe

Das vorliegende Handbuch wendet sich an Netz- und Systembetreuer, die BS2000-Systeme in ein SNMP-basiertes Netz-, System- und Anwendungsmanagement integrieren bzw. ein solches System bedienen wollen. Kenntnisse des Betriebssystems BS2000 sowie der TCP/IP-Grundbegriffe werden vorausgesetzt.

1.3 Wegweiser durch das Handbuch

Das vorliegende Handbuch ist wie folgt strukturiert:

- Kapitel 2: Überblick

  Dieses Kapitel führt in die SNMP-Architektur ein, stellt Grundlagen vor und beschreibt die Einbettung in BS2000.

- Kapitel 3: Installation und Konfiguration

  In diesem Kapitel werden die Installationsvoraussetzungen sowie die Installation selbst beschrieben. Die Konfigurationsschritte im BS2000-System werden ausführlich dargestellt.

- Kapitel 4: Betrieb

  Kapitel 4 beschreibt die POSIX-Skripts zum automatischen Starten und Beenden des SNMP-Dämon und der Agenten sowie die Möglichkeiten, Dämonen und Agenten manuell zu starten und zu beenden.

- Kapitel 5: Funktionen von NET-SNMP

  Dieses Kapitel beschreibt die Gruppen der MIB-II (RFC1213) sowie weiterer Gruppen, die durch den SNMP-Dämon gemanaged werden. Außerdem beschreibt es die Implementierung von DISMAN Monitoring (auch als Event Services bezeichnet) und DISMAN Scheduling, die als Teil des SNMP-Dämons in NET-SNMP enthalten sind.

- Kapitel 6: Funktionen von SNMP-AGENTS

  Dieses Kapitel beschreibt die Funktionen und den Betrieb der in SNMP-AGENTS enthaltenen Subagenten einschließlich einem Überblick über die zugehörigen MIBs.

- Kapitel 7: Funktionen von BCAM-, FTP- und SESAM/SQL-Subagent

  Dieses Kapitel beschreibt die Funktionen und den Betrieb der Agenten von BCAM und interNet Services.

  Hinweis: der SESAM/SQL-Agent wird in der SESAM/SQL-Dokumentation beschrieben.

- Kapitel 8: Beispiel für den Betrieb der Management-Stationen

- Anhang

  Im Anhang werden Hinweise für das Verhalten im Fehlerfall gegeben.

1.4 Typografische Gestaltungsmittel

In diesem Handbuch werden folgende Mittel zur Darstellung von funktional wichtigen Textteilen verwendet:

für Hinweistexte

ACHTUNG!
für Warnhinweise

kursive Schrift
: für Dateinamen, Namen von Auftragsfenstern, Parameterbezeichnungen, Menütitel und Menüeinträge sowie Kommandos und Variablen im Fließtext.

<spitze Klammern>
: kennzeichnen Variable, wenn Sie dafür Werte einsetzen müssen.

`dicktengleiche Schrift`
: für die Darstellung von Eingaben für das System, Systemausgaben und für Dateinamen in Beispielen.

kommando
: In der Syntaxbeschreibung für Kommandos werden diejenigen Bestandteile (Bezeichnungen von Kommandos und Parametern) fett dargestellt, die unverändert eingegeben werden müssen.

1.5 Änderungen gegenüber der Vorgängerversion

SNMP wurde mit mehreren Komponenten im BS2000 neu realisiert: NET-SNMP, SNMP-AGENTS sowie den produkt-spezifischen Agenten von BCAM, FTP und SESAM/SQL. Diese Komponenten ersetzen die Emanate-basierten Produkte SBA-BS2, SSC-BS2, SSA-OUTM-BS2 und SSA-SM2-BS2.

Im Folgenden wird die bisherige Realisierung von SNMP im BS2000 mit "SNMP V6.x" bezeichnet, die neue Lösung mit "NET-SNMP/SNMP-AGENTS".

Im Vergleich zum Benutzerhandbuch SNMP Management V6.0A ergeben sich folgende Änderungen:

Neue Produktstruktur und neue Basis-Software

SNMP-Komponenten:

- NET-SNMP V5.7 ist Bestandteil des Betriebssystems BS2000 OSD/BC.
  NET-SNMP V5.7 basiert auf der OpenSource Software Net-SNMP (BSD Lizenz).

  Es enthält die Teile der MIB-II einschließlich der Gruppen, die für openNet Server und interNet Services benötigt werden (IP, TCP, UDP, ...). NET-SNMP V5.7 enthält außerdem die Funktionalität von Event und Scheduler Services, wie sie in den Event und Scheduler Agenten der vorherigen SNMP-Versionen V6.x implementiert war.
- SNMP-AGENTS ist ein optionales Produkt, das die folgenden Subagenten enthält:
  - Application Monitor
  - Console Monitor
  - Host Resources
  - HSMS
  - openFT
  - openSM2
  - openUTM
  - Spool & Print Services
  - Storage
- Die produktspezifischen Agenten zu BCAM (private BCAM-MIB), FTP und SESAM/SQL wurde ebenfalls neu implementiert.

Entfallene Agenten

Folgende Agenten werden nicht mehr angeboten:

- Supervisor Agent
- AVAS
- HIPLEX-AF

– OMNIS
– HTML Agent

Sonstige Änderungen

- Das manuelle Starten der Agenten ist jetzt ausschließlich per POSIX-Kommando möglich. Es gibt weiterhin Autostart- und Autostop-Skripts, so dass es möglich ist, die Agenten beim Starten bzw. Beenden von POSIX automatisch zu starten/stoppen.
- in der Bezeichnung der MIBs ändert sich das Kürzel "sni" zu "fj".

1.6 Readme-Datei

Funktionelle Änderungen der aktuellen Produktversion und Nachträge zu diesem Handbuch entnehmen Sie bitte ggf. der produktspezifischen Readme-Datei.

Readme-Dateien stehen Ihnen online bei dem jeweiligen Produkt zusätzlich zu den Produkthandbüchern unter *http://manuals.ts.fujitsu.com* zur Verfügung. Alternativ finden Sie Readme-Dateien auch auf der Softbook-DVD.

Informationen unter BS2000

Wenn für eine Produktversion eine Readme-Datei existiert, finden Sie im BS2000-System die folgende Datei:

`SYSRME.<product>.<version>.<lang>`

Diese Datei enthält eine kurze Information zur Readme-Datei in deutscher oder englischer Sprache (<lang>=D/E). Die Information können Sie am Bildschirm mit dem Kommando `/SHOW-FILE` oder mit einem Editor ansehen.
Das Kommando `/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=<product>` zeigt, unter welcher Benutzerkennung die Dateien des Produkts abgelegt sind.

Ergänzende Produkt-Informationen

Aktuelle Informationen, Versions-, Hardware-Abhängigkeiten und Hinweise für Installation und Einsatz einer Produktversion enthält die zugehörige Freigabemitteilung. Solche Freigabemitteilungen finden Sie online unter *http://manuals.ts.fujitsu.com*.

2 Überblick

SNMP steht für **S**imple **N**etwork **M**anagement **P**rotocol und wurde als Protokoll für Netzmanagement-Dienste in TCP/IP-Netzen entwickelt. Die Überwachung und Administration von LAN-Komponenten, wie z.B. Bridges, Routers, Hubs usw. in heterogenen Netzen mit TCP/IP-Protokollen war ursprünglich die einzige Aufgabe von SNMP. Inzwischen hat sich der Anwendungsbereich von SNMP um System- und Anwendungsmanagement erweitert. Ähnlich wie bei TCP/IP, wo der Begriff nicht nur die Protokolle als solche, sondern das gesamte entsprechende Netzwerk bezeichnet, steht auch SNMP nicht nur für das Protokoll allein, sondern für das gesamte aufs SNMP basierte Management-System.

2.1 Grundlagen der SNMP-Management-Architektur

Zentraler Bestandteil einer SNMP-Installation ist die Management-Plattform. Die Management-Plattform ermöglicht eine übersichtliche Darstellung der verwalteten Komponenten und eine komfortable Bedienung. Von der Management-Plattform aus lässt sich das Netz mit all seinen Systemen und Anwendungen überwachen und steuern. SNMP ist nicht auf eine bestimmte Management-Plattform fixiert.

Auf der Management-Plattform residiert der SNMP-Manager, auch Management-Station genannt. Der SNMP-Manager ist eine Anwendung, die via SNMP über ein TCP/IP-Netz mit Partneranwendungen, den SNMP-Agenten, kommuniziert. Auf jeder verwalteten Komponente liegt ein Agent, der dem SNMP-Manager aktuelle Informationen über diese Komponente liefert. Die Initiative zur Steuerung der Aktivitäten liegt überwiegend auf Seiten des SNMP-Managers, wodurch die Belastung der verwalteten Komponenten mit Management-Aufgaben gering gehalten wird.

Grundlage für das Management der zu verwaltenden Komponenten ist die genaue Beschreibung der zu administrierenden Bestandteile (Objekte) dieser Komponenten in der MIB (Management Information Base). Die MIB ist das informationstechnische Rückgrat eines jeden Management Agents. Sie enthält Informationen zu Eigenschaften, wie z.B. Name, Syntax, Zugriffsrechte und Status jeder einzelnen Komponente. Für viele Hard- und Softwarekomponenten werden vom Hersteller eigene MIBs mitgeliefert. Die Codierung der MIB erfolgt in ASN.1 (Abstract Syntax Notation One). ASN.1 wurde auch von ISO als Standard für den Presentation Layer genormt (siehe ISO/IEC 8824 und 8825).

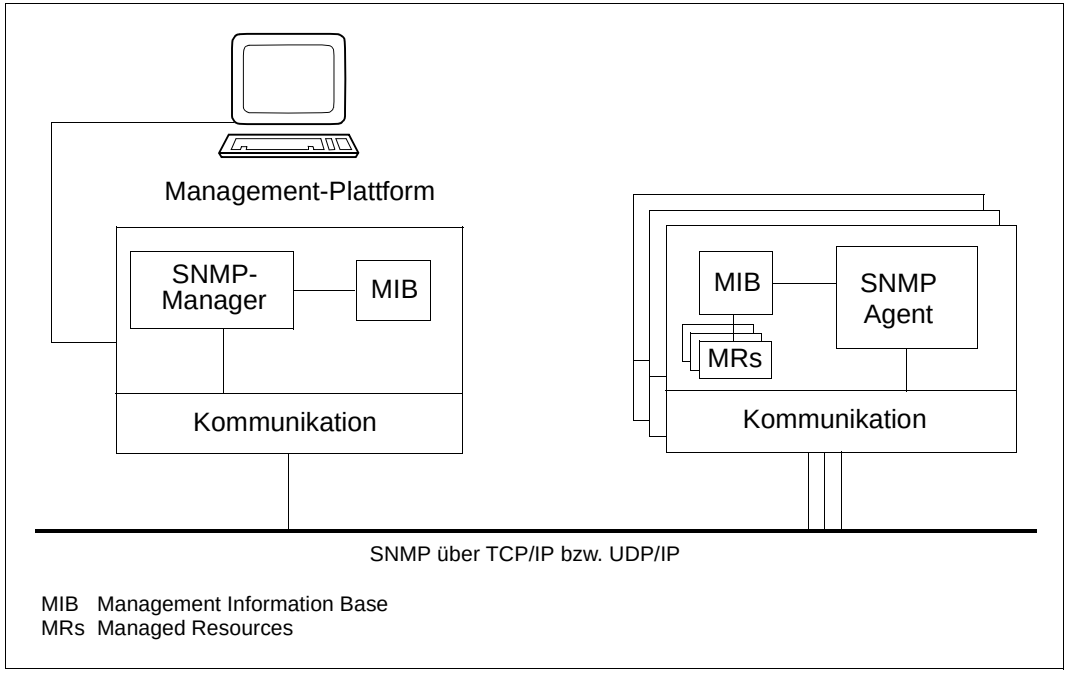


Bild 1: Kommunikation zwischen SNMP-Manager und Agenten

SNMP-Protokollelemente

Der Transport der Informationen über das Netz erfolgt mithilfe von funktionsabhängigen SNMP-Protokollelementen. SNMPv1 benötigt zum Abfragen, Setzen und Anzeigen von Werten, die relevante Management-Informationen (Objektwerte) enthalten, nur vier verschiedene Protokollelemente. Ein fünftes Protokollelement, Trap, dient dem Agenten zum asynchronen Melden wichtiger Ereignisse.

Protokollelement	Typ	Funktion
GetRequest-PDU	0	Leseanforderung des Managers für ein genau definiertes Objekt
GetNextRequest-PDU	1	Leseanforderung des Managers für das nächste (unbekannte) Objekt
GetResponse-PDU	2	Rückmeldung des Agenten mit den geforderten Werten
SetRequest-PDU	3	Schreibanforderung des Managers auf ein genau definiertes Objekt
Trap-PDU	4	Asynchrone Meldung des Agenten bei besonderen Ereignissen

SNMPv1-Protokollelemente

Die eigentliche SNMP-Nachricht ist recht einfach aufgebaut. Sie besteht aus dem SNMP-Header sowie der PDU (Protocol Data Unit). Der SNMP-Header enthält ein Versionskennzeichen und den Community-Namen.

Die PDU besteht aus dem Feld für den PDU-Typ sowie einer Liste von

– zu lesenden Variablen (bei GetRequest und GetNextRequest) oder
– zu setzenden Variablen (bei SetRequest).

Jede Variable besteht aus dem Namen eines überwachten Objekts und dem zugehörigen Wert. Die Liste der zu einer SNMP-Nachricht gehörenden Variablen wird Variable-Bindings genannt (kurz „varbinds“).

SNMPv3 bietet zusätzliche Sicherheitsfunktionen, die über Security Parameter realisiert sind. Details siehe Abschnitt „Zugangskontrolle“.

2.2 SNMP-Management in BS2000 - Einbettung und Funktionalität

Für den Anschluss des BS2000 an ein SNMP-Management werden Lösungen mit unterschiedlicher Zielsetzung angeboten.

- Mit den Produkten NET-SNMP V5.7 (Bestandteil von OSD/BC ab V11.0) und SNMP-AGENTS V1.0 lassen sich BS2000-Systeme direkt in SNMP-basierte Management-Plattformen wie z.B. Nagios oder Icinga integrieren. Im Kapitel „Beispiel für den Betrieb der Management-Station“ finden Sie dazu einige Beispiele. Sowohl NET-SNMP als auch SNMP-AGENTS ermöglichen Netz-, System- und Anwendungsmanagement über eine Implementierung des SNMP-Protokolls in BS2000.
- Die SNMP Agenten für BCAM, FTP (Teil des Produkts TCP-IP-AP) und SESAM/SQL ergänzen die Funktionalität von SNMP-AGENTS und stellen weitere Möglichkeiten zum BS2000-Management bereit. Die Konfiguration und Funktionalität dieser Agenten werden ebenfalls in diesem Handbuch beschrieben.

Bild 2 auf der nächsten Seite gibt einen Überblick über die SNMP-Integration von BS2000.

Fujitsu Technology Solutions bietet die Integration in verschiedene SNMP-Management-Systeme als Service an. Um weitere Informationen zu erhalten, kontaktieren Sie bitte Ihren zuständigen Vertriebsbeauftragten.

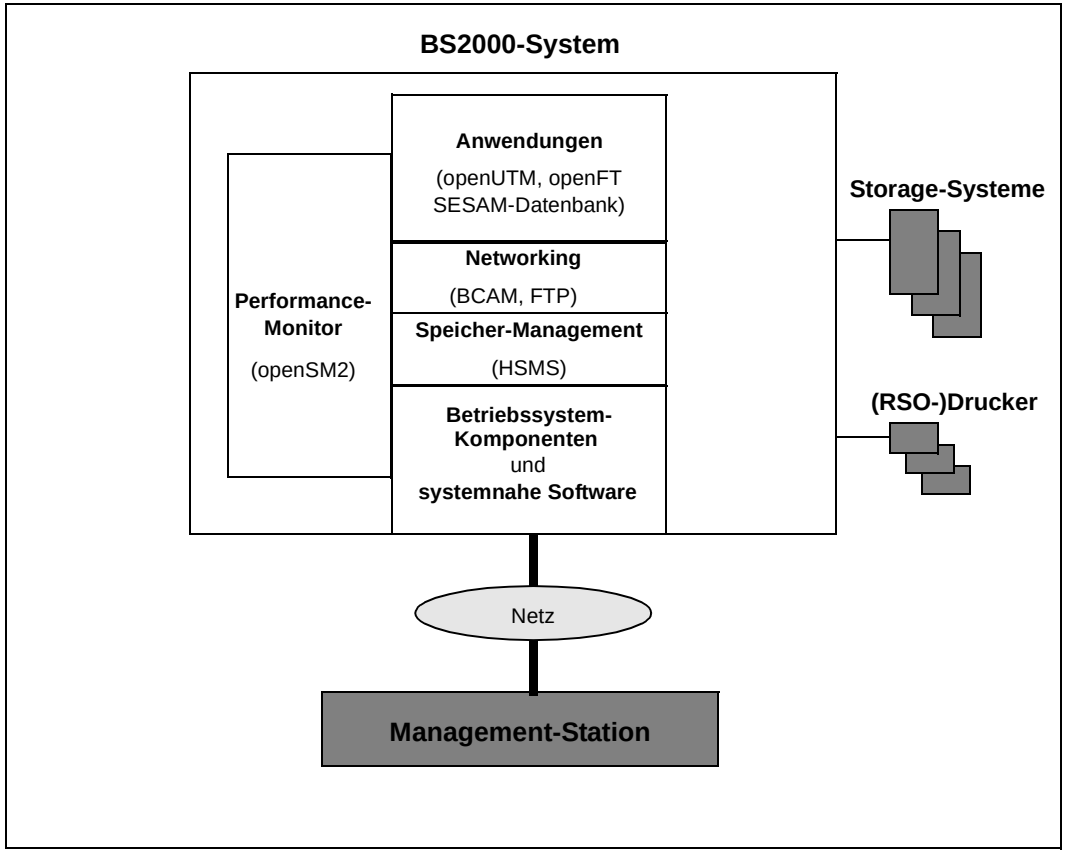


Bild 2: Überblick über die mit SNMP administrierbaren Systeme

2.2.1 Produktstruktur

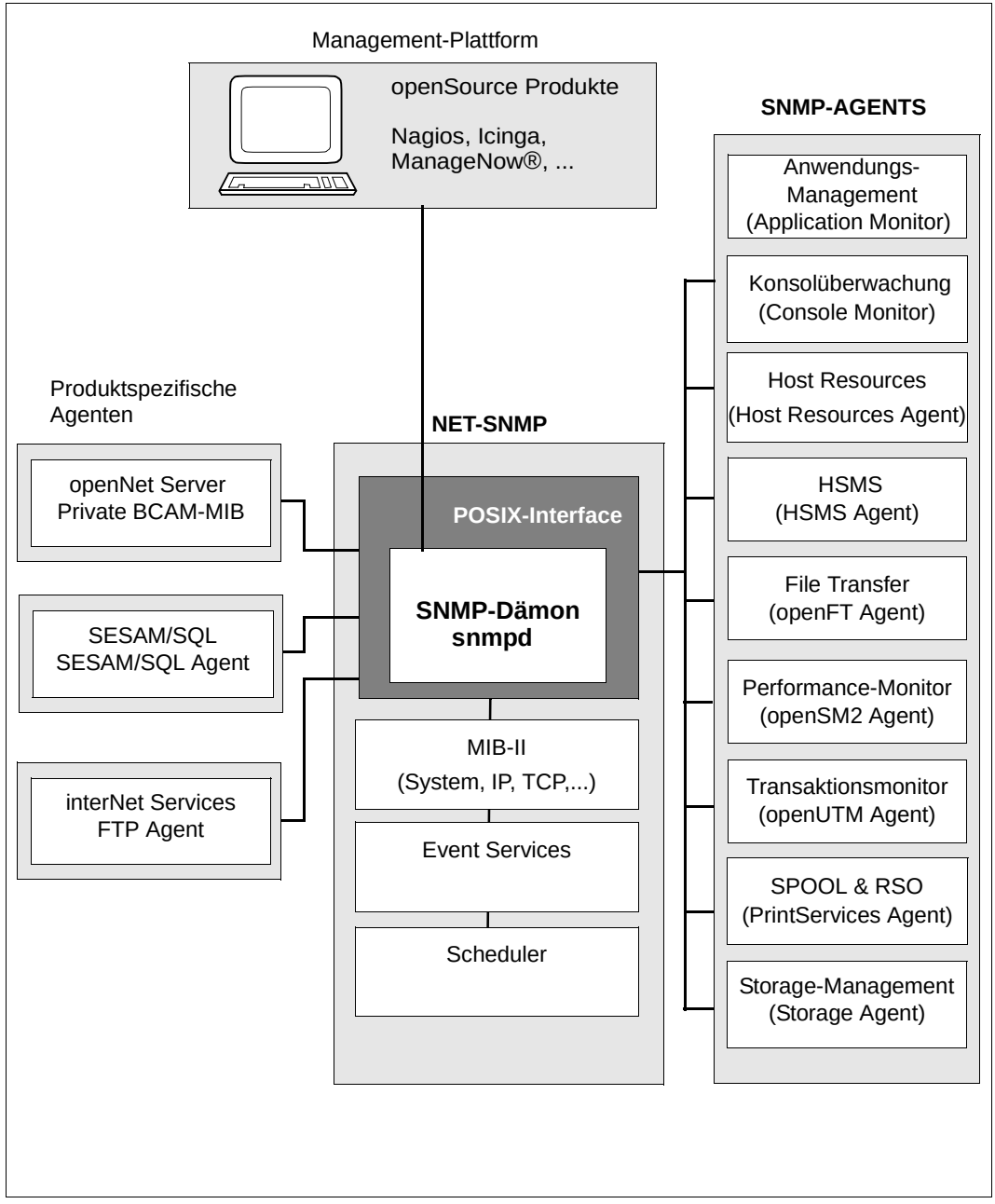


Bild 3: Aufbau des SNMP-Managements in BS2000

NET-SNMP

NET-SNMP V5.7 wird mit BS2000 OSD/BC ausgeliefert und enthält den SNMP-Dämon *snmpd*, den SNMP-Trap-Dämon snmptrapd und SNMP-Client-Tools (*snmpwalk*, *snmpget* und *snmpset*). Der SNMP-Dämon enthält außerdem die Funktionalität von Event und Scheduler Services, wie sie in die Event- und Scheduler-Agenten der vorherigen SNMP-Versionen V6.x implementiert war.

- Der SNMP-Dämon ist einerseits der BS2000-Kommunikationspartner der Management-Station, der das SNMP-Protokoll abwickelt. Andererseits steuert er die Kommunikation mit den Agenten und bietet zusätzlich Zugriffe auf verschiedene Gruppen der MIB-II (System, SNMP, Interface, ICMP, IP, TCP, UDP) sowie auf Objekte weiterer standardisierter SNMP-MIBs (RFC 2272 - 2275) und erlaubt so die Überwachung des Systems und der SNMP-relevanten Werte.
- Mit Event Services lassen sich MIB-Objekte anderer Agenten überwachen und einfache Aktionen ausführen, sobald bestimmte Bedingungen (Trigger Tests) erfüllt sind.
- Scheduling Services ermöglicht es, Änderungen an Objekten periodisch oder zu definierten Zeitpunkten auszuführen.

SNMP-AGENTS

Mit SNMP-AGENTS V1.0 wird ein Set von Agenten für BS2000-spezifische Management-Aufgaben ausgeliefert.

- Der Application Monitor Agent überwacht Benutzeranwendungen, BCAM-Anwendungen, Tasks, Jobvariablen, BS2000-Subsysteme und Logging-Dateien von BS2000, POSIX und NFS. DCAM-Anwendungen, Protokoll-Dateien und Subsystem-Zustände können zyklisch überwacht werden. Logisch zusammengehörige Objekte aus einem Business-Prozess können mit dem Application Monitor Agent als Gruppe zusammengefasst werden und sowohl gemeinsam als auch einzeln überwacht werden.
- Der Console Monitor Agent dient der Konsolüberwachung. Er bietet Ihnen einerseits die Möglichkeit, Konsolmeldungen als Traps weiterzuleiten; die Menge der zu erfassenden Meldungen kann dabei gezielt definiert werden. Andererseits können Sie von der Management-Station aus BS2000-SDF-Kommandos absetzen und das Resultat dieser Kommandos abfragen.
- Der Agent Host Resources liefert Informationen über das System, über Geräte und Datei-Systeme sowie über die installierte Software und meldet Zustandswechsel.
- Der HSMS Agent ermöglicht das Lesen und Ändern von globalen HSMS-Daten. Ferner liefert er detaillierte Informationen über HSMS-Aufträge. Der Umfang der Aufträge kann durch die Auswahlkriterien „Zustand“ und „Herkunftsort“ eingeschränkt werden.

- Der openFT (BS2000) Agent liefert Informationen über FT-Systemparameter und Statistikdaten des laufenden Betriebs. Weitere Funktionen sind das Starten und Stoppen des FT, die Steuerung der Diagnose, das Erzeugen neuer Public-Keys zur Verschlüsselung und das Ändern des Status eines FT-Partners.
- Der openSM2 Agent zur Performance-Überwachung liefert vom SM2-Subsystem zur Verfügung gestellte Realzeit-Werte zur Überwachung des CPU-Verbrauchs, der I/O-Raten und der Speicherbelegung.
- Der openUTM Agent ermöglicht die Überwachung und Steuerung ausgewählter UTM-Anwendungen, z.B. Informationen über UTM-Objekte, Änderung von Anwendungseigenschaften und Systemparametern, Beenden einer UTM-Anwendung.
- Der Agent für Spool & Print Service überwacht die Geräte für SPOOL und RSO und liefert Informationen zu Druckaufträgen.
- Der Agent für das Storage-Management liefert Informationen über Pubsets und Platten. Außerdem kann der Agent den Zustand ausgewählter Pubsets und Platten überwachen.

BCAM-, FTP- (TCP-IP-AP) und SESAM/SQL-Agenten

Diese Produkte stellen produkt-spezifische Agenten zur Verfügung, die die Funktionalität von SNMP-AGENTS ergänzen:

- Der BCAM-Agent liefert Informationen über NEA- ISO- und TCP-IP-Protokolle und stellt sie auf dem Transportsystem zur Verfügung.
- Der FTP-Agent liefert Informationen über FTP-basierte Datenübertragungen. Er ermöglicht es außerdem, die lokalen FTP-Dämonen über die Änderung von Zustand und Parametern zu steuern.
- Der SESAM/SQL-Agent liefert Statístikinformationen in Hinblick auf die verfügbaren Data Base-Handler. Er liefert außerdem die Messwert-Ausgaben des SESAM/SQL Perfomancemonitors.

2.2.2 Aufbau des SNMP-Kollektion im BS2000

Der Anschluss des BS2000 an SNMP erfolgt über einen mit TCP/IP-Protokollen betriebenen LAN-Anschluss. Im BS2000 wird eine Agent-Applikation installiert, die die SNMP-Protokollelemente bedienen kann. Die Funktionalität des SNMP-Agenten ist aufgeteilt in den SNMP-Dämon und mehrere Agenten. Die Vorteile dieser Lösung liegen u.a. im Bereich der Ausfallsicherheit und Benutzerfreundlichkeit hinsichtlich Wartungs- und Änderungsaufwand.

Die Basis für diese Lösung ist die Portierung der Open Source Software Net-SNMP (unter BSD-Lizenz freigegeben) in das BS2000. Als Open Source Software steht Net-SNMP auf vielen Plattformen anderer Hersteller zur Verfügung.

2.2.2.1 SNMP-Dämon (snmpd)

Die heutigen Anforderungen an den SNMP-Agenten in einem Endsystem gehen über das normale Netzmanagement hinaus, sie reichen über System- und Anwendungsmanagement bis zum Management von Middleware (Transaktionssysteme und Datenbanken). Gerade bei größeren Endsystemen kommt auf Grund der vielfältigen Anforderungen der Wunsch auf, mehrere aufgabenspezifische Agenten einsetzen zu können, was durch die Strukturierung in Master- und Agenten unterstützt wird.

Der SNMP-Dämon ist den Agenten übergeordnet. Er stellt die Basisfunktionalität zur Verfügung wie z.B. Abwicklung des SNMP-Protokolls, Benutzerzugriffe, Sicherheits-Management sowie die Verbindung zu anderen Agenten. Bitte beachten Sie, dass der SNMP-Dämon auch als stand-alone Agent ohne Verbindung zu anderen Agenten ablauffähig ist. Daher ist der SNMP-Dämon auch zuständig für das Ausgeben und Setzen der Werte der verschiedenen Objekte in den MIB-II-Gruppen sowie weiterer Objekte in standardisierten MIBs (RFC 2272 - 2275).

Die Möglichkeit, Agenten einzeln starten und beenden zu können, erleichtert Änderung und Einsatz einzelner Agenten, ohne das gesamte Management-System beenden zu müssen, und erlaubt ein unterbrechungsfreies Management des restlichen Systems bei Ausfall einer Komponente sowie die parallele Bearbeitung von Aufträgen verschiedener Agenten.

Die Management-Station kommuniziert nur mit dem SNMP-Dämon. Die Kommunikation zwischen SNMP-Dämon und Agent erfolgt über eine asynchrone Nachrichtenschnittstelle. Die asynchrone Nachrichtenschnittstelle garantiert ein performantes Verhalten des Masteragenten bei der Auftragsbearbeitung, da er bei der Bearbeitung längerer Aufträge nicht blockiert wird, sondern parallel weitere SNMP-Requests bearbeiten kann.

2.2.2.2 Agenten

Die Agenten sind nur bei gestartetem SNMP-Dämon funktionsfähig. In der Initialisierungsphase meldet sich der Agent beim SNMP-Dämon über AgentX-Socket an und übergibt dem SNMP-Dämon seine MIB.

Agenten arbeiten ereignisorientiert. Nach der Initialisierung läuft der Agent in einer Warteschleife. Er verlässt die Schleife bei Eintreffen eines Ereignisses, das er bearbeiten muss. Als Ereignis werden u.a. Anforderungen des SNMP-Dämon, Timer-Abläufe oder das Eintreffen eines vereinbarten Signals verstanden. Hat der Agent alle vorliegenden Ereignisse abgearbeitet, kehrt er in seine Warteschleife zurück.

2.2.3 Bedienoberflächen für das SNMP-Management des BS2000

Über das Standard-Protokoll SNMP können BS2000-Systeme grundsätzlich an jede Management-Plattform angeschlossen werden, die SNMP beherrscht. Dies ist für alle markt-relevanten Management-Plattformen der Fall. Die Management-Plattformen der verschiedenen Hersteller bringen dabei ein unterschiedliches Leistungsspektrum ein.

Fujitsu Technology Solutions bietet die Integration in verschiedene SNMP-Management-Systeme als Service an, siehe Kapitel „Beispiel für den Betrieb der Management-Station“.

2.2.4 Paralleler Betrieb von SNMP V6.x und NET-SNMP

SNMP V6.x und NET-SNMP können parallel auf einem System vorhanden sein. D.h. eine bisherige Installation von SNMP V6.x kann weiterhin genutzt werden, wobei zu beachten ist:

Zu einer Zeit kann nur eine Instanz des FTP-Agenten ablaufen (entweder der alte oder der neue), da der alte FTP-Agent und der neue FTP-Agent mit dem FTP-Server über dieselbe Portnummer kommunizieren.

Standardmäßig meldet sich SNMP an den UDP-Port 161 und wartet dann dort auf Anforderungen. Falls ein paralleler Betrieb von SNMP V6.x und NET-SNMP zu erwarten ist, dann sollte einer der beiden Versionen mit einer anderen Portnummer konfiguriert werden.

Die NET-SNMP-Tools wie z.B. *snmpget*, *snmpwalk* und *snmpset* können auch für die Interaktion mit dem SNMP V6.x-Dämon *snmpdm* eingesetzt werden, sofern die SNMP-Aufrufe die Option -v1 verwenden.

Ein Mischbetrieb ist nicht möglich, d.h. die Agenten von SNMP-AGENTS können nicht zusammen mit SNMP V6.x und die Agenten von SNMP V6.x können nicht zusammen mit NET-SNMP betrieben werden.

2.3 Sicherheitsbewusste Nutzung von SNMP

Dieses Kapitel gibt Hinweise und Empfehlungen zur sicheren Nutzung des SNMP-basierten BS2000-Managements. Es ist jedoch keine Anleitung zur Sicherheitsanalyse oder zur Aufstellung eines Sicherheitsregelwerks. Diese beiden wichtigen Themen gehen über den Rahmen des vorliegenden Handbuchs hinaus.

Die funktionellen Details zur sicherheitsbewussten Einstellung der Konfigurationsparameter des SNMP-Agenten in BS2000 finden Sie im Abschnitt „Erweiterte Sicherheitsfunktionen für den Empfang von SNMP-Anforderungen“.

2.3.1 Empfehlungen zur allgemeinen Netz- und Systemsicherheit

Wenn Sie mit dem SNMP-Protokoll über das Internet kommunizieren, dann setzen Sie die beteiligten Systeme den möglichen Angriffen und Gefahren des öffentlichen Internet aus.

Empfehlung

Stellen Sie die zu verwaltenden BS2000-Systeme und die Management-Plattform in ein ausschließlich von Ihnen selbst kontrolliertes Teilnetz, zum Beispiel hinter eine Firewall. Auf diese Weise können Sie den Zugang zu Ihren Systemen einfacher und zentraler kontrollieren und mögliche Angriffe abwehren.

Die im SNMP-Agenten implementierten Sicherheitsfunktionen beruhen darauf, dass seine Konfigurations- und Programmdateien nur für privilegierte Nutzer - meist ist dies ein Administrator - zugreifbar sind. Die Installation richtet diese Dateien mit den korrekten Rechten ein.

Empfehlung

Vergewissern Sie sich in regelmäßigen Abständen, dass die Konfigurations- und Programmdateien des SNMP-Agenten im BS2000-System nur für privilegierte Nutzer zugreifbar sind.

2.3.2 Empfehlungen für die sichere Nutzung des SNMP-Service

SNMP ist bei naiver Nutzung unsicher. Die Standardkonfigurationsdatei, die nach der Installation wirksam wird, kommt mit einer Minimalmenge an Regeln aus und stellt ein Beispiel für eine mögliche Konfiguration dar. Dies ist ein Kompromiss zwischen Sicherheit und umfassender Interoperabilität im SNMP-Verbund, der zu Gunsten der Interoperabilität ausfällt.

Vermeiden Sie daher den operativen Betrieb mit der Minimalkonfiguration des BS2000-Systems. Wenn Sie die Konfiguration am verwalteten BS2000-System ändern, passen Sie bitte auch die korrespondierenden Einstellungen an der Management-Plattform an.

Empfehlung

Ändern Sie die Minimalkonfiguration des SNMP-Agenten sowie die korrespondierende Einstellungen an der Management-Plattform entsprechend den Vorgaben Ihres Sicherheitsregelwerks.

Bei folgenden Konfigurationsparametern sollten Sie unter Sicherheitsaspekten besonders aufmerksam verfahren:

- Community Strings für den Empfang von SNMP-Anforderungen (SNMP Protokollversion 1 und 2c)
- Community Strings und Zugriffskontrolle auf MIB-Objekte
- Erweiterte Sicherheitsfunktionen für den Empfang von SNMP-Anforderungen (SNMPv3)
- Community Strings und Absenderadressen
- Empfängeradressen für SNMP-Traps

2.3.2.1 Community Strings für den Empfang von SNMP-Requests

Unter einer Community versteht man in SNMP der Protokollversion 1 und 2c eine Gruppe, bestehend aus einer oder mehreren Management-Plattformen und mehreren von diesen Plattformen betreuten SNMP-Agenten.

Jede Community wird durch einen sog. Community String identifiziert. Der Community String ist unverschlüsselter Bestandteil jedes SNMP-Requests und weist den Absender des Requests als Mitglied der betreffenden Community aus. Die Berechtigung für einen lesenden oder schreibenden Request, den eine Management-Plattform an einen SNMP-Agenten sendet, wird über diesen Community String geregelt.

Mit dem Community String steht ein einfacher Authentisierungsmechanismus in SNMP zur Verfügung. So dürfen Management-Plattform und SNMP-Agent nur dann miteinander kommunizieren, wenn Sie derselben Community angehören, d.h. der SNMP-Agent akzeptiert SNMP-Requests nur von Management-Plattformen, deren Community Strings ihm bekannt, also vorkonfiguriert, sind.

Da der Community String unverschlüsselt mit der SNMP-Nachricht versendet wird, besteht immer die Gefahr seiner unberechtigten Verwendung. Dies kann für eine sicherheitsbewusste Nutzung von SNMP problematisch sein. Andererseits verwenden die meisten Communities ohnehin den voreingestellten Community String "public".

Empfehlung

Wählen Sie geeignete Communities entsprechend Ihrer System- und Betriebsorganisation und ordnen Sie passende Community Strings zu. Ändern Sie den Community String entsprechend den Vorgaben Ihres Regelwerkes, ähnlich wie Sie es z.B. von Passwörtern kennen. Beachten Sie, dass Sie den Community String in allen beteiligten Systemen der Community ändern müssen.

Empfehlung

Wenn es Ihr Umfeld aus SNMP-Agenten und Management-Plattform(en) erlaubt, sollten Sie die benutzerspezifische Authentisierung des SNMPv3-Protokolls verwenden.

2.3.2.2 Erweiterte Sicherheitsfunktionen für den Empfang von SNMP-Anforderungen

SNMPv3 verwendet das erweiterte Sicherheitsmodell USM (User-based security model), welches eine Liste von Benutzern und zugehörigen Attributen enthält. USM ist in RCF2574 beschrieben

Jeder Benutzer besitzt einen Namen (*securityName*), einen Authentifizierungstyp (*authProtocol*) und ein privates Protokoll (*privProtocol*) sowie zugehörige Schlüssel (*authKey* und *privKey*).

Die Authentifizierung durchgeführt, indem die zu sendende Nachricht mit dem *authKey* des Benutzers gekennzeichnet wird. Das *authProtocol* kann dabei entweder MD5 oder SHA sein. *authKeys* (und *privKeys*) werden aus einer Passwort-Phrase generiert, die mindestens 8 Zeichen lang sein muss.

Um die Nutzdaten der zu sendenden Nachricht zu verschlüsseln, wird der *privKey* des Benutzers verwendet. Das *privProtocol* kann entweder AES oder DES sein.

Durch Setzen des *securityLevel* kann festgelegt werden, ob Nachrichten nicht-authentifiziert, authentifiziert oder authentifiziert und verschlüsselt gesendet werden.

2.3.2.3 Community Strings und Kontrolle des Zugriffs auf MIB-Objekte

Community Strings können mit Zugriffrechten „read-only“, „read-write“ usw. versehen werden. Entsprechend den Zugriffrechten dürfen SNMP-Requests, die diesen Community String enthalten, die „read-only“ definierten Objekte lesen und die „read-write“ definierten Objekte lesen und ändern. Ohne weitere Vorkehrungen können dann *alle* zugreifbaren Objekte gelesen oder *alle* änderbaren Objekte gelesen und/oder geändert werden.

Empfehlung

Nutzen Sie die Möglichkeit, an bestimmte Community-Strings selektive Lese- bzw. Schreibrechte zu vergeben für

- MIB-Zweige,
- Objekte,
- Objektinstanzen (in Tabellen).

Empfehlung

Wenn es Ihr Umfeld aus SNMP-Agenten und Management-Plattform(en) erlaubt, sollten Sie die benutzerspezifische Autorisierung des SNMPv3-Protokolls verwenden.

2.3.2.4 Community Strings und Absenderadressen

Die IP-Adressen der autorisierten Management-Plattformen können Sie explizit vorkonfigurieren. Damit veranlassen Sie den SNMP-Agenten, SNMP-Anforderungen nur von diesen Systemen zu akzeptieren.

Die Management-Plattformen müssen dabei konstante IP-Adressen besitzen. Eine dynamische Zuweisung via Dynamic Host Configuration Protocol (DHCP) ist nicht möglich.

Empfehlung

Nutzen Sie die Prüfung der Absenderadressen der Management-Plattformen, indem Sie deren IP-Adressen im SNMP-Agenten konfigurieren.

2.3.2.5 Empfängeradressen für SNMP-Traps

Die IP-Adressen der autorisierten Management-Plattformen, an die ein SNMP-Agent SNMP-Traps senden soll, können Sie explizit vordefinieren. Die Management-Plattformen müssen dabei konstante IP-Adressen besitzen. Eine dynamische Zuweisung via Dynamic Host Configuration Protocol (DHCP) ist nicht möglich.

Empfehlung

Definieren Sie die Empfängeradressen der Management-Plattformen vor, die SNMP-Traps empfangen sollen. Konfigurieren Sie hierfür die IP-Adressen dieser Management-Plattformen im SNMP-Agenten.

2.3.2.6 Community String für SNMP-Traps

Sie können den Community String konfigurieren, den ein SNMP-Agent als Teil eines SNMP-Traps an die Management-Plattform sendet. Die Management-Plattform wird dann nur SNMP-Traps mit diesem Community String akzeptieren.

Empfehlung

Wählen Sie geeignete Communities entsprechend Ihrer System- und Betriebsorganisation. Ändern Sie den Community String entsprechend den Vorgaben Ihres Regelwerkes, ähnlich wie Sie es z.B. bei Passwörtern kennen. Beachten Sie, dass Sie den Community String in allen beteiligten Systemen der Community ändern müssen.

3 Installation und Konfiguration

Das BS2000-SNMP-Management besteht aus folgenden Produkten für den Einsatz auf dem BS2000-System:

- NET-SNMP V5.7 (Liefereinheit von B2000 OSD/BC ab V11.0)
- SNMP-AGENTS V1.0
- BCAM V24.0
- TCP-IP-AP V5.3 (FTP Agent)
- SESAM/SQL V9.1

3.1 Software-Voraussetzungen

Allgemeine Software-Voraussetzungen für NET-SNMP, SNMP-AGENTS, BCAM, TCP-IP-AP, SESAM/SQL

- BS2000 OSD/BC ab V11.0 mit entsprechender Software-Konfiguration
- POSIX ab A45

Software-Voraussetzungen für NET-SNMP V5.7

- openNet Server ab V4.0

Software-Voraussetzungen für SNMP-AGENTS V1.0

- NET-SNMP ab V5.7
- SDF-P-BASYS ab V2.5

Falls einzelne Agenten eingesetzt werden, dann wird Folgendes vorausgesetzt:

- JV ab V15.1 (Application Monitor Agent)
- openFT (BS2000) ab V12.1 (openFT Agent)
- SPOOL ab V4.9 (Spool Agent)

- HSMS ab V11.0 (HSMS Agent)
- SM2 ab V20.0 (openSM2 Agent
- openUTM ab V6.4 (openUTM Agent)

Weitere Details finden Sie in den Freigabemitteilungen zu B2000 OSD/BC V11.0 und der zugehörigen Produkte.

3.2 Installation in BS2000 OSD/BC

NET-SNMP ist Bestandteil des Betriebssystems BS2000 OSD/BC. Die notwendigen Dateien sind nach der Installation von BS2000 OSD/BC auf dem BS2000-System vorhanden. Für die Ablauffähigkeit von NET-SNMP ist aber noch eine Paket-Installation in POSIX erforderlich. Diese kann entweder manuell oder durch eine automatische POSIX-Paketinstallation mit IMON durchgeführt werden.

Die notwendigen Schritte, um die Installation manuell abzuschließen, finden Sie in Produkte manuell installieren.

Als optionale Produkte werden SNMP-AGENTS, (Bestandteil von interNet Services), BCAM und SESAM/SQL nicht standardmäßig auf einem BS2000-System installiert.

Die Installation erfolgt mit dem Installationsmonitor IMON. Soweit erforderlich, führt die IMON-Installation BS2000-spezifische Arbeiten durch wie das Erstellen von Subsystem-Katalog-Einträgen, POSIX-Installation etc.

Details finden Sie im aktuellen IMON-Handbuch.

Es ist darauf zu achten, dass im Subsystemkatalog ein Eintrag für das Subsystem SNMPAGT erstellt wird.

Das Löschen der SINLIB nach der Installation führt zu Fehlern, da die Agenten die SINLIB auch für den Betrieb benötigen.

Wenn Sie die Installation für eines der SNMP-Produkte manuell abschließen möchten, lesen Sie bitte den Abschnitt „Produkte manuell installieren“.

Für die Installation der Produkte sollten Sie IMON verwenden. Es wird nicht empfohlen, eines der Produkte manuell zu installieren.

3.2.1 Standardeinstellungen nach Installation

Alle SNMP-Produkte ermöglichen eine Logging über *syslog*. Die Logging-Optionen können global geändert werden (durch Editieren von *snmp.conf*) oder individuell für jeden Dämonen/Agenten (durch Editieren der entsprechenden rc-Datei mit den Auto-Start-/Stop-Skripts, welche Kommando-Optionen übergeben):

– Die rc-Dateien in *etc/rc2.d* werden beim Starten des POSIX-Subsystems aufgerufen und haben als Standard-Namenskonvention ein „S“ als ersten Buchstabe.

– Die rc-Dateien in *etc/rc0.d* werden beim Beenden des POSIX-Subsystems aufgerufen und haben als Standard-Namenskonvention ein „K“ als ersten Buchstaben.

Im Verzeichnis */usr/share/snmp/mibs* werden die MIB Dateien der installierten Produkte abgelegt.

Im Verzeichnis */etc/snmp* werden die Konfigurationsdateien und die gesicherten Start rc-Dateien abgelegt.

Folgende Tabelle fasst den Lieferumfang der SNMP-Produkte zusammen:.

Produkt	**Pfad der ausführbaren Programme**	**Programm-Namen**	**Konfiguations-dateien**	**rc-Skripte**
NET-SNMP	/opt/net-snmp	snmpd snmptrapd snmpwalk snmpget snmpset	snmpd.conf snmp.conf snmptrapd.conf	S90net-snmp K11net-snmp
SNMP-AGENTS	/opt/snmp-agents	appMonAgent consoleAgent openFTAgent openSM2Agent spoolAgent storageAgent hostAgent hsmsAgent utmAgent	(keine)	S91snmp-agents K11snmp-agents
TCP-IP-AP		ftpAgent	(keine)	S91snmpftp K11snmpftp
BCAM		bcamAgent	(keine)	S91snmpbcam K11snmpbcam
SESAM/SQL		sesAgent	(keine)	S91snmpsesam K11snmpsesam

3.2.2 Lieferumfang von NET-SNMP

Mit NET-SNMP werden folgende Binärdateien ausgeliefert:

– `snmpd` (SNMP-Dämon)

– `snmptrapd` (SNMP-Trap-Dämon)

– SNMP-Tools `snmpwalk`, `snmpget`, `snmpset`

Zusätzlich werden für die Dämonen und Tools folgende Beispiel-Konfigurationsdateien ausgeliefert:

– `snmpd.conf`: Konfigurationsdatei für den SNMP-Dämon `snmpd`

– `snmptrapd.conf`: Konfigurationsdatei für den Trap-Dämon `snmptrapd`

– `snmp.conf`: allgemeine Konfigurationsdatei für SNMP-Dämonen und -Tools

Die rc-Datei */etc/rc2.d/S90net-snmp* kann auch dazu verwendet werden, den beim POSIX-Start gestarteten SNMP-Dämon neu zu starten (z.B. zum Rekonfigurieren).

Dazu geben Sie unter einer privilegierten Benutzerkennung folgendes Kommando ein:

```
/etc/rc2.d/S90net-snmp restart
```

3.2.3 Produkte manuell installieren

Die Installation erfolgt mit dem POSIX-Installationsprogramm, das mit dem Kommando START-POSIX-INSTALLATION gestartet wird. Dazu muss das Subsystem POSIX gestartet sein. Details finden Sie im Handbuch „POSIX (BS2000) - Grundlagen für Anwender und Systemverwalter“.

Bei der Installation werden zusätzlich Informationsmeldungen ausgegeben.

Beispiel-Ausgabe für NET-SNMP

```
New configuration file was created </etc/snmp/snmp.conf.new>.
Please, review it and apply to snmp.conf if needed.
New configuration file was created </etc/snmp/snmpd.conf.new>.
Please, review it and apply to snmpd.conf if needed.
New configuration file was created </etc/snmp/snmptrapd.conf.new>.
Please, review it and apply to snmptrapd.conf if needed.

NET-SNMP v057 is INSTALLED:
Binaries are located in /opt/net-snmp with symlinks to the /usr/bin
Daemons binary links are in /usr/sbin
PATH for the config files: /etc/snmp
PATH for the MIB files: /usr/share/snmp/mibs
```

3.2.4 Deinstallation

De Deinstallation von SNMP-Produkten erfolgt ebenfalls über IMON oder manuell mit dem POSIX-Installationsprogramm (START-POSIX-INSTALLATION).

Bei der Deinstallation wird Folgendes gelöscht:

- Symbolische Links in */usr/sbin* und/oder */usr/bin*
- Die entsprechenden Objekt-Dateien in */opt/net-snmp* und/oder */opt/snmp-agents*
- Das Verzeichnisse */opt/net-snmp* und/oder */opt/snmp-agents*
- rc-Skripts werden gelöscht, aber Start-Skripts (nur aus */etc/rc2.d*) werden gesichert, damit die Benutzer-Konfiguration erhalten bleibt. Die gesicherten rc-Skripts befinden sich im Verzeichnis */etc/snmp* unter dem Namen vom Typ *ORIGINAL_NAME.bkp*.

Weder MIB-Dateien noch die Konfigurationsdateien werden gelöscht!

3.3 SNMP-Konfiguration in BS2000

Die im BS2000 auszuführenden Tätigkeiten sind in den nachfolgenden Abschnitten beschrieben. Allgemeine Empfehlungen zu diesem Thema finden Sie im Abschnitt „Sicherheitsbewusste Nutzung von SNMP“.

Die Konfiguration dient dazu, das grundsätzliche Verhalten der gesamten SNMP-Anwendungen zu steuern. Dazu können Sie eine der verfügbaren Konfigurationsmethoden verwenden (in der angegebenen Präferenz):

- Kommando-Optionen
- Umgebungsvariablen (MIBS und MIBDIRS)
- Konfigurationsdateien mit Suffix *.conf* oder *.local.conf* (wird als letztes gelesen) in folgenden Verzeichnissen:
  - `/etc/snmp` (wird zuerst gesucht)
  - `/usr/local/share/snmp`
  - `/usr/local/lib/snmp`
  - `~/.snmp`
  - `/var/net-snmp` (wird zuletzt gesucht)

Es gibt eine Reihe von SNMP-Konfigurationsdateien für die verschiedenen Komponenten von NET-SNMP:

- `<PROG_NAME>.conf` oder `<PROG_NAME>.local.conf`

  steuert die Parameter und Funktionsumfang des angegebenen Programms.

  Beispiel:
  `snmpd` -> `snmpd.conf`
  `utmAgent` -> `utmAgent.conf`
  `snmpwalk` -> `snmpwalk.conf`

- `snmp.conf` oder `snmp.local.conf`

  Allgemeine NET-SNMP Konfigurationsdatei.

- `agentx.conf` oder `agentx.local.conf`

  Master/Agent-Konfiguration für AgentX.

- `snmpapp.conf` oder `snmpapp.local.conf`

  Konfiguration der SNMP-Tools (*snmpwalk*, *snmpget*…).

Jede Anwendung kann mehrere Konfigurationsmethoden verwenden. Die meisten Anwendungen können auf jeden Fall den Inhalt der *snmp-conf*-Dateien lesen. Beachten Sie jedoch, dass Konfigurationsanweisungen, die in einer Datei verstanden werden, nicht unbedingt in einer anderen (Konfigurations-)datei verstanden werden. Anwendungen unterstützen auf der Kommandoebene eine Option *-H*, welche die Konfigurationsdateien auflistet, nach denen gesucht wird, sowie die Anweisungen, welche die betreffende Anwendung in jeder dieser Dateien versteht.

3.3.1 „Listening-Adressen“ in BS2000

Standardmäßig wartet *snmpd* auf eintreffende SNMP-Anforderungen auf allen IPV4-Adressen an UDP-Port 161 (Listening-Adresse). Dieses Verhalten lässt sich jedoch ändern, indem eine oder mehrere Listening-Adressen als Optionen an *snmpd* übergeben werden.

Eine Listening-Adresse hat folgendes Format:

```
[<transport-specifier>:]<transport-address>
```

Im einfachsten Fall besteht eine Listening-Adresse nur aus einer Portnummer,d.h. *snmpd* wartet auf allen IPv4-Adressen an diesem UDP-Port.

Andernfalls wird die Angabe *<transport-address>* wie folgt analysiert:

Transport-Addressen-Format

udp (Standard)	hostname[:port] oder IPv4-address[:port]
tcp	hostname[:port] oder IPv4-address[:port]
unix	pathname
udp6 oder udpv6 oder udpipv6	hostname[:port] oder IPv6-address[:port]
tcp6 oder tcpv6 oder tcpipv6	hostname[:port] oder IPv6-address[:port]

Bitte beachten Sie, dass für die Zeichenketten *<transport-specifier>* die Groß-/Kleinschreibung nicht relevant ist, d.h. beispielsweise sind "tcp" und "TCP" äquivalent.

Hier sind einige Beispiele mit Erläuterung:

127.0.0.1:161
: wartet an UDP-Port 161, aber nur auf der loopback-Adresse. Dies verhindert, dass *snmpd* von remote aufgerufen werden kann.

TCP:1161
: wartet auf allen IPv4-Adressen an TCP-Port 1161.

unix:/tmp/local-agent
: wartet auf dem Unix-Domänen-Socket */tmp/local-agent*.

/tmp/local-agent
: ist identisch zur vorherigen Angabe, da automatisch die Unix-Domäne angenommen wird, wenn das erste Zeichen ein „/“ ist.

udp6:10161
: wartet auf allen IPv6-Adressen an Port 10161.

Bitte beachten Sie, dass nicht immer alle oben aufgeführten Transport-Domänen verfügbar sind; beispielsweise können Host mit "noIPv6 Support" keine udp6-Adressen verwenden, sodass entsprechende Versuche mit dem Fehler "Error opening specified endpoint" quittiert werden.

3.3.2 Allgemeine SNMP-Konfiguration (snmp.conf)

3.3.2.1 Client-Verhalten

defDomain application domain
: Die Transport-Domäne, die für Anwendungen eines bestimmten Typs verwendet werden soll, sofern nichts anderes angegeben ist.

defTarget application domain target
: Das Ziel, das für Verbindungen zu einer bestimmten Anwendung verwendet werden soll, falls die Verbindung in einer bestimmten Domäne sein soll.

defaultPort PORT
: Definiert den Standard-UDP-Port, an den sich SNMP-Client-Anwendungen anmelden. Dieser Port kann überschrieben werden, indem in der AGENT-Spezifikation explizit eine Portnummer definiert wird. Weitere Details siehe man page zu *snmpcmd(1)*.
Standard: Port 161 (wenn keine Portnummer angegeben wurde).

defVersion (1|2c|3)
: Definiert die SNMP-Version, die verwendet werden soll.
Dies kann durch die Option *-v* überschrieben werden.

defCommunity STRING
: Definiert die Standard-Community, die für SNMPv1- und SNMPv2c-Anforderungen verwendet werden soll. Dies kann durch die Option *-c* überschrieben werden.

alias NAME DEFINITION
: erzeugt einen Alias (NAME) für die angegebene Transport-Definition. Der Alias kann durch alias: NAME referenziert werden .Z.B. wäre es durch die Zeile "alias home udp:127.0.0.1:6161" möglich, den Ziel-Rechner mit "alias:home" anstatt "udp:127.0.0.1:6161" zu adressieren. Dies bringt z.B. Vorteile bei der Behandlung komplexer Transport-Adressen im IPv6-Format.

dumpPacket yes
: Gibt an, ob ein hexadezimaler Dump von „raw“ SNMP-Anforderungen angezeigt werden soll, die an die Anwendung gesendet oder von ihr empfangen werden. Dies ist äquivalent zur Option *-d*..

doDebugging (1|0)
: Schaltet das Debugging für alle laufenden Anwendungen ein (1) oder aus (0).

debugTokens TOKEN[,TOKEN...]
: Definiert die Debugging-Parameter (Tokens), die aktiviert werden sollen, wenn *doDebugging* gesetzt ist. Dies ist äquivalent zur Option *-D*.

16bitIDs yes
: Schränkt RequestIDs etc auf 16-Bit-Werte ein.

  Die SNMP-Spezifikationen definieren diese ID Felder als 32-Bit-Einheiten und die Net-SNMP-Bibliothek initialisiert sie aus Sicherheitsgründen mit Zufallswerten. Es gibt jedoch bestimmte (broken) Agenten, die ID-Werte größer 2^{16} nicht verarbeiten können. Diese Option ermöglicht die Interoperabilität mit solche Agenten.

clientaddr [<transport-specifier>:]<transport-address>
: Gibt die Quell-Adressen an, die durch Kommando-basierte Anwendungen beim Senden von SNMP-Anforderungen verwendet werden sollen. Dieser Wert wird auch von *snmpd* beim Erzeugen von Benachrichtigungen verwendet.

clientSendBuf INTEGER
: Ist ähnlich zu *clientRecvBuf*, gilt aber für die Größe des Puffers, der zum Senden von SNMP-Anforderungen verwendet wird.

noRangeCheck yes
: Deaktiviert für die relevante OID die Prüfung von varbind-Werten gegen die MIB-Definition.

noTokenWarnings
: Deaktiviert Warnungen, wenn die Konfigurationsdatei unbekannte Parameter enthält.

reverseEncodeBER (1|yes|true|0|no|false)
: Steuert, wie die Codierung von SNMP-Anforderungen behandelt wird.

  Standardmäßig werden die Pakete vom Ende einer PDU an rückwärts codiert.Diese Anweisung dient dazu, dieses Verhalten zu deaktivieren und die codierte Anforderung (in plausiblerer Weise) von vorne nach hinten aufzubauen.

3.3.2.2 SNMPv3-Einstellungen

defSecurityName STRING
Definiert den Standard-Security-Namen, der für SNMP-Anforderungen verwendet werden soll. Dies kann durch die Option *-u* überschrieben werden.

defSecurityLevel noAuthNoPriv|authNoPriv|authPriv
Definiert den Standard-Security-Level, der für SNMP-Anforderungen verwendet werden soll. Dies kann durch die Option *-l* überschrieben werden.

Wenn diese Option nicht angegeben wurde dann wird *noAuthNoPriv* angenommen.

authPriv steht nur zur Verfügung, wenn die Software so compiliert wurde, dass sie die OpenSSL-Bibliotheken verwendet.

defPassphrase STRING
defAuthPassphrase STRING
defPrivPassphrase STRING
Definiert die Standard-Authentifizierung und privaten Passphrasen, die für SNMPv3-Anforderungen verwendet werden sollen. Dies kann durch die Optionen *-A* bzw. *-X* überschrieben werden.

Der Wert von *defPassphrase* wird dann für die Authentifizierung und/oder privaten Passphrasen verwendet, wenn eine der anderen Anweisungen nicht angegeben wurden.

defAuthType MD5|SHA
defPrivType DES|AES
Definiert die Standard-Authentifizierung und privaten Protokolle, die für SNMPv3-Anforderungen verwendet werden sollen. Dies kann durch die Optionen *-a* bzw. *-x* überschrieben werden.

Wird die Option nicht angegeben, dann wird für SNMPv3-Anforderungen MD5-Authentifizierung und DES-Verschlüsselung verwendet.

Falls die Software nicht so compiliert wurde, dass sie die OpenSSL-Bibliotheken verwendet, dann wird nur MD5-Authentifizierung unterstützt. Damit stehen weder SHA-Authentifizierung noch irgendeine Form der Verschlüsselung zur Verfügung.

defContext STRING
Definiert die Standard-Kontext, der für SNMPv3-Anforderungen verwendet werden soll. Dies kann durch die Option *-n* überschrieben werden.

Wird die Option nicht angegeben, dann wird als Standard-Kontext "" verwendet (d.h. die leere Zeichenkette).

defSecurityModel STRING
Definiert das Security-Modell, das für SNMPv3-Anforderungen verwendet werden soll. Der Standardwert ist *usm*, welches das einzige breit genutzte Security-Modell für SNMP darstellt.

defAuthMasterKey 0xHEXSTRING
defPrivMasterKey 0xHEXSTRING
defAuthLocalizedKey 0xHEXSTRING
defPrivLocalizedKey 0xHEXSTRING
Definiert die (hexadezimalen) Schlüssel, die für sichere SNMPv3-Kommunikation verwendet werden sollen.

SNMPv3-Schlüssel werden häufig von einer Passphrase abgeleitet, siehe obiger Abschnitt zu *defPassphrase*. Für die erhöhte Sicherheit kann jedoch ein echter „Zufalls"-Schlüssel erzeugt und stattdessen genutzt werden (der normalerweise eine bessere „Entropie" besitzt als ein Passwort, es sei denn dieses ist ungewöhnlich lang). Die Anweisungen sind äquivalent zu den Kommando-Optionen *-3m*, *-3M*, *-3k*, und *-3K*.

Lokalisierte Schlüssel sind Master-Schlüssel, die zu einem eindeutigen Schlüssel konvertiert wurden, der nur für eine bestimmte SNMP-Engine (Agent) passt.Die Länge des Schlüssels muss zur verwendeten Authentifizierung oder zum verwendeten Verschlüsselungs-Typ passen:

– Authentifizierungs-Schlüssel: MD5=16 Bytes, SHA1=20 Bytes;

– Private Schlüssel:
DES=16 Bytes (davon werden 8 Bytes als IV und nicht als Schlüssel verwendet),
AES=16 Bytes.

3.3.2.3 Server-Verhalten

persistentDir DIRECTORY
Definiert das Verzeichnis, in dem *snmpd* und *snmptrapd* die persistenten Konfigurationseinstellungen speichern.

Wird die Option nicht angegeben, wird im Verzeichnis */var/net-snmp* gespeichert.

noPersistentLoad yes
noPersistentSave yes
deaktiviert das Laden und Sichern der persistenten Konfigurationseinstellungen.

Dies unterbricht SNMPv3-Operationen (und anderweitiges Verhalten, das sich darauf verlässt, dass Änderungen beim Anwendungs-Restart erhalten bleiben). Bitte mit Vorsicht verwenden!

tempFilePattern PATTERN
Definiert ein Dateinamen-Template für das Erzeugen temporärer Dateien und für die Behandlung von Eingaben an und Ausgaben von externen Shell-Kommandos. Wird von den Funktionen *mkstemp()* und *mktemp()* verwendet.

Wird die Option nicht angegeben, wird */tmp/snmpdXXXXXX* als Muster verwendet.

3.3.2.4 MIB-Handling

mibdirs DIRLIST
Gibt eine Liste von Verzeichnissen an, in denen nach MIB-Dateien gesucht wird. Dieser Wert kann von der Umgebungsvariablen MIBDIRS und der Option *-M* überschrieben werden.

mibs MIBLIST
Gibt eine Liste von MIB-Modulen (nicht Dateien) an, die geladen werden sollen. Dieser Wert kann von der Umgebungsvariablen MIBS und der Option *-m* überschrieben.

mibfile FILE
Gibt eine (einzelne) MIB-Datei an, die geladen werden soll, zusätzlich zu der im Parameter *mibs* angegeben Liste (oder äquivalenter Konfigurationsangaben). Dieser Wert kann von der Umgebungsvariablen MIBFILES überschrieben werden.

showMibErrors (1|yes|true|0|no|false)
Gibt an, ob Fehler beim Parsen der MIB angezeigt werden sollen.

commentToEOL (1|yes|true|0|no|false)
Gibt an, ob beim Parsen der MIB das Kommentar-Ende streng konform behandelt werde soll. Viele MIB-Autoren nehmen an, dass ASN.1-Kommentare sich bis zum Ende der Kommentarzeile ausdehnen anstatt mit dem nächsten "--" Token beendet zu werden.
Dieser Parameter kann dazu verwendet werden, um solche (streng genommen inkorrekten) MIBs zu akzeptieren.
Beachten Sie, dass diese Anweisung früher fälschlicherweise *strictCommentTerm* hieß, sich jedoch genau umgekehrt verhielt als dem Namen nach zu erwarten war. Der frühere Parameter wird aus Kompatibilitätsgründen noch akzeptiert.

mibAllowUnderline (1|yes|true|0|no|false)
Gibt an, ob das Unterstreichen von Zeichen in MIB-Objektnamen und Aufzählungswerten erlaubt sein soll. Dieser Parameter kann dazu verwendet werden, um solche (streng genommen) inkorrekten MIBs zu akzeptieren.

mibWarningLevel INTEGER
Minimale Warnstufe für die vom MIB-Parser ausgegebene Warnungen.

3.3.2.5 Ausgabe-Konfiguration

Die meisten Optionen in diesem Abschnitt können auch über Kommando-Optionen konfiguriert werden.

Diese Optionen sind nur für die Programme des Produkts NET-SNMP relevant.

logTimestamp (1|yes|true|0|no|false)
: Gibt an, ob die Kommandos das Fehler-/Meldungs-Logging mit Zeitstempel ausgeben sollen oder nicht. Bitte beachten Sie, dass die Ausgabe mit Zeitstempel unübersichtlicher aussieht, wenn der Programmcode, der das Logging veranlasst, diejenigen Meldungen aufsteigend protokolliert, die vor Ausgabe an die Logging-Routine nicht zeilenweise gepuffert werden. Diese Option kann nur bei aktiviertem Datei-Logging genutzt werden.

printNumericEnums (1|yes|true|0|no|false)
: Äquivalent zu -*Oe*. Entfernt die symbolischen Bezeichner aus Aufzählungswerten:

```
$ snmpget -c public -v 1 localhost ipForwarding.0
IP-MIB::ipForwarding.0 = INTEGER: forwarding(1)
$ snmpget -c public -v 1 -Oe localhost ipForwarding.0
IP-MIB::ipForwarding.0 = INTEGER: 1
```

printNumericOids (1|yes|true|0|no|false)
: Äquivalent zu -*On*. Zeigt die OIDs numerisch an:

```
.1.3.6.1.2.1.1.3.0 = Timeticks: (14096763) 1 day, 15:09:27.63
```

dontBreakdownOids (1|yes|true|0|no|false)
: Äquivalent zu -*Ob*. Zeigt die Tabellenindizes numerisch an anstatt zu versuchen, die Sub-Identifier der Instanzen als String oder OID-Wert zu interpretieren:

```
$ snmpgetnext -c public -v 1 localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0."wes" = xxx
$ snmpgetnext -c public -v 1 -Ob localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0.3.119.101.115 = xxx
```

escapeQuotes (1|yes|true|0|no|false)
: Äquivalent zu -*OE*. Modifiziert Index-Strings durch Entwerten („escape“) von Anführungszeichen:

```
$ snmpgetnext -c public -v 1 localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0."wes" = xxx
$ snmpgetnext -c public -v 1 -OE localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0.\"wes\" = xxx
```

Dadurch kann die Ausgabe in Shell-Kommandos verwendet werden.

quickPrinting (1|yes|true|0|no|false)
Äquivalent zu *-Oq*. Entfernt das Gleichheitszeichen und die Typ-Information bei der Ausgabe von varbind-Werten:

```
SNMPv2-MIB::sysUpTime.0 1:15:09:27.63
```

printValueOnly (1|yes|true|0|no|false)
Äquivalent zu *-Ov*. Gibt nur den varbind-Wert aus, ohne OID:

```
$ snmpget -c public -v 1 -Oe localhost ipForwarding.0
INTEGER: forwarding(1)
```

dontPrintUnits (1|yes|true|0|no|false)
Äquivalent zu *-OU*. Die Einheit eines Werts wird nicht mit ausgegeben.

numericTimeticks (1|yes|true|0|no|false)
Äquivalent zu *-Ot*. Gibt die Zeitwerte (TimeTicks) als nicht-aufbereitete Zahlen aus:

```
SNMPv2-MIB::sysUpTime.0 = 14096763
```

printHexText (1|yes|true|0|no|false)
Äquivalent zu *-OT*. Für hexadezimale Werte wird zusätzlich eine abdruckbare Version mit ausgegeben.

hexOutputLength integer
Gibt an, wo die Ausgabe von hexadezimalen Werten umgebrochen werden soll. Der Wert 0 bedeutet keinen Zeilenumbruch. Standardwert 16.

suffixPrinting (0|1|2)
Der Wert 1 ist äquivalent zu *-O*. Er gibt den MIB-Objektnamen aus (plus die Instanz oder andere Sub-Identifier):

```
sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63
```

Der Wert 2 ist äquivalent zu *-OS*. Er gibt den Namen der MIB und den Objektnamen aus:

```
SNMPv2-MIB::sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63
```

Dies ist das Standardausgabeformat für OIDs.

oidOutputFormat (1|2|3|4|5|6)
Bildet die Optionen *-O* wie folgt ab:
-Os=1, *-OS=2*, *-Of=3*, *-On=*4, *-Ou=5*.
Der Wert 6 passt zu keiner Option *-O*. Er unterdrückt die Ausgabe.

‘-Of’
Bei der OID-Ausgabe wird die gesamte Liste der MIB-Objekte mit ausgegeben:

```
iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0
```

‘-Ou’
Gib die OID im „klassischen“ UCD-Stil aus (vererbt Original CMU-Code). D.h. es wird eine Reihe von "Standard"-Präfixen aus der OID entfernt und die verbleibende Liste der MIB-Objektnamen angezeigt (plus andere Sub-Identifier):

```
system.sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63
```

extendedIndex (1|yes|true|0|no|false)
Äquivalent zu *-OX*. Gibt die Tabellenindizes in einem "programmfreundlicheren" Format aus, indem ein typisches Index-Format im Stil eines Arrays imitiert wird:

```
$ snmpgetnext -c public -v 1 localhost ipv6RouteTable

IPv6-MIB::ipv6RouteIfIndex.63.254.1.0.255.0.0.0.0.0.0.0.0.0.0.64.1 =
INTEGER: 2

$ snmpgetnext -c public -v 1 -OE localhost ipv6RouteTable

IPv6-MIB::ipv6RouteIfIndex[3ffe:100:ff00:0:0:0:0:0][64][1] = INTEGER: 2
```

noDisplayHint (1|yes|true|0|no|false)
Deaktiviert die Verwendung von DISPLAY-HINT Informationen beim Parsen von Indizes und zu setzenden Werten.

3.3.3 AgentX-Konfiguration (agentx.conf)

Folgende Parameter sind relevant :

agentxsocket
Bind-Adresse von AgentX

agentxperms
Socket-Berechtigungen von AgentX

agentxRetries
Wiederholungen von AgentX

agentxTimeout
Timeout (Sekunden) von AgentX

Die Beschreibung finden Sie in Abschnitt „AgentX Optionen“.

3.3.4 Kommando-Optionen

Die Konfigurationsanweisungen aus Konfigurationsdateien (bezogen auf das entsprechende Programm) können auch per Kommando übergeben werde, z.B auf folgende Weise:

```
--CONF_DIRECTIVE="VALUE1[ VALUE2…]"
```

Beispiel:

```
snmpd --rwcommunity="public "
```

Es gibt auch programmspezifische Kommando-Optionen, die per Programmaufruf mit Option --*help* ermittelt werden können. Einige davon lassen sich nicht per Konfigurationsdatei konfigurieren. Die folgende Tabelle zeigt eine Reihe wichtiger Optionen, die von Master-Agent, den Agenten und SNMP-Tools verwendet werden können.

Option	**Beschreibung**
Allgemeine Optionen	
-m MIBLIST	MIBLIST anstatt der Standard-MIB-Liste verwenden
-M DIRLIST	DIRLIST für die Suche nach den Anlageorten der MIBs verwenden
-d	Dump (in hexadezimal) von den „ raw“ gesendeten und empfangenen SNMP-Pakete erzeugen
-D[TOKEN[,...]]	Debugging-Ausgabe für die angegebenen Token(s) aktivieren. Für spezielle Tokens siehe Abschnitt „Debug-Optionen“
-H	Liste der Konfigurationsanweisungen ausgeben, die das Kommando versteht, und danach beenden („exit)
-f	Nicht aus der Shell verzweigen („fork“)
-p FILE	Prozess-ID in FILE speichern
-L	Logging-Optionen
-Le	Nachrichten-Logging nach Standard-Error
-Lf FILE	Nachrichten-Logging in die angegebene Datei
-Lo	Nachrichten-Logging in Standard-Out
-Ls FACILITY	Nachrichten-Logging via *syslog*, wobei folgende FACILITY-Einstellungen verwendet werden: 'd' für LOG_DAEMON, 'u' für LOG_USER, '0'-'7' für LOG_LOCAL0 - LOG_LOCAL7)
-LE pri	Nachrichten-Logging mit Priorität 'pri' und höher nach Standard-Error.
-LS pri	Nachrichten-Logging mit Priorität 'pri' und höher via *syslog*
-LE p1-p2	Nachrichten-Logging mit Priorität zwischen ‘p1’ und 'p2' (inklusive) nach Standard-Error.

Option	Beschreibung
-LS p1-p2	Nachrichten-Logging mit Priorität zwischen ‘p1’ und 'p2' (inklusive) nach s*y*slog
Dämon-spezifische Optionen	
-c FILE[,...]	FILE(s) als Konfigurationsdatei(en) einlesen
-C	Die Standard-Konfigurationsdatei(en) nicht einlesen
-x ADDRESS	ADDRESS als AgentX-Adresse verwenden.
-X	Als AgentX-Master ablaufen lassen
snmpd-spezifische Optionen	
-a	Logging für Adressen durchführen
-q	Informationen in einem leichter zu analysierenden Format ausgeben
snmptrapd-Optionen	
-a	Traps zu Authentifizierungsfehlern ignorieren
-n	Numerische Adressen verwenden anstatt zu versuchen Hostnamen zu ermitteln (kein DNS)
-t	Traps nicht nach *syslog* protokollieren

Für *-LF* und *-LS* muss die Priorität vor dem Token FILE oder FACILITY angegeben werden. Folgenden Prioritäten werden akzeptiert:

Prioritäts-Stufe	Bedeutung
0 oder !	LOG_EMERG
1 oder a	LOG_ALERT
2 oder c	LOG_CRIT
3 oder e	LOG_ERR
4 oder w	LOG_WARNING
5 oder n	LOG_NOTICE
6 oder i	LOG_INFO
7 oder d	LOG_DEBUG

Standardmäßig wird die Ausgabe mit Priorität LOG_INFO nach *syslog* protokolliert.

3.4 NET-SNMP konfigurieren

3.4.1 SNMP-Dämon snmpd konfigurieren (snmpd.conf)

Der Net-SNMP-Dämon verwendet eine oder mehrere Konfigurationsdateien, mit denen sich sein Ablauf und die zur Verfügung gestellten Management-Informationen steuern lassen. Beispielsweise kann *snmpd* sowohl die Konfigurationsanweisungen in der Datei *snmpd.conf* als auch in der Datei *snmp.conf* richtig interpretieren.

Der folgende Abschnitt gibt allgemeine Informationen über die Konfiguration des Dämons mittels der Datei *snmpd.conf*. Da NET-SNMP ein OpenSource-Projekt ist, finden Sie weitere Informationen und Beispiele bei den entsprechenden Web Services und man pages.

Ein Teil der OpenSource-Funktionalität von Net-SNMNP steht im BS2000 nicht zur Verfügung. Daher vergleichen die bitte die Informationen im Web mit dem vorliegenden Handbuch.

3.4.1.1 Verhalten des Agenten

Obwohl die meisten Konfigurationsanweisungen die vom Agenten gelieferten MIB-Informationen betreffen, gibt es einige Anweisungen, die das Verhalten von *snmpd* steuern, wenn man ihn einfach als Dämon betrachtet, der einen Netwerk-Service zur Verfügung stellt.

agentaddress [<transport-specifier>:]<transport-address>[,...]
Definiert eine Liste von "Listening"-Adressen an, denen auf ankommende SNMP-Anforderungen gewartet wird. Details siehe Abschnitt „„Listening-Adressen“ in BS2000“.

Standard ist das Warten an UDP-Port 161 für alle IPv4-Adressen.

Sie können mehr als eine Listening-Adresse angeben.

Aus technischen Gründen können IPv6-Ports nicht auf IPv4-Ports abgebildet werden, z.B.:

`agentaddress udp:161,tcp:161` - funktioniert

`agentaddress udp:161,udp6:161` – funktioniert nicht

`agentaddress udp:161,udp6:163` – funktioniert

agentgroup {GROUP|#GID}
Wechselt nach dem Öffnen der Listener-Port(s) zur angegebenen Gruppe. Diese kann als Gruppenname (GROUP) oder als numerische Gruppen-Id beginnend mit '#' (#GID) angegeben werden.

agentuser {USER|#UID}
: Wechselt nach dem Öffnen der Listener-Port(s) zum angegebenen Benutzer. Dieser kann als Benutzername (USER) oder als numerische Benutzer-Id beginnend mit '#' (#UID) angegeben werden.

leave_pidfile yes
: weist den Agenten an, seine *pid* -Datei beim Shutdown nicht zu löschen. Äquivalent zur Angabe von *-U* beim Kommando.

maxGetbulkRepeats NUM
: Legt die maximale Anzahl der Antworten fest, die für eine einzelne Variable in einer *getbulk* Anforderung erlaubt sind. Der Wert 0 aktiviert die Standardeinstellung, der Wert -1 bedeutet keine Beschränkung. Da der Speicher vorab zugewiesen wird, wird es als unsicher angesehen, den Wert bei 'unbegrenzt' zu belassen, falls Sie der Gesamtheit der Benutzer nicht vertrauen. Eine größere Wiederholungsanzahl wird auf diesen Wert gekürzt.

  Standard ist -1.

maxGetbulkResponses NUM
: Legt die maximale Anzahl der Antworten fest, die für eine *getbulk* Anforderung erlaubt sind. Diese wird standardmäßig auf 100 gesetzt. Der Wert 0 aktiviert die Standardeinstellung, der Wert -1 bedeutet keine Beschränkung. Da der Speicher vorab zugewiesen wird, wird das Setzen des Wertes auf 'unbegrenzt' als unsicher angesehen, falls Sie der Gesamtheit der Benutzer nicht vertrauen.

  Im Allgemeinen darf die Gesamtanzahl der Antworten den Wert in *maxGetbulkResponses* nicht überschreiten, d.h. die Gesamtanzahl ergibt sich, indem ein ganzzahliges Vielfaches der Anzahl der angeforderten Variablen multipliziert mit der errechneten Anzahl der Wiederholungen gerade noch unter diesem Wert bleibt.

  Beachten Sie auch, dass *maxGetbulkRepeats* zuerst abgehandelt wird.

3.4.1.2 AgentX Optionen

Net-SNMP und die ergänzenden Produkte unterstützen das AgentX Protokoll (RFC 2741) sowohl im Master-Agent als auch in den Agenten-Rollen. Die Verwendung dieses Mechanismus‘ setzt voraus, dass der Dämon das Agentx Modul explizit aktiviert hat. (z.B. über die Datei *snmpd.conf*).

Es gibt zwei Anweisungen, die speziell für den Ablauf als AgentX Master relevant sind:

master agentx
: aktiviert die AgentX-Funktionalität und veranlasst den Agenten, auf ankommende AgentX Registrierungen zu warten. Diese Funktion kann auch über die Kommando-Option *-x* aktiviert werden (um einen alternativen Listener-Socket anzugeben).

agentXPerms SOCKPERMS [DIRPERMS [USER|UID [GROUP|GID]]]
: Definiert die Berechtigungen und den Eigentümer des Unix Domänen-Socket von AgentX sowie der Eltern-Verzeichnisse dieses Socket. SOCKPERMS und DIRPERMS müssen oktale Ziffern sein (siehe *chmod(1)*). Standardmäßig ist dieser Socket nur für jene Agenten zugreifbar, die dieselbe Benutzer-Id wie der (Master-)Agent besitzen.

Folgende Anweisung ist speziell für den Ablauf als AgentX Agent relevant:

agentXPingInterval NUM
: bewirkt, dass der Agent alle NUM Sekunden versucht, sich (wieder) mit dem Master zu verbinden, falls er noch keine Verbindung hat oder keine Verbindung mehr hat.

Die restlichen Anweisungen sind sowohl für den AgentX Master als auch für die Agenten relevant:

agentXSocket [<transport-specifier>:]<transport-address>[,...]
: Definiert die Adresse, an welcher der Master-Agent wartet oder zu welcher der Agent die Verbindung aufbauen soll. Standard ist der Unix Domänen-Socket */var/agentx/master*. Eine andere übliche Alternative ist *tcp:localhost:705*.

Die Angabe eines AgentX Socket aktiviert nicht automatisch die AgentX-Funktionalität (im Gegensatz zur Kommando-Option *-x*).

agentXTimeout NUM
: Definiert die Timeout-Zeitspanne (NUM Sekunden) für eine AgentX-Anforderung. Standard ist 1 Sekunde.

agentXRetries NUM
: Definiert die Anzahl der Wiederholungen für eine AgentX-Anforderung. Standard sind 5 Wiederholungen.

3.4.1.3 SNMPv3 Konfiguration

SNMPv3 fordert, dass ein SNMP-Agent eine eindeutige *engineID* festlegen muss, um auf SNMPv3 Anforderungen zu antworten. Diese Id wird normalerweise automatisch bestimmt, indem zwei nicht rational vorhersagbare Werte verwendet werden:

– eine (Pseudo-)Zufallszahl und
– die aktuelle Zeit in Sekunden.

Diese ist die empfohlene Vorgehensweise.

Es gibt jedoch die Möglichkeit, die *engineID* auf andere Art zu bestimmen:

engineID STRING
: Gibt an, dass *engineID* aus der in STRING angegebenen Zeichenkette erzeugt werden soll.

engineIDType 1|2|3
: Gibt an, dass die *engineID* aus der IPv4-Adresse (1), der IPv6-Adresse (2) oder der MAC-Adresse erzeugt werden soll. Bitte beachten Sie, dass eine Änderung der IP-Adresse (oder ein Austausch/Umschalten der Netzwerk-Karte) Probleme verursachen kann.

engineIDNic INTERFACE
: Legt fest, welche Schnittstelle zur Bestimmung der MAC-Adresse verwendet werden soll. Fall *engineID* Typ 3 nicht angegeben ist, dann ist diese Anweisung wirkungslos.

  Standard ist *eth0*.

3.4.1.4 SNMPv3 Authentifizierung

SNMPv3 wurde ursprünglich mittels des User-Based Security Model (USM) definiert, das eine private Liste von Benutzern und Schlüsseln enthält, die spezifisch für das SNMPv3-Protokoll sind.

Um USM-basierte, SNMPv3-spezifische Benutzer zu verwenden, müssen Sie diese explizit erzeugen:

```
createUser [-e ENGINEID] username (MD5|SHA) authpassphrase [DES|AES]
           [privpassphrase]
```

Als Authentifizierungs-Typen müssen MD5 oder SHA verwendet werden. Als private Protokolle müssen DES oder AES verwendet werden. Wird die private Passphrase (*privpassphrase*) nicht angegeben, dann wird als private Passphrase die Authentifizierungs-Passphrase (*authpassphrase*) genommen. Bitte beachten Sie, dass die erzeugten Benutzer nicht nutzbar sind, solange sie nicht in die VCAM-Zugangskontroll-Tabellen aufgenommen wurden (siehe Abschnitt „Zugangskontrolle“).

ACHTUNG!
Die Minimallänge der Passphrase ist 8 Zeichen.

3.4.1.5 Zugangskontrolle

snmpd unterstützt das in RFC 2575 definierte View-Based Access Control Model (VACM), um festzulegen, wer Informationen suchen oder ändern darf. Zu diesem Zweck gibt es für *snmpd* bezüglich Zugangskontrolle verschiedene Anweisungen.

Klassische Zugangskontrolle

Die einfachsten Anforderungen an die Zugangskontrolle können über die Anweisungen *rouser*/*rwuser* (für SNMPv3) oder *rocommunity*/*rwcommunity* (für SNMPv1 oder SNMPv2c) festgelegt werden.

rouser [-s SECMODEL] USER [noauth|auth|priv [OID | -V VIEW [CONTEXT]]]
rwuser [-s SECMODEL] USER [noauth|auth|priv [OID | -V VIEW [CONTEXT]]]
Spezifiziert einen SNMPv3-Benutzer und gibt ihm das Zugriffsrecht read-only (GET und GETNEXT) bzw. read-write (GET, GETNEXT und SET).
Standardmäßig ermöglicht dies authentifizierten (inkl. verschlüsselten) SNMPv3-Anforderungen, mittels Standard-Kontext auf den kompletten OID-Baum zuzugreifen .

Alternativ kann *noauth* (minimaler Security-Level, erlaubt nicht-authentifizierte Anforderungen) oder *priv* (erzwingt Verschlüsselung) angegeben werden. Der Parameter OID schränkt den Zugriff für diesen Benutzer ein auf den Teilbaum, der zur angegebenen OID gehört bzw. auf den benannten VIEW. Zusätzlich kann ein Kontext oder mit <context>* ein Kontext-Präfix angegeben werden. Wird keine Kontext-Feld angegeben, dann lässt die Anweisung alle Kontexte zu, die möglich sind.

rocommunity COMMUNITY [SOURCE [OID | -V VIEW [CONTEXT]]]
rwcommunity COMMUNITY [SOURCE [OID | -V VIEW [CONTEXT]]]
Spezifiziert eine SNMPv1- oder SNMPv2c-Community und gibt dieser das Zugriffsrecht read-only (GET und GETNEXT) bzw. read-write (GET, GETNEXT und SET).
Standardmäßig ermöglicht dies den Zugriff auf den kompletten OID-Baum, unabhängig davon, von wo aus sie gesendet wurden.

Mit dem SOURCE-Parameter kann man den Zugriff auf Anforderungen einschränken, die vom angegebenen System (oder Systemen) kommen; Details siehe *com2sec*.

Der Parameter OID schränkt den Zugriff für diese Community ein auf den Teilbaum, der zur angegebenen OID gehört bzw. auf den benannten VIEW. Kontexte sind typischerweise für Community-basierte SNMP-Versionen weniger relevant, aber das Verhalten ist hier dasselbe.

rocommunity6 COMMUNITY [SOURCE [OID | -V VIEW [CONTEXT]]]
rwcommunity6 COMMUNITY [SOURCE [OID | -V VIEW [CONTEXT]]]
sind Anweisungen für empfangene Anforderungen über IPv6 (falls der Agent derartige Transport-Domänen unterstützt). Die Parameter OURCE, OID, VIEW und CONTEXT werden genauso interpretiert wie bei IPv4.

Für eine bestimmten SNMPv3-Benutzer (bzw. für eine Community) sollte nur **eine** Anweisung angegeben werden. Es ist nicht sinnvoll, für einen und denselben SNMPv3-Benutzer (bzw. Community) sowohl eine *rouser* als auch eine *rwuser* Anweisung anzugeben. Die Anweisung *rwuser* umfasst alle Rechte von *rouser* (ebenso wie das Zulassen der SET-Unterstützung). Dasselbe gilt für Community-basierte Anweisungen. Für komplexere Zugriffsanforderungen (z.B. Zugriff auf zwei oder mehr verschiedene OID-Teilbäume oder unterschiedliche Views für GET- und SET-Anforderungen) sollte man einen der anderen Zugangskontroll-Mechanismen nutzen. Bitte beachten Sie, dass falls mehreren verschiedenen Communitys oder SNMPv3-Benutzern derselbe Zugriffs-Level gewährt werden soll, es effizienter ist, die Haupt-Anweisungen der VACM-Konfiguration zu verwenden (siehe unten).

VACM Konfiguration

Die volle Flexibilität der VACM-Konfiguration wird in Form der vier Anweisungen *com2sec*,*group*, *view* und *access* bereit gestellt. Damit lassen sich zugrunde liegenden VACM-Tabellen direkt konfigurieren.

com2sec [-Cn CONTEXT] SECNAME SOURCE COMMUNITY
com2sec6 [-Cn CONTEXT] SECNAME SOURCE COMMUNITY
Bilden einen SNMPv1 oder SNMPv2c Community String auf einen Security-Namen ab, entweder von einem bestimmten Bereich von Source-Adressen oder global (Standard) Eine beschränkte Source (SOURCE) kann entweder ein bestimmter Hostname (oder Adresse) sein oder ein Subnetz sein; dargestellt als IP-Maske (z.B. 10.10.10.0/255.255.255.0), IP/BITS (z.B. 10.10.10.0/24) oder IPv6-Äquivalente.

Derselbe Community-String kann in mehreren verschiedenen Anweisungen angegeben werden (wahrscheinlich mit verschiedenen Source-Parametern), wobei die erste Source/Community-Kombination, die zu einer eintreffenden Anforderungen passt, ausgewählt wird. Verschiedene Source/Community-Kombinationen können also auf denselben Security-Namen abgebildet werden.

Falls mit *-Cn* ein CONTEXT angegeben wird, dann wird der Community-String auf einen Security-Namen im benannten SNMPv3-Kontext abgebildet. Andernfalls wird der Standard-Kontext ("") verwendet.

com2secunix [-Cn CONTEXT] SECNAME SOCKPATH COMMUNITY
ist die Unix-Domänen-Version von *com2sec*.

group GROUP {v1|v2c|usm|tsm|ksm} SECNAME
Bildet einen Security-Namen (im angegebenen Security-Modell) auf eine benannte Gruppe ab. Derselbe Gruppenname kann in mehrere *group*-Anweisungen angegeben und ermöglichen es damit, dass eine einzige Zugriffseinstellung für mehrere Benutzer und/oder Community Strings gilt.
Bitte beachten Sie, dass die Gruppen für die beiden Security-basierten Modelle getrennt eingerichtet werden müssen: eine *singlecom2sec*-Anweisung (oder äquivalent) wird typischerweise von zwei *group*-Anweisungen begleitet.

view VNAME TYPE OID [MASK]
Definiert einen benannten "View" - eine Untermenge des gesamten OID-Baums. Dies wird meist ein einzelner Teilbaum sein, aber es können mehrere *view*-Anweisungen mit demselben View-Namen (VIEW) angegeben werden, um eine komplexere Menge von OIDs zu definieren. TYPE ist entweder *included* oder *excluded*, womit man wiederum einen komplexeren View definieren kann (z.B. indem man bestimmte sensitive Objekte von einem ansonsten zugreifbaren Teilbaum ausschließt).

MASK ist eine Liste hexadezimaler Oktette (optional durch '.' oder ':' getrennt) mit den gesetzten Bits, die angeben, gegen welche Sub-Identifier im View OID geprüft wird. Wird MASK nicht angegeben, dann muss die OID standardmäßig genau passen (alle Bits gesetzt), wodurch ein einfacher OID-Teilbaum definiert wird.

Beispiele

```
view iso1 included .iso 0xf0
view iso2 included .iso
view iso3 included .iso.org.dod.mgmt 0xf0
```

Diese Anweisungen definieren alle denselben View, der den gesamten Teilbaum *iso(1)* umfasst (wobei das dritten Beispiel die Sub-Identifier ignoriert, die nicht durch die Maske abgedeckt werden).

Noch besser ist es, die Maske für die Definition eines Views zu verwenden, der eine bestimmte Reihe (oder Reihen) in einer Tabelle abdeckt, indem gegen den entsprechenden Tabellenindex-Wert geprüft, aber der Spalten-Sub-Identifier weggelassen wird:

```
view ifRow4 included .1.3.6.1.2.1.2.2.1.0.4 0xff:a0
```

Bitte beachten Sie, dass in einer Maske mit mehr als 8 Bits einzelne Oktette per ':' getrennt werden müssen.

access GROUP CONTEXT {any|v1|v2c|usm|tsm|ksm} LEVEL PREFX READ WRITE NOTIFY

Bildet eine Gruppe von Benutzern/Communitys (mit bestimmtem Security-Modell und minimalem Security-Level sowie einem spezifischem Kontext) auf einen von drei Views ab, abhängig von der Anforderung, die bearbeitet wird.

LEVEL ist einer der Werte *noauth*, *auth*, oder *priv*. PREFX gibt an, wie CONTEXT zum Kontext der Anforderung passen muss, entweder exakt oder per Präfix.

READ, WRITE und NOTIFY geben den View an, der für GET*-, SET- und TRAP-/INFORM-Anforderungen verwendet werden soll (obwohl der NOTIFY View aktuell nicht verwendet wird). Für den Zugriff über *v1*- oder *v2c* muss LEVEL den Wert *noauth* haben.

Typed-View Konfiguration

Die letzte Anweisungsgruppe erweitert den VACM-Ansatz in Richtung flexiblerer Mechanismen, was sich für weitere Zugangskontroll-Mechanismen verwenden lässt. Dies wird eher für die Definition verschiedener unterschiedlicher View-Typen benutzt, als für die festen drei Views des Standard-VACM-Mechanismus. Soweit es den Haupt-Agenten von SNMP betrifft, sind es die zwei Haupt-Viewtypen *read* und *write*, die den Werten READ und WRITE der Haupt-Anweisung *access* entsprechen.

authcommunity TYPES COMMUNITY [SOURCE [OID | -V VIEW [CONTEXT]]]
: ist eine Alternative zu den Anweisungen *rocommunity*/*rwcommunity*. TYPES hat normalerweise den Wert *read* oder *read/write*.
Die VIEW-Spezifikation kann entweder ein OID-Teilbaum (wie zuvor) oder - für größerer Flexibilität - ein benannter View sein (definiert mit der *view*-Anweisung). Wird diese weggelassen, dann wird der Zugriff auf den gesamten OID-Baum erlaubt.
Wird CONTEXT angegeben, dann wird der SNMPv3-Zugriff innerhalb dieses Kontexts definiert. Andernfalls gilt der Standard-Kontext ("").

authuser TYPES [-s MODEL] USER [LEVEL [OID | -V VIEW [CONTEXT]]]
: ist eine Alternative zu den Anweisungen *rouser*/*rwuser*. Die Parameter TYPES, OID, VIEW und CONTEXT haben dieselbe Bedeutung wie bei *authcommunity*

authgroup TYPES [-s MODEL] GROUP [LEVEL [OID | -V VIEW [CONTEXT]]]
: ist ein "Begleiter" der Anweisung *authuser*, der den Zugriff auf eine bestimmte Gruppe angibt, die wie üblich mit der *group*-Anweisung definiert ist. Sowohl *authuser* als auch *authgroup* sind Standard für authentifizierte Anforderungen. LEVEL kann auch bei *noauth* oder *priv* angegeben werden, um nicht-authentifizierte Anforderungen bzw. Verschlüsselung zu ermöglichen.
Sowohl die *authuser* als auch *authgroup* Anweisung sind die Voreinstellung, um den Zugriff für SNMPv3/USM zu definieren. Um ein alternatives Security-Modell anzugeben, verwenden Sie die Option *-s* (es werden dieselben Werte für den Zugriff verwendet wie oben).

authaccess TYPES [-s MODEL] GROUP VIEW [LEVEL [CONTEXT]]
: konfiguriert ebenfalls den Zugriff auf eine bestimmte Gruppe durch Angabe von Name und Typ des Views, der verwendet werden soll. Die Parameter MODEL und LEVEL werden genauso interpretiert wie bei *authgroup*. Falls CONTEXT angegeben wird, dann wird der Zugriff innerhalb dieses SNMPv3-Kontexts konfiguriert (bzw. im Kontext mit einem Präfix wenn CONTEXT mit einem '*' endet). Andernfalls gilt der Standard-Kontext ("").

setaccess GROUP CONTEXT MODEL LEVEL PREFIX VIEW TYPES
: ist das direkte Äquivalent zur ursprünglichen *access*-Anweisung mit den entsprechenden View-Typen wie *read* oder *read/write*. Alle anderen Parameter werden genauso interpretiert wie bei *access*.

3.4.1.6 System Gruppe

Die meisten der skalaren Objekte in der 'System' Gruppe lassen sich wie folgt mittels der Datei *snmpd.conf* konfigurieren:

sysLocation STRING
sysContact STRING
sysName STRING
: Setzt die System Location und den System Contact bzw. den System Name für den Agenten (*sysLocation.0*, *sysContact.0* und *sysName.0*). Normalerweise sind diese Objekte per Anforderung SNMP SET mit der passenden Berechtigung beschreibbar. Wenn jedoch eine dieser Anweisungen angegeben wurde, dann wird das zugehörige Objekt auf read-only gesetzt und ein Schreibversuch mit SET wird mit der Fehlermeldung *notWritable* quittiert.

sysServices NUMBER
: Setzt den Wert des Objekts *sysServices.0*. Für ein Host-System ist 72 ein vernünftiger Wert (Anwendung + die end-to-end Layer). Falls diese Anweisung nicht angegeben wird, dann für das Objekt *sysServices.0* kein Wert zurückgemeldet.

sysDescr STRING
sysObjectID OID
: Setzt die System Description oder die Object Id für den Agenten, Obwohl diese Objekte nicht per SNMP beschreibbar sind, können diese Anweisungen vom einem Netzwerk-Administrator verwendet werden, um passende Werte für sie zu definieren.

3.4.1.7 Aktives Monitoring

Das normale Verhalten eines SNMP-Agenten besteht darin, auf ankommende SNMP-Anforderungen zu warten und sie dann zu beantworten; Falls keine Anforderungen eintreffen, dann wird ein Agent typischerweise auch keine Aktionen anstoßen.

Dieser Abschnitt beschreibt verschiedene Anweisungen, mit denen *snmpd* so konfiguriert werden kann, dass er eine aktivere Rolle einnimmt.

Behandlung von Benachrichtigungen (Notifications)

trapcommunity STRING
: Definiert den Standard Community String für das Senden von Traps. Bitte beachten Sie, dass diese Anweisung vor jeder anderen Community-basierten Anweisung für Trap-Ziele, welche die Anweisung *trapcommunity* benötigen, stehen muss.

trapsink HOST [COMMUNITY]
trap2sink HOST [COMMUNITY]
informsink HOST [COMMUNITY]
Definiert die Adresse eines Benachrichtigungs-Empfängers, an den Benachrichtigungen des Formats SNMPv1 TRAPS, SNMPv2c TRAP2s, oder SNMPv2 INFORM gesendet werden sollen. Falls COMMUNITY nicht angegeben wird, dann wird der neueste *trapcommunity* String verwendet.

Falls die Transportadresse keine explizite Port-Angabe enthält, dann wird PORT verwendet. Falls dieser nicht angegeben wurde, dann wird der übliche SNMP-Port (162) verwendet.

Falls mehrere *sink*-Anweisungen angegeben werden, dann werden von jeder Benachrichtigung mehrere Kopien (mit passendem Format) erzeugt.

authtrapenable {1|2}
legt fest, ob Authentifizierungsfehler-Traps erzeugt werden sollen (*disabled(2)*, Standard). Normalerweise hat das zugehörige MIB-Objekt (*snmpEnableAuthenTraps.0*) die Eigenschaft read-write. Wird jedoch diese Anweisung angegeben, dann wird das Objekt auf read-only gesetzt und ein Setzen des Wertes mit SET wird mit der Fehlermeldung *notWritable* quittiert.

v1trapaddress HOST
Definiert die Agenten-Adresse, welche in SNMPv1 TRAPs eingefügt wird. Wird diese Option weggelassen, dann wird eine beliebige lokale IPv4 gewählt. Diese Option ist vor allem dann nützlich, wenn der Agent von außen her nur als spezifische Adresse sichtbar ist (z.B. wegen Adresse-Umsetzung oder Firewall).

3.4.2 DisMan Event MIB

Mit den im Abschnitt Abschnitt „Aktives Monitoring“ beschriebenen Anweisungen kann man konfigurieren, wohin Traps gesendet werden sollen, nicht aber, wann solche Traps gesendet werden sollen (oder welche Traps erzeugt werden sollen). Dies ist die Aufgabe der Event MIB, die durch die Arbeitsgruppe Distributed Management (DisMan) der IETF entwickelt wurde und deren Anweisungen nachfolgend beschrieben sind.

iquerySecName NAME
agentSecName NAME
Gibt den Standard SNMPv3-Benutzernamen an, der für interne Anfragen zum Suchen nach beliebigen notwendigen Informationen verwendet werden soll (entweder um den überwachten Ausdruck auszuwerten oder eine Inhalt einer Nachricht zusammenzustellen). Diese internen Anfragen nutzen immer SNMPv3, und zwar auch dann, wenn normale Anfragen des Agenten per SNMPv1 oder SNMPv2c erfolgen.

Bitte beachten Sie, dass dieser Benutzer ebenfalls explizit erzeugt (*createUser*) und mit passenden Rechten (z.B. *rouser*) ausgestattet werden muss. D.h. diese Anweisung dient nur zum Festlegen, welcher Benutzer verwendet werden soll, nicht zum eigentlichen Einrichten des Benutzers.

monitor [OPTIONS] NAME EXPRESSION

Definiert ein MIB-Objekt zum Überwachen. Falls die im Ausdruck EXPRESSION (siehe unten) definierte Bedingung zutrifft, dann initiiert dies das zugehörige Ereignis und es wird entweder eine Benachrichtigung verschickt oder ein SET-Auftrag gegeben (oder beides). Bitte beachten Sie, dass das Ereignis nur einmal angestoßen wird, wenn die Bedingung zum ersten Mal erfüllt ist. Das bedeutet, dass diese *monitor*-Anweisung erst dann wieder aktiv wird, nachdem die Bedingung einmal nicht mehr erfüllt ist und danach wieder erfüllt wird.
NAME ist ein zu Administrationszwecken verwendeter Name für diesen Ausdruck und dient zum Indizieren der Tabelle *mteTriggerTable* (sowie Tabellen, die sich auf diese beziehen). Bitte beachten Sie, dass solche *monitor*-Anweisungen einen internen SNMPv3-Auftrag anstoßen, um die überwachten Werte zu ermitteln (auch dann, wenn normale Anfragen des Agenten per SNMPv1 oder SNMPv2c erfolgen). Siehe Anweisung *iquerySecName* oben.

EXPRESSION

Die Event MIB unterstützt drei Typen von *monitor*-Ausdrücken: *existence*-Tests, *Boolean*-Tests und *threshold* Tests.

OID | ! OID | != OID

Definiert einen *existence(0)* Überwachungstest. Eine reine OID definiert einen *present(0)* Test, der dann aktiv wird, wenn (eine Instanz) der überwachten OID erzeugt wird. Ein Ausdruck der Form ! OID definiert einen *absent(1*) Test, der dann aktiv wird, wenn die überwachte OID gefunden wird. Ein Ausdruck der Form != OID definiert einen *changed(2)* Test, der dann aktiv wird, wenn sich der überwachte Wert ändert. Bitte beachten Sie, dass vor dem Parameter OID ein Leerzeichen stehen muss.

OID OP VALUE

Definiert einen *Boolean(1)* Überwachungstest. OP sollte einer der definierten Vergleichsoperatoren (!=, ==, <, <=, >, >=) sein, VALUE sollte ein ganzzahliger Wert sein, mit dem vergleichen wird. Bitte beachten Sie, dass vor und hinter dem Parameter OP Leerzeichen stehen müssen. Ein Vergleich wie z.B. OID !=0 wird nicht korrekt verarbeitet.

OID MIN MAX [DMIN DMAX]

Definiert einen *threshold(2)* Überwachungstest. MIN und MAX sind ganzzahlige Werte und definieren den unteren und oberen Schwellwert. Falls der Wert der überwachten OID unter den unteren Schwellwert fällt (MIN) oder den oberen Schwellwert übersteigt (MAX), dann stößt die *monitor*-Anweisung das zugehörige Ereignis an.

Bitte beachten Sie, dass das für das Überschreiten des Schwellwertes definierte Ereignis erst dann wieder reaktiviert wird, wenn der überwachte Wert unter den unteren Schwellwert (MIN) fällt. Analog dazu wird das Ereignis zum unteren Schwellwert erst wieder durch den oberen Schwellwert (MAX) reaktiviert.

Die optionalen Parameter DMIN und DMAX konfigurieren ein zwei ähnliche Schwellwert-Tests, benutzen aber die Differenzen zwischen zwei aufeinanderfolgenden Überwachungswerten.

OPTIONS

Das Verhalten des überwachten Ausdrucks kann mit verschiedenen Optionen gesteuert werden. Dazu gehören:

-D

Gibt an, dass der Ausdruck anhand der Differenzen zwischen Überwachungswerten (anstelle der Werte selbst) ausgewertet werden soll.

-d OID
-di OID

Gibt eine Diskontinuitätsmarkierung für die Bestätigung von Differenzwerten an. Eine *-di* Objektinstanz wird genau wie angegeben verwendet.
Für ein *-d* Objekt werden die Instanz-Sub-Ids aus dem entsprechenden (mit Wildcard versehenen) Ausdrucksobjekt angehängt. Wird die Option *-I* angegeben, dann gibt es keinen Unterschied zwischen den beiden Optionen.

Diese Option impliziert auch *-D*.

-e EVENT

Definiert das Ereignis, das initiiert wird, wenn diese *monitor*-Anweisung angestoßen wird. Wird die Option nicht angegeben, dann wird einer der in der DISMAN-EVENT-MIB definierten Standard-Benachrichtigungen erzeugt.

-I

Legt fest, dass der überwachte Ausdruck für die angegebene OID als einzelne Instanz angewendet wird. Standardmäßig wird die OID als Wildcard-Objekt behandelt und die Überwachung auf alle passenden Instanzen ausgedehnt.

-i OID
-o OID

Definiert zusätzliche varbinds, die zu den Nutzdaten der Benachrichtigung hinzugefügt werden, wenn die Überwachung anschlägt. Für Wildcard-Ausdrücke wird das Suffix der passenden Instanz an alle mit *-o* spezifizierten OIDs angehängt, während mit *-i* spezifizierte OIDs als exakte Instanzen behandelt werden. Wird die Option *-I* angegeben, dann gibt es keinen Unterschied zwischen den beiden Optionen.

Details zum Anordnen von Benachrichtigungs-Nutzdaten siehe *strictDisman*.

-r FREQUENCY
Überwacht den angegebenen Ausdruck alle FREQUENCY Sekunden. Standardmäßig wird der Ausdruck alle 600s (10 Minuten) überwacht.

-S
Gibt an, dass der *monitor*-Ausdruck beim ersten Start des Agenten nicht ausgewertet werden soll. Die erste Auswertung findet erst statt, nachdem das erste Wiederholungsintervall abgelaufen ist.

-s
Gibt an, dass der *monitor*-Ausdruck schon beim ersten Start des Agenten ausgewertet werden soll. Dies ist das Standardverhalten.


Benachrichtigungen, die bei der ersten Auswertung angestoßen werden, werden vor dem *coldStart* Trap gesendet.

-u SECNAME
gibt den Security-Benutzernamen an, der anstelle des Standard *iquerySecName* zum Scannen des lokalen Hosts verwendet wird. Auch hier gilt, dass dieser Benutzer explizit erzeugt und mit den passenden Rechten ausgestattet werden muss.

notificationEvent ENAME NOTIFICATION [-m] [-i OID | -o OID]*
Definiert ein Benachrichtigungs-Ereignis namens ENAME. Dieses kann durch eine *monitor*-Anweisung angestoßen werden, in welcher die Option *-e* ENAME angegeben wurde (siehe oben). NOTIFICATION sollte die OID der NOTIFICATION-TYPE Definition sein, für welche die Benachrichtigung erzeugt wird.

Wird die Option *-m* angegeben, dann enthalten die Nutzdaten der Benachrichtigung die Standard varbinds wie sie in der OBJECTS-Klausel der MIB Definition angegeben sind. Diese Option muss nach der NOTIFICATION OID kommen (und die betreffende MIB muss verfügbar und durch den Agent geladen sein). Andernfalls müssen diese varbinds explizit aufgelistet werden (entweder hier oder in der entsprechenden *monitor*-Anweisung).

Die Optionen *-i OID* und *-o OID* geben zusätzliche varbinds an, die an die Nutzdaten der Benachrichtigung angehängt werden sollen (nach der Standardliste). Wenn die *monitor*-Anweisung, die dieses Ereignis angestoßen hat, einen Wildcard-Ausdruck enthält, dann wird das Suffix der passenden Instanz zu allen mit *-o* spezifizierten OIDs hinzugefügt, während OIDs, die mit *-i* spezifiziert sind, als exakte Instanz behandelt werden. Wurde bei *monitor* die Option *-I* angegeben, dann gibt es keinen Unterschied zwischen diesen beiden Optionen.

setEvent ENAME [-I] OID = VALUE
Definiert das Setzen eines Ereignisses ENAME, indem der angegebenen OID der ganzzahlige Wert VALUE zugeordnet wird. Dieses Ereignis kann durch eine *monitor*-Anweisung ausgelöst werden, in der die Option *-e* ENAME (siehe oben) angegeben wurde.

Falls die *monitor*-Anweisung, die dieses Ereignis angestoßen hat, einen Wildcard-Ausdruck enthält, dann wird das Suffix der passenden Instanz normalerweise zur OID hinzugefügt. Wurde bei der *monitor*- oder *setEvent*-Anweisung die Option *-I* angegeben, dann wird die angegebene OID als exakte Instanz behandelt.

strictDisman yes

Die Definition der SNMP-Benachrichtigungen besagt, dass die in der OBJECT-Klausel definierten varbinds zuerst kommen sollten (in der angegebenen Reihenfolge), gefolgt von beliebigen "zusätzlichen" varbinds, die der Benachrichtigungs-Generator für nützlich hält. Der plausibelste Ansatz wäre:

– diese Pflicht-varbinds mit der *notificationEvent* Anweisung zu verbinden
– und dann die varbinds an das Ende der Liste anzuhängen, die mit derjenigen *monitor*-Anweisung verknüpft sind, die die Benachrichtigung ausgelöst hat.

Dies ist das Standardverhalten der Net-SNMP Event MIB Implementierung. Unglücklicherweise besagen die DisMan Event MIB Spezifikationen jedoch, dass die Auslöser-bezogenen varbinds zuerst kommen sollten, gefolgt von den Ereignis-bezogenen.

Diese Anweisung kann dazu verwendet werden, dieses grundsätzlich korrekte (aber unangebrachte) Verhalten wiederherzustellen.

Die *strictDisMan* Anordnung kann zur Folge haben, dass ungültige Nutzdaten der Benachrichtigung erzeugt werden, falls die Option *notificationEvent -n* zusammen mit den varbind-Optionen *monitor -o* (oder *-i*) angegeben wird.

Falls es keine *monitor*-Anweisungen mit varbinds für Nutzdaten gibt (weder *-i* noch *-o* in *monitor*), dann ist die Angabe dieser Anweisung irrelevant.

linkUpDownNotifications yes

konfiguriert die Event MIB Tabellen zum Überwachen von *ifTable* für Netzwerk-Schnittstellen, sodass beim Aktivieren oder Deaktivieren eine entsprechende *linkUp* oder *linkDown* Benachrichtigung ausgelöst wird.

3.4.3 DisMan Schedule MIB

Die DisMan Arbeitsgruppe entwarf auch einen Mechanismus für ein Scheduling bestimmter Aktionen (eine spezielle SET Zuweisung) zu bestimmten Zeiten.

Dies setzt voraus, dass der Agent so erstellt wurde, dass er das disman/schedule Modul unterstützt (welcher ein Bestandteil der Standard-Build-Konfiguration der neuesten Distribution ist).

Es gibt drei Möglichkeiten, eine "geschedulte" Aktion zu definieren:

repeat FREQUENCY OID = VALUE

konfiguriert eine SET Zuweisung von VALUE (ganzzahlig) zu einer MIB Instanz OID, die alle FREQUENCY Sekunden durchgeführt werden soll.

cron MINUTE HOUR DAY MONTH WEEKDAY OID = VALUE

konfiguriert eine SET Zuweisung von VALUE (ganzzahlig) zu einer MIB Instanz OID, die zu bestimmten Zeiten (in den Optionen MINUTE bis WEEKDAY spezifiziert) durchgeführt werden soll. Diese folgen demselben Muster wie das Äquivalent in den *crontab(5)* Optionen.

Diese Optionen sollten als per Komma getrennte Liste numerischer Werte angegeben werden. Benannte Werte für MONTH und WEEKDAY werden nicht unterstützt, genauso wenig wie Wertebereiche. Eine Wildcard kann als '*' angegeben werden.

Die Option DAY kann auch negative Werte annehmen, um anzuzeigen, dass vom Ende des Monats rückwärts gezählt wird.

at MINUTE HOUR DAY MONTH WEEKDAY OID = VALUE

konfiguriert einen einmalige SET Zuweisung, die durchgeführt werden soll, wenn die Zeit zum ersten Mal mit den Angaben in MINUTE bis WEEKDAY übereinstimmt. Die Interpretation dieser Option ist exakt dieselbe wie bei der Anweisung *cron*.

3.4.4 Beliebige Erweiterungskommandos

Der erste Erweiterungsmechanismus bestand darin, beliebige Kommandos oder Shell-Skripts ablaufen zu lassen. Solche Kommandos müssen sich keiner SNMP-Operationen bewusst oder konform zu einem bestimmte Verhalten sein; die MIB-Strukturen sind so konzipiert, dass sie sich auf jede Art der Kommandoausgabe anpassen lassen.

exec [MIBOID] NAME PROG ARGS
sh [MIBOID] NAME PROG ARGS

Ruft das Kommando bzw. das Shell-Skript PROG mit den Argumenten ARGS auf. Standardmäßig werden der Exit-Status und die erste Zeile der Kommandoausgabe via *extTable* gemeldet, alle weiteren Ausgaben werden verworfen.

die Einträge in dieser Tabelle erscheinen in der Reihenfolge wie sie aus der Konfigurationsdatei gelesen werden. D.h. dass das Hinzufügen einer neuen *exec* (or *sh*) Anweisung und Neustart des Agenten die Indizes der anderen Einträge beeinflussen kann.

Die PROG-Angabe für eine *exec*-Anweisung muss ein vollständiger Pfadname zu einem ausführbaren Programm sein, da es via *exec()* ausgeführt wird. Für Shell-Skripts wird *sh* statt *exec* verwendet.

Wird MIBOID angegeben, dann werden die Ergebnisse an diesem Punkt des OID-Baums aufgehängt, die Exit-Anweisung wird als MIBOID.100.0 zurückgegeben und die vollständige Kommandoausgabe in einer Pseudotabelle basierend auf MIBNUM.101, mit einer Reihe für jede Ausgabezeile.

Das Layout dieser "verschiebbaren" Form der *exec* (oder *sh*) Ausgabe entspricht nicht exakt dem Format einer gültigen MIB-Struktur. Der Mechanismus ist veraltet - bitte verwenden Sie stattdessen die unten beschriebene erweiterte Anweisung.

Der Agent puffert den Exit-Status oder die Ausgabe des ausgeführten Programm nicht.

exec und *sh* Erweiterungen können nur über die Datei *snmpd.conf* konfiguriert werden. Sie können nicht per SNMP SET Anforderungen eingerichtet werden.

extend [MIBOID] NAME PROG ARGS

funktioniert in ähnlicher Weise wie die *exec*-Anweisung, jedoch mit einer Reihe von Verbesserungen. Die MIB-Tabellen (*nsExtendConfigTable* etc) sind über die Option NAME indiziert und daher unabhängig von der Reihenfolge, in der die Einträge aus der Konfigurationsdatei gelesen werden.
Er gibt zwei Ergebnis-Tabellen, eine Tabelle (*nsExtendOutput1Table*) enthält den Exit-Status, die erste Zeile und (als ein einziger String) die vollständige Ausgabe von jedem *extend* Eintrag. Die andere Tabelle (*nsExtendOutput2Table*) enthält die vollständige Ausgabe als eine Folge von getrennten Zeilen.

Wenn MIBOID angegeben wird, dann werden die Konfigurations- und Ergebnis-Tabellen an diesem Punkt des OID-Baums aufgehängt, sind aber ansonsten auf genau dieselbe Arte und Weise strukturiert. D.h dass mehrere unterschiedliche *extend*-Anweisungen dieselbe MIBOID angeben können, ohne das es zu Konflikten kommt.

Exit-Status und Ausgabe werden für jede Anweisung einzeln gepuffert; außerdem können sie über die *nsCacheTable* gelöscht oder das Pufferverhalten konfiguriert werden.

Diese Funktion kann dynamisch durch SNMP SET Anforderungen an NET-SNMP-EXTEND-MIB konfiguriert werden.

3.4.5 Konfigurationsbeispiel

```
## Master behavior ##
# listen for all incoming IPv4 connections on UDP port 161
# listen for all incoming IPv6 connections on UDP port 163
agentAddress   udp:161,udp6::163
# enable AgentX master on agentXSocket
master        agentx
# it is better to use TCP/IP socket for agentx communication
# if notifications are expected.
agentXSocket   tcp:127.0.0.1:705
## Access control ##

# IPv4 connections #
# grant read-write perms on all OIDs from given IP with given community
rwcommunity snmpPriv 172.17.66.202

# grant read-only perms on system group (OID) only from all IPs with given
community
rocommunity snmpPublic default system

# IPv6 connections #

# grant read-write perms on all OIDs from all IPs with given community
rwcommunity6 snmpIPv6

# All connections #
# grant read-only perms on all OIDs from all IPs with given user
createUser  snmpdInternalUser MD5 "password"
rouser      snmpdInternalUser

## System Information ##
sysName     AU1.BS2
sysLocation Augsburg Limited
sysContact  admin@abg.com

## Active Monitoring ##

# send traps to the IP address with community name
trap2sink   tcp:172.17.66.202:162 publicTraps

# Event Services Configuration #
## don't forget to configure traps, as without them some part of event
services
## are pointless
# set up credentials
iquerySecName snmpdInternalUser
```

```
# This defines the traps to be sent (using notificationEvent), and explicitly
references# the relevant notification in the corresponding monitor entry :

notificationEvent  linkUpTrap    linkUp   ifIndex ifAdminStatus ifOperStatus
notificationEvent  linkDownTrap  linkDown ifIndex ifAdminStatus ifOperStatus

monitor  -r 60 -e linkUpTrap   "Generate linkUp"   ifOperStatus != 2
monitor  -r 60 -e linkDownTrap "Generate linkDown" ifOperStatus == 2


# Schedule Services Configuration #
# reload configuration once an hour, using:
repeat 3600 versionUpdateConfig.0 = 1

# reload configuration on the gicen our each day:
cron 10 0 * * * versionUpdateConfig.0 = 1
```

Wenn Sie einen anderen AgentX Socket als 27.0.0.1:705 oder */var/agentx/master* benutzen möchten, dann ist es besser, diesen nicht in *snmpd.conf*, sondern in *agentx.conf* (in */etc/snmp* erzeugt) zu konfigurieren, damit sich alle anderen Agenten von selbst korrekt konfigurieren. Auf *snmpd.conf* hat nur der Master Zugriff, während auf *agentx.conf* der Master und sämtliche Agenten zugreifen können.

3.4.6 Dämon rekonfigurieren

Um einen gerade laufenden Dämon zu rekonfigurieren (d.h. die Konfigurationsdatei neu einzulesen), kann eine der beiden folgenden Aktionen durchgeführt werden:

- Wenn Sie zuvor die Community/Benutzer mit Schreibrechten konfiguriert hatten, können Sie *versionUpdateConfig.0* mit Hilfe von *snmpset* auf *'1'* setzen.

  Beispiel:

  ```
  snmpset -v2c -c writeComm SNMP_ADDR versionUpdateConfig.0 i 1
  ```

- Sie können das rc-Skript */etc/rc2.d/S90net-snmp* unter einem privilegierten Benutzer aufrufen.

3.4.7 SNMP Trap-Dämon snmptrapd konfigurieren (snmptapd.conf)

Für alle eintreffenden Benachrichtigungen werden Zugangskontroll-Prüfungen durchgeführt. *snmptrapd.conf* bietet Unterstützung, indem Operationen des Dämons sowie die Art, wie eintreffende Traps verarbeitet werden sollen, konfiguriert werden können.

Wenn *snmptrapd* ohne geeignete Konfigurationsdatei (oder entsprechenden Zugangskontroll-Einstellungen) abläuft, dann werden solche Traps **nicht** verarbeitet. Details siehe Abschnitt „Zugangskontrolle“.

3.4.7.1 Verhalten von snmptrapd

snmpTrapdAddr [<transport-specifier>:]<transport-address>[,...]
Definiert eine Liste von "Listening"-Adressen, an denen SNMP-Benachrichtigungen empfangen werden sollen. Details zum Format von Listening-Adressen siehe Abschnitt „„Listening-Adressen“ in BS2000“.

Standard ist das Warten an UDP-Port 162 für alle IPv4-Adressen.

doNotRetainNotificationLogs yes
Deaktiviert die Unterstützung der NOTIFICATION-LOG-MIB. Normalerweise behält das Programm *snmptrapd* einen Eintrag des empfangenen Traps, der durch Suchen in den Tabellen *nlmLogTable* und *nlmLogvariableTable* ermittelt werden kann. Diese Anweisung dient dazu, dieses Verhalten zu unterdrücken.

doNotLogTraps yes
Deaktiviert das gesamte Logging der Benachrichtigungen. Dies kann angebracht sein, wenn das *snmptrapd* Programm nur *traphandle* "Hooks" verarbeiten und keine Traps an irgendeiner Stellen protokollieren soll.

doNotFork yes
Kein Verzweigen aus der aufrufenden Shell.

pidFile PATH
Gibt eine Datei an, in der die Prozess-Id des Empfängers der Benachrichtigung gespeichert werden soll. Standardmäßig wird diese Id nicht gesichert.

3.4.7.2 Zugangskontrolle

Es ist notwendig, explizit anzugeben, wer Traps senden und den Empfänger der Benachrichtigung informieren darf (und welche Typen von Aktionen diese anstoßen dürfen). Dabei wird eine Erweiterung des VACM-Modells benutzt, das im Haupt-SNMP-Agent verwendet wird.

Es können derzeit drei Aktionstypen angegeben werden:

- ‘log’ - Protokolliert die Details de Benachrichtigung - entweder in eine spezifizierte Datei, auf Standard-Ausgabe (oder *stderr*) oder via *syslog* (oder ähnlich).

- ‘execute’ - Übergibt die Details des Traps zur Verarbeitung an ein spezielles "Handler"-Programm.
- ‘net’ - Leitet den Trap an einen anderen Benachrichtigungs-Empfänger weiter.

In den folgenden Anweisungen ist TYPES eine per Kommas getrennte Liste von einem oder mehreren dieser Parameter. Am häufigsten wird dies typischerweise *log*,*execute*,*net* sein, um alle Aktionstypen für eine bestimmte Benachrichtigung-Kategorie abzudecken. Aber es durchaus möglich (und sogar wünschenswert), bestimmte Benachrichtigungsquellen auf die ausgewählten Aktionen zu beschränken.

authCommunity TYPES COMMUNITY [SOURCE [OID | -v VIEW]]
: Berechtigt Traps (und SNMPv2c INFORM Anforderungen) mit der angegebenen Community, die aufgelisteten Aktionstypen anzustoßen. Standardmäßig können damit alle Benachrichtigungen, die diese Community verwenden, verarbeitet werden. Mit dem Parameter SOURCE kann angegeben werden, dass die Konfiguration nur für Benachrichtigungen gilt, die aus bestimmten Quellen stammen.

authUser TYPES [-s MODEL] USER [LEVEL [OID | -v VIEW]]
: Berechtigt SNMPv3-Benachrichtigungen mit dem angegebenen Benutzer, die aufgelisteten Aktionstypen anzustoßen. Standardmäßig werden dadurch authentifizierte Anforderungen (*authNoPriv* oder *authPriv*) akzeptiert. Mit dem Parameter LEVEL können auch nicht-authentifizierte Benachrichtigungen (*noauth*) erlaubt oder Verschlüsselung (*priv*) verlangt werden, genau wie beim SNMP-Agenten.

Bei beiden Anweisungen kann die Konfiguration über den Parameter OID (oder *-v* VIEW) auf die Verarbeitung bestimmter Benachrichtigungen beschränkt werden.

authGroup TYPES [-s MODEL] GROUP [LEVEL [OID | -v VIEW]]
authAccess TYPES [-s MODEL] GROUP VIEW [LEVEL [CONTEXT]]
setAccess GROUP CONTEXT MODEL LEVEL PREFIX VIEW TYPES
: Berechtigt Benachrichtigungen in der angegebene GROUP (die mittels *group*-Anweisung konfiguriert ist), die aufgelisteten Aktionstypen anzustoßen.

createUser username (MD5|SHA) authpassphrase [DES|AES]
: Die Beschreibung, wie SNMPv3-Benutzer erzeugt werden, finden Sie in Abschnitt „SNMPv3 Authentifizierung“.

disableAuthorization yes
: Deaktiviert die oben angegebenen Zugangskontroll-Prüfungen und kehrt zum früheren Verhalten zurück, d.h. alle eintreffenden Benachrichtigungen werden akzeptiert .

3.4.7.3 Verarbeiten von Benachrichtigungen

Zur speziellen Verarbeitung können Benachrichtigungen an einen anderen Benachrichtigungs-Empfänger oder an ein externes Programm weitergeleitet werden.

traphandle OID|default PROGRAM [ARGS ...]

Ruft das mit PROGRAM spezifizierte Programm mit den angegebene Argumenten (ARGS) auf, sobald eine Benachrichtigung eintrifft, die zu OID passt. Für SNMPv2c und SNMPv3 Benachrichtigungen wird dieser Parameter mit dem Wert *snmpTrapOID* aus der Benachrichtigung verglichen. Für SNMPv1 Traps werden die generischen und die speziellen Trap-Werte sowie die Unternehmens OID in die äquivalente OID gemäß RCF2576 konvertiert.

Typischerweise ist der Parameter OID der Name (oder die numerische OID) eines NOTIFICATION-TYPE Objekts, und das angegebene Programm wird für Benachrichtigungen aufgerufen, die genau mit dieser OID übereinstimmen. Dieser Parameter unterstützt jedoch auch eine einfache Form Wildcard-Suffixen. Durch Anhängen der Zeichens wird eine innerhalb des Teilbaums basierte Benachrichtigung an der angegebenen OID aufgehängt.

Zum Beispiel würde ein OID-Parameter der Form 1.3.6.1.4.1* zu jeder Unternehmens-spezifischen Benachrichtigung passen (einschließlich der OID selber). Ein OID-Parameter der Form 1.3.6.1.4.1.* würde fast genauso funktionieren, würde aber nicht mit dieser exakten OID übereinstimmen, d.h. nur Benachrichtigungen zutreffen die streng unterhalb diese Punktes liegen.

Bitte beachten Sie, dass diese Syntax keine vollständig regulären Ausdrücke oder Wildcards unterstützt, d.h ein OID-Parameter der Form oid..*.subid ist ungültig.

Falls für den Parameter OID *default* angegeben wird, dann wird das Programm für jede Benachrichtigung aufgerufen, die nicht zu irgendeiner einer anderen (OID-spezifischen) *traphandle*-Anweisung passt.

Die Detailinformationen der Benachrichtigung werden dem Programm über dessen Standard-Eingabe übergeben. Bitte beachten Sie, dass das Programm immer das Benachrichtigungsformat im SNMPv2-Stil verwendet, wobei SNMPv1-Traps vor Übergabe an das Programm gemäß RFC2576 konvertiert werden. Das Eingabe-Format sieht immer wie folgt aus, mit einem Eintrag pro Zeile:

HOSTNAME

Name des Hosts, der die Benachrichtigung gesendet hat.

IPADDRESS

IP-Adresse des Hosts, der die Benachrichtigung gesendet hat.

VARBINDS
Eine Liste von so genannten "varbinds" (variable bindings), die den Inhalt der Benachrichtigung beschreiben (eine varbind pro Zeile). Der erste Parameter jeder Zeile (bis zu einem Leerzeichen) ist die OID der varbind, der Rest der Zeile ist deren Wert. Das Format der beiden Parameter wird durch die Anweisung *outputOption* (oder ähnliche Konfiguration) gesteuert.

Die erste OID sollte immer *SNMPv2-MIB::sysUpTime.0* sein, die zweite *SNMPv2-MIB::snmpTrapOID.0*. Die restlichen Zeilen enthalten die varbind-Liste mit den Nutzdaten. Für SNMPv1traps ist die letzte *SNMPv2-MIB::snmpTrapEnterprise.0*.

forward OID|default DESTINATION
Leitet Benachrichtigungen, die zur angegebenen OID passen, an einen anderen Empfänger weiter, der an DESTINATION wartet. OID (und *default*) werden genauso interpretiert wie bei *traphandle*-Anweisung.

3.4.7.4 Logging

format1 FORMAT
format2 FORMAT
Geben das Format an, das zum Anzeigen von SNMPv1 TRAPs bzw. SNMPv2-Benachrichtigungen verwendet wird. Bitte beachten Sie, dass SNMPv2c und SNMPv3 beide dasselbe SNMPv2 PDU Format benutzen.

Die verfügbaren Layout-Zeichen sind im Abschnitt „Formatierungs-Spezifikationen“ beschrieben.

ignoreAuthFailure yes
weist den Empfänger an, *authenticationFailure* Traps zu ignorieren.

Diese betrifft aktuell nur das Logging solcher Benachrichtigungen. *authenticationFailure* Traps werden weiterhin an Trap-Bearbeitungs-Skripts übergeben und an andere Benachrichtigungs-Empfänger weitergeleitet. Auf dieses Verhalten sollte man sich nicht verlassen, da es wahrscheinlich ist, dass es sich in Zukunft ändert.

logOption string
Gibt an, wo Benachrichtigungen protokolliert werden sollen: Standard-Ausgabe, Standard-Error (*stderr*), eine spezielle Datei oder via *syslog*. Details siehe Abschnitt „Ausgabe-Konfiguration“.

outputOption string
Definiert verschiedene Eigenschaften für die Anzeige von OIDs und anderen Werten. Details siehe Abschnitt „Ausgabe-Konfiguration“.

3.4.7.5 Formatierungs-Spezifikationen

snmptrapd interpretiert Format-Strings ähnlich wie *printf()*. Es versteht folgende Formatierungs-Sequenzen:

String	Bescheibung
%%	Ein Literal %
%a	Der Inhalt des Feldes *agent-addr* der PDU (nur v1 Traps)
%A	Der Hostname, der zum Inhalt des Feldes *agent-addr* der PDU gehört (falls verfügbar), andernfalls der Inhalt des Feldes *agent-addr* der PDU (nur v1 Traps).
%b	PDU Quell-Adresse (Hinweis: Dies muss nicht unbedingt eine IPv4-Adresse sein)
%B	PDU Quell-Hostname (falls verfügbar), andernfalls PDU Quell-Adresse (siehe Hinweis oben)
%h	Aktuelle Stunde auf dem lokalen System
%H	Feld *hour* aus der *sysUpTime.0* varbind
%j	Aktuelle Minute auf dem lokalen System
%J	Feld *minute* aus der *sysUpTime.0* varbind
%k	Aktuelle Sekunde auf dem lokalen System
%K	Feld *seconds* aus der *sysUpTime.0* varbind
%l	Aktueller Tag im Monat auf dem lokalen System
%L	Feld *day of month* aus der *sysUpTime.0* varbind
%m	Aktueller (numerischer) Monat auf dem lokalen System
%M	Numerische Feld *month* field aus der *sysUpTime.0* varbind
%N	Enterprise-String
%q	Trap Subtyp (numerisch, in Dezimaldarstellung)
%P	Security-Information aus der PDU (*community name* für v1/v2c, *user* und *context* für v3)
%t	Dezimale Anzahl von Sekunden seit der Betriebssystem-Epoche
%T	Der Wert der *sysUpTime.0* varbind in Sekunden
%v	Liste von varbinds aus den Nutzdatenliste der Benachrichtigung. Diese sind durch ein Tabulatorzeichen getrennt, oder durch ein Komma + Leerzeichen, falls die alternative Form verlangt wird.
%V	Gibt den varbind-Trenner an. Dies erfordert eine Folge von Zeichen bis zum nächsten %. Zum Einbetten eines %-Zeichens muss \% verwendet werden.
%w	Trap-Typ (numerisch, in Dezimaldarstellung)
%W	Trap-Beschreibung
%y	Aktuelles Jahr auf dem lokalen System
%Y	Feld *year* aus der *sysUpTime.0* varbind

Zusätzlich zu diesen Werten können auch optionale Angaben *width* und *precision* verwendet werden (genau wie in *printf(3)*), sowie ein Kennzeichen. Folgende Kennzeichen werden unterstützt:

- '-' (links ausrichten)
- '0' (führende Nullen verwenden)
- '#' (alternatives Format verwenden)

Das Kennzeichen "altenatives Format verwenden" ändert das Verhalten von verschiedenen Format-String-Folgen:

- Die Zeitinformation wird auf Basis von GMT ausgegeben (anstatt der lokalen Zeitzone)
- Die varbinds-Liste wird mit Kommas getrennt (statt mit Tabulatorzeichen)
- Die Systemverfügbarkeit (system uptime) wird in ein verständliches Format umgewandelt (statt einer einfachen, ganzen Zahl)

Beispiel

Um eine Meldung wie z.B.`14:03 TRAP3.1 from humpty.ucd.edu` zu erhalten, können Sie Folgendes eingeben:

```
snmptrapd -P -F "%02.2h:%02.2j TRAP%w.%q from %A\n"
```

Wenn Sie dasselbe in GMT statt lokaler Zeit haben möchten, dann geben Sie ein:

```
snmptrapd -P -F "%#02.2h:%#02.2j TRAP%w.%q from %A\n"
```

3.5 Konfiguration von SNMP-AGENTS

3.5.1 Konfiguration des Application Monitor Agenten

Der Application Monitor Agent gestattet die Überwachung von:

– Benutzer-Anwendungen
– DCAM-Anwendungen
– BCAM-Anwendungen
– Subsystemen
– Jobvariablen
– Protokolldateien

Außerdem können Gruppen zusammengehöriger Anweisungen als Einheit (Objekt) verwaltet werden.
Art und Umfang der Anwendungsüberwachung werden über eine Konfigurationsdatei im BS2000-Dateisystem individuell gesteuert. Der Name der Konfigurationsdatei wird dem Application Monitor Agenten im Startkommando bekannt gegeben. Bei Syntaxfehlern in der Konfigurationsdatei wird der Startvorgang abgebrochen. Wenn keine Konfigurationsdatei angegeben wird, ist die Überwachung auf Subsysteme beschränkt.

Die folgende Tabelle beschreibt die agentenspezifischen Kommando-Optionen:

Option	Standard	Beschreibung
-c <BS2:config-fle>	keine Datei; es werden nur Subsysteme überwacht	Wertet die angegebene Konfigurationsdatei aus, um die ausgewählten Objekte zu überwachen.
-t <sec>	5 Sekunden	Basiszähler, der festlegt, wie oft Objekte geprüft werden: – Prüfung von Subsystemen (5 * Zähler) sek (Standard, alle 25 sek) – Prüfung von Dateien (1 * Zähler) sek. (Standard, alle 5 sek) – Prüfung von DCAM-Anwendungen (60 * Zähler) sek, (Standard, alle 5 min)

3.5.1.1 Anweisungen für die Konfigurationsdatei

Die Konfigurationsdatei enthält Informationen darüber, welche Anwendungen, Tasks, Subsysteme, Jobvariablen und Protokolldateien überwacht werden sollen. Es können jeweils bis zu 256 Benutzer-, BCAM-Anwendungen, Jobvariablen und Protokolldateien sowie 128 DCAM-Anwendungen überwacht werden. Benutzer- und BCAM-Anwendungen sowie Tasks, die überwacht werden sollen, müssen mit Jobvariablen angestartet werden. Die Anzahl der zu überwachenden Subsysteme ist unbegrenzt.

Die Einträge in der Konfigurationsdatei werden über SDF-Anweisungen erzeugt. Mit der Anweisung //REMARK können Kommentare in der Konfigurationsdatei hinterlegt werden. Die letzte Anweisung der Datei muss immer die Anweisung //END sein. Anweisungen, die hinter der END-Anweisung stehen, werden ignoriert.

Überwachung	**Anweisung**
Anwendung	ADD-APPLICATION-RECORD
DCAM-Anwendung	ADD-DCAM-APPLICATION-RECORD
Subsystem	ADD-SUBSYSTEM-RECORD
Protokolldatei	ADD-LOG-FILE-RECORD
Jobvariable	ADD-JV-RECORD
Gruppe von zusammengehörigen Anwendungen	DEFINE-OBJECT
Überwachungsintervalle	SET-TIMER-OPTIONS

ADD-APPLICATION-RECORD

Die Anweisung ADD-APPLICATION-RECORD benennt die BCAM- und Benutzeranwendungen, die überwacht werden sollen. Unter Anwendungen sind Programme oder Tasks zu verstehen.

//ADD-APPLICATION-RECORD
APPLICATION-NAME = <composed-name_1 .. 54_with-underscore>
, **VERSION** = ***NONE** / <product-version>
, **TYPE** = ***BCAM** / ***USER**
, **JV-NAME** = <filename_1 .. 54>
, **TRAP-CONDITION** = **A** / list-poss (6) : <name_1 .. 1>
, **WEIGHT= 0 /** <integer 0 .. 999>

APPLICATION-NAME=<composed-name_1..54_with-underscore>
bestimmt die Anwendung, die der Agent überwachen soll.

VERSION=*NONE / <product-version>
Versionsnummer der Anwendung
Standardwert: *NONE

TYPE=*BCAM / *USER
Typ der Anwendung.

JV-NAME = <filename_1 .. 54>
Jobvariable (MONJV), mit der die Anwendung bzw. die Task überwacht wird.

TRAP-CONDITION=A / list-poss (6) : <name_1 .. 1>
Zustände, bei denen ein Trap erzeugt werden soll.

WEIGHT= 0 / <integer 0 .. 999>
Gewichtung der für den Application Monitor Agenten spezifischen Traps. Beim Senden eines Traps versorgt der Application Monitor Agent beim generischen Trap das Trap-Objekt *appMonWeight* und die Trap-Nummer mit dem angegebenen Wert (siehe Abschnitt „Verarbeiten von Benachrichtigungen“). Sollen in einer Anwendung für verschiedene Ereignisse verschiedene Gewichte verwendet werden, dann muss die zugehörige ADD-APPLICATION-RECORD-Anweisung mehrmals in der Konfigurationsdatei angegeben werden.
Standardwert: 0

ADD-DCAM-APPLICATION-RECORD

Die Anweisung ADD-DCAM-APPLICATION-RECORD benennt die DCAM-Anwendungen, die zyklisch überwacht werden sollen. Das Überwachungsintervall für DCAM-Anwendungen liegt beim 60-fachen Wert der Timer-Einstellung, beträgt also standardmäßig 5 Minuten. Mit der Anweisung //SET-TIMER-OPTIONS können Sie als Überwachungsintervall ein beliebiges Vielfaches der Timer-Einstellung festlegen.

Maximal können 128 DCAM-Anwendungen überwacht werden.

//ADD-DCAM-APPLICATION-RECORD
APPLICATION-NAME = <name_1 .. 8> , **HOST**= ***OWN** / <name_1 .. 8>> , **KEEP-CONNECTION** = ***YES** / ***NO** , **MSG**= ***NONE** / <c-string> / <x-string> , **TRAP-CONDITION** = list-poss (2) : ***NOT-AVAILABLE** / ***AVAILABLE** , **WEIGHT= 0 / <integer 0 .. 999>**

APPLICATION-NAME=<name_1..8>
bestimmt die DCAM-Anwendung, die der Agent überwachen soll.

HOST=*OWN / <name_1..8>
Rechner auf dem die DCAM- Anwendung läuft
Standardwert: *OWN

KEEP-CONNECTION=*YES / *NO
Angabe, ob die Verbindung wieder abgebaut werden soll
Standardwert: *YES

MSG= *NONE / <c-string> / <x-string>
Verbindungsnachricht
Standardwert: *NONE

TRAP-CONDITION=*NOT-AVAILABLE / *AVAILABLE
Zustände, bei denen ein Trap erzeugt wird.
Standardwert: *NOT-AVAILABLE

WEIGHT= 0 / <integer 0 .. 999>
Gewichtung der für den Application Monitor Agenten spezifischen Traps. Beim Senden eines Traps versorgt der Application Monitor Agent beim generischen Trap das Trap-Objekt *appMonWeight* und die Trap-Nummer mit dem angegebenen Wert (siehe Abschnitt „Verarbeiten von Benachrichtigungen“).
Standardwert: 0

ADD-SUBSYSTEM-RECORD

Die Anweisung ADD-SUBSYSTEM-RECORD definiert die zu überwachenden Subsysteme. Das Überwachungsintervall liegt bei dem fünffachen Wert der Timer-Einstellung, beträgt standardmäßig also 25 Sekunden.

Mit der Anweisung //SET-TIMER-OPTIONS können Sie als Überwachungsintervall ein beliebiges Vielfaches der Timer-Einstellung festlegen.

//ADD-SUBSYSTEM-RECORD
NAME = <structured-name 1 .. 8> / ***ALL** , **VERSION** = ***NONE** / <product-version> , **TRAP-CONDITION** = ***NONE** / list-poss (8) : ***CREATED** / ***NOT-CREATED** / ***IN-DELETE** / ***IN-CREATE** / ***IN-RESUME** / ***IN-HOLD** / ***NOT-RESUMED** / ***LOCKED** , **WEIGHT= 0 / <integer 0 .. 999>**

NAME=<structured-name 1..8> / *ALL
bestimmt das Subsystem, das der Agent überwachen soll.

VERSION=*NONE / <product-version>
Versionsnummer des Subsystems
Standardwert: *NONE

TRAP-CONDITION=*NONE / list-poss (8) : *CREATED / *NOT-CREATED / *IN-DELETE / *IN-CREATE / *IN-RESUME / *IN-HOLD / *NOT-RESUMED / *LOCKED
Zustände, bei denen ein Trap erzeugt werden soll.
Standardwert: *NONE

ACHTUNG!
Bei der Angabe NAME=*ALL sollten Sie TRAP-CONDITION=*NONE verwenden, da andernfalls Performance-Probleme auftreten können.

WEIGHT= 0 / <integer 0 .. 999>
Gewichtung der für den Application Monitor Agenten spezifischen Traps. Beim Senden eines Traps versorgt der Application Monitor Agent beim generischen Trap das Trap-Objekt *appMonWeight* und die Trap-Nummer mit dem angegebenen Wert (siehe Abschnitt „Verarbeiten von Benachrichtigungen“). Sollen in einem Subsystem für verschiedene Ereignisse verschiedene Gewichte verwendet werden, dann muss die zugehörige ADD-SUBSYSTEM-RECORD-Anweisung mehrmals in der Konfigurationsdatei angegeben werden.
Standardwert: 0

ADD-LOG-FILE-RECORD

Die Anweisung ADD-LOG-FILE-RECORD definiert die zu überwachenden Protokolldateien. Standardmäßig sendet der Application Monitor Agent bei jeder Änderung einer Protokolldatei einen Trap. Es ist jedoch möglich, die Traps bzw. Einträge zu filtern. Mit der Anweisung //SET-TIMER-OPTIONS können Sie als Überwachungsintervall ein beliebiges Vielfaches der Timer-Einstellung festlegen.

//ADD-LOG-FILE-RECORD
NAME = <filename_1 .. 54> / <posix-pathname>
, APPLICATION-NAME = ***NONE** / <composed-name_1 .. 54_with-underscore>
, **MONITORING** = ***YES** / ***NO**
, **FORMAT** = ***EBCDIC** / ***ASCII**
, **PATTERN** = ***NONE** / list-poss (8) : <c-string_1 .. 256_with-lower-case>
, **WEIGHT= 0 / <integer 0 .. 999>**

NAME=<filename_1 .. 54> / <posix-pathname>
bestimmt die Protokolldatei, die der Agent überwachen soll.

APPLICATION-NAME=*NONE / <composed-name_1 .. 54_with-underscore>
Name der Anwendung.
Standardwert: *NONE

MONITORING=*YES / *NO
Angabe, ob die Protokolldatei überwacht werden soll.

FORMAT=*EBCDIC / *ASCII
Format der Protokolldatei.
Standardwert: *EBCDIC

PATTERN = *NONE / list-poss (8) : <c-string_1 .. 256_with-lower-case>
Angabe eines oder mehrerer Suchmuster. Fehlt die Angabe PATTERN werden alle Einträge in eine Protokolldatei per Trap gemeldet.
Es sind folgende Wildcard-Angaben zulässig:

? : ersetzt ein beliebiges Zeichen
* : ersetzt eine beliebig lange Folge von Zeichen
[s] : ersetzt genau ein Zeichen aus der Zeichenkette s
[c1 - c2]: ersetzt ein beliebiges Zeichen aus dem Bereich von c1 bis c2
Das Zeichen "\" (Backslash) muss zur Entwertung der Sonderzeichen angegeben werden.
Es wird zwischen Groß- und Kleinschreibung unterschieden.
Standardwert: *NONE

WEIGHT= 0 / <integer 0 .. 999>
Gewichtung der für den Application Monitor Agenten spezifischen Traps. Beim Senden eines Traps versorgt der Application Monitor Agent beim generischen Trap das Trap-Objekt *appMonWeight* und die Trap-Nummer mit dem angegebenen Wert (siehe Abschnitt „Verarbeiten von Benachrichtigungen“).
Standardwert: 0

ADD-JV-RECORD

Die Anweisung ADD-JV-RECORD definiert die zu überwachenden Jobvariablen. Standardmäßig sendet der Application Monitor Agent jede Änderung einer Jobvariablen als Trap. Es ist jedoch möglich, die Traps zu filtern.

//ADD-JV-RECORD
JV-NAME = <filename_1 .. 54>
, **APPLICATION-NAME** = ***NONE** / <composed-name_1 .. 54_with-underscore>
, **PASSWORD** = ***NONE** / <c-string_1 .. 4 > / < x-string_1 .. 8>
, **PATTERN** = ***NONE** / list-poss (8) : <c-string_1 .. 256_with-lower-case>
, **WEIGHT= 0 / <integer 0 .. 999>**

JV-NAME = <filename_1 .. 54>
bestimmt die Jobvariable, die der Agent überwachen soll.

APPLICATION-NAME = *NONE / <composed-name_1 .. 54_with-underscore>
Name der Anwendung.
Standardwert: *NONE

PASSWORD = *NONE / <c-string_1 .. 4 > / <x-string_1 .. 8>
Lesepasswort der Jobvariablen.
Standardwert: *NONE

PATTERN = *NONE / list-poss (8) : <c-string_1 .. 256_with-lower-case>
Angabe eines oder mehrerer Suchmuster. Fehlt die Angabe PATTERN werden alle JV-Änderungen per Trap gemeldet.
Es sind folgende Wildcard-Angaben zulässig:

? : ersetzt ein beliebiges Zeichen
* : ersetzt eine beliebig lange Folge von Zeichen
[s] : ersetzt genau ein Zeichen aus der Zeichenkette s
[c1 - c2]: ersetzt ein beliebiges Zeichen aus dem Bereich von c1 bis c2
Das Zeichen "\" (Backslash) muss zur Entwertung der Sonderzeichen angegeben werden.
Es wird zwischen Groß- und Kleinschreibung unterschieden.
Standardwert: *NONE

WEIGHT= 0 / <integer 0 .. 999>
Gewichtung der für den Application Monitor Agenten spezifischen Traps. Beim Senden eines Traps versorgt der Application Monitor Agent beim generischen Trap das Trap-Objekt *appMonWeight* und die Trap-Nummer mit dem angegebenen Wert (siehe Abschnitt „Verarbeiten von Benachrichtigungen“).
Standardwert: 0

DEFINE-OBJECT

Logisch zusammengehörige Bestandteile eines Prozesses (Anwendungen, Protokolldateien, Subsysteme und Jobvariablen) können mit der Anweisung DEFINE-OBJECT in einer Gruppe (Objekt) zusammengefasst werden. Alle in der DEFINE-OBJECT-Anweisung genannten Elemente müssen mit den entsprechenden ADD...-Anweisungen ebenfalls in der Konfigurationsdatei definiert werden.

```
//DEFINE-OBJECT

OBJECT-NAME = <composed-name_1 .. 8_with-underscore>
, BCAM-APPLICATION = *NONE / list-poss(5): <composed_name_1 .. 54_with-underscore>
, USER-APPLICATION = *NONE / list-poss(5): <composed_name_1 .. 54_with-underscore>
, DCAM-APPLICATION = *NONE / list-poss(5): <name_1 .. 8>
, LOG-FILE = *NONE / list-poss(5): <filename_1 .. 54> / <posix-pathname>
, SUBSYSTEM = *NONE / list-poss(5): <structured-name_1 .. 8>
, JV = *NONE / list-poss(10): <filename_1 .. 54>
, MONITORING-TIME = *ALWAYS / *INTERVAL (...)
    *INTERVAL ( ...)
        , START-TIME = hh:mm
        , STOP-TIME = hh:mm
        , EXCEPT-DAYS = *NONE / list-poss(6): MON / TUE / WED / THU / FRI / SAT / SUN
```

OBJECT-NAME = <composed-name_1 .. 8_with-underscore>
Name des Objekts.

BCAM-APPLICATION = *NONE / list-poss(5): <composed_name_1 .. 54_with-underscore>
BCAM-Anwendungen, die zu diesem Objekt gehören.
Standardwert: *NONE

USER-APPLICATION = *NONE / list-poss(5): <composed_name_1 .. 54_with-underscore>
Benutzeranwendungen, die zu diesem Objekt gehören.
Standardwert: *NONE

DCAM-APPLICATION = *NONE / list-poss(5): <name_1 .. 8>
DCAM-Anwendungen, die zu diesem Objekt gehören.
Standardwert: *NONE

LOG-FILE = *NONE / list-poss(5): <filename_1 .. 54> / <posix-pathname>
Protokolldateien, die zu diesem Objekt gehören.
Standardwert: NONE

SUBSYSTEM = *NONE / list-poss(5): <structured-name_1 .. 8>
Subsysteme, die zu diesem Objekt gehören.
Standardwert: *NONE

JV = *NONE / list-poss(10): <filename_1 .. 54>
Job-Variablen, die zu diesem Objekt gehören.

MONITORING-TIME = *ALWAYS / *INTERVAL (...)
spezifiziert den Zeitraum der Überwachung.
Standardwert: *ALWAYS

***INTERVAL (...)**
spezifiziert das Überwachungsintervall. Wenn STOP-TIME größer als START-TIME ist, zählen bei der Überprüfung der EXCEPT-DAYS die Stunden nach Mitternacht zum vorherigen Tag.

Beispiel:

Die Überwachungszeit erstreckt sich von 20:00 bis 3.00 außer Samstag und Sonntag. Die Überwachung endet daher am Samstag um 3:00 morgens und beginnt wieder am Montag um 20:00 abends.

START-TIME = HH:MM
Zeitpunkt, ab dem das Objekt überwacht werden soll.

STOP-TIME = HH:MM
Zeitpunkt, bis zu dem das Objekt überwacht werden soll.

EXCEPT-DAYS = *NONE / list-poss(6): MON / TUE / WED / THU / FRI / SAT / SUN
Wochentage, an denen das Objekt nicht überwacht werden soll.
Standardwert: *NONE

Beispiel: Überwachung eines MAREN-Systems

Ein MAREN-System besteht u.a. aus folgenden Komponenten:

– Subsystem MAREN
– Steuerprogramm MARENCP
– automatische Freibandzuweisung MARENUCP

Darüber hinaus wird jede von der automatischen Freibandzuweisung reservierte VSN in der Jobvariablen TAPE.FILE.MAREN hinterlegt.

Folgende Definition eines Objekts „MAREN“ fasst diese Komponenten zusammen:

```
//DEFINE-OBJECT OBJECT-NAME = MAREN
//, USER-APPLICATION = (MARENCP, MARENUCP)
//, SUBSYSTEM = MAREN
//, JV = TAPE.FILE.MAREN
```

SET-TIMER-OPTIONS

Der Application Monitor Agent verwendet einen Timer. Den Wert für den Timer spezifizieren Sie beim Starten des Application Monitor Agenten (Option *-t*). Die Anweisung SET-TIMER-OPTIONS legt das Überwachungsintervall (Polling-Faktor) fest. Das Überwachungsintervall bestimmt, nach wievielen Timer-Abläufen die nächste Überprüfung durchgeführt werden soll.

//SET-TIMER-OPTIONS
FILES = **1** / <integer> , **SUBSYSTEMS** = **5** / <integer> , **DCAM-APPLICATIONS** = **60**/ <integer>

FILES = 1 / <integer>
legt den Polling-Faktor für Dateien fest.
Standardwert: 1

SUBSYSTEMS = 5 / <integer>
legt den Polling-Faktor für Subsysteme fest.
Standardwert: 5

DCAM-APPLICATIONS = 60 / <integer>
legt den Polling-Faktor für DCAM-Anwendungen fest.
Standardwert: 60

3.5.1.2 Wechsel der Konfigurationsdatei im laufenden Betrieb

Änderungen der aktuellen Konfigurationsdatei im laufenden Betrieb können dem Application Monitor durch Setzen des Objekts *appMonConfFile0* vorgenommen werden.

Beispiel

```
snmpset -v2c -cpublic SNMP_ADDR appMonConfFile.0 s "APPMON.CONF"
```

Bei Syntaxfehlern in *appMonConfFile* wird mit der ursprünglichen Konfiguration weitergearbeitet.

3.5.2 Konfiguration des Console Monitor Agenten

Über UCON erhält der Console Monitor den Zugriff auf die Konsolkommandos von $CONSOLE. Folgende Vorbereitungen sind notwendig, um dem Console Monitor den Zugriff auf die BS2000-Konsole zu ermöglichen:

– Operator-Kennung <operator-id> einrichten

– Zugangsberechtigung für die Operatorkennung freischalten

Operator-Kennung <operator-id> einrichten

```
/ADD-USER USER-ID=<operator-id>,                              -
          PROTECTION-ATTRIBUTE=*PAR(LOGON-PASSWORD=<pass>),   -
          ACCOUNT-ATTRIBUTES=*PAR(ACCOUNT=<account-nr>)
```

Die hier festgelegten LOGON-Attribute müssen in der rc-Datei beim Starten des Console Monitor Agenten angegeben werden (siehe Console Monitor Agent).

Zugangsberechtigung für die Operatorkennung freischalten

Für den Betrieb mit SECOS muss zusätzlich noch die Zugangsberechtigung für die Operator-Kennung zu $CONSOLE freigeschaltet werden:

```
/MOD-LOGON-PROTECTION USER-IDENTIFICATION=<operator-id>,     -
                      OPERATOR-ACCESS-PROG=*YES(PASSWORD-CHECK=*YES)
```

Der Klasse-2-Systemparameter NBBAPRIV muss auf den Standardwert N eingestellt sein.

Die folgenden Tabellen beschreiben die agentenspezifischen Kommando-Optionen:

Pflicht-Optionen	**Beschreibung**
-o <operid>	Gibt die Operator-Id an
-y <op-role1> [,<op-role2>, ..., <op-role10>]	Gibt die Rolle(n) des Operators an; bis zu 10
oder	
-a <auth-file>	Lädt die in der Datei angegebenen Berechtigungsdaten

Optionale Optionen	**Beschreibung**
-k <password>	Passwort für den Zugriff auf $CONSOLE
-c <msg-filter>	Lädt Meldungsfilter
-n <negative-msg-filter>	Lädt negativen Meldungsfilter
-A <auth-file>	Liefert ein Prompt zum Erzeugen einer Authentifizierungsdatei. Diese Datei bleibt mit AES verschlüsselt.

Optionale Optionen	Beschreibung
-s <auth-file>	Gibt die Operator-Ids und die Rollen aus, die in der Authentifizierungsdatei definiert sind. Falls ein Passwort definiert wurde, dann wird *SET ausgegeben, andernfalls *NONE.

Definition von Meldungsfiltern

Bei der Auswahl von Meldungen verwendet der Console Monitor Agent zwei Filtervarianten:

– positiver Meldungsfilter
wählt Meldungen aus, die an die Management-Station geschickt werden sollen.

– negativer Meldungsfilter
wählt Meldungen aus, die nicht an die Management-Station geschickt werden dürfen.

3.5.2.1 Positiver Meldungsfilter

Für die Auswahl der Meldungen, die an die Management-Station geschickt werden, stehen zwei Filtermöglichkeiten zur Verfügung:

– Routingcode (ist jeder Konsolmeldung zugeordnet)
– Meldungsschlüssel (identifiziert jede Meldung eindeutig)

Auswahlkriterium Routingcode

Jede Meldung ist einem bestimmten Routingcode zugeordnet. Operator-Rollen enthalten die Routingcodes derjenigen Meldungen, die an die Management-Station geschickt werden sollen. Die Operator-Rollen werden beim Starten des Console Monitor angegeben (siehe Abschnitt „Agenten-spezifische Optionen zum manuell Starten der Agenten“). Die folgenden Anweisungen zeigen, wie Operator-Rollen erzeugt und der Operator-Kennung zugeordnet werden. Voraussetzung für das Absetzen der folgenden Anweisungen ist das Privileg
SECURITY-ADMINISTRATION, das standardmäßig die Benutzerkennung SYSPRIV hat.

Erzeugen der Operator-Rolle:

```
/CREATE-OPERATOR-ROLE OP-ROLE=<op-role-name>,                -
                      ROUTING-CODES=.......
```

Zuordnung der Operator-Rollen zur Operator-Kennung:

```
/MODIFY-OPERATOR-ATTR USER-ID=<operator-id>,                 -
                      ADD-OPERATOR-ROLE=(<op-role-name1>,...,<op-role-namex>)
```

Bei Einsatz von SECOS muss außerdem der Operator-Kennung das Privileg OPERATING zugewiesen werden:

```
/SET-PRIVILEGE PRIV=OPERATING,USER-ID=<operator-id>
```

Auswahlkriterium Meldungsschlüssel

Die Meldungsschlüssel derjenigen Meldungen, die der Management-Station zugestellt werden sollen, werden in der positiven Meldungsfilter-Datei hinterlegt. Drei Filtermöglichkeiten stehen mit den folgenden Anweisungen zur Verfügung:

– *msgid*,
– *QUESTION*
– *TYPIO*

Der Name der Meldungsfilter-Datei wird dem Console Monitor bei dessen Start über díe Kommando-Option *-c* mitgeteilt. Im laufenden Betrieb kann der Dateiname in dem MIB-Objekt *consMonMsgFilter* eingetragen werden.

Fehlt die Angabe einer Meldungsfilter-Datei beim Start des Console Monitor, werden alle Meldungen ausgegeben, deren Routingcode in der Operator-Rolle angegeben ist.

Enthält die Meldungsfilter-Datei keine bzw. keine gültigen Meldungsschlüssel, dann werden der Management-Station keine Traps zugestellt.

Für die Meldungsfilter-Datei gelten folgende Namenskonventionen:

/BS2/<datei>	BS2000-Datei
[:<catid>:]$<userid>.<datei>	BS2000-Datei
*POSIX(<datei>)	POSIX-Datei
/<pfad>/<datei>	POSIX-Datei
<datei>	in diesem Fall ist ausschlaggebend, in welcher Umgebung der Agent gestartet wurde.

3.5.2.2 Aufbau des positiven Meldungsfilters

msgid-Anweisung

```
<msgid [wgt] [SOURCE=src] [DEVICE=dev]
       [PATTERN=/pat1[/..patx]] [ACKNOWLEDGE=YES]>
```

msgid

Angabe eines Meldungsschlüssels.
Bei der Angabe von Meldungsschlüsseln sind folgende Wildcard-Angaben zulässig:

- ? : ersetzt ein beliebiges Zeichen
- * : ersetzt eine beliebig lange Folge von Zeichen
- [s] : ersetzt genau ein Zeichen aus der Zeichenkette s
- [c1 - c2]: ersetzt ein beliebiges Zeichen aus dem Bereich von c1 bis c2

Das Zeichen "\" (Backslash) muss zur Entwertung der Sonderzeichen angegeben werden. Es wird zwischen Groß- und Kleinschreibung unterschieden.

wgt

Angabe eines Meldungsgewichts (weight). Den Meldungsschlüsseln kann ein Gewicht zugeordnet werden. Dieses Gewicht wird im Trap-String der eigentlichen Meldung vorangestellt. Damit hat der Anwender die Möglichkeit, die Wichtigkeit der Meldungen selbst einzustellen und entsprechend an der Management-Station darzustellen. Fehlt die Gewichtsangabe, erhält der Meldungsschlüssel standardmäßig den Wert 0.

Die Angabe wird als Integer mit dem Wertebereich 0 - 999 erwartet.

src

Angabe eines Quellennamens (source). Im Trap-String wird Quelle mit BS2-<source> versorgt. Fehlt diese Angabe, wird der Standardwert *BS2Console* eingesetzt. Mit dieser Angabe können Sie einen Alarm gezielt auf ein Objekt im Netzbild lenken. Die Angabe erfolgt alphanumerisch in der Länge 1 - 12.

pat

Angabe eines oder mehrerer Suchmuster (pattern).

- ? : ersetzt ein beliebiges Zeichen
- * : ersetzt eine beliebig lange Folge von Zeichen
- [s] : ersetzt genau ein Zeichen aus der Zeichenkette s
- [c1 - c2]: ersetzt ein beliebiges Zeichen aus dem Bereich von c1 bis c2

Das Zeichen "\" (Backslash) muss zur Entwertung der Sonderzeichen angegeben werden. Es wird zwischen Groß- und Kleinschreibung unterschieden.

dev

Ist ein DEVICE angegeben, sendet der Console Monitor Agent diesen Trap mit der DEVICE-Angabe als Community.

ACKNOWLEDGE=YES
Durch die Angabe ACKNOWLEDGE=YES wird dem Agenten angezeigt, dass dieser Trap bestätigt werden muss.

QUESTION-Anweisung

Question filtert alle Meldungen heraus, die eine Frage beinhalten, d.h. die eine Antwort erwarten.Tritt eine Frage auf, überprüft der Console Monitor zuerst, ob ein Muster der QUESTION-Einträge passt. Ist das nicht der Fall, werden dem Meldungstyp entsprechend die MSGID-Einträge oder die TYPIO-Einträge durchsucht.

```
<QUESTION [wgt] [SOURCE=src] [DEVICE=dev] [PATTERN=/pat1[/..patx]]
[ACKNOWLEDGE=YES]>
```

QUESTION Meldungsschlüssel einer Konsolanfrage

wgt
siehe oben

src
siehe oben

dev
siehe oben

pat
siehe oben

ACKNOWLEDGE=YES
siehe oben

Beispiel:

`<QUESTION PATTERN=[0-9]*>`	Auswahl aller Fragen, die mit einer Ziffer beginnen.

TYPIO-Anweisung

Eine Sonderstellung nehmen so genannte TYPE I/O-Meldungen ein. Zu den TYPE I/O-Meldungen zählen beispielsweise Nachrichten, die mit /SEND-MSG der BS2000-Konsole zugestellt werden. Die Weiterleitung der TYPE I/O-Meldungen als SNMP-Trap wird ebenfalls über die Meldungsfilter-Datei gesteuert. Der Eintrag für eine TYPE I/O-Meldung ist folgendermaßen aufgebaut:

```
<TYPIO [wgt] [SOURCE=src] [DEVICE=dev] [PATTERN=/pat1[/..patx]] [ACKNOWLEDGE=YES]>
```

wgt
siehe oben

src
siehe oben

dev
siehe oben

pat
siehe oben

ACKNOWLEDGE=YES
siehe oben

Beispiel:

`<TYPIO PATTERN=/*abc*/xyz>`
`<TYPIO PATTERN=/Hallo*>`
`<TYPIO PATTERN=/?\?*>`
	Alle TYPE I/O-Meldungen, die den String "abc" enthalten, nur aus "xyz" bestehen, mit "Hallo" beginnen oder an zweiter Stelle ein Fragezeichen haben, werden als Trap der Management-Station zugestellt.

3.5.2.3 Negativer Meldungsfilter

Für den Console Monitor Agenten wird auch ein negativer Meldungsfilter angeboten. Die Meldungsschlüssel derjenigen Konsolmeldungen, die nicht an die Management-Station durchgereicht werden sollen, werden in der negativen Meldungsfilter-Datei hinterlegt. Fragen können nicht unterdrückt werden. Das MIB-Objekt *consMonNegMsgFilter* verweist auf den Namen der negativen Meldungsfilter-Datei. Der Name der negativen Meldungsfilter-Datei wird beim Start des Console Monitor mit der Option *-n* definiert. Diese Definition kann nur beim Start des Console Monitor, aber nicht im laufenden Betrieb geändert werden.

Die Länge des Eintrags darf maximal 179 Zeichen betragen.

```
<msgid> [PATTERN=/pat1[/..patx]]> [<msgid [PATTERN=/pat1[/..patx]]>] ...
```

Zur Beschreibung von *msdid* und *pat* siehe „Aufbau des positiven Meldungsfilters“.

3.5.2.4 Ändern der Meldungsfilter-Datei im laufenden Betrieb

Änderungen der aktuellen Meldungsfilter-Datei im laufenden Betrieb können mit der Console Monitor-Anwendung durch Setzen des Objekts *consMonMsgFilter* vorgenommen werden.

Bei Syntaxfehlern in der Meldungsfilter-Datei *consMonMsgFilter* wird mit der ursprünglichen Meldungsfilter-Datei weitergearbeitet.

Beispiel: Filtern von Konsolmeldungen

Die Meldung EXC0858 soll nur an die Management-Plattform geschickt werden, wenn sie weder den String "CLAQ" noch den String "TEST" enthält. Der Trap soll mit der Trapnummer 99 geschickt werden und als Quelle soll "Hardware" eingetragen sein.

Sie erreichen dies wie folgt:

1. Tragen Sie im positiven Meldungsfilter ein: `<EXC0858 99 SOURCE=Hardware>`
2. Tragen Sie in der negativen Meldungsfilterdatei ein:

   ```
   <EXC0858 PATTERN = *CLAQ* / *TEST*>
   ```

3.5.3 Konfiguration des Storage Agenten

Mit dem Storage Management Agenten können Platten und Pubsets überwacht werden. Dazu übergeben Sie beim Starten die Konfigurationsdatei via Kommando (wird in der rc-Datei konfiguriert).

Die Konfigurationsdatei kann während des Betriebs geändert werden, indem das Objekt *storMgmtGlobalDataInputFile.0* gesetzt wird. Falls die Konfigurationsdatei Syntaxfehler enthält, dann läuft der Betrieb mit dem Original-Meldungsfilter weiter.

Die Konfiguration erfolgt über eine Input-Datei:

– Für die Überwachung des Saturation-Levels einzelner Public Volumes Sets (Pubsets) müssen die betreffenden PVS in der Input-Datei des Agenten spezifiziert werden. Dies erfolgt mit der ADD-PUBSET-RECORD-Anweisung.

– Um den Zustand der ausgewählten ROBAR-Anwendung zu überwachen, verwenden Sie die Anweisung ADD-ROBAR-RECORD.

– Für die Überwachung des Reconfiguration State einzelner Platten müssen die betreffenden Platten in der Input-Datei des Agenten spezifiziert werden. Dies erfolgt mit der ADD-DISK-RECORD-Anweisung.

– Mit der Anweisung //REMARK können Kommentare in der Konfigurationsdatei hinterlegt werden.

– Die letzte Anweisung in der Konfigurationsdatei sollte die Anweisung //END sein. Alle Anweisungen, die auf die //END-Anweisung folgen, werden ignoriert.

– Maximal können 128 Pubsets und/oder Platten überwacht werden.

ADD-PUBSET-RECORD - Hinzufügen eines zu überwachenden Pubsets

//ADD-PUBSET-RECORD
PUBSET= <cat_id 1..4>
, CHECK=**SATURATION-LEVEL**
, TRAP-COMMUNITY= ***STORAGE** / ***PUBSET-NAME** / <c-string 1..64>

PUBSET=<cat_id 1..4>
CAT-ID des Pubsets, das überwacht werden soll.

CHECK=SATURATION-LEVEL
Objekt, das überwacht werden soll; derzeit ist nur die Angabe SATURATION-LEVEL möglich (Standardwert).

TRAP-COMMUNITY=*STORAGE / *PUBSET-NAME / <c-string 1..64>
Community-String, mit dem der Trap verschickt wird.
Bei Angabe von *PUBSET wird die <cat-id> als Community-Name verwendet.
Bei Angabe von <c-string 1..64> wird dieser String als Community-Name verschickt.
Standardwert: *STORAGE

ADD-DISK-RECORD - Hinzufügen einer zu überwachenden Platte

//ADD-DISK-RECORD
DISK-MN =<alphanum-name 1..4> **, CHECK=RECONFIGURATION-STATE** **, TRAP-COMMUNITY**= ***STORAGE** / ***DISK-MN** / <c-string 1..64>

DISK-MN=<alphanum-name 1 ..4>
mnemotechnischer Name des Geräts, das überwacht werden soll.

CHECK=RECONFIGURATION-STATE
Objekt, das überwacht werden soll; derzeit ist nur die Angabe RECONFIGURATION-STATE möglich (Standardwert).

TRAP-COMMUNITY=*STORAGE / *DISK-MN / <c-string 1..64>
Community-String, mit dem der Trap verschickt wird.
Bei Angabe von *DISK-MN wird der bei DISK-MN angegebene Name als Community-Name verwendet.
Bei Angabe von <c-string 1..64> wird dieser String als Community-Name verschickt.
Standardwert: *STORAGE

ADD-ROBAR-RECORD - Hinzufügen einer zu überwachenden ROBAR-Anwendung

//ADD-ROBAR-RECORD
LOCATION =< composed-name_1..8_with-underscore> **, VERSION=*NONE** / <product-version mandatory-man-corr> / <product-version without-man-corr> **, JV-NAME**= <filename_1..54> **, TRAP-CONDITION** = **A** / list-poss (6) : <name_1 .. 1>

LOCATION=< composed-name_1..8_with-underscore>
Lagerort der ROBAR-Anwendung.

VERSION=*NONE / <product-version mandatory-man-corr> / <product-version without-man-corr>
Version der zu überwachenden Anwendung. Bei Angabe einer Versionsnummer muss das hier angegebene Format mit dem bei der Definition des Subsystems benutzten Format übereinstimmen (Freigabe- und Korrekturstand müssen angegeben werden oder dürfen nicht angegeben werden).

JV-NAME= <filename_1..54>
Job-Variable (MONJV), mit der die Anwendung überwacht wird.

TRAP-CONDITION = A / list-poss (6) : <name_1 .. 1>
Zustände der MONJV, bei denen ein Trap erzeugt werden soll. Der Wert A bedeutet alle Zustände.

3.5.4 Konfiguration des openUTM Agenten

Der folgende Abschnitt beschreibt die Tätigkeiten, die zur Inbetriebnahme des openUTM Agenten notwendig sind.

3.5.4.1 Einsatzvorbereitung

Die Kommunikation zwischen dem Agenten und einer UTM-Anwendung erfolgt über UPIC(BS2000). UPIC benötigt für die Kopplung zwischen dem Agenten und der UTM-Anwendung eine Side-Information-Datei (*upicfile*). Diese Datei muss *upicfile* heißen und im BS2000 katalogisiert sein. Der Eintrag in der *upicfile* besteht in der Regel aus vier Teilen:

– einem Kennzeichen, in diesem Fall HD als Kennzeichen für eine Kopplung zwischen UPIC(BS2000) und UTM(BS2000),

– dem Kommunikationspartner der Anwendung, der für die aktuell ausgewählte UTM-Anwendung auf den Wert SNMP4UTM gesetzt werden sollte,

– dem Partnernamen, dargestellt durch den Haupt-BCAM-Namen der Anwendung. Es wird empfohlen, auch den Hostnamen des Rechners anzugeben, vom Partnernamen durch einen "." (Punkt) getrennt.

– dem Transaktionscode; diese Angabe ist in diesem Fall nicht erforderlich, da der Agent den Transaktionscode mit dem *Set_TP_Name* Aufruf angibt.

Die Datei *upicfile* ist unter BS2000 eine editierbare Datei. Da es im BS2000 kein <newline>-Zeichen gibt, wird das Zeilenende-Zeichen durch das Semikolon (";") dargestellt, siehe Beispiel. D.h. falls in einer editierten Zeile ein Semikolon steht, reagiert UPIC so, als ob die Zeile dort abgeschlossen wäre und interpretiert den Rest der Zeile als neue Zeile (bis zum nächsten ";"-Zeichen), das gilt auch für Kommentarzeilen.

Beispiel

Side Information Datei

```
*symbolic destination names for (BS2000) application ZENTRBS2;
;*application is running on BSENGINE;
HDSNMP4UTM ZENTRBS2.BSENGINE;
```

Der UTM Agent meldet sich mit dem lokalen Namen SNMPUPIC beim UPIC-Kommunikationssystem an. Der Name SNMPUPIC sollte mit den KDCDEF-Anweisungen PTERM bzw. TPOOL als Kommunikationspartner der Anwendung definiert werden.

Da der openUTM Agent zum Absetzen von UTM-Administrationskommandos die entsprechende Berechtigung benötigt, muss eine mit `STATUS=ADMIN` oder `PERMIT=ADMIN` definierte UTM-Benutzerkennung angegeben und über Job-Variablen an den openUTM Agenten übergeben werden, siehe Abschnitt „Ablaufumgebung“. Falls keine Job-Variablen definiert sind, dann werden die Standardberechtigungen von SNMP-AGENTS verwendet.

openUTM verwendet folgende TACs zur Überwachung mit SNMP:

- KDCWADMI zum Lesen von Informationen
- KDCLPAP, KDCLTAC, KDCSHUT, KDCPOOL, KDCSWITCH, KDCPTERM, KDCTCL, KDCLTAC, KDCUSER zum Setzen oder Modifizieren von Objekt-Parametern.

Bitte stellen Sie sicher, dass diese TACs in der UTM-Anwendung mit den nötigen Administrationsrechten generiert sind.

3.5.4.2 Konfiguration des openUTM Agenten zur Überwachung mehrerer UTM-Anwendungen

Für die Überwachung mehrerer UTM-Anwendungen wird eine Konfigurationsdatei benötigt, in der jede überwachte UTM-Anwendung spezifiziert werden muss.

Die Konfigurationsdatei kann beim Starten via Kommando-Option (in der rc-Datei konfiguriert) übergeben werden:

```
-c <BS2:config-file>
```

Die Konfigurationsdatei kann während des Betriebs geändert werden, indem das Objekt *utmGlobalConfFile.0 object* gesetzt wird. Falls die Konfigurationsdatei Syntaxfehler enthält, dann läuft der Betrieb mit dem Original-Meldungsfilter weiter.

Die Einträge in der Konfigurationsdatei werden mit der SDF-Anweisung //ADD-APPLICATION-RECORD erzeugt. Mit der Anweisung //REMARK können Kommentare in der Konfigurationsdatei hinterlegt werden. Die Datei muss mit der Anweisung //END abgeschlossen werden.

ADD-APPLICATION-RECORD

Die Anweisung //ADD-APPLICATION-RECORD benennt die UTM-Anwendungen, die überwacht werden sollen.

//ADD-APPLICATION-RECORD
APPLICATION-NAME = <structured-name 1..8>
, **FILEBASE**= ***SAME** / <full-filename_1..54>
, **USER-ID**= **TSOS** / <name_1 .. 8>
, **TRAP-CONDITION** = list-poss (3): ***ABNORMAL-TERMINATED / *NORMAL-TERMINATED / *RUNNING**

APPLICATION-NAME=<structured-name 1.. 8>
bestimmt die UTM-Anwendung, die der Agent überwachen soll.

FILEBASE= *SAME / <full-filename_1..54>
Basisname der A-Teile der KDCFILE.
*SAME ist Standard.

USER-ID=TSOS / <name 1 .. 8>
Kennung, unter der die UTM-Anwendung gestartet wird.
TSOS ist Standard.

TRAP-CONDITION= list-poss (3): *ABNORMAL-TERMINATED / *NORMAL-TERMINATED / *RUNNING
Zustände, bei denen ein Trap erzeugt werden soll.
*ABNORMAL-TERMINATED ist Standard.

3.5.4.3 Ablaufumgebung

Im BS2000-System wird das UPIC-Programm über Jobvariablen gesteuert.
Dazu wertet UPIC folgende Jobvariablen aus:

Jobvariable	Linkname	Bedeutung
UPICPATH	*UPICPAT	Die Jobvariable UPICPATH bestimmt das Dateiverzeichnis, unter dem die Side Information Datei abgespeichert ist. Wenn die Jobvariable nicht gesetzt ist, wird die Datei unter dem aktuellen Dateiverzeichnis gesucht. Beim Start des Agenten unter POSIX muss die Jobvariable UPICPATH mit dem Wert „BS2/$<userid>“ versorgt werden, da UPIC andernfalls versucht, die *upicfile* im POSIX-Dateisystem zu öffnen.
UPICFILE	*UPICFIL	legt den rechten Teil des Namen der Side Information Datei fest. Ist die Variable nicht gesetzt, wird der Dateiname upicfile gesetzt.Der vollständige Dateiname setzt sich zusammen aus UPICPAT.UPICFIL.Sind weder UPICPAT noch UPICFIL gesetzt, so lautet er „$progid.UPICFILE“.
UPIC4SNMP.USER	---	UTM-Benutzerkennung, die der openUTM Agent verwendet, um Administrationskommandos abzusetzen. Wenn die Jobvariable nicht gesetzt ist, wird USER=SNMPADM verwendet.
UPIC4SNMP.PASS	---	Passwort für die UTM-Benutzerkennung. Wenn die Jobvariable nicht gesetzt ist, wird Passwort=SNMPUPIC verwendet.
UPICTRACE	*UPICTRA	Die Jobvariable UPITRACE steuert die Trace-Erzeugung (siehe nächste Seite „Diagnoseunterlagen).
UPICLOG	*UPICLOG	Die Jobvariable UPICLOG legt den Namen der Logging-Datei fest. Fehlt diese Angabe, so lautet der Name „##.USR.TMP.UPICL<tsn>“.

Beachten Sie, dass die Zuweisung nach LOGOFF verloren geht.

3.5.4.4 Diagnoseunterlagen

Neben der Trace-Datei des openUTM Agenten gibt es weitere Dateien, die im Fehlerfall hilfreich sein können:

– UPIC-Trace-Datei
– UPIC-Logging-Datei
– SYSLOG-Datei

UPIC-Trace-Datei

Beim Trägersystem UPIC ist es möglich, Trace-Information für sämtliche Schnittstellenaufrufe zu generieren. Diesen Vorgang steuern Sie durch das Setzen der Jobvariable *UPICTRACE*. Beim Aufruf *Enable_UTM_UPIC* wird der Inhalt der Jobvariable ausgewertet. Falls die Jobvariable gesetzt ist, werden beim Aufruf jeder Funktion die Parameter und die Benutzerdaten bis zu einer Länge von 128 Bytes prozess-spezifisch in einer Datei protokolliert.

Einschalten des UPIC-Trace

Der UPIC-Trace wird wie folgt eingeschaltet:

```
/SET-JV-LINK LINK-NAME=*UPICTRA,JV-NAME=UPICTRACE
/MODIFY-JV UPICTRACE,VALUE='-S[X] [-R wrap] [-Dprefix]'
```

Bedeutung:

-S	Ausführliche Protokollierung der Aufrufe, der zugehörigen Argumente und der Benutzerdaten in der maximalen Länge von 128 Bytes (Pflichtangabe)
-SX	Es werden zusätzlich interne Informationen an der Schnittstelle zum Transportsystem protokolliert.
-R *wrap*	Mithilfe der durch *wrap* spezifizierten Dezimalzahl wird die maximale Größe der temporären Trace-Datei bestimmt. Defaultwert: 128.
-D*prefix*	Die Trace-Dateien werden unter folgenden Namen angelegt: – *prefix*.UPICT<tsn> – *prefix*.UPICU<tsn> Wenn *prefix* nicht angegeben ist, wird „##.USR.TMP“ als Präfix verwendet.

Ausschalten des UPIC-Trace

Der UPIC-Trace wird mit einem der beiden folgenden Kommandos ausgeschaltet:

```
/DELETE-JV UPICTRACE
/MODIFY-JV UPICTRACE,VALUE=''
```

UPIC-Logging-Datei

Falls die UTM-Anwendung eine Conversation abnormal beendet, wird in die UPIC-Logging-Datei eine UTM-Fehlermeldung geschrieben. Die UPIC-Logging-Datei wird nur zum Schreiben der Fehlermeldung geöffnet (Modus *append*) und anschließend wieder geschlossen.

SYSLOG-Datei

Beim Start einer Anwendung legt openUTM eine anwendungsspezifische Protokolldatei SYSLOG an. In dieser Datei werden Ereignisse, die während des Ablaufs der Anwendung eintreten, in Form von UTM-Meldungen protokolliert.

3.5.5 Konfiguration des openSM2 Agenten

Dieser Abschnitt beschreibt die notwendigen Schritte, um den openSM2 Agenten funktionsfähig zum machen. Ohne diese Schritte ist der openSM2 Agent nutzlos und gibt für die meisten Objekte den Status *not-data(-1)* zurück.

Voraussetzungen:

- Das SM2-Subsystem ist eingerichtet.
- Die Ermitteln der SM2-Messungen ist konfiguriert und aktiviert. Dies können Sie durch folgende Anweisungen im BS2000 erreichen.

```
/EXEC $SM2
*call-admin-part
%//set-periodic-task-parameter log-tasks=*none
%//start-measurement-program per
%//start-measurement-program utm
%//call-eval-part
*end
```

Falls die Performance-Überwachung von openUTM zu erwarten ist, dann muss für die betreffenden UTM-Anwendungen die openSM2-Überwachung aktiviert werden, z.B. durch das UTM-Administrationskommando KDCAPPL SM2=ON.

3.5.6 Konfiguration des HSMS Agenten

Um den Agenten funktionsfähig zu machen, muss der System-Name der SYS-LIB.HSMS übergeben werden. Dies können Sie über folgende Kommando-Option erreichen:

```
'-l <SYSLIB.HSMS>'
```

Beispiel

```
hsmsAgent -l \$SYSHSMS.SYSLIB.HSMS.110 &
```

3.5.7 Konfiguration von TCP-IP-AP

Dieser Abschnitt beschreibt die Voraussetzungen, die zum Starten des FTP-Dämons via *ftpAgent* notwendig sind.

Als erstes muss die Start-Prozedur SYSENT.TCP-IP-AP.053.FTPD vorhanden sein. Diese Prozedur enthält Startparameter für den FTP-Dämon.

Ein Neustart des FTP-Dämons über *snmpset* mit angegebenem Port und FTAC-Level ist nur dann möglich, wenn die beiden folgenden Optionen in SYSDAT.TCP-IP-AP.053.FTPD.OPT gesetzt sind:

```
FTAClevel | -B

childName | -C
```

Danach können Sie den Dämon mit Standard *FTACLevel (0)* wie folgt starten:

```
snmpset [OPTIONS] ftpServerPort.portNumber i portNumber
```

wobei *portNumber* der neue Server-Port für die Steuerung der FTP-Clients ist.

Weitere Informationen finden Sie im aktuellen Systemverwalter-Handbuch zu "interNet Services".

4 Betrieb

Die Liefereinheit NET-SNMP enthält den SNMP-Dämon (*snmpd*), den SNMP-Trap-Dämon, die SNMP-Client-Tools und die Funktionalität von Event und Scheduling Services.

Mit SNMP-AGENTS wird ein Set von Agenten für System- und Anwendungs-Management-Aufgaben ausgeliefert.

Zusätzlich stellen die Produkte BCAM, SESAM/SQL und TCP-IP-AP produkt-spezifische Agenten zur Verfügung, die die Funktionalität von SNMP-AGENTS ergänzen.

In diesem Kapitel werden die In- und Außerbetriebnahme der einzelnen Komponenten in BS2000 sowie die Kommandos zum Abrufen von Informationen beschrieben. Der letzte Abschnitt informiert über das Verhalten im Fehlerfall.

Wichtig!

– Keine der Agenten ist standardmäßig "damonisiert", weshalb empfohlen wird, sie im Hintergrund zu starten (Anfügen eines "&" an Schluss der Anweisung).

– Es wird **nicht** empfohlen, die Agenten manuell zu starten. Verwenden Sie stattdessen rc-Skripts zum Starten/Stoppen.

4.1 rc-Skripts

Für alle betroffenen SNMP-Produkte werden rc-Skripts installiert, die ein automatisches Starten der Agenten beim Hochfahren von POSIX bzw. ein automatisches Stoppen beim Beenden von POSIX erlauben. Standardmäßig werden alle Start-rc-Skripte in auskommentierter Form ausgeliefert. Um das automatische Starten von NET-SNMP oder einer seiner Agenten zu aktivieren, müssen Sie die zugehörigen rc-Dateien modifizieren und an Ihre Konfiguration anpassen.

Beispiel (SNMP-Dämon)

Für den automatischen Start von *snmpd* muss im Start-Script */etc/rc2.d/S90net-snmp* der Kommentar in folgender Zeile entfernt werden:

```
#/opt/net-snmp/snmpd $_OPTIONS &
```

Beispiel für den Console Agent

```
From original rc file:
    _OPT="-LSO-6d -p /var/run/consoleAgent.pid"
# IMPORTANT! Don't forget to add mandatory options:
# -o  Operator's ID; -y Operator's Role(-s); -k password;
# -a authentification file's path (use this option for auto-start)
# To create authentification file run agent with -A and follow the prompt.
# uncomment following line to start subagent consoleAgent
#    /opt/snmp-agents/consoleAgent $_OPT  &
```

Zum automatischen Starten des Console Monitor Agenten müssen Sie Folgendes tun:

– Ändern Sie die Variable _OPT, indem Sie Pflciht-OPtionen/Cedentials hinufügen, z.B.:

  ```
  _OPT="-LSO-6d -p /var/run/consoleAgent.pid -o tsos -y sysadm -k 12345678"
  ```

– Entfernen Sie den Kommentar in der Zeile, die den Agenten aufruft.

Hinweis

Klartext-Passwörter sind nicht sicher. Der Console Agent unterstützt die Verwendung von Authentifizierungs-Dateien, die mit AES verschlüsselt sind. Eine solche Datei können Sie wie folgt erstellen:

– Rufen Sie `"consoleAgent -A"` auf
– Folgen Sie dem Prompt.
– Editieren Sie die rc-Datei, indem Sie im Argument *-a* den absoluten Pfad der erzeugten *auth-file* angeben.

4.2 NET-SNMP-Dämonen und SNMP-Tools

4.2.1 SNMP-Dämon snmpd

Vor dem ersten Start des *snmpd* muss im BS2000 die Datei */etc/snmp/snmpd.conf* an die eigene Konfiguration angepasst werden (siehe Abschnitt „System Gruppe“).

Um den SNMP-Dämon zu starten, muss die BS2000-Kennung die POSIX-UserID 0 (SYSROOT) besitzen.

NET-SNMP umfasst die Funktionalität von MIB-II, Event und Scheduling Services. Sobald *snmpd* gestartet ist, kann diese Funktionalität genutzt werden.

Konfigurationsbeispiele für snmpd

Beispiele

1. `snmpd –LS0–4d –Lf /var/adm/snmpd.log –p /var/run/snmpd.pid –a -q`

   -LS0-4d
   Logging in die syslog mit maximalem Logging-Level LOG_WARINING.

   -Lf /var/adm/snmpd.log
   Protokolliert in die angegebene Logging-Datei.

   -p /var/run/snmpd.pid
   Sichert die PID des Dämons in Datei.

   -a
   Protokolliert die Adressen der eintreffenden Anfragen.

   -q
   Gibt die Informationen in einem einfach zu parsenden Format aus.

   Dieser *snmpd* liest die Standard *snmpd.conf* aus dem Verzeichnis */etc/snmp*.

2. `snmpd -C -c /etc/snmp/example.conf`

   -C
   Die Standard-Konfigurationsdatei soll nicht verwendet werden.

   -c
   Die angegebene Konfigurationsdatei wird geladen.

3. `snmpd -C --rwcommunity=public --master=agentx -x tcp:705 tcp:1111`

   -C
   Die Standard-Konfigurationsdatei soll nicht verwendet werden.

--rwcommunity=public
: Setzt die read\write community auf "public".

--master=agentx
: Aktiviert das AgentX Protokoll.

-x tcp:705
: Wartet auf der angegebenen Adresse auf AgentX-Verbindungen.

tcp:1111
: Wartet auf der angegebenen Adresse auf SNMP-Anfragen.

4.2.2 SNMP Trap-Dämon snmptrapd

Standardmäßig erwartet *snmptrapd* die Anfragen auf dem UDP-Port 162 (für alle IPv4-Adressen). Diese Einstellung können Sie ändern, die zugehörige Konfigurationsdatei (Standard: *snmptrapd.conf*) modifizieren oder in der Kommandozeile entsprechende Argumente übergeben. Weitere Informationen fínden Sie in Abschnitt „SNMP Trap-Dämon snmptrapd konfigurieren (snmptapd.conf)“.

Bitte beachten Sie, dass *snmptrapd* vor *snmpd* gestartet werden sollte.

Konfigurationsbeispiele für snmptrapd

1. `snmptrapd -C -c ./trap.conf -Lo &`

   -C
   : Die Standard-Konfigurationsdatei soll nicht verwendet werden.

   -c
   : Die angegebene Konfigurationsdatei wird geladen.

   -Lo
   : Logging nach stdout.

2. `snmptrapd -C -Lo --authcommunity="log public" udp:1162 &`

   --authcommunity="log public"
   : Alle Traps mit trapcommunity "public" protokollieren.

   udp:1162
   : Wartet an dem angegebenen Socket auf Trap-Benachrichtigungen von allen IPs.

4.2.3 SNMP-Tools snmpwalk, snmpget und snmpset

snmpwalk

snmpwalk ermöglicht es, Teilbäume zu durchsuchen.

Details zu diesem Thema siehe *snmpwalk* Beschreibung in der offiziellen Net-SNMP Open Source Seite.

Beispiel

Folgendes Kommando fragt alle Variablen unter *system* ab:

```
snmpwalk –Os –c public –v 1 zeus system
```

-Os
: (Parser-Option) Nur das letzte symbolische Element des OID ausgeben (OID = Objekt-Identifier).

-c public
: public ist der Community name.

-v 1
: Es wird die Version 1 des SNMP-Protokolls wird verwendet (community based).

zeus
: Name des Zielsystems.

system
: OID-Name des Objekts im Netz.

snmpget und snmpset

snmpget ermittelt Informationen über ein Objekt im Netz.

snmpset setzt Werte für ein Objekt im Netz.

Beispiele

1. `snmpget –c public zeus system.sysDescr.0`

   Gibt den Wert des OID *sysDescr.0* zurück (sofern verfügbar).

2. `snmpset –c private –v 1 test–hub system.sysContact.0 s dpz@noc.rutgers.edu`

   Setzt den OID *sysContact.0* auf den String *dpz@noc.rutgers.edu*.

   test-hub
   : Name des Zielsystems.

   system.sysContact.0

OID des Objekts, für das der Wert gesetzt werden soll.

s
Typ der Variable, die gesetzt werden soll (hier: string).

dpz@noc.rutgers.edu
Neuer Wert.

4.3 Agenten von SNMP-AGENTS starten und beenden

Es wird **nicht** empfohlen, die Agenten manuell zu starten. Verwenden Sie stattdessen rc-Skripts zum Starten/Stoppen.

Die Liefereinheit SNMP-AGENTS V1.0 umfasst folgende Agenten:

- `appMonAgent` - Application Monitor Agent zur Überwachung von Subsystemen, BCAM- und User-Anwendungen sowie Jobvariablen und Logging-Dateien
- `consoleAgent` - Console Monitor Agent zur Überwachung der Konsole
- `hostAgent` - Host Resources Agent zum Informieren über das System, über Geräte und Dateisysteme sowie über die installierte Software.
- `hsmsAgent` - HSMS Agent zur Überwachung des Speichermanagement Systems HSMS
- `openFTAgent` - openFT Agent zur Überwachung des File Transfers von openFT
- `openSM2Agent` - openSM2 Agent zur Überwachung der Performance
- `utmAgent` - openUTM Agent zur Überwachung von UTM-Anwendungen
- `spoolAgent` - Spool & Print Service Agent zur Überwachung der SPOOL- und RSO-Geräte
- `storageAgent` - Storage Agent zur Überwachung von Platten und Pubsets

Die Agenten können über rc-Skripts gestartet und gestoppt werden, siehe rc-Skripts. Dies ist das empfohlene Vorgehen.

Zusätzlich ist es möglich, die Agenten einzeln in POSIX über ihren Programmnamen zu starten, siehe Abschnitt „Agenten-spezifische Optionen zum manuell Starten der Agenten“.

Voraussetzungen

Voraussetzung für das manuelle Starten der Agenten sind:

– eine betriebsbereite LAN-Verbindung zwischen BS2000-Rechner und Management-Plattform
– ein gestartetes POSIX-Subsystem
– ein installiertes Subsystem SNMPAGT
– ein laufender SNMP-Dämon *snmpd*

Zum Starten der Agenten wird das Privileg SYSROOT benötigt.

4.3.1 Agenten-spezifische Optionen zum manuell Starten der Agenten

Application Monitor Agent

```
appMonAgent [-c <inputfile>]
            [-t <int>]
            [-r <old-file> <new-file>]
```

Beendet wird der Application Monitor in der POSIX-Shell mit:

```
appmoncmd T
```

Beschreibung der Optionen:

-c <BS2:inputfile>

Beim Start des Application Monitor kann eine Konfigurationsdatei angegeben werden (siehe Abschnitt „Anweisungen für die Konfigurationsdatei“). Wird keine Konfigurationsdatei angegeben, werden all diejenigen Subsysteme überwacht, die beim Starten des Application Monitor Agenten dem BS2000-System bekannt waren. Die Konfigurationsdatei, definiert durch die Angabe *<inputfile>*, muss im BS2000-Filesystem abgespeichert sein.

-t <int>

Zeitintervall, in dem der Agent überprüft, ob Anforderungen vom Kommandoprogramm vorliegen. Das Zeitintervall ist standardmäßig auf fünf Sekunden eingestellt.

Die Dateiüberwachung wird bei Ablauf des Zeitintervalls durchgeführt.
Das Überwachungsintervall für Subsysteme errechnet sich aus dem fünffachen Wert des eingestellten Zeitintervalls, also im Standardfall 25 Sekunden.
Zustandsänderungen von Anwendungen bzw. Jobvariablen werden u.U. erst bei Ablauf des Zeitintervalls gemeldet.
Das Überwachungsintervall für DCAM-Anwendungen errechnet sich aus dem 60-fachen Wert des eingestellten Zeitintervalls, beträgt also im Standardfall 5 Minuten.

-r <BS2:old-file> <BS2:new-file>

Konvertiert die im SDF-Format vorliegende „alte“ Konfigurationsdatei <old-file> in die Konfigurationsdatei <new-file> mit dem neuen Format.

Console Monitor Agent

```
consoleAgent -o <operid>
             -y <op-role1> [,<op-role2>, ...., <op-role10>]
             -a <auth-file>
             [-k <password>]
             [-c <BS2:msg-filter>]
             [-n <BS2:negative-msg-filter>]
             [-r <proc>]
```

Der Agent benötigt Berechtigungsdaten für das Console Logging. Diese können auf zwei Arten übergeben werden:
- direkt über die Optionen *-o*, *-y* und optional *-k*
- oder über die Authentifizierungsdatei per Option *-a*

D.h. Sie müssen entweder *-a* oder *-o*, *-y* und *-k* angeben!

Der Console Monitor besitzt noch weitere Funktionen, siehe Zusätzliche Funktionen des Console Monitor Agenten.

Beschreibung der Optionen:

-o <operid>

definiert die Operator-Id.
Pflichtoperand, wenn *-a* nicht angegeben wurde.

-y <op-role1> [,<op-role2>, ..., <op-role10>]

Name der Operator-Rolle, die die zur Konsolüberwachung relevanten Routingcodes enthält.
Pflichtoperand, wenn *-a* nicht angegeben wurde.

-a <auth-file>

Alternativ kann mit *-a <auth-file>* der vollständige Pfadname der Authentifizierungsdatei angegeben werden, in der die Rolle(n) und ggf. das Passwort definiert sind.
Beispiel: */etc/snmp/.conMonAuth*.
Pflichtoperand, wenn *-o* und *-y* nicht angegeben wurden.

Bitte beachten Sie, dass *<auth-file>* erstellt werden muss, bevor der Agent gestartet wird. Dazu können Sie die Option *-A* verwenden, siehe unten.

-k <password>

Definition des Passworts, das den Agenten zum Zugriff auf $CONSOLE berechtigt. Keine Angabe (Standardwert) bedeutet, dass kein Passwort angegeben werden muss.

-k darf nicht zusammen mit *-a* angegeben werden.

-c <BS2:msg-filter>

Pfadname der Datei, die die Filterkonfiguration des Agenten enthält. Keine Angabe (Standardwert) bedeutet, dass alle Meldungen einen Trap erzeugen.

-n <BS2:negative-msg-filter>

Pfadname der Datei, die die negative Filterkonfiguration des Agenten enthält. Keine Angabe (Standardwert) bedeutet, dass keine Meldungen ausgefiltert wird.

-r <proc-name>

Name des Rechners, für den der Console Agent gestartet werden soll.

Zusätzliche Funktionen des Console Monitor Agenten

– `consoleAgent -A <auth-file>`

Gibt ein Prompt aus, mit dem eine Authentifizierungsdatei erzeugt werden kann. Diese Datei wird mit AES verschlüsselt gehalten.

– `consoleAgent -s <auth-file>`

Gibt die in der Authentifizierungsdatei definierten Operator-Ids und Rolle(n) aus. Ist ein Passwort definiert, wird *SET ausgegeben, andernfalls *NONE.

<*auth-file*> ist der absolute Pfadname der Authentifizierungsdatei.

HSMS Agent

Starten des HSMS Agenten in der POSIX-Shell mit:

```
hsmsAgent -l <HSMS-library>
```

Beschreibung der Option:

-l <HSMS-library>

Pfadname der HSMS SYSLIB, Pflichtparameter.

openUTM Agent für die Überwachung von UTM-Anwendungen

```
utmAgent [-c <BS2:inputfile>]
```

Beschreibung der Option:

-c <BS2:inputfile>

Beim Start des Agenten kann eine Konfigurationsdatei angegeben werden (siehe Abschnitt „Konfiguration des openUTM Agenten“). Die Konfigurationsdatei muss im BS2000-Dateisystem gespeichert sein. Standard: Es wird keine Konfigurationsdatei verwendet.

Storage Agent

Starten des Storage Agenten in der POSIX-Shell mit:

```
storageAgent [-c <BS2:inputfile>]
```

Beschreibung der Option:

-c <BS2:inputfile>

Name der BS2000-Konfigurationsdatei mit den Pubsets, Platten oder ROBAR-Anwendungen, die überwacht werden sollen.

4.4 BCAM, FTP und SESAM/SQL Agenten manuell starten

Diese Agenten sind Bestandteil der jeweiligen Produkte und ergänzen die Funktionalität von SNMP-AGENTS wie folgt:

- `bcamAgent` - Agent zum Überwachen und Administrieren von openNet Server Parametern
- `ftpAgent` - Agent zum Überwachen und Administrieren von FTP-Servern
- `sesAgent` - Agent zum Überwachen SESAM/SQL Datenbanken

Alle diese Agtenten besitzen keine agenten-spezifische Optionen , weshalb sie als einfaches Binärprogramm zm Ablauf gebracht werden können.

Es wird **nicht** empfohlen, die Agenten manuell zu starten. Verwenden Sie stattdessen rc-Skripts zum Starten/Stoppen.

5 Funktionen von NET-SNMP

5.1 Unterstützung der MIB-II (RFC 1213)

NET-SNMP V5.7 unterstützt das in RFC 1213 definierte SNMP-Management mithilfe von folgenden Gruppen der MIB-II:

- System-Gruppe zur Überwachung des Systems
- SNMP-Gruppe zur SNMP-Überwachung
- Interface-Gruppe
- IP-Gruppe
- ICMP-Gruppe
- TCP-Gruppe
- UDP-Gruppe

Darüber hinaus stehen eine Reihe standardisierter und proprietärer MIBs für die SNMP-Administration zur Verfügung.

NET-SNMP unterstützt außerdem einzelne, für das SNMP-Management in BS2000-Systemen relevante Objekte anderer MIBs.

Die Werte (Definition, Zugriff,...) für die MIB-Gruppen können Sie sich mit einem MIB-Browser anzeigen lassen.

5.2 Weitere von NET-SNMP unterstütze MIBs

5.2.1 SNMP-Framework-MIB (SNMP Engine)

Die SNMP Engine ist in RFC 2271 definiert.

5.2.2 Von NET-SNMP unterstützte Objekte anderer MIBs

Neben den MIBs für das System- und SNMP-Management unterstützt NET-SNMP weitere, für das SNMP-Management in BS2000-Systemen relevante Objekte der nachfolgend aufgelisteten MIBs.

Standardisierte MIBs

MIB	root OID der MIB
SNMP-MPD-MIB	snmpMPDMIB
SNMP-TARGET-MIB	snmpTargetMIB
SNMP-NOTIFICATION-MIB	snmpNotificationMIB
NET-SNMP-EXTEND-MIB	netSnmpExtendMIB
SNMPv2-MIB	snmpMIB
IF-MIB	ifMIB
NOTIFICATION-LOG-MIB	notificationLogMIB
DISMAN-EVENT-MIB	dismanEventMIB
DISMAN-SCHEDULE-MIB	schedMIB
NET-SNMP-AGENT-MIB	netSnmpAgentMIB
SNMP-USER-BASED-SM-MIB	snmpUsmMIB
SNMP-FRAMEWORK-MIB	snmpFrameworkMIB
SNMP-VIEW-BASED-ACM-MIB	snmpVacmMIB
NET-SNMP-VACM-MIB	netSnmpVacmMIB

5.3 Funktionalität von Event Services

Event Services implementiert die Event MIB (RFC 2981). Die Event-MIB ist in die folgenden beiden Abschnitte (Sections) unterteilt, die den Funktionsumfang von Event Services festlegen:

– Trigger-Section
– Event-Section

Diese Sections werden über Tabellen in der Datei *snmpd.conf* entsprechend den Richtlinien in Abschnitt „DisMan Event MIB“konfiguriert.

Trigger Section

Die Trigger Section definiert die zu überwachenden MIB-Objekte sowie die Bedingungen (Trigger Tests), bei denen ein Ereignis auslöst wird, wie z.B.

- **mteTriggerValueID** spezifiziert die OID des zu prüfenden MIB-Objekts.
- **mteTriggerTest** spezifiziert die zu testenden Bedingungen, z.B.
  – existence: ob das in TriggerValueID spezifizierte MIB-Objekt existiert.
  – boolean: ob der Wert dieses MIB-Objekts mit einem in der BooleanTable definierten Wert (TriggerBooleanValue) übereinstimmt.
  – threshold: ob dieser Wert einen in der ThresholdTable definierten Schwellenwert über- oder unterschreitet.
- **mteTriggerSampleType** spezifiziert, ob der Vergleichswert als absoluter Wert (absolute) oder als Differenz (delta) zu einem bei einer früheren Abfrage ermittelten Wert interpretiert werden soll.
- **mteTriggerFrequency** spezifiziert das Zeitintervall (in Sekunden) zwischen zwei aufeinanderfolgenden Abfragen.

Abhängig vom Test-Typ sind Einträge in einer weiteren Tabelle erforderlich:

- **mteTriggerExistenceTable:** Objekt existiert / verschwindet / ändert Wert.
- **mteTriggerBooleanTable:** Vergleichstest des Objekt- bzw. Delta-Werts mit dem Vergleichswert.
- **mteTriggerThresholdTable:** Objekt- bzw. Delta-Wert überschreitet / unterschreitet Grenzwert.

Bei positivem Testergebnis wird für die auszuführende Aktion ein passender Eintrag in der EventTable gesucht.

Event Section

Die Event Section definiert im Objekt EventAction, welche Aktion - SNMP-Trap und/oder SNMP-SetRequest - als Reaktion auf einen erfolgreichen Trigger Test ausgelöst werden soll:

- **notification** (SNMP-Trap): Die OID des zu sendenden Traps wird in der EventNotificationTable festgelegt.
- **set** (SNMP-SetRequest): Die OID des MIB-Objekts und der zu setzende Wert werden in derEventSetTable festgelegt.

Notifications

Die Event-MIB bietet folgende Traps an, die als Reaktion auf ausgelöste Ereignisse gesendet werden können.

- **mteTriggerFired** meldet, dass der Trigger, der ein Objekt überwacht, ausgelöst wurde.
- **mteTriggerRising** meldet, dass der Schwellenwert überschritten wurde.
- **mteTriggerFalling** meldet, dass der Schwellenwert unterschritten wurde.

Die bei einer Notification mitgegebenen Objekte sind in der **Objects Table** eingetragen und werden angegeben im jeweiligen Eintrag in der

– Trigger Table
– ExistenceTable / Boolean Table / Threshold Table und / oder
– Notification Table

Folgende Notifications sollten nur zur Diagnose von Problemen verwendet werden, die im Fehlerzähler auftauchen und anders nicht gefunden werden können:

- **mteTriggerFailure** meldet, dass ein Versuch, einen Trigger zu überprüfen, fehlgeschlagen ist.
- **mteEventSetFailure** meldet, dass ein Versuch, ein 'set' als Antwort auf einen Event zu setzen, fehlgeschlagen ist.

5.4 Funktionalität von Scheduling Services

Scheduling Services implementiert die Scheduler MIB (RFC 2591), die folgende Arten des Scheduling unterstützt:

– Periodisches Scheduling
– Scheduling auf der Basis von kalendarischen Daten
– Einzel-Scheduling

Die Scheduling Service sollten in der Datei *snmpd.conf* entsprechend den Richtlinien in Abschnitt „DisMan Schedule MIB“konfiguriert werden.

Periodisches Scheduling

Periodisches Scheduling basiert auf festgelegten Zeitintervallen zwischen zwei aufeinanderfolgenden, von Scheduler Services initiierten SNMP-Set-Operationen. Ein Zeitintervall definieren Sie durch die Anzahl der Sekunden, die zwischen zwei aufeinander folgenden SNMP-Anweisungen vergehen.

Scheduling auf Basis kalendarischer Daten

Scheduling auf Basis kalendarischer Daten initiiert Aktionen an festgelegten Wochentagen oder bestimmten Tagen eines Monats. Einen kalendarischen Zeitpunkt spezifizieren Sie durch Angabe von Monat, Tag, Wochentag, Stunde und Minute.

Für jedes Datum können Sie eine Vielzahl von Werten spezifizieren und auf diese Weise ein komplexes Scheduling definieren. Das Scheduling kann beispielsweise an einem festgelegten Wochentag alle 15 Minuten eine bestimmte Aktion anstoßen.

Datumsangaben, basierend auf Monaten, Tagen und Wochentagen können Sie mithilfe der folgenden Scheduler MIB-Objekte des Typs BITS festlegen:

– *schedMonth*
– *schedDay*
– *schedWeekDay*

Das Setzen mehrerer Bits in einem dieser MIB-Objekte hat die Wirkung einer logischen ODER-Knüpfung. Wenn Sie beispielsweise in *schedWeekDay* die Bits Monday (1) und Friday (5) setzen, initiiert das Scheduling die Aktionen genau an Montagen und Freitagen.

Die objekt-übergreifende Kombination der Bitfelder von *schedMonth*, *schedDay* und *schedWeekDay* hat den Effekt einer logischen UND-Verknüpfung. Wenn Sie z.B. die Bits June (5) und July (6) in *schedMonth* setzen und die Bitfelder Monday (1) und Friday (5) in *schedWeekDay* setzen, beschränkt sich das Scheduling darauf, Aktionen ausschließlich montags und freitags in den Monaten Juni und Juli zu initiieren.

Wildcard-Funktionalität bei Datumsangaben erzielen Sie, wenn Sie alle Bits auf „1“ setzen.

Einzel-Scheduling

Einzel-Scheduling ähnelt dem Scheduling auf Basis kalendarischer Daten. Der Unterschied besteht darin, dass Einzel-Scheduling sich nach dem Anstoßen einer Aktion automatisch außer Kraft setzt.

Aktionen

Die vom Scheduling initiierten Aktionen modellieren SNMP-Set-Operationen auf MIB-Objekte, deren OID im Objekt *schedVariable* konfiguriert wird. Der zu setzende Wert wird im Objekt *schedValue* spezifiziert. In dieser MIB definierte Aktionen sind auf Objekte des Typs INTEGER beschränkt. Diese Einschränkung mindert jedoch nicht die Verwendbarkeit der Scheduler MIB. So ist einfaches Scheduling möglich, wie z.B. in Betrieb/ausser Betrieb-Scheduling für Ressourcen, zu denen es ein korrespondierendes Status-MIB-Objekt (z.B. *ifAdminStatus*) gibt.

6 Funktionen von SNMP-AGENTS

Die in SNMP-AGENTS enthaltenen Agenten stellen die Funktionalität zur Verfügung, die in folgende MIB-Dateien beschrieben ist:

- FJ-Application-Monitoring-MIB.txt
- FJ-Console-Monitoring-MIB.txt
- HOST-RESOURCES-MIB.txt
- FJ-HSMS.txt
- FJ-OPENFT-MIB.txt
- FJ-OPENSM2-MIB.txt
- FJ-SPOOL-MIB.txt
- FJ-Storage-Management-MIB.txt
- FJ-UTM-MIB.txt

Die MIBs für die Agenten sind im Verzeichnis `/usr/share/snmp/mibs` zu finden und lassen sich mit einem MIB-Browser anzeigen, daher werden nicht im Detail beschrieben.

6.1 Application Monitor Agent

Der Application Monitor Agent gestattet die Überwachung von

- Benutzer-Anwendungen,
- BCAM-Anwendungen,
- DCAM-Anwendungen,
- Subsystemen,
- Jobvariablen und
- Protokolldateien.

Logisch zusammengehörige Bestandteile eines Prozesses (Anwendungen, Protokolldateien, Subsysteme und Jobvariablen) können gemeinsam als Gruppe überwacht werden.

Unter dem Begriff Anwendungen werden hier Programme und Tasks verstanden. Art und Umfang der Anwendungsüberwachung werden über die Konfigurationsdatei individuell gesteuert. Hinweise zur Erstellung der Konfigurationsdatei entnehmen Sie bitte dem entsprechenden Abschnitt „Anweisungen für die Konfigurationsdatei“.

Protokoll-Dateien

Die Überwachung durch Protokolldateien ist für diejenigen Anwendungen vorgesehen, die selbst keinen Trap an die Management-Station senden können. Stattdessen legen diese Anwendungen Meldungen in einer Protokolldatei ab, die durch den Application Monitor Agent überwacht wird. Der Application Monitor Agent wertet diese Meldungen aus und sendet gefilterte Meldungen als Trap an die Management-Station.

BS2000-Protokolldateien müssen vom Typ ISAM und SHAREUPD=YES sein. NFS- bzw. POSIX-Protokolldateien können ASCII- oder EBCDIC-Format haben, EBCDIC-Format ist Standard, das ASCII-Format muss in der Konfigurationsdatei entsprechend gekennzeichnet werden. Die Angabe des Dateinamens in der Konfigurationsdatei muss die Benutzerkennung im Fall BS2000 bzw. den absoluten Pfadnamen im NFS-/ POSIX-Fall enthalten. Der Agent ist sonst nicht in der Lage, zwischen BS2000- und NFS-/ POSIX-Datei zu unterscheiden.

Standardmäßig werden Protokolldateien alle 5 Sekunden vom Agenten überprüft, eine Änderung dieses Wertes ist im Startkommando mit der Option-t möglich. Werden vom Agenten Dateiänderungen erkannt, wird für neue Meldungen ein Trap an die Management-Station geschickt, wenn sie zu einem angegebenen Muster passen. Wurde kein Muster angegeben, dann lösen alle Meldungen einen Trap aus.

6.2 Console Monitor Agent

Der Console Monitor Agent überwacht die Konsolschnittstelle. Er dient zur Erfassung von Konsolmeldungen sowie zur Eingabe von Konsolkommandos.

6.2.1 Erfassung von Konsolmeldungen

Konsolmeldungen werden vom Console Monitor Agenten empfangen und einzeln mit Rechnername und Uhrzeit versehen als Trap an die Management-Station versandt. Abhängig von Anzahl, Auslastung und Größe der Rechner, die Sie vom Console Monitor Agent überwachen lassen, haben Sie eine mehr oder weniger große Meldungsflut zu bewältigen. Es wird jedoch in den seltensten Fällen sinnvoll sein, alle Konsolmeldungen zur Management-Station durchzureichen. Daher bietet der Console Monitor Agent zwei Möglichkeiten zum Filtern von Konsolmeldungen. Es werden positive und negative Meldungsfilter angeboten.

positive Meldungsfilter

1. Jeder Konsolmeldung ist ein bestimmter Routingcode zugeordnet. Durch die Auswahl bestimmter Routingcodes, die in Operator-Rollen festgelegt werden, definieren Sie die auf der Management-Station auszugebenden Meldungen anhand ihres Routingcodes.
2. Der Meldungsschlüssel der Konsolmeldung bzw. -frage oder TYPE I/Os ist ein weiteres Auswahlkriterium, mit dem Sie festlegen können, welche Meldungen der Management-Station zugestellt werden sollen. Dazu werden die relevanten Meldungsschlüssel in einer Meldungsfilter-Datei hinterlegt, die vom Console Monitor Agent beim Start bzw. im Falle einer Aktualisierung auch im laufenden Betrieb ausgewertet wird.

negativer Meldungsfilter

Der Console Monitor Agent bietet die Möglichkeit, bereits bei der Anmeldung an UCON bestimmte Meldungen zu unterdrücken.

Die Erstellung der Meldungsfilter-Datei, die beim Start des Agenten angegeben werden muss, ist ab Definition von Meldungsfiltern beschrieben. Änderungen an der Meldungsfilter-Datei im laufenden Betrieb sind durch Schreiben des MIB-Objekts *consMonMsgFilter* (positiver Meldungsfilter) und über das Kommandoprogramm möglich.

Kann die neu zugewiesene Meldungsfilter-Datei nicht geöffnet werden, wird dies mit dem Returncode *General Error* abgewiesen, und die alte Datei weiter benutzt. Enthält die Meldungsfilter-Datei keine bzw. keine gültigen Meldungsschlüssel, so werden der Management-Station keine Traps zugestellt. Der negative Meldungsfilter *consMonNegMsgFilter* kann im laufenden Betrieb nicht geändert werden.

6.3 Host Resources Agent

Der Agent Host Resources liefert Informationen über das System, über Geräte und Datei-Systeme sowie über die installierte Software entsprechend dem Standard RFC 1514.

6.4 HSMS Agent

Der HSMS Agent ermöglicht das Lesen und Ändern von globalen HSMS-Daten. Darüber hinaus liefert er detaillierte Informationen über HSMS-Aufträge und deren Zustände. Den Umfang der Anzeige können Sie durch die Auswahlkriterien "Zustand" und "Herkunftsort" einschränken. Der HSMS Agent sendet selbst keine Traps.

In einer Tabelle werden alle HSMS-Aufträge angezeigt, die vom betreffenden BS2000-Rechner bearbeitet werden. Der HSMS Agent ermittelt diese Information durch Auswerten einer OPS-Variablen.

Damit der Agent die Aufträge auch nach deren Beendigung anzeigen kann, dürfen die Aufträge nicht per Kommando gelöscht werden. Aufträge mit dem Status *COMPLETED* werden jedoch zu Beginn jeder HSMS-Session automatisch durch die implizite Recovery gelöscht.

Die Anzahl der angezeigten Aufträge kann eingeschränkt werden, abhängig vom

– Bearbeitungsstand der Aufträge,
– Rechner, von dem der Auftrag stammt.

6.5 openFT Agent

Der Filetransfer Agent dient

- zum Starten und Stoppen von openFT (BS2000)
- zur Informationsbeschaffung über Systemparameter
- zum Ändern des Public-Key zur Verschlüsselung
- zur Ausgabe von Statistikdaten
- zur Steuerung der Diagnose
- zur Ausgabe von Partner-Informationen

openFT wird via openFT Agent gestartet und gestoppt, indem der Wert von *ftStartandStop.0* auf *START* bzw. *STOP* gesetzt wird.

6.6 openSM2 Agent

Der Performance Agent für openSM2 liefert Basisinformationen zum openSM2 selbst, d.h. zum Status des Subsystems, zur Version, zur Größe des Messintervalls und zum Stichprobenzyklus.

Die eigentlichen Messwerte entsprechen den SM2-bekannten Reportgruppen und informieren über

- die CPU-Auslastung,
- I/O-Aktivitäten,
- die Auslastung des Hauptspeichers und des virtuellen Adressraums,
- die Belegung des Hauptspeichers durch die vier Standardkategorien von Tasks,
- Ein- und Ausgabeoperationen auf periphere Geräte während eines Messintervalls,
- applikationsspezifische Daten von UTM-Anwendungen,
- Verbrauchswerte einzelner Tasks.
- Messwerte für VM2000-Systeme

6.7 openUTM Agent

Der openUTM Agent bietet folgende Leistungen:

- Überwachung und Steuerung ausgewählter UTM-Anwendungen
- Informationen über Systemparameter, physikalische und logische Terminals, Terminal-Pools, Transaktionscodes, Transaktionsklassen, Benutzerdaten, Verbindungen und Statistikdaten
- Änderung von Anwendungseigenschaften und Systemparametern
- Sperren bzw. Entsperren von UTM-Datenstationen
- Wechsel der Konfigurationsdatei
- Beenden einer UTM-Anwendung

6.8 SPOOL Agent

Der Agent für Spool & Print Service dient zur Überwachung der SPOOL- und RSO-Geräte, er liefert Informationen über Geräte und Druckaufträge. Der PrintService Agent wird ausgeliefert mit einer proprietären MIB, die die Device- und die Job-Gruppe umfasst.

6.9 Storage Agent

Der Agent für das Storage-Management liefert Informationen zu Pubsets und Platten sowie über die Verfügbarkeit der Storage-Management-Produkte HSMS, MAREN und ROBAR. Dementsprechend wird mit dem Agenten eine proprietäre MIB ausgeliefert, die neben den globalen Daten des Storage-Management Agenten vier Gruppen mit folgenden Informationen enthält:

- allgemeinen Informationen zu HSMS, MAREN und ROBAR,
- Ressourcen-Informationen,
- Anzeige aller Pubsets in einer Tabelle
- Anzeige aller Platten in einer Tabelle

Für folgende Parameter gibt es konfigurierbare Überwachungseinstellungen (siehe Abschnitt „Konfiguration des Storage Agenten“):

- Änderung des Saturation-Levels für Pubsets
- Reconfiguration State von Platten

7 BCAM Agent und FTP Agent

Dieses Kapitel beschreibt die MIBs des BCAM Agenten (Bestandteil von openNet Server) und des FTP Agenten (Bestandteil von interNet Services).

7.1 BCAM Agent

Der BCAM Agent realisiert eine private MIB FJ-BCAM-MIB, die folgende Überwachungsmöglichkeiten definiert:

- BCAM-Speicherbelegung
- BCAM-Trace-Einstellungen
- in BCAM konfigurierte Anwendungen
- Verbindungen
- Router, Routen und Netzanschlüsse
- Hosts
- Mapping von Adressen und Anwendungen

7.2 FTP Agent (Bestandteil von TCP-IP-AP)

Der FTP Agent Agent realisiert eine private MIB FJ-FTP-MIB, um den FTP-Server zu überwachen und zu steuern. Es werden folgende Informationen zur Verfügung gestellt:

- Benachrichtigung über Start und Shutdown des FTP-Servers
- Server-spezifische Daten von verschiedenen Parametern und Zuständen des Servers
- Verbindungsdaten

Es stehen folgende Steuerungsmöglichkeiten für den Server stehen zur Verfügung:

- Starten des FTP-Servers
- Setzen des FTAC-Levels (wenn der FTP-Server gestartet ist)
- Shutdown des FTP-Servers
- Aktivieren/Deaktivieren eines Socket-Trace
- Aktivieren/Deaktivieren des Debugging
- Sichern der Protokolldatei
- Erhöhen der maximalen Anzahl paralleler Verbindungen
- Ändern des Timeout-Wertes für Verbindungen
- Setzen der FTAC-Jobklasse (FTACJob)

7.3 SESAM/SQL Agent

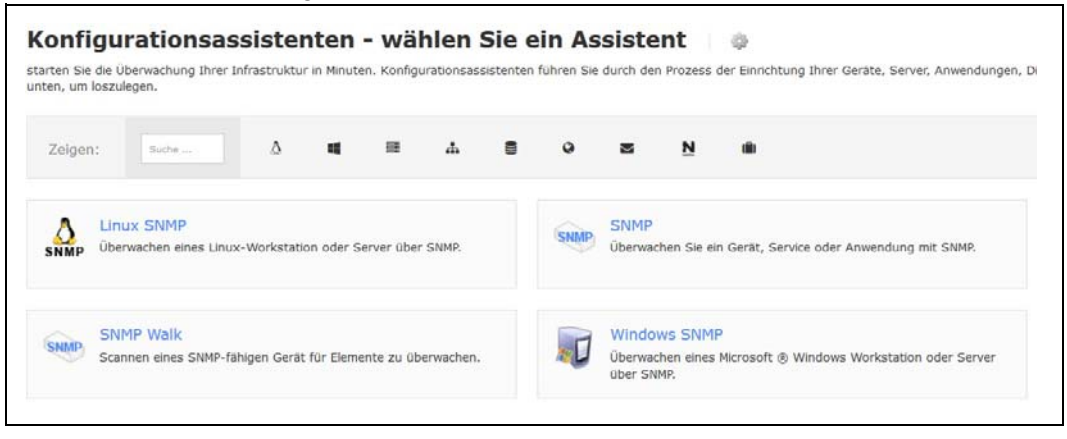
Der SESAM/SQL Agent liefert Informationen über SESAM/SQL-Datenbanken und über DBHs und DCNs, die zum Ablauf dieser Datenbanken genutzt werden.

8 Beispiel für den Betrieb der Management-Station

Diese Kapitel beschreibt, wie ein Wert aus einem BS2000 Server über SNMP ausgelesen und überwacht werden kann.

Da es eine große Anzahl an Werten und verschiedenen Parametern für jeden dieser Werte gibt, wird hier ein einfaches Beispiel dargestellt.

Rufen Sie die NagiosXI URL auf und navigieren Sie über den Reiter *Konfigurieren* zur Übersichtsseite der Konfigurationswizards. Wählen Sie dort *SNMP-Wizard* aus:

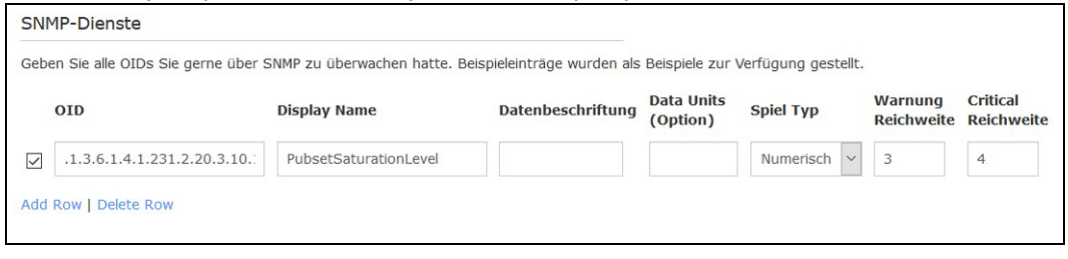


Definieren Sie das zu überwachende BS2000-System mit Hilfe der IP-Adresse (*Geräte Adresse*):

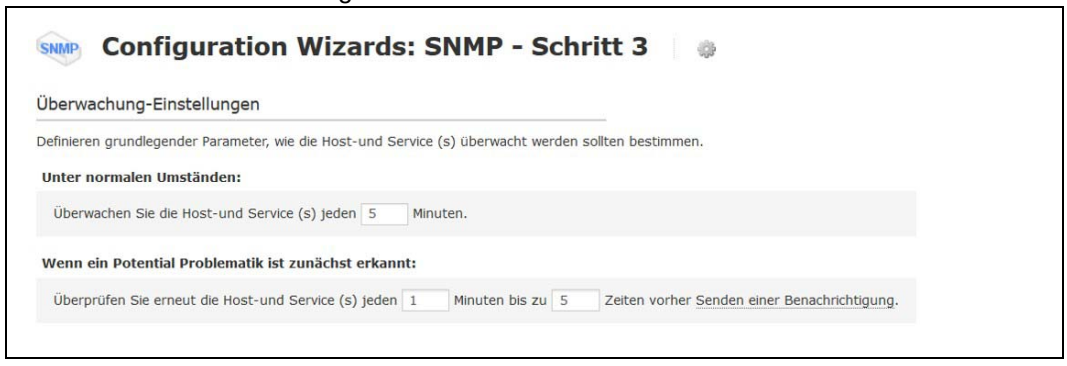
Wählen Sie (falls gewünscht oder benötigt), einen Alias/Anzeigename für den Server aus:

Wählen Sie jetzt die SNMP-Version, den Port und die SNMP-Community aus:

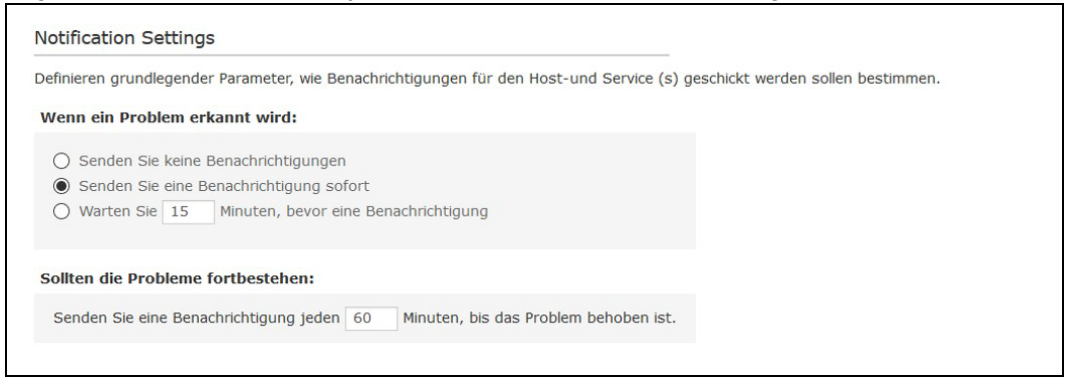
Wählen Sie aus der Management Information Base (MIB) das gewünschte Objekt (oder mehrere Objekte) mit Hilfe des Object Identifier (OID) aus:



Stellen Sie das Überwachungsintervall ein:

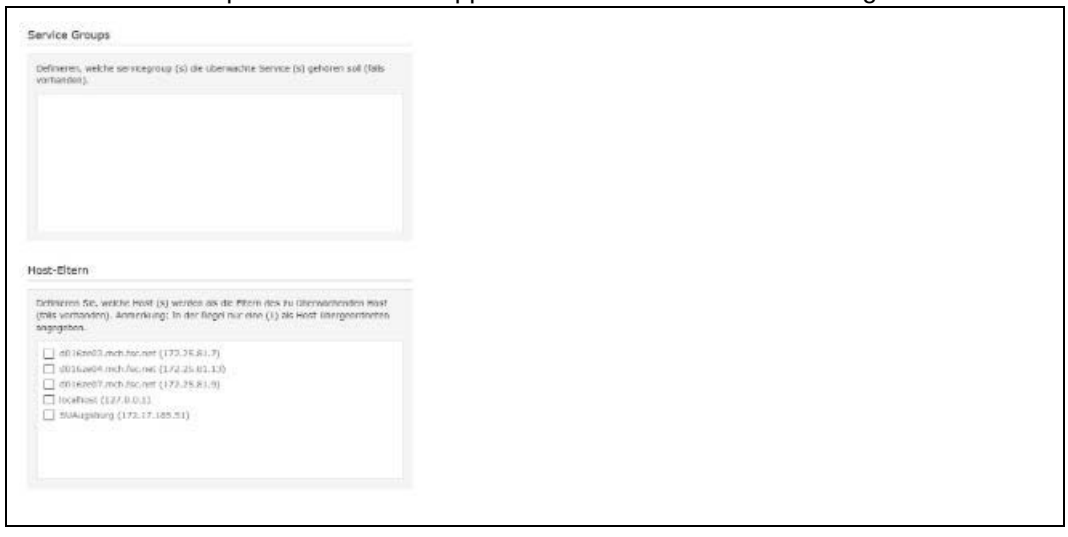


Legen Sie fest, ob und wann jemand bei Problemen benachrichtigt werden soll:

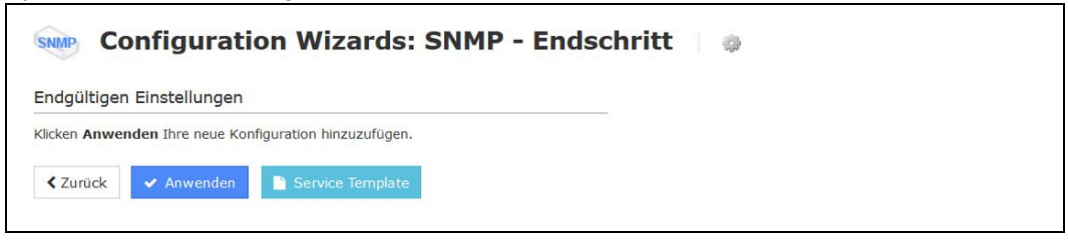


Legen Sie fest die Personen/Gruppen fest, die ggf. benachrichtigt werden sollen:

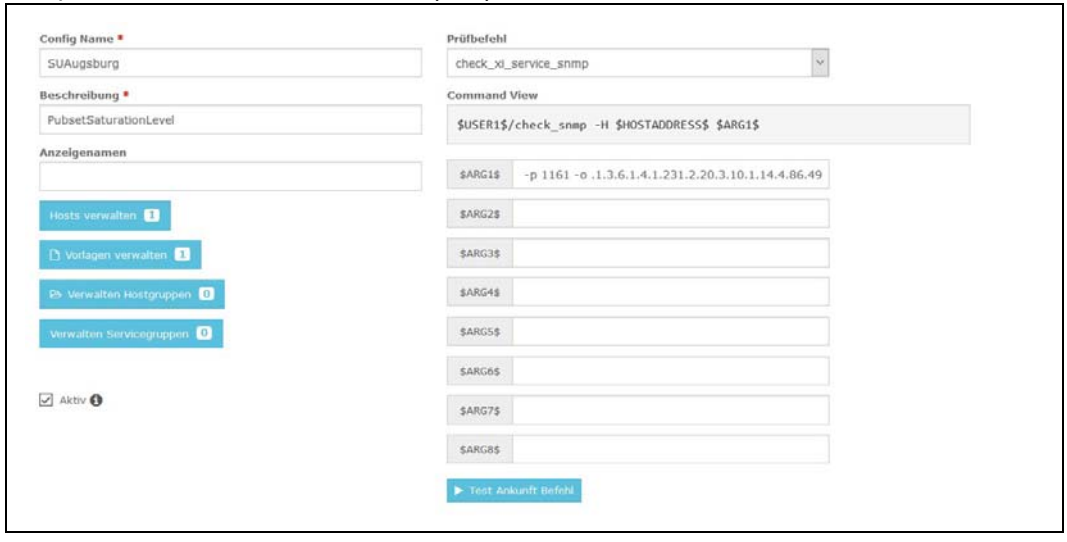
Sie können auch optional Service-Gruppen benutzen und Hosteltern ausgewählen:



Speichern Sie die Konfigurations mit *Anwenden*:

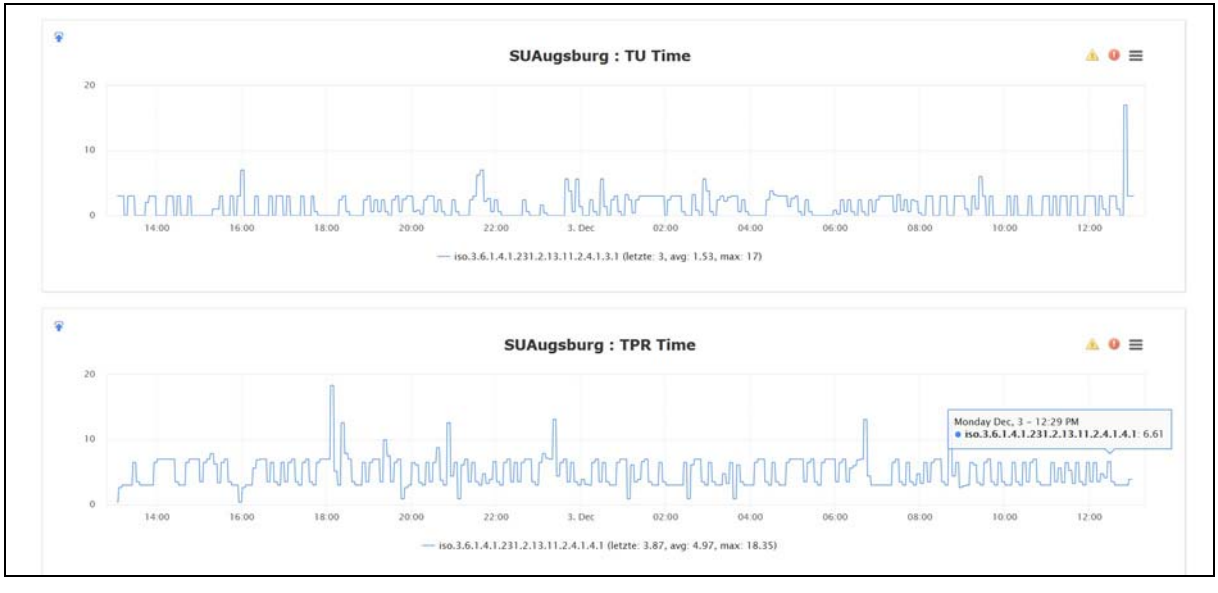


Unter *KonfigurierenCore Config Manager Dienstleistungen* können Sie die Konfiguration noch überprüfen, indem Sie auf *Test Ankunft Befehl* klicken:



Wird unser Befehl richtig verarbeitet, sollte SNMP OK und danach ein plausibler Wert ausgegeben werden. Was die einzelnen Werte bedeuten, kann in der MIB nachgesehen werden.

Unter *ZuhauseHost Status Hosteintrag - Leistungsdiagramme* (Graphsymbol)" können Sie die Leistungsdiagramme für den Host anschauen:



Mit Hilfe der blauen Pfeile oben links können diese an Dashboards angeheftet werden.

9 Anhang: Verhalten im Fehlerfall

Jeder Agent führt Logging-Dateien, in denen standardmäßig Fehlermeldungen protokolliert werden.

9.1 Format der Logging-Einträge

Jeder Agent legt seine Logging-Informationen standardmäßig in der Datei */var/adm/syslog* ab. Die Einträge dieser Logging-Datei haben das folgende Grundformat:

```
<date> <time> Schlüsselwort <daemon>[pid]: <meldungstext>
```

<date> <time>
: Zeitstempel des Logging-Eintrags.

Schlüsselwort
: Schlüsselwort für die Klassifizierung der Meldung (z.B. LOG_INFO, LOG_NOTICE, LOG_DEBUG und LOG_ERR).

daemon[pid]
: Prozessname (Dämonname) des Agenten mit der Information über das dazugehörige Prozesskennzeichen PID. Falls zur Meldungsausgabe keine gültige PID verfügbar ist (beispielsweise bei einer Meldungsausgabe während der Start-Phase des Dämons), wird ein leerer Klammerausdruck ausgegeben.

meldungstext
: Klartext der Meldung.

9.2 Logging-Dateien von Agenten konfigurieren

Das Logging ist konfigurierbar und wird in die POSIX-Datei geschrieben, die beim Start des Agenten angegeben wurde. Wird keine Datei angegeben, wird das Logging in die Standard-Logging-Datei */var/adm/syslog* geschrieben.

Beispiele:

– `bcamAgent -Lf /var/adm/bcam.log`

  startet *bcamAgent* und schreibt die Logging-Einträge in die Datei */var/adm/bcam.log*.

– `bcamAgent -Ls d`

  startet *bcamAgent* und schreibt die Logging-Einträge mit der Facility des Dämons in die Standard-Logging-Datei */var/adm/syslog*.

9.3 Debug-Optionen

Alle SNMP-bezogenen Produkte unterstützen die Debug-Optionen, die entweder in *snmp.conf* (siehe Abschnitt „Client-Verhalten“) konfiguriert oder im Kommando über die Argumente *-D[TOKEN[,...]]* übergeben werden können.

Folgende Token stehen zur Verfügung::

- in alle SNMP-Produkten:

  agentx
  init_mib
  mib_init
  parse-file
  parse-mibs
  read_config
  recv
  snmp
  transport
  trap

- nur in snmpd:

  disman
  dumpv
  dumph
  exec
  mteTrigger
  run

snmpd
snmptls
snmpusm
snmpv3
usm
usmUser

- in SNMP-AGENTS sowie in den BCAM, FTP und SESAM/SQL Agenten:

  DEBUG
  (bitte beachten Sie, dass dies eine große Menge an Diagnoseinformationen erzeugt)

Außerdem besitzt jeder Agent eigene Dubug-Token, siehe folgende Tabelle:

Agent	**Token**
appMonAgent	appMon
bcamAgent	bcam
consMonAgent	console
spoolAgent	spool
	fjSpool
ftpAgt	ftp
	ftpAgent
hostAgent	host
hsmsAgent	hsms
openFTAgent	openFT
openSM2Agent	openSM2
sesAgent	sesam
storageAgent	storage
utmAgent	utm

Fachwörter

Agent

Der Agent wird auch als Management Agent bezeichnet. Dabei handelt es sich um die Implementierung eines Management-Protokolls, die mit einer Management-Station Management-Informationen austauscht. Ein Agent ist also eine Software, die auf einem System oder Gerät abläuft und die aktuellen Informationen über das System/Gerät an einen Manager oder eine entsprechende Manager-Anwendung meldet.

Alarm

Eine Gruppe von Zuständen und Zustandsübergängen. Die Zustände entsprechen Instanzen eines Objekttyps mit Attributwerten, die vom Netzverwalter angegeben werden. Immer wenn der überwachte Objekttyp eines Geräts oder einer Leitung in einen Zustand übergeht, der vom Verwalter als Alarmzustand gekennzeichnet wurde, meldet die Management-Plattform das Ereignis durch Anzeigen eines entsprechenden Icons und durch eine Farbänderung für die Alarm- und Geräte-Icons.

Attribut

Ein Attribut ist Teil einer Objekttypdefinition in einem MIB-Modul. Es bezeichnet eine Eigenschaft in einem Objekttyp. Enthält der Objekttyp mehr als eine Instanz, so definieren die Attribute die Spalten und die Instanzen die Zeilen in einer Tabelle für den Objekttyp. Die Tabelleneinträge sind die Instanzwerte für die Attribute.
Siehe auch *Objekttyp* und *Objektinstanz*.

Community-String

Ein einfaches Passwort, das bei Hinzufügen eines Geräte-Icons im Netzbild angegeben wird. Der Agent, der auf dem Gerät läuft, benötigt dieses Passwort vom Manager, bevor Informationen über das Gerät zur Verfügung gestellt werden.

Eigenschaft

Dabei handelt es sich entweder um einen Objekttyp oder einen Eigenschaften-String. Beide können einer Eigenschaftsgruppe angehören. Ein Eigenschaften-String ist eine Eigenschaft (aber kein Objekttyp), die vom Hersteller oder dem Netzverwalter zur Begrenzung des Gültigkeitsbereichs von Polls und Alarmen

einer Eigenschaftsgruppe hinzugefügt wird. Ein Eigenschaften-String definiert das Menü und die Untermenüs, die unter dem Aktionsknopf *Objekte* in der Geräteübersicht zur Verfügung stehen. Außerdem definiert ein Eigenschaften-String die Anwendungen, die unter dem Aktionsknopf *Anwdgen* in der Geräteübersicht zur Verfügung stehen.

Ereignismeldung

Ereignismeldungen zeigen Fehler, Zustandsänderungen und ähnliche wichtige Ereignisse im System an. Sie entstehen asynchron ("spontan"), sind kommandounabhängig, objektorientiert und werden einer beliebigen Netzmanagement-Station im Netz nach dem Bestellerprinzip zugestellt.

Gateway

Ein Gateway verknüpft heterogene Netze.

Gerät

Ein Netzsystem, Router, Hub oder eine andere adressierbare Einrichtung im Netz. Nicht: eine Leitung, ein Tap oder ein Netzbild-Icon.

Herstellerspezifische Erweiterungen

Zusätzliche SNMP-Management-Objekte für ein Gerät, die von einem Hersteller für den Agenten dieses Geräts zur Verfügung gestellt werden. Sie werden häufig auch als Hersteller-MIB bezeichnet.

HTML

HTML (HyperText Markup Language) ist eine genormte Auszeichnungssprache und stellt eine Teilmenge des SGML-Standards (Standard Generalized Markup Langunage) dar. HTML-Dokumente können über das genormte Kommunikationsprotokoll HTTP zwischen beliebigen Rechnersystemen ausgetauscht werden.

HTTP

HTTP (HyperText Transfer Protocol) ist das Kommunikationsprotokoll zwischen den Systemen im World Wide Web (WWW). Mit HTTP lassen sich HTML-Dokumente zwischen beliebigen Rechnersystemen und Anwendungen austauschen.

Internet

Kommunikationsarchitektur, gekennzeichnet durch die Verwendung von TCP und IP, entstanden aus dem ARPA-Netz in USA. Erweiterungen werden durch den IAB über den RFC-Prozess kontrolliert.

IP-Adresse

Darstellung eines Anschlusspunkts im Internet:
IPv4: 4 byte (32 bit), IPv6: 16 byte.

Major-Trap-Nummer

Der SNMP-Standard (RFC 1157) definiert sieben Trap-Kategorien mit den Nummern 0 bis 6. Diese Nummern werden als Major-Trap-Nummern bezeichnet.

Management-Station

Ein System im Netz, auf dem eine Management-Anwendung abläuft.

MIB

MIB steht für "Management Information Base". Der Begriff MIB bezeichnet ein Datenmodell, das die mithilfe von Netzmanagement zu verwaltenden Netzelemente (Managed Nodes) in einer abstrakten Form beschreibt. Dieses Datenmodell besteht aus den formalen Beschreibungen von Objekttypen (Objektklassen), die nach Konventionen aus dem RFC 1157 aufgebaut sind.

MIB-II

Die MIB-II ist eine Standard-MIB, deren Verwendung im Internet verbindlich ist. Sie bietet für die Verwaltung von Geräten ein ausreichendes Datenmodell. Die MIB-II ist genormt und im RFC 1213 definiert.
Sie ist eine Erweiterung der MIB-I (RFC 1156).

Netzbild

Eine Ansammlung von Leitungen und Icons, die in einer Gruppe von verschachtelten Netzbildern angeordnet werden. Optional sind entsprechende Hintergrundbilder für die Netzbilder, die ein Netz und dessen Teilnetze darstellen.

Netzbilddatei

Eine Textdatei, die die Konfigurationsinformationen zu Ihrem Netz enthält: die Dateinamen der Hintergrundbilder für Netz- und Teilnetzbilder; die Dateinamen und Positionen der Icons für Systeme, Router, Hubs und Leitungen; Konfigurationsinformationen zu Polls, Masken und Alarmen; Eigenschaftsgruppen. Diese Datei wird auch "Map Database"-Datei oder "Map_db"-Datei genannt.

Netzbild-Icon

Ein Icon, das ein Netzbild in einer Gruppe verschachtelter Netzbilder darstellt. Das Icon wird im nächsthöheren Netzbild angezeigt. Netzbild-Icons können auch benutzerspezifisch definiert werden.

Netzmanagement-Protokoll

Das Protokoll für den Austausch von Management-Informationen.

Objekt

In einer MIB: ein Objekttyp oder Attribut.
Auf der grafischen Bedienoberfläche: Gerät, Leitung, Tap, Poll, Maske oder Alarm - bzw. eine bestimmte Instanz davon.

Objektbezeichner (object identifier, OID)

Eine Notation, die die Position eines Objekts in einem MIB-Baum angibt. So gibt 1.3.6.1.4.1.231.1.3.2 (iso.org.dod.internet.private.enterprise.fj.1.3.2) zum Beispiel ein RM600-System an. Es gibt auch MIB-Namen für den Objektbezeichner (z. B. *cisco* für einen Cisco-Router).

Objektinstanz

Repräsentant für Eigenschaften (Attributwerte) eines Geräts. Die Instanzen werden von dem Agenten des Geräts verwaltet.
Die Objektinstanz wird durch den Instanz-Bezeichner oder Index angegeben.

Objekttyp

Eine Klasse gleichartiger Objektinstanzen, die durch eine formale Beschreibung festgelegt ist. Zu einem Objekttyp kann es auf einem Gerät genau eine oder mehrere Instanzen geben. Wenn es mehrere Instanzen zu einem Objekttyp auf einem Gerät geben kann, ist der Objekttyp als Tabelle konstruiert. Die Zeilen dieser Tabelle repräsentieren jeweils eine Objektinstanz, die Spalten die Attribute des Objekttyps.
Ein anderer Name für Objekttyp ist Objektklasse.

Ping

Ein Protokoll, mit dem die IP-Ebenen-Konnektivität von einer IP-Adresse zu einer anderen geprüft wird.

Poll

Zyklische Anforderung von Informationen über MIB-Objekttypen. Die Konfiguration kann vom Netzverwalter vorgenommen werden.

Pollzyklus

Der Pollzyklus ist der Parameter, der bestimmt, wie oft SNMP Kontakt mit einem Agenten auf einem Gerät aufnimmt, um Informationen von der MIB dieses Geräts abzurufen.

Protokoll

Eine Menge an Regeln, mit deren Hilfe Systeme miteinander kommunizieren. Siehe auch *SNMP* und *Ping*.

RFC

Request for Comments. Die Dokumentreihe, die die Internet-Protokolle und verwandte Standards beschreibt.

SNMP

SNMP steht für "Simple Network Management Protocol". SNMP ist ein Standardprotokoll für das Netzmanagement in TCP/IP-Netzen.

Tap

Ein Tap stellt in einem Netzbild den Anschlusspunkt zwischen einem Gerät und dem Netz dar. Ein Tap kann erzeugt, konfiguriert und gelöscht werden, aber er kann nicht verwaltet werden.

TCP/IP

TCP/IP steht für "Transmission Control Protocol/Internet Protocol", d.h. die Internet-Protokolle. Eine Regelmenge, die definiert, wie Systeme in einer offenen (nicht herstellergebundenen) Umgebung miteinander kommunizieren. Dabei handelt es sich normalerweise um eine große Kommunikationsinfrastruktur (Internet).

Teilnetz

Ein physikalisches Netz innerhalb eines IP-Netzes.

Teilnetzbild-Icon

Ein Icon in einem Root-Netzbild oder Teilnetzbild, das ein verschachteltes Teilnetzbild eine Ebene unter dem aktuellen Netzbild oder Teilnetzbild darstellt.

Trap

Unter SNMP sind Traps Problemmeldungen, die automatisch von einem Agenten gesendet werden.

Trigger

Ein Trigger ist eine Meldung, die vom Poll- oder Maskensystem an das Alarmsystem gesendet wird. Ein Alarm führt einen Zustandsübergang durch, wenn ein bestimmter Trigger empfangen wird.

URL

URL (Uniform Resource Locator) ist eine Zeichenfolge, die der Benutzer am Web-Browser eingibt, um ein WWW-Dokument anzuwählen.
Die URL für das WWW enthält die Adresse der gewünschten Web-Seite und besteht aus den Komponenten Protokoll, Rechneradresse (Hostdomain-Name bzw. IP-Adresse), evtl. Portnummer, evtl. Pfad- und Dateiname sowie (optional) der Angabe einer Textstelle im Dokument.

Variable

Unter SNMP ist eine Variable das Ergebnis der Verknüpfung eines Objektinstanz-Namens mit einem zugeordneten Wert.

Verbindung

Die Objektinstanz, die eine (Leitungs-) Verbindung zu einem Netzmanagement-Gerät beschreibt.

Verbindungsinstanz

Eine Objektinstanz einer Verbindung zu einem Gerät. Siehe *Objektinstanz*. Einem Gerät können beide Enden eines Leitungs-Icons zugeordnet werden. Diese Verbindung hat zwei Aspekte. Zum einen ist sie eine grafische Darstellung eines Teils des physikalischen Netzes; zum anderen ist sie ein Objekttyp des Geräts (z.B. ein Objekttyp für Anschluss- oder Verbindungsinformationen).

Zustand

Alarmzustand: Ein Element in einer Alarmdefinition. (Siehe *Alarm*.)
MDC-Zustand: Das Fenster *Domain Table View* führt unter dem Eintrag *State* einen Code an. Dieser Code beschreibt, ob ein lokaler oder ferner Client Manager eine Domäne überträgt oder zurückholt.

Zustandsübergang

Änderung des Zustands für einen Alarm, die durch einen Trigger ausgelöst wird.

Literatur

BS2000-Handbücher

Die Handbücher finden Sie im Internet unter *http://manuals.ts.fujitsu.com*. Handbücher, die mit einer Bestellnummer angezeigt werden, können Sie auch in gedruckter Form bestellen.

openNet Server V4.0
BCAM V24.0A Band 1/2
Benutzerhandbuch

openNet Server V4.0
SNMP-Management für openNet Server
Benutzerhandbuch

interNet Services V3.4B (BS2000)
Administratorhandbuch

interNet Services V3.4B (BS2000)
Benutzerhandbuch

BS2000 OSD/BC V11.0
DSSM V4.3
Verwaltung von Subsystemen
Benutzerhandbuch

HSMS (BS2000)
HSMS Funktionen
Benutzerhandbuch

openFT (BS2000)
Kommandoschnittstelle
Benutzerhandbuch

openFT (BS2000)
Installation und Betrieb
Systemverwalterhandbuch

openUTM
Anwendungen generieren
Benutzerhandbuch

openSM2
Software Monitor
Benutzerhandbuch

SPOOL V4.6A (BS2000)
Benutzerhandbuch

RSO V3.5A / V3.6A(BS2000)
Remote SPOOL Output
Benutzerhandbuch

SESAM/SQL-Server V9.1 (BS2000)
Datenbankbetrieb
Benutzerhandbuch

POSIX (BS2000)
Grundlagen für Anwender und Systemverwalter
Benutzerhandbuch

Sonstige Literatur

Douglas Steedman
Abstract Syntax Notation One (ASN.1): The Tutorial and Reference
Isleworth, 1990
(ISBN 1-871802-06-7)

Marshall T. Rose
The Simple Book: An Introduction to Management of TCP/IP-based Internets
Prentice-Hall
(ISBN 0-13-812611-9)

Bestellen von RFCs

Die im Text zitierten Request for Comments (RFCs) sind unter der URL *https://www.rfc-editor.org/* erhältlich.

Stichwörter