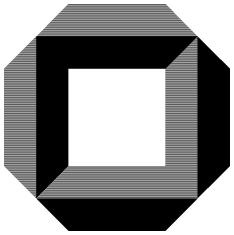


Universität Karlsruhe

Fakultät für Informatik

Informatik-Rechnerabteilung
Institut für Betriebs- und Dialogsysteme

Seminar Netzwerkmanagement

Sommersemester 1994

Betreuer und Herausgeber:

Dipl.-Inform. G. Schreiner

Prof. Dr. W. Zorn

Einleitung

Die Datenkommunikation ist in den letzten Jahren wesentlicher funktionaler Bestandteil einer Datenverarbeitungsanlage geworden, so daß fast jedes Rechensystemen in ein Netzwerk eingebunden wird. Während die theoretischen Grundlagen einer DV-Anlage in Bezug auf ihr Inneres in Forschungsbereichen wie Betriebssysteme u.ä. ergiebig untersucht wurde, blieb der Bereich der Netzverwaltung längere Zeit unbetrachtet.

Betrachtet man das Themengebiet "Netzwerkmanagement", lassen sich zwei fast gegensätzliche Entwicklungstendenzen entdecken: zum einen der OSI-Standard, der theoretisch sehr fundiert das Aufgabengebiet angeht, aber aufgrund der Komplexität noch wenig Verbreitung gefunden hat und zum anderen der Standard der TCP/IP-Welt, das Simple Network Management Protocol, welcher ein weites Einsatzfeld vorweisen kann, aber auch noch konzeptionelle Schwachstellen in sich birgt.

Dieses Seminar soll Einblick in beide Standardisierungsrichtungen, um die sich selbst eine Meinung bilden zu können.

Der Herausgeber

Inhaltsverzeichnis

1 Netzwerkmanagement — Netzwerkkontrolle **1**
1.1 Literaturverzeichnis . . . 1

2 OSI-Modell und Internet-Protokollwelt — Netzwerküberwachung **3**
2.1 Einführung in das OSI-Basisreferenzmodell . . . 3
2.1.1 Aufgabenstellung . . . 3
2.1.2 Das Modell . . . 3
2.1.3 Die einzelnen Schichten . . . 4
2.2 Einblick in die Internet-Protokollwelt . . . 5
2.2.1 Die Schichten . . . 5
2.2.2 TCP/IP-Operationen . . . 5
2.2.3 Anwendungen . . . 6
2.3 Netzwerküberwachung (Network Monitoring) . . . 6
2.3.1 Bestimmung und Gewinnung der benötigten Informationen . . . 6
2.3.2 Anwendung der erhaltenen Informationen . . . 8
2.4 Literaturverzeichnis . . . 9

3 OSI Systemmanagement **11**
3.1 Einleitung . . . 11
3.2 Systemmanagementkonzepte . . . 11
3.2.1 Systems Management Standards . . . 11
3.2.2 Architekturmodell . . . 12
3.2.3 Management Funktionsbereiche . . . 12
3.2.4 System Managementfunktionen . . . 13
3.3 Management Information Base . . . 13
3.3.1 Management Informationsmodell . . . 13
3.3.2 Benennung der Objekte in der MIB . . . 14
3.3.3 Definition von Managementinformation . . . 15
3.3.4 Templates - Beschreibungsstruktur für Managementinformation . . . 15
3.4 Literaturverzeichnis . . . 16

4 OSI Managementprotokoll **17**
4.1 Einleitung . . . 17
4.2 Was macht CMIS? . . . 17
4.2.1 Management-Notification-Service . . . 17
4.2.2 Management-Operation-Service . . . 18
4.2.3 Management-Association-Service . . . 19
4.3 Common Management Information Protocol (CMIP) . . . 20
4.3.1 CMIP - Was ist das ? . . . 20
4.3.2 CMIP im Detail . . . 20
4.4 Zusätze . . . 20
4.4.1 Remote Operation Service Element (ROSE) . . . 20
4.4.2 Association Control Service Element (ACSE) . . . 21
4.5 Zusammenfassung . . . 21
4.6 Literaturverzeichnis . . . 21

5 OSI Systemmanagementfunktionen (Teil 1) 23
5.1 Einleitung . . . 23
5.1.1 System Management Functional Areas . . . 23
5.1.2 Abrechnungsmanagement (Accounting Management) . . . 23
5.1.3 Leistungsmanagement (Performance Management) . . . 23
5.1.4 Fehlermanagement (Fault Management) . . . 23
5.1.5 Sicherheitsmanagement (Security Management) . . . 23
5.1.6 Konfigurationsmanagement (Configuration Management) . . . 23
5.2 Systemmanagementfunktionen (System Management Functions) . . . 23
5.2.1 Object-Management Function . . . 24
5.2.2 State Management Function . . . 24
5.2.3 Relationship-Management Function . . . 26
5.2.4 Alarm-Reporting Function . . . 26
5.2.5 Event-Reporting-Management Function . . . 27
5.2.6 Log-Control Function . . . 27
5.2.7 Security-Alarm-Reporting Function . . . 28
5.2.8 Security-Audit-Trail Function . . . 28
5.3 Anmerkung . . . 29
5.4 Literaturverzeichnis . . . 29

6 OSI Systemmanagementfunktionen (Teil 2) 31
6.1 Einleitung . . . 31
6.2 Access-Control-Management Function . . . 31
6.2.1 Das Zugriffskontrollmodell . . . 31
6.2.2 Zugangskontrollmechanismen . . . 33
6.2.3 Zugangskontrollobjekte . . . 33
6.3 Accounting-Meter Function . . . 35
6.4 Workload-Monitoring Function . . . 35
6.4.1 Metric-Monitoring Process . . . 35
6.4.2 Monitor Objects . . . 35
6.5 Test-Management Function . . . 37
6.5.1 Testmodell . . . 37
6.5.2 Berichtung von Testergebnissen . . . 37
6.5.3 Testbeendigung . . . 37
6.6 Summarization Function . . . 37
6.6.1 Summarization-Function Object . . . 37
6.6.2 Homogeneous Scanners . . . 37
6.6.3 Heterogeneous Scanners . . . 39
6.6.4 buffered Scanners . . . 39
6.6.5 Dynamic Scanner . . . 39
6.7 Literaturverzeichnis . . . 39

7 SNMP - Konzepte und Datenbasis 41
7.1 Einführung . . . 41
7.1.1 Die Entwicklung von TCP/IP . . . 41
7.1.2 Ursprünge des TCP/IP Netzwerkmanagements . . . 41
7.2 Grundlegende Konzepte . . . 42
7.2.1 Netzwerkmanagement-Architektur . . . 42
7.2.2 Netzwerk Management Protokoll Architektur . . . 42
7.2.3 Stellvertreter-Mechanismus (Proxy) . . . 43
7.3 Management Information Base . . . 43
7.3.1 SMI . . . 44
7.3.2 MIB-II . . . 46
7.4 Literaturverzeichnis . . . 47

8 SNMP - Simple Network Management Protocol **49**
8.1 Grundlagen . . . 49
8.1.1 SNMP Operationen . . . 49
8.1.2 Gemeinschaften (Communities) . . . 49
8.1.3 Instanzidentifikatoren . . . 50
8.2 Protokollspezifikation . . . 50
8.2.1 PDU Formate . . . 50
8.2.2 Senden einer SNMP Nachricht . . . 51
8.2.3 Empfangen einer SNMP Nachricht . . . 52
8.2.4 Variable-bindings Feld . . . 53
8.2.5 GetRequest . . . 53
8.2.6 GetNextRequest . . . 53
8.2.7 SetRequest . . . 54
8.2.8 Trap . . . 55
8.3 Probleme in der Praxis . . . 55
8.4 Einschränkungen von SNMP . . . 55
8.5 Literaturverzeichnis . . . 56

9 Secure SNMP **57**
9.1 Literaturverzeichnis . . . 57

10 Remote Network Monitoring **59**
10.1 Grundkonzept und Ziele des Remote Monitoring . . . 59
10.2 Monitorsteuerung, Mehrfach- und Tabellenmanagement . . . 59
10.3 Die RMON MIB . . . 60
10.3.1 Struktur der RMON MIB . . . 60
10.3.2 Die Statistics-Gruppe . . . 60
10.3.3 Die History-Gruppe . . . 60
10.3.4 Die Alarm-Gruppe . . . 60
10.3.5 Die Host-Gruppe . . . 61
10.3.6 Die HostTopN-Gruppe . . . 61
10.3.7 Die Matrix-Gruppe . . . 61
10.3.8 Die Filter-Gruppe . . . 61
10.3.9 Die Packet-Capture-Gruppe . . . 62
10.3.10 Die Event-Gruppe . . . 62
10.4 Praktische Aspekte . . . 63
10.5 Literaturverzeichnis . . . 63

11 SNMP Version 2 **65**
11.1 Motivation . . . 65
11.2 Gliederung von SNMPv2 . . . 65
11.2.1 Structure of Management Information (SMI) . . . 65
11.2.2 Protokoll Operationen . . . 66
11.2.3 SNMPv2 Management Informantion Base (MIBv2) . . . 66
11.2.4 Manager–zu–Manager MIBv2 . . . 67
11.2.5 Conformance Statements . . . 67
11.2.6 Koexistenz mit SNMP . . . 68
11.3 Zusammenfassung . . . 68
11.4 Literaturverzeichnis . . . 68

12 SNMPv2 Security **69**
12.1 Einleitung . . . 69
12.2 Das Sicherheitsmanagement . . . 69
12.3 Entwicklung des SNMPv2 . . . 69
12.4 Verbesserungen durch SNMPv2 . . . 70
12.5 Sicherheitsaspekte von SNMPv2 . . . 70
12.5.1 Das administrative model . . . 70
12.5.2 SNMPv2 Sicherheitsprotokolle . . . 72
12.6 Verbesserungen durch SNMPv2 . . . 74
12.7 Literaturverzeichnis . . . 75

Literaturverzeichnis **76**

Abbildungsverzeichnis

2.1 Dienstleistungsmodell nach OSI . . . 3
2.2 Datenflußdurch die OSI-Schichten . . . 4
2.3 Vergleich: OSI-Modell — TCP/IP . . . 5
2.4 IP-Kommunikation . . . 6
2.5 IP-Anwendungsarchitektur . . . 6
2.6 Netzwerküberwachung . . . 7

3.1 OSI Managementstandards . . . 11
3.2 OSI Architekturmodell . . . 12
3.3 Managementobjekt . . . 13
3.4 Beispiel eines Containmentbaumes . . . 14
3.5 Template für Objektklassen . . . 16

4.1 CMIS in der Anwendungsschicht . . . 17
4.2 Weg/Zeit-Diagramm für M-Get-Service . . . 19

6.1 Zugriffskontrollmodelle . . . 32
6.2 Beziehungen zwischen verwalteten Objekten . . . 34
6.3 Metrische Objekte . . . 36
6.4 Modell der Testverwaltungsfunktion . . . 38
6.5 Zusammenfassungsfunktion . . . 39

7.1 Struktur des SNMP Konzepts . . . 42
7.2 Protokollkonfiguration . . . 43
7.3 SNMP Architektur . . . 43
7.4 Proxy Mechanismus . . . 43
7.5 MIB Struktur . . . 44
7.7 Die Gruppe `udp` im Überblick . . . 46
7.6 Macro zur Definition on Objekten (aus RFC1212) . . . 46
7.8 Beispiel einer Objektdefinition . . . 46

8.2 `tcpConnTable` . . . 50
8.3 Instanzidentifikatoren . . . 50
8.1 Beispiel eines Instanzidentifikators . . . 51
8.4 Beispiel eines MIB-Baums . . . 51
8.6 Ablauf beim Senden einer SNMP Nachricht . . . 51
8.5 PDU Formate . . . 52
8.7 Ablauf beim Empfangen einer SNMP Nachricht . . . 52

10.1 Generierung von Alarm Events . . . 60
10.2 Ausschnitt der Datenfilter-Logik . . . 62

11.1 Übergangsdiagamm RowStatus . . . 65
11.2 SNMPv2 MIB . . . 67
11.3 Coexistence by Means of Proxy Agent . . . 67
11.4 Coexistence by Means of Bilingual Manager . . . 68

12.1 Entwicklungsgeschichte des SNMPv2 . . . 70
12.2 Erweiterungen und Verbesserungen durch SNMPv2 . . . 70
12.3 Beispiel einer Interaktion eines local context . . . 71

12.4 Beispiel einer Interaktion eines remote context 71
12.5 Darstellung einer access control policy 71
12.6 Foreign proxy relationship 72
12.7 Native proxy relationship 72
12.8 Uhrensynchronisation und Authentifizierung anhand der Zeitstempel 73
12.9 Zwei Formate einer SNMPv2-Nachricht 74

Kapitel 1

Einführung in den Begriff Netzwerkmanagement und Betrachtung des Bereiches Netzwerkkontrolle

Simone Welle

Dieser Vortrag wurde aus persönlichen Gründen auf das Wintersemester 94/95 verschoben.

1.1 Literaturverzeichnis

[LF89a] [Sta93c]

Kapitel 2

Einführung in das OSI-Modell und die Internet-Protokollwelt und Betrachtung des Bereiches Netzwerküberwachung

Rainer Hartmann

2.1 Einführung in das OSI-Basisreferenzmodell

2.1.1 Aufgabenstellung

In den letzten Jahrzehnten wurden nicht nur immer mehr Rechnersysteme eingesetzt, sondern es stieg auch der Bedarf an deren Vernetzung. Ein wichtiges Problem bei der Vernetzung verschiedener Rechnersysteme ist die Heterogenität, die Unterschiedlichkeit der zahlreichen Systeme. Eine naheliegende Lösung für dieses Problem ist Standardisierung. Wobei eine Standardisierung der Rechnersysteme zu große Einschränkungen in der weiteren Rechnerentwicklung mit sich brächten. Deshalb wandte man sich der Standardisierung von Telekommunikationssystemen zu.

Die International Organization for Standardization (ISO) begann 1977 mit der Festlegung einer prinzipiellen Architektur von Telekommunikationssystemen. Die Ergebnisse wurden 1984 als ISO/OSI-Basisreferenzmodell veröffentlicht und sollen als Gerüst für die Entwicklung von Standards dienen. OSI ist die Abkürzung für Open System Interconnection und bedeutet Kommunikation offener Systeme.

2.1.2 Das Modell

Die dem Modell zugrundeliegende Idee ist die Aufteilung der komplexen Gesamtaufgabe in überschaubare Teilaufgaben. Dies führte zu einer hierarchischen Gliederung eines Telekommunikationssystems in sieben Schichten, die gemeinsam die Kommunikation verschiedener Rechnersysteme über räumliche Distanzen hinweg ermöglichen sollen. Dabei war die Anzahl sieben ein Kompromiß folgender zwei Grundprinzipien: zum einen wollte man möglichst wenig Schichten, um die Übersichtlichkeit des Gesamtsystems zu bewahren, zum anderen aber auch nicht zu wenig Schichten, um die Aufgaben der einzelnen Schichten überschaubar spezifizieren zu können.

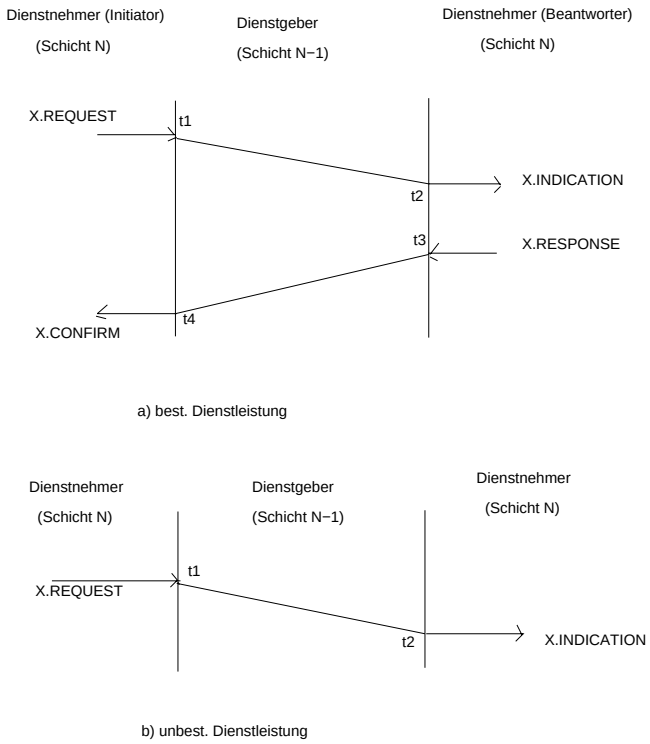


Abbildung 2.1: Dienstleistungsmodell nach OSI

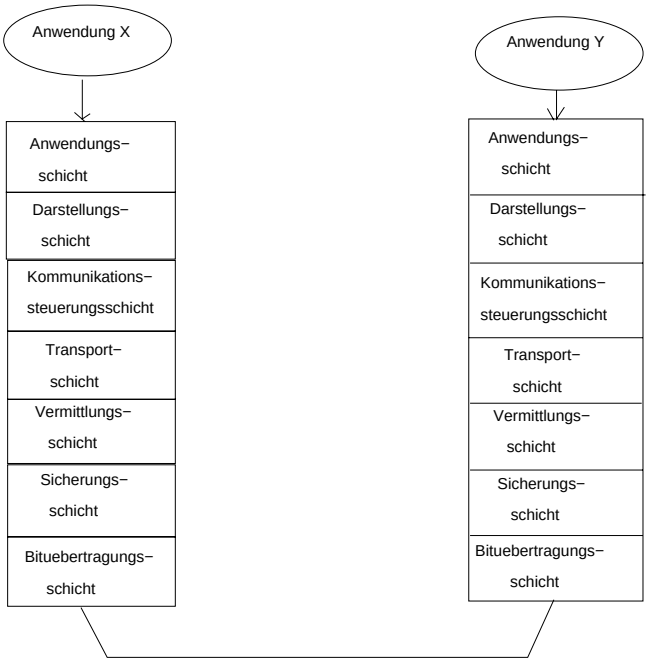


Abbildung 2.2: Datenfluß durch die OSI-Schichten

In diesem Modell realisiert also jede Schicht eine Menge von Funktionen zur Unterstützung der Telekommunikation. Sie stellt ihre Dienste der nächst höheren Schicht zur Verfügung und nimmt dazu Dienste der jeweils nächst niedrigeren Schicht in Anspruch. Die Dienstfunktionen werden der nächst höheren Schicht an sogenannten Dienstzugangspunkten angeboten. Die Menge aller Dienstzugangspunkte einer Schicht bildet die Schnittstelle zwischen den beiden beteiligten Schichten. Diese Schnittstellen sind genormt. Somit haben Änderungen an Standards einer Schicht keinen Einfluß auf die anderen Schichten, da die Art, wie die Dienste einer Schicht an ihrer Schnittstelle in Anspruch genommen werden, sich aufgrund der Normung nicht ändern darf.

Es gibt zwei Arten von Dienstleistungen, bestätigte und unbestätigte. Bei beiden Arten fordert ein Dienstnehmer, der sogenannte Initiator, eine Dienstleistung beim Dienstgeber an (request). Daraufhin erzeugt der Dienstgeber bei einem oder mehreren anderen Dienstnehmern (je nach Wunsch des Initiators) eine Anzeige (indication). Damit wäre nun die unbestätigte Dienstleistung abgeschlossen. Bei der bestätigten Dienstleistung hat nun zusätzlich der benachrichtigte Dienstnehmer noch die Möglichkeit, eine Dienstleistung in umgekehrter Richtung anzufordern, d.h. zu antworten (response). Die Antwort wird schließlich dem Initiator vom Dienstgeber angezeigt, sozusagen als Bestätigung seiner anfangs angeforderten Dienstleistung (confirmation) (siehe Bild 2.1). Als Dienstgeber fungiert entsprechend den obigen Ausführungen die nächst niedrigere Schicht, bzw. deren Instanzen.

Das Verhalten und das Zusammenspiel der möglicherweise räumlich weit voneinander entfernten Instanzen einer Schicht wird durch ein sogenanntes Telekommunikationsprotokoll geregelt. Zu diesem Zweck tauschen die Instanzen Kontrollinformationen aus. Das kann man sich folgendermaßen vorstellen: Eine Instanz der Schicht i möchte Nutzdaten zu einer anderen Instanz der Schicht i übertragen. Dazu nimmt sie die Nutzdaten, fügt die oben erwähnten Kontrollinformationen hinzu und übergibt beides zusammen an der $(i-1)$-Schnittstelle einer Instanz der Schicht $(i-1)$. Diese so entstandene Dateneinheit heißt Protokolldateneinheit oder kurz PDU (Protocol Data Unit). Um auszudrücken, daß die Kontrollinformationen in dieser Dateneinheit das Zusammenspiel von Instanzen der Schicht i regeln soll, bezeichnet man sie auch mit i-PDU. Die Instanz der Schicht $(i-1)$, der diese i-PDU übergeben wurde, soll nun die Übertragung der PDU für die darüberliegende Schicht im Zusammenspiel mit anderen Instanzen der Schicht $(i-1)$ durchführen. Zu diesem Zweck fügt sie der erhaltenen i-PDU ihre Kontrollinformationen hinzu, bildet so eine $(i-1)$-PDU und übergibt diese der nächst niedrigeren Schicht $(i-2)$. Diese Prozedur wiederholt sich bis zur untersten Schicht, in der dann die Daten mit Hilfe des physikalischen Mediums wirklich übertragen werden (siehe Bild 2.2).

Zusammenfassend kann man sagen, daß ein Telekommunikationsprotokoll dafür sorgen soll, daß die Instanzen einer Schicht den an ihrer Schnittstelle geforderten Dienst leisten. Typische Protokollmechanismen dienen der Fehlererkennung, der Fehlerbehebung, der Längenanpassung, der Flußkontrolle, der Übertragungsleistungsanpassung, der Zuteilung geteilter Medien und/oder dem Routing (siehe Abschnitt 2.2.3).

2.1.3 Die einzelnen Schichten

1 Bitübertragungsschicht (Physical Layer) Diese Schicht soll eine zu übertragende Bitfolge an

das physikalische Medium anpassen. Dazu muß eine Verstärkung eines gegebenen elektrischen Signals (Basisbandübertragung) oder eine Umformung (z.B. Frequenz-, Amplituden- oder Phasenmodulation) in ein anderes elektrisches, elektromagnetisches, optisches, usw. Signal vorgenommen werden.

2 Sicherungsschicht (Data Link Layer) Protokollmechanismen dieser Schicht dienen vor allem der Fehlerkennung und Fehlerbehebung mittels Hinzufügen und Auswerten zusätzlicher Bits.

3 Vermittlungsschicht (Network Layer) Die Instanzen dieser Schicht sind zuständig für das Finden eines Weges vom Rechnerknoten des Senders zum Rechnerknoten des Empfängers und für das Weiterleiten innerhalb dieses Weges. Diese Mechanismen nennt man Routing.

4 Transportschicht (Transport Layer) Hier wird die Verbindung von Teilnehmer zu Teilnehmer, d. h. von Anwendungsprozeß zu Anwendungsprozeß hergestellt. Falls nötig (bei unzuverlässiger Vermittlungsschicht), werden noch zusätzlich Fehlererkennungs- und Fehlerbehebungsmechanismen angewandt.

5 Kommunikationssteuerungsschicht (Session Layer) Diese Schicht steuert den Ablauf der Kommunikation, d. h. sie regelt z. B. wer wann senden darf.

6 Darstellungsschicht (Presentation Layer) Für den Fall, daß die miteinander kommunizierenden Anwendungen mit unterschiedlichen Datenformaten arbeiten, wird hier eine Anpassung vorgenommen.

7 Anwendungsschicht (Application Layer) Im Rahmen dieser Schicht werden verschiedene Anwendungsdienstelemente angeboten, die in die Anwendungsprozesse eingebunden werden können und diese unterstützen sollen. Dazu gehören z. B. Elemente zur Unterstützung von Dateitransfers und zum Aufruf entfernter Operationen.

2.2 Einblick in die Internet-Protokollwelt

TCP (Transport Control Protocol) und IP (Internet Protocol) sind Elemente eines Satzes von Protokollstandards, die vom US-Verteidigungsministerium im Zusammenhang mit dem ARPANET (Advanced Research Projects Agency NET) herausgegeben wurden. TCP/IP sind die weltweit am weitesten verbreiteten Telekommunikationsprotokolle.

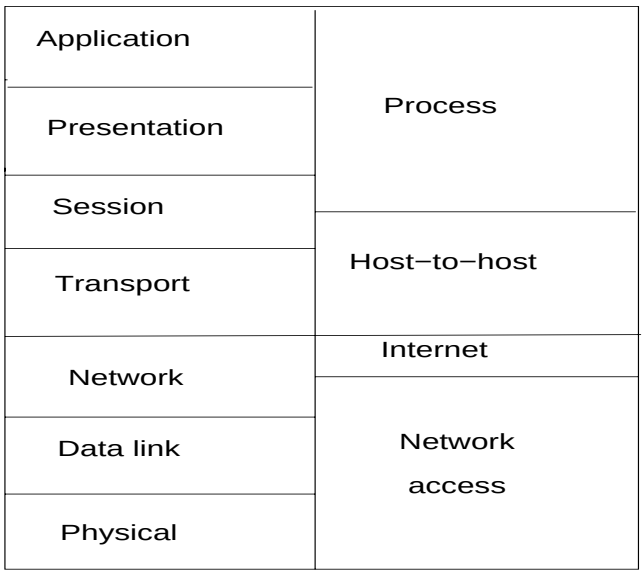


Abbildung 2.3: Vergleich: OSI-Modell — TCP/IP

2.2.1 Die Schichten

1 Network Access Layer Die Protokolle dieser Schicht erfüllen die Aufgaben der OSI-Schichten 1 und 2 (Bitübertragung und Sicherung) und zusätzlich den Netzwerkzugang, der in OSI der Schicht 3 zugeordnet wird. Allerdings stellt die TCP/IP-Serie in der Network Access Layer meist keine eigenen Protokolle zur Verfügung, sondern stützt sich auf vorhandene Implementierungen, wie z. B. X.25.

2 Internet Layer Diese Schicht sorgt dafür, daß die Daten ihren Weg vom Rechnerknoten des Senders zum Rechnerknoten des Empfängers finden, d. h. sie erledigt das Routing (siehe OSI-Schicht 3). Die Dienste der Internet Layer sind unzuverlässige Datagramm-Dienste (verbindungslos).

3 Host-to-Host Layer Im Rahmen dieser Schicht wird eine zuverlässige Ende-zu-Ende-Verbindung (d. h. von Teilnehmer zu Teilnehmer) zur Verfügung gestellt. Das Protokoll dieser Schicht heißt transport control protocol (TCP) und sieht auch Maßnahmen zur Fehlererkennung und -behebung vor, da IP wie gesagt einen unzuverlässigen Dienst anbietet.

4 Process Layer Hier werden verschiedene Protokolle für zahlreiche spezielle Anwendungen angeboten. Eine kleine Auswahl wird in Abschnitt 2.2.3 vorgestellt. Ein Überblick über die verschiedenen Schichten und der Vergleich mit den OSI-Schichten ist in Bild 2.4 dargestellt.

2.2.2 TCP/IP-Operationen

TCP ist nur in den Endsystemen implementiert, IP dagegen auch in den sogenannten Routern, die die Auf-

gabe haben, Daten zwischen verschiedenen Subnetzen weiterzuleiten (siehe Bild 2.4). Jeder Rechner hat eine weltweit eindeutige Netzadresse (IP-Adresse) und jeder Prozeß eine auf dem ausführenden Rechner eindeutige Adresse (Port).

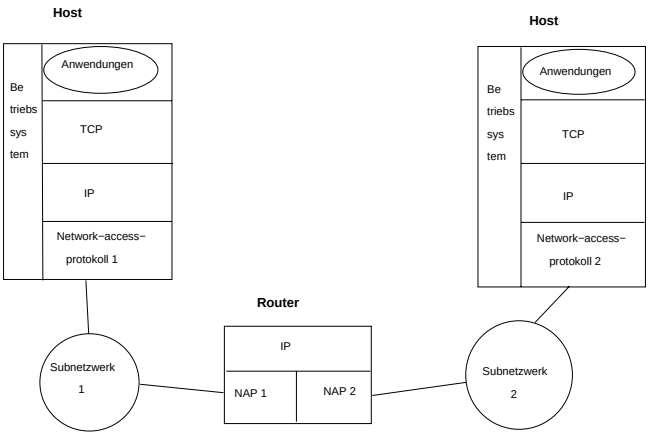


Abbildung 2.4: IP-Kommunikation

Das Versenden einer Nachricht geht nun wie schon beim OSI-Modell beschrieben vor sich (Beispiel in Bild 2.2):

Ein mit Port 15 assoziierter Prozeß auf Rechner A möchte Daten an einen mit Port 20 assoziierten Prozeß auf Rechner B senden. Die Dateneinheit wird wieder vertikal weitergereicht, wobei jede Schicht wiederum ihre Kontrollinformationen (hier mit Header bezeichnet) an die Daten anhängt. TCP ist zuständig für die Ende-zu-Ende-Verbindung und für zusätzliche Sicherungsmaßnahmen. Deshalb enthält der TCP-Header die Adresse des Empfängerprozesses (Port 20) und redundante Bits zur Fehlererkennung und -behebung. Die host-to-host layer übergibt nun die Daten an die Internet Layer mit dem Hinweis, daß diese für Rechner B bestimmt sind. Die Internet Layer fügt die Adresse des Zielrechners (B) als IP-Header hinzu und beauftragt die Network Access Layer, die Daten an den Router zu übertragen. Dieser packt den IP-Header auf der Internet Layer wieder aus und weiß damit, daß die Daten für Rechner B bestimmt sind und gibt der Network Access Layer daraufhin den Auftrag, die Daten an Rechner B zu übertragen. In Rechner B packt dann jeweils die zuständige Instanz die für sie bestimmten Header wieder aus und gibt die Daten nach oben weiter bis sie schließlich beim Zielprozeß (Port 20) ankommen.

Im Gegensatz zum ISO/OSI-Modell können hier Schichten übersprungen werden, d. h. ein Anwendungsprozeß kann z. B. direkt einen Dienst der Internet Layer in Anspruch nehmen (siehe Bild 2.5).

2.2.3 Anwendungen

Typische Protokolle der Process Layer sind SMTP, FTP und TELNET.

SMTP (Simple Mail-Transfer Protocol) ermöglicht den Transfer elektronischer Post zwischen verschiedenen

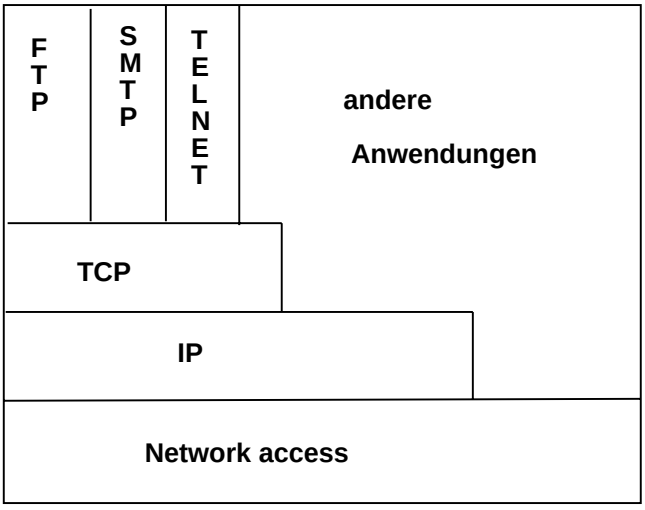


Abbildung 2.5: IP-Anwendungsarchitektur

Rechnern. Zum Angebot gehören auch Empfangsbestätigungen, Weiterleitung und Lagern von elektronischen Briefen.

Das File-Transfer Protocol (FTP) bietet das Versenden von Binär- und Textdateien an. Es regelt dazu auch den Benutzerzugriff, d. h. es gewährleistet, daß unberechtigte Benutzer keinen Zugriff auf die zu versendende Datei haben. FTP benutzt zwei Verbindungen, eine Kontroll- und eine Datenverbindung. Über die Kontrollverbindung werden die Übertragungsmodalitäten festgelegt und über die Datenverbindung werden die Dateien dann schließlich übertragen.

Mithilfe von TELNET kann man sich an entfernten Rechnern einloggen und an diesen arbeiten.

2.3 Netzwerküberwachung (Network Monitoring)

Netzwerkmanagement kann in zwei Kategorien von Aufgaben eingeteilt werden, in die Netzwerküberwachung und die Netzwerkkontrolle. Die Netzwerküberwachung ist sozusagen die "read"-Funktion oder, anschaulicher ausgedrückt, die "Diagnose", während die Netzwerkkontrolle einer "write"-Funktion bzw. der "Therapie" entspricht.

2.3.1 Bestimmung und Gewinnung der benötigten Informationen

Um ein Netzwerk sinnvoll überwachen zu können, muß man zuerst einmal über die folgenden Fragen nachdenken:

- Welche Informationen benötige ich zur Netzwerküberwachung ?
- Wo und wie erhalte ich diese ?

Informationsklassen

Man unterscheidet drei Informationsklassen: statische, dynamische und statistische Informationen.

statische Informationen Zu dieser Klasse gehören Daten, die sich selten ändern, so z. B. die Größe und die Anzahl von Hardwarebausteinen. Diese Daten werden in der jeweiligen Komponente erzeugt und meist auch dort verwaltet. Sie werden dem Netzwerkmonitor mittels einer in der Komponente installierten Vermittlungssoftware (agent function) zugänglich gemacht.

dynamische Informationen In diesen Bereich fallen ereignisabhängige Daten, z. B. Daten über Protokollzustandswechsel, etc. Diese werden entweder ebenso wie die statischen Daten in der jeweiligen Komponente verwaltet oder bei LANs auch häufig in einem sogenannten remote monitor. Dieser beobachtet den Datenverkehr im Netz und sammelt die erhaltenen Informationen.

statistische Informationen Die statistischen Informationen werden aus den dynamischen Informationen abgeleitet. Hierbei handelt es sich z. B. um Durchschnittswerte oder andere statistische Größen. Es gibt für den Monitor zwei Wege, die gewünschten Daten zu erhalten. Entweder er sammelt die dynamischen Daten und erzeugt daraus die statistischen Daten oder die für die dynamischen Daten verantwortliche Komponente berechnet die statistischen Daten selbst und sendet sie an den Monitor.

Funktionsstruktur der Netzwerküberwachung

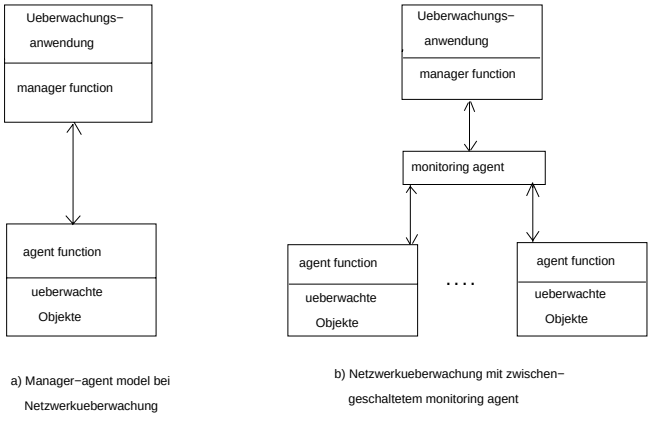


Abbildung 2.6: Netzwerküberwachung

Im Allgemeinen baut ein Netzwerküberwachungssystem auf den in Bild 2.6 dargestellten Funktionen auf. In den überwachten Komponenten (managed objects) ist eine Vermittlungssoftware (agent function) installiert, die Daten über die überwachte(n) Komponente(n) sammelt und dem Monitor mittels eines Netzes übermittelt. Auf Monitorseite werden die Daten von einer Überwachungssoftware (manager function) empfangen, bzw. beschafft. Darauf setzt dann die eigentliche Überwachungsanwendung (monitoring application) auf. Typische Überwachungsanwendungen sind Leistungsüberwachung, Fehlerüberwachung und Benutzungsüberwachung, die in Abschnitt 2.2.3 beschrieben werden. Manchmal ist zusätzlich noch ein monitoring agent zwischengeschaltet, der Zusammenfassungen und statistische Analysen erstellt (siehe Bild 2.6b). Er kann entweder beim Monitor oder in der überwachten Komponente installiert sein.

Die Station, auf der der Netzwerkmonitor installiert ist, ist natürlich selbst wieder ein Netzwerkelement und muß deshalb ebenfalls überwacht werden. D. h. auf ihr ist eine entsprechende Vermittlungssoftware eingerichtet.

Zur Überwachung von Netzwerkelementen, die ein anderes Netzwerkmanagementprotokoll benutzen, werden besondere Vermittlungsmodule, sogenannte "Stellvertreter" , eingesetzt, die die nötigen Anpassungen vornehmen.

Informationsgewinnung

Zur Informationsgewinnung gibt es zwei Methoden, die Ereignisbericht-Methode und die Polling-Methode (Umfrage).

Beim Polling richtet der Netzwerkmonitor eine Anfrage an eine bestimmte agent function mit der Bitte um Übermittlung von Informationen. Wenn der Monitor die dafür nötige Berechtigung hat, versorgt ihn die agent function daraufhin mit den gewünschten Daten aus seiner MIB (management information base). Diese Methode findet besonders in der Konfigurationsanalyse Verwendung. Außerdem ist sie hilfreich bei detaillierter Untersuchung eines Teilbereichs nach Auftreten eines Fehleralarms.

Im Gegensatz zur Polling-Methode liegt bei der Ereignisbericht-Methode die Initiative bei der agent function. Sie sendet periodisch oder beim Eintreten von bestimmten Ereignissen (z. B. Zustandswechsel, Fehler, etc.) einen Bericht an den Monitor. Vorteil dieser Methode ist vor allem, daß Probleme schneller erkannt werden. Außerdem wird ein unnötig häufiges Abfragen bei Objekten, deren Zustände sich selten ändern, vermieden.

Meist werden in Netzwerküberwachungssystemen beide Methoden mit unterschiedlicher Gewichtung eingesetzt. Welche der Methoden stärker gewichtet werden soll, hängt unter anderem von den folgenden Faktoren ab:

- Größe des Datenverkehrs, den jede Methode im Netzwerk verursacht
- Zeitverzögerung bei der Benachrichtigung des Netzwerkmonitors
- Verarbeitungsaufwand in den überwachten Komponenten
- unterstützte Anwendung

2.3.2 Anwendung der erhaltenen Informationen

Die diesem Kapitel zugrundeliegende Frage ist, wie man die erhaltenen Informationen in den verschiedenen Bereichen des Netzwerkmanagements benutzt. Wobei sich diese Arbeit auf die Anwendungen Leistungsüberwachung, Fehlerüberwachung und Benutzungsüberwachung beschränkt.

Leistungüberwachung

Leistungsindikatoren Für den Umgang mit Indikatoren ist es wichtig, besser nur eine kleinere Anzahl, dafür aber weit verbreite Indikatoren zu benutzen. Das erleichtert den Vergleich anhand der Indikatoren. Darüberhinaus sollte man bei der Wahl eines Indikators darauf achten, daß er sich gut für Vergleiche eignet, d. h. daß er z. B. gut zu quantifizieren ist. Außerdem sollte sich ein Indikator effizient berechnen lassen. Typische Leistungsindikatoren sind Verfügbarkeit, Antwortzeit, Zuverlässigkeit, Durchsatz und Auslastung.

Verfügbarkeit Die Verfügbarkeit (A) einer Komponente setzt die Zeitdauer zwischen zwei Ausfällen (t_v) ins Verhältnis zur Summe aus t_v und der durchschnittlichen Zeitdauer der Fehlerbehebung (t_r). D. h. die Verfügbarkeit errechnet sich aus $A = \frac{t_v}{t_v+t_r}$.

Die Verfügbarkeit eines Systems kann dann aus der Verfügbarkeit seiner Komponenten berechnet werden. Wobei die Verfügbarkeitsanalyse natürlich umso komplexer ist, je komplexer die Konfiguration des Systems ist. Ein kleines Beispiel für die Erhöhung der Verfügbarkeit eines Systems ist folgendes: Eine Komponente habe die Verfügbarkeit A = 0,9. Man nehme eine zweite gleichartige Komponente und schalte diese zur ersten parallel. Zur Erfüllung der Aufgabe reiche es, wenn eine der beiden Komponenten funktioniert. Die Verfügbarkeit des Systems aus den zwei parallel geschalteten Komponenten errechnet sich aus $A_p = 1 - (1 - A)^2 = 0,99$. D. h. die Verfügbarkeit dieser Funktion konnte durch Parallelschaltung einer redundanten Komponenten von 0,9 auf 0,99 erhöht werden.

Zur Bedeutung der Verfügbarkeit bedenke man folgendes: Ein einstündiger Ausfall z. B. eines Banknetzwerkes kann einen Verlust von mehreren Millionen Dollar verursachen.

Antwortzeit Die Antwortzeit ist die Zeitdauer, die ein System benötigt, eine Anfrage zu bearbeiten und zu beantworten.

Sie setzt sich zusammen aus sieben Größen,

1. der Eingabeverzögerung des Terminal (Transportdauer vom Terminal zur Kommunikationsleitung), welche direkt abhängt von der Übertragungsrate (Beispiel: bei einer Übertragungsrate von 2400 bps erhält man bei 100 Zeichen eine Eingabeverzögerung von 0,33 s);
2. der Wartezeit bei der Eingabe ins Netz, falls mehrere Terminals über einen Controller auf das Netz zugreifen;
3. der Zeit für die Übertragung der Eingabe vom Controller zum Host;
4. der Verarbeitungszeit im Hostrechner;
5. der Wartezeit vor der Übertragung des Ergebnisses (siehe 2);
6. der Zeit für die Übertragung des Ergebnisses (siehe 3);
7. der Ausgabeverzögerung des Terminal (siehe 1).

Die Antwortzeit ist eine relativ einfach zu berechnende Größe und darüberhinaus sehr wichtig für das Netzwerkmanagement. Die Forderung nach einer kurzen Antwortzeit verursacht leider meist hohe Kosten, z. B. wegen der geforderten höheren Prozessor- und Übertragungsleistung. Ein typischerweise gewünschter Wert ist eine Antwortzeit von zwei Sekunden.

Zuverlässigkeit Die Gewährleistung der Zuverlässigkeit einer Verbindung ist eigentlich Aufgabe der Protokollmechanismen (siehe Teil 2.1.3). Dennoch ist eine Fehlerüberwachung wichtig für die Entdeckung fehlerhafter Leitungen und anderer Fehler, die die Protokollmechanismen nicht verdecken können.

Durchsatz Für den Durchsatz gibt es verschiedene Definitionen, z. B. die Anzahl an Transaktionen in einer gegebenen Zeitspanne, die Anzahl an Benutzersitzungen bzgl. einer bestimmten Anwendung in einer gegebenen Zeitspanne, etc.

Auslastung Die Auslastung ist die Antwort auf die Frage, wie lange eine Komponente in einer bestimmten Zeitspanne wirklich gebraucht wird (Angabe in Prozent). Ziel ist das Auffinden von "Flaschenhälsen" und Stauungsgebieten und die anschließende Systemanpassung. Diese Aufgabe ist sehr wichtig, da die Antwortzeit eines Systems exponentiell mit der Auslastung der Ressourcen wächst.

Leistungsüberwachungsfunktionen Die Leistungsüberwachung läßt sich in drei Teilbereiche gliedern, die Leistungsmessung, die Leistungsanalyse und die künstliche Lasterzeugung.

Die Leistungsmessung wird meist von der agent function in der jeweiligen Komponente durchgeführt. Sie beobachtet den Datenverkehr in den Knoten hinein, aus dem Knoten heraus und die Anzahl der Verbindungen. Dies geht natürlich auf Kosten der Verarbeitungsleistung im Knoten. Deshalb werden in LANs häufig remote monitors eingesetzt, die einfach den Datenverkehr auf dem Netzwerk beobachten und damit die anderen Komponenten entlasten. Bei starkem Datenverkehr ist

es dem remote monitor allerdings nicht möglich, alle Datenpakete zu registrieren. Aus diesem Grund nimmt er dann nur Stichproben und wertet diese statistisch aus.

Die Leistungsanalyse wird von einem Programm durchgeführt, das Daten zusammenfaßt und darstellt. Die künstliche Lasterzeugung dient der Beobachtung des Netzwerkes unter einer kontrollierten Last.

Als Ergebnis erhält man Aussagen über die Verteilung des Datenverkehrs im Netz, die Häufigkeit einzelner Datenpakettypen, das Verhältnis zwischen Last und Leistung, die Kapazität, die Auswirkung verschiedener Paketgrößen, man erhält Leistungsindikatoren, usw. .

Fehlerüberwachung

Ziel der Fehlerüberwachung ist, Fehler möglichst schnell zu identifizieren, ihre Ursache zu bestimmen und Korrekturmaßnahmen einzuleiten.

Dabei stellen sich allerdings einige grundlegende Probleme. So gibt es unbeobachtbare Fehler, z. B. mangels unterstützender Software. Globale Verklemmungen sind häufig ebenfalls lokal nicht zu entdecken. Außerdem kann ein beobachteter Fehler seine Ursache in mehreren Komponenten haben, da ja mehrere Komponenten an einer Übertragung beteiligt sind. Umgekehrt kann ein Fehler mehrere Fehlerbeobachtungen hervorrufen. Ebenfalls möglich ist eine störende Überlagerung zwischen Diagnose und lokaler Wiederherstellungsprozedur. So können lokale Wiederherstellungsprozeduren wichtige Hinweise auf Fehler zerstören und somit eine Diagnose unmöglich machen. Auf eine vollständige Auflistung aller Probleme wird hier natürlich verzichtet.

Meist wird im Rahmen der Fehlerüberwachung die Ereignisbericht-Methode zur Benachrichtigung des Monitors angewandt. Um eine Überlastung des Netzwerkes zu vermeiden, sind die Kriterien für die Ausgabe eines Berichts angemessen streng. Verwendet man die Polling-Methode, so führt die agent function ein Logbuch über Fehler und andere signifikante Ereignisse.

Ein gutes Fehlerüberwachungssystem soll auch Fehler vorausahnen können. Zu diesem Zweck werden Schwellwerte eingerichtet, bei deren Überschreiten ein Alarm ausgelöst wird. Wenn z. B. der Anteil fehlerhaft übertragener Datenpakete einen gewissen Schwellwert übersteigt, dann kann das ein Hinweis auf einen entstehenden Fehler sein und es kann evtl. das Eintreten eines schlimmeren Fehlers noch verhindert werden. Ein gutes Fehlerüberwachungssystem führt darüberhinaus verschiedene Tests zum Isolieren und Diagnostizieren von Fehlern durch, so z. B. Konnektivitäts-, Datenkorrektheits-, Antwortzeit-, Verbindungssättigungstests, usw. .

Bei Fehlerüberwachungssystemen ist die Kooperation zwischen Benutzer und Überwachungssoftware besonders ausgeprägt, so daß eine effektive Benutzerschnittstelle von hoher Wichtigkeit ist.

Benutzungsüberwachung

Mithilfe der Benutzungsüberwachung (accounting monitoring) kann man feststellen, wer welche Netzwerkressourcen wie oft und wie lange genutzt hat. Ziel ist natürlich meist die Berechnung der Kosten der Nutzung für ein Institut, eine Projektgruppe, eine Einzelperson, etc. .

Typische Ressourcen in diesem Zusammenhang sind LANs, WANs, Server, gemietete Leitungen, Workstations, bestimmte Software, Kommunikationsdienste, Informationsdienste, usw. Wichtige Daten, die im Rahmen der Benutzungsüberwachung gesammelt werden müssen, sind die Initiator-ID, die Empfänger-ID, die Anzahl der Datenpakete, die Sicherheitsstufe, Zeitstempel, die betroffenen Ressourcen, etc.

2.4 Literaturverzeichnis

[Sta93b] [Sta93c] [Sta93m] [LKK93]

Sämtliche Abbildungen in dieser Arbeit sind dem oben genannten Buch von W. Stallings entnommen und zum Teil etwas abgewandelt worden.

Kapitel 3

OSI Systemmanagementkonzepte - Management Information Base

Ulrike Ahlf

3.1 Einleitung

Die folgende Ausarbeitung befaßt sich mit den OSI Systemmanagementkonzepten und der OSI Management Information Base. Im ersten Teil werden die OSI Systems Management Standards vorgestellt, insbesondere das Architekturmodell und die System Management Funktionen. Der zweite Teil umfaßt die Ablage der Managementdaten in der sogenannten MIB[1]. Hier findet sich die Definition von Managementinformation sowie deren Beschreibung in der Datenbank wieder.

3.2 Systemmanagementkonzepte

3.2.1 Systems Management Standards

Unter allen Standards die von OSI entworfen wurden, sind die OSI Systems Management Standards die umfangreichsten und komplexesten. Der erste Standard, der sich auf Netzwerkmanagement bezog, beschreibt das OSI Managementarchitekturmodell und wurde von ISO[2] unter dem Namen OSI Management Framework and Overview herausgegeben. Die folgenden Standardisierungen bezüglich des Managements wurden von ISO und dem internationalen Organ aller Postgesellschaften ITU[3] zusammen herausgegeben. Die Standards gliedern sich in fünf Kategorien:

- **OSI Management Framework and Overview:** ISO 7498-4 gibt eine allgemeine Einführung in Managementkonzepte, ISO 10040 gibt einen Überblick über die restlichen Dokumente,

- **CMIS/CMIP:** ISO 9595/6 beschreibt den Common Management Information Service, der Dienste für Managementanwendungen sowie das zugehörige Protokoll bereitstellt,

- **Systems Management Functions:** ISO 10164-1/-14 definiert spezielle Funktionen, die vom OSI Systemmanagement ausgeführt werden können,

- **Management Information Modell:** ISO 10165-1/-6 beschreibt die Management Information Base MIB, die alle OSI Objekte bezügl. des Netzwerkmanagements repräsentiert,

- **Layer Management:** definiert Managementinformationen, Dienste und Funktionen bezogen auf bestimmte OSI Schichten.

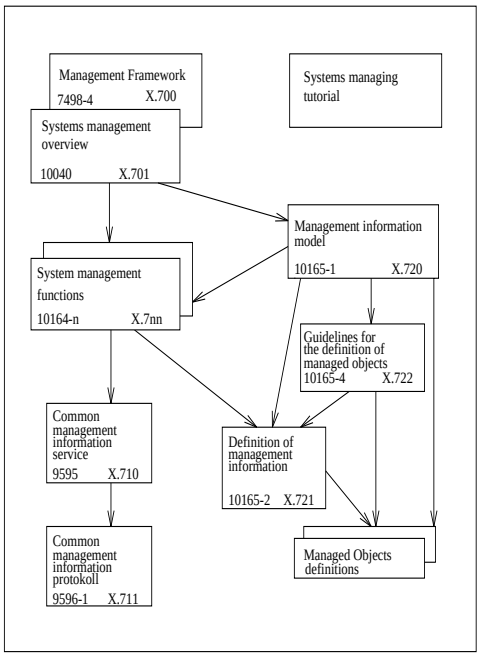


Abbildung 3.1: OSI Managementstandards

Die Standards basieren natürlich alle auf dem ISO/OSI Schichtenmodell. Ihr Zusammenwirken untereinander ist in Abbildung 3.1 dargestellt.

[1] MIB = Management Information Base
[2] ISO = International Organization for Standardisation
[3] ITU = International Telecommunication Union, früher CCITT

3.2.2 Architekturmodell

Das Architekturmodell eines OSI Teilnehmers ist in Abbildung 3.2 dargestellt.

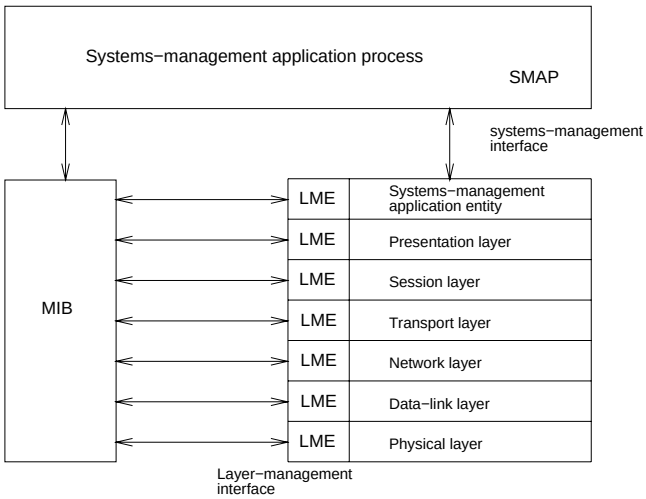


Abbildung 3.2: OSI Architekturmodell

Es bilden sich vier Kernelemente heraus:

- SMAP Systems-management application process
  Dies ist die lokale Software innerhalb eines Systems, die für die Ausführung der Systemmanagementfunktionen zuständig ist. Von hier aus wird das System verwaltet und die Koordination mit anderen SMAP's gesteuert.
- SMAE Systems-management application entity
  Dieser Dienst befindet sich in der Anwendungsschicht. SMAE ist verantwortlich für der Austausch von Managementinformationen sowohl mit Partneranwendungsinstanzen als auch mit der Netzwerkzentrale. Zum Datenaustausch wird CMIS als Standard verwendet.
- LME Layer-management entity
  Dieses Kernelement stellt Netzwerkfunktionen für die jeweiligen Schichten bereit.
- MIB Management information base
  Jeder Knoten im Netzwerk enthält spezifische Managementdaten, die hier abgelegt sind.

SMAE besteht aus einer Sammlung von mehreren ASE's[4]. Neben den Standardelementen ACSE[5] und ROSE[6] sind in SMAE noch zwei Elemente enthalten, die speziell das Netzwerkmanagement betreffen: Common Management information service element CMISE und System management application service element SMASE.

Die Hauptaufgabe von SMASE besteht darin, Dienste für den Netzwerkmanager und auch für Anwendungen verfügbar zu machen, die Netzwerkmanagementfunktionen implementieren. SMASE implementiert Managementfunktionen in den folgenden fünf Bereichen:

- Fault Management
- Accounting Management
- Configuration and Name Management
- Performance Management
- Security Management

Auf die Funktionalitäten dieser Bereiche wird im nächsten Unterkapitel noch eingegangen.

Das Element CMISE hat die Aufgabe Basismanagementfunktionen bereitzustellen, um SMASE zu unterstützen. Es soll hier jedoch nur am Rande erwähnt bleiben.

SMAP kann unterschiedliche Rollen im System einnehmen. Entweder agiert der Prozess als Manager oder als Agent, dem Manager unterstellt. In der Rolle als Netzwerkmanagers gibt der SMAP Anweisungen, Operationen und Anforderungen an die unter ihm liegenden Agenten aus. Diese leiten Managementinformationen weiter und sind für die Rückantworten sowie Meldungen verantwortlich. Die Ablage der Daten ist nicht standardisiert, lediglich der Datenaustausch.

3.2.3 Management Funktionsbereiche

Die Aufgaben des OSI Systemmanagements werden in fünf Teilbereiche gegliedert. Im folgenden werden diese kurz vorgestellt.

Das OSI Fault Management ist ein Teilbereich des Netzwerkmanagements, der es dem Systemmanager ermöglicht Fehler im Netzwerk und in der OSI Umgebung zu behandeln. Dazu werden Mittel zur Entdeckung, Eliminierung und Korrektur der auftretenden Fehler bereitgestellt. Die zugehörigen Prozeduren beinhalten die Maßnahmen:

- Fehlerentdeckung und Protokollierung
- Ausführen von Diagnosetests zur Fehlerfindung
- Fehlerkorrektur

Für die Berechnung der entstehenden Kosten und Information der Anwender bei Benutzung von Systemressourcen stellt das Accounting Management Prozeduren bereit um:

- den Benutzer über entstandene Kosten zu informieren
- Abrechnungskosten festzulegen
- Abrechnungsinformation zu protokollieren

Das Ziel des Configuration and Name Managements ist es dem Netzwerkmanager die Möglichkeit zu geben, die Systemtopologie zu kontrollieren und zu verändern. Dazu bieten Prozeduren folgende Funktionalitäten:

- Sammlung und Änderung von Ressourcendaten
- Setzen und Verändern von Netzwerkparametern

[4] ASE = Application Service Element
[5] ACSE = association-control-service element
[6] ROSE = remote-operation-service element

- Konfigurationsänderungen

Die Leistungsüberwachung des Netzes fällt in den Teilbereich Performance Managements. Dazu dienen die Netzwerkmanagementprozeduren:

- Sammeln von Daten zur Leistungsanalyse von Ressourcen
- Analyse und Planung anhand von Protokollen bezüglich der Effektivität

Der fünfte Teilbereich des Netzwerkmanagements wird vom Security Managements gebildet. Dieser Bereich umfaßt die Verwaltung sicherheitsrelevanter Informationen:

- Zugangskontrolle
- Verschlüsselung von Managementinformationen
- Archivierung von Sicherheitsinformationen

3.2.4 System Managementfunktionen

Die Bereiche des Netzwerkmanagements, die im OSI Framework definiert sind, sind nicht als solche standardisiert. Sie nehmen bezug auf die System Management Functions. Jeder Funktionsbereich nutzt einen oder mehrere dieser Funktionen. Im folgenden werden die Funktionen kurz benannt, die Erklärung der Funktionalitäten erfolgt in späteren Unterkapiteln:

- Object management
- State management
- Relationship management
- Alarm reporting
- Event-report management
- Log control
- Security alarm reporting
- Security audit trail
- Access control
- Accounting meter
- Workload monitoring
- Test management
- Summarization

3.3 Management Information Base

Grundlage jedes Managementsystems bildet eine Datenbank, in der die Informationen über Ressourcen und zu verwaltende Elemente abgelegt sind. In OSI wird die sogenannte MIB Management Information Base verwendet.

Der Rahmen zum Entwurf der MIB und zur Festlegung von Datentypen wird von SMI[7] gestellt. Im OSI Systemmanagement wird stark mit objektorientierten Konzepten gearbeitet.

Die Datenbankelemente, die verwaltet, kontrolliert und gesteuert werden sollen werden als Managed Objects MO bezeichnet. Die MIB stellt eine Sammlung von MO's dar. Es können sowohl Softwareressourcen, wie Speicherverwaltungen oder Wartschlangenprogramme als auch Hardwareressourcen, z.B. Workstations, Drucker durch MO's repräsentiert werden. Managementobjekte, die nur eine spezielle Schicht betreffen, werden als (N)-Layer-MO's, übergreifende MO's dagegen als System MO's bezeichnet. Eine Systemressource kann durch ein oder mehrere Managed Objects dargestellt werden, ebenso kann ein MO mehrere Ressourcen umfassen. Liegen Ressourcen im System vor, die nicht in der MIB enthalten sind, so sind diese für das OSI Management nicht zugänglich und können somit nicht verwaltet werden.

[7] SMI = Structure of management information

3.3.1 Management Informationsmodell

Der Standard ISO 10165-1, siehe auch Abbildung 3.1, liefert im Managementmodell einen Beschreibungsrahmen für Managementinformationen. Dort werden folgende Punkte festgelegt, die im weiteren Verlauf näher betrachtet werden sollen:

- Modell für Managed Objekts und deren Attribute
- Prinzipien der Benennung der MO's
- Definition von SMI
- Konzepte von MO-Klassen und deren Beziehung zueinander

Das Modell eines Managed Objects

Ein Managed Objekt beinhaltet Attribute, Operationen, die darauf angewendet werden können, Meldungen, die ausgegeben werden können, und Verhaltensweisen. In Abbildung 3.3 ist das Modell eines MOs dargestellt:

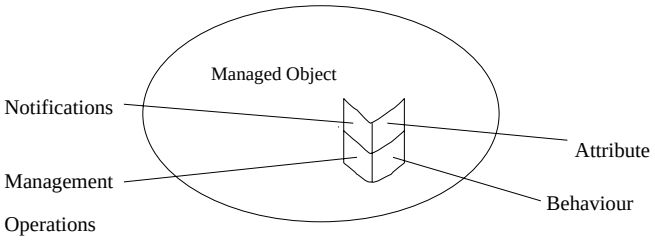


Abbildung 3.3: Managementobjekt

Das Managementobjekt ist Teil einer Klasse, die durch gleiche Eigenschaften charakterisiert ist. Die vier Eigenschaften eines MOs beinhalten folgendes:

Attribute: Jedes Attribut spiegelt die Eigenschaft einer Systemressource wieder. Es ist ein aktuelles Datenelement, welches den momentanen Status der Ressource wiedergibt. Durch Wertüberschreitung eines Attributes kann auch eine Statusänderung im System erfolgen.

Management Operation: Eine Managementoperation auf Managementobjekte macht natürlich nur Sinn, wenn die Datenkonsistenz nicht verletzt ist, und die entsprechenden Zugriffsrechte bestehen. Managementoperationen werden in attributorientierte und objektorientierte Operationen gegliedert. Attributorientierte Operationen beeinflussen nur ein einzelnes Attribut, z.B. durch Setzen eines Wertes, während die zweite Variante das ganze Objekt betrifft, beispielsweise Löschen eines MOs.

Behaviour: Ein MO beinhaltet gewisse Verhaltensweisen je nachdem wie es auf Einflüsse reagiert. Dabei wird zwischen internen und externen Stimuli unterschieden. Hinter dem Begriff Interne Stimuli verbergen sich Vorgänge, die direkt zwischen der Ressource und dem MO ablaufen. Es wäre denkbar sich einen abgelaufenen Timer als einen solchen internen Anstoß vorzustellen. Externe Stimuli dagegen treten auf, wenn Systemmanagementoperationen von anderen Stationen über CMIP auf das Managementobjekt wirken, und dieses dann durch definierte Verhaltensweisen reagiert.

Meldungen: Meldungen müssen immer dann ausgegeben werden, wenn Ereignisse auftreten, die auf das Managementobjekt einwirken. Die Meldungen werden in Protokollen an den Netzwerkmanager weitergeleitet oder in einem Ereignisbericht abgelegt.

Nutzung objektorientierter Methoden

In den Standardisierungen ist keine explizite Forderung nach objektorientierter Datenablage in der Management Information Base vorhanden. Dennoch bieten sich diese Konzepte an, da sie die Repräsentation der Ressourcen einfach verwirklichen können. Die drei wichtigsten in der MIB benutzten objektorientierten Methoden sind Kapselung, Vererbung und Allomorphismus.

Kapselung hat im Zusammenhang mit Netzwerkmanagement die folgende Bedeutung: Jeder Ressourcentyp wird durch eine Managementobjektklasse, jede Instanz dieser Ressource wird durch ein Managementobjekt dargestellt. Die Managementinformation und die Managementoperation, die auf das MO ausgeführt werden können, sind im Managementobjekt zusammengepackt. Managementanwendungen haben somit nur über das MO Zugang zur Ressource, um diese zu überwachen und zu verwalten. Daraus ergibt sich der Vorteil, das das Innere des MOs nach außen unsichtbar ist. Nur Informationen, die freigegeben sind, werden durch die Objektgrenze hinweg sichtbar.

Die Methode der Vererbung ermöglicht die Unterklassenbildung aus bereits vorhandenen Klassen. Oftmals entstehen neue Klassen durch Hinzunahme weiterer Attribute oder Meldungen. Neugenerierung von Klassen durch Löschen von Eigenschaften ist jedoch nicht erlaubt. Die Nutzung des Unterklassenprinzips bringt zwei Vorteile mit sich. Es ermöglicht die Bildung einer Hierarchie, die die Ressourcen wiederspiegeln. Außerdem wird der Beschreibungsaufwand verringert, da vererbte Charakteristiken nicht erneut definiert werden müssen.

Die dritte objektorientierte Methode ist die des Allomorphismus. Allomorphismus bietet die Möglichkeit verschiedenartige Managementobjekte zu implementieren, die aber das selbe Verhalten gegenüber der Managementstation aufweisen. So können beispielsweise alte Ressourcenmodelle durch neue ersetzt werden, indem das Verhalten der alten Elemente weiterhin simuliert wird.

3.3.2 Benennung der Objekte in der MIB

Im vorigen Abschnitt wurde deutlich, daß mittels der objektorientierten Methode der Vererbung Klassenhierarchien aufgebaut werden können. Jede Klasse ist durch einen eindeutigen Identifikator gekennzeichnet. Dieser Bezeichner ist eine Kette von Integerzahlen, die der jeweiligen Klasse einen eindeutigen Platz im sogenannten ISO Registration-Tree zuordnet. In diesem Baum sind die Definitionen für

- Managementklassen
- Attributdefinitionen
- Aktionen
- Meldungen
- Packages

abgelegt.

Ein weiterer ebenfalls klassenbezogener Baum ist der Inheritance-Tree. Dieser zeigt wie Definitionen von Objektklassen durch Verwendung von objektorientierten Methoden aus anderen Objektklassen abgeleitet werden können.

Die beiden Baumarten spiegeln das Verhältnis der Klassen zueinander wieder. Die aktuelle MIB Struktur wird jedoch durch das sogenannte Containment (engl. Enthaltensein) wiedergespiegelt. Die Möglichkeit des Containment erlaubt, daß ein Managementobjekt ein oder mehrere andere Objekte beinhaltet. Die Zuordnung erfolgt mit Hilfe einer Referenz in Form eines Objektbezeichners vom höheren zum enthaltenen Objekt. Das übergeordnete Objekt speichert diesen Objektbezeichner als Attributwert ab. Es entsteht ein Containment-Tree wie er in Abbildung 3.4 dargestellt ist.

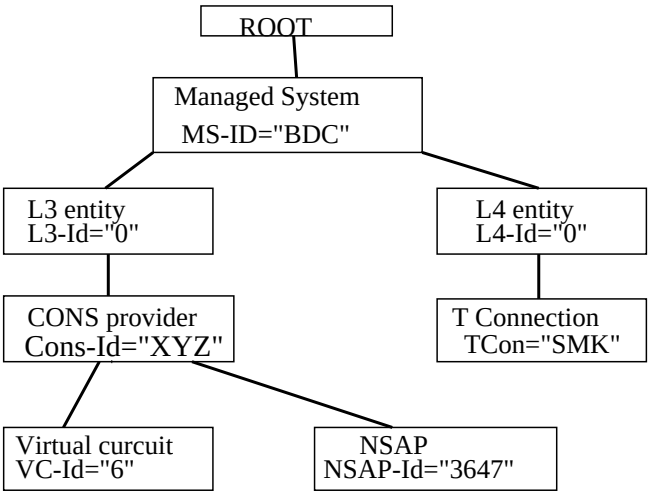


Abbildung 3.4: Beispiel eines Containmentbaumes

Jedes Managementobjekt enthält zur Benennung ein relative distinguished name und ein distinguished name. Der erstere korrespondiert mit einem bestimmten Attributwert. Dieser Wert ist für alle Objekte gleich, die aus

dem selben Objekt abgeleitet wurden. In Abbildung 3.4 ist beispielsweise MS-Id der Attributname, BDC der Wert und beide Angaben zusammen bilden den relative distinguished name. Der relative name eines Objektes setzt sich aus dem relative distinguishes names von der Wurzel des Containmentbaumes bis zum betrachteten Objekt zusammen. Im betrachteten Beispiel setzt sich der relative name des rechten Teilbaumes aus Ms-Id='BDC', L4-Id='0', TCON-Id='MSK' zusammen.

3.3.3 Definition von Managementinformation

ISO 1065-2 (X.720) definiert Basiselemente für die MIB Entwicklung. Zu diesen Elementen gehören Attribute, Objektklassen und Meldungstypen.

Oft werden im Netzwerkmanagement Zählfunktionen benötigt, diese können aus Attributen entwickelt werden. Man unterscheidet Basisattribute und die verfeinerten speziellen Attribute.

Zu den generischen Attributen gehören Zähler und Pegel. Zähler sind Abstraktionen eines zählenden Prozesses im System. Sie sind nicht negative Integerwerte und nur inkrementierbar. Erreicht ein Zähler das Maximum so springt sein Wert auf Null zurück.

Zwei Typen von Zählern sind definiert, um unterschiedliche Managementanforderungen zu erfüllen. Nichtsetzbare Zähler verändern nur ihren Wert, wenn ein Zählereignis auftritt. Managementoperationen haben keinen Einfluß auf diesen Typ. Den Haupteinsatz findet dieser Zähltyp in Situationen bei denen mehrere Managementstationen Zugang zu diesem Zähler haben. Der setzbare Zähler kann auch durch Managementoperationen beeinflußt werden, und wird daher eher von einzelnen Managementstationen benutzt.

Um eine oder mehrere Meldungen in Abhängigkeit eines Zählers auszugeben, kann ein Schwellwertattribut genutzt werden. Die Grundeigenschaft dieses Typs ist es, Meldungen bei Erreichen von Vergleichswerten auszugeben. Es kann auch Offset mit angegeben werden. Beispielsweise wenn eine Meldung nach einer bestimmten Anzahl von angekommenen Datenpaketen ausgegeben werden soll. Diese Menge wird dann als Offset definiert und jeweils zum erreichten Vergleichswert dazuaddiert.

Die Zähler sind noch nicht standardisiert. Es liegen jedoch Vorschläge vor wie z.B. PDU received, Outgoing Connection Requests oder Incoming Protokoll Error.

Unter einem Pegel versteht man die Abstraktion einer zugrundeliegenden dynamischen Variablen. Pegel sind nichtnegative Integerwerte, die sowohl in- als auch dekrementierbar sind. Außerdem besitzen sie einen Minimal- und einen Maximalwert. Ein Pegel ist stets ein nicht setzbares Attribut. Zur Meldungsausgabe kann ein Pegelschwellwert benutzt werden. Diese Pegeltypen geben pro Schwellwert zwei Meldungen aus. Den ersten wenn der Schwellwert in positiver Richtung überschritten wird, den zweiten bei Unterschreitung des Wertes zuzüglich eines Offsets, um zu häufige Meldungen zu vermeiden, falls der Pegel genau um den Schwellwert steigt und fällt. Ein weiteres Attribut ist das sogenannte Tidemarkattribut. Es dient zu Speicherung von Minimal- und Maximalwerten des Pegels bei einer Messung. Das Tidemarkattribut ist dreiwertig. Es enthält den aktuellen Wert, den Wert vor dem letzten Reset und die Resetzeit.

Die speziellen Attribute sind im Gegensatz zu den generischen Attributen völlig spezifiziert. Sie sind in der Abstract Syntax Notation One (ASN.1) beschrieben, und sind direkt in einem Managementobjekt einsetzbar. Einige Beispiele sind die speziellen Attribute Event Time, System Id, Member, Security Alarm Cause.

Ein weiteres Basiselement zur Entwicklung einer MIB stellen die Meldungstypen. ISO 10165 (X.720) stellt eine Menge von Meldungen bereit. Sie sind auf mehrere Objektklassen anwendbar, und jede Definition enthält dabei:

- Format der Meldung im Managementprotokoll
- Vehalten der Meldung
- Format der Ergebnisdaten des Protokolls
- Objektbezeichner, Wert der die Meldung identifiziert

Als Beispiel sei hier der Meldungstyp Time domain Violation mit den Attributen securityAlarmCause, securityAlarmSeverity, securityAlarmdetector usw. genannt. Die Meldungen erscheinen dann, wenn ein unerwartetes Ereignis eintritt.

Als drittes Element zur Entwicklung der Managementinformationsdatenbasis dienen die Objektklassen. ISO 10165 (X.720) stellt eine Reihe von Basisklassen zur Verfügung. Die Objektklassen können von den Systemmanagementfunktionen genutzt werden oder treten als Superklassen auf, von denen andere Klassen durch Vererbung abstammen. Beispielsweise ist TOP die Klasse aus der alle anderen Klassen entspringen.

3.3.4 Templates - Beschreibungsstruktur für Managementinformation

ISO 10165-1 schlägt Richtlinien für das Format von Definitionen von MO's, Attributen, Packages und Meldungen vor. Dieses Richtlinien liegen in Form von Templates, einer Beschreibungssprache dargestellt durch ASN.1, vor. Es existieren insgesamt neun verschiedene Templatestrukturen. Das Template der Objektklassen ist das oberste, welches alle anderen beinhalten kann. Deshalb soll es an dieser Stelle stellvertretend erwähnt werden.

Derived From gibt die Oberklasse, aus der die Klasse abstammt, bzw. TOP an. Charakteristiken, die Meldungen, Attribute usw. werden nicht nochmal erwähnt, es sei denn es liegen Änderungen vor.

Der Konstrukt Characterized by ermöglicht es, eine oder mehrere Packages in die Objektklasse einzubinden. Ein Package enthält Verweise auf Verhalten, Attribute und Meldungen. Weiterhin ist es möglich Conditional Packages anzugeben, die in die Klasse eingebunden werden, wenn bestimmte Voraussetzungen gegeben sind.

<class-label> MANAGED OBJECT CLASS

[DERIVED FROM <class-label> *Verweis auf Superklasse*
<,<class-label>]*;]

[CHARACTERIZED BY <package label> *Pflichtmerkmale*
[,<package label>]*;]

[CONDITIONAL PACKAGE <package label> *bed. Merkmale*
PRESENT IF condition-definition
[,<package label>
PRESENT IF condition-definition]*;]

REGISTERED AS object-identifier; *MO-Klassenidentifikator im ISO-Registrierbaum*

Abbildung 3.5: Template für Objektklassen

Registered as gibt den Identifikator der Objektklasse im Registrationsbaum an.

3.4 Literaturverzeichnis

[Sta93d]

Kapitel 4

OSI Managementprotokoll - CMIS und CMIP

Markus Hering

4.1 Einleitung

Diese Ausarbeitung befasst sich mit dem Thema Common Management Information Service und Common Management Information Protocol. Dabei wird zuerst eine Begriffserklärung vorgenommen und anschließend eine Einordnung in die OSI-Welt gegeben. Nachdem die Dienste von CMIS und CMIP dargestellt worden sind, werde ich als Erweiterung noch auf zusätzliche Dienste wie ACSE[1] und ROSE[2] eingehen.

Die wichtigste Funktionalität im OSI Systemmanagement (OSI = Open System Interconnection) ist der Austausch von Managementinformationen zwischen zwei Instanzen mit Hilfe eines Protokolls. Diese Funktionalität spiegelt sich wieder im Common Management Information Service Element(CMISE). Dieses Element läßt sich in zwei Teile untergliedern: CMIS und CMIP.

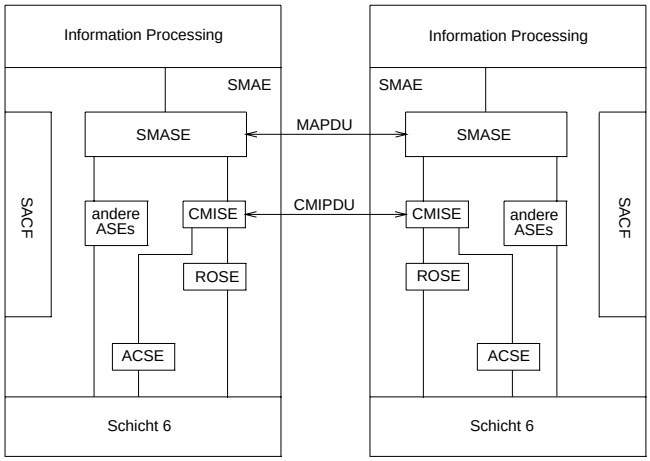


Abbildung 4.1: CMIS in der Anwendungsschicht

4.2 Was macht CMIS?

In Abbildung 4.1 erkennt man die Anordnung von CMIS und CMIP in der Schicht 7 des OSI Modells eingebettet in das CMISE. Deutlich zu erkennen ist, daß CMIS und CMIP Gebrauch von ROSE und ACSE machen. Doch dazu mehr in Kapitel 4.3.2 bzw. Kapitel 4.4.1. Alle hier aufgezählten Elemente gehören zur Klasse der Application Service Elements (ASEs).

Durch CMIS werden Dienste definiert, die für das OSI Systemmanagement durch jede Netzwerkkomponente vorgesehen sind. Die Dienste sind aufrufbar durch Dienstprimitive.

CMIS kennt drei Dienstklassen:

- Management-Notification-Services
- Management-Operation-Services
- Management-Association-Services

Diese drei Klassen wollen wir jetzt näher betrachten.

[1] ACSE = Application Control Service Element
[2] ROSE = Remote Operation Service Element

4.2.1 Management-Notification-Service

Aufgabe dieser Dienstklasse ist es, in einem Netzwerk über Ereignisse zwischen zwei Partnerinstanzen zu berichten. Diese Instanzen fungieren dabei in einer Manager/Agenten-Beziehung. In Abbildung 4.1 wird das deutlich durch die identische Anordnung der beiden großen Kästen. Wenn die linke Hälfte zum Beispiel die Managerrolle übernimmt, dann kommt der rechten die Rolle des Agenten zu.

Als einziges Dienstprimitiv dieser Klasse findet man M-EVENT-REPORT. Dieses Primitiv wird vom Agenten aufgerufen und vom Manager beantwortet. Dieser Gesichtspunkt bildet eine Ausnahme, denn bei allen anderen Diensten des CMIS ist der Manager der Aufrufende. Die Operation mit M-EVENT-REPORT kann bestätigt oder unbestätigt ablaufen. Jedem Aufruf können mehrere Parameter beigefügt werden. Wenn die Übertragung mißlingt, werden beim Antwortprimitiv Fehlerparameter beigefügt, die Auskunft darüber geben, was falsch gelaufen ist und wo der Fehler liegt.

4.2.2 Management-Operation-Service

In dieser Dienstklasse werden durch CMIS fünf Dienstprimitive zur Verfügung gestellt:

- M-GET
- M-SET
- M-ACTION
- M-CREATE
- M-DELETE

Stellvertretend für die anderen Dienste wollen wir jetzt den M-GET-Dienst etwas näher durchleuchten. Dieses ist ausreichend, denn die Art des Dienstablaufes ist bei den anderen Diensten nahezu identisch.

Parameter Name	Request/ Indication	Response/ Confirm
Invoke identifier	M	M
Linked identifier	-	C
Base-object class	M	-
Base-object instance	M	-
Scope	U	-
Filter	U	-
Access control	U	-
Synchronization	U	-
Attribute-identifier list	U	-
Managed-object class	-	C
Managed-object instance	-	C
Current time	-	U
Attribute list	-	C
Errors	-	C

Tabelle 4.1: Parameter des M-GET-Dienstes

M-Get-Service

Die wichtigste Aufgabe des M-Get-Dienstes ist es, Daten von der Management-Informations-Datenbasis (MIB) der Partnerinstanz zu holen. Hierbei werden vom Manager aus beim Request ein oder mehrere "Managed Objects" (MOs) von der MIB des Agenten angesprochen. Dabei kann für jedes MO der Wert von einem, mehreren oder allen Attributen verlangt werden. Der M-Get-Dienst ist immer bestätigt.

Abbildung 4.2 zeigt Weg/Zeit-Diagramme für den Ablauf einer Get-Operation. Teil b zeigt den einfachen Fall. Die Operation beginnt mit einem `M-Get.request` des Managers, wobei er eine Parameterliste mitschickt, die nähere Informationen über die Operation und die zu holenden Daten gibt. Auf der Agentenseite kommen diese Parameter unverändert in Form eines `M-Get.indication` an. Daraufhin weiß der Agent, auf welches MO in seiner MIB zugegriffen werden soll und führt die Operaion aus. Nach erfolgreicher Beendigung teilt er dem Manager die Ergebnisse in Form eines `M-Get.response` mit. Diese kommen in unveränderter Art bei ihm mit Hilfe eines `M-Get.confirmation` an. In diesem Beispiel wurde nur auf ein MO in der Agenten-MIB zugegriffen.

Teil c zeigt den umgekehrten Fall. Hier wird jetzt durch den Manager auf mehrere MOs zugegriffen. Die Operation beginnt wie oben mit einem Request, nur bekommen hier die Aufrufparameter (Invoke Identifier (`II`)) eine Nummer zugewiesen, die die Anzahl der anzusprechenden MOs wiederspiegelt. Dabei ist `II0` ein Initialisierungsparameter, der die Operation startet, worauf dann die Operationen auf die 1 bis n MOs ausgeführt werden. Der Parameter Linked Identifier (`LI`) sorgt für die richtige Abfolge der `M-Get.response` Operationen. Jeder Response hat einen eindeutigen Wert bzgl. des `II`. Abgeschlossen wird die Operation mit der Rückgabe des `II0`-Parameters an den Manager.

Man nennt diese Art der mehrfachen Antworten Multiple Reply.

Wenn die erfolgreiche Ausführung einer Operation mißlingt, dann wird vom Agenten ein Fehlerparameter mit in den `M-Get.response` gegeben. Diese Situation findet man in Teil a. Dieser Fehlerparameter (Error (`E`)) kommt dann unverändert beim Manager an, so daß dieser das Problem analysieren kann.

Jetzt schauen wir uns noch etwas genauer die Parameter an, die bei einer solchen Get-Operation auftreten können. Tabelle 4.1 zeigt diese detailliert. Den Invoke- und Linked Identifier haben wir schon vorgestellt. Das `c` bei `LI` bedeutet conditional, der Parameter tritt nur beim Multiple Reply auf. Das `m` bei `II` steht für mandatory, das ist ein Pflichtparameter, dieser ist also bei jeder Get-Operation dabei. Die nächsten beiden, `BC` und `BI`, wählen das oder die MOs für die Operation aus. Zur weiteren MO-Auswahl stehen die Parameter Scope, Filter und Synchronization zur Verfügung. Darauf werde ich gleich noch ausführlicher eingehen. Das `u` steht für User-optional. Der Nutzer entscheidet also über den Gebrauch des Parameters. Access control ist von OSI bisher nicht belegt worden. In der AI werden weitere Attribute übergeben. Die jetzt noch ausstehenden Parameter treten nur auf der response/confirm-Seite auf. `MC` und `MI` geben an, auf welche MOs zugegriffen wurde. `CT` gibt den Zeitpunkt der Übertragung an und AL beinhaltet die Werte von Attributen. Den Fehlerparameter habe ich schon weiter oben vorgestellt.

Als nächstes wollen wir uns mit den noch fehlenden Parametern Scope, Filter und Synchronization befassen. Alle drei sind als Hilfsmittel zur Auswahl von MOs gedacht.

Den Parameter Scope könnte man mit Wirkungsbereich übersetzen. Managed Objects werden durch Namen identifiziert und sind diesbezüglich hierarchisch in Form eines Baumes in der MIB angeordnet. Scoping funktioniert nun folgendermaßen: Man betrachtet ein MO innerhalb des Baumes, das als Basisobjekt definiert wird. Alle in der Hierarchie nachfolgenden Objekte bilden einen Teilbaum. Für den Teilbaum sind vier Wirkungsbereiche definiert:

- nur das Basisobjekt,
- die n-te Stufe des Teilbaumes vom Basisobjekt an

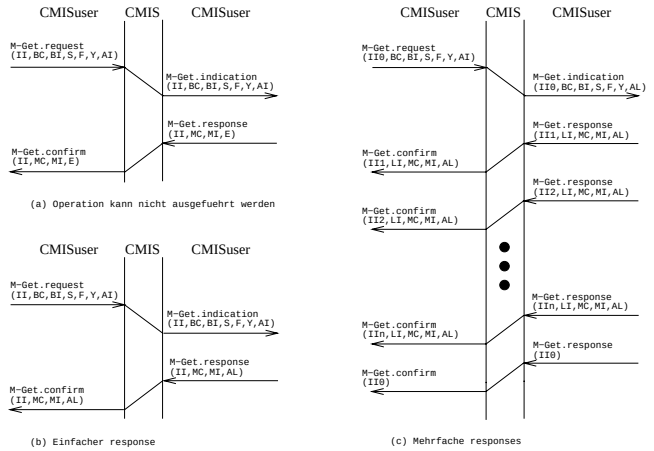


Abbildung 4.2: Weg/Zeit-Diagramm für M-Get-Service

(BO = 0-te Stufe),

- das Basisobjekt und alle nachfolgenden Objekte bis zu einer n-ten Stufe,
- das Basisobjekt und der gesamte Teilbaum.

Durch Auswahl des entsprechenden Parameters ist es dann möglich, eine gewiße Anzahl von MOs anzusprechen.

Um das Netz nicht unnötig zu belasten oder bestimmte MOs gezielt ansprechen zu können, wird ein Filtermechanismus eingeführt. Ein Filter ist ein boolescher Ausdruck, der aus einer oder mehrerer Forderungen über das Vorhandensein von Attributen oder deren aktuellen Wert besteht. Nur die MOs, die die angegebenen Bedingungen erfüllen, werden von den Diensten direkt angesprochen.

Es fehlt noch die Synchronization. Bei Zugriff auf mehrere MOs werden geeignete Synchronisationsbedingungen benötigt:

atomic: Alle für die Operation ausgewählten MOs werden untersucht, ob sie die Operation erfolgreich ausführen können. Wenn ein oder mehrere MOs das nicht können, wird die Operation von keinem ausgeführt.

best effort: Alle ausgewählten MOs werden ersucht, die Operation auszuführen.

M-Set-Service

Die Hauptaufgabe dieses Dienstes ist es, Daten in der MIB des Agenten zu modifizieren. Es können Werte von einem oder mehreren Attributen in einem oder mehreren MOs verändert werden. Dieser Dienst ist entweder bestätigt oder unbestätigt. Desweiteren ist Multiple Reply möglich. Probleme können bei M-Set auftauchen, wenn bei Auswahl von mehreren MOs diese nicht die gleiche Menge an Attributen haben. Dadurch können recht komplexe Operationen entstehen. Abhilfe können hier Mechanismen wie Filtering und Synchronization schaffen.

M-Action-Service

Der M-Action-Dienst erlaubt den Aufruf einer vordefinierten Action-Prozedur, als Teil eines MOs. Also die Ausführung einer Aktion von der Partnerinstanz, normalerweise des Agenten. Auch dieser Dienst kann bestätigt oder unbestätigt sein. Ebenfalls kennt M-Action das schon weiter oben angesprochene Multiple Reply.

M-Create-Service

Die Basisaufgabe dieses Dienstes ist es, in einer Objektklasse eine neue Instanz zu erzeugen, also praktisch ein neues MO. Dabei können die Attributwerte für die neue Instanz mit in den Request gegeben werden oder man bezieht sich auf eine schon existierende Instanz als Modell. Wichtig ist, daß dieser Dienst immer bestätigt ist.

M-Delete-Service

Dieser Dienst verhält sich komplementär zu M-Create. Er sorgt dafür, daß ein oder mehrere MOs von der MIB gelöscht werden. Auch dieser Dienst ist immer bestätigt. Zusätzlich kennt M-Delete auch Multiple Reply.

M-Cancel-Get-Service

Dieser Dienst ist eigentlich nur ein Hilfsdienst. Denn er hat nur die Aufgabe, sehr lange Get-Operationen zu stoppen. Seine Ausführung ist immer bestätigt.

4.2.3 Management-Association-Service

Die Dienste dieser Klasse kontrollieren die Anwendungsassoziationen zwischen den beteiligten Systemen. In erster Linie sind das Assoziationsaufbau und -abbau. Beteiligte Dienste sind:

M-INITIALIZE: Richtet eine Assoziation mit einem Partner-CMIS-Nutzer ein,

M-TERMINATE: Beendet eine Assoziation mit einem Partner auf normalem Wege,

M-ABORT: Bricht die Verbindung ab, wenn etwas Unvorhergesehenes passiert.

Alle beschriebenen Dienste greifen auf ACSE (Association Control Service Element) zu[3]. Dabei vereinbaren zwei Nutzer schon während des Aufbaus einer Assoziation, welche CMIS-Eigenschaften in dieser benutzt werden sollen. Diese Eigenschaften werden in Funktionseinheiten (functional units) erklärt.

4.3 Common Management Information Protocol (CMIP)

4.3.1 CMIP - Was ist das ?

CMIP ist das Protokoll, das CMIS-Dienste implementiert. Dabei werden von CMIP Prozeduren zur Übermittlung von Managementinformationen definiert. Dies passiert konkret in der Form von Protocol Data Units (PDUs), die zwischen zwei kooperierenden CMISEs ausgetauscht werden, um einen CMIS-Dienst auszuführen. CMIP benutzt die Dienste von ROSE (Remote Operation Service Element)[4]. ROSE und ACSE beziehen sich auf die Darstellungsschicht (Schicht 6) der OSI-Welt. CMIP kennt elf verschiedene PDU-Typen.

4.3.2 CMIP im Detail

Jetzt möchten wir zeigen, wie CMIP einen CMIS-Dienst implementiert. Als Beispiel wählen wir den schon voher ausführlich besprochenen Dienst M-Get (siehe Abbildung 4.3.2).

Der Manager schickt einen `M-Get.request` an seine CMIP-Machine (CMIPM) (1), das ist ein Stück Software, das den Request in PDUs umwandelt. Dabei bleiben die Parameter erhalten (2). CMIPM benutzt ROSE in Form eines `RO-INVOKE.requests`, um die PDUs weiterzuschicken (3). ROSE liefert die PDUs zur korrespondierenden CMIPM des Agenten in einem `RO-INVOKE.indication` (4). Sind die PDUs akzeptierbar, dann sckickt die CMIPM ein `M-Get.indication` zum Agenten, wobei die Ausgangsparameter unverändert übertragen werden (5). Danach führt der Agent die geforderte Operation aus und schickt ein `M-Get.response` an seine CMIPM mit den Ergebnisparametern (6). Jetzt wird wieder der Response in PDUs umgewandelt (7) und mit Hilfe von ROSE mit einem `RO-RESULT.request` zum anfragenden System zurückgeschickt (8). Die CMIPM des Managers empfängt die Daten in Form eines `RO-RESULT.indication` (9) und wandelt diese wiederum um und schickt sie mit einem M-Get.confirmation zurück zum Manager (10).

[3] siehe Abbildung 4.1
[4] siehe Abbildung 4.1

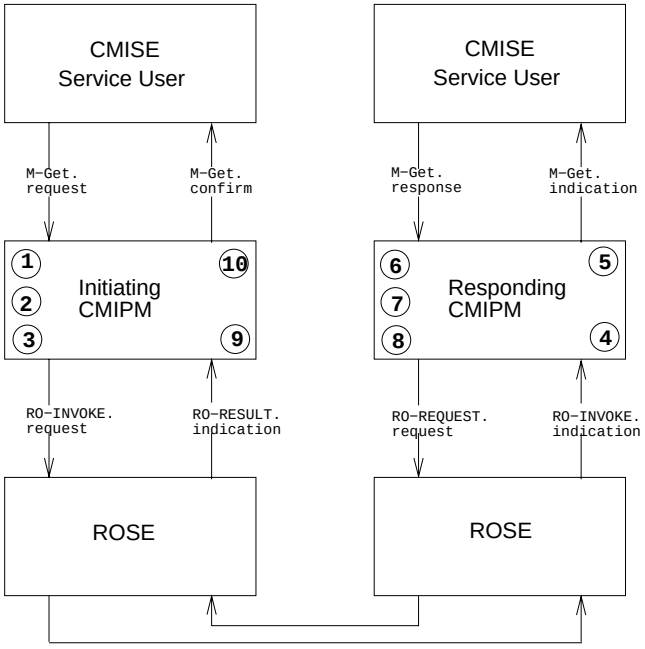


Abb. 3: CMIP Get Operation

4.4 Zusätze

4.4.1 Remote Operation Service Element (ROSE)

Die Dienste dieser Klasse sind alle Bestandteil der Systems Management Application Entity (SMAE), die in Abbildung 4.1 gezeigt wird. Der Basisdienst von ROSE ist der Aufruf einer Operation in einem abgesetzten, offenen System. ROSE definiert fünf Operationsklassen:

1. synchron mit Rückmeldung im Erfolgs- u. Fehlerfall,
2. asynchron mit Rückmeldung im Erfolgs- u. Fehlerfall,
3. asynchron mit Rückmeldung nur im Fehlerfall,
4. asynchron mit Rückmeldung nur im Erfolgsfall,
5. asynchron ohne Rückmeldung.

CMIP benutzt bei bestätigten CMIS-Operationen die Klassen 1 + 2. Bei unbestätigten Operationen wird die Klasse 5 benutzt. Weiterhin definiert ROSE drei Assoziationsklassen:

1. nur der Assoziationsinitiator ruft eine Operation auf,
2. nur der Assoziationsbeantworter ruft eine Operation auf,
3. beide können aufrufen (verbundene Operationen mit parent/child-Beziehung).

CMIP benutzt immer die Assoziationsklasse 3.

4.4.2 Association Control Service Element (ACSE)

Auch ACSE ist Bestandteil von SMAE. Diese Dienstklasse ist in erster Linie verantwortlich für den Auf- und Abbau einer Anwendungsassoziation. Darunter versteht man die logische Verbindung zwischen gleichberechtigten Anwendungen. ACSE benutzt folgende Dienstelemente:

- A-ASSOCIATE: baut die logische Verbindung zwischen gleichberechtigten Anwendungen auf,
- A-RELEASE: beendet die Verbindung sanft,
- A-ABORT: bricht die Verbindung unsanft ab.

4.5 Zusammenfassung

CMIS stellt einen Basisdienst zum Managementinformationsaustausch zwischen Systemmanagementfunktionen zur Verfügung. CMIS besteht aus den Primitiven: M-EVENT-REPORT, M-GET, M-SET, M-ACTION, M-CREATE, M-DELETE und M-CANCEL-GET. Zur MO-Auswahl stehen zur Verfügung: Scoping, Filtering, Synchronization. CMIP implementiert die Dienste von CMIS. CMIP besteht aus einer Menge von PDUs und benutzt ACSE, ROSE und die Darstellungsschicht.

4.6 Literaturverzeichnis

[LF89b] [Sta93a] [Ker92]

Kapitel 5

OSI Systemmanagementfunktionen (Teil 1)

Sonia Pascual Romero

5.1 Einleitung

5.1.1 System Management Functional Areas

ISO (International Organisation for Standardization) hat die Anforderung, die an das Mangement gestellt wurde, in Funktionsbereiche, den sogenannten System management functional areas (SMFAs) zusammengefaßt. Managementaktivitäten werden so verschiedenen Funktionsbereichen zugeordnet. OSI (Open Systems Interconnection) kennt fünf solche Funktionsbereiche:

- Abrechnung
- Leistung
- Fehler
- Sicherheit
- Konfiguration

5.1.2 Abrechnungsmanagement (Accounting Management)

Ziel des Abrechnungsmangement ist es, Kosten zu sparen: deswegen muß Buch geführt werden über sämtliche Dienstleistungen. Dazu gehört die Benutzerverwaltung, Festlegung und Abrechnen von MO-Nutzung und Betriebsmittelverbrauch.

5.1.3 Leistungsmanagement (Performance Management)

Hier werden statische Daten, wie zum Beispiel Durchsatz, Anzahl von Übertragungsfehlern gesammelt und ausgewertet. Die Ergebnisse dienen zur Verbesserung des Leistungsverhalten von Betriebsmitteln.

5.1.4 Fehlermanagement (Fault Management)

Fehlermanagement ist zuständig für die Erkennung, Lokalisierung und Behebung von Störungen und Fehlern. Weiterhin werden Ereignisse protokolliert, die in irgendeiner Weise mit Fehlern zu tun haben könnten, um diese möglichst früh zu erkennen.

5.1.5 Sicherheitsmanagement (Security Management)

Durch Authentisierung, Zugangsüberwachung und Schlüsselverwaltung wird versucht, illegale Zugriffe möglichst früh zu erkennen und zu verhindern.

5.1.6 Konfigurationsmanagement (Configuration Management)

Betriebsmittel / MOs werden definiert, benannt, initialisiert oder beendet. MO-Attribute werden eingestellt oder verändert. Zustandsdaten werden gesammelt. All diese Daten werden benötigt zur Planung des Gesamtsystems, wie z.B. Erweiterung des Systems durch neue Rechner.

5.2 Systemmanagementfunktionen (System Management Functions)

Auf diesen SMFAs wurden eine Reihe von System Management Functions (SMFs) entwickelt, die als allgemeine Netzwerkmanagement-Hilfsfunktionen den Managementanwendungen zur Verfügung stehen. Die Aufgaben der SMFA werden im System Mangement Application Process (SMAP) wahrgenommen. Der SMAP hat die Möglichkeit, die Parameter aller Instanzen in allen Schichten zu verändern. Über die Common Management Information Protocols kann der SMAP mit anderen Prozessen kommunizieren. Alle sieben Schichten des ISO/OSI-Modells können auf diese Funktionen, den SMFs, zugreifen.

Im folgenden wird nur auf einen Teil dieser Funktionen näher eingegangen:

- Objekt-Management Function

- State-Management Function
- Relationship-Management Function
- Alarm-Reporting Function
- Event-Reporting Function
- Log-Control Function
- Security-Alarm-Reporting Function
- Security-Audit-Trail Function

5.2.1 Object-Management Function

Object-Management Function behandelt im Wesentlichen mit der Verwaltung von MOs, d.h. gehört zum Bereich des Configuration Management. MOs können erzeugt und gelöscht, die Werte ihrer Attribute können gelesen oder verändert werden. Änderung des Namens eines MOs oder dessen Attribut können gemeldet werden, sowie die Erzeugung und das Löschen eines solchen. Aktionen der MOs können gestartet werden und dessen Meldungen werden übertragen.

Es gibt neun Dienste die von der Object Management Function unterstützt werden. Bei sechs davon handelt es sich um Pass-Through Services, d. h. die Dienste werden nicht direkt von der Funktion zur Verfügung gestellt, sondern an untere Schichten weitergegeben. Dies sind:

- PT-CREATE: Erzeugen eines neuen Objektes
- PT-DELETE: Löschen eines Objektes
- PT-ACTION: Ausführen einer Aktion auf einem oder mehreren Objekten
- PT-SET: Modifikation eines Attributwertes durch einen Dienstnehmer
- PT-GET: Wiedererlangen eines Attributes von einem Dienstnehmer
- PT-EVENT-REPORT: Berichtet einem Dienstnehmer ein Ereignis

Die übrigen drei Dienste sind Direct Services, diese benutzen zwar CMIS Dienste, fügen aber noch eigene Werte hinzu:

- Object-creation reporting: Meldet das Erzeugen eines MOs
- Object-deletion reporting: Meldet das Löschen eines MOs.
- Attribute-value-change-reporting: Meldet Änderungen an Attributen.

5.2.2 State Management Function

Stellt ein allgemeines Modell für die Beschreibung des Managementzustands eines MO dar. Das Modell erlaubt dem Nutzer die Abfrage und Modifikation des Management-Zustandes eines MO und Meldungen von Zustandsänderungen zu empfangen, die nicht durch Management-Aktivitäten hervorgerufen worden sind. Zur Beschreibung des Managementzustands gibt es zum einen State Attribute und zum anderen Status Attribute.

State Attribute

Der Management Zustand eines MOs gibt aus der Sicht des Management an, ob eine Operation auf diesem MO ausführbar oder ob es überhaupt verfügbar ist, beispielsweise welchen Zustand ein Gerät wie der Drucker einnehmen kann. Entsprechend der drei wichtigsten Faktoren, die sich auf den Zustand eines Objekts auswirken, sind drei Zustandsdiagramme definiert worden.

1. Ausführbarkeit
2. Anwendung
3. Verwaltung/Organisation

In jedem dieser Gebiete gibt es ein Zustandsattribut, dessen Wert der laufende Zustand des Objektes ist, entsprechend dem Gebiet.

Ausführbarkeit hat die Zustände:

- belegt
- freigegeben

Anwendung hat die Zustände:

- inaktiv
- aktiv
- belegt

Hier werden teilbare und unteilbare Objekte unterschieden. Ein unteilbares Objekt ist zum Beispiel ein Drucker, wenn dieser inaktiv war und nun belegt wird, gerät er sofort in den Zustand inaktiv, da kein weiterer Benutzer zur selben Zeit den Drucker belegen darf. Ein teilbares Objekt ist ein Speicher auf dem mehrere Nutzer arbeiten können, so daß bei belegen eines Speichers zunächst der Zustand aktiv eingenommen wird. Nun können noch weitere Benutzer belegen, bis die Kapazität erschöpft ist.

Verwaltung/Organisation hat die Zustände:

- gesperrt
- außer Betrieb nehmen
- frei verfügber

Ein Beispiel für den "außer Betrieb nehmen"-Zustand wäre ein Netzwerk an dem mehrere Benutzer an verschiedenen Rechnern arbeiten. Das System soll runtergefahren werden, bevor es aber nun in den "gesperrt"-Zustand übergehen kann, müssen sich erst alle Benutzer des Systems ausloggen, das wäre der "außer Betrieb nehmen"-Zustand.

Status Attibutes

Zusätzlich zu den State Attributes gibt es die Status Attributes, die detailliertere Informationen über den Zustand des Objektes geben.

- Alarm status: Stellt eine Anzeige zur Verfügung von der Anwesenheit verschiedener Alarme. Kann die Werte annehmen
  - under repair: Wird gerade repariert
  - critical: Einer oder mehrere Fehleralarme wurden gemeldet und sind noch nicht geklärt worden.
  - major: Ein oder mehrere größere Fehler wurden gemeldet, sind aber noch nicht geklärt.
  - minor: Ein oder mehrere kleinere Fehler wurden gemeldet, sind aber noch nicht geklärt.
  - Alarm outstanding: Ein oder mehr Fehler wurden gemeldet.
- Procedural status unterstützt nur jene Objektklassen, die Prozeduren darstellen, welche eine Anzahl von Phasen durchlaufen müssen, um ausgeführt zu werden. Kann folgende Werte annehmen
  - Initialization required: Initialisierung durch den Manager notwendig, Initialisierungsprozedur wurde nicht aufgerufen.
  - not initialized: Autonome Initialisierung notwendig, Initialisierungsprozedur wurde nicht aufgerufen.
  - Initialized: Initialisierungsprozedur wurde aufgerufen, ist aber noch nicht beendet.
  - Reporting: Objekt hat eine Operation ausgeführt und meldet die Ergebnisse.
  - Terminating: Objekt ist in der Endphase.
- Availability status gibt den Ausführbarkeitszustand eines Objektes an. Zustände:
  - In test: Mittel unterliegt einer Testprozedur.
  - Failed: Fehler vorhanden, der die Durchführbarkeit verhindert.
  - Power off: Objekt ist ausgeschaltet.
  - Off-line: Objekt ist an, aber außer Betrieb.
  - Off duty: Objekt wurde durch einen internen Kontrollprozeß inaktiviert.
  - Dependency: Objekt kann nicht arbeiten, da notwendiges Mittel nicht verfügbar ist.
  - Degraded: Reduziert, Komponenten nicht komplett verfügbar.
  - Not installed: Objekt ist nicht vorhanden.
  - Log full: Bedingung ist aufgetreten, die wertvoll genug ist, daß sie protokolliert wird.
- Controll status informiert detaillierter über den Verwaltungszustand eines Objektes. Zustände:
  - Subject to test: Objekt ist verfügbar, es sollte aber noch getestet werden.
  - Part of services locked: Teile der Dieste sind durch die Verwaltung eingeschränkt.
  - Reserved for test: Objekt ist nicht verfügbar, da es gerade einer Testprozedur unterliegt.
  - Suspended: Dienst ist für Nutzer des Objekts eingestellt.
- Standby status gibt den Uebernahmezustand an und enthält die Zustände:
  - Hot standby: Komponente ist bereit zur sofortigen Übernahme
  - Cold standby: Zur Übernahme nur unter Vorbehalt bereit, d. h. nicht synchron.
  - Providing service: Diensterbringend
- Unknown status beschreibt, ob der Zustand des Objekts bekannt ist oder nicht. Möglicher Zustand ist:
  - True: Zustand des Objektes ist unbekannt
  - False: Zustand ist bekannt.

State-Management Services

Die State-Management Function unterstützt folgende Dienste:

- das Berichten der Änderungen im State Attribute
- Lesen des State Attribute
- Ändern des State Attribute

Das Berichten der Änderungen im State Attribute benutzt den CMIS Befehl M-EVENT-REPORT. Der Ereignisinformationsparameter enthält eine Reihe von Elementen, wobei aber nur zwei obligatorisch sind: Bei der Zustandswechseldefinition:

- Attribut-identifier
- new attribute value

Die weiteren Element sind freiwillig wie zum Beispiel die Quellenbezeichnung.

5.2.3 Relationship-Management Function

Organisiert Beziehungen, die zwischen den MOs existieren können. Dienste sind definiert zum Gründen, Prüfen und Abfragen der Beziehungen zwischen den Objekten, so daß es möglich ist einen Überblick zu gewinnen, welche Abhängigkeiten zwischen Teilen des Systems herrschen.

Eine Beziehung wird durch eine Menge von Regeln beschrieben, die angeben, wie sich eine Operation eines MOs auf die Operationen eines anderen MOs auswirken.

Relationship Model

Beziehungen sind durch Beziehungsattribute definiert. Es gibt verschiedene Typen von Beziehungen, die jeweils eigene Beziehungsattribute haben (role). Ein Beziehungsattribut ist ein Mengenwertattribut, dessen Werte die Namen anderer MOs sind, mit denen das Objekt in Beziehung steht. Es werden $1:1$, $1:n$, $n:1$ und $m:n$ Beziehungen unterstützt.

Drei Kategorien von Beziehungen sind von OSI anerkannt:

- Containment relationship: Ein MO enthält ein anderes MO, ohne das es nicht existieren kann.
- Reciprocal relationship: Dies ist eine Verbindung zwischen zwei Objekten. Um diese Verbindung darzustellen, enthält jedes Objekt ein Attribut dessen Wert der Name des anderen Objektes ist (Bild **??**).
- One-way relationship: Dies ist eine asymmetrische Beziehung zwischen zwei Objekten, die nur im Beziehungsattribut eines Objekts dargestellt wird.

Beziehungstypen

Beziehungen werden in fünf verschiedene Klassen eingeteilt:

- Service relationship: Hierbei handelt es sich um eine asymtrische Beziehung, wobei eines der Objekte die Dienstgeberrolle und das andere Objekt die Nutzerrolle hat.
- Peer relationship: Ist eine symmetrische Beziehung zwischen seinesgleichen.
- Fallback relationship: Redundanzbeziehung, d. h. es gibt ein zweites ausgesuchtes Objekt, das die Rolle des Ersten übernehmen kann, falls dieses nicht zur Verfügung steht.
- Backup Relationship: Echte Redundanzbeziehung, eine Ersatzbeziehung.
- Group relationship: Beziehung zwischen zwei Objekten, wobei ein Objekt (member role) zu einer Grupppe gehört, die durch das andere Objekt dargestellt wird (owner role).

Relationship-Management Services

Der relationship-change-reporting Dienst gibt einem Nutzer die Möglichkeit über die Änderungen des Managed-object Relationship Attribut zu berichten. Der Dienst benutzt den CMIS Befehl M-EVENT-REPORT: Der Ereignisinformationsparameter enthält wie beim state management service eine Reihe freiwilliger und obligatorischer Elemente. Auch hier sind die obligatorischen Elemente:

- Attribute identifier und
- New attribute value.

5.2.4 Alarm-Reporting Function

Fehler sollen erkannt werden bevor sie sich auf das Netz auswirken. Komponenten des Netzwerkes übertragen Fehlermeldungen an ihre Manager. Die Definition dieser Meldungen ist Aufgabe der Alarm Reporting Function. Fehlerbenachrichtigungen beziehen sich hauptsächlich auf Verwaltungsfehler. Die Benachrichtigung beinhaltet Fehlertypen und mögliche Gründe.

Alarmdefinition

Ein Alarm ist eine spezielle Benachrichtigung über einen aufgetretenen Fehler. Welche Information ein Alarm beinhaltet ist nicht vorgegeben, es ist vielmehr dem MO-Entwickler überlassen, welche er bei Fehlermeldungen einfügen möchte.

Es sind fünf Kategorien definiert:

1. Kommunikationsfehler
2. Qualität eines Dienstes
3. Vearbeitungsfehler (Fehler innerhalb der Software)
4. Ausstattungsfehler (Komponentenfehler)
5. Umgebungsfehler (z.B Lüftung eines Rechners)

Mögliche Gründe für Alarme sind:

- Verlust eines Signals
- Übermittlungsfehler
- Antwortzeit ist zu lang
- Bandbreite ist reduziert
- Softwarefehler
- Zeitproblem
- Prozessorproblem
- Feuer
- Lüftungsproblem, usw.

Es folgt ein Ausschnitt der wichtigsten Attribute:

- möglicher Grund

- angenommener Schwierigkeitsgrad, es wurden sechs Stufen definiert:
  1. kritisch
  2. schwerwiegend
  3. weniger schwerwiegend
  4. Warnung
  5. unentscheidbar
  6. geklärt
- Spezifikation des Problems

Alarm-Reporting Service

Alarm-Reporting Service gibt einem Nutzer die Möglichkeit einen anderen Nutzer über einen Alarm zu informieren. Es wird der CMIS Befehl M-EVENT-REPORT verwendet. Ein Ereignistypparameter bezeichnet den Typ des Alarms. Der Alarminformationsparameter enthält eine Reihe obligatorischer, freiwilliger, bedingter Elemente. Obligatorisch sind:

- mögliche Gründe
- angenommener Schwierigkeitsgrad

Ein bedingtes Element wäre zum Beispiel Grenzwertinformation. Wenn kein Grenzwert vorhanden ist, kann hier auch keiner angegeben werden.

5.2.5 Event-Reporting-Management Function

Ziel der Event-Reporting-Management Funcion ist es, Meldungen über Ereignisse im Netz zu erhalten. Aufgaben:

- Die zu übertragenden Meldungen beschreiben und diese Beschreibung auch während des Betriebs ändern zu können.
- Das Ziel der Meldungen angeben zu können.
- Die Weiterleitung von Meldungen auch temporär unterbinden zu können.

Damit eine Meldung zu einem bestimmten System weitergeleitet wird, muß diese bestimmte Bedingungen erfüllen. Diese Bedingungen sind im Event Forwarding Discriminator gespeichert, um sie hier zu überprüfen.

Modell des Event Reporting Management

Folgendes Bild zeigt das konzeptionelle Modell des Event Report Management:

Die MOs sind in der Lage, Nachrichten zu erzeugen und zu senden. Diese Nachrichten werden von einer lokalen Instanz Event Detection and Processing empfangen. Die Event Detection and Processing formt die Nachrichten in Ereignisberichte um, indem sie z.B. zur schon vorhandenen Information die Ereigniszeit und Objektklasse hinzufügt. Diese Ereignisse werden an alle Event Forwarding Discriminators verteilt, wo sie an die angegeben Zieladressen weitergeleitet werden, wenn sie angegebenen Bedingungen erfüllt. Der Event Forwarding Discriminator wirkt wie ein Filtermechanismus. Der Diskriminator besteht somit aus Booleanausdrücken, die entscheiden über welche Ereignisse berichtet werden soll. Es werden beispielweise folgende Attribute getestet:

- MO-Klasse
- MO-Instanz
- Ereignistyp
- spezielle Ereignistypattribute, z. B. der Schweregrad einer Fehlermeldung.
- Startzeit
- Endzeit

Event-Report-Mangement Sevices

- Erzeugen eines Diskriminators
- Löschen eines Diskriminators
- Verändern der Diskriminatorattribute
- Einstellen einer Diskriminatoraktivität
- Wiederaufnahme einer Diskriminatoraktivität

5.2.6 Log-Control Function

Ähnlich wie bei der Verwaltung im Alltag, wie z. B. Haushaltsverwaltung, ist es manchmal notwendig gewisse Ereignisse über längere Zeit hinweg zu speichern. Aufgabe der Log-Control Funktion ist es, Auswahlkriterien zu setzen oder zu verändern, die entscheiden, ob ein Ereignis gespeichert werden soll. Es ähnelt einem Filter, der entscheidet, was aufgezeichnet wird oder nicht. Der Speicher, der diese Auzeichununngen enthält, wird als MO dargestellt, das Log-Objekt, die einzelnen Einträge in diesem Speicher sind die Log-Records. Die Log-Control Funktion erfüllt folgende Punkte:

- Definition eines Log-Control Dienstes, der eine Auswahl trifft über die Aufzeichnungen.
- Die Möglichkeit eines externen Systems das Kriterium, welche Ereignisse aufgezeichnet werden sollen, zu verändern.
- Die Möglichkeit für externe Systeme zu bestimmen, ob Merkmale verändert wurden oder Aufzeichnungen verloren gegangen sind.
- Spezifikation eines Mechanismus, der die Zeit der Aufzeichnungen kontrolliert.
- Möglichkeit für ein externes Systems Aufzeichnungen zu löschen und wiederzuerlangen.

- Möglichkeit für ein externes System Log-Objects zu erzeugen und zu löschen.

Jedes Log-Object besteht aus einer Menge von Records, für jedes Ereignis eins. Jede aufgezeichnete Information wird als Ereignisbericht empfangen und unterstützt durch die Event-Report Funktion. Schlüsselattributes Log-Objects sind

- zeitliche Information, wann eine Aufzeichnung vorgenommen wurde
- Diskriminatorkonstrukt

Weitere Attribute sind

- Log ID
- Zustand
- maximale Aufzeichnungsgröße
- aktuelle Aufzeichnungsgröße
- Verhalten bei Überlauf: zwei Möglichkeiten
  - Stop
  - Älteste Aufzeichnung wird Überschrieben (Wrap-Around)
- Alarm bei Schwellenüberschreitung der Kapazität
- Benachrichtigungen: bei Erzeugen, Löschen, Einstellen, Wiederaufnehmen und Verändern.

Log-Control services

Die Log-Control Funktion unterstützt folgende Dienste:

- Erzeugen eines Log-Objects
- Löschen eines Log-Objects
- Verändern eines Log-Objects
- Einstellen einer Log Aktivität
- Löschen eines Log-Records
- Wiedererlangen eines Log-Records
- Wiederaufnahme einer Log Aktivität

5.2.7 Security-Alarm-Reporting Function

Die Security-Alarm Function berichtet über sicherheitsrelevante Ereignisse und Fehloperationen in Sicherheitsdiensten. Sie unterstützt Dienste zum

- Erzeugen
- Löschen
- Verändern des Diskriminators,

der entscheidet, welche Sicherheitsalarme wohin gesendet werden.

Hauptaufgabe der Security-Alarm Function besteht darin, den Nutzer über Angriffe auf die Systemsicherheit zu warnen.

Sicherheitsalarme

Sicherheitsbezogene Ereignisse werden über einen Diskriminator ausgewählt. Diese Ereignisse werden Sicherheitsalarme genannt und werden in fünf Typen eingeteilt:

1. Integrity violation: Unterbrechung eines Informationsflusses, d. h. illegales Verändern, Hinzufügen oder Löschen einer Information. Möglicher Grund wäre z. B. fehlende Information.
2. Operational violation: die Bereitstellung eines gewünschten Dienstes war nicht möglich aufgrund von Nichtverfügbarkeit, Fehlfunktion oder inkorrekter Aufruf des Dienstes.
3. Physical violation: Lücke der physikalischen Mittel.
4. Security-service or mechanism violation: Angriff auf Sicherheitsdienst oder Sicherheitsmechanismus. Ein Beispiel wäre Versuch eines Zugriffs von nicht autorisiertem Nutzer.
5. Time-domain violation: es fand ein Ereignis außerhalb der erlaubten Zeitspanne statt. Z. B. eine verspätete Information.

5.2.8 Security-Audit-Trail Function

Die Security-Audit-Trail Function ist im Grunde eine Verfeinerung der oben schon erklärten Log-Control Function, d. h. sie zeichnet Ereignisse auf, wobei aus Sicherheitsgründen sämtlich Diensterbringungen und Dienstverweigerungen aufgezeichnet werden. Sicherheitsalarme sind hier in feinere Klassen unterteilt:

- Authentifizierungsfehler
- Vertrauensverstoß
- physikalischer Fehler
- verspätete Information
- Ablehnung eines Dienstes
- doppelte Information
- Informationsverlust
- Information wurde geändert
- Nicht-Zurückweisung einer Nachricht durch nichtauthorisierten Benutzer
- Information nicht in geordneter Reihenfolge
- Störung in der Ausrüstung
- falscher Schlüssel wurde benutzt
- Aktivität, die außerhalb der erwarteten Zeit stattfindet
- außer Betrieb
- Prozedurfehler

5.3 Anmerkung

Die bisher vorgestellten Systemmanagementfunktionen sind von OSI sehr ausführlich ausgearbeitet worden, vor allem was die Sicherheit des Systems anbelangt. Diese Ausführlichkeit bringt aber eine hohe Komplexität mit sich, die die Übersichtlichkeit der Aufgaben/Dienste der Funktionen enorm einschränkt. Sinnvoller wäre es gewesen einige Funktionen, wie z. B. Log-Control Funktion und Audit-Trail Funktion zusammenzufassen um eine vor allem kompaktere Darstellung der angebotenen Dienste zu erreichen.

5.4 Literaturverzeichnis

[Sta93j]

Kapitel 6

OSI Systemmanagementfunktionen (Teil 2)

Gerhard Schäl

6.1 Einleitung

In diesem Kapitel möchte ich 5 weitere OSI Systemmanagementfunktionen vorstellen. Im einzelnen handelt es sich um folgende:

- Access-Control-Management Function:
  Dient der Zugangskontrolle zu Managementinformationen und -operationen.
- Accounting-Meter Function:
  Dient der Buchführung von Ressourcenbenutzungsdaten und der Festsetzung von Ressourcenbenutzungslimits.
- Workload-Monitoring Function
  Dient der Anzeige der Systemleistung, zur Vermeidung von Ressourcenüberlastungen.
- Test-Management Function
  Dient der Durchführung von Diagnose- und Vertrauenstests entfernter Systeme.
- Summarization Function
  Dient der Ausführung und Zusammenfassung statistischer Analysen auf Verwaltungsinformationen.

6.2 Access-Control-Management Function

X.741/ISO 10164-9 spezifiziert MOs[1] und Attribute, die benutzt werden, um den Zugriff, gemäß der durch die Zugriffskontrollverwaltungsinformation repräsentierten Zugriffskontrolltaktik, zu gewähren bzw. zu verweigern.

Die Ziele der Zugangskontrolle sind die folgenden:

- Vermeidung der Versendung von Verwaltungsmeldungen an nicht autorisierte Empfänger
- Zugriffsschutz der Managementoperationen vor nicht autorisierten Aufrufern,
- Schutz der Managementinformationen vor nicht vorgesehener Veröffentlichung

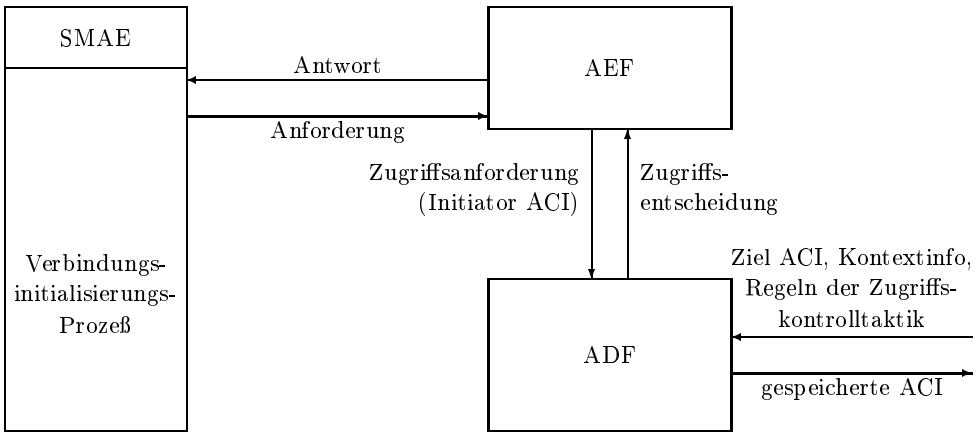
[1] MO = Managed Object

6.2.1 Das Zugriffskontrollmodell

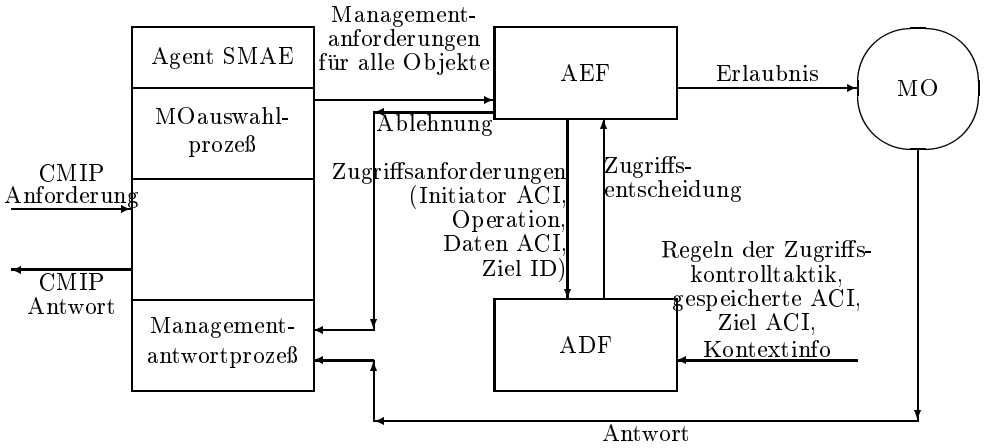
Eine Zugriffsanforderung auf ein Objekt wird durch eine Zugriffskontrollfunktion, die gemäß einer Zugriffskontrolltaktik den Zugriff gewährt oder verweigert, gerechtfertigt. Für drei Zugriffsarten möchte ich die Kontrollschritte erläutern:

1. Zugriffskontrolle bei Einrichtung einer Verwaltungsverbindung [Abb.6.1.(a)]
   (a) Ein AP[2] übergibt seine Anforderung der AEF[3].
   (b) AEF gibt die ACI[4] an die ADF[5] weiter.
   (c) Die ADF vergleicht die ankommende ACI mit der Target ACI. prüft Zusammenhangsinformationen und die entsprechenden Regeln der Zugriffskontrolltaktik.
   (d) Danach sendet die ADF die Zugriffsentscheidung an die AEF zurück.
   (e) Bei einer positiven Entscheidung wird die ACI für zukünftige Entscheidungen in der AEF gespeichert und die AEF erteilt dem AP den Zugriff.
2. Zugriffskontrolle beim Zugriff auf Verwaltungsoperationen [Abb.6.1.(b)]
   (a) Eine Zugriffsanforderung auf ein MO, ausgelöst durch eine CMIP Aktion führt zu einer Verwaltungsanforderung bei der AEF.
   (b) Die AEF unterstützt die ADF bei der Entscheidungsfindung durch die Übermittlung folgender Informationen:

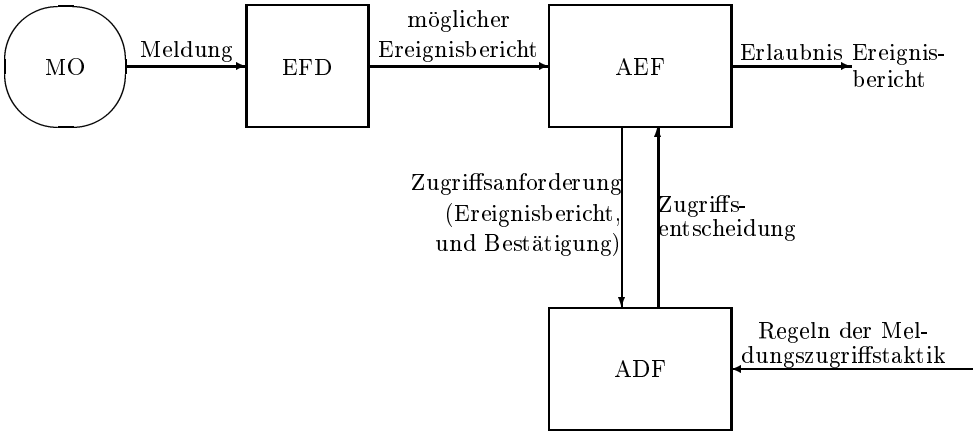
[2] AP = association process
[3] AEF = Access-Control-Enforcement Function
[4] ACI = Access-Control Information
[5] ADF = Access-Decision Function



(a) Zugriffskontrolle bei Verwaltungsverbindungen



(b) Zugriffskontrolle bei Managementoperationen



(c) Zugriffskontrolle bei einer Managementmeldung

Abbildung 6.1: Zugriffskontrollmodelle

- ACI des Initiators
- die Verwaltungsoperation (action, create, delete, get, set)
- ACI der Daten
- Identifikation des Targets (Objektklasse und Instanz, Aktionsidentifikator, Attributsidentifikator).

(c) Die ADF prüft die gespeicherten ACIs der Verbindung, die die Anforderung erzeugt hat, die ACI des Targets, Zusammenhangsinformationen und die Regeln der Zugriffskontrolltaktik.

(d) Bei Gewährung erlaubt die AEF der Verwaltungsanforderung den Zugriff auf das entsprechende MO.

3. Zugriffskontrolle bei Verwaltungsmeldungen [Abb.6.1.(c)]

(a) Eine Meldung wird von einem MO an einen EFD[6] geleitet.

(b) Hat eine Meldung den Filter durchlaufen, wird ein Bericht mit möglichen Ereignissen erzeugt und an die AEF geschickt.

(c) Die AEF unterstützt die ADF bei der Entscheidungsfindung durch Übersendung der angeforderten Informationen, bestehend aus Ereignisbericht des Filters und vorgesehener Bestimmung des Berichts.

(d) Die ADF prüft diese Informationen und die Regeln der Meldungszugriffstaktik.

(e) Bei Gewährung erlaubt die AEF die Aussendung der Meldung.

6.2.2 Zugangskontrollmechanismen

Die Zugriffskontrollverwaltungsfunktion unterstützt den Gebrauch folgender Zugangskontrollmechanismen:

- Zugriffskontrolllisten
- Fähigkeitstickets
- Sicherheitsmarken

Zugriffskontrolllisten und Fähigkeitstickets ergeben sich aus der Zugriffsmatrix. In dieser Matrix sind auf der einen Achse die Initiatoren[7] und auf der anderen Achse die Targets[8] aufgelistet. Im gemeinsamen Feld von einem Initiator und einem Target sind die Zugriffsrechte des Initiators auf das entsprechende Target eingetragen. Die Zugriffskontrolliste listet nun für jedes Target die Initiatoren mit den zugehörigen Zugriffsrechten auf. Die Liste kann einen Defaulteingang für jedermann mit beschränkten Zugriffsrechten (z.B. nur lesen) haben. Weitere Listenelemente können einzelne Benutzer oder Benutzergruppen sein.

Ein Fähigkeitsticket spezifiziert für jeden Initiator Ziele mit den erlaubten Operationen. Jeder Benutzer hat mehrere Tickets. Er kann sich Tickets borgen, oder selbst welch verleihen. Dadurch können sich Tickets über das gesammte Netz verteilen. Man hat somit keine Kontrolle darüber, wer nun welche Zugriffsrechte auf bestimmte Targets hat. Sie stellen ein größeres Sicherheitsproblem als Zugriffskontrollisten dar.

Der dritte Mechanismus ist der der Sicherheitsmarken. Dabei bekommt jedes Ziel einen classification level und jeder Initiator einen clearance level. Anwendungsgebiet sind hoch sichere Systeme, z.B. im militärischen Bereich. Mögliche Einstufungen der Levels ist z.B. öffentlich, vertraulich, geheim, streng geheim. Die Zugangskontrolltaktik muß dabei dem clearance level den entsprechenden classification level zuordnen. Das geschieht nach folgenden Regeln:

- No read up
  Ein Subjekt kann nur Objekte lesen, deren classification level kleiner oder gleich dem clearance level des Subjekts ist.
- No write down
  Ein Subjekt kann nur Objekte beschreiben, deren classification level kleiner oder gleich dem clearance level des Subjekts sind.

6.2.3 Zugangskontrollobjekte

Ein Zielvorhaben der Zugriffskontrollverwaltungsfunktion ist die Trennung der Elemente der Verwaltungsinformation von der Spezifikation der Mechanismen, die zum Schutz der Verwaltungsinformation benutzt werden. Dazu ist die Zugriffskontrollinformation als Menge von MOs modelliert. Diese MOs werden vom Zugangskontrollmechanismus benutzt, um den Zugang zur Verwaltungsinformation des Systems zu regeln. Folgende MO-Klassen werden in der Zugriffskontrollverwaltungsfunktion benutzt:

- Objekte der Zugriffskontrolltaktik
- Zielobjekte
- Objekte autorisierter Initiatoren

Abbildung 6.2 zeigt die Beziehungen zwischen den MO-Klassen.

Zum Verständnis der 3 MO-Klassen und ihrer Beziehungen untereinander wird definiert:

Definition 6.1

1. *Security policy: Eine Menge von Kriterien oder Regeln, zur Einhaltung der Sicherheitsanforderungen.*
2. *Security domain: Eine Menge von Elementen, die unter einer security policy für bestimmte Klassen sicherheitsrelevanter Aktivitäten, von einer einzigen Autorität verwaltet wird.*

[6] EFD = Event-Forwarding Discriminator (ein Filter)

[7] Initiator = Wesen mit Zugangsfähigkeit zu einem Target, z.B. ein Benutzer, eine Gruppe von Benutzern, Prozesse etc.

[8] der Zugang zu einem Target soll kontrolliert werden. Als Target kommen Dateien, Mengen von Dateien, Programme, Speichersegmente etc in Frage.

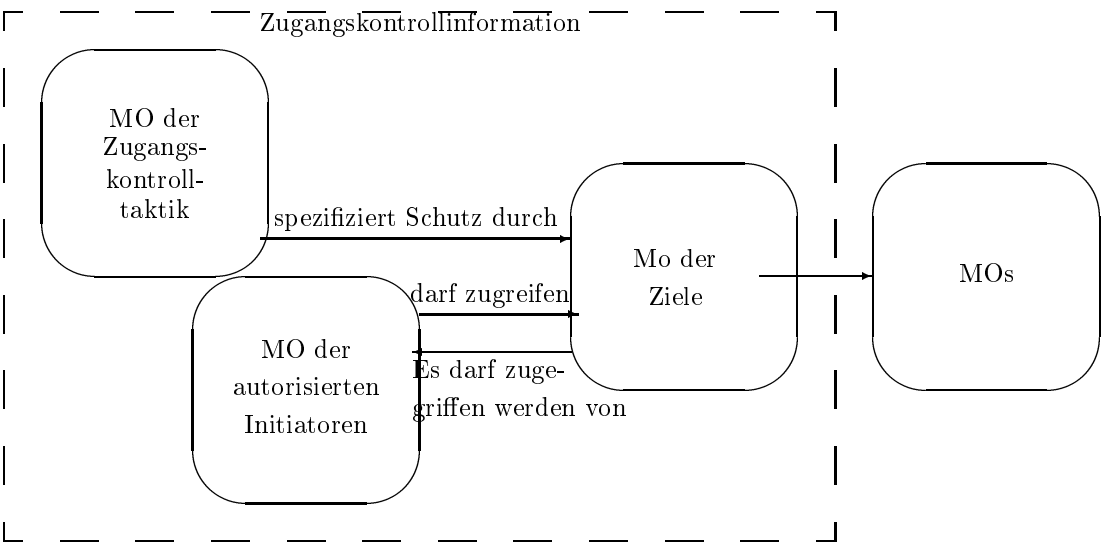


Abbildung 6.2: Beziehungen zwischen verwalteten Objekten

Jedes verwaltete offene System hat mindestens eine domain. Um Sicherheit für verschiedene Mengen zu beschreiben kann ein System mehrere Domains haben.

Die security policy beschreibt die Bedeutung der Sicherheit im Zusammenhang mit der domain, die Sicherheitsregeln für die Elemente in der domain und die Aktivitätsklassen, auf die sich die Regeln beziehen. Mit jeder security domain ist ein access-control-policy object verbunden. Dieses Objekt identifiziert die Regeln und Typen der Zugangskontrollmechanismen, die zur domain gehören. Die domain wird durch eine Menge von targets objects spezifiziert, die im *access-control-policy object* enthalten ist. Jedes targets object identifiziert MOs, auf die sich die Zugangskontrolle bezieht und spezifiziert die Initiatoren, die auf die geschützten Objekte durch einen Zeiger auf die Liste autorisierter Initiatoren (authorized-initiators object) zugreifen. Die in einem access-control-policy object enthaltene Menge von Zielobjekten definieren die Menge der Objekte, die unter die zugehörige security domain fallen. Das access-control-policy object enthält ebenfalls eine Menge von authorized-initiator objects. Jedes authorized-initiator object identifiziert dieZugriffsrechte einer Initiatormenge und enthält Zeiger auf eine Liste mit den Targets, auf die zugegriffen werden darf.

access-control-policy objects

Eine Zugangskontrolltaktik kann folgende Informationen enthalten:

- Zugriffsregeln:
  Definieren die Taktik bei
  - der Einrichtung von Verwaltungsverbindungen,
  - der Kontrolle von Verwaltungsmeldungen
  - der Ausführung von Operationen auf MOs.
- den Namen der security domain
- die Identität der Autorität in der security domain
- Informationen zur Rechtsgültigkeit
- Schutzinformationen

Die Taktitk spezifiziert die Art der Zugriffskontrollen, die angewendet werden können. Die spezifischen Zugriffskontrollen sind im target object definiert.

targets objects

Targets objects enthalten folgende Informationen:

- Mengen von MOs, ausgewählt durch Name, Inhalt oder Filter
- Operationen, die auf dem MO ausgeführt werden dürfen
- Initiatoren, die die Operationen aufrufen dürfen.

authorized-initiators objects

Dieses Objekt spezifiziert autorisierte Initiatoren, die folgendermaßen identifiziert werden können:

- Fähigkeit:
  Durch den Besitz einzelner Fähigkeiten bekommt ein Initiator bestimmte Zugriffsrechte.
- Label:
  Der Initiator legt Zeugnis über die Zugriffsberechtigung auf Verwaltungsinformationen durch den Wert eines security labels ab.
- Identität:
  Der Initiator wird durch eine access-control list anhand seines Individualnamens, seiner Gruppenkennung oder als unbekannt identifiziert.

6.3 Accounting-Meter Function

Diese Funktion bietet Buchführungsmaße und Services zur Aufzeichnung und Wiedergabe von Daten über die Benutzung von Ressourcen und zur Entscheidung, welche von den Daten zu sammeln sind und unter welchen Voraussetzungen über sie ein Bericht angefertigt werden soll. Dazu definiert die Accounting-Meter Function drei Objekte:

- accounting-meter-control object
  Dieses Objekt spezifiziert die Regeln zur Sammlung von Daten über die Benutzung spezieller Ressourcen. Es kann als Teil der Ressource-Objekt Instanz oder erst wenn eine Verwaltungsanforderung an die Buchhaltungsfunktion vorliegt erzeugt werden. Die accounting-management Kontrolle erlaubt dem verwaltenden System

  1. Die Sammlung von Ressourcenbenutzungsdaten einer Ressource, und die Steuerung der Sammlung durch Operationen auf dem accounting-meter-control Object.
  2. Die Auswahl, welche Ressourcenbenutzungsdaten gesammelt werden sollen, und unter welchen Umständen über sie berichtet werden soll.

- accounting-meter-data object
  Dieses Objekt enthält Informationen über die Identität des Benutzers der Ressource, und ein quantitatives Maß der Benutzung.

- accounting-record object
  Die Daten der accounting records werden entweder durch das Lesen von Buchhaltungsdaten eines accounting-meter-data Objects, oder aus Meldungen, die von accounting-meter-data Instances erzeugt werden, abgeleitet. Sie werden nach einem Auswahlverfahren aufgezeichnet.

6.4 Workload-Monitoring Function

Diese Funktion wird eingesetzt, um potentielle Ressourcenüberlastungssituationen, wodurch Fehler auftreten können, festzustellen. Dazu definiert sie MOs, die die Systemleistung reflektieren. Der Service, den die Workload-Monitoring Funktion zur Verfügung stellt, läßt sich durch folgende Dimensionen charakterisieren:

- Ressourcenbenutzungsmodell:
  Die Funktion erlaubt dem Verwalter zur Erreichung seiner Anforderungen das Benutzen eines oder mehrerer der folgenden Modelle.

  - Ressourcenbenutzung: zeigt den allgemeinen Kapazitätsbedarf an,
  - Ressourcenzurückweisungsrate: mißt die Rate, bei der Ressourcenanforderungen wegen Überlastung zurückgewiesen werden und
  - Ressourcenanforderungsrate: mißt den Ressourcenbedarf.

  Bei der Verwendung mehrerer Modelle für eine Ressource besteht ein Zusammenhang zwischen den gemessenen Parametern.

- Attribut des MOs:
  Folgende Attribute reflektieren die Leistung einer Ressource:

  - nichtsetzbare Zähler (simple counter): ist nicht durch Verwaltungsoperationen beeinflußbar. Er zählt von 0 bis zu einem Maximalwert, springt von dort wieder auf 0 zurück und fängt wieder an zu zählen.
  - setzbare Zähler: zählt wie der simple counter, kann jedoch von Verwaltungsaktionen zurückgesetzt (reinitialisiert) werden und
  - Gauge: zeigt den Mittelwert einer dynamischen Variablen an.

- Metrisches Objekt:

  - gauge-monitor metric object und
  - mean-monitor metric object.

6.4.1 Metric-Monitoring Process

Der der Workload-Monitoring Function zugrunde liegende Prozess enthält folgende Schritte

1. Datenerfassung:
   In der Basic-Monitoring Function werden Daten von den beobachteten Objekten entnommen. Die Beobachtungswerte werden durch Testen von Attributswerten in Intervallen, die durch die Granularitätsperiode spezifiziert werden, berechnet.

2. Datenumformung:
   Zur Beobachtung der Veränderung eines Zählers wird der Zähler durch das Conditional-Counter-Difference Package in einen Gauge transformiert.

3. Datenerhöhung:
   Zur Trendfeststellung mit den Daten wird ein Data-Enhancement Algorithm zur Glättung der beobachteten Daten benutzt. Die Ergebnisse werden in einem Gauge gespeichert.

4. Datenanalyse:
   Die berechneten Meßwerte können zur Alarmauslösung mit Grenzwerten verglichen werden.

6.4.2 Monitor Objects

Die Workload-Monitoring Function benutzt viele MOs, um Statistiken über die angezeigte Ressource zu speichern. Jedes (metrische Objekt kann alle 3 leistungsabhängigen Attribute (simple counter, settable counter, gauge) anzeigen, und eines der 3 Leistungsmodelle

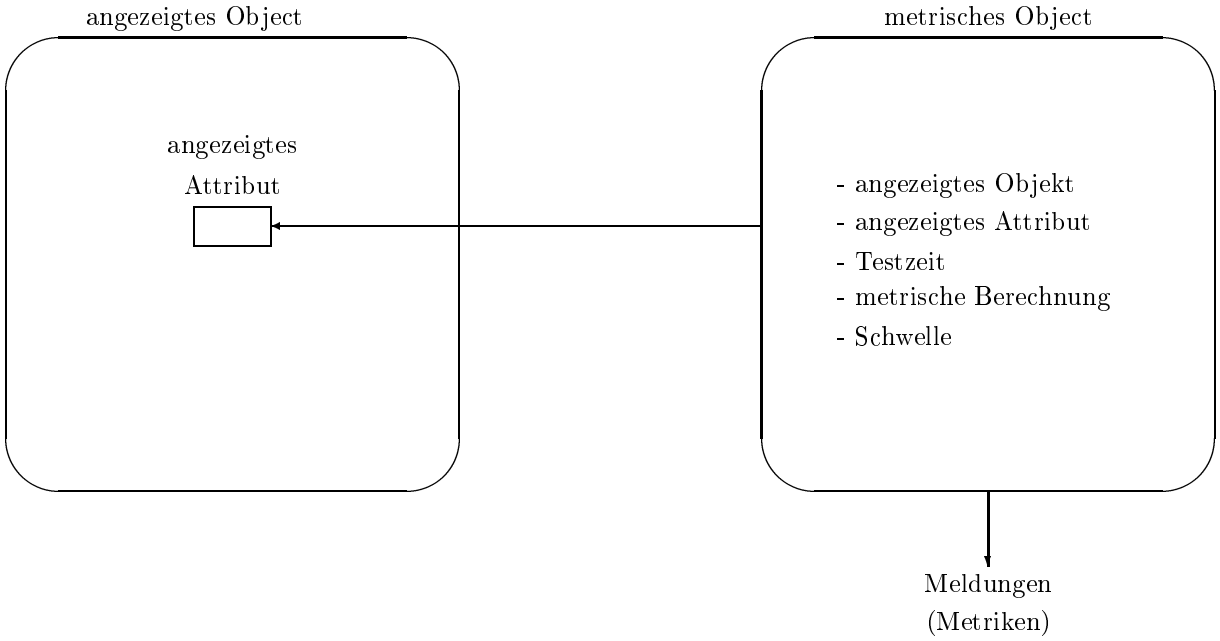


Abbildung 6.3: Metrische Objekte

(ressource utilization, ressource-rejection rate, ressource-request rate) unterstützen. Der Zusammenhang zwischen metrischem Objekt und angezeigtem Objekt ist in Abbildung 6.3 dargestellt.

Ein Metrisches Objekt holt sich Daten von einem einzelnen beobachteten Attribut. Das Gaugeattribut enthält die neueste Beobachtung und wird nach folgenden Regeln berechnet:

- Ist das beobachtete Attribut ein Gauge, so ist der abgeleitete Gaugewert der zuletzt beobachtete Gaugewert.
- Ist das Attribut ein simple counter, so ist der abgeleitete Gaugewert gleich der Differenz zwischen den suxessiven Beobachtungen des Zählers.
- * ist das Attribut ein settable counter, so ist der abgeleitete Gaugewert gleich dem Wert des Zählers vor der letzten Rücksetzung.

Beim simple counter muß bei der Berechnung das Umsetzen vom Maximum auf Null berücksichtigt werden.

Gauge-Monitor Metric Object

Das Gauge-Monitor object liefert einen Wert, der von einem counter oder einem gauge unter Beobachtung abgeleitet wird. Als spezifisches Ressourcenattribut kann ein gauge jeden Wert zwischen 0 und der Kapazität der Ressource annehmen.

Mit dem gauge metric object sind zwei Schwellenpaare zur Ausgabe von Meldungen verbunden. Eine scharfe Schwelle zeigt an, daß die Ressource nahe an ihrer Kapazität arbeitet. Eine frühwarnende Schwelle zeigt an, daß die Ressource sich ihrer Kapazität nähert.

Jede Schwelle sendet zwei Arten von Meldungen aus:

- Bei Überschreiten der Schwelle in positiver Richtung, daß die Bedingung erfüllt ist.
- Bei Überschreiten der Schwelle in negativer Richtung, daß die Bedingung nicht mehr zutrifft.

Zur Vermeidung wiederholter Aussendung von Meldungen, z.B. bei einem gauge, der geringfügig um den Schwellenwert schwankt, werden die Schwellen in Paaren spezifiziert.

Mean-Monitor Metric Object

Dieses Objekt kann benutzt werden, um Meldungen zu erzeugen, die zu über der Zeit gemittelten Ressourcenbenutzungswerten gehören. Wie der gauge-monitor leitet der mean-monitor seinen Wert von einem counter oder gauge nach obigen Regeln ab. Der abgeleitete Wert dient hier jedoch als Eingabe für die Ermittlung eines geschätzten Mittels benutzt wird. Zur Auslösung von Meldungen werden auch hier Schwellen benutzt.

Es gibt drei Typen von mean-monitor metric objects:

- Moving average mean monitor object:
  Diese Objekt bietet nur eine Schätzung des Mittelwertes an. Dazu spezifiziert es eine von zwei Techniken:
  1. uniformly weighted moving average (`uwma`):
     Der Mittelwert wird aus Beobachtungen über einem Zeitraum ermittelt, die alle gleiche Gewichtung bekommen.
  2. exponentially weigted moving average (`ewma`):
     Bei der Mittelwertberechnung werden hier alle vergangenen Werte berücksichtigt, wobei weiter zurückliegende Werte geringere Gewichtung bekommen.

- ewma-mean variance monitor object:
  Diese Objekt bietet die Schätzung von Mittelwert und Varianz der abgeleiteten Meßgröße nach exponentiellen Algorithmen.

- ewma-mean percentile monitor object:
  Diese Objekt bietet exponentielle Schätzungen von Mittelwert, Median, n-ten Percentile und den größten und kleinsten Wert der abgeleiteten Meßgröße.

6.5 Test-Management Function

Diese Funktion spezifiziert ein Modell zur Verwaltung von Vertrauens- und Diagnosetestprozeduren. Sie enthält MO-Klassen zur Kontrolle der Tests, die interaktiv oder asynchron geführt werden und deren Ergebnisse später geliefert werden. Ferner bietet sie folgende Testoptionen:

- Synchroner oder asynchroner Test
- Gelieferter oder anzufordernder Bericht
- Implizite oder explizite Testbeendigung

6.5.1 Testmodell

Abbildung 6.4.(a) zeigt das allgemeine Testmodell. Ein Anwendungsprozeß im verwaltenden System (Testführer) gibt eine Testanforderung an einen Anwendungsprozeß im verwalteten System (Testausführer).

6.5.2 Berichtung von Testergebnissen

- Synchroner Test:
  Abbildung 6.4.(b) zeigt einen synchronen Test. Die Testergebnisse werden in der Testbestätigung an die initiierende Funktion zurückgeliefert.

- Asynchroner Test:
  Abbildung 6.4.(c) zeigt den asynchronen Test. Die letzten Testergebnisse müssen erst durch vorhergehende Verwaltungsoperationen oder Meldungen zugänglich gemacht werden. Der Bericht wird dann entweder automatisch durch Meldungen geliefert, oder muß vom Testführer durch Senden einer Repräsentationsanforderung an ein TO angefordert werden. Schlüsselelemente des asynchronen Tests sind:

  - test-action-request receiver (TARR):
    Ein MO mit dieser Fähigkeit kann auf Testanforderungen reagieren. Es kann eine Instanz einer bestimmten Testobjektklasse erzeugen, die einen bestimmten Testaufruf repräsentiert.
  - Testobjekt (TO):
    Dieses Objekt ist erforderlich, um Tests zu kontrollieren und anzuzeigen, und zur Ausgabe von Testmeldungen.
  - managed object under test (MOT):
    Dieses Objekt bietet einen Verwaltungsblick auf Testsubjekte.

6.5.3 Testbeendigung

- explizite Testbeendigung:
  Der Testführer fordert die Beendigung des Tests an (in asynchronen Tests). Unabhängig von seiner Vollständigkeit wird der Test bei Eingang der Beendigungsanforderung von Testausführer beendet. Er bestätigt die Beendigung und liefer die gültigen Ergebnisse an den Testführer zurück.

- implizite Testbeendigung:
  Bei der Erfüllung vorherbestimmter Kriterien wird der Test beendet (in synchronen Tests). In diesem Fall ist der Test vollständig durchgeführt worden.

6.6 Summarization Function

Diese Funktion definiert ein Modell und eine MO-Klasse zur Anwendung statistischer Analyse auf und zur Zusammenfassung von Verwaltungsinformationen. Die Services umfassen die Spezifizierung von Verwaltungsobjekten und Attributen, die in die Zusammenfassungsberichte mit aufgenommen werden sollen, das Aufzeichnen von Beobachtungen dieser Objekte und Attribute und das Aufstellen der Zusammenfassungsreporte.

Die Zusammenfassungsfunktion zieht Informationen aus MOs und bringt sie in Zusammenfassungsobjekten unter. Informationsquellen sind Attribute von MOs, die unterlegene Ressourcen repräsentieren, von metrischen Objekten und von log records. Das Zusammenfassungsobjekt spezifiziert den Algorithmus zur Berechnung der Zusammenfassungsinformation aus den beobachteten Attributen. In Abbildung 6.5 ist die Zusammenfassungfunktion skizziert.

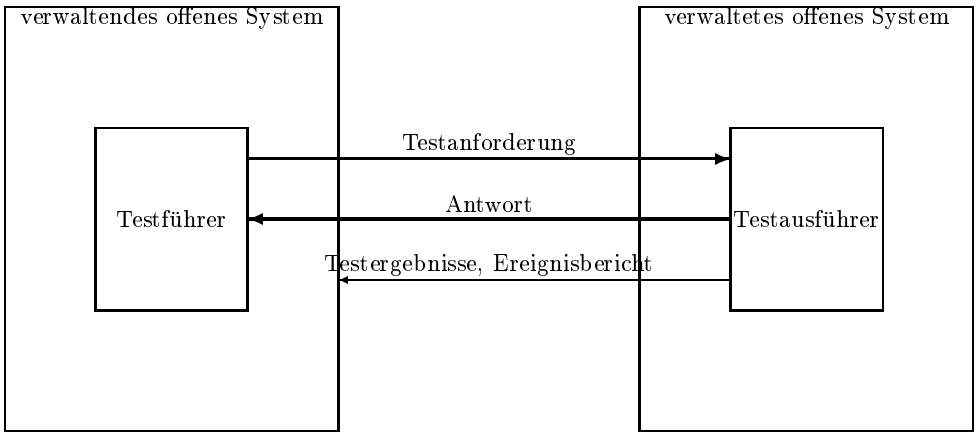
Alle Zusammenfassungsfunktionen basieren auf dem Scannerkonzept. Ein Scanner ist ein Testprozeß zur Beobachtung von Attributswerten zu bestimmten Zeitpunkten.

6.6.1 Summarization-Function Object

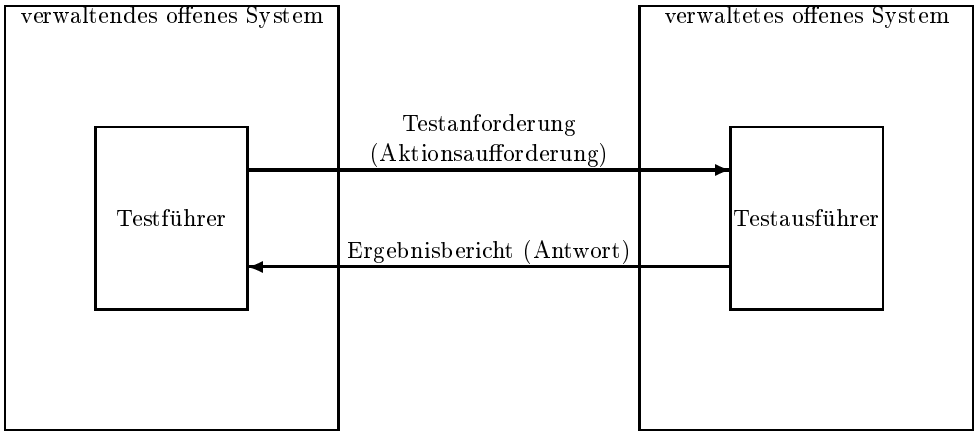
Die Scanner-Object Klasse ist eine Oberklasse, von der andere Zusammenfassungsobjekte abgeleitet werden. Ein Scannerobjekt definiert Vereinfachungen zum periodischen Testen der Werte einer spezifizierten Attributmenge zusammen mit einem gegebenen MO. Es kann Attributswerte anderer Objekte wiederherstellen und eine Zusammenfassungsinformation aus diesen Werten produzieren.

6.6.2 Homogeneous Scanners

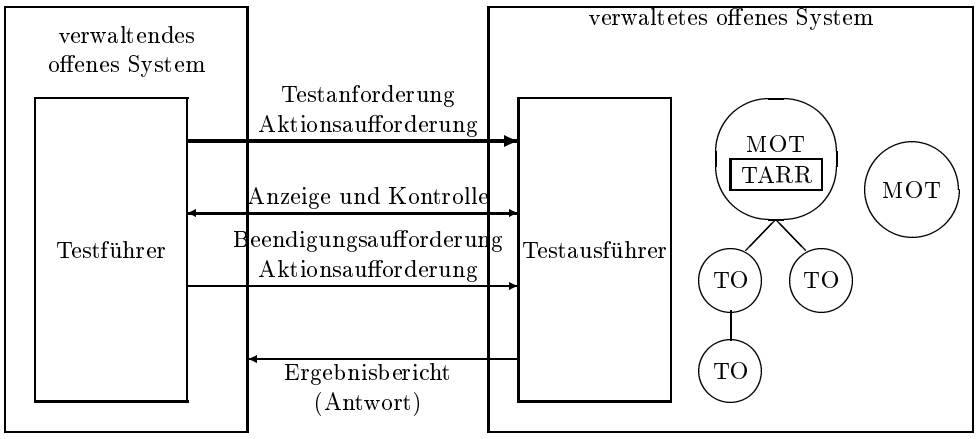
Diese Scanner beobachten eine Menge von Attributen, die für viele Objektklassen gültig sind. Die zubeobachtenden Objektinstanzen werden durch eine Liste, durch



(a) allgemeines Modell



(b) synchroner Test



(c) asynchroner Test

Abbildung 6.4: Modell der Testverwaltungsfunktion

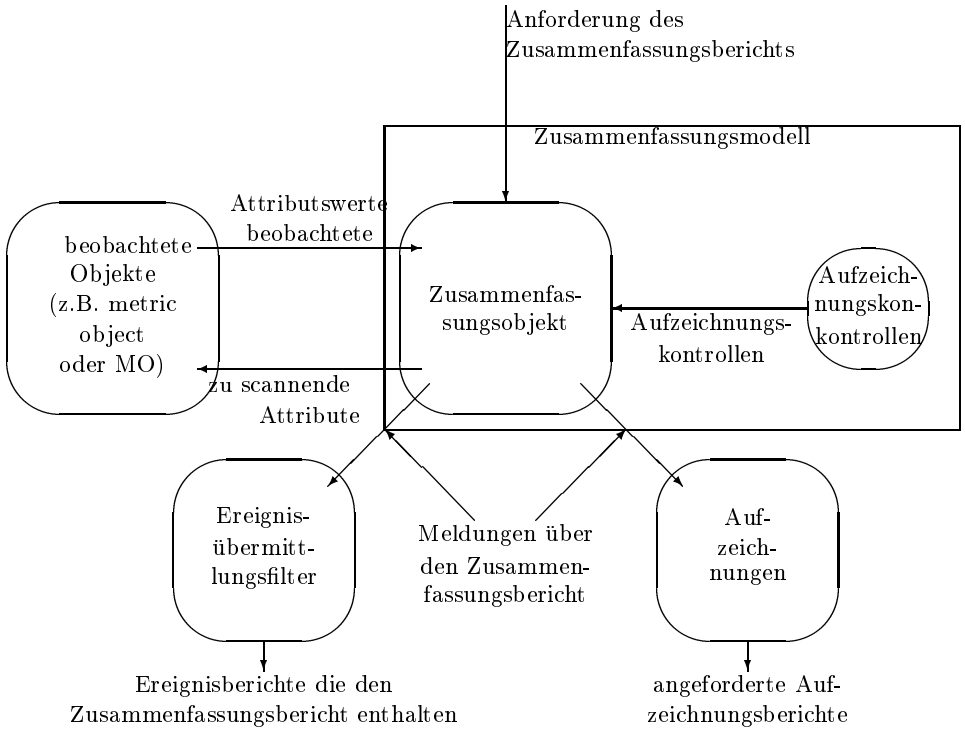


Abbildung 6.5: Zusammenfassungsfunktion

Eingrenzung oder durch Filterung festgelegt. Die zu beobachtenden Attribute in allen ausgewählten Objektinstanzen werden durch eine Liste identifiziert. Folgende Typen von Scannerobjekten sind definiert:

- simple scanner: liefert die Werte der gescannten Attribute
- mean scanner: liefert Werte und Testmittelwerte der gescannten Attribute
- mean-variance scanner: liefert die Werte und Testvarianz der gescannten Attribute.
- percentile scanner: liefert die Werte und Testprozenanteile der gescannten Attribute.
- min max scanner: liefert die Werte und das Minimum und Maximum der gescannten Attribute.

6.6.3 Heterogeneous Scanners

Dieser Scanner liefert am Ende jeder Granularitätsperiode die getesteten Werte einer Menge von Attributen, die aus MOs gesammelt wurden. Er enthält einen observation-identifier-list parameter, der eine Menge von Objektklassen, von Objektinstanzen und von zugehörigen Attributsidentifikatoren ist. Es können unterschiedliche Attribute aus verschiedenen Objektinstanzen beobachtet werden. Die Ergebnisattributswerte werden im Bericht den Attributsidentifikatoren zugeordnet.

6.6.4 buffered Scanners

Dieses Scannerobjekt arbeitet wie ein heterogener Scanner. Er speichert jedoch die gescannnten Attributswerte und gibt sie in einer von der Granularitätsperiode unabhängigen Reportperiode aus.

6.6.5 Dynamic Scanner

Dieser Scanner wird durch eine Aktionsanfrage, spezifizierte Attribute zu beobachten und Ergebnisse zurückzuliefern, aktiviert. Der bislang einzige dynamische Scanner ist der dynamic simple scanner, der nur die beobachteten Werte zurückliefert.

6.7 Literaturverzeichnis

[Sta93l]

Kapitel 7

Simple Network Management Protocol (SNMP) - Konzepte und Datenbasis

Hajo Brunne

7.1 Einführung

Mit dem Begriff SNMP wird eine Menge von Spezifikationen im Bereich Netzwerkmanagement bezeichnet, wie einmal das Protokoll selbst, sowie die Definition der Datenbasis und die mit SNMP verbundenen Konzepte.

SNMP ist ein Teil der TCP/IP (transmission control protocol/internet protocol) Protokollsuite, deshalb zunächst ein kurzer Rückblick auf die Entwicklung von TCP/IP.

7.1.1 Die Entwicklung von TCP/IP

Etwa 1969 begann die advanced research projects agency (ARPA) im Auftrag des amerikanische Verteidigungsministerium ein Rechnernetz aufzubauen, um die Datenkomunikation für eigene Zwecke untersuchen zu können. Aus dieser Entwicklung ging das sog. ARPANET hervor, eines der ersten Paketvermittlungsnetze überhaupt.

Innerhalb des ARPANET mußte zunächst für verschiedenste Rechner entsprechende Kommunikationssoftware entwickelt werden, um Geräte verschiedenster Hersteller im Netz vereinigen zu können. Als später das ARPANET sich in das heutige Internet (ARPANET vereinigt mit vielen wide area networks (WAN) und lokalen Netzwerken) wandelte, wurde das Problem immer größer. Um die Kommunikationsprobleme zu lösen, entwickelten die Forscher beim ARPANET eine standarisierte Protokollfamilie, aus der 1970 TCP/IP hervorging. Dieses Protokoll wurde später vom DoD (engl.: Department of Defence) als Standard akzeptiert.

Ab 1980 begann TCP/IP seinen Siegeszug im kommerziellen Bereich und ein Aussterben ist im Moment nirgendwo zu beobachten. Dies ist auch auf die Trägheit der offiziellen Normungsgremien (ISO: International Standard Organisation) zurückzuführen, die eine schnelle Standardprotokolleinführung unmöglich macht. TCP/IP ist (im Gegensatz zu überladenen OSI-Protokollen) schnell und einfach zu implementieren und findet deshalb bei Herstellern wesentlich größere Akzeptanz.

7.1.2 Ursprünge des TCP/IP Netzwerkmanagements

Bis Ende der siebziger Jahre gab es kein Netzwerkmanagement in dem Sinne, wie wir es heute betrachten. Die einzigen Möglichkeiten Kontrollinformationen auszutauschen waren durch das internet-control message protocol (ICMP) gegeben. ICMP ist auf allen Geräten, die über TCP/IP kommunizieren, verfügbar. Damit ist es möglich, einfache Kontrollinformationen zwischen Routern und/oder Rechnern auszutauschen. Aus der Sicht des Netzwerkmanagement ist die Möglichkeit der Erreichbarkeitsanalyse die einzige Anwendung von ICMP. Mit einem bestimmten Nachrichtentyp ist es möglich eine IP-Netzverbindung auf Existenz hin zu überprüfen, d.h. ob ein Rechner überhaupt im Netz erreichbar ist. Damit ist es aber nicht möglich zu überprüfen, ob die darüber liegenden Protokollschichten arbeiten.

Da in den Jahren 1988 bis 1992 eine Verzehnfachung der Internetteilnehmer stattgefunden hat, wurde die Dringlichkeit eines standarisierten Netzwerkmanagementprotokolls mit umfangreicheren Möglichkeiten als ICMP immer größer.

Es wurden für ein neues Protokoll folgende Vorschläge gemacht:

- High-level entity-management system (HEMS): Verallgemeinerung des ersten Netzwerkmanagementprotokolls überhaupt, dem Host monitoring protocol (HMP)
- SNMP: Eine Erweiterung des simple gateway monitoring protocol (SGMP)
- CMIP over TCP/IP (CMOT): war ein Versuch das von der ISO standarisierte common management information protocol (CMIP), Dienste und Datenbankstruktur für das Netzwerkmanagement zu übernehmen

Anfang 1988 entschied man sich für folgende Lösung: es wurde SNMP als kurzfristige und CMOT als langfristige Lösung empfohlen. Man ging davon aus, daß

früher oder später die TCP/IP-Welt sich in Richtung OSI-Protokolle entwickeln würde. SNMP ist schnell zu implementieren und kann so für das Management genutzt werden. In der Zwischenzeit sollte versucht werden CMIP auf TCP aufzusetzen, um so dem künftigen Standard ein Stück näher zu kommen. Wenn dann der Wechsel zu OSI-orientierten Protokollen vorgenommen würde, wäre der Aufwand entsprechend geringer.

Außerdem wurde festgelegt, daß SNMP und CMOT die selbe Datenbasis der gemanageden Objekte verwenden sollte. Eine einzige Managementstruktur (structure of management (SMI)) und eine einzige Datenbasis (management information base (MIB)) wurden für beide Protokolle definiert.

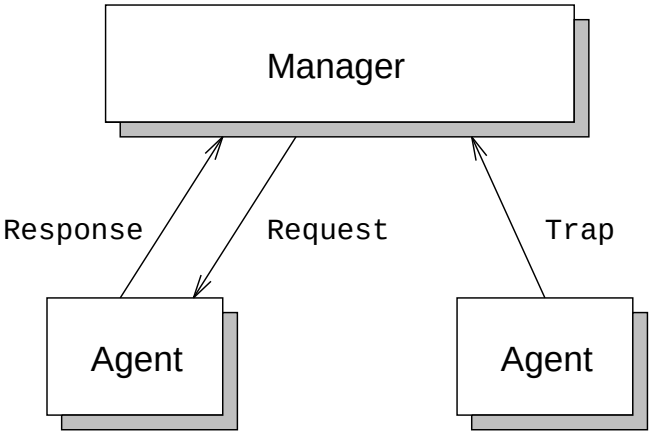


Abbildung 7.1: Struktur des SNMP Konzepts

7.2 Grundlegende Konzepte

7.2.1 Netzwerkmanagement-Architektur

Das Modell, welches für TCP/IP Netzwerkmanagement benutzt wird, bedient sich folgender Schlüsselelemente:

- Management Station
- Management Agent
- Management Datenbank (MIB)
- Management Protokoll

Die Management Station ist als ein herausgehobener Knoten im Netz zu sehen, der als Schnittstelle für den Netzwerkmanager (als Person) dient. Im Minimalfall hat der Management-Knoten

- eine Menge von Applikationen zur Datenanalyse, Fehlererkennung und -behebung etc.
- ein Interface, über das der Netzwerk-Manager das Netz beobachten und kontrollieren kann (z.B. graphisch unterstützt)
- eine Datenbasis mit Informationen aus den MIB's aller im Netz verwalteten Knoten

Im Prinzip fällt aber nur der letzte Punkt unter das Thema SNMP.

Die andere wichtige Komponente neben dem Manager ist der Management-Agent. Alle wichtigen Komponenten im Netz (Rechner, Hubs, Bridges) können mit SNMP ausgestattet sein (als Agent), so daß sie von einem Manager im Netz verwaltet werden können.

Der verwaltete Agent antwortet dem Manager auf von ihm gesendete Anforderungen, der Agent kann wiederum unaufgefordert wichtige Informationen an den Manager senden (trap).

Die Architektur von SNMP sieht den strukturellen Aufbau, wie in Abb. 7.1 dargestellt, vor.

Die Idee einzelne Resourcen im Netz zu verwalten ist, sie als Objekte darzustellen. Jedes Objekt ist letztendlich ein Variablenwert, der eine bestimmte Größe wiedergibt, die von Interesse ist (als Beispiel sei die Anzahl der eingegangenen TCP-Packete in einem Rechner genannt). Die Sammlung dieser Objekte wird MIB genannt. Die MIB fungiert als eine Sammlung von Zugriffspunkten des Agents für den Manager. Die Objekte der MIB sind netzwerkweit standarisiert, d.h. z.B. alle Bridges halten die selben Objekte bereit. Die Managementaktivität der Management Station besteht nun darin einzelne Objektwerte aus der MIB eines Gerätes zu lesen, zu beurteilen und ggf. eine Umkonfigurierung eines Knotens vorzunehmen, in dem der Manager einen Wert (oder mehrere) in der MIB des Agents ändert.

Diese Kommunikation wird mit Hilfe des Netzwerkmanagementprotokolls abgewickelt, in dem im Wesentlichen folgende Schlüsselfunktionalitäten zur Verfügung stehen:

- `Get`: ermöglicht der Management Station den Wert eines Objektes von einem Agent zu lesen
- `Set`: ermöglicht der Management Station den Wert eines Objektes von einem Agenten zu setzen
- `Trap`: ermöglicht einem Agenten die Management Station über ein wichtiges Ereignis zu informieren.

Bezüglich dem Verhältnis verwaltete Agenten zu Management Stationen gibt es keine Richtlinien. Es ist aber sicherlich vernünftig, Ausfallsicherheit durch die Präsenz von mindestens zwei Management Stationen im Netz zu realisieren.

7.2.2 Netzwerk Management Protokoll Architektur

SNMP wurde als Protokoll in der Anwendungsschicht entworfen und ist Teil der TCP/IP Protokollklasse. Es setzt im Allgemeinen auf dem user datagram protocol (UDP) auf. Die Abbildung 7.2 verdeutlicht die typische Protokollkonfiguration.

Jeder Agent muß also Implementierungen der Ebenen SNMP, UDP und IP zur Verfügung haben.

Zusätzlich gibt es einen Agenten-Prozeß, der die SNMP-Nachrichten interpretiert und die MIB des Agenten verwaltet.

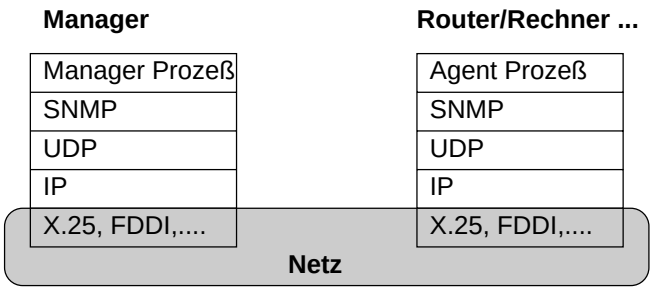


Abbildung 7.2: Protokollkonfiguration

Die Abbildung 7.3 bietet einen etwas detaillierteren Einblick in den Protokollaufbau und die Rolle von SNMP.

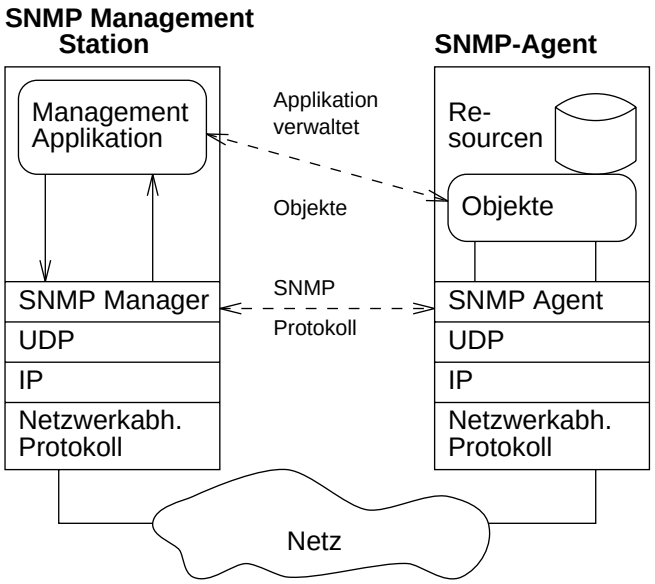


Abbildung 7.3: SNMP Architektur

Die Management Applikation schickt an den SNMP-Manager drei Typen von Nachrichten: `GetRequest`, `GetNextRequest` und `SetRequest`. Alle drei Nachrichten werden vom SNMP-Agent mit `GetResponse` beantwortet. Darüberhinaus kann ein Agent eine `Trap`-Nachricht verschicken, wie bereits erwähnt. Anlaß für einen `Trap` könnten beispielsweise der Reboot des Gerätes, Agentenabsturz, Netzschnittstellenversagen oder eine Überlastsituation sein.

Da UDP ein verbindungsloser Dienst ist, ist die Verbindung zwischen Management Station und einem Agent ebenfalls verbindungslos. Jeder einzelne Nachrichtenaustausch ist somit eine eigene Transaktion.

7.2.3 Stellvertreter-Mechanismus (Proxy)

Die Benutzung von SNMP erfordert es, daß alle Agenten UDP und IP unterstützen. Diese Tatsache schließt zunächst manche Geräte (wie manche Bridges oder Modems) vom Netzwerkmanagement aus. Außerdem kann es in einem Netz Geräte geben, die zwar über TCP/IP vernetzt sind, auf Grund ihrer technischen Leistungsdaten nicht in der Lage sind, die zusätzliche Last von SNMP, der Agentenlogik inklusive einer MIB-Verwaltung zu bewältigen.

Um nun Geräte ohne SNMP-Implementierung mit einbeziehen zu können, wurde das Proxy-Prinzip (engl.: Proxy = Stellvertreter) entwickelt.

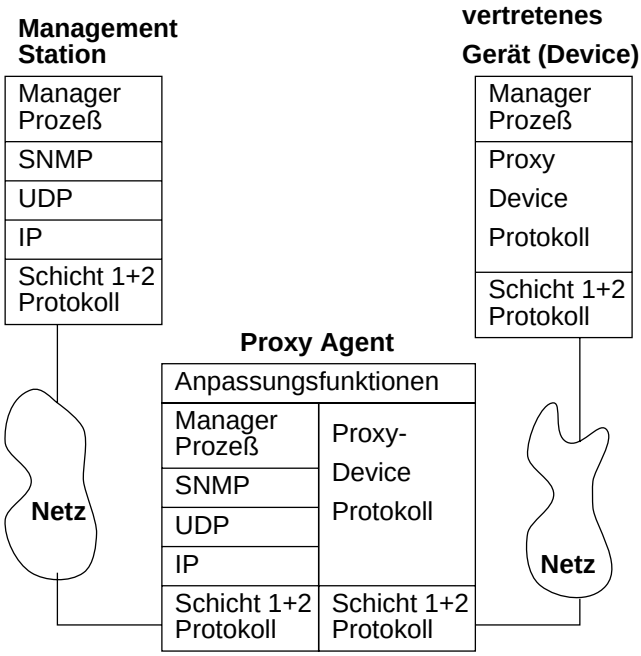


Abbildung 7.4: Proxy Mechanismus

Die in Abb. 7.4 dargestellte Konfiguration zeigt das Prinzip. Die Management-Station sendet Anfragen, die das Proxy-Gerät betreffen, an den Proxy-Agenten. Der Proxy-Agent konvertiert nun diese Anfrage in das vom Gerät benutzte Management-Protokoll. Wenn der Agent eine Antwort vom Proxy-Gerät erhält, schickt der Agent eine SNMP-Antwort zurück zum Manager.

Durch diese Konfiguration ist es nun auch möglich, Geräte mit einem von SNMP verschiedenen Netzwerkmanagement-Protokoll in ein gemeinsames Management einzubeziehen.

7.3 Management Information Base

Wie in jedem Netzwerkmanagementsystem ist auch im TCP/IP-Netzwerkmanagement die Datenbasis, die die zu verwaltenden Elemente enthält, wesentlich. Sowohl in TCP/IP, als auch in OSI Umgebungen wird diese als Management Information Base (MIB) bezeichnet. Jede Resource, die verwaltet werden soll, wird in der MIB durch ein Objekt repräsentiert. Jeder Knoten im Netz verwaltet seine eigene MIB, die den Status der für ihn relevanten Objekte enthält. Die Management Instanz (Management Station) kann nun die Resourcen beobachten, in dem sie sie liest, bzw. kann Resourcen kon-

trollieren, in dem sie Werte modifiziert. Um diesen Anforderungen gerecht werden zu können, muß die MIB zwei Operationsanforderungen erfüllen:

1. Das Objekt bzw. die Objekte, die eine bestimmte Information repräsentieren, müssen in jedem Knoten gleich sein.
2. Eine gemeinsame Darstellungsform muß benutzt werden, um die Interoperabilität (von Geräten verschiedenster Hersteller) realisieren zu können.

Der zweite Punkt wird durch die Definition der SMI (structure of management information) erreicht. Die Erfüllung der ersten Forderung erreicht man durch die Definition der Objekte und der Struktur in der MIB.

7.3.1 SMI

Die structure of management information, welche im RFC 1155 (request for comments) spezifiziert wird, definiert die Rahmenbedingungen, unter denen eine MIB konstruiert und definiert werden kann. Außerdem identifiziert die SMI Datentypen, die in der MIB Verwendung finden und definiert, wie Resourcen in der MIB dargestellt und benannt werden.

Die Philosophie bei der SMI ist Einfachheit und Erweiterbarkeit innerhalb einer MIB. Deshalb können in der MIB nur einfache Datentypen: Skalare und Arrays aus Skalaren gespeichert werden. Die SMI erlaubt es **nicht**, komplexe Datenstrukturen zu kreieren bzw. sie abzufragen; komplexe Datenstrukturen werden zwecks einfacherer Implementierbarkeit und Kompatibilität vermieden.

Um eine standarisierte Darstellungsform für Daten anbieten zu können, muß die SMI standarisierte Techniken für

- Strukturdefinition einer MIB
- Objektdefinition (inklusive Syntax und Wertebereich)
- Kodierung der Objektwerte

anbieten. Der letzte Punkt wird mit den BER (Basic Encoding Rules) abgedeckt, auf die hier nicht eingegangen wird.

Struktur der MIB

Verbunden mit jedem Objekttyp in der MIB ist eine Bezeichner vom ASN.1-Typ (abstract syntax notation one) `OBJECT IDENTIFIER`. Diese Variable dient dazu, das Objekt zu benennen. Zusätzlich dient die Namenskonvention zur Identifikation von Objekttypen, da der Wert, der mit dem Typ `OBJECT IDENTIFIER` verbunden ist, hierarchisch ist.

Die Menge der definierten Objekttypen wird als Baum organisiert, bei dem die Wurzel sich auf den ASN.1 Standard bezieht.

Von der Wurzel ausgehend identifiziert jeder Wert einer Objektvariable einen Teilbaum.

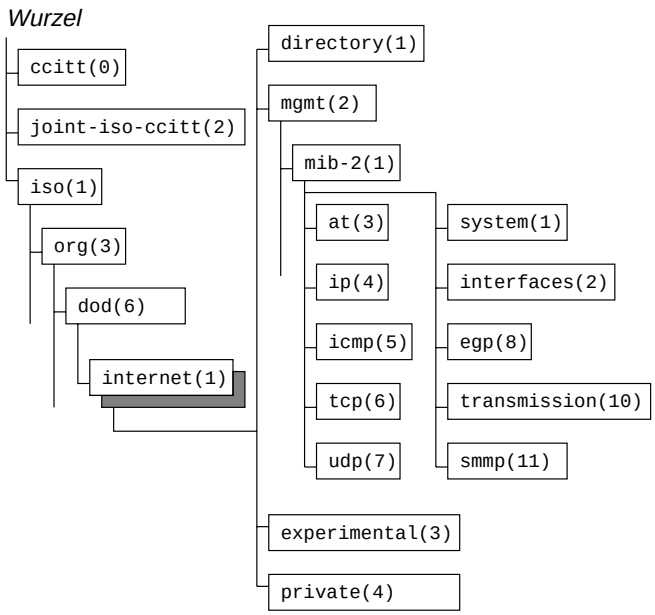


Abbildung 7.5: MIB Struktur

Im Teilbaum `iso` gibt es einen weiteren Teilbaum `org` für andere Organisationen, wie z.B. `dod` (departement of defence). RFC 1155 gibt nun die Empfehlung, daß ein Teilbaum unterhalb von `dod` für die Administrierung durch das IAB (internet activities board) reserviert werden soll. In der ASN.1 lautet diese Definition:

```
internet OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }
```

Der Knoten `internet` hat also den Wert 1.3.6.1, wodurch er eindeutig identifiziert ist. Dieser Wert dient als Präfix für alle Objekte, die eine Ebene tiefer liegen. Wie auch in Abb. 7.5 zu sehen, definiert das SMI-Dokument vier Knoten unterhalb des Internetknotens:

- `directory`: reserviert für künftige OSI-Anwendungen (X.500)
- `mgmt`: für vom IAB bestätigte Objekte
- `experimental`: enthält Objekte, die bei Internet Experimenten verwendet werden
- `private`: für private persönliche Objekte

Im `mgmt`-Unterbaum befinded sich auch die MIB, die vom IAB verabschiedet wurde. Zur Zeit existieren MIB-I und MIB-II, wobei letztere eine Obermenge von MIB-I darstellt. Beide haben den selben Variablennamen; es kann im gesamten Baum immer nur eine `mib` (1.3.6.1.2.1) vorhanden sein.

Objektdefinitionen in der MIB

Alle Objekte in der MIB werden mit Hilfe von ASN.1 definiert. Da allerdings die Einfachheit gefordert wurde, wird nur eine begrenzte Teilmenge von Elementen und Leistungsmerkmalen von ASN.1 genutzt.

Es werden zwei Klassen von Typen benutzt, einmal `UNIVERSAL` und `APPLICATION`.

Zur Definition von MIB-Objekten dürfen aus der Klasse `UNIVERSAL` nur folgende Typen verwendet werden:

- `INTEGER (UNIVERSAL 2)`
- `OCTET STRING (UNIVERSAL 4)`
- `OBJECT IDENTIFIER (UNIVERSAL 6)`
- `SEQUENCE, SEQUENCE OF (UNIVERSAL 16)`

Ein Objekt Bezeichner (OBJECT IDENTIFIER) ist ein eindeutiger Bezeichner eines Objektes, der aus einer Folge von Integer-Zahlen besteht (Unter-Bezeichner). Durch Lesen einer solchen Folge von links nach rechts wird der Ort eines Objektes in der MIB-Baumstruktur eindeutig definiert. Der Objekt Bezeichner für das Objekt `tcpMaxConn` leitet sich her, wie folgt:

iso	org	dod	internet	mgmt	mib-2	tcp	tcpMaxConn
1	3	6	1	2	1	6	4

Diesen Bezeichner würde man normaler Weise `1.3.6.1.2.1.6.4` schreiben.

Die beiden Typen `SEQUENCE` und `SEQUENCE OF` werden benutzt, um Tabellen zu konstruieren.

Nun zu den Typen der Klasse APPLICATION, welche im RFC1155 definiert werden:

- `NetworkAddress`: Universeller Typ für Netzwerkadressen, im Moment wird nur der Typ `IpAddress` unterstützt.
- `IpAddress`: 32-Bit-Adresse gemäß IP-Spezifikation
- `Counter`: Ein nicht negativer Zähler, der nur inkrementiert, jedoch nicht dekrementiert werden kann. Wird der Zählerstand $2^{32} - 1$ erreicht, springt der Zähler wieder auf Null.
- `Gauge`: Ein nicht negativer Zähler, der dekrementiert und inkrementiert werden kann. Er springt beim Inkrementieren bei Zählerhöchststand von $2^{32} - 1$ nicht auf Null, sondern verharrt mit diesem Wert bis zu einem Reset.
- `TimeTicks`: ein nicht negativer Zähler, der im Abstand von einer Hundertstel Sekunde erhöht wird. Der Zähler beginnt bei seiner Definition mit einem beliebigen Wert zu laufen.
- `Opaque`: beliebige ASN.1 Syntax kann durch Kodierung in einen `OCTET STRING` transparent übertragen werden.

Der Typ `Counter` ist auch als Rollover Counter bekannt. Typische Anwendungen sind das Zählen von Datenpaketen oder Bytes, die gesendet oder empfangen wurden. Ein Problem hierbei ist eben die Überlaufcharakteristik, die zu Fehlinterpretationen führen kann. Bei einem Zählerstand von x kann genauso der Wert $(2^{32} - 1) * n + x$ gemeint gewesen sein. Da der Zähler aber eine Breite von 32 Bit hat, sollte dieser Fall recht selten auftreten.

Der Typ `Gauge` wird typischer Weise verwendet, um den momentanen Wert eines Objektes, wie zum Beispiel die momentane Anzahl von Packeten in einer Warteschlange. Gauge kann ebenfalls dazu benutzt werden, um die maximale Wertdifferenz eines Objektes in einem Zeitraum zu speichern. In diesem Fall kann man so die Änderungsrate eines Objektwertes in einem Zeitraum beobachten.

Der Typ `TimeTicks` ist ein relativer Timer. Die Zeit wird relativ zu einem Ereignis gemessen, z.B. Systeminitialisierung. Es macht daher keinen Sinn, Timer-Werte mit den Werten aus anderen Systemen zu vergleichen. Es wäre eigentlich wünschenswert, einen Absoluttimer zu haben (um dann auch Timer zu vergleichen), jedoch unterstützt TCP/IP kein exaktes Zeitsynchronisationsprotokoll.

Ein wichtiger Typ der OSI SMI wurde in der SNMP SMI weggelassen, nämlich der Typ `threshold` (Schwellwert). Dieser Typ arbeitet wie folgt: übertritt der Wert des Objektes den Schwellwert in negativer oder positiver Richtung (je nach Definition), wird ein Ereignis ausgelöst und eine Nachricht an die Manager Station geschickt. Dieses Verhalten kann aber im Zweifelsfall zu einer Überflutung des Netzes mit solchen Nachrichten führen, wenn der Schwellwert ungünstig ausgewählt wurde. In SNMPv2 ist der Schwellwerttyp trotzdem enthalten.

In der SMI wird ein Makro definiert, welches die Definition von Objekten vereinheitlichen soll. Ein Ausschnitt aus diesem Makro ist in der Abb. 7.6 wiedergegeben.

Mit Hilfe dieses Makros wird festgelegt, welche Syntax ein Objekt haben soll, wie die Zugriffsmöglichkeiten auf ein Objekt aussehen sollen (lesbar, schreibbar, nicht zugreifbar etc.). Außerdem wird der Status eines Objektes festgelegt. Ein Objekt kann den Status mandatory (engl. zwingend) haben, dann muß das Objekt an der entsprechenden Stelle in der MIB in jedem Fall implementiert sein. Ebenso sind die Werte obsolete bzw. optional zu sehen. Des Weiteren wird mit jedem Objekt eine Beschreibung des Objektes im Klartext definiert. Diese Beschreibung dient dazu die Bedeutung der Objektinformation in Form eines Kurztextes festzuhalten. Weitere Details sind u.a. in RFC112 zu finden.

In der Abb. 7.8 ist eine Beispieldefinition eines MIB-Objektes zu sehen (entnomen aus RFC1213).

Hier wird das Object `tcpConnRemAddress` als vierter Eintrag unter dem Baum `tcpConnEntry` definiert. Die Syntax dieses Objektes ist eine IP-Adresse, es besteht Nur-Lese-Zugriff und der Status dieses Objektes ist `mandatory`, bei einer Implementierung des TCP-MIB-Teibaumes ist die Implementierung dieses Objektes Pflicht.

Objekt	Syntax	Zugriff	Status	Beschreibung
udpInDatagrams	Counter	RO	M	Summe der ausgelieferten UDP-Datagramme
udpNoPorts	Counter	RO	M	Anzahl der empf. Datagramme, für die am Zielport keine Applikation existierte
udpInErrors	Counter	RO	M	Anzahl der nicht ausgelieferten Datagramme aus einem anderen Grund
udpOutDatagrams	Counter	RO	M	Anzahl der gesendeten Datagramme
udpTable	SEQUENCE OF UdpEntry	NA	M	Enthält Informationen über UDP-Empfänger
udpEntry	SEQUENCE	NA	M	Informationen über einen einzelnen UDP-Empfänger
udpLocalAddress	IpAddress	RO	M	Lokale IP-Adresse des UDP-Empfängers
udpLocalPort	Integer	RO	M	Lokale Portnummer des UDP-Empfängers

Abbildung 7.7: Die Gruppe udp im Überblick

```
BEGIN
  TYPE NOTATION ::=
     "SYNTAX" type(ObjectSyntax)
     "ACCESS" Access
     "STATUS" Status

  VALUE NOTATION ::= value (VALUE ObjectName)

  Access ::= "read-only"
           | "read-write"
           | "write-only"
           | "not-accessible"
  Status ::= "mandatory"
           | "optional"
           | "obsolete"
           | "deprecated"

  DescrPart ::=
          "DESCRIPTION" value
               (description DisplayString)
           | empty
 ..... (Weiteres in RFC1212)
```

Abbildung 7.6: Macro zur Definition on Objekten (aus RFC1212)

```
tcpConnRemAddress OBJECT-TYPE
  SYNTAX  IpAddress
  ACCESS  read-only
  STATUS  mandatory
  DESCRIPTION
          "The remote IP address for this TCP
               connection."
 ::= { tcpConnEntry 4 }
```

Abbildung 7.8: Beispiel einer Objektdefinition

7.3.2 MIB-II

In diesem Kapitel werden die einzelnen Gruppen des Teilbaumes mib-2 (RFC 1213) kurz vorgestellt und eine Gruppe exemplarisch herausgenommen und ihre Objekte etwas genauer beleuchtet.

Gegenüber der MIB-I (RFC 1156) wurden für Objekte in der MIB-II folgende Einschränkungen vorgegeben:

- Ein Objekt gehört entweder zum Bereich Fehler- oder Konfigurationsmanagement
- Es dürfen nur Kontrollobjekte aufgenommen werden, die beschränkten Schaden durch unsachgemäße Benutzung verursachen können. Dieses Kriterium deutet darauf hin, daß die aktuellen Managementprotokolle nicht hinreichend sicher für umfangreichere Kontrolloperationen sind.
- Nachweis der Benutzbarkeit und Brauchbarkeit wird gefordert.
- Die maximal zulässige Anzahl von 100 Objekten wurde angehoben.
- Um redundante Variablen zu vermeiden wurde gefordert, daß kein Objekt hinzugenommen wird, was aus anderen Objekten der MIB abgeleitet werden kann.
- Betriebssystemspezifische Objekte wurden ausgeklammert.

Die Objekte der MIB-II werden in folgende Gruppen aufgeteilt:

- system: Globale Systeminformationen
- interfaces: Informationen über jede Schnittstelle von einem System zu einem Teilnetz
- at (address translation): beschreibt die Adressenübersetzungstabelle zur ISO/OSI Schicht 3 auf 2 Adressenumsetzung (z.B. IP auf Ethernet)

- `ip, icmp, tcp, udp, egp`: Informationen zur Implementierung und zu lokalen Gegebenheiten bzgl. des entsprechenden Protokolls
- `transmission`: Bietet Informationen über Übertragungsverfahren und Zugriffsprotokolle für jede Schnittstelle
- `snmp`: Informationen über die Implementierung und Ablaufverhalten von SNMP auf diesem System

Alle Gruppen der MIB-II hier genauer zu behandeln, würde den Umfang dieses Beitrages sprengen.

Rein exemplarisch will ich nun kurz die einzelnen Objekte der Gruppe `udp` vorstellen.

Die Objekte der Gruppe `udp` beschreiben verschiedene Attribute im Zusammenhang mit dem User Datagram Protocol, wie in Abb. 7.7 zusammenfassend dargestellt ist.

In der ersten Spalte ist der Objektname angegeben, unter dem das Objekt in der MIB referenziert werden kann. In der zweiten Spalte ist die Syntax wiedergegeben. Hier findet der oben genannte Typ `Counter` seine Anwendung. In der dritten Spalte ist der Zugriffsmodus angegeben. `RO` steht für read only, `NA` für not accessable. Dieses Attribut ist hier bei den beiden Objekten `udpTable` und `udpEntry` auf `NA` gesetzt, weil diese Objekte lediglich dazu dienen, eine Tabelle von UDP-Verbindungen zu definieren. Die eigentliche Information wird jedoch durch die Objekte `udpLocalAddress` und `udpLocalPort` repräsentiert, welche die Tabelleneinträge bilden.

Für den interessierten Leser sei hier auf RFC 1213 (Management Information Base for Network Management of TCP/IP-based internets: MIB-II) verwiesen. Dieses Dokument enthält die gesamte Definition der MIB-II in ASN.1.

7.4 Literaturverzeichnis

[Sta93i] [Sta93h] [Sch91] [Com88] [RM91] [MR91] [Ros91]

Kapitel 8

SNMP - Simple Network Management Protocol

Holger Wilhelm

8.1 Grundlagen

8.1.1 SNMP Operationen

SNMP unterstützt nur drei Operationen. Zwei davon sind für die Management Stationen:

- **Get/GetNext** - Informationen von einem Agenten empfangen
- **Set** - Informationen bei einem Agenten verändern

Mit diesen beiden Operationen lassen sich nur Werte aus einer Management Datenbank (MIB - Management Information Base) lesen oder verändern. Es ist nicht möglich, die Struktur der MIB durch Löschen oder Setzen von Objekten zu verändern. Es ist auch nicht möglich, eine Aktion des Agenten direkt auszulösen (z.B. Neustart des Agenten).

Die dritte Operation ist für die Agenten:

- **Trap** - Ein Agent sendet unaufgefordert einen Wert an die Management Station

Mit dieser Operation kann ein Agent ein Ereignis an die Management Station melden.

8.1.2 Gemeinschaften (Communities)

Beim SNMP basierten Netzwerkmanagement kann es mehrere Verwalter geben, die alle oder eine Teilmenge der Agenten steuern. Die Teilmengen können sich auch überschneiden. Jeder Agent kann von mehreren Verwaltern gesteuert werden. Deshalb muß er den Zugriff auf seine MIB kontrollieren können. Das SNMP, wie es im RFC1157 definiert wurde, enthält nur eine primitive und beschränkte Möglichkeit dies zu realisieren, nämlich das Gemeinschaftskonzept: Jeder Agent definiert für jede nötige Kombination von Zugriffsrechten einen eindeutigen Gemeinschaftsnamen. Dieser muß dem Verwalter bekannt sein, denn bei jeder Operation muß der Gemeinschaftsname als Identifikation mit übergeben werden. Die Gemeinschaftsnamen müssen nur bei jedem einzelnen Agenten eindeutig sein. Gleiche Namen können bei verschiedenen Agenten vorkommen, haben aber keinerlei Beziehung zueinander.

Authentication Service

Beim SMNP gibt es nur eine primitive Möglichkeit, die Authorisierung der anderen Stelle festzustellen. Bei jeder Get- oder Set-Operation des Verwalters muß er den Gemeinschaftsnamen mitsenden. Dieser Gemeinschaftsname funktioniert als Passwort.

Zugriffsrechte (Access Policy)

Durch die Definition mehrerer Gemeinschaften mit verschiedenen Zugriffsrechten kann ein Agent den Zugriff mehrerer Verwalter auf die MIB steuern. Um die Zugriffsrechte einer Gemeinschaft auf die MIB zu steuern, hat der Agent zwei Möglichkeiten:

- MIB-Sicht - Eine Gemeinschaft darf nur auf bestimmte Objekte der MIB zugreifen
- SNMP-Zugriffsrecht - Für jede Gemeinschaft kann ein Zugriffsrecht vergeben werden. Entweder “nur lesen” oder “lesen und schreiben”.

Die MIB-Sicht und das SNMP-Zugriffsrecht einer Gemeinschaft wird als SNMP Gemeinschaftsprofil (SNMP community profile) bezeichnet.

Jeder Eintrag in der MIB hat unabhängig vom SNMP-Zugriffsrecht ein Zugriffsrecht, das beachtet werden muß. Aus diesen beiden Zugriffsrechten ergeben sich folgende Zugriffsmöglichkeiten:

MIB-Zugriffsklasse	**SNMP-Zugriffsrecht**	
	nur lesen	*lesen-schreiben*
NUR LESEN	get,trap	get,trap
LESEN-SCHREIBEN	get,trap	get,set,trap
NUR SCHREIBEN	get,trap	get,set,trap
KEIN ZUGRIFF	kein Zugriff	kein Zugriff

Stellvertreter Betrieb (Proxy Service)

Für jedes repräsentierte Gerät wird ein Zugriffsprofil erstellt. Dadurch weiß der Stellvertreter durch die MIB-Sicht, mit welchen MIB-Objekten das Gerät gesteuert wird.

8.1.3 Instanzidentifikatoren

Jedes Objekt in der MIB hat einen eigenen Identifikator. Bei einem Zugriff auf die MIB will man aber nicht das Objekt, sondern die Instanz zu diesem Objekt.

Skalare Objekte

Bei skalaren Objekten gibt es keine Unklarheiten zwischen Objektidentifikator und Instanzidentifikator, da jedes skalare Objekt nur eine Instanz besitzt. Um aber zwischen Objektidentifikator und Instanzidentifikator unterscheiden zu können und um eine einheitliche Adressierung zu den Tabellenobjekten zu erreichen, wird an den Objektidentifikator eine Null (.0) angehängt, um den Instanzidentifikator zu erhalten.

Beispiel:

Objektname	Objektidentifikator	Instanzidentifikator
`tcpMaxConn`	`1.3.6.1.2.1.6.4`	`1.3.6.1.2.1.6.4.0`

Spaltenobjekte

Bei Objekten in einer Tabelle, die auch Spaltenobjekte genannt werden, reicht der Objektidentifikator nicht aus, um die Instanz zu identifizieren, da eine Spalte aus mehreren Instanzen besteht. Beim SNMP gibt es zwei Möglichkeiten, diese Instanzen anzusprechen: den seriellen Zugriff und den wahlfreien Zugriff. Zunächst wird nur der wahlfreie Zugriff betrachtet.

`tcpConn State` (`x.1`)	`tcpConn LocalAddress` (`x.2`)	`tcpConn LocalPort` (`x.3`)	`tcpConn RemAddress` (`x.4`)	`tcpConn RemPort` (`x.5`)
5	10.0.0.99	12	9.1.2.3	15
2	0.0.0.0	99	0	0
3	10.0.0.99	14	89.1.1.42	84
	↑ INDEX	↑ INDEX	↑ INDEX	↑ INDEX

Abbildung 8.2: `tcpConnTable`

`tcpConn State` (`x.1`)	`tcpConn LocalAddress` (`x.2`)	`tcpConn LocalPort` (`x.3`)	`tcpConn RemAddress` (`x.4`)	`tcpConn RemPort` (`x.5`)
x.1.z1	x.2.z1	x.3.z1	x.4.z1	x.5.z1
x.1.z2	x.2.z2	x.3.z2	x.4.z2	x.5.z2
x.1.z3	x.2.z3	x.3.z3	x.4.z3	x.5.z3

- z1 = 10.0.0.99.12.9.1.2.3.15
- z2 = 0.0.0.0.99.0.0
- z3 = 10.0.0.99.14.89.1.1.42.84

Abbildung 8.3: Instanzidentifikatoren

Jede Spalte hat den gleichen Objektidentifikator. Um die einzelnen Spalten zu unterscheiden, werden einfach die Werte der INDEX-Objekte angehängt. Als einfaches Beispiel betrachten wir die `ifTable` aus der Interface-Gruppe. Die Tabelle hat nur ein INDEX-Objekt, ifIndex, dessen Wert zwischen 1 und dem Wert von `ifNumber.0` liegt. Jedes Interface erhält eine eindeutige Nummer. Wir wollen jetzt den Interfacetyp des zweiten Interfaces wissen. Der Objektidentifikator von `ifType` ist `1.3.6.1.2.1.2.2.1.3`. Der Wert von `ifIndex`, der uns interessiert, ist 2. Also ist der entsprechende Instanzenidentifikator `1.3.6.1.2.1.2.2.1.3.2`. Es wird also einfach der Wert des INDEX-Objektes angehängt. Bei mehreren INDEX-Objekten werden diese der Reihenfolge nach angehängt. Ein Beispiel hierfür ist die Tabelle `tcpConnTable` aus der TCP-Gruppe. Diese Tabelle hat vier INDEX-Objekte. Also setzt sich der Instanzenidentifikator aus dem Objektidentifikator und den Werten der vier INDEX-Objekte der entsprechenden Spalte zusammen.

Die Abbildungen 8.2 und 8.3 zeigen die Tabelle `tcpConnTable` und die dazugehörigen Instanzenidentifikatoren.

Tabellen- und Spaltenobjekte

Da Tabellen- und Spaltenobjekte keine Blätter der Baumstruktur sind, besitzen sie auch keine Instanzen. Dadurch lassen sie sich aber nicht über SNMP ansprechen. Deshalb wird ihre MIB Zugriffsklasse in der MIB Definition auf 'kein Zugriff' (not accessible) gesetzt.

Lexikographische Sortierung

Die Identifikatoren sind eine Folge von nicht negativen, ganzen Zahlen. Ein Identifikator $(x_1, x_2, ..., x_n)$ kommt nach der Lexikographischen Sortierung vor einem Identifikator $(y_1, y_2, ..., y_m)$, wenn gilt:

$[(\forall 1 \leq j < k : x_j = y_j) \wedge (x_k < y_k)] \vee [(\forall 1 \leq j \leq n : x_j = y_j) \wedge (n < m)]$

Wenn man die Objekte in einem Baum darstellt, dessen Blätter von links nach rechts aufsteigend sortiert sind, dann läßt sich die lexikographische Sortierung als Tiefensuche realisieren.

Der Baum wird folgendermaßen rekursiv durchsucht:

1. Nehme die Wurzel
2. Gehe zu jedem Blatt von links nach rechts

Daraus ergibt sich die Lexikographische Sortierung:

1	2
1.1	2.1
1.2	2.1.1
1.2.1	2.1.1
	2.1.1.2
	2.1.1.3

Durch die lexikographische Sortierung kann ein Verwalter, der nicht weiß, welche Objekte der MIB von einem Agent unterstützt werden, die MIB in lexikographischer Reihenfolge durchsuchen, ohne die Objektnamen zu kennen.

8.2 Protokollspezifikation

8.2.1 PDU Formate

Beim SNMP werden Informationen zwischen einem Verwalter und einem Agenten über Nachrichten ausge-

`x.i.(tcpConnLocalAdress).(tcpConnLocalPort).(tcpConnRemAddress).(tcpConnRemPort)`

wobei

- `x` = `1.3.6.1.2.1.6.13.1`: Objektidentifikator für `tcpConnEntry`
- `i` = Identifikator für die Spalte der Tabelle
- (`name`) = Wert des Objektes `name`

Abbildung 8.1: Beispiel eines Instanzidentifikators

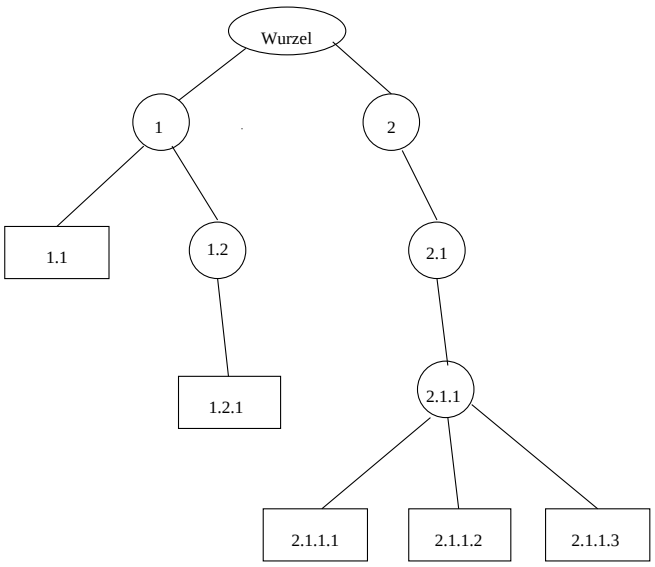


Abbildung 8.4: Beispiel eines MIB-Baums

tauscht. Jede Nachricht enthält SNMP Versionsnummer, einen Gemeinschaftsnamen, der für diesen Zugriff verwendet wird und einen der fünf Protokolldateneinheiten (`protocol data unit - PDU`). Das Bild 8.5 zeigt den Aufbau der SNMP Nachricht und der PDUs:

Man kann erkennen, daß die GetRequest, GetNextRequest und SetRequest PDU das gleiche Format wie die GetResponse PDU haben, nur die Felder error-status und error-index werden auf Null gesetzt. Dies verringert die Anzahl der verschiedenen PDU Formate und macht die Entschlüsselung damit einfacher.

8.2.2 Senden einer SNMP Nachricht

Eine SNMP Einheit führt folgende Schritte aus, um eine der fünf PDU Typen einer anderen SNMP Einheit zu übertragen:

1. Die PDU wird nach der ASN.1 Struktur, die im RFC 1157 definiert ist, zusammengesetzt.

2. Zusammen mit der Quell-, der Zieladresse und dem Gemeinschaftsnamen passiert die PDU einen Berechtigungsdienst. Dieser führt die nötigen Transformationen wie Verschlüsselung und Einfügen eines Berechtigungscodes ein und liefert das Ergebnis zurück.

3. Die Protokolleinheit setzt die Nachricht aus der SNMP Versionsnummer, dem Gemeinschaftsnamen

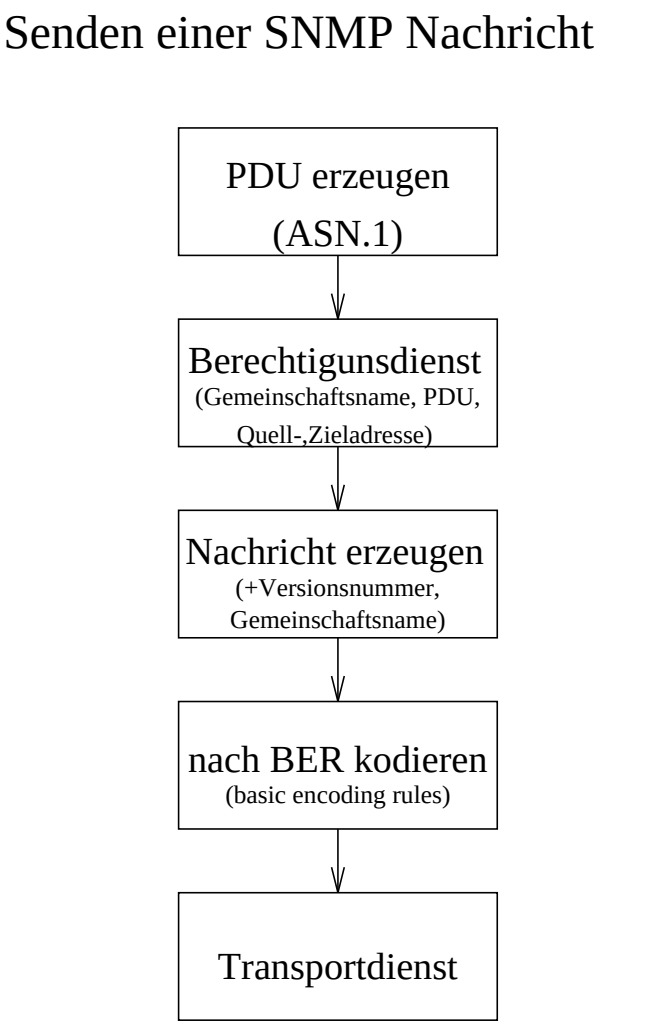


Abbildung 8.6: Ablauf beim Senden einer SNMP Nachricht

SNMP Nachricht:

version	community	SNMP PDU

GetRequest PDU, GetNextRequest PDU, SetRequest PDU:

PDU type	request-id	0	0	variable-bindings

GetResponse PDU:

PDU type	request-id	error-status	error-index	variable-bindings

Trap PDU:

PDU type	enterprise	agent-addr	generic-trap	specific-trap	time-stamp	variable-bindings

variable-bindings Feld:

name 1	value 1	name 2	value 2	$\cdots$	name n	value n

Abbildung 8.5: PDU Formate

und dem Ergebnis von (2) zusammen.

4. Dieses neue ASN.1 Objekt wird nach den Grundkodierungsregeln (`basic encoding rules` - BER) kodiert und zum Transportdienst weitergeleitet.

In der Praxis wird der Berechtigungsdienst (Schritt 2) nicht verwendet.

8.2.3 Empfangen einer SNMP Nachricht

Beim Empfang einer Nachricht werden von einer SNMP-Einheit folgende Schritte ausgeführt:

1. Es wird eine grundsätzliche BER-Konforme Syntaxprüfung durchgeführt. Wenn die Prüfung nicht erfolgreich war, wird die Nachricht verworfen.

2. Die Versionsnummer wird überprüft; wenn diese falsch ist, wird die Nachricht verworfen.

3. Dann wird der Gemeinschaftsname, die PDU-Einheit der Nachricht und die Quell- und Zieladresse, die vom Transportdienst geliefert werden, einem Berechtigungsdienst zugeführt. Mögliche Ergebnisse:

   keine Berechtigung: SNMP Protokolleinheit sendet einen Trap (authentication Failure) und die Nachricht wird verworfen.

   Berechtigung erfolgreich: Es wird eine PDU im ASN.1 Format zurückgeliefert.

4. Es wird eine Syntaxprüfung der PDU durchgeführt. Wenn diese Prüfung nicht erfolgreich war, wird die Nachricht verworfen. Andernfalls wird mit dem Gemeinschaftsnamen das SNMP-Zugriffsrecht gewählt und die PDU ausgeführt.

In der Praxis überprüft der Berechtigungsdienst nur, ob der Gemeinschaftsname dazu berechtigt, Nachrichten von der SNMP Quelleinheit zu empfangen.

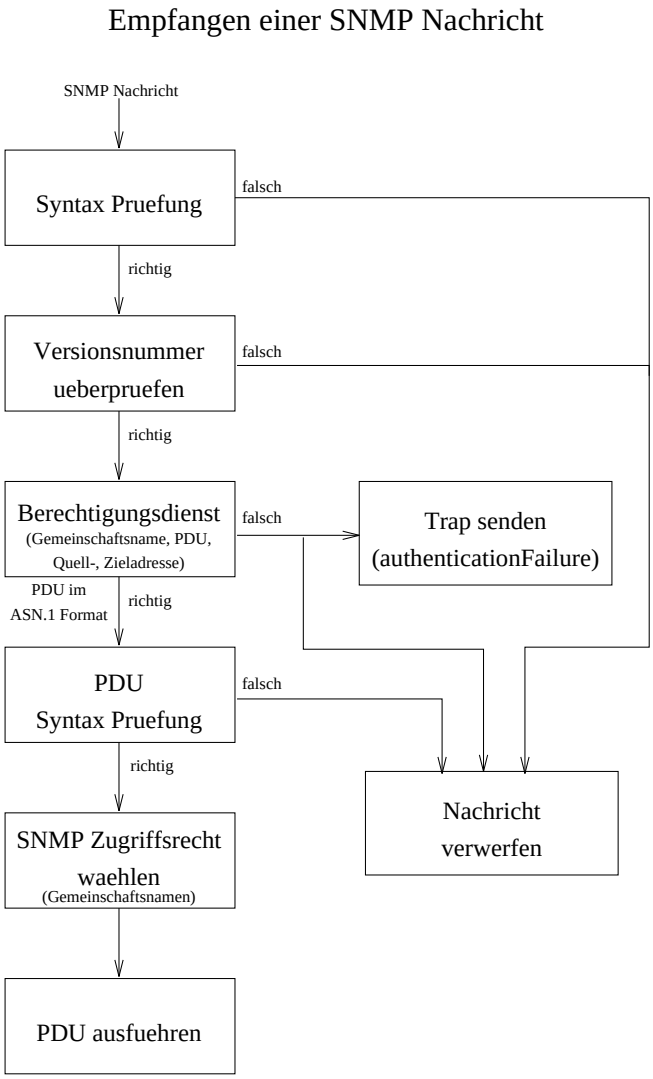


Abbildung 8.7: Ablauf beim Empfangen einer SNMP Nachricht

8.2.4 Variable-bindings Feld

In der Baumstruktur kann nur auf Blätter des MIB-Baumes zugegriffen werden, d.h. nur auf Objektinstanzen, also skalare Werte. SNMP bietet aber auch die Möglichkeit, auf mehrere Werte mit nur einem Befehl zuzugreifen (Set,Get,Trap). Dies verringert den Netzwerkaufwand für die Managementfunktionen.

Um einen solchen Zugriff zu machen, enthält jede PDU ein Feld, welches eine Reihe von Referenzen auf Objektinstanzen mit ihren zugehörigen Werten enthält. Bei PDUs, die keinen Wert benötigen (z.B. Get), wird der Eintrag für dieses Feld von der Protokolleinheit des Empfängers ignoriert. RFC 1157 schreibt vor, daß in diesem Fall das Wertfeld als NULL (definiert in ASN.1) gesetzt wird.

Zu beachten ist, daß SNMP nur den Zugriff auf Blätter des MIB-Baumes erlaubt, es ist also auch nicht möglich, durch Angabe z.B. eines Tabellenobjektes (z.B. `ipRouteTable`), eine ganze Tabelle abzufragen. Um dies zu realisieren, muß jeder Wert der Tabelle einzeln abgefragt werden.

8.2.5 GetRequest

Eine SNMP Einheit, die eine GetRequest Nachricht sendet, füllt folgende Felder der GetRequest PDU aus:

- **PDU type**: Kennzeichnung, daß es sich um eine GetRequest PDU handelt
- **request-id**: Eine Nummer, die die PDU eindeutig kennzeichnet, so daß die Rückantwort eindeutig der Anfrage zugeordnet werden kann. Dadurch können verlorengegangene Anfragen und duplizierte Antworten erkannt und richtig behandelt werden.
- **variable-bindings**: Eine Liste der Objektnamen, deren Werte angefordert werden.

Der Agent, der die Nachricht erhält, antwortet mit einer GetResponse-PDU, die die gleiche `request-id` enthält. Die GetRequest-Nachricht ist atomar, d.h. entweder werden alle Werte zurückgeliefert oder, wenn nur ein Wert der angeforderten Objekte nicht zurückgeliefert werden kann, keiner. In diesem Falle wird eine GetResponse-Nachricht gesendet die im Feld `error-status` den Fehlertyp und im Feld `error-index` die Nummer des Objektes enthält, das den Fehler verursacht hat. Mögliche Fehler sind:

- **noSuchName**: Der Objektinstanzname existiert nicht in der MIB-Sicht oder das Objekt ist keine Instanz, d.h. es ist kein Blatt des MIB-Baumes.
- **tooBig**: Es werden alle Werte zur Verfügung gestellt, aber aufgrund lokaler Beschränkungen können nicht so viele Werte gleichzeitig zurückgeliefert werden.
- **genError**: Ein Wert wird aus irgendeinem anderen Grund nicht unterstützt.

8.2.6 GetNextRequest

Die GetNextRequest-PDU ist identisch mit der GetRequest-PDU, der einzige Unterschied ist der Wert, der zurückgeliefert wird. Es wird nicht der Wert des Objektes zurückgeliefert, der in dem variable-bindings Feld steht, sondern die Instanz, die in lexikographischer Reihenfolge danach steht. Das Interessante daran ist, daß das übergebene Objekt nicht überprüft wird, also auch nicht unbedingt ein Blatt des MIB-Baumes oder in der MIB-Sicht vorhanden sein muß. Dadurch ergeben sich interessante Anwendungsmöglichkeiten dieser Operation.

Mehrere skalare Objekte abfragen

Angenommen, ein Verwalter will alle skalaren Werte der UDP (User Datagram Protocol) Gruppe mit einer SNMP-Nachricht abfragen, dann hätte die GetRequest Nachricht folgende Form:

`GetRequest(udpInDatagrams.0, udpNoPorts.0, udpInErrors.0, udpOutDatagrams.0)`

Wenn jetzt aber auf einen Wert nicht zugegriffen werden kann, liefert der Agent keinen einzigen Wert, sondern nur eine Fehlermeldung. Würde man dagegen folgende GetNextRequest Nachricht verwenden:

`GetNextRequest(udpInDatagrams, udpNoPorts, udpInErrors, udpOutDatagrams)`

werden alle Werte, auf die zugegriffen werden darf, zurückgeliefert. Bei allen anderen Werten wird der Wert, der in lexikographischer Reihenfolge als nächstes kommt, zurückgeliefert. Darf z.B. in vorigem Beispiel auf den Wert udpNoPorts nicht zugegriffen werden, würde folgende Nachricht zurückgeliefert:

`GetResponse((udpInDatagrams.0 = 100), (udpInErrors.0 = 2), (udpInErrors.0 = 2), (udpOutDatagrams.0 = 200))`

Man braucht nicht eine neue Nachricht zu senden, sondern erhält alle Werte, auf die zugegriffen werden darf. Die Werte, auf die nicht zugegriffen werden darf, lassen sich leicht durch die verschiedenen Objektnamen bei der Anfrage und der Antwort erkennen.

Unbekannte Objekte abfragen

Dadurch, daß die Objektnamen nicht überprüft werden, sondern nur der nächste Wert in lexikographischer Reihenfolge gesucht wird, lassen sich Objekte abfragen, deren Namen nicht bekannt sind. Es lässt sich so z.B. auch die gesamte MIB-Sicht durchsuchen.

Tabellenwerte abfragen

Die GetNextRequest-Nachricht läßt sich auch günstig einsetzen, um Tabellenwerte oder ganze Tabellen auszulesen, auch wenn nicht bekannt ist, wieviele Zeilen die Tabelle enthält.

Als Beispiel wollen wir folgende Tabelle abfragen:

ipRouteDest	ipRouteMetric1	ipRouteNextHop
9.1.2.3	3	99.0.0.3
10.0.0.51	5	89.1.1.42
10.0.0.99	5	89.1.1.42

Die Abfrage der Tabelle läuft folgendermaßen ab:

```
GetNextRequest(ipRouteDest, ipRouteMetric1,
    ipRouteNextHop)

GetResponse((ipRouteDest.9.1.2.3 = 9.1.2.3),
    (ipRouteMetric1.9.1.2.3 = 3), (ipRouteNextHop.9.1.2.3
    = 99.0.0.3))

GetNextRequest(ipRouteDest.9.1.2.3,
    ipRouteMetric1.9.1.2.3, ipRouteNextHop.9.1.2.3)

GetResponse((ipRouteDest.10.0.0.51 = 10.0.0.51),
    (ipRouteMetric1.10.0.0.51 = 5), (ipRouteNextHop.10.0.0.51
    = 89.1.1.42))

GetNextRequest(ipRouteDest.10.0.0.51,
    ipRouteMetric1.10.0.0.51, ipRouteNextHop.10.0.0.51)

GetResponse((ipRouteDest.10.0.0.99 = 10.0.0.99),
    (ipRouteMetric1.10.0.0.99 = 5),
    (ipRouteNextHop.10.0.0.99 = 89.1.1.42))

GetNextRequest(ipRouteDest.10.0.0.99,
    ipRouteMetric1.10.0.0.99, ipRouteNextHop.10.0.0.99)

GetResponse((ipRouteMetric1.9.1.2.3 = 3),
    (ipRouteNextHop.9.1.2.3 = 99.0.0.3),
    (ipNetToMediaIfIndex.1.3 = 1))
```

Die Abfrage der Tabelle läuft folgendermaßen ab: Bei der letzten Abfrage weiß der Verwalter nicht, daß die Tabelle zu Ende ist. Der Agent liefert die Werte, die in lexikographischer Reihenfolge als nächstes kommen. Wenn die Objektnamen in der Anfrage und der Antwort (`ipRouteDest` $\neq$ `ipRouteMetric1`) nicht gleich sind, erkennt der Verwalter, daß das Ende der Tabelle erreicht ist.

8.2.7 SetRequest

Die SetRequest-PDU wird verwendet, um Werte zu verändern, neue Zeilen in Tabellen einzufüen und aus einer Tabelle zu löschen. Der Aufbau einer SetRequest-PDU ist gleich mit dem einer GetRequest-PDU. Der einzige Unterschied ist, daß zu jedem Objektnamen ein Wert mit übergeben wird. Die SetRequest-PDU ist auch atomar, also werden entweder alle übergebenen Werte verändert oder keiner. Wenn kein Wert verändert wird, liefert der Agent die gleichen Fehlermeldungen wie bei der GetRequest PDU zurück (`noSuchName`, `tooBigErr`, `genError`). Zusätzlich gibt es noch die Fehlermeldung `badValue` (Falscher Wert). Diese wird zurückgegeben, wenn der übergebene Wert nicht zu dem Objekt paßt (Typ, Länge, aktueller Wert des Objektes).

Tabellenwert verändern

Um einen Tabellenwert zu ändern, der kein INDEX-Objekt ist, muß nichts Besonderes beachtet werden.

Neue Tabellenzeile einfügen

Wenn man z.B. in die `ipRouteTable` aus 2.3.3 eine Zeile mit den Werten (11.3.3.12, 9, 91.0.0.5) einfügen will, muß folgende Nachricht gesendet werden:

```
SetRequest((ipRouteDest.11.3.3.12 = 11.3.3.12),
    (ipRouteMetric1.11.3.3.12 = 9),
    (ipRouteNextHop.11.3.3.12 = 91.0.0.5))
```

Der Wert von `ipRouteDest.x` muß immer `x` sein, da dies ein INDEX Objekt der Tabelle ist. Der Name `ipRouteDest.11.3.3.12` ist dem Agenten unbekannt. RFC1212 bietet dem Agenten 3 Möglichkeiten, auf diese Anfrage zu reagieren:

1. Nachricht abweisen und Fehlermeldung NoSuchName zurückgeben.
2. Versucht die Operation auszuführen, entdeckt aber einen ungültigen Wert und gibt deshalb die Fehlermeldung badValue zurück.
3. Führt die Operation aus und erzeugt eine neue Zeile. Als Antwort liefert er (bezogen auf obiges Beispiel):

```
GetResponse((ipRouteDest.11.3.3.12 = 11.3.3.12),
    (ipRouteMetric1.11.3.3.12 = 9),
    (ipRouteNextHop.11.3.3.12 = 91.0.0.5))
```

Es wird von SNMP nicht vorgeschrieben, ob eine neue Zeile eingefügt werden soll oder nicht.

Es besteht aber auch die Möglichkeit, eine neue Zeile mit folgendem Befehl einzufügen:

```
SetRequest(ipRouteDest.11.3.3.12 = 11.3.3.12)
```

dann hat der Agent 2 Möglichkeiten zu reagieren:

1. Eine neue Zeile einfügen mit Standardwerten für die Werte, die nicht übergeben wurden.
2. Nachricht abweisen, z.B. wenn der Agent für ein Objekt der Tabelle einen Wert benötigt und keinen Standardwert verwenden kann.

Auch hier schreibt SNMP dem Agenten nicht vor, wie er reagieren muß.

Tabellenzeile löschen

Um eine Tabellenzeile aus einer Tabelle zu löschen, besitzt die Tabelle ein spezielles Objekt, welches für die Tabellenzeile auf ungültig (`invalid`) gesetzt wird.

Beispiel für die `ipRouteTable`:

```
SetRequest(ipRouteType.11.3.3.12 = invalid)
```

Ob die Zeile physikalisch gelöscht wird oder nur als ungültig markiert wird, ist implementationsabhängig. Es gibt allerdings nur zwei Tabellen in der MIB II (`ipRouteTable`, `ipNetToMediaTable`), die über ein spezielles Objekt verfügen (`ipRouteType`, `ipNetToMediaType`), um eine Zeile aus der Tabelle zu löschen. Bei den anderen Tabellen gibt es keine Möglichkeit, eine Zeile zu löschen.

Operation auf einem Agenten ausführen

Normalerweise unterstützt SNMP keine Befehlsausführung auf einem Agenten, sondern nur Get- und Set-Befehle. Man kann aber den Agenten durch Setzen bestimmter Werte Befehle ausführen lassen. Zum Beispiel kann ein Agent ein Feld `reBoot`, mit dem Initialwert 0, in seine MIB einfügen. Wenn nun ein Verwalter den Wert auf 1 setzt, bootet der Agent sein System neu und setzt den Wert von `reBoot` wieder auf 0.

8.2.8 Trap

Mit einer Trap-Nachricht kann ein Agent einen Verwalter über eine besonderen Vorfall informieren. Der Aufbau unterscheidet sich stark von den anderen PDUs.

Die Trap-PDU besteht aus folgenden Feldern:

- **PDU type**: Kennzeichnet die PDU als Trap PDU
- **enterprise**: Kennzeichnet das Netzwerkmanagement Subsystem, das den Trap erzeugt hat. (`sysObjectID` aus der System Gruppe).
- **agent-addr**: Die IP-Adresse des Agenten, der den Trap erzeugt hat.
- **generic-trap**: Einer der vordefinierten Trap-Typen.
- **specific-trap**: Ein Code, der den Trap genauer beschreibt.
- **time-stamp**: Die Zeit zwischen der letzten Initialisierung der Netzwerk-Einheit und der Trap Erzeugung.
- **variable-bindings**: Zusätzliche Informationen zur Trap-Nachricht. Die Bedeutung dieses Feldes ist implementierungsabhängig.

Mögliche Werte für das generic-trap Feld sind:

- **coldStart (0)**: Die sendende SNMP Einheit hat sich neu initialisiert, so daß sich die Konfiguration des Agenten oder die Protokolleinheit sich geändert haben könnte. Normalerweise handelt es sich um einen unerwarteten Neustart wegen einem Crash oder einem schwerwiegenden Fehler.
- **warmStart (1)**: Die SNMP-Einheit hat sich neu initialisiert, ohne dabei etwas an der Konfiguration des Agenten oder der Protokolleinheit zu ändern. Normalerweise ist dies ein Routineneustart.
- **linkDown (2)**: Signalisiert einen Fehler in einer Komunikationsverbindung des Agenten. Im ersten Element des `variable-bindings` Feld wird der Name und der Wert der `IfIndex` Instanz des betreffenden Schnittstelle übergeben.
- **linkUp (3)**: Signalisiert, daß eine Komunikationsverbindung des Agenten aufgebaut wurde. Im ersten Element des `variable-bindings` Feldes wird der Name und der Wert der `IfIndex` Instanz des betreffenden Interfaces übergeben.
- **authenticationFailure (4)**: Signalisiert, daß die sendende Protokolleinheit eine Protokoll Nachricht empfangen hat, die nicht die richtige Berechtigung hatte.
- **egpNeighbourLoss (5)**: Signalisiert, daß der EGP (**e**xternal **g**ateway **p**rotocol) Nachbar, für den die sendende Protokolleinheit ein EGP Partner war, sich abgehängt hat und dadurch die Partnerbeziehung nicht mehr besteht.
- **enterpriseSpecific (6)**: Signalisiert, daß die sendende Protokolleinheit einen herstellerspezifischen Fehler erkannt hat. Das `specific-trap` Feld enthält die Fehlernummer des Traps.

Anders als die anderen Nachrichten erhält die Trap Nachricht keine Rückantwort.

8.3 Probleme in der Praxis

Es gibt große Unterschiede bei der Unterstützung des SNMP. Manche Geräte lassen nur Lesezugriffe auf ihre MIB zu oder sie lassen mit dem Gemeinschaftsnamen 'public' den Zugriff auf die gesamte MIB zu (lesen und schreiben). Dadurch kann im ersten Fall überhaupt kein schreibender Zugriff auf die MIB erfolgen, und im zweiten ist die MIB dagegen total ungeschützt.

Bei einem Test von Routern und Bridges wurde der Strom der Geräte abgeschaltet und dann wieder eingeschaltet. Normalerweise müßten die Geräte einen Cold-Start Trap senden. Die wenigsten Geräte waren in der Lage, diesen Trap zu erzeugen. Dies macht das Verhalten des Netzwerks sehr schwierig, da der Verwalter den Kaltstart des Gerätes nicht mitgeteilt bekommt.

Die MIB- und MIB-II-Spezifikation schreibt vor, daß ein Gerät alle Objekte einer Gruppe unterstützen muß. Ein Gerät kann also nicht nur einen Teil der Objekte einer Gruppe unterstützen. Leider haben ein paar Hersteller die Idee gehabt, Objekten, die nicht unterstützt werden, den festen Wert 0 zu geben, das heißt, bei jedem Zugriff wird der feste Wert 0 zurückgeliefert. Dies kann aber, zum Beispiel bei einem Fehlerzähler, sehr unvorteilhaft sein.

8.4 Einschränkungen von SNMP

Bei Verwendung von SNMP muß man folgende Einschränkungen beachten:

- SNMP eignet sich nicht für sehr große Netzwerke, da der Kommunikationsaufwand sehr hoch ist (man muß eine Nachricht senden, um eine Nachricht mit Informationen zu erhalten).
- SNMP ist nicht geeignet, um große Datenmengen, wie z.B. eine ganze Routing Tabelle, zu empfangen.
- SNMP-Traps werden nicht bestätigt. Deshalb kann ein Agent, der einen wichtigen Trap gesendet hat, sich nicht sicher sein, ob die Nachricht angekommen ist.
- SNMP unterstützt nur eine einfache Berechtigungskontrolle.
- SNMP unterstützt keine direkte Befehlsausführung auf einem Agenten.
- SNMP unterstützt keine Verwalter - Verwalter Kommunikation. Dadurch kann ein Verwalter nichts darüber erfahren, welche Geräte von einem anderen Verwalter verwaltet werden.

8.5 Literaturverzeichnis

[Sta93g]

Kapitel 9

Secure SNMP

N.N.

Dieser Vortrag fiel wegen Absage des Referenten aus.

9.1 Literaturverzeichnis

[Sta93f]

Kapitel 10

Remote Network Monitoring

Tobias Rothfuchs

10.1 Grundkonzept und Ziele des Remote Monitoring

Die nun im folgenden vorgestellte RMON-Spezifikation ist als wichtigste Ergänzung zu den SNMP-Standards SMI, MIB und SNMP anzusehen.

Die in den vorhergehenden Kapiteln vorgestellte MIB-II ist nicht ausdrücklich zur Leistungs- und Fehlerüberwachung in lokalen Netzwerken geeignet. Durch Integration der RMON MIB über SNMP bzw. MIB-II in Netzwerk-Devices wird für einen Netzwerkmanager eine effiziente Überwachung ermöglicht. In einer Netzwerkumgebung kann diese Überwachung mit einer oder mehreren zentralen Managementstationen und sogenannten Remote Monitors – in jedem Teilnetzwerk wird ein für das Monitoring geeignetes Device mit einigen durch RMON ermöglichten Überwachungsfunktionen ausgestattet – stattfinden. Durch das selbständige Arbeiten und der Ereignismeldung – dem Off-Line-Operieren, durch Präventivüberwachung und Problemerkennung, d.h. automatischem Diagnostizieren des Leistungsverhaltens bei passiver Beobachtung des Verkehrs, und durch lokale Datenanalyse der Remote Monitors werden Kommunikationskosten erspart und zentrale Managementstationen, die zur Erhöhung der Zuverlässigkeit, für verschiedene Aufgaben oder Organisationsstellen im Netzwerk vorhanden sein können, in ihrer Auswertungstätigkeit entlastet.

10.2 Monitorsteuerung, Mehrfach- und Tabellenmanagement

Die Implementierung des Monitors kann zum einen als dediziertes Device und zum anderen als Funktion auf einem System, das das Monitoring unterstützt erfolgen. Die Integration kann u.a. auf einer Workstation, einem PC, Server oder Router erfolgen. Zur Steuerung der Monitore werden die einzelnen Funktionen des Monitorings, die durch RMON unterstützt werden, in Form von Kontroll- und Datentabellen konfiguriert. Hierbei entsprechen einzelne Terme in den Kontrolltabellenzeilen den auszuführenden Funktionen, die dadurch definiert und implementiert werden. Mit Organisation der RMON MIB in funktionale Gruppen, werden diese Gruppen i.a. durch je eine Kontrolltabelle, in der die gewünschten Funktionen festgelegt werden, und mehrere Datentabellen, in denen die vom Monitoragent ermittelten Verkehrsdaten abgelegt werden, beschrieben. Aktionsaufrufe seitens der Managementstation werden durch Objektwertänderungen – Objekte repräsentieren zum Teil Aktionsstati – in den Kontrolltabellen veranlaßt.

Ein eventuelles Mehrfachmanagement in der Netzwerkumgebung, d.h. ein Arbeiten mit mehreren Managementstationen, kann zum Erreichen der Kapazitätsgrenzen, zum Blockieren von Monitorresourcen für wichtigere Arbeiten und bei Absturz von Managementstationen zum Nichtwiederfreigeben von Recourcen führen. Diese Probleme werden durch Zuordnung eines Objektes, dem `OwnerShipLabel`, das die funktionsbesitzende Managementstation identifiziert, zu jeder Kontolltabellenzeile gelöst. Hierdurch ist ein Netzwerkoperateur in der Lage nicht benötigte Monitorresourcen freizugeben. Diese Freigabe kann auch durch eine Managementstation bei einer Reinitialisierung nach deren Absturz erfolgen. Durch Zuordnung dieses Objekts ist es einer Managementstation möglich, bereits vorhandene Monitoringfunktionen mittels Scanning über die jeweilige Kontrolltabelle zu erkennen und durch Beobachtung der zugeordneten Datentabellen die entsprechenden Verkehrsdaten zu erhalten.

Zum Tabellenmanagement sind in der RMON-Spezifikation, aufgrund der beim Mehrfachmanagement möglichen Problematiken, die zum SNMP-Rahmen zusätzlichen Datentypen `OwnerString`, als Objekttyp des oben vorgestellten `OwnerShipLabels`, und `EntryStatus` hinzugefügt. Zusätzlich ist ein Index-Objekt zur Referenzierung von Datentabellenzeilen zu den zugehörigen Kontrolltabellenzeilen und zur Zeilenidentifizierung ergänzt. Eine Zeilenaddition wird wie unter SNMP durch Sendung einer SetRequest-PDU mit den entsprechenden Objektidentifikatoren und deren Instanziierungswerten, mit anschließender Konsistenzkontrolle bezüglich der Spezifikation durch den Monitoragenten, durchgeführt. Das `EntryStatus`-Objekt wird zur Zeilenaddition unter Konkurrenz bei Mehrfachmanagement benötigt, da hier Konflikte bei Anforderung

gleicher Funktionen, d.h. Zeilen mit gleichen Parametern, durch unterschiedliche Manager auftreten können. Mit Durchführung der sogenannten "RMON Polka" lassen sich Kontrolltabellen konsistent anlegen:

1. Falls eine Managementstation eine nicht bereits vorhandene Zeile erzeugen will, wird der zugehörige `EntryStatus` auf `createRequest` gesetzt und die Zeile kreiert.

2. Unmittelbar nach Abschluß der Zeilenkreation setzt der Agent den `EntryStatus` auf `underCreation`.

3. Der Status underCreation bleibt in allen Zeilen erhalten, solange die Managementstation weitere Funktionen anfordert. Hiernach setzt die Managementstation den `EntryStatus` auf `vaild`.

4. Falls mit Status `createRequest` eine Zeile kreiert werden soll, die bereits existiert, wird der anfordenden Managementstation ein Fehler zurückgegeben.

Mit Übergang auf den `EntryStatus valid` beginnt der Agent mit der so spezifizierten Überwachung.

Zeilen werden durch Setzen des `EntryStatus` zu `invalid` und Senden einer entsprechenden SetRequest-PDU gelöscht oder mittels Laden neuer Zeilenwerte modifiziert.

10.3 Die RMON MIB

10.3.1 Struktur der RMON MIB

Die RMON MIB ist in der MIB-II als Teilbaum mit Identifikator 16 aufgenommen. Die MIB ist in neun Gruppen gegliedert:

1. `statistics:` Dient zur Erfassung von low-level Nutzungs- und Fehlerstatistiken,

2. `history:` Dient zur periodischen Aufzeichnung statistischer Daten,

3. `alarm:` Dient zum Festlegen von Erfassungszeiträumen und Alarmschwellwerten,

4. `host:` Dient zum Erfassen des Hostverkehrs in einzelnen Teilnetzwerken,

5. `hostTopN:` Dient zur geordneten Darstellung von stark-aktiven Hosts,

6. `matrix:` Dient bei Paaren miteinander kommunizierender Hosts zur Informationsdarstellung,

7. `filter:` Dient zur Beobachtung bestimmter filterpassierender Packete,

8. `packet capture:` Dient zur Regelung der Datenübertragung zu einer Managementstation,

9. `event:` Dient zur Aufzeichnung der Ereignisse durch den Agenten

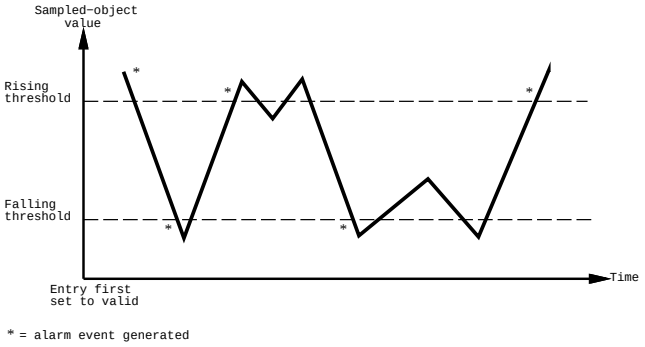


Abbildung 10.1: Generierung von Alarm Events

Jede dieser Gruppen wird zur Speicherung der vom Monitor erfassten Daten und Statistiken genutzt. Da Monitore mehrere physikalische Interfaces haben können, ist ein Anschluß an mehrere Teilnetzwerke möglich. Die Nutzung der einzelnen Gruppen ist unter Einschränkungen – bspw. setzt die Nutzung der Packet-Capture-Gruppe die Implementierung der Filter-Gruppe voraus – optional.

10.3.2 Die Statistics-Gruppe

Die Statistics-Gruppe besteht aus einer einzigen Tabelle, die Kontrollparameter und Daten gleichzeitig hält und in jeder Zeile Statistiken bzgl. eines Teilnetzwerks in Form von Zählern führt. Erfaßt werden u.a. Auslastung durch Erfassen der einzelnen Packetgrößen, CRC-Fehler und Over- bzw. Undersizepackete. Zur Zeit ist die Definition nur für Ethernet-Interfaces ausgelegt.

10.3.3 Die History-Gruppe

In der History-Gruppe, die sich in eine Kontroll- und Datentabelle gliedert, werden durch Angabe der `DataSource`, die das zu betrachtende Teilnetzwerk identifiziert, der `BucketsRequested` bzw. `BucketsGranted` und `Interval` – die Anzahl der Erfassungsintervalle und die Erfassungszeitdauer in Sekunden pro Interval – die Rahmen zur periodischen Aufzeichnung des Verkehrs gesetzt. Bei Gewährung der Defaultwerte durch den Monitor werden bspw. in Form eines zirkulären Puffers die letzten 50 Zeilen der Datentabelle gehalten. Dabei werden pro Zeile die Verkehrsstatistiken der letzten 1800 Sekunden, d.h. der letzten 30 Minuten, erfaßt. Die History-Gruppe findet bspw. ihre Anwendung in der Beobachtung von Teilnetzwerken in kurzen Zeiträumen – um plötzliche Wechsel in Verkehrshäufigkeiten zu erkennen – und, zur Betrachtung des globalen Netzwerkverhaltens, in längeren Erfassungszeiträumen. Diese Anwendung wird durch die Definition mehrerer Erfassungsintervalle pro Teilnetzwerk ermöglicht.

10.3.4 Die Alarm-Gruppe

Die Alarm-Gruppe wird zur Definition von Schwellwerten zur Alarmauslösung verwendet. Neben den Objekten `Interval`, zur Festlegung des Beobachtungszeit-

raums, `Variable` – Objektidentifikator der zu beobachtenden Variable der korrespondierenden lokalen MIB – dienen `SampleType`, `StartupAlarm` und `Rising-` bzw. `FallingTreshold` zur Festlegung der Kriterien für eine Alarmauslösung. Mit `SampleType AbsoluteValue` wird ein laufender Vergleich mit den gesetzten Schwellwerten durchgeführt und bei Über- bzw. Unterschreitung der Alarm ausgelöst. Mit `SampleType DeltaValue` wird hingegen ein Vergleich mit der Differenz zum Wert im letzten Beobachtungsintervall durchgeführt. Dieser Typ eignet sich bei kleinen Beobachtungsintervallen zum Erkennen zwischenzeitlicher Spitzen im Aufkommen bzgl. der betrachteten Variablen. Der `StartupAlarm` legt den Zeitpunkt einer ersten Alarmauslösung, bspw. bei `risingAlarm` Triggerung des Alarms erst bei Überschreiten des `RisingTresholds`, fest. Ein Alarm wird durch eine neue Zeile in der Alamtabelle definiert. Um die Last des Monitors durch eventuell zu bestimmten Zeitpunkten häufig eintretenden Überschreitungen von Schwellwerten bzgl. der Alarmmeldung zu reduzieren, findet eine Alarmtriggerung erst wieder nach erfolgtem Unter- bzw. Überschreiten der entgegengesetzten Schwellwerte statt. Dieser Mechanismus wird in Abbildung 10.1 verdeutlicht.

Mit Festlegung eines Alarmgenerierungsstatus – `rising-alarm-state` und `falling-alarm-state` – wird dieser Mechanismus in der RMON Spezifikation auch als *Hysterese Mechanismus* bezeichnet.

Die Nutzung der Alarm-Gruppe selbst erfordert die Implementierung der Event-Gruppe, da das Ereignis des Alarms in den dortigen Tabellen aufgezeichnet wird.

10.3.5 Die Host-Gruppe

In der Erfassung statistischer Daten wie bspw. die Anzahl ein- und ausgehender Datenpackete bei spezifischen Hosts findet die Host-Gruppe ihren Einsatz. Die Gruppe besteht aus einer Kontrolltabelle und zwei Datentabellen. In der Kontrolltabelle können über `DataSource` wiederum verschiedene, an den Interfaces des Monitors befindliche Teilnetzwerke erreicht werden. In den Datentabellen werden durch den Monitor automatisch entdeckte und hinzukommende Hosts durch Zeilenaddition und Speicherung der statistischen Daten erfaßt. Ein Host wird hierbei durch seine Aktivität im Netz erkannt. Im Falle der Resourcenknappheit werden die Zeilen der Datentabellen vom Monitor wiederum als zirkulärer Puffer verwaltet. Die Zeilen der Datentabelle `hostTable` werden vom Monitor nach aufsteigenden MAC-Hostaddressen angelegt. Mit einer Sortierung dieser Art ist eine Übersicht über örtlich benachbarte Hosts und deren Auslastung leicht möglich. Mit der Sortierung nach der Zeit des Eintretens in das Netz, d.h. durch den Zeitpunkt des Beginns von Netzaktivitäten eines Hosts, in der Datentabelle `hostTimeTable`, ist hingegen eine Informationsgewinnung über hinzugekommene Hosts und eine effiziente Übertragung von Informationen mittels GetRequest- oder GetNextRequest-PDU durch Kenntnis der Tabellengröße einfacher. Physikalisch kann hier durchaus eine Tabelle implementiert werden, wobei dann die beiden Datentabellen durch logische Views realisiert werden.

10.3.6 Die HostTopN-Gruppe

Die HostTopN-Gruppe dient zur Herstellung sogenannter Top-N-Reports. Dies sind in sortierter Form Informationen über Hosts, die nach einem bestimmten Kriterium im Netz die stärksten Aktivitäten zeigen. Die mit `GrantedSize` zur Verfügung gestellte Größe N gibt hierbei die Anzahl der gelisteten Hostinformationen an. Beispielsweise ist es mittels dieser Funktion möglich, eine Liste über die N Hosts zu erhalten, die in einem gewissen Zeitintervall die meisten Daten übermitteln. Die hierfür benötigten Daten stammen direkt aus den Tabellen der Host-Gruppe. Die Top-N-Reports können für jedes durch den Monitor betrachtete Teilnetzwerk angefordert werden und werden dann jeweils durch den Monitor unter Löschung der vorherigen Tabelle neu bestimmt.

10.3.7 Die Matrix-Gruppe

Mit der Implementierung der Matrix-Gruppe ist für einen Netzwerkmanager ein Überblick zu Paaren miteinander kommunziernder Hosts auf einfache Art und Weise zu erhalten. Einen Einsatz kann diese Gruppe daher bspw. bei der Fragestellung "Welche Devices haben den meisten Gebrauch eines Servers ?" finden. Neben einer Kontrolltabelle bestehen hier wiederrum zwei Datentabellen: Zum einen die SD-Tabelle – `Source-Destination`-Table – für eine rasche Informationsgewinnung bei Fragen der Art "Welcher Verkehr fließt von einem bestimmten Host zu allen anderen Hosts ?" und zum anderen die DS-Tabelle – `Destination-Source`-Table – für Informationen bzgl. des Verkehrs von allen Hosts zu einem bestimmten Host. Die Zeilen in den beiden Datentabellen werden wiederum vom Monitor bei Neuauftreten von Kommunikation zwischen Hosts ergänzt und sollten bei Resourcenknappheit nach der Spezifikationsempfehlung mit der Least-Recently-Used-Methode entfernt werden.

10.3.8 Die Filter-Gruppe

Unter Einsatz der Filter-Gruppe ist es einer Managementstation möglich, den Monitor zur Beobachtung selektierter Packete zu veranlassen. Die Selektion kann hierbei bzgl. der Packetdaten durch Datenfilter und/oder der Packetstati durch Statusfilter erfolgen. Die verschiedenen Filter können mittels logischem UND und ODER zu Kanälen verknüpft werden. Zur Selektion wird hierfür eine Filter- und Kanal-Logik zugrundegelegt.

Abbildung 10.2 soll einen Teil des Tests zur Filterpassierung mittels der Filter-Logik bei einem Datenfilter verdeutlichen.

Hier wird durch `filterPktDataOffset` die gewünschte Position, d.h. der Beginn des Bitmusters, im zu testenden Packet festgelegt. Der input wird bitweise

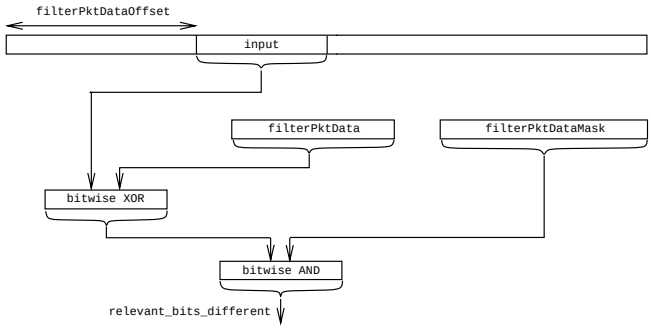


Abbildung 10.2: Ausschnitt der Datenfilter-Logik

mit dem zu testenden Bitmuster `filterPktData` durch ein logisches XOR-Element verglichen. Dies führt an nicht übereinstimmenden Bitpostionen zu einem Bitwert von 1. Durch anschließenden Vergleich mit der Maske `filterPktDataMask`, die mittels dem Bitwert 1 für den Test relevante Bitpositionen festlegt (der Bitwert 0 entspricht dann don't-care-Bits), werden die Bitpositionen bestimmt, die an den relevanten Stellen des Packetdatenteils nicht dem entsprechenden Muster genügen.

Durch nachgeschaltete Vergleiche mit einer `filterDataNotMask`, in der durch die einzelnen Bits die Positionen spezifiziert sind, an denen ein Test auf Matching oder Mismatching stattfindet, ergibt sich der abschließend zusammengefaßte dreistufige Test:

1. Test auf Länge des Packets. Das Packet muß mindestens die im Muster `filterPktData` vorgegebene Länge zuzüglich `filterDataOffest` haben.

2. Test auf Matching an den mittels `filterPktDataMask` festgelegten relevanten Bitpositionen durch `filterDataNotMask`.

3. Test auf Mismatching an den mittels `filterPktDataMask` festgelegten relevanten Bitpositionen durch `filterDataNotMask`.

Ein Paket passiert den Filter nur im Falle des Passierens aller drei Testteile.

Kanäle werden wie oben erläutert durch Filterkombinationen definiert. Hierbei werden durch das Objekt `channelAcceptType` zwei Möglichkeiten zur Steuerung gegeben: Im Falle der Objektausprägung `acceptMatched` passieren Packete den Kanal, wenn sie sowohl den Daten- als auch den Statusfilter eines diesem Kanal zugeordneten Filter passieren. Liegt hingegen `acceptFailed` vor, passieren Packete den Kanal nur, falls in allen assoziierten Filtern der Test im Daten- oder Statusfilter fehlschlägt. Nach Passieren eines Kanals führt der Monitor eine abschließende Kanaloperation durch. Hier wird zum ersten im Objekt Matches die Anzahl der passierten Packete durch Inkrementierung eines Zählers aktualisiert. Falls eine Eventbehandlung gewünscht ist – in diesem Falle hat `channelDataControl` den Wert on – folgt hiernach eine Erfassung des Events in der Event-Gruppe. Hierbei ist mittels dem Objekt `channelEventStatus` und dessen Wert `eventReady` eine Rückmeldung der Managementstation zur Bestätigung des Kanalereignisses möglich. Durch den Wert `eventAlwaysReady` wird hingegen auf diese Bestätigung verzichtet und der Monitor ist zur unabhängigen Eventerfassung in der Lage.

Die Filter-Gruppe selbst besteht aus zwei Kontrolltabellen. Die Zeilen der `channelTable` definieren hierbei einen Kanal, der sich – durch referenzierte Zeilen – aus den einzelnen Filtern der `filterTable` zusammensetzt.

10.3.9 Die Packet-Capture-Gruppe

Die Packet-Capture-Gruppe dient zu einer eventuellen Speicherung der Packete, die Kanäle der Filter-Gruppe passieren, und regelt die Übertragung einzelner Packete – das Downloading – vom Monitor zur Managementstation. Hierbei werden in der `bufferControlTable` u.a. mit `ChannelIndex` der Kanal, der Quelle der zu speichernden Packete ist, festgelegt und in `CapturedPkts` die Anzahl der in den Zeilen der `captureBufferTable` unter `BufferPacketData` abgelegten aktuell gespeicherten Packete gehalten. Bei Erreichen der Pufferkapazität läßt sich mittels `FullAction wrapWhenFull` durch Löschen früher gespeicherter Packete wieder freier Pufferplatz schaffen. Ist der Wert von `FullAction lockWhenFull` werden neuankommende Packete verworfen. Für den Fall, daß die gespeicherten Packetdaten für einen einzigen SNMP-Retrival mittels Get- oder GetNextRequest zu umfangreich sind, läßt sich durch die Objekte `DownloadSliceSize`, das die Länge der zu übertragenden Daten in Oktetten beschreibt, und `DownloadOffset`, das den Offset des ersten zu übertragenden Oktetts im Puffer angibt, ein sukzessives, vollständiges Downloading – eben durch gleichmäßiges Erhöhen des Offsetwerts – erreichen.

10.3.10 Die Event-Gruppe

Die Event-Gruppe unterstützt die Definition von Events. Events werden hierbei durch bestimmte Konditionen in der MIB ausgelöst und können ihrerseits Aktionen triggern und SNMP-Traps erzeugen. Der Zusammenhang mit den anderen Gruppen der RMON MIB wurde bereits bei deren Beschreibungen aufgezeigt. Bei Eintreten eines Events wird gemäß dem Objektwert von eventType in der Kontrolltable das eingetretene Ereig-

nis in die Datentabelle logTable eingetragen (`log`), ein SMNP-Trap an die in `eventCommunity` spezifizierten Managementstationen übermittelt (`snmp-trap`), oder bei `log-and-trap` beide vorher beschriebenen Aktionen durchgeführt. Als mögliches Einsatzbeispiel der Event-Gruppe mag die Regelung der Kanalaktivitäten in der Filter-Gruppe in Abhänigigkeit der bereits erfaßten Packete dienen. Durch die Eventtriggerung kann eine Managementstation Kanäle entsprechend an- und ausschalten, um so Detailbetrachtungen zu ermöglichen.

10.4 Praktische Aspekte

Zum Abschluß der Betrachtungen im Bereich des Remote Network Monitoring sollen noch einige mögliche Problematiken und Einsatzmöglichkeiten der RMON MIB kurz aufgeführt werden.

Durch die im vorherigen Abschnitt vorgestellte breite Funktionalität und Mächtigkeit der RMON MIB kann bei extensivem Funktionseinsatz ein Remote Monitor rasch überlastet werden. Auch führt eine häufige Informationsübertragung zur Managementstation zu einer Reduzierung der Netzleistung. Aus diesen Gründen ist durch sorgfältige Funktionsauswahl und lokale Verkehrsanalyse auf dem Monitor, mit anschließender zusammengefasster Ergebnisübermittlung, eine effizientere Netzwerküberwachung erreichbar. Besonders im Bereich der Filter-Gruppe sollten Restriktionen bzgl. der Filteranzahl zur Lastreduktion angewendet werden.

Neben dem Einsatz zur Lokalisierung von bestimmten Ereignissen im Netzwerk mittels Filter- und Packet-Capture-Funktionen kann mit RMON ein Netzwerkverzeichnis, durch automatisches Erkennen von aktiven Hosts, effizient erhalten werden.

Die Integration von RMON findet bei Einsatz in ausgelasteten Netzen vorzugsweise in dedizierten Devices und bei wenig ausgelasteten Netzen auch als Funktion auf nichtdedizierten Devices statt.

Diese Aspekte machen deutlich, daß erst durch Einsatz der RMON MIB als wichtigster SNMP-Zusatz mit den zur Verfügung gestellten Funktionen zum Remote Monitoring eine effiziente Netzwerküberwachung und Profilgewinnung möglich ist.

10.5 Literaturverzeichnis

[Sta93e]

Kapitel 11

SNMP Version 2

Bui Anh Tuan

11.1 Motivation

Nach der Veröffentlichung der Standarddokumente des SNMP im Juli 1992 wurde herausgefunden, daß SNMP Mängel an Funktionalität, Leistung sowie Sicherheit hat und damit seine Anwendung auch beschränkt ist. Folglich kann SNMP nur minimale Netzwerke unterstützen. Aus diesen Gründen wurde versucht, ein Protokoll zu entwickeln, das auf SNMP basiert und gegenüber SNMP mehr Funktionalität, mehr Leistung, so wie mehr Sicherheit besitzt, damit es auch größere Netzwerke unterstützen kann. Dieses Protokoll wurde SNMPv2 genannt.

11.2 Gliederung von SNMPv2

SNMPv2 wird in fünf Kategorien gegliedert:

11.2.1 Structure of Management Information (SMI)

Der Aufbau SNMPv2 SMI ist auf der SNMPv1 SMI basiert. Außerdem umfaßt SMIv2 gegenüber SMIv1 mehr Sorgfältigkeit bei der Spezifikation und Dokumentation von verwalteten Objekten. SNMPv2 SMI wird in vier Teilen aufgebaut:

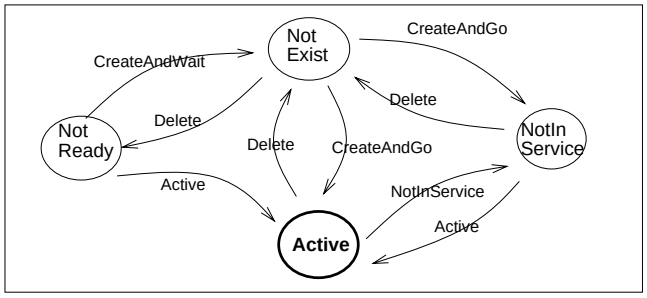


Abbildung 11.1: Übergangsdiagamm RowStatus

Objektdefinition

Wie in SNMP SMI können die verwalteten Objekten in SNMPv2 mit Objektdefinition beschrieben werden.

In SNMPv2 SMI werden einige Objekttypen zusätzlich eingeführt:

- bit string: ist ein Aufzählungstyp und wird beispielsweise für die Erkennung von Vorhanden von Charakteristika eingesetzt.
- Counter64: ist eine Zähler mit der Obergrenze von 2^{64}-1. Dieser Typ wird zum Beispiel verwendet, um die Anzahl von Paketen in schnellen Netzwerken anzugeben.
- UInteger32: ist ein positive Zähler mit Obergrenze von 2^{32}-1. Dieser Typ wird beispielsweise verwendet für die Anzahl von Benutzern, Hauptspeicher, usw.
- max_access: gibt den maximalen möglichen Zugriffen auf die verwalteten Objekte an: Im Vergleich mit SNMP `MAX--ACCESS` erlaubt SNMPv2 `MAX--ACCESS` einen zusätzlichen Zugriff auf das verwaltete Objekt:
  - `read--create`: Lesen, Schreiben und die Neu–Instanzierung von Objekten

SNMPv2 Tabellen

In SNMPv2 werden die Tabellen für komplexe Informationen verwendet. Die Definitionen der Tabellen sind schon in SNMP zu finden. Im Gegensatz zu SNMP ist es in SNMPv2 möglich, die Tabellen mit zusätzlicher Typdefinition `AUGMENTS` zu erweitern. Außerdem existiert es in SNMPv2 zwei Operationen, mit denen kann die Tabellen erzeugen bwz. löschen:

- Tabellen erzeugen (`CreateAndGo` , `CreateAndWait`)
- Tabellen löschen (`Delete`)

Erzeugen und Löschen der Tabellenzeilen funktionieren nach dem Zustandübergangsdiagramm RowStatus (siehe Abb. 11.1).

Die Werte von RowStatus teilen sich in zwei(nicht disjunkte) Gruppen: **Befehle** und **Zustände**. Zustände können nur gelesen werden, Befehle können nur geschrieben werden.

Folgende Zustände gibt es:

- `Not Exist`: Keine Objekt der Zeile wurde instanziiert.
- `Active`: Die Zeile ist komplett und wird vom Agent benutzt.
- `NotInService`: Die Zeile ist zwar komplett, im Moment jedoch "außer Betrieb".
- `NotReady`: Die Zeile ist nicht komplett und kann auch nicht benutzt werden.

Des weiteren existieren folgende Kommandos:

- `Active`: Versetzt eine (komplette) Zeile in den aktiven Zustand. Wurden noch nicht alle Instanzen der Spalten gesetzt, kann der Agent entweder die definierten Standardwerte einsetzen oder einen Fehler zurückmelden.
- `NotInService`: Nimmt eine aktive Zeile "außer Betrieb".
- `CreateAndGo`: Erzeugt eine neue Zeile und versucht sie in den Zustand `active` zu versetzen. Falls dies wegen fehlender oder inkonsister Werte nicht möglich ist, wird der Fehler `inconsistentValue` zurückgeliefert.
- `CreateAndWait`: Erzeugt ebenfalls eine neue Zeile, versetzt sie jedoch, je nachdem ob alle Spaltenobjekte instanziiert wurden, entweder in den Zustand `NotReady` (es fehlt noch Werte) oder `NotInService` (alle Werte vorhanden).
- `Destroy`: Löscht eine Zeile aus der Tabelle.

Um die Zeilenerzeugen und –löschen zu unterstützen, müssen innerhalb einer Tabelle Objekte mit der `SYNTAX RowStatus`, `MAX-ACCESS read-create` definiert werden.

evalStatus Object-Type

SYNTAX	RowStatus
MAX–ACCESS	read-create
STATUS	current
DESCRIPTION	"The status column used for creating, modifying , and delecting instance of the columnar objects in the evalution table"
DEFVAL	{active}

:= {evalEntry4}

11.2.2 Protokoll Operationen

Es existieren in SNMPv2 insgesamt drei Zugriffstypen auf die Managementinformationen:

1. Manager-agent `request-response`:
2. Agent-Manager `unconfirmed`: Die oben genannten Zugrifftypen sind auch in SNMPv1 zu finden.
3. Manager-Manager `request-response`: Dieser Zugriffstyp ist neu in SNMPv2. Mit diesem Typ können Manager Informationen zwischeneinander austauschen.

In SNMPv2 sind viele Operationen vorhanden, die bezüglich ihrer Funktionalität und Semantik identisch wie die Operationen von SNMPv1. Insbesondere gilt dies für:

- `GetNextRequest` PDU,
- `SetRequest` PDU

Zwischen `GetRequest PDU` von SNMP und `GetRequest PDU` von SNMPv2 existiert ein Unterschied: Bei der `GetRequest` PDU von SNMP können entweder alle definierten Objektenwerte, die als Parameter mitgegeben werden, zuruckgeliefert werden, oder, wenn ein Objektwert undefiniert ist, keine Objektwerte zurückgeliefert. In SNMPv2 werden bei `GetRequest` PDUs immer die einzelnen Objektwerte (ggf. mit ihren Zuständen) zurückgeliefert.

In SNMPv2 definiert zusätzlich zwei Operationen:

- `InformRequest`: Mit dieser Funktion können zwei Manager Informationen miteinander austauschen.
- `GetBulkNextRequest`: ist eine wichtige Erweiterung in SNMPv2. Mit dieser Operation werden alle mögliche Objektwerten auf einmal nach lexikographischer Reihenfolge selektiert. Diese Operation kann also als eine Mischung aus den Operationen `GetRequest` und `GetNextRequest` betrachtet werden.

11.2.3 SNMPv2 Management Information Base (MIBv2)

MIBv2 ist eine Ansammlung von Objekten, die die Eigenschaften von SNMPv2 beschreiben (siehe Abb. 11.2). Diese Objekten werden in fünf Gruppen geteilt.

1. SNMPv2 Statistikgruppe: enthält die Objekte, die die rein statistischen Werte beschreiben.

   Bsp: das Objekt `snmpStatsPackets` gibt die angekommenen Nachrichten von dem TransportService an.

2. SNMPv1 Statistikgruppe: enthält die Objekte, die schon in der SNMPv1-Statistikgruppe implemmentiert sind.

   Bsp: das Objekt `snmpV1BadCommunityNames` gibt die Anzahl von SNMP Nachrichten an, die zu einem SNMPv2 Agenten gelangten, deren Kommunikationsname aber nicht bekannt ist.

3. `Object Resource`-Gruppe: dient zur Beschreibung von den Objekten, die unter der dynamischen Konfiguration eines Managers stehen.

   Bsp: Das Objekt `snmpORTable` ist die Tabelle von dynamischen Konfigurationsobjekten in einem Agenten.

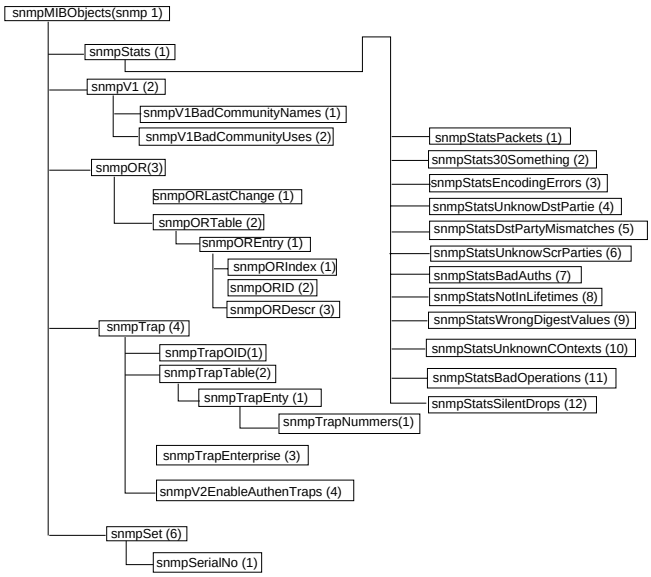


Abbildung 11.2: SNMPv2 MIB

4. `Trap`-Gruppe: enthält die Objekte, die die Ereignisse zwischen Agent und Manager beschreiben.

   Bsp: Das Objekt `snmpTrapOID` ist die Objektidentität von dem Ereignis bei dem Senden.

5. `Set`-Gruppe: besteht aus einem einzigen Objekt, welches dafür sorgt, daß Objektdaten konsistent bleiben, beispielsweise bei gleichzeitigem Zugriff von verschiedenen Managern auf ein Objekt. Dieses Objekt ist `snmpSetSerialNo`.

11.2.4 Manager–zu–Manager MIBv2

ist eine Menge von Objekten, die das Verhalten von Managern beschreiben. Diese MIB besteht aus zwei Gruppen, der Alarmgruppe, und der Ereignisgruppe. Die beide Gruppen sind schon aus dem RMON-Standard bekannt.

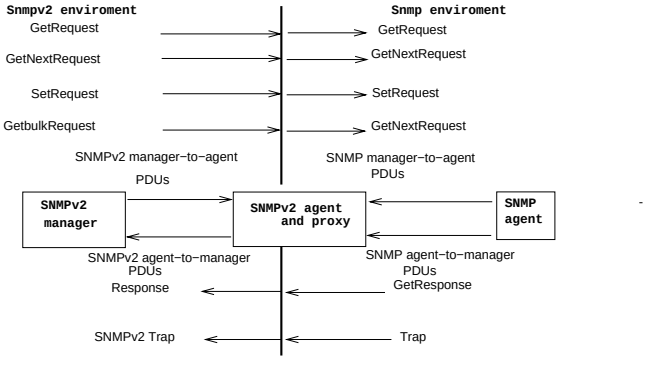


Abbildung 11.3: Coexistence by Means of Proxy Agent

11.2.5 Conformance Statements

Hier handelt es sich um eine minimale Anforderungen sowohl an die Objektdefinitionen als auch Objektimplementierungen.

Dafür gibt es drei Makrodefinitionen:

Objektgruppe–Makro

Hier handelt es sich um eine Gruppierung von Objekten, die in einer Beziehung stehen.

```
snmpV1Group OBJEKT-GROUP
    OBJECTS      {snmpV1BadCommunityNames,
                  snmpV1BadCommunityUses}
    STATUS       current
    GROUPS       snmpV1Group
    DESCRIPTION  "A collection of objects providing basic
                  instrumentation of an SNMPv2 entity which
                  also implements SNMPv2."
    := {snmpMIBGroups 2}
```

Die OBJECT-Klausel enthält die zu einer Gruppe zusammenzufassenden Objekte.

Die DESCRIPTION–Klausel ist eine verbale Beschreibung der Gruppe und soll eventuell eine Beschreibung von Beziehung zu anderen Gruppen beinhalten.

Modulübereinstimmung

Hier handelt es sich um die Festlegung minimaler Anforderungen an Objekte der MIB.

```
snmpMIBCompliance MODULE–COMPLIANCE
    STATUS              current
    MANDATORY-GROUPS    {snmpStatsGroup, snmpORGroup
                         snmpTrapGroup, snmpSetGroup}
    GROUPS              snmpV1Group
    DESCRIPTION         "The snmpV1 group is mandatory only
                         for those SNMPv2 entities which also
                         implement SNMPv1."
    := {snmpMIBCompliances 1}
```

- MANDATORY–GROUPS: Diese Klausel kann höchstens einmal vorkommen. Die Objekte in dieser Gruppe sind unbedingt zu implementieren.

- GROUP: Diese Klausel kann mehrfach vorkommen. Die Objekte einer so bezeichneten Gruppe sind entweder nur unter bestimmten Bedingung, die in einer DESCRIPTION-Klausel angeben werden müssen, notwendig; oder sie sind optional.

11.2.6 Koexistenz mit SNMP

Um die beiden SNMPv1 und SNMPv2 mit einander kommunizieren zu können , benötigt es ein Manager, der die Nachrichten von einer Protokoll in anderer Protokoll konvertiert, oder die Nachrichten von den beiden Protokollen verstehen.

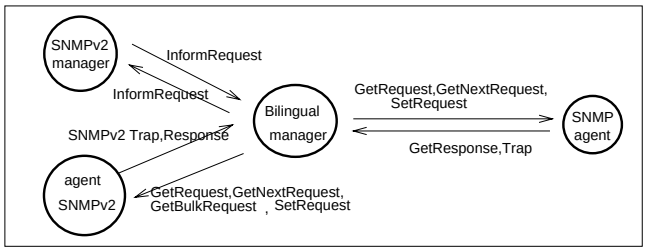


Abbildung 11.4: Coexistence by Means of Bilingual Manager

1. Möglichkeit: Aus der Abbildung 11.3 ist ersichtlich, daß die Funktionen `GetRequest`, `GetNextRequest`, `SetRequest` von SNMPv2 werden durch Stellvertreter (Proxy) in SNMP agent gesendet und die Funktionen von SNMPv2 `GetResponse`, `Trap` von SNMP in SNMPv2 ungeändert gesendet. Nur die Funktion `GetBulkRequest` von SNMPv2 muß in eine Folge von `GetNextRequest`-Nachrichten konvertiert werden.

2. Möglichkeit: Bei dieser Möglichkeit "spricht" ein Manager zu beiden Protokollen (biliguage). Wenn der "zweisprachige" Manager eine Nachricht von SNMPv2- Manager oder -Agent bekommt, analysiert er diese Nachricht und schickt er seine eigene Nachricht zum SNMP-agent und wartet auf die von SNMP Antwort. Und zuletzt schickt der 'zweisprachige'Manager seine eigene Antwort entsprechend zu SNMPv2–Manager oder SNMPv2–Agent zurück.

11.3 Zusammenfassung

SNMPv2 ist eine zweite Generation von SNMP. Die SNMPv2 SMI stellt gründlichere Spezifikationen und Dokumentationen für verwaltete Objekten auf. Die SNMPv2 SMI erweitert einige Typdefinitionen: `BIT STRING, Counter64 , UInteger,`...

Ebenso ist SNMPv2 um zwei wichtige zusätzliche Funktionen `GetBulkRequest` und `InfoRequest` erweitert. Außerdem stellt SNMPv2 die bessere Spezifikation und Dokumentationsmöglichkeit für verwaltete Objekte zur Verfügung.

SNMPv2 kann die Übertragung größer Datenmengen vereinfachen und die Zugriffkontrolle verbessern.

11.4 Literaturverzeichnis

[Sta93j]

Kapitel 12

SNMPv2 Security

Jörg Fischer

12.1 Einleitung

Das Protokoll SNMP (Simple Network Management Protocol) und das darauf basierende SNMP-Management haben sich in den letzten Jahren zu einem Defacto-Standard für das Management von lokalen Netzen entwickelt.[1] Als besondere Vorzüge des SNMP-Management sind seine Einfachheit und seine Implementierungsfreudigkeit zu nennen. Allerdings zeigte sich im praktischen Einsatz recht bald, daß das SNMP-Management neben seinen vielen Vorteilen an verschiedenen Stellen extreme Lücken aufweist. Als Beispiel ist in diesem Zusammenhang die mangelnden Sicherheitsmaßnahmen in der Interaktion beispielsweise zwischen Manager und Agent zu nennen.

Dieses Kapitel[2] zeigt auf, wie diese Mängel des SNMP in SNMPv2 behoben werden. Die daraus resultierenden Sicherheitsmaßnahmen wurden als SNMPv2 zusammengefaßt.

Der nächste Abschnitt befaßt sich mit dem Begriff des Sicherheitsmanagements im allgemeinen und den daraus resultierenden Aspekten, die im SNMPv2 Security umgesetzt wurden. Der darauf folgende Abschnitt beschreibt kurz die Entwicklungsgeschichte des SNMPv2. Daraufhin werden Erweiterungen und Verbesserungen des SNMPv2 gegenüber SNMP aufgezeigt. Im Abschnitt 12.5 wird auf die Spezifikation der Sicherheitsaspekte von SNMPv2 eingegangen. Im einzelnen werden hier zwei Dokumente besprochen: Das des administrative model und das Dokument der Sicherheitsprotokolle. Abschließend werden die Verbesserungen der Sicherheitsverfahren von SNMPv2 gegenüber SNMP zusammengefaßt.

12.2 Das Sicherheitsmanagement

Für das Sicherheismanagement als integrierter Bestandteil des Netzwerkmanagements lassen sich folgende, grundlegende Aufgaben und Funktionen nennen[3]:

- Die Verwaltung von sicherheitsrelevanter Information
- Das Melden von sicherheitsbezogenen Ereignissen (events)
- Die Schaffung, Steuerung und Beseitigung von sicherheitsrelevanten Diensten und Mechanismen

Bei der Entwicklung des SNMPv2 wurde in diesem Zusammenhang auf folgende Aspekte des Sicherheitsmanagement besonders Wert gelegt:

- Regelung der Zugriffssteuerung (access control) auf sicherheitsrelevante Dienste
- Schaffung einer Sicherheitsverwaltung (security administration)

Eine Regelung der Zugriffssteuerung bezieht sich besonders auf die Art und Weise, wie mit der Managementinformation, die der Agent in seiner MIB (Management Information Base) hält, umzugehen ist.

Die Sicherheitsverwaltung wurde durch das administrative model in SNMPv2 verwirklicht. In diesem Modell sind die Sicherheitsverfahren des SNMPv2 Security eingebettet.

Als Sicherheitsverfahren sind in SNMPv2 gegenüber SNMP ein verbessertes Authentifizierungsverfahren sowie ein Verfahren zur verschlüsselten Übertragung der Managementinformation neu aufgenommnen worden.

12.3 Entwicklung des SNMPv2

SNMP besteht aus einer minimalen, aber sehr mächtigen Menge von Managementfunktionen, die zur Überwachung und Steuerung von Netzwerkelementen dienen. Unter Netzwerkelemente versteht man dabei Devices wie zum Beispiel Hostrechner, Terminalserver sowie Gateways etc. Das SNMP ist zuständig für die Kommunikation zwischen den Netzwerkmanagementstationen und den agents, die sich innerhalb der Netzwerkelemente befinden.

Da SNMP die Anzahl und Komplexität der Managementfunktionen minimieren soll, sind in SNMP eine Reihe von Mängel aufgetreten, die durch die Entwicklung einer zweiten Version von SNMP behoben werden sollten. Der Entwicklungsstrang ist in Abbildung 12.1 dargestellt.

[1] Vgl. [Spr92] und [Abe93]
[2] Es stützt sich im wesentlichen auf [Stall]
[3] Vgl. [HS92]

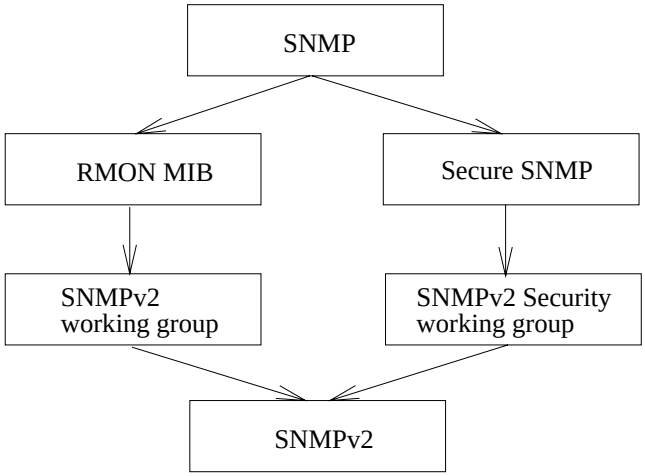


Abbildung 12.1: Entwicklungsgeschichte des SNMPv2

Als unzulänglich erwies sich in SNMP die Netzwerküberwachung. Diese Schwierigkeiten wurden dadurch behoben, daß eine Menge von standardisierten Managementobjekten definiert wurden, die dann in die remote monitoring MIB (RMON MIB) eingegangen sind.

Als weit aus gravierender erwies sich, daß Sicherheitsverfahren in SNMP völlig fehlten. Um diesem Problem abzuhelfen, wurde ein Satz von Dokumenten definiert, der dann als Secure SNMP[4] herausgegeben wurde.

Schließlich wurden zwei Arbeitsgruppen ins Leben gerufen, die sich zum einen mit nicht sicherheitsrelevanten Aspekten und zum anderen mit Sicherheitsaspekten auseinandersetzten. Die Ergebnisse beider Arbeitsgeruppen wurden als SNMPv2 herausgegeben.

12.4 Verbesserungen durch SNMPv2

Die ungenügenden Sicherheitsverfahren in SNMP waren für viele Kritiker der größte Mangel dieses Internet-Managementkonzeptes. Die strikt verfolgte Devise, das Management möglichst einfach zu halten, stellte sich im nachhinein als schwerwiegender Fehler dar. Besonders die nicht vorhandene Verschlüsselungsmöglichkeit führte dazu, daß man keinen schreibenden Zugriff auf den Agenten-MIB zuließ.

Alle Verbesserungen des SNMPv2 gegenüber SNMP sind in Abbildung 12.2 dargestellt.

Wie oben schon erwähnt, sind in SNMPv2 als neue Sicherheitsverfahren die Möglichkeit der Verschlüsselung einer Nachricht sowie ein verbessertes Authentifizierungsverfahren neu hinzugekommen.

12.5 Sicherheitsaspekte von SNMPv2

Die Spezifikation der SNMPv2 Security ist in drei Dokumente festgelegt worden:

[4] Kurz: S-SNMP

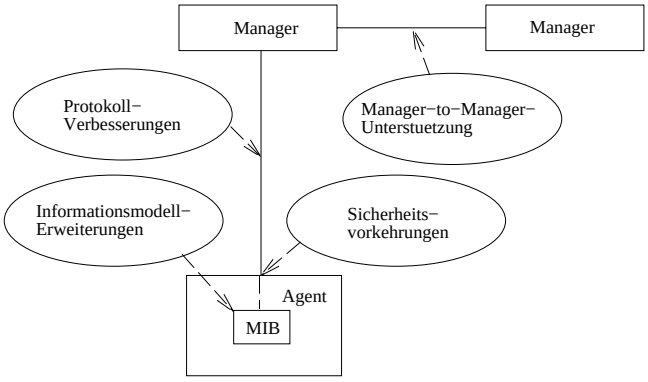


Abbildung 12.2: Erweiterungen und Verbesserungen durch SNMPv2

1. SNMPv2 administrative model
2. SNMPv2 Sicherheitsprotokolle
3. SNMPv2 Party MIB

Die ersten beiden Dokumente werden im fortfolgenden besprochen. Bei dem Dokument der Party MIB sei auf die Literatur verwiesen.[5]

12.5.1 Das administrative model

In diesem Modell agiert jede am Management beteiligte Instanz als eine sogenannte SNMPv2 Party. Eine SNMPv2 Party legt dabei neben anderem die in seinem domain zu benutzenden Sicherheitsmechanismen fest. Die Beschreibung erfolgt in Form einer Party MIB. **Die Definition einer SNMPv2 Party** läßt sich wie folgt angeben:

```
SnmpParty ::= SEQUENCE {
    partyIdentity           OBJECT IDENTIFIER,
    partyTDomain            OBJECT IDENTIFIER,
    partyTAddr              OCTET STRING,
    partyMaxMessageSize     INTEGER,
    partyAuthProtocol       OBJECT IDENTIFIER,
    partyAuthClock          INTEGER,
    partyAuthPrivate        OCTET STRING,
    partyAuthPublic         OCTET STRING,
    partyAuthLifetime       INTEGER,
    partyPrivProtocol       OBJECT IDENTIFIER,
    partyPrivPrivate        OCTET STRING,
    partyPrivPublic         OCTET STRING }


AclEntry ::= SEQUENCE {
    aclTarget               OBJECT IDENTIFIER,
    aclSubject              OBJECT IDENTIFIER,
    aclResources            OBJECT IDENTIFIER,
    aclPrivileges           INTEGER }
```

Das Partykonzept, das in S-SNMP entwickelt wurde, wird im wesentlichen übernommen. In SNMPv2 fehlen allerdings drei Variablen in der Definition einer einer Party im Vergleich zu S-SNMP.

[5] Vgl. [Stall]

Die Variablen `PartyAuthLastMsg` und `PartyNonce` fehlen, da es in SNMPv2 keinen ordered delivery mechanism mehr gibt. Der ordered delivery mechanism ist ein Verfahren, das sicherstellt, daß Datenpakete von einer empfangenden Party nur in aufsteigender Reihenfolge angenommen werden.

Die Variable `PartyProxyFor` fehlt, weil in SNMPv2 das Konzept des context neu eingeführt wurde.

SNMPv2 context

Unter einem SNMPv2 context versteht man eine Sammlung von zu verwaltenden Resourcen, die von einer SNMPv2 Instanz aus zugänglich sind. Der SNMPv2 context gibt an, wo diese Resourcen gehalten werden.

Es werden hierbei zwei verschiedene context-Typen unterschieden: Der **remote SNMPv2 context** und der **local SNMPv2 context**. Im folgenden werden beide context-Typen erläutert.

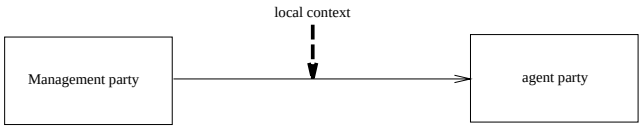


Abbildung 12.3: Beispiel einer Interaktion eines local context

Ein local SNMPv2 context repräsentiert eine MIB view. Eine MIB view stellt dabei eine Untermenge aller möglichen Objekte der agent MIB dar. Im Falle eines local context benutzt eine SNMPv2 Instanz nur lokale Mechanismen, um auf die Managementinformation zugreifen zu können, die durch den SNMPv2 context repräsentiert wird.

Ein remote SNMPv2 context repräsentiert eine Stellvertreterbeziehung. In diesem Fall handelt eine SNMPv2 Instanz als ein Stellvertreter-Agent, um auf die Managementinformation zugreifen zu können, die von diesem SNMPv2 context identifiziert werden.

Der context ist also ein Konzept, das sowohl mit der Zugriffssteuerung als auch mit den MIB views zu tun hat.

Dies soll nun anhand eines Beispiels erläutert werden: Eine management station interagiert mit einem agent eines Netzwerkelementes, um auf die Managementinformation beim agent zugreifen zu können.

Für den Fall eines local context ist diese Interaktion in Abbildung 12.3 dargestellt. Verbunden mit dem local context ist eine MIB view beim Agenten. Der context referenziert diese MIB view, die angibt, welche Objekte nun zugänglich sind. Der Agent entscheidet mit einer **access control policy**, welche Operationen erlaubt sind. Die Managementparty ist in diesem Fall die Quellparty, die agent party ist die Zielparty.

Für den Fall eines remote context ist dies Interaktion in Abbildung 12.4 dargestellt. Die Managementparty interagiert hier mit einem Stellvertreter-Agenten, der die Instanz, die die eigentliche, gesuchte Managementinformation enthält, vertritt.

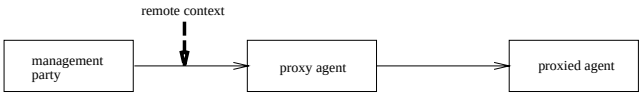


Abbildung 12.4: Beispiel einer Interaktion eines remote context

access control policy

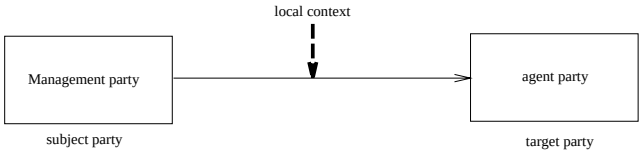


Abbildung 12.5: Darstellung einer access control policy

Eine access control policy besteht aus vier Elementen:

1. Das **target** führt angeforderte Managementoperationen aus.
2. Das **subject** fordert diese Managementoperationen an.
3. Die **Resourcen** stellen den gewählten context dar und sind die Managementinformation, auf die die angeforderten Managementoperationen ausgeführt werden.
4. **Die Privilegien** repräsentieren die erlaubten Operationen, die das target im Auftrag des subject ausführen darf. Sie gehören zu einem bestimmten context.

Abbildung 12.5 stellt diese subject/target-Beziehung anhand des vorigen Beispiels dar. Die access control policy wird also durch die drei Parameter subject, target und context bestimmt. Für ein subject/target-Paar lassen sich also verschiedene access control policies unterscheiden, indem der context von Fall zu Fall variiert.

Der Vorteil des Konzeptes des context gegenüber dem S-SNMP-Konzept

Wie schon erwähnt, ist beim S-SNMP der Platz für die Managementinformation in der Variablen `PartyProxyFor` der Partytabelle.[6] Dies hat zur Folge, daß diese Tabelle unnötig groß und komplex wird. Ferner muß bei den Stellvertreterbeziehungen für gleiche subject/target-Paare eigene proxy parties, also separate Einträge in die Party-Tabelle, geschaffen werden, da die variierende Managementinformation in der Partytabelle festgehalten werden muß.

Die Vorteile des SNMPv2-Konzeptes liegen auf der Hand:

1. Einerseits wird zwischen einer direkten (local context) und indirekten (remote context) Beziehung unterschieden, aber andererseits wird aus der Sicht

[6] Vgl. S. 71

der manager party davon abstrahiert.

Das bedeutet, daß es dem Manager nicht bekannt ist, ob der Agent, von dem er Operationen auf der gesuchten Managementinformation ausführen lassen will, diese Information selber hält, oder ob der Agent eine Instanz, die diese Information hält, vertritt. Ist letzteres der Fall, so ist dem Manager die Identität des proxy agent nicht bekannt.

2. Da die Managementinformation nicht in der Partytabelle, sondern für jeden context als Eintrag in der context-Tabelle gehalten wird, ergibt sich aus den genannten Gründen eine erhebliche Speicherplatzersparnis.

Context-Typen

Zusammenfassend werden die verschiedenen context-Typen unterschieden. Wie schon erwähnt, unterscheidet man im wesentlichen zwei Hauptgruppen: Der local context und der remote context.

Beim letzteren gibt es wiederum zwei Arten von Stellvertreterbeziehungen:

- **Foreign proxy relationship**
- **Native proxy relationship**

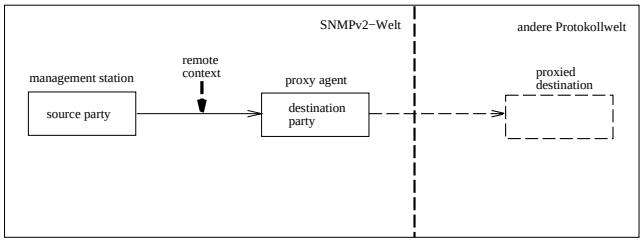


Abbildung 12.6: Foreign proxy relationship

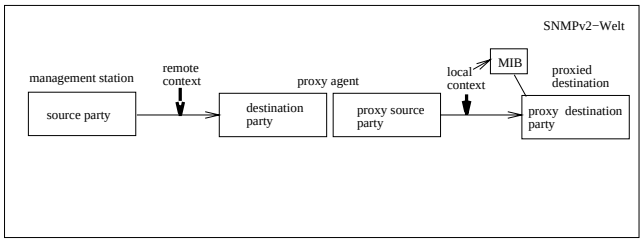


Abbildung 12.7: Native proxy relationship

Die Abbildungen 12.6 und 12.7 stellen nun beide Stellvertreterbeziehungen dar. Als Beispiel wurde hier eine Interaktion von einer manager party zu einer agent party gewählt. Wie zu sehen ist, besteht der Hauptunterschied zwischen beiden Beziehungen darin, daß die Managementinformation im vom proxy agent vertretenen Ziel einmal außerhalb der SNMPv2-Protokollwelt (foreign proxy relationship) das andere Mal innerhalb der SNMPv2-Welt (native proxy relationship) gehalten wird. Beiden gemeinsam ist natürlich, daß der remote context keine MIB view im proxy agent repräsentiert. Der proxy agent vermittelt vielmehr zwischen manager station und dem Ziel, das die Information hält.

Beide Beziehungen benötigen ferner Zugriffssteuerungsinformation. Denn die access control policy zeigt an, welche Operationen zwischen manager station und dem proxy agent ausgeführt werden können.

Bei der foreign proxy relationship ist zu beachten, daß hier die Protokollgrenzen verlassen werden.[7] Das bedeutet, daß der proxy agent in der Lage sein muß, auf die Managementinformation im Ziel ohne SNMPv2-verwandte Mittel zuzugreifen.

Bei der native proxy relationship greift der proxy agent mittels eines local context auf die Managementinformation bei der Zielparty zu. Das bedeutet, daß der proxy agent zwei Rollen bei dieser Interaktion annimmt: Zum einen bei derInteraktion mit der Zielparty die Rolle des Managers, zum anderen bei der Interaktion mit der manager station die Rolle eines Agenten. Diese Rollenteilung drückt sich dadurch aus, daß beim proxy agent zwei verschiedene Parties laufen.

Zusammenfassend läßt sich also sagen, daß mit Hilfe der foreign proxy relationship eine Interaktion mit protokollfremden Zielen ermöglicht wird. Bei der native proxy relationship wird hingegen über eine indirekte Beziehung auf eine MIB view zugegriffen. Mit Hilfe dieser Beziehung lassen sich verteilte Managementstrukturen ermöglichen.

12.5.2 SNMPv2 Sicherheitsprotokolle

Ein weiteres Dokument der Spezifikation der SNMPv2 Security, das hier besprochen werden soll, ist das der Sicherheitsprotokolle.

Die SNMPv2 Sicherheitsprotokolle stimmen im wesentlichen mit denen überein, die in S-SNMP definiert sind. Die einzigen, bedeutsamen Unterschiede zwischen SNMP und S-SNMP sind folgende:

- Kein ordered delivery mechanism
- Vereinfachte clock-Synchronisation
- Context-Parameter in Nachrichten, um u. a. clock-Synchronisation zu ermöglichen

Im folgenden werden alle Punkte erläutert.

ordered delivery mechanism

Bei diesem Verfahren werden Datenpakete von der empfangenden Party nur in aufsteigender Reihenfolge angenommen. Hierzu ist im header einer Nachricht ein sogenannter `nonce value`, eine monoton wachsende Funktion vorhanden, mit der die Nachrichten mit einem sogenannten Zeitstempel versehen werden. Beim Empfang der Pakete werden nur solche angenommen, die in Reihenfolge ankommen. Diese Reihenfolge ergibt sich aus der Kombination von `nonce value` und Zeitstempel.

[7] Beispielsweise in die "alte" Protokollwelt des SNMP

Dieses Verfahren hat den Nachteil, daß bei hoher Last des Netzes die Reihenfolge der Pakete nicht gewährleistet werden kann. Daher wird in SNMPv2 versucht, auf dieses Konzept zu verzichten, zumal sich aus diesem Verfahren auch kein zusätzlicher Sicherheitsgewinn ergibt. Auf die `nonce value` wird in SNMPv2 folglich verzichtet.

Der clock-Synchronisationsalgorithmus

Nun wird darauf eingegangen, wie die Uhrensynchronisation in SNMPv2 funktioniert. Jede Nachricht enthält in ihrem header Authentifizierungsinformation. Diese Authentifizierung beruht unter anderem auf der Uhrensynchronistion zwischen beispielsweise Manager und Agent.

Die Authentifizierungsinformation im header einer Nachricht ist wie folgt definiert:

```
AuthInformation := IMPLICIT SEQUENCE {
    authDigest              OCTET STRING
    authDstTimestamp        UInteger32
    authSrcTimestamp        Uinteger32 }
```

Auf die Funktion der Variablen `authDigest` wird später eingegangen. Jede Nachricht enthält ferner einen Zeitstempel sowohl von der sendenden als auch von der empfangenden Party. Ferner ist in der Partytabelle für eine Nachricht zwischen zwei Parties eine Lebenszeit vereinbart.

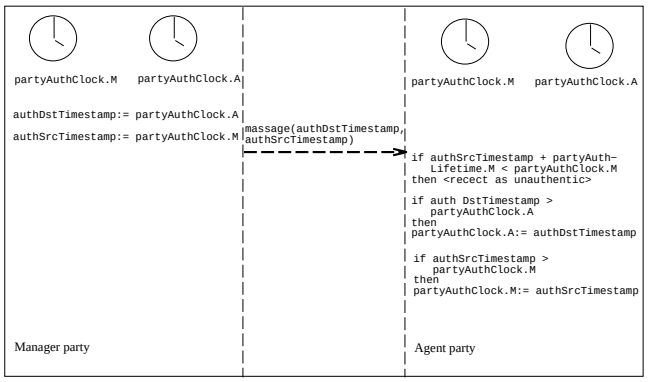


Abbildung 12.8: Uhrensynchronisation und Authentifizierung anhand der Zeitstempel

Wie Abbildung 12.8 am Beispiel einer Interaktion zwischen Manager und Agent zeigt, laufen in jeder der beteiligten Parties zwei Uhren sowohl der sendenden, als auch der empfangenden Party. Die sendende Party liest die Uhren und weist sie den Zeitstempeln zu. Mit Hilfe der Bedingung `authSrcTimestamp + partyLifetime.a < partyAuthclock.a` wird geprüft, ob die Nachricht angenommen oder zurückgewiesen wird. Problematisch wird dieses Authentifizierungsverfahren, wenn die beteiligten Uhren[8] nicht synchron zueinander sind. Hierzu gibt es vier Bedingungen, die eine Uhrenkorrektur erforderlich machen:

1. Die Manager Version der Agenten-Uhr ist größer als die Agenten Version der Agenten-Uhr. Bei dieser Bedingung besteht das Risiko einer falschen Zurückweisung einer Nachricht vom Agent zum Manager.

2. Die Manager Version der Manager-Uhr ist größer als die Agenten Version der Manager-Uhr. Bei dieser Bedingung besteht das Risiko einer falschen Annahme einer Nachricht vom Manager zum Agenten.

3. Die Agent Version der Agent-Uhr ist größer als die Manager Version der Agent-Uhr. Bei dieser Bedingung besteht das Risiko einer falschen Annahme einer Nachricht vom Agent zum Manager.

4. Die Agent Version der Manager-Uhr ist größer als die Manager Version der Manager-Uhr. Bei dieser Bedingung besteht das Risiko einer falschen Zurückweisung einer Nachricht vom Manager zum Agenten.

Wie man sieht, sind die ersten beiden Bedingungen zu den letzten beiden symmetrisch.

In Abbildung 12.8 ist nun dargestellt, wie sich diese Bedingungen korregieren lassen. Durch die Bedingung `authDstTimestamp > PartyAuthclock.Agent` wird Bedingung 1 korregiert. Durch `authSrcTimestamp > PartyAuthclock.Manager` wird obige Bedingung 2 korregiert. Die Bedingungen 3 und 4 lassen sich korregieren, wenn man eine Nachricht vom Agenten zum Manager schickt.

Auf diese Weise läßt sich die Uhrensynchronität wiederherstellen. Dennoch gibt es noch ein Problem. Normalerweise läuft eine Interaktion zwischen Manager und Agent in der Weise ab, daß der Manager eine Anfrage an den Agenten richtet und der Agent dann antwortet. Die Uhrensynchronität ist durch ein solches Nachrichtenpaar[9] wiederhergestellt. Sollte aber zwischen zwei solchen Anfragen eine erhebliche Zeitspanne liegen, so kann aufgrund der mangelnden Synchronität der beteiligten Uhren Bedingung 4 automatisch erfüllt sein. Das bedeutet, daß jede authentifizierte Nachricht vom Manager zum Agenten zurückgewiesen werden würde.

Um dieses Dilemma zu vermeiden, muß der Manager nach folgendem Verfahren die manager clock bei der manager party und allen beteiligten agent parties periodisch synchronisieren:

1. Der Manager sichert seinen Wert von der manager clock

2. Der Manager benutzt ein nichtauthentifiziertes `Get`, um den Wert der manager clock beim Agent zu holen.[10]

3. Wenn der Agentwert den Managerwert von der manager clock übersteigt, wird der Agentwert in die manager clock beim Manager übernommen.

[8] Zum Beispiel die Manager Version der Agenten-Uhr und die Agenten Version der Agenten-Uhr

[9] Dies ist also eine Anfrage

[10] Die Nachricht darf nicht authentifiziert sein, weil sie nach Bedingung vier zurückgewiesen werden könnte

4. Überprüfung der Synchronisierung mit authentifiziertem `Get`

context-Information

Im Gegensatz zum S-SNMP muß der header einer Nachricht beim SNMPv2 context-Parameter beinhalten.

Damit laßt sich nun die vollständige Nachrichtenstruktur von SNMPv2 angeben. Abbildung 12.9 gibt dabei zwei der möglichen Formate einer SNMPv2-Nachricht an.

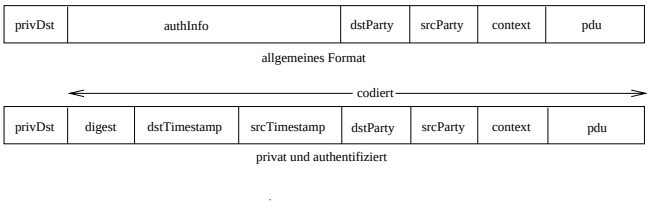


Abbildung 12.9: Zwei Formate einer SNMPv2-Nachricht

Hiermit kann man das Verfahren zur Erzeugung einer Nachricht angeben:

1. Zunächst wird der Variablen `srcParty` der Bezeichner der sendenden Party, der Variablen `dstParty` die Identität der empfangenden Party zugewiesen. Ferner wird in `context` der verlangte SNMPv2 `context` geschrieben. `Pdu` repräsentiert die verlangten Managementoperationen.

2. Den Zeitstempeln werden nach dem oben beschriebenen Verfahren die Stände der clocks zugewiesen. Der Variablen `authDigest` wird dann der private Authentifizierungsschlüssel der sendenden Party zugewiesen. Dieser Schlüssel ist bei der Partyinformation sowohl der sendenden als auch der empfangenden Party abgespeichert.

   Über die gesamte Nachricht wird dann mit Hilfe eines Checksummenalgorithmus[11] ein 16-Bit-langer Wert berechnet. Dieser Wert wird dann der Variablen `digest` zugewiesen und stellt den "Fingerabdruck" dieser Nachricht dar, weil der Algorithmus verhindert, daß eine zweite Nachricht mit gleicher Summe erzeugt wird. Der in die Berechnung mit eingeflossene, aber nicht mit der Nachricht verschickte Authentifizierungsschlüssel verhindert, daß eine beliebige, vom Empfänger als gültig erkannte Nachricht durch Zufall durch einen Dritten erkannt wird. Der Empfänger kann anhand des Schlüssel auch erkennen, ob die Nachricht vom richtigen Sender übertragen wurde.

3. Die Nachricht wird verschlüsselt.

4. Der Zielparty-Bezeichner wird in `privDst` gesetzt.

Eine Nachricht nach dem SNMPv2-Format kann nun nach dem folgenden Verfahren empfangen werden:

[11] Vgl. [KK93]

1. Wenn der `privDst`-Schlüssel nicht gültig ist, wird die Nachricht abgewiesen.

2. Falls die Authentifizierungsinformation fehlerhaft ist oder falls `dstParty` nicht zu `privDst` paßt oder `srcParty` unbekannt ist, wird die Nachricht abgewiesen.

3. Nun erfolgt die Authentifizierung anhand der Zeitstempel:

   $$\texttt{partyauthClock.a} - \texttt{authSrcTimestamp} \leq \texttt{partyAuthLifetime.a}$$

   Wenn dieses Kriterium nicht erfüllt ist, wird die Nachricht verworfen.

4. Hier erfolgt die Authentifizierungsprüfung anhand des `authDigest`-Wertes.

   Der Authentifizierungsschlüssel der sendenden Party ist ebenfalls in der Partyinformation der empfangenden Party abgespeichert. Anhand dieses Schlüssel wird der Checksummenalgorithmus noch einmal über die Nachricht ausgeführt. Der dann erhaltene Wert wird mit dem Wert der Nachricht in der Variablen `authDigest` verglichen. Nur wenn beide Werte gleich sind, wird die Nachricht akzeptiert.

5. Falls der context unbekannt ist, wird die Nachricht abgewiesen.

6. Anhand der Zeitstempel werden die clocks in der empfangenden Party aktualisiert.

7. Die weitere Bearbeitung hängt von dem gewählten context ab.

12.6 Verbesserungen durch SNMPv2

In SNMP hängt die gesamte Authentifizierung von einem einzigen string, dem sogenannten **community string** ab. Dieser string repräsentiert folgende sicherheitsrelvante Information:

- Die Identität der anfordernden Instanz (management station)
- Die Identität der ausführenden Instanz (agent)
- Die Identität der Speicherstelle der Managementinformation, auf die zugegriffen werden soll.
- Die Authentifizierungsinformation
- Die Zugriffssteuerungsinformation
- Die MIB view-Information

In SNMP hängt somit die gesamte Authentifizierung davon ab, ob der community string eines SNMPv2-Paketes mit einem dem Agenten bekannten übereinstimmt oder nicht. Derjenige, der das Netzwerk beobachtet und die

Struktur eines SNMP-Paketes kennt, ist damit nach dem ersten beobachteten Paket in der Lage, den community string zu identifizieren. Damit hat er alles, was er braucht, um eigene Netzwerkmanagementnachrichten zu erzeugen. Als Folge davon haben viele Netzwerkbetreiber den schreibenden Zugriff auf die agent MIB völlig verboten.

Die gleiche sicherheitsrelevante Information wird in SNMPv2 verteilt repräsentiert:

- Anfordernde InstanzFeld im header einer Nachricht
- Ausführende InstanzFeld im header einer Nachricht
- Speicherstelle der Managementinformation in context-Tabelle
- usw.

Abschließend muß festgestellt werden, daß die Verwendung eines privaten Authentifizierungsschlüssels, der nur dem Sender und Empfänger bekannt ist, sowie die Möglichkeit zur Verschlüsselung einer Nachricht bei SNMPv2 einen erheblichen Sicherheitsgewinn bedeuten. Der verwandte Checksummenalgorithmus stellt wiederum in SNMPv2 nicht sicher, daß die Datenpakete in korrekter Reihenfolge beim Empfänger ankommen. Da jedoch eine Reihenfolgevertauschung eine Folge der normalen Netzwerkübertragung von UDP-Paketen ist, stellt eine erzwungene Reihenfolge jedoch auch keinen zusätzlichen Sicherheitsgewinn dar.

12.7 Literaturverzeichnis

[Sta93k] [HS92] [KK93] [Abe93] [Spr92]

Literaturverzeichnis

[Abe93] Sebastian Abeck. Snmpv2. PIK, Ausgabe 16, 1993.

[Com88] Douglas Comer. *Internetworking with TCP/IP: principles, protocols and architecture.* Prentice-Hall, 1988. ISBN 0-13-470154-2.

[HS92] T. Henrich and G. Schreiner. Netzwerkmanagement: Theoretische Basis — Blick in die Realität. Interner Bericht Nr. 29/92, Universität Karlsruhe (TH), 1992.

[Ker92] H. Kerner. Rechnernetze nach OSI. Addison-Wesley Publishing Company, Inc., 1992.

[KK93] Michael Kienle and Michael Kuschke. Neue version für netzwerkstandard. iX-Magazin, Ausgabe 7, July 1993.

[LF89a] A. Leinwand and K. Fang. ... In *Network Management: A Practical Perspective*, number ..., pages 1–17, ..., 1989.

[LF89b] A. Leinwand and K. Fang. ... In *Network Management: A Practical Perspective*, number ..., pages 131–140, ..., 1989.

[LKK93] P. Lockemann, G. Krüger, and H. Krumm. Telekommunikation und Datenhaltung. Carl Hanser Verlag, 1993.

[MR91] K. McCloghrie and M. T. Rose. *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II.* Network Working Group, March 1991. Request For Comments 1213, ersetzt RFC 1158.

[RM91] M. T. Rose and K. McCloghrie. *Concise MIB Definitions.* Network Working Group, March 1991. Request For Comments 1212.

[Ros91] M. T. Rose. *A Convention for Defining Traps for use with the SNMP.* Network Working Group, March 1991. Request For Comments 1215.

[Sch91] G. Schreiner. Hierarchisches SNMP-basiertes Netzwerkmanagement; Entwicklung von Managementprotokoll und Applikationen. Diplomarbeit an der Universität Karlsruhe (TH); Informatikrechnerabteilung, 1991.

[Spr92] Fabian Sprengel. Unterschiede zwischen snmp und osi-cmip. UNIX-Magazin, Ausgabe 8, August 1992.

[Sta93a] William Stallings. CMIS and CMIP. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 428–472, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93b] William Stallings. Network Monitoring. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 21–49, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93c] William Stallings. The Open Systems Interconnection (OSI) Reference Model. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 541–558, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93d] William Stallings. OSI Systems-Management Concepts. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 371–428, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93e] William Stallings. Remote Network Monitoring. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 173–218, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93f] William Stallings. Secure SNMP. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 219–271, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93g] William Stallings. (Simple Network Management Protocol) SNMP. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 133–172, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93h] William Stallings. SNMP Management Information Base. In *SNMP, SNMPv2 and*

CMIP: The Practical Guide to the Network-Management Standards, number ISBN 0-201-63331-0, pages 78–132, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93i] William Stallings. SNMP Network-management Concepts. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 65–77, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93j] William Stallings. SNMP Version 2 (SNMPv2): Management Information and Protocol. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 272–341, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93k] William Stallings. SNMPv2 Security. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 342–368, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93l] William Stallings. Systems-Management Functions. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 473–540, Addison-Wesley Publishing Company, Inc., 1993.

[Sta93m] William Stallings. The TCP/IP Protocol Suite. In *SNMP, SNMPv2 and CMIP: The Practical Guide to the Network-Management Standards*, number ISBN 0-201-63331-0, pages 559–567, Addison-Wesley Publishing Company, Inc., 1993.