Netzwerkmanagement mit SNMP

→ Teil 3: SNMPv2

Prof. Dr. Norbert Pohlmann

Fachbereich Informatik

Verteilte Systeme und Informationssicherheit

Fachhochschule Gelsenkirchen

Inhalt

- **Verwendung unterschiedlicher Transportdienste**
- **Erweiterungen der SMI (Structure of Management Information)**
- **Erweiterung der MIB-II (Management Information Base)**
- **Erweiterung des SNMP Protokolls**

Weiterentwicklung von SNMP: SNMPv2

Weiterentwicklung von SNMP: SNMPv2

- Vier Jahre nach der Einführung von SNMP wurde das Nachfolgeprotokoll SNMPv2 (Version 2) von der *Internet Engineering Task Force (IETF)* aus der Taufe gehoben.
- SNMPv2 entstand als Antwort auf die Mängelliste zu SNMP, die sich aus dem umfangreichen praktischen Einsatz von SNMP ergeben hatte.
- Die wichtigsten Neuerungen von SNMPv2 beziehen sich auf folgende Themen:
  - Verwendung von weiteren Transportprotokollen
  - Erweiterung der *Structure of Management Information (SMI)*
  - Erweiterung des eigentlichen SNMP-Protokolls
  - Erweiterung der MIB-II
  - Komplette Neugestaltung der Zugriffskontrolle und Sicherheit

Verwendung unterschiedlicher Transportdienste → (1/2)

- SNMPv1 wurde als Netzmanagement-Protokoll für das TCP/IP basierende Netz entwickelt.
- Die IP-Adressierungsart ist in SNMPv1 im Protokoll fest verankert: Der einzige mögliche Objekttyp für Adressen ist IpAddress, eine 32-Bit Internet Adresse.
- Dieser Objekttyp ist jedoch für die Darstellung von OSI-Adressen ungeeignet.
- Eine Verwaltung von Netzen, die auf anderen Protokollen basiert, wie z.B. OSI- oder IPX-Netze, ist damit nicht möglich.

Verwendung unterschiedlicher Transportdienste → (2/2)

- Um eine weitere Verbreitung von SNMP zu ermöglichen und um heterogene Netze komplett über SNMP verwalten zu können, wurden zusätzlich zu der UDP/IP weitere Protokollstacks definiert, über die SNMP Pakete transportiert werden können.
- Zu diesem Zweck wurde ein neuer Adresstyp **NsapAdress** eingeführt, der aus bis zu 21 Oktetten besteht und zur der Aufnahme einer beliebigen OSI-Adresse gedacht ist.
- Hiermit wurde auch gleichzeitig die Möglichkeit geschaffen, 128-Bit IP-Adressen darzustellen.
- Neben dem üblichen (und auch empfohlenen) *User Datagram Protocol* (UDP) werden folgende Transportprotokolle vorgeschlagen.
  - OSI Connectionless-Mode Network Service CLNS
  - OSI Connection-Oriented Network Service COTS
  - Novell Internetwork Packet Exchange (IPX)

Erweiterungen der SMI
→ Definition von Managed Objects (1/3)

- Die Erweiterung der *Structure of Management Information* in SNMPv2 bezieht sich im wesentlichen auf eine überarbeitete Version des OBJECT-TYPE Makros sowie auf die Bearbeitung von Tabellen.

- **Definition von Managed Objects (SNMPv2)**

```
OBJECT-TYPE MACRO          ::=
    BEGIN
        TYPE NOTATION::=
            „SYNTAX“          Syntax
            „UNITS“           Text  |  empty
            „MAX-ACCESS“      Access
            „STATUS“          Status
            „DESCRIPTION“     Text  |  empty
            „REFERENCE“       Text  |  empty
            „INDEX“           “{“  IndexTypes “}“  |  empty
            „DEFVAL“          “{“  value “}“  |  empty
            VALUE NOTATION  ::=  value (VALUE ObjectName)
    END
```

Erweiterungen der SMI
→ Definition von Managed Objects (2/3)

- **UNITS Feld**
  Neu hinzugekommen ist die Möglichkeit im UNITS Feld eine Maßeinheit für ein Objekt zu beschreiben.
  Diese Erweiterung ist besonders hilfreich für Objekte, die in irgendeiner Art messbare Werte beinhalten, z.B. Zeit oder Temperatur.
  Die Maßeinheit wird dabei als einfacher Text eingegeben.

- **MAX-ACCESS**
  Das Schlüsselwort zur Beschreibung der Zugriffsrechte eines Objektes wurde um den Präfix „MAX“ erweitert, um damit den maximalen Zugriff zu verdeutlichen und kann die folgenden Werte annehmen.

  - not-accessible: Auf das Objekt kann nicht zugegriffen werden
  - accessible-for-notify: Das Objekt kann in einer Notifikation enthalten sein
  - read-only: Auf das Objekt kann nur lesend zugegriffen werden.
  - read-write: Auf das Objekt kann lesend und schreibend zugegriffen werden
  - read-create: Auf das Objekt kann lesend und schreibend zugegriffen und das Objekt kann (innerhalb einer Tabelle) vom Manager erzeugt werden

Erweiterungen der SMI
→ Definition von Managed Objects (3/3)

- **STATUS**
  Beim Status Feld wurden die möglichen Werte neu festgelegt:
  - current: Objekt ist Teil des aktuellen Standards
  - deprecated: Objekt ist veraltet, kann aber aus Kompatibilitätsgründen implementiert sein.
  - obsolete: Objekt ist veraltet und sollte nicht mehr implementiert werden

Neue Datentypen in SNMPv2

- In SNMPv2 wurden zwei neue Datentypen für das Management mit SNMP definiert:
  - **Counter64 ::=**
    [APPLICATION 6]
    IMPLICIT INTEGER (0 ... 18446744073709551615)
  - **Unsigned32 ::=**
    [APPLICATION 2]
    IMPLICIT INTEGER (0 ... 4294967295)
- Der Datentyp Unsigned ist dabei nur ein Aliasname für den Gauge Datentyp und ist von diesem nicht zu unterscheiden.
- Des weiteren wurden die Bezeichnungen der Datentypen Gauge und Counter in Gauge32 bzw. Counter32 geändert.

Tabellen in SNMPv2

- Die Definitionen von Tabellen wurden komplett von SNMPv1 übernommen.
- SNMPv2 erweitert die Konventionen in RFC1212, um das Erzeugen, Löschen und den Zugriff auf Zeilen einer Tabelle zu ermöglichen.
- Im wesentlichen können zwei unterschiedliche Arten von Tabellen in SNMPv2 definiert werden:
  - Tabellen, die es dem **Manager nicht erlauben,** Tabellenzeilen zu erzeugen oder zu löschen.
    Diese Tabellen werden ausschließlich vom Agenten kontrolliert. Die Objekte dieser Tabellen haben entweder read-write oder read-only Zugriffsrechte.
  - Tabellen, die es dem **Manager erlauben,** neue Zeilen zu erzeugen, bzw. Zeilen zu löschen.
    Objekte innerhalb dieser Tabellen besitzen entweder die Zugriffsrechte read-create oder read-only.
    Es ist ebenfalls möglich, dass sowohl der Manager als auch der Agent selbst die Anzahl der Tabellenzeilen variieren kann.

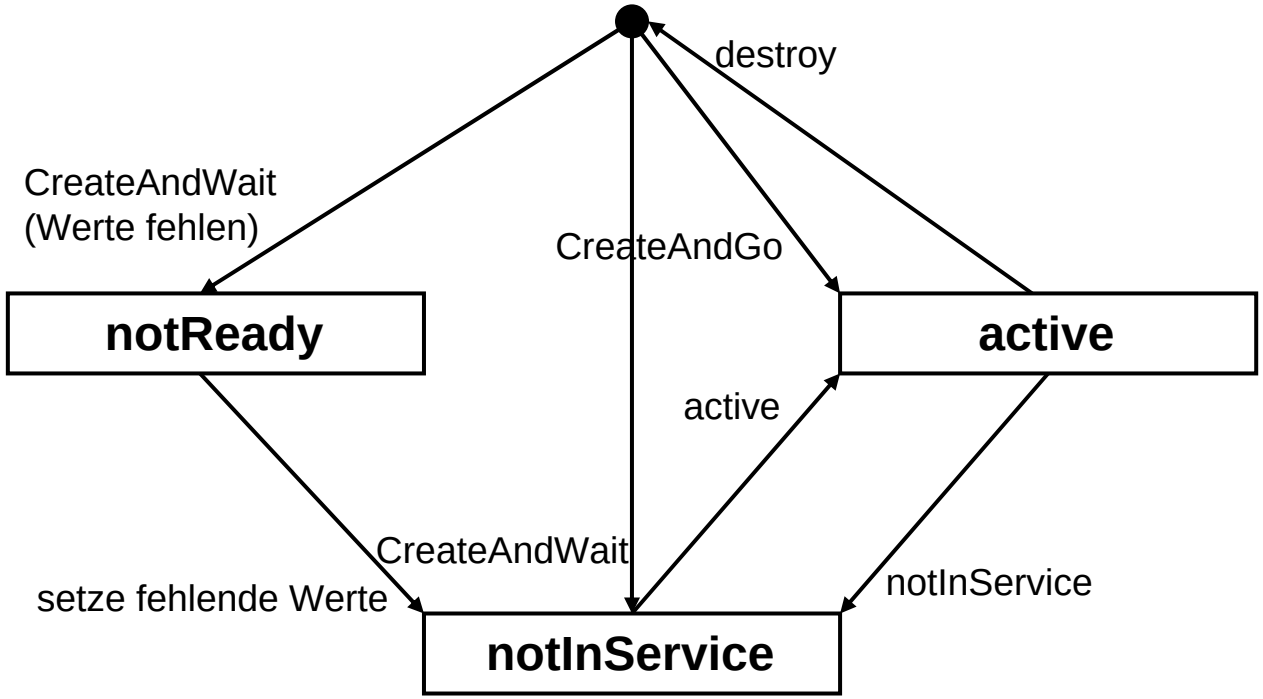
Tabellen in SNMPv2
→ Indizieren von Tabellen

- SNMPv2 schreibt vor, dass Tabellenzeilen eindeutig über einen Index identifiziert werden können.
- Der Index einer Tabelle wird in dem INDEX Teil des OBJECT-TYPE Makros festgelegt und kann auch aus mehreren Spaltenobjekten zusammengesetzt sein.
- Neu in SNMPv2 ist, dass per Konvention alle Indexobjekte das Zugriffrecht not-accessible besitzen.

Tabellen in SNMPv2
→ Erzeugen von Tabellenzeilen

- Eines der am heißesten diskutierten Punkte in SNMPv2 war, wie Tabellenzeilen hinzugefügt oder gelöscht werden können.
- SNMPv1 lässt diesen völlig offen, was dazu führte, dass für jede MIB ein unterschiedlicher Ansatz gewählt wurde.
- In SNMPv2 wurde ein Ansatz standardisiert, der sich im praktischen Einsatz der RMON MIB bewährte.
- Demzufolge besitzt **jede Tabelle**, die es dem Manager erlaubt, Zeilen hinzuzufügen oder zu löschen, per Konvention ein **Statusspaltenobjekt**.
- Dieses Statusobjekt besitzt den Datentyp RowStatus und die Zugriffsrechte read-create.
- Tabellenzeilen werden neu erzeugt oder gelöscht, indem der Wert diesen Statusobjektes über eine set Operation verändert wird.
- SNMPv2 sieht zwei Möglichkeiten vor, wie Zeilen erzeugt werden können: CreateAndWait und CreateAndGo.

Zustände einer Tabellenzeile

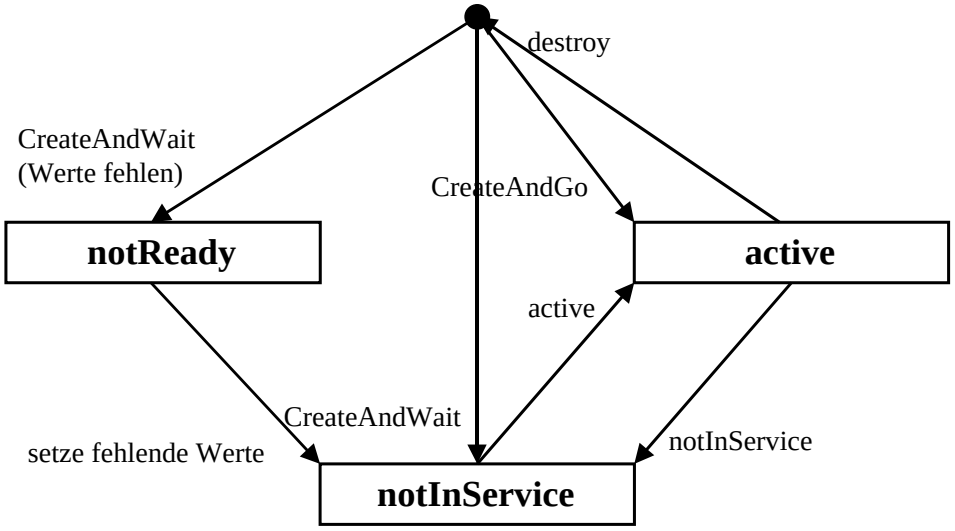


Erzeugen von Tabellenzeilen
→ CreateAndWait (1/2)

- Bei der CreateAndWait Methode erzeugt der Manager eine neue Tabellenzeile, indem er das **Statusobjekt** mit einem neuen Index auf den Wert „CreateAndWait“ setzt.
- Es liegt in der Verantwortung des Managers einen neuen eindeutigen Index für die erzeugende Tabellenzeile zu wählen.
- Der Agent erzeugt daraufhin die Zeile mit dem neuen Index und setzt alle Objekte der neuen Zeile mit read-create Zugriffsrechten auf ihre Defaultwerte.
- Anschließend setzt der Agent das **Statusobjekt** auf den Wert NotInService.
- Gibt es keine Defaultwerte für ein oder mehrere Objekte der Tabelle, setzt der Agent das **Statusobjekt** auf den Wert NotReady und zeigt damit dem Manager an, dass dieser noch Werte für einzelne Objekte der neuen Zeile setzen muss.

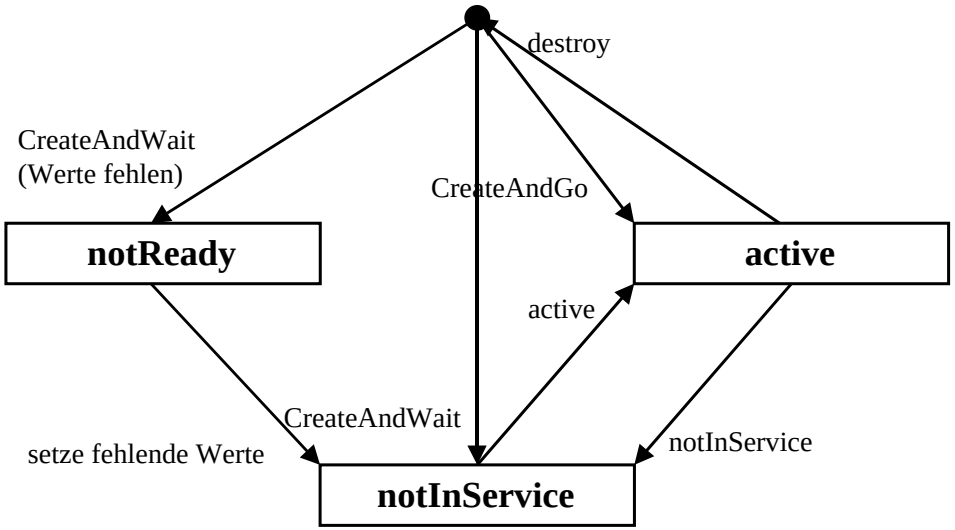
Erzeugen von Tabellenzeilen
→ CreateAndWait (2/2)

- In beiden Fällen kann der Manager über eine get Operation Werte und Zustände der neuen Spaltenobjekte überprüfen.
- Bei Spaltenobjekte, die keine Defaultwerte besitzen, liefert der Agent den Wert NoSuchInstance zurück.
- Bei Objekten, die der Agent überhaupt nicht unterstützt, wird der Wert NoSuchObject an den Manager gesendet.
- Mit Hilfe dieser Information kann der Manager nun die noch fehlenden Werte in einen abschließenden set Befehl setzen und den Statuts auf active umschalten.



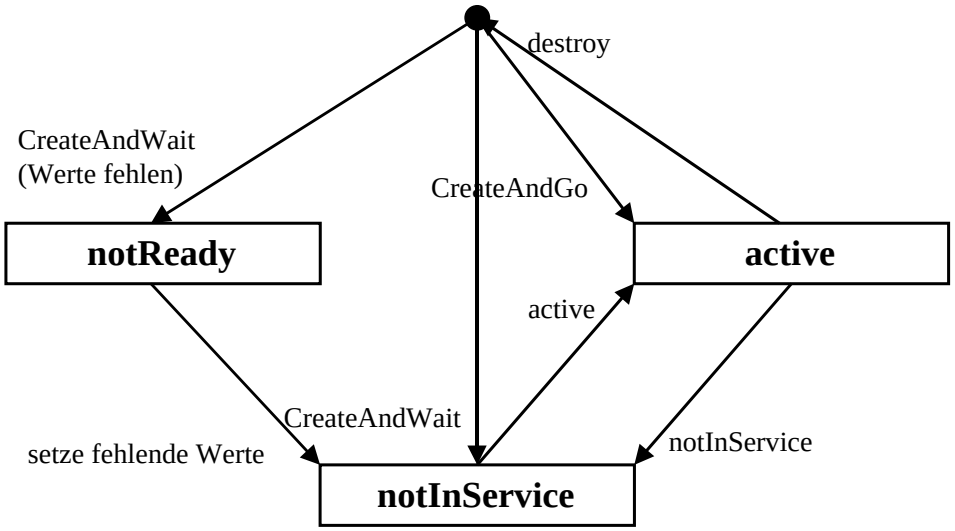
Erzeugen von Tabellenzeilen
→ CreateAndGo

- Im Gegensatz dazu erlaubt es die CreateAndGo Methode, eine Tabellenzeile in einem einzigen set Befehl zu erzeugen und zu aktivieren.
- Dazu setzt der Manager alle read-create Spaltenobjkete auf ihre entsprechenden Werte und das **Statusobjekt** auf den Wert CreateAndGo.
- Der Agent erzeugt daraufhin eine neue Tabellenzeile, initialisiert die Spaltenobjekte entsprechend und setzt das Statusobjekt auf aktiv.



Tabellen in SNMPv2
→ Löschen von Tabellenzeilen

- Um eine Tabellenzeile zu löschen, setzt der Manager den Wert des **Statusobjektes** auf destroy.
- Der Agent entfernt daraufhin die entsprechende Zeile.
- Der Manager hat auch die Möglichkeit eine Tabellenzeile vorübergehend „auszuschalten“, indem er das **Statusobjekt** auf den Wert NotInService setzt.



Definition von Traps

- Für die Beschreibung von SNMPv2 Traps wurde ein neues Makro NOTIFICATION-TYPE definiert:

```
NOTIFICATION-TYPE MACRO::=
    BEGIN
        TYPE NOTATION::=
            „OBJECTS“        { Objects }   |   empty
            „STATUS“         Status
            „DESCRIPTION“    Text   |   empty
            „REFERENCE“      Text   |   empty
            VALUE NOTATION  ::=  value (VALUE ObjectName)
    END
```

Beispiel: linkup Traps

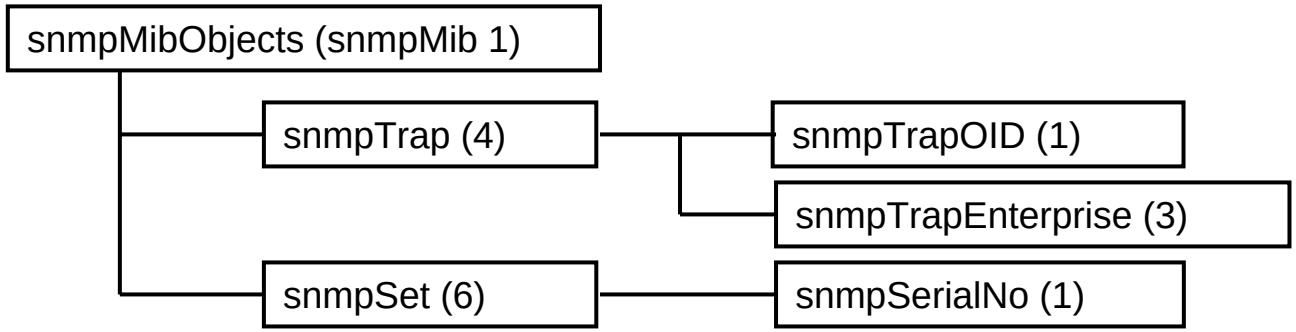
- linkup NOTIFICATION-TYPE
  OBJECTS { ifIndex, ifAdminStatus, ifOperStatus }
  STATUS current
  DESCRIPTION „A linkUp trap signifies that the SNMPv2 entity, acting in an agent role, has detected that the ifOperStatus object for one of its communication links has transitioned out of the down state.“
  ::= { snmpTraps 4 }
- Dieser Trap beschreibt somit, dass ein linkUp Trap immer dann von einem Agenten gesendet wird, falls eine Kommunikationsverbindung (wieder) hergestellt werden konnte.
- Das OBJECTS Feld besagt dabei, dass der Agent die Objekte ifIndex, ifAdminStatus und ifOperStatus (und zwar genau in dieser Reihenfolge) in dem VarBindList Feld des Traps mitsendet.
- Für jeden SNMPv2 Trap wird zusätzlich zu den optionalen *Managed* immer der *Object Identifier* des Traps (hier snmpTraps 4) sowie die aktuelle Zeit (sysUpTime) des Traps mitgesendet.

Erweiterung der MIB-II

- Die Erweiterung in der SNMPv2 Management Information Base bezieht sich im wesentlichen auf die folgenden Gruppen:
  - system Gruppe — enthält Erweiterungen zu der MIB-II system Gruppe
  - SNMP Gruppe — enthält Objekte für das SNMP Protokoll selbst
  - interfaces Gruppe — enthält Objekte zur Beschreibung der Schnittstellen
  - MIB objects Gruppe — enthält Objekte für Traps PDUs sowie für die Koordination von set Operationen
- Die Änderungen in den ersten drei Gruppen resultieren aus dem praktischen Einsatz von SNMP.
- Im wesentlichen wurden neue nützliche Objekte definiert oder wie bei der SNMP Gruppe viele nicht sehr sinnvolle Objekte gestrichen.

MIB Objekts Gruppe (1/3)

- Die Objects Gruppe enthält zwei Untergruppen, wie in den folgenden Abbildungen dargestellt:



Die erste Untergruppe enthält zwei Objekte snmpTrapOID und snmpTrapEnterprise, die beim Senden von Traps oder InformRequests eine Rolle spielen:

snmpTrapOID	repräsentiert den *Objekt Identifier* des Traps oder der inform-Operation. Diese Varibable erscheint immer an zweiter Stelle in jeder SNMPv2 Trap und inform PDU.
snmpTrapEnterprise	repräsentiert den „enterprise“ *Object Identifier* für den jeweiligen Trap.

MIB Objekts Gruppe (2/3)

- Im zweiten Teil der MIB Objects Gruppe wurde eine zweite Untergruppe snmpSet definiert, die z.Z. nur ein einziges Objekt snmpSerialNo vom Typ INTEGER enthält.
- Dieses Objekt wurde eingeführt, um Inkonsistenzen durch konkurrierende Managementzugriffe zu vermeiden.
- Dieses Objekt ist vom Typ TestAndIncr, für den die folgende Konvention definiert wurde.

  Der aktuelle Wert von snmpSerial sei **K**, dann gilt:

  - Falls der Agent eine set Operation mit dem Wert **K** für dieses Objekt empfängt, so inkrementiert der Agent snmpSerialNo und die set Operation liefert den Wert **K** zurück.
  - Falls der Agent eine set Operation mit einem Wert ungleich **K** empfängt, so wird der Fehler inconsistentValue zurückgeliefert.

MIB Objekts Gruppe (3/3)

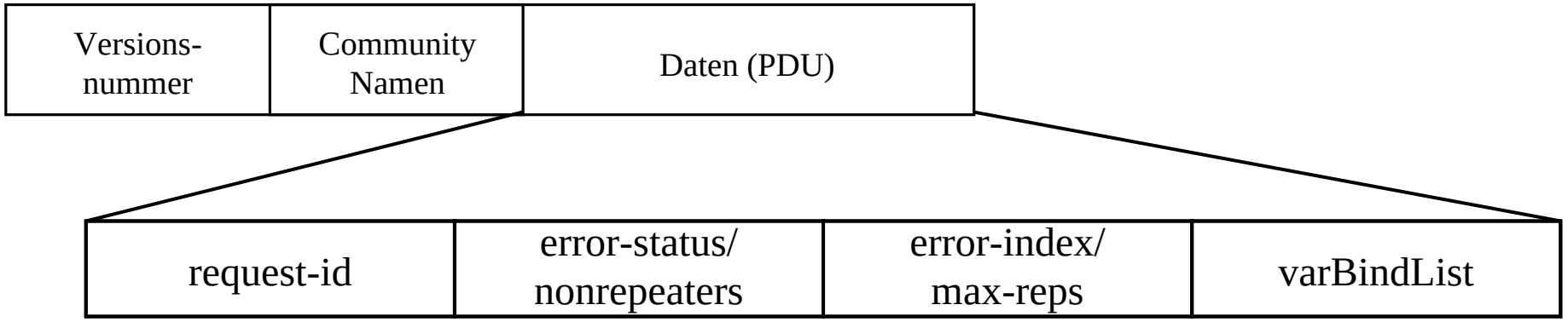
- Möchte nun ein Manager A eine set Operation ausführen, so holt er sich zunächst den aktuellen Wert von snmpSerialNo und führt dann die gewünschte set Operation aus, wobei er zusätzlich versucht, das snmpSerialNo Objekt auf den zuvor abgefragten Wert zu setzen.
- Sollte in der Zwischenzeit ein anderer Manager B bereits eine entsprechende set Operation erfolgreich ausgeführt haben, so schlägt die set Operation des Managers fehl, da der Wert von snmpSerialNo durch die erste set Operation bereits inkrementiert wurde.
- Auf diese Weise kann garantiert werden, dass zu jeder Zeit immer nur ein Manager eine set Operation ausführen kann.
- Dies setzt natürlich voraus, dass jeder Manager sich an diese Konvention hält!
- Es ist ebenfalls denkbar und auch sinnvoll, dass ein entsprechendes Objekt für jede Gruppe einer MIB definiert wird, um parallelen Zugriff von mehreren Managern auf diese Gruppen zu erlauben.

Erweiterung des SNMP Protokolls (1/2)

- Die Erweiterung des SNMP Protokolls gegenüber der Version SNMPv1 beziehen sich hauptsächlich auf die Einführung neuer Operationen sowie einzelne Verbesserungen auf die Einführung neuer Operationen sowie einzelne Verbesserungen im Bereich Effizient und Fehlerbehandlung.
- Wie bei SNMPv1 teilt sich eine SNMPv2 Nachricht in drei Anteile: Versionsnummer, *Community*-Name und Datenteil (PDU).
- Die Versionsnummer einer SNMPv2 Nachricht besitzt den **Wert 1** (0 für SNMPv1).
- In SNMPv2 wurden neben den bereits bestehenden get, get-next und set Operationen zwei zusätzliche Operationen definiert:
  - **get-bulk** für das effiziente Auslesen von Tabellen
  - **inform** für die Kommunikation zwischen Managern

Erweiterung des SNMP Protokolls (2/2)

- Die Struktur der SNMPv1-Trap-PDU wurde an die PDUs der übrigen Operationen angepasst.
- Da auch die beiden hinzugekommenen Operationen denselben PDU Aufbau besitzen, gibt es für SNMPv2 nunmehr nur noch eine PDU, in der alle Operationen verpackt werden können.



Der get Operator (1/2)

- Die PDU der get Operation ist identisch mit der SNMPv1 PDU.
- Der einzige Unterschied besteht in der Art und Weise wie Ergebnisse zurückgegeben werden.
- Eine SNMPv1 get Operation ist atomar, d.h. entweder werden alle gewünschten *Managed Objects* zurückgegeben oder - falls ein Fehler auftreten sollte - überhaupt keine.
- Dies führt in der Praxis häufig dazu, dass SNMP get Operationen mehrfach gesendet werden müssen, falls ein Agent ein Managed Object nicht unterstützt.

Der get Operator (2/2)

- InSNMPv2 kann ein Agent das Wertefeld eines *Managed Object* auf zwei **Ausnahmebedingungen** (*exception condition*) setzen, falls der Agent den Wert des Objektes nicht ermitteln kann.
  - noSuchObject — Das Objekt existiert nicht in der MIB des Agent
  - noSuchInstance — Die Instanz des Objektes existiert nicht. Diese Fehlermeldung wird immer dann von Agenten zurückgegeben, wenn ein Manager versucht auf ein Objekt in einer Tabelle mit einem falschen Index zuzugreifen.
  - Damit ist es für den Agenten möglich, eine get Operation nur teilweise zu beantworten.
  - Auf der anderen Seite erkennt der Manager sofort, wieso der Agent nicht die gewünschten Informationen liefern kann.

Der get-next Operator

- Die get-next PDU ist identisch zu der entsprechenden SNMPv1 PDU.
- Die Semantik einer SNMPv2 get-next Operation wurde auch hier um eine Ausnahmebedingung endOfMibView erweitert.
- Diese Ausnahmebedingung wird dann vom Agenten gesetzt, falls eine get-next Operation auf die am Ende stehende Variable einer MIB ausgeführt werden soll.

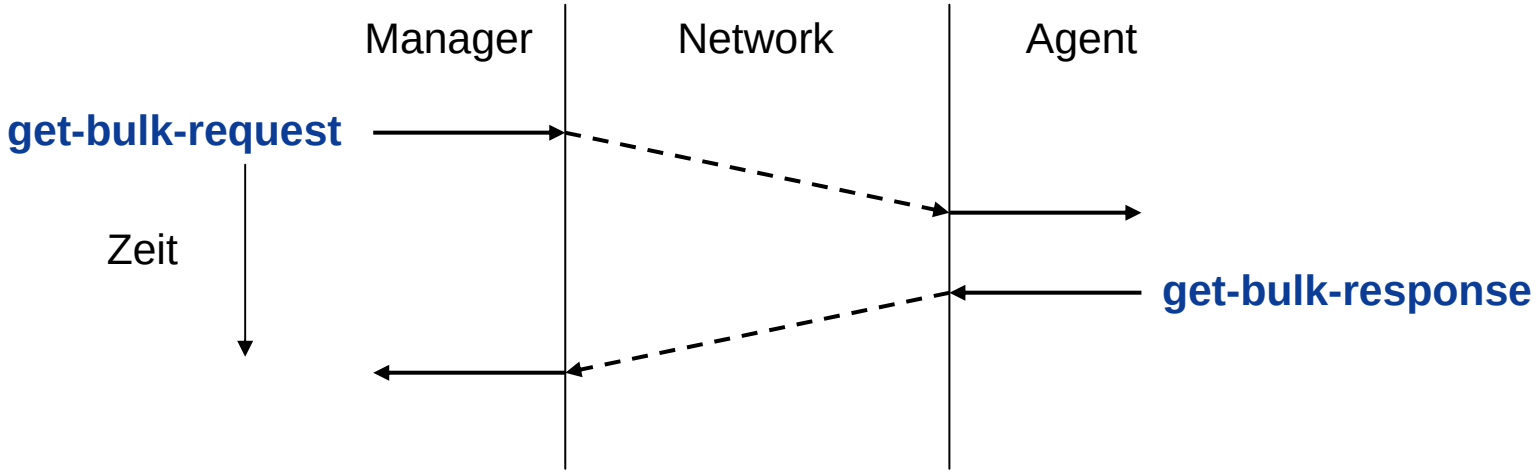
Der set Operator

- Die SNMPv2 set PDU ist ebenfalls identisch zu der PDU der Vorgängerversion.
- Allerdings wurden die Fehlercodes für die set Operation erheblich verfeinert.
- Gegenüber 5 möglichen Fehlercodes in SNMPv1 definiert SNMPv2 18 verschiedene Fehlerarten, die es dem Manager ermöglichen, die genaue Ursache für das mögliche Scheitern einer set Operation herauszufinden.

Der get-bulk Operator (1/3)

- Ein großes Problem bei SNMPv1 ist der relativ hohe Managementverkehr, der das Netz zusätzlich belastet.
- Gerade für den Bereich des Fehlermanagements arbeitet SNMP deswegen sehr ineffizient, die Relation zwischen der Anzahl der abgefragten Daten und den dadurch erkannten Problemsituationen fällt sehr schlecht aus.
- Um die Anzahl der SNMP-Pakete zu vermindern, können mit dem neuen get-bulk-Kommando in einem einzigen Paket ganze Tabellen übertragen werden.
- SNMPv1 Systeme wiederholen, wenn sie mehrere Variablen eines Agenten abfragen wollen, für jeden Parameter jeweils das get-next-Kommando.
- Um beispielsweise die Routing-Tabelle eines Routers (eine unter Umständen sehr umfangreiche Tabelle) abzufragen, muss eine SNMP-Managementstation für jeden Eintrag ein get-next-Kommando absetzen.

Der get-bulk Operator (2/3)

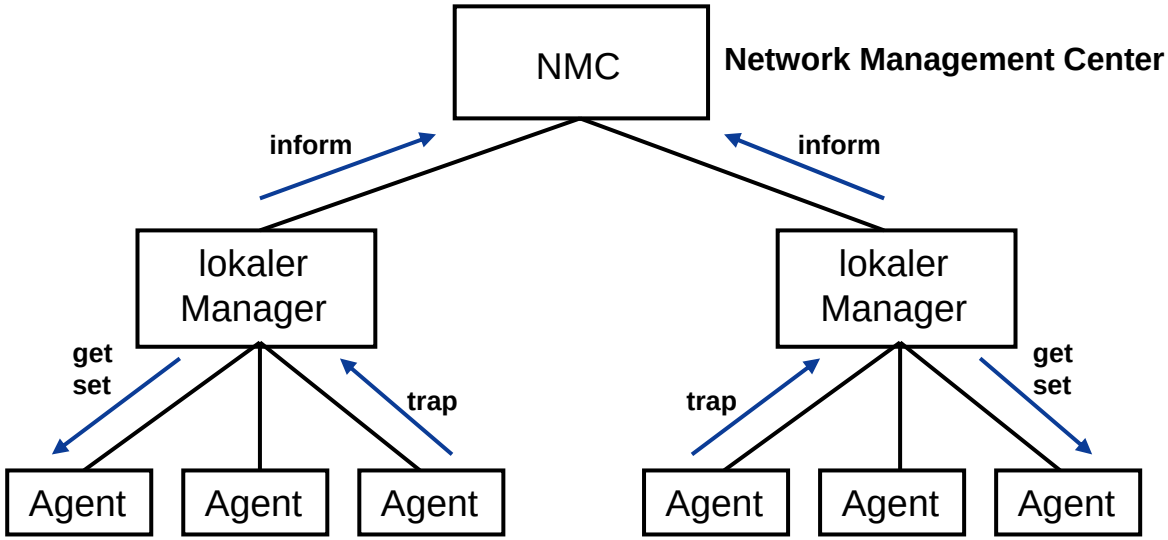


- Die get-bulk Operation verwendet dasselbe Auswahlprinzip wie die get-next Operation, d.h. der Agent liefert immer den in der MIB-Hierarchie folgenden Wert zurück. In der get-bulk PDU können mehrere Nachfolgervariablen zurückgegeben werden.
- Die get-bulk PDU definiert dazu zwei Felder non-repeaters und max-repetitions, die in keiner anderen PDU auftauchen.
- Das erste Feld enthält die Anzahl der MIB-Variablen im varBindList Feld.
- Für die restlichen Variablen im varBindList Feld zeigt max-repetition an, wie viele Nachfolgervariablen zurückgegeben werden sollen.

Der get-bulk Operator (3/3)
→ Beispiel

- get-bulk (non-repeaters=1, max-repetitions=5, sysUpTime, ifDescr, ifType, ifSpeed)
- Der Agent kann aus den folgenden Gründen die Bearbeitung einer get-bulk Operation abbrechen:
  - die Größe der PDU überschreitet ein (lokales) Maximum
  - das Ende der MIB wurde erreicht (in diesem Fall enthalten die Variablem im VarBindList Feld den Wert endOfMibView)
  - die Verarbeitung der gesamten Operation ist für den Agenten zu zeitaufwendig
- In jedem Fall liefert der Agent aber immer so viele Informationen zurück wie er kann und bricht nicht einfach die Operation komplett ab.
- Damit kann der Manager die noch fehlenden Informationen in einer zweiten get-bulk Operation abfragen.

Der inform Operator (1/3)

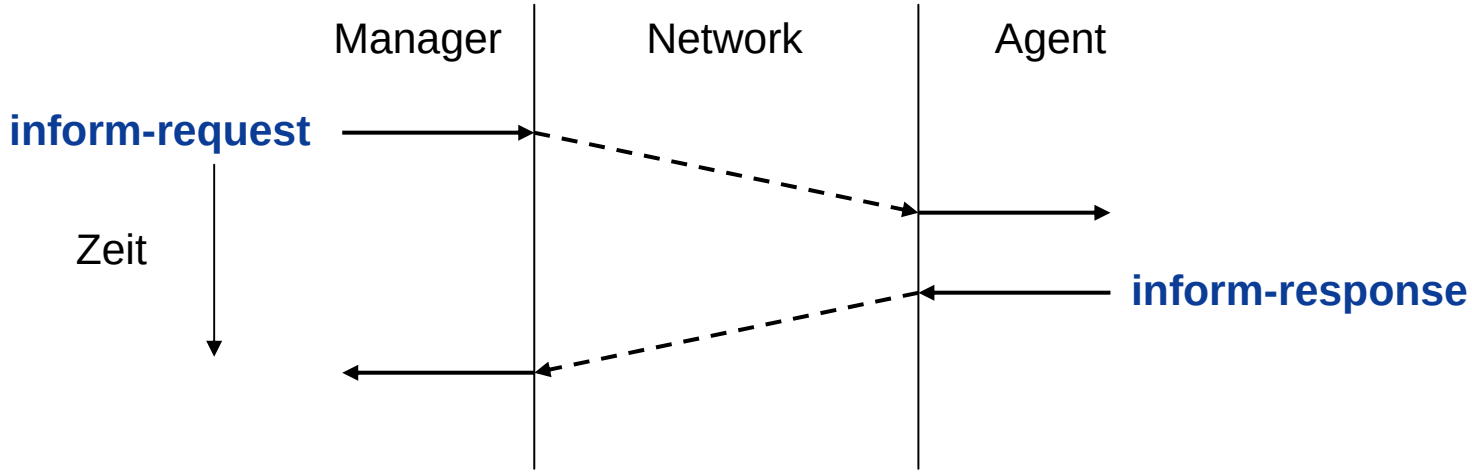


- Eine der wichtigsten Neuerungen von SNMPv2 ist die Manager-zu-Manager Kommunikation, diese bedeutet, dass eine Station als Manager oder/und als Agent agieren kann.
- Damit können hierarchische Management-Strukturen aufgebaut werden, in denen untergeordnete Manager Aufgaben für eine zentrale Managementstation übernehmen können.

Der inform Operator (2/3)
→ Zweistufiges Konzept

- Die **lokalen Manager** überwachen die lokalen Komponenten der lokalen Netze und informieren die darüberliegenden Ebenen nur bei wichtigen Ereignissen.
- Das **Netzwork Management Center (NMC)** überwacht primär die Verbindungen zwischen den lokalen Netzen und hat die globale Sicht auf diese.
- Weil die lokalen Manager mit diesem Konzept die zentrale Station (NMC) entlasten, lassen sich weit größere SNMP-Installationen als bisher realisieren.

Der inform Operator (3/3)



- Die inform PDU ist genauso aufgebaut wie auch die get/get-next oder set-PDU.
- Die ersten beiden Variablen innerhalb des VarBindList Feldes sind immer eine Zeitmarke (sysUpTime) und ein Ereignis-Identifier.
- Ein Unterschied zu Traps besteht darin, dass inform Operationen vom Empfänger bestätigt werden müssen.
- Obwohl ursprünglich für die Manager-zu-Manager Kommunikation gedacht, wird der inform-Operator häufig auch zwischen Agent und Manager als eine Art bestätigter Trap eingesetzt.

Der SNMPv2 trap Operator

- Die SNMPv2 trap Operation erfüllt denselben Zweck wie die trap Operation von SNMPv1, allerdings mit einem unterschiedlichen PDU Format.
- SNMPv2 Traps verwenden dasselbe PDU Format wie alle anderen Operationen auch, was letztendlich die Implementierung von SNMP vereinfacht.
- Das varBindList Feld einer SNMPv2 trap Operation enthält neben möglichen spezifischen Variablen immer die folgenden Objekt Werte-Paare:
  - sysUpTime.0 — Aktuelle Zeit beim Auftreten des Traps
  - snmpTrapOID.0 — Diese MIB-Variable ist Teil der erweiterten SNMPv2 MIB und enthält als Wert den *Object Identifier* des jeweiligen Traps.
- SNMPv2 Traps werden mit Hilfe des NOTIFICATION-TYPE Makros definiert.

SNMPv2 Versionen

- Die größten Veränderungen fanden im Bereich der Sicherheit statt, da dort die größten Defizite bestanden.
- In SNMPv1 werden die Kommunikationsbeziehungen zwischen Managementstationen und Agenten durch deren Zugehörigkeit zu Gruppen (Communities) definiert.
- Dieses Modell wurde in SNMPv2 durch die Einführung der **„SNMPv2-Party“** und des **„SNMPv2-context“** wesentlich erweitert.
- Beide Konzepte konnten sich jedoch in der Internet-Gemeinde aufgrund ihrer Komplexität nicht durchsetzten und sind nur noch von historischem Interesse.
- Die fehlende Akzeptanz von Seiten der Anwender führte letztendlich dazu, dass die SNMPv2 Sicherheitskonzeption nach einer Überarbeitung des SNMPv2 Standards (RFC1901-RFC1908) komplett fallengelassen und das Community Modell von SNMPv1 übernommen wurde.

SNMPv2 Versionen
→ SNMPv2c

- Diese als **Community based** SNMPv2 bezeichnete Version stellte den kleinsten gemeinsamen Nenner dar und hat den Vorteil, dass sie schnell und einfach in die Praxis umzusetzen ist.
- Jedoch ist die Verwendung des unsicheren, nicht mehr zeitgemäßen Sicherheitsmodells von SNMPv1 ein gravierender Nachteil.
- Aus diesem Grund gingen die Arbeiten weiter, die zu zwei konkurrierenden Ansätzen führten.

SNMPv2 Versionen
→ SNMPv2u

- SNMPv2u ähnelt in vielen Bereichen der ursprünglichen Spezifikation von SNMPv2.
- Anstelle der parties tritt hier ein **User/Agent-Konzept**, daher der Name User-Based-Security Model (oder kurz USEC-Modell).
- Der Anwender authentisiert sich mittels Name und zugehörigem Passwort.
- Zu jedem Anwender kann ein Schlüssel generiert werden, der dann zur Verschlüsselung der übermittelten Daten benutzt wird.

SNMPv2 Versionen
→ SNMPv2*

- Der zweite Ansatz zur Lösung des Sicherheitsproblems wird als SNMPv2* (v2 star) bezeichnet, das zu etwa 80% auf den Dokumenten zu SNMPv2u basiert.
- Zusätzlich zu den bei SNMPv2u beschriebenen Punkten beschäftigt sich SNMPv2* noch mit Funktionen, die auf der Managementstation implementiert werden und mit der Möglichkeit der Fernkonfiguration eines Agenten, um z.B. die MIB eines Agenten von der Managementstation aus zu verändern.

Netzwerkmanagement mit SNMP

→ Teil 3:SNMPv2

Vielen Dank für Ihre Aufmerksamkeit

Fragen ?

norbert.pohlmann@informatik.fh-gelsenkirchen.de

Fachhochschule Gelsenkirchen