INTERNET MANAGEMENT PROTOCOLS

STATE OF THE ART / RECENT DEVELOPMENTS

TUTORIAL T6 - PRESENTED AT IM’2003
COLORADO SPRINGS, USA
24 MARCH 2003

AIKO PRAS
UNIVERSITY OF TWENTE
THE NETHERLANDS

pras@cs.utwente.nl
http://wwwwhome.cs.utwente.nl/~pras

http://www.simpleweb.org/tutorials
OVERVIEW

BACKGROUND
• HISTORY, GOALS & STANDARDS

• STRUCTURE OF MANAGEMENT INFORMATION
  • SCALARS
  • TABLES

• MANAGEMENT INFORMATION BASES
  • OVERVIEW
    • MIB-II
    • SNMPv2, IF & IP MIB

• SIMPLE NETWORK MANAGEMENT PROTOCOL
  • VERSION 1
  • VERSION 2
  • VERSION 3

AGENTX
DISMAN

RECENT DEVELOPMENTS

FURTHER INFORMATION
SNMP HISTORY


CMOT

HEMS/HEMP

proposed
standard

historic

implementation
experience

draft
standard

full
standard

full
standard

SNMP security

SNMP (parties)

SNMPv2 (community)

SNMPv3

draft
standard

proposed
standard

draft
standard

SNMP

SGMP

SNMPv2

SNMPv3

(SNMPv2)

(proposed
standard)

full
standard

full
standard

draft
standard

proposed
standard

draft
standard

draft
standard

(SNMPv2)

full
standard

full
standard
SNMP GOALS

UBIQUITY
- PCs AND CRAYs

INCLUSION OF MANAGEMENT SHOULD BE INEXPENSIVE
- SMALL CODE
- LIMITED FUNCTIONALITY

MANAGEMENT EXTENSIONS SHOULD BE POSSIBLE
- NEW MIBs

MANAGEMENT SHOULD BE ROBUST
- CONNECTIONLESS TRANSPORT
PRINCIPLE OPERATION

MANAGER

SNMP

AGENTs
PRINCIPLE OPERATION

AGENTS

GET / SET

TRAP

MIB

MANAGER

GET / SET

PRINCIPLE OPERATION
PRINCIPLE OPERATION

MANAGER

AGENTS

TABLES

VARIABLES
SNMP STRUCTURE

MANAGER

Management Application

SNMP PDUs

CONNECTIONLESS TRANSPORT SERVICE PROVIDER

UDP

AGENT

MIB
STANDARDS

SMI
- STRUCTURE OF MANAGEMENT INFORMATION
  - TWO VERSIONS
    - RFC 1155, RFC 2578, ...

MIBs
- MANAGEMENT INFORMATION BASES
  - A LARGE NUMBER OF MIBs EXIST
    - RFC 1213, ...

SNMP
- SIMPLE NETWORK MANAGEMENT PROTOCOL
  - NAME IS USED IN A MORE GENERAL SENSE
    - VERSION 1: HISTORIC (RFC 1157)
    - VERSION 3: STANDARD (RFC 3411-3416)
SMI

STRUCTURE OF MANAGEMENT INFORMATION

RFC 1155: SMIv1

RFC 1212: CONCISE MIB DEFINITIONS

RFC 2578: SMIv2

RFC 2579: TEXTUAL CONVENTIONS

MAKES THE DEFINITION OF (NEW) MIBs EASIER
MANAGEMENT INFORMATION WITHIN MANAGED SYSTEMS MUST BE REPRESENTED AS:
• SCALARS
• TABLES
(= TWO DIMENSIONAL ARRAYS OF SCALARS)

THE SNMP PROTOCOL CAN ONLY EXCHANGE (A LIST OF) SCALARS

DEFINED IN TERMS OF ASN.1 CONSTRUCTS
### SMI: DATA TYPES FOR SCALARS

<table>
<thead>
<tr>
<th>SMIv1</th>
<th>SMIv2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SIMPLE TYPES:</strong></td>
<td><strong>APPLICATION-WIDE TYPES:</strong></td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
</tr>
<tr>
<td>OCTET STRING</td>
<td>OCTET STRING</td>
</tr>
<tr>
<td>OBJECT IDENTIFIER</td>
<td>OBJECT IDENTIFIER</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Gauge</td>
<td>Gauge32</td>
</tr>
<tr>
<td>Counter</td>
<td>Counter32</td>
</tr>
<tr>
<td>-</td>
<td>Counter64</td>
</tr>
<tr>
<td>TimeTicks</td>
<td>TimeTicks</td>
</tr>
<tr>
<td>IpAddress</td>
<td>IpAddress</td>
</tr>
<tr>
<td>Opaque</td>
<td>Opaque</td>
</tr>
<tr>
<td>NetworkAddress</td>
<td>-</td>
</tr>
</tbody>
</table>

- **SIMPLE TYPES:**
- **APPLICATION-WIDE TYPES:**
- **PSEUDO TYPES:**
EXAMPLE OF SCALAR OBJECTS

MANAGER AGENT

SNMP

MANAGED OBJECT INSTANCES

name

address

uptime
OBJECT NAMING

INTRODUCE NAMING TREE

NEW-MIB:

1

address (1)
130.89.16.2

info (2)

name (1)
printer-1

uptime (2)
123456

THE LEAVES OF THE TREE REPRESENT THE MANAGED OBJECTS

NODES ARE INTRODUCED FOR NAMING PURPOSES
OBJECT NAMING

• address
  Object ID = 1.1
  Object Instance = 1.1.0
  Value of Instance = 130.89.16.2

• info
  Object ID = 1.2

• name
  Object ID = 1.2.1
  Object Instance = 1.2.1.0
  Value of Instance = printer-1

• uptime
  Object ID = 1.2.2
  Object Instance = 1.2.2.0
  Value of Instance = 123456

ALTERNATIVE:
Object ID = NEW-MIB info uptime
### OBJECT TYPE DEFINITION

**OBJECT-TYPE:**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>OCTET STRING</td>
</tr>
<tr>
<td>OBJECT IDENTIFIER</td>
<td>BITS</td>
</tr>
<tr>
<td>IPAddress</td>
<td>Integer32</td>
</tr>
<tr>
<td>Counter32</td>
<td>Counter64</td>
</tr>
<tr>
<td>Gauge32</td>
<td>TimeTicks</td>
</tr>
<tr>
<td>Opaque</td>
<td>New Type</td>
</tr>
<tr>
<td>read-only</td>
<td>read-write</td>
</tr>
<tr>
<td>read-create</td>
<td>accessible-for-notify</td>
</tr>
<tr>
<td>not-accessible</td>
<td>current</td>
</tr>
<tr>
<td>deprecated</td>
<td>obsolete</td>
</tr>
</tbody>
</table>

"\""
EXAMPLE: ROUTING TABLE

<table>
<thead>
<tr>
<th>destination</th>
<th>next</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

TO RETRIEVE INDIVIDUAL TABLE ENTRIES
EACH ENTRY SHOULD GET AN IDENTIFIER
NAMING OF TABLE ENTRIES - I

POSSIBILITY 1 (NOT BEING USED BY SNMP): USE ROW NUMBERS

NEW-MIB:

1

- address (1)
  - 130.89.16.2

- info (2)

- name (1)
  - printer-1

- uptime (2)
  - 123456

- routeTable (3)

  - dest(1)
    - 2
    - 3
    - 5
    - 7
    - 8
    - 9

  - next(2)
    - 2
    - 3
    - 2
    - 2
    - 3
    - 3

EXAMPLE: THE VALUE OF NEW-MIB routeTable next 5 IS 3

this is row 5
NAMING OF TABLE ENTRIES - II

POSSIBILITY 2 (USED BY SNMP): INTRODUCE AN INDEX COLUMN

NEW-MIB:

1

- address (1)
  - 130.89.16.2

- info (2)
  - name (1)
  - uptime (2)

- routeTable (3)
  - dest(1)
  - next(2)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

EXAMPLE: THE VALUE OF NEW-MIB routeTable next 5 IS 2
**TABLE INDEXING**

**GENERAL SCHEME**

**EXAMPLES:**

- OID of Table = 1.3
- Column number: X.C.I
- Entry does not exist
- Index value:
  - 1.3.2.1 => 1
  - 1.3.2.7 => 2
  - 1.3.2.9 => 3
  - 1.3.1.9 => 9
  - 1.3.1.7 => 2
  - 1.3.1.5 => 5
  - 1.3.1.1 = entry does not exist
  - 1.3.2.1 = entry does not exist
  - OID of Table = 1.3

**GENERAL SCHEME**

**TABLE INDEXING**
TABLE INDEXING - NON-INTEGER INDEX

AN INDEX NEED NOT BE AN INTEGER

routeTable (3)

dest (1) | next (2)
---|---
130.89.16.1 | 130.89.16.1
130.89.16.4 | 130.89.16.4
130.89.16.23 | 130.89.16.1
130.89.19.121 | 130.89.16.1
192.1.23.24 | 130.89.16.4
193.22.11.97 | 130.89.16.4

EXAMPLES:
OID of Table = 1.3
1.3.1.130.89.16.23 => 130.89.16.23
1.3.2.130.89.16.23 => 130.89.16.1
1.3.1.193.22.11.97 => 193.22.11.97
1.3.2.193.22.11.97 => 130.89.16.4
1.3.2.130.89.19.121 => 130.89.16.1
TABLE INDEXING - MULTIPLE INDEX FIELDS

USE OF MULTIPLE INDEX FIELDS

OID of Table

Column number

Index value 1

Index value 2

X.C.I1.I2
EXAMPLE:

```
routeTable (3)
  dest (1)  policy (2)  next (3)
  130.89.16.23  1  130.89.16.23
  130.89.16.23  2  130.89.16.23
  130.89.19.121  1  130.89.16.1
  192.1.23.24  1  130.89.16.1
  192.1.23.24  2  130.89.16.4
  193.22.11.97  1  130.89.16.1
```

1.3.3.192.1.23.24.1 ⇒ 130.89.16.1

1.3.3.192.1.23.24.2 ⇒ 130.89.16.4

1 = low costs
2 = high reliability
DEFINITION OF NEW TYPES

TEXTUAL CONVENTIONS

TO REFINE SEMANTICS OF EXISTING TYPES

EXAMPLE:

RunState ::= TEXTUAL CONVENTION
  STATUS current
  DESCRIPTION "..."
  SYNTAX INTEGER{
    running(1)
    runnable(2)
    waiting(3)
    exiting(4)}
TEXTUAL CONVENTIONS

- PhysAddress
- MacAddress
- TruthValue
- AutonomousType
- InstancePointer
- VariablePointer
  - RowPointer
  - RowStatus
  - TimeStamp
  - TimeInterval
- DateAndTime
- StorageType
  - TDomain
  - TAddress

- Inet-Address...
NOTIFICATION TYPES

SMIPv2:
• MIBs MAY NOW INCLUDE NOTIFICATION TYPE MACROS

EXAMPLE:

linkUp  NOTIFICATION-TYPE
OBJECTS   {ifIndex}
STATUS   current
DESCRIPTION
   "A linkUp trap signifies that the entity has detected that the ifOperStatus object has changed to Up"
::= {snmpTraps 4}
SMIv2 relied on the 1988 version of ASN.1.

Tools for SMIv2 were relatively complex.

Certain data types were missing in SMIv2, such as 64-bit integers.

Limited facilities to reuse definitions.

SMIv2 did not allow for extensions.

New, possibly incompatible variants appeared, such as SPPI.
SMIng

TO RESOLVE THESE PROBLEMS

THE IRTF NMRG DEFINED:

SMI Next Generation (ng)

THE CHALLENGE:
CREATE A COMMON DATA DEFINITION LANGUAGE,
INDEPENDENT OF SPECIFIC PROTOCOLS
DATA VERSUS INFORMATION MODEL

INFORMATION MODEL:

"formal" / UML, ...

DATA MODEL:

MIBs

PIBs

SMIv2

SPPI

LDAP SCHEMA

CORBA IDL
DATA MODEL & "TRANSFER" PROTOCOL

DATA MODEL:
- MIBs
- PIBs

"TRANSFER" PROTOCOL:
- SNMP - BER
- COPS - BER
- LDAPv3
- LDAPv3
- CORBA IDL

INFORMATION MODEL
ROLE OF SMIng

DATA MODEL:

- MIBs / PIBs
- SNMP - BER
- COPS - BER
- LDAP SCHEMA
- CORBA IDL
IRTF-NMRG PROPOSAL FORWARDED TO IETF

SMIng WORKING GROUP FORMED

2000
SMIv3 OBJECTIVES

ALLOW ARBITRARILY NESTED DATA STRUCTURES
EASE REUSABILITY OF COMPLEX STRUCTURED TYPES

DETAILED LIST OF OBJECTIVES: RFC 3216

HOWEVER:

PROTOCOL INDEPENDENCE NO LONGER DESIRABLE
LANGUAGE EXTENSIBILITY NO LONGER DESIRABLE

FINALLY SMIv3 IS A RELATIVE SMALL UPDATE OF SMIv2
MIBs

MANAGEMENT INFORMATION BASES

CONTAIN THE MANAGED OBJECTS (VARIABLES)

THAT REPRESENT THE RESOURCES OF A SYSTEM

AND WHICH MAY BE MONITORED AND MODIFIED BY A (REMOTE) MANAGER

TO CONTROL THE BEHAVIOUR OF THAT SYSTEM
MIB DEFINITION AND MIB INSTANCE

MIB DEFINITIONS SHOULD BE KNOWN BY:

• THE IMPLEMENTORS OF THE MANAGED SYSTEM
  • THE MANAGER

THE MIB IS INSTANTIATED WITHIN THE MANAGED SYSTEM
MODULARITY

THE MANAGED OBJECTS OF A SYSTEM ARE USUALLY DEFINED IN MULTIPLE MIB DEFINITIONS

MODULES

• DIFFERENT MODULES CAN BE DEFINED BY DIFFERENT TEAMS

• MANAGEMENT FUNCTIONALITY CAN GRADUALLY BE EXTENDED

  • DIFFERENT TYPES OF SYSTEMS CAN SUPPORT DIFFERENT MIB MODULES

• VENDORS CAN EXTEND THE MANAGEMENT FUNCTIONALITY VIA PROPRIETARY MIBS
HARDWARE MIBS

HOST RESOURCES MIB

MODEM MIB

PRINTER MIB
PROTOCOL MIBS

APPLICATION
- WWW
- MAIL
- DNS
- X.500
- RDBMS
- SNMP
- ...

TRANSPORT
- TCP
- UDP

NETWORK
- OSPF
- BGP
- EGP
- IP
- ICMP
- ARP
- ...

TRANSMISSION
- 802.3
- 802.5
- FDDI
- ATM
- ADSL
- SONET
- ...

INTERFACES
PROTOCOL MIBS - EXAMPLE: MIB-II

APPLICATION
- WWW
- MAIL
- DNS
- X.500
- RDBMS
- SNMP
- ...

TRANSPORT
- TCP
- UDP

NETWORK
- OSPF
- BGP
- EGP
- IP
- ICMP
- ARP
- ...

INTERFACES
- 802.3
- 802.5
- FDDI
- ATM
- ADSL
- SONET
- ...

TRANSMISSION
NAMING OF MIBs

root

ccitt (0)
iso (1)
joint-iso-ccitt (2)

std (0) reg-auth (1) mb (2)

org (3)
dod (6)

internet (1)
directory (1) mngt (2) experimental (3) private (4) security (5) snmpV2 (6)

mib-2 (1)
system (1) interfaces (2) transmission (10) snmp (11)

... ethernet (6) token ring (9) fddi (15) adsl (94) ...
MIB-II defines the variables to manage the TCP/IP protocol stack.

- 170 variables
- RFC 1213
- SMIV1
- RFC 1156

Enhancement of MIB-I
DESIGN CRITERIA

- ESSENTIAL FOR FAULT OR CONFIGURATION MANAGEMENT
- ONLY WEAK CONTROL OBJECTS
- SMALL NUMBER OF OBJECTS
- AVOID REDUNDANCY
- EVIDENCE OF UTILITY
- DO NOT DISTURB NORMAL OPERATION
- NO IMPLEMENTATION SPECIFIC Issues
STRUCTURE

MIB-II

SYSTEM (1)  AT (3)  ICMP (5)  UDP (7)  TRANSMISSION (10)

INTERFACES (2)  IP (4)  TCP (6)  EGP (8)  SNMP (11)
MIB-II GROUPS IN A PROTOCOL STACK

- IP
- ICMP
- EGP
- AT
- TCP
- UDP
- SNMP
- SYSTEM
- INTERFACES
- TRANSMISSION
NEW VERSIONS

SYSTEM GROUP ➔ SNMPv2 MIB (RFC 3418)

INTERFACES (IF) GROUP ➔ IF-MIB (RFC 2863)

ADDRESS TRANSLATION (AT) GROUP ➔ DEPRECATED

IP & ICMP GROUPS ➔ IP-MIB (RFC 2011)

TCP GROUP ➔ TCP-MIB (RFC 2012)

UDP GROUP ➔ UDP-MIB (RFC 2013)

EGP GROUP ➔ OUTDATED (BGP)

TRANSMISSION GROUP ➔ IS PLACEHOLDER

SNMP GROUP ➔ SNMPv2 MIB (RFC 3418)
TRANSMISSION GROUP

transmission (10)

- 802.4 (8)
- 802.5 (9)
- FDDI (15)
- DS1/E1 (18)
- LAPD (16)
- DS3/E3 (30)
- RS-232 (33)
- Par.Printer (34)
- X.25 (38)
- SONET (39)
SNMPv2 MIB

RFC 3418
STANDARD

- SYSTEM GROUP
- SNMP GROUP
  - SNMP MIBObjects GROUP
    snmpTrap
    snmpTraps
    snmpSet (snmpSetSerialNo)
RFC 2863
DRAFT STANDARD

REPLACES IF GROUP OF MIB-II
  • RFC 1213
  • RFC1229 (EXTENSIONS TO THE IF GROUP)

DEFINES THE FOLLOWING MAIN TABLES:
  • ifStackTable
    • ifTable
    • ifXTable
<table>
<thead>
<tr>
<th>index</th>
<th>ifSpecific</th>
<th>ifOutOctets</th>
<th>ifOutErrors</th>
<th>ifOutUcastPkts</th>
<th>ifInErrors</th>
<th>ifInUcastPkts</th>
<th>ifLastChange</th>
<th>ifOperStatus</th>
<th>ifAdminStatus</th>
<th>ifPhysAddress</th>
<th>ifSpeed</th>
<th>ifMail</th>
<th>ifIpDest</th>
<th>ifDescr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IF PACKET COUNT

ifInUcastPkts + ifInNUcastPkts

ifInDiscards

ifInUnknownProtos

ifInErrors

ifOutUcastPkts + ifOutNUcastPkts

ifOutErrors

ifOutDiscards
IP MIB

RFC 2011

- IP GROUP
- ICMP GROUP
- IP MIB Conformance

PROPOSED STANDARD
IP PACKET COUNT

- ipInDelivers
- ipInUnknownProtos
- ipInDiscards
- ipReasmOKs
- ipReasmFails
- ipReasmReqds
- ipInAddrErrors
- ipInHdrErrors
- ipInReceives
- ipForwDatagrams
- ipOutRequests
- ipOutNoRoutes
- ipOutDiscards
- ipFragOKs
- ipFragFails
- ipFragCreates
SNMP PROTOCOL

VERSION 1

VERSION 2
  • HISTORY
  • PROTOCOL OPERATIONS

VERSION 3
  • ARCHITECTURE
  • MESSAGE STRUCTURE
  • SECURITY
OVERVIEW OF PDUs

- Manager:
  - Get
  - Set

- Agent:
  - MIB
  - Response

- Manager:
  - Get
  - GetNext

- Agent:
  - MIB
  - Response

- Manager:
  - Trap
MESSAGE & PDU STRUCTURE

variable bindings:

<table>
<thead>
<tr>
<th>NAME 1</th>
<th>VALUE 1</th>
<th>NAME 2</th>
<th>VALUE 2</th>
<th>⋯</th>
<th>⋯</th>
<th>NAME n</th>
<th>VALUE n</th>
</tr>
</thead>
</table>

SNMP PDU:

<table>
<thead>
<tr>
<th>PDU TYPE</th>
<th>REQUEST ID</th>
<th>ERROR STATUS</th>
<th>ERROR INDEX</th>
<th>VARIABLE BINDINGS</th>
</tr>
</thead>
</table>

SNMP message:

<table>
<thead>
<tr>
<th>VERSION</th>
<th>COMMUNITY</th>
</tr>
</thead>
</table>

SNMP PDU
SNMPv2

OVERVIEW:

RFCs

LIMITATIONS OF SNMPv1

HISTORY OF SNMPv2
  • HIERARCHIES
  • SECURITY

SNMPv2 PROTOCOL OPERATIONS
SNMPv2 RFCs

COMMUNICATION MODEL
- DRAFT STANDARD
- RFC 3416, RFC3417

SECURITY MODEL - SNMPv2C:
- COMMUNITY BASED SNMP
- SAME ‘SECURITY MECHANISMS’ AS SNMPv1
  - HISTORIC
  - RFC 1901

SECURITY MODEL - SNMPv2U:
- USER BASED SECURITY (AUTHENTICATION / ENCRYPTION / ACCESS CONTROL)
  - HISTORIC
  - RFC 1909, RFC1910

INFORMATION MODEL:
- STANDARD
- RFC2578, RFC2579, RFC2580
LIMITATIONS OF SNMPv1

• LIMITED ERROR CODES
• LIMITED NOTIFICATIONS
• LIMITED PERFORMANCE
• TRANSPORT DEPENDENCE
• LACK OF HIERARCHIES
• LACK OF SECURITY
• UNDOCUMENTED RULES (SMIv1)
• LIMITED DATA TYPES (SMIv1)
HIERARCHIES: ORIGINAL IDEA

MANAGER TO MANAGER (M2M) MIB

- STANDARD MIB APPROACH
- LIMITED FUNCTIONALITY
- RUN-TIME BEHAVIOUR MUST BE DEFINED AT IMPLEMENTATION TIME
HIERARCHIES: STATUS

WORK HAS MOVED TO A SEPARATE DISTRIBUTED MANAGEMENT GROUP (DISMAN)

THREE APPROACHES ARE STANDARDIZED:

• MIB BASED (EXPRESSION, EVENT AND NOTIFICATION LOG MIB)
  • SCRIPT BASED (SCRIPT AND SCHEDULE MIB)
  • REMOTE OPERATIONS BASED (REMOPS MIB)
SNMPv2 SECURITY: WHAT HAPPENED?

APRIL 1993:
PROPOSED STANDARD
FOUR EDITORS
SECURITY BASED ON PARTIES
FIRST PROTOTYPES APPEARED SOON

JUNE 1995:
PROPOSED STANDARD REJECTED BY TWO OF THE ORIGINAL EDITORS!

AUGUST 1995:
GENERAL AGREEMENT THAT PARTY BASED MODEL WAS TOO COMPLEX!
MANY NEW PROPOSALS APPEARED:
• SNMPv2C: COMMUNITY BASED
• SNMPv2U: USER BASED
• ...

1997:
NEW SNMPv3 WORKING GROUP WAS FORMED
WITH NEW EDITORS
SNMPv2 PROTOCOL OPERATIONS

- **get** response from manager to agent
- **set** response from manager to agent
- **getNext** response from manager to agent
- **getBulk** response from manager to agent
- **trap** from manager to agent
- **inform** from manager to "agent"
GET

SIMILAR TO SNMPv1, EXCEPT FOR "EXCEPTIONS"

POSSIBLE EXCEPTIONS:
- noSuchObject
- noSuchInstance

EXCEPTIONS ARE CODED WITHIN THE VARBINDS

EXCEPTIONS DO NOT RAISE ERROR STATUS AND INDEX
GET-NEXT

SIMILAR TO SNMPv1, EXCEPT FOR "EXCEPTIONS"

POSSIBLE EXCEPTIONS:
- endOfMibView

EXAMPLE
getNext(7.4.0)
response(error-status => noError, 7.4.0 => endOfMibView)
GET-BULK

NEW IN SNMPv2

TO RETRIEVE A LARGE NUMBER OF VARBINDS

IMPROVES PERFORMANCE!
GET-BULK

getBulk REQUEST HAS TWO ADDITIONAL PARAMETERS:

- non-repeators
- max-repetitions

- THE FIRST N ELEMENTS (non-repeators) OF THE VARBIND LIST ARE TREATED AS IF THE OPERATION WAS A NORMAL getnext OPERATION

- THE NEXT ELEMENTS OF THE VARBIND LIST ARE TREATED AS IF THE OPERATION CONSISTED OF A NUMBER (max-repetitions) OF REPEATED getnext OPERATIONS
SIMILAR TO SNMPv1

CONCEPTUAL TWO PHASE COMMIT:
• PHASE 1: PERFORM VARIOUS CHECKS
• PHASE 2: PERFORM THE ACTUAL SET

MANY NEW ERROR CODES ARE DEFINED
## SET: NEW ERROR CODES

<table>
<thead>
<tr>
<th>SNMPv1</th>
<th>SNMPv2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PHASE 1:</strong></td>
<td></td>
</tr>
<tr>
<td>badValue</td>
<td>wrongdoing</td>
</tr>
<tr>
<td>badValue</td>
<td>wrongEncoding</td>
</tr>
<tr>
<td>badValue</td>
<td>wrongType</td>
</tr>
<tr>
<td>badValue</td>
<td>wrongLength</td>
</tr>
<tr>
<td>badValue</td>
<td>inconsistentValue</td>
</tr>
<tr>
<td>NoSuchName</td>
<td>noAccess</td>
</tr>
<tr>
<td>NoSuchName</td>
<td>notWritable</td>
</tr>
<tr>
<td>NoSuchName</td>
<td>noCreation</td>
</tr>
<tr>
<td>NoSuchName</td>
<td>inconsistentName</td>
</tr>
<tr>
<td>genErr</td>
<td>resourceUnavailable</td>
</tr>
<tr>
<td>genErr</td>
<td>genErr</td>
</tr>
<tr>
<td><strong>PHASE 2:</strong></td>
<td></td>
</tr>
<tr>
<td>genErr</td>
<td>CommitFailed</td>
</tr>
<tr>
<td>genErr</td>
<td>undoFailed</td>
</tr>
</tbody>
</table>
SNMPv1:
- COLD START
- WARM START
- LINK DOWN
- LINK UP
- AUTHENTICATION FAILURE
- EGP NEIGHBOR LOSS

SNMPv2:
- MIBs MAY NOW INCLUDE NOTIFICATION TYPE MACROS
- FIRST TWO VARBINDS: sysUptime AND snmpTrapOID
- USES SAME FORMAT AS OTHER PDUs
CONFIRMED TRAP

ORIGINALLY TO INFORM A HIGHER LEVEL MANAGER

SAME FORMAT AS TRAP PDU

POSSIBLE ERROR: tooBig
NEW PDU TO SIGNAL PROTOCOL EXCEPTIONS / ERRORS

NO SEMANTICS DEFINED IN SNMPv2
SNMPv3

OVERVIEW:

DESIGN DECISIONS

ARCHITECTURE

SNMP MESSAGE STRUCTURE

SECURE COMMUNICATION
  • USER SECURITY MODEL (USM)

ACCESS CONTROL
  • VIEW BASED ACCESS CONTROL MODEL (VACM)

RFCs
DESIGN DECISIONS

ADDRESS THE NEED FOR SECURITY SET SUPPORT

DEFINE AN ARCHITECTURE THAT ALLOWS FOR LONGEVITY OF SNMP

ALLOW THAT DIFFERENT PORTIONS OF THE ARCHITECTURE MOVE AT DIFFERENT SLOWSTARDS TOWARDS STANDARD STATUS

ALLOW FOR FUTURE EXTENSIONS

KEEP SNMP AS SIMPLE AS POSSIBLE

ALLOW FOR MINIMAL IMPLEMENTATIONS

SUPPORT ALSO THE MORE COMPLEX FEATURES, WHICH ARE REQUIRED IN LARGE NETWORKS

RE-USE EXISTING SPECIFICATIONS, WHENEVER POSSIBLE
SNMPv3 ARCHITECTURE

SNMP ENTITY

- SNMP APPLICATIONS
  - COMMAND GENERATOR
  - COMMAND RESPONDER
  - NOTIFICATION ORIGINATOR
  - NOTIFICATION RECEIVER
  - PROXY FORWARDER
  - OTHER

SNMP ENGINE

- MESSAGE PROCESSING SUBSYSTEM
- SECURITY SUBSYSTEM
- ACCESS CONTROL SUBSYSTEM
- DISPATCHER
- OTHER
SNMPv3 ARCHITECTURE: AGENT

- COMMAND RESPONDER
- ACCESS CONTROL SUBSYSTEM
  - VIEW BASED ACCESS CONTROL
  - NOTIFICATION ORIGINATOR
- MESSAGE PROCESSING SUBSYSTEM
  - SNMPv1
  - SNMPv2c
  - SNMPv3
  - OTHER
- SECURITY SUBSYSTEM
  - COMMUNITY BASED SECURITY MODEL
  - USER BASED SECURITY MODEL
  - OTHER SECURITY MODEL

MANAGEMENT INFORMATION BASE
CONCEPTS: snmpEngineID

- SNMP ENGINE
  - snmpEngineID=1
  
- SNMP ENTITY
  
- SNMP ENGINE
  - snmpEngineID=2
  
- SNMP ENTITY
  
- SNMP ENGINE
  - snmpEngineID=3
  
- SNMP ENTITY
  
- SNMP ENGINE
  - snmpEngineID=4
CONCEPTS: snmpEngineID

SYNTAX DEFINED VIA TEXTUAL CONVENTION

OCTET STRING (5..32)

THE VALUE OF snmpEngineID MAY BE DETERMINED BY:
  • HUMAN OPERATOR
  • AUTOMATIC ALGORITHM

AUTOMATIC ALGORITHM USES:
  • PRIVATE ENTERPRISE NUMBER
  • IPv4 ADDRESS / IPv6 ADDRESS / MAC ADDRESS

TEXTUAL CONVENTION DEFINED IN SNMP FRAMEWORK MIB
### CONCEPTS: snmpEngineID

**THE TERM EngineID IS FREQUENTLY USED**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SnmpEngineID</td>
<td>The textual convention.</td>
</tr>
<tr>
<td>snmpEngineID</td>
<td>The identifier of an SNMP engine.</td>
</tr>
<tr>
<td>securityEngineID</td>
<td>Parameter of primitives in the architecture. The authoritative SNMP entity</td>
</tr>
<tr>
<td></td>
<td>(which is the receiver of a confirmed PDU, the sender of a trap).</td>
</tr>
<tr>
<td>contextEngineID</td>
<td>Parameter in messages. Identifies the engine associated with the data.</td>
</tr>
<tr>
<td>msgAuthoritativeEngineID</td>
<td>Parameter in messages. USM security parameter.</td>
</tr>
<tr>
<td>usmUserEngineID</td>
<td>An object in the snmpUsmMIB. In a simple agent, this is the agent’s own</td>
</tr>
<tr>
<td></td>
<td>snmpEngineID. It may also be the snmpEngineID of a remote SNMP engine with</td>
</tr>
<tr>
<td></td>
<td>which this user can communicate.</td>
</tr>
<tr>
<td>usmStatsUnknownEngineID</td>
<td>An object in the snmpUsmMIB.</td>
</tr>
<tr>
<td>snmpCommunityContextEngineID</td>
<td>An object in the communityMIB.</td>
</tr>
<tr>
<td>entLogicalContextEngineID</td>
<td>An object in the entityMIB.</td>
</tr>
<tr>
<td>snmpProxyContextEngineID</td>
<td>An object in the proxyMIB.</td>
</tr>
</tbody>
</table>
CONCEPTS: Context

The context can be reached from this engine, thus:
contextEngineID=1

contextName=card1
contextName=card2
MODULES OF THE SNMPv3 ARCHITECTURE

DISPATCHER AND MESSAGE PROCESSING MODULE
• SNMPv3 MESSAGE STRUCTURE
  • snmpMPDMIB
    • RFC 3412

APPLICATIONS
• snmpTargetMIB
• snmpNotificationMIB
• snmpProxyMIB
  • RFC 3413

SECURITY SUBSYSTEM
• USER BASED SECURITY MODEL
  • snmpUsmMIB
    • RFC 3414

ACCESS CONTROL SUBSYSTEM
• VIEW BASED ACCESS CONTROL MODEL
  • snmpVacmMIB
    • RFC 3415
## SNMPv3 MESSAGE STRUCTURE

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgVersion</td>
<td>USED BY MESSAGE PROCESSING SUBSYSTEM</td>
</tr>
<tr>
<td>msgID</td>
<td>USED BY SNMPv3 PROCESSING MODULE</td>
</tr>
<tr>
<td>msgMaxSize</td>
<td></td>
</tr>
<tr>
<td>msgFlags</td>
<td></td>
</tr>
<tr>
<td>msgSecurityModel</td>
<td></td>
</tr>
<tr>
<td>msgSecurityParameters</td>
<td>USED BY SECURITY SUBSYSTEM</td>
</tr>
<tr>
<td>contextEngineID</td>
<td></td>
</tr>
<tr>
<td>contextName</td>
<td></td>
</tr>
<tr>
<td>PDU</td>
<td>USED BY ACCESS CONTROL SUBSYSTEM AND APPLICATIONS</td>
</tr>
</tbody>
</table>

The diagram illustrates the structure of an SNMPv3 message, with each component's use indicated for different subsystems and applications.
### SNMPv3 PROCESSING MODULE PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgVersion</td>
<td>0..2147483647</td>
</tr>
<tr>
<td>msgID</td>
<td>0..2147483647</td>
</tr>
<tr>
<td>msgMaxSize</td>
<td>484..2147483647</td>
</tr>
<tr>
<td>msgFlags</td>
<td>authFlag, privFlag, reportableFlag</td>
</tr>
<tr>
<td>msgSecurityModel</td>
<td>SNMPv1, SNMPv2c USM</td>
</tr>
<tr>
<td>msgSecurityParameters</td>
<td></td>
</tr>
<tr>
<td>contextEngineID</td>
<td></td>
</tr>
<tr>
<td>contextName</td>
<td></td>
</tr>
</tbody>
</table>

**PDU**
SECURE COMMUNICATION VERSUS ACCESS CONTROL

AGENT

MANAGER

MIB

MANAGER AGENT

APPLICATION PROCESSES

TRANSPORT SERVICE

SECURE COMMUNICATION

ACCESS CONTROL

GET / GET-NEXT / GETBULK
SET / TRAP / INFORM

APPLICATION PROCESSES
<table>
<thead>
<tr>
<th>Threat</th>
<th>Mechanism</th>
<th>Addressed?</th>
<th>Replay</th>
<th>Masquerade</th>
<th>Integrity</th>
<th>Disclosure</th>
<th>Denial of Service</th>
<th>Traffic Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Stamp</td>
<td>MD5 / SHA-1</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>DES</td>
<td>MD5 / SHA-1</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>USM: Security Threats</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# USM MESSAGE STRUCTURE

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgVersion</td>
<td></td>
</tr>
<tr>
<td>msgID</td>
<td></td>
</tr>
<tr>
<td>msgMaxSize</td>
<td></td>
</tr>
<tr>
<td>msgFlags</td>
<td></td>
</tr>
<tr>
<td>msgSecurityModel</td>
<td></td>
</tr>
<tr>
<td>msgAuthoritativeEngineID</td>
<td></td>
</tr>
<tr>
<td>msgAuthoritativeEngineBoots</td>
<td></td>
</tr>
<tr>
<td>msgAuthoritativeEngineTime</td>
<td></td>
</tr>
<tr>
<td>msgUserName</td>
<td></td>
</tr>
<tr>
<td>msgAuthenticationParameters</td>
<td></td>
</tr>
<tr>
<td>msgPrivacyParameters</td>
<td></td>
</tr>
<tr>
<td>contextEngineID</td>
<td></td>
</tr>
<tr>
<td>contextName</td>
<td></td>
</tr>
</tbody>
</table>

**Context:**
- **REPLAY**
- **MASQUERADE/INTEGRITY/DISCLOSURE**
- **MASQUERADE/INTEGRITY**
- **DISCLOSURE**
IDEA BEHIND REPLAY PROTECTION

Nonauthoritative Engine

LOCAL NOTION OF REMOTE CLOCK

| ID | BOOTS | TIME | DATA |

Authoritative Engine

ALLOWED LIFETIME

LOCAL CLOCK

| ID | BOOTS | TIME | DATA |

+ > ?
IDEA BEHIND DATA INTEGRITY AND AUTHENTICATION

ADD THE MESSAGE AUTHENTICATION CODE (MAC) TO THE DATA AND SEND THE RESULT
IDEA BEHIND AUTHENTICATION

- **Hash Function**
- **Key**
- **Data**
- **MAC**
- **User**

Diagram shows the process of authentication using hash functions and MACs.
IDEA BEHIND THE DATA CONFIDENTIALITY (DES)

DES-KEY  DATA

DES ALGORITHM

ENCRYPTED DATA
IDEA BEHIND ENCRYPTION

DES ALGORITHM

DATA

DES-KEY

ENCRYPTED DATA

USER

ENCRYPTED DATA

DATA

DES-KEY

ENCRYPTED DATA

USER

ENCRYPTED DATA
VIEW BASED ACCESS CONTROL MODEL

ACCESS CONTROL TABLE

MIB VIEWS
# ACCESS CONTROL TABLES

<table>
<thead>
<tr>
<th>REQUIRED LEVEL OF SECURITY</th>
<th>AUTHENTICATION</th>
<th>ENCRYPTION</th>
<th>SET</th>
<th>GET / GETNEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Table</td>
<td>John</td>
<td>George</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Systems Group</td>
<td>John, Paul</td>
<td>George</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MIB VIEW</th>
<th>ALLOWED OPERATIONS</th>
<th>GET / GETNEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Table</td>
<td>GET / GETNEXT</td>
<td></td>
</tr>
<tr>
<td>Systems Group</td>
<td>GET / GETNEXT</td>
<td></td>
</tr>
</tbody>
</table>
MIB VIEWS
SNMPv3 RFCs

SNMP ENTITY

SNMP APPLICATIONS

RFC 3411

SNMP ENGINE

RFC 3412
Dispatcher

RFC 3412
Message Processing Subsystem

USM: RFC 3414
Security Subsystem

VACM: RFC 3415
Access Control Subsystem

RFC 3413

RFC 3412

RFC 3414

RFC 3415
EXTENSIBLE AGENTS FACILITATE THE EXTENSION OF SNMP AGENTS WITH NEW MIB MODULES

- SEPARATE SNMP PROTOCOL ENGINE FROM MIB INSTRUMENTATION
- ALLOW DYNAMIC ADDITION OF NEW MIB MODULE IMPLEMENTATIONS
- EXTENSIBLE AGENTS SHOULD BE TRANSPARENT
CHARACTERISTICS

REQUIRES OID REGISTRATION:

• TOP REGISTRATION
  EXAMPLE: REGISTER(mib-2)

• RANGE REGISTRATION
  EXAMPLE REGISTER(interfaces -> tcp)
POTENTIAL PROBLEMS

• TABLE ENTRIES MAY BE CREATED AND DELETED AT RUN-TIME

• ENTRIES OF A SINGLE TABLE MAY BE LOCATED IN DIFFERENT SUBAGENTS

• DUPLICATED OIDs

• GAPS

• SETS

• sysUpTime
EXAMPLE: GAPS

GET-NEXT ...
SETS AND ATOMICITY

SUB AGENT
MIB-1

SUB AGENT
MIB-2

SUB AGENT
MIB-3

MASTER AGENT

X ⇒ 1

Y ⇒ 2

TRANSACTION-LIKE APPROACH

• TEST
• COMMIT
• UNDO / CLEAN
HISTORY

SMUX (1991: RFC 1227)
SNMP MULTIPLEXING PROTOCOL

DISTRIBUTED PROTOCOL INTERFACE

RESEARCH PROTOTYPES
FOR EXAMPLE: UNIVERSITY OF TWENTE - UT-SNMPv2

COMMERCIAL PRODUCTS
FOR EXAMPLE: SNMP RESEARCH - EMANATE
(ENHANCED MANAGEMENT AGENT THROUGH EXTENSIONS)

AGENTX

PROPOSED IETF STANDARD

- RFC 2741 & RFC 2742
- http:www.scguild.com/agentx/

HAS EFFICIENT MESSAGE FORMAT AND CODING

SUPPORTS

- SUBAGENTS IMPLEMENTING SEPARATE MIB MODULES
- SUBAGENTS IMPLEMENTING ROWS IN "SIMPLE TABLES"
- SUBAGENTS SHARING TABLES ALONG NON-ROW BORDERS

NON-GOALS

- SUBAGENTS SHARING "COMPLEX TABLES"
- SUBAGENT TO SUBAGENT COMMUNICATION
AGENTX - NORMAL PDUS

- Get master
- GetNext master
- GetBulk master
- TestSet master
- CommitSet master
- UndoSet master
- CleanupSet master
- Notify master
**EXAMPLE: PDU FORMAT OF GetNext**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESSION ID</td>
<td></td>
</tr>
<tr>
<td>VERSION</td>
<td></td>
</tr>
<tr>
<td>TYPE</td>
<td></td>
</tr>
<tr>
<td>FLAGS</td>
<td></td>
</tr>
<tr>
<td>RESERVED</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td></td>
</tr>
<tr>
<td>PREFIX</td>
<td></td>
</tr>
<tr>
<td>INCLUDE</td>
<td></td>
</tr>
<tr>
<td>RESERVED</td>
<td></td>
</tr>
<tr>
<td>FIRST SUB IDENTIFIER</td>
<td></td>
</tr>
<tr>
<td>LAST SUB IDENTIFIER</td>
<td></td>
</tr>
<tr>
<td>TRANSACTION ID</td>
<td></td>
</tr>
<tr>
<td>PACKET ID</td>
<td></td>
</tr>
<tr>
<td>PAYLOAD LENGTH</td>
<td></td>
</tr>
<tr>
<td>CONTEXT (OPTIONAL)</td>
<td></td>
</tr>
<tr>
<td>OBJECT 1</td>
<td>START OF RANGE</td>
</tr>
<tr>
<td>OBJECT 1</td>
<td>END OF RANGE</td>
</tr>
<tr>
<td>OBJECT N</td>
<td>START OF RANGE</td>
</tr>
<tr>
<td>OBJECT 1</td>
<td>END OF RANGE</td>
</tr>
</tbody>
</table>
AGENTX - ADMINISTRATIVE PDUS

Open
Close

AddAgentCaps
RemoveAgentCaps

Register
Unregister

IndexAllocate
IndexDeallocate

Ping

Response
OPEN & CLOSE

TO ESTABLISH A SESSION

A UNIQUE sessionID IS ASSIGNED

SUBAGENT SPECIFIES DEFAULT TIME-OUT

RESPONSES FROM MASTER ALWAYS INCLUDE sysUpTime

SESSION CAN BE CLOSED BY MASTER OR SUBAGENT
AGENT CAPABILITIES

TO INFORM THE MASTER OF THE AGENT’S CAPABILITIES

CAPABILITIES ARE DEFINED AS:

- AN OBJECT ID
- A HUMAN READABLE STRING

THE CAPABILITIES ARE STORED IN THE sysORTable
REGISTRATION

CHOICE BETWEEN:
- TOP REGISTRATION
- RANGE REGISTRATION

PRIORITY CAN BE SPECIFIED
- TO DETERMINE THE AUTHORITATIVE SUBAGENT

TIME-OUT CAN BE SPECIFIED
INDEXALLOCATION

TO ALLOCATE ONE OR MORE TABLE ROWS

SUBAGENT REQUESTS ALLOCATION OF:
- A SPECIFIC INDEX VALUE
- AN INDEX VALUE THAT IS NOT CURRENTLY ALLOCATED
- AN INDEX VALUE THAT HAS NEVER BEEN ALLOCATED

MASTER AGENT MAINTAINS DATABASE

AFTER ALLOCATION REGISTRATION IS STILL NEEDED
TO MONITOR IF THE MASTER AGENT IS STILL ABLE TO RECEIVE AND SEND AGENTX PDUs
DISTRIBUTED MANAGEMENT

THREE APPROACHES ARE BEING DEFINED

MIB BASED
- EXPRESSION MIB
- EVENT MIB
- NOTIFICATION LOG MIB

SCRIPT BASED
- SCRIPT MIB
- SCHEDULE MIB

REMOTE OPERATIONS BASED
- REMOTE OPERATIONS MIB
EXPRESSION AND EVENT MIB

TOP LEVEL MANAGER

INTERMEDIATE LEVEL MANAGER

AGENT
EXPRESSION AND EVENT MIB: CHARACTERISTICS

- STANDARD MIB APPROACH
  - RESEMBLES THE OLD SNMPv2 M2M MIB

**EXPRESSION MIB:**
- INPUT ARE (WILDCARDED) VARIABLES OF A (LOCAL) MIB
- OPERATES ON ABSOLUTE AS WELL AS DELTA VALUES
  - RICH SET OF EXPRESSIONS
  - THE OUTPUT IS STORED IN THE VALUE TABLE
- THIS TABLE MAY SERVE AS INPUT FOR OTHER EXPRESSIONS

**EVENT MIB:**
- INPUT ARE VARIABLES OF A (REMOTE) MIB
- TRIGGERS ON CHANGES, OR TRESHOLD CROSSING
- GENERATES A NOTIFICATION OR SET OPERATION
SCRIPT MIB: CHARACTERISTICS

• FUNCTIONALITY CAN BE DEFINED AT RUN-TIME

• POWERFUL AUTONOMOUS ACTIONS

• MAY BE EASIER TO OPERATE FOR THE TOP-LEVEL MANAGER

• PROTECTION MECHANISMS NECESSARY

• DIFFERENT SCRIPT LANGUAGES
SCRIPT MIB: STRUCTURE

CONSISTS OF 6 TABLES
SCRIPT MIB: LANGUAGE TABLE

DEFINES THE LANGUAGES THIS SYSTEM SUPPORTS

- AN OID TO INDICATE THE LANGUAGE
  - THE VERSION
- AN OID TO INDICATE THE VENDOR
  - THE REVISION
  - A DESCRIPTION

TABLE IS READ ONLY
SCRIPT MIB: EXTENSIONS TABLE

DEFINES THE EXTENSIONS FOR EACH LANGUAGE

- AN OID TO INDICATE THE EXTENSION
  - THE VERSION
- AN OID TO INDICATE THE VENDOR
  - THE REVISION
  - A DESCRIPTION

TABLE IS READ ONLY
SCRIPT MIB: SCRIPT TABLE

LISTS ALL SCRIPTS KNOWN TO THE SYSTEM

ALLOWS TO:
- DOWNLOAD SCRIPTS FROM A URL (PULL MODEL)
- READ SCRIPTS FROM LOCAL NON-VOLATILE STORAGE
- STORE SCRIPTS IN LOCAL NON-VOLATILE STORAGE
- DELETE SCRIPTS FROM LOCAL NON-VOLATILE STORAGE
- LIST PERMANENT SCRIPTS (THAT CAN NOT BE CHANGED OR REMOVED)
- READ AND MODIFY THE SCRIPT STATUS (ENABLED, DISABLED, EDITING)
SCRIPT MIB: CODE TABLE

LISTS THE CODE OF A SCRIPT

ALLOWS TO:

• DOWNLOAD SCRIPTS VIA SNMP (PUSH MODEL)
• MODIFY SCRIPTS VIA SNMP (EDITING)

IMPLEMENTATION IS OPTIONAL
SCRIPT MIB: LAUNCH TABLE

ALLOWS TO:

- ASSOCIATE A SCRIPT WITH A ‘PERSON’ WHO INVOKES EXECUTION
- PROVIDE ARGUMENTS AND PARAMETERS FOR SCRIPT INVOCATION
  - INVOKE SCRIPTS WITH A SINGLE SET OPERATION
  - CONTROL THE NUMBER OF ACTIVE INVOCATIONS
  - CONTROL THE TOTAL NUMBER OF INVOCATIONS
**SCRIPT MIB: RUN TABLE**

ALLOWS TO:
- RETRIEVE STATUS INFORMATION FROM RUNNING SCRIPTS
- CONTROL RUNNING SCRIPTS (SUSPEND, RESUME, ABORT)
- RETRIEVE RESULTS FROM RECENTLY TERMINATED SCRIPTS
- CONTROL THE REMAINING MAXIMUM LIFETIME OF A RUNNING SCRIPT
- CONTROL HOW LONG SCRIPT RESULTS ARE ACCESSIBLE
SCHEDULE MIB

PERFORMS SET OPERATIONS

FOR EXAMPLE ON THE SCRIPT MIB
  • TARGET MUST BE Integer32

ON A PERIODIC OR CALENDER DRIVEN BASE
REMOTE OPERATIONS MIB

PING MIB
• TO PERFORM PING FROM A REMOTE HOST

TRACEROUTE MIB
• TO PERFORM TRACEROUTE FROM A REMOTE HOST

NAME LOOKUP MIB
• TO PERFORM NAME LOOKUP FROM A REMOTE HOST
IETF WGs - I

PROTOCOL AND DATA DEFINITION:
• SNMPv3
  • SMIlng
  • EOS
  • DISMAN

MIBs
• ADSL
• BRIDGE
• ENTITY
• ETHERNET INTERFACES & HUB

MEASURING
• BENCHMARK
  • RMON
  • IPFIX
  • PSAMP
  • PTOMAINE
NEW DEVELOPMENTS

POLICY BASED MANAGEMENT

- BANDWIDTH BROKER
- PDP (POLICY DECISION POINT)
- LDAP?
- POLICY REPOSITORY
- COPS? / SNMP?
- PEP ROUTER
- PDP ROUTER
- PDP ROUTER
COPS VERSUS SNMP

COPS:
• SPECIAL CASE OF CONFIGURATION MANAGEMENT
• HIGHER LEVEL OBJECTS THAN USUAL WITH SNMP
  • POLICY INFORMATION BASE (PIB)
• SINGLE OPERATION TO ADD OR DELETE TABLE ROWS
• RELIABLE COMMUNICATION BETWEEN PDP AND PEP (BECAUSE OF TCP)
  • EACH PEP IS CONNECTED TO SINGLE PDP

SNMP:
• INTEGRATED APPROACH TO MANAGEMENT
• POLICIES CAN BE DEFINED WITHIN MIBs
• EACH PEP MAY BE CONNECTED TO MULTIPLE PDPs
EVOLUTION OF SNMP

EOS
2001

BASIC GOAL:
IMPROVE PERFORMANCE OF SNMP BULK DATA RETRIEVALS

DIFFERENT ALTERNATIVES:
• OID DELTA COMPRESSION
• NEW PDUs (LIKE GET-TABLE)
• NEW BULK DATA TRANSFER MIBS

MANY IDEAS COME FROM IRTF-NMRG

SNMP OVER TCP MAPPING
RFC3430
XML BASED MANAGEMENT

DMTF HAS ALWAYS BEEN ACTIVE IN THIS FIELD

INTERNET STANDARDIZATION ORGANIZATIONS:

- IRTF-NMRG
- XMLCONF MAILING LIST
- IETF BOFs
- IAB WORKSHOP

VENDORS:

- JUNIPER

GOOD FOR:

- SOLVING SNMP’s DEFICIENCIES
- CONFIGURATION MANAGEMENT
- CLI INTEGRATION / CODE REUSE
<rpc>
  <get-interface-information>
    <statistics/>
  </get-interface-information>
</rpc>

<rpc-reply>
  <interface-information>
    <InOctets>123456</InOctets>
    <InErrors>789</InErrors>
    <OutOctets>654321</OutOctets>
    <OutErrors>0</OutErrors>
  </interface-information>
</rpc-reply>
WEB SERVICES FOR MANAGEMENT

RECENT RESEARCH

OASIS

ADVANTAGES:
• COMMON MIDDLEWARE TECHNOLOGY
• MANY SOFTWARE COMPONENTS
• FAST (MANAGEMENT APPLICATION) DEVELOPMENT
<definitions name="InterfaceInformation">
  ...

  <message name="Statistics">
  </message>

  <message name="StatisticsResult">
    <part name="InOctets" element="xsd:unsignedInt"/>
    <part name="InErrors" ...
    <part name="OutOctets" ...
    <part name="OutErrors" ...
  </message>

  <service name="InterfaceInfoService">
    <port ...
      {mapping on underlying protocol}
      {URI of web service}
    </port>
  </service>

</definitions>
MORE INFO ON WEB SERVICES FOR MANAGEMENT

SEE PANEL 1 TOMORROW

(10:30-12:00)
WWW SERVERS

- IETF
  http://www.ietf.org/

- The SimpleWeb
  http://www.simpleweb.org/

- The Simple Times
  http://www.simple-times.org/

- The Smurfland NM Web Server
  http://netman.cit.buffalo.edu/
BOOKS

• W. Stallings
SNMP, SNMPv2, SNMPv3 and RMON1 and 2
Third edition, Addison-Wesley, 1999
ISBN: 0-201-48534-6

• D. Zeltserman
A Practical Guide to SNMPv3 and Network Management
Prentice Hall, 1999
ISBN: 0-13-021453-1

• D. Perkins, E. McGinnis
Understanding SNMP MIBs
Prentice Hall, 1996
ISBN: 0-13-437708-7
ARTICLES

The Simple Times: *Special issue on Agent Extensibility*
Issue 4-2, April 1996

The Simple Times: *Special issue on SNMPv3*
Issue 5-1, December 1997

The Simple Times: *An overview of the AgentX Protocol*
Issue 6-1, March 1998

The Simple Times: *Special issue on SNMPv3*
Issue 7-2, November 1999

William Stallings,
*Security Comes to SNMP: The New SNMPv3 Proposed Internet Standards*

William Stallings,
*SNMPv3: A Security Enhancement for SNMP,*
IEEE Communications Survey, Q4, 1998