SNMPv2 Overview

Network Mgmt/Sec.

Jim Binkley
Outline

◆ intro
◆ SMI
◆ protocol (changes)
◆ MIB (changes)
◆ conclusion
bibliography (RFCS)

- 1901 - Intro to Community-Based v2
- 1902 - SMI
- 1903 - Textual Conventions
- 1904 - Conformance Statements
- 1905 - Protocol
- 1906 - Transport Mappings (ipx is ok ...)
- 1907 - MIB
- 1908 - Coexistence v1/v2
focus areas:

◆ managers can have managers - may enhance scalability
  – note: inform request PDU
◆ getbulkrequest for one stop table retrieval
  – actually may not be one-stop, just less overhead
◆ security (total failure, hence stick with community until v3)
◆ anti-simplicity? (see previous page)
SMI

• basically additive vis-a-vis v1

• focus on:
  – object definitions
  – tables
  – notification definitions
  – information modules
SMIv2 extends basic OID tree

- directory(1)
- mgmt(2)
- expmt(3)
- private(4)
- security(5)
- snmpv2(6)
snmpV2(6) has sub-branches

- snmpDomains (1) - transports
- snmpProxys(2) - never mind
- snmpModules(3) - module identities
- “Collections of related objects are defined in MIB modules” - rfc1906
- snmpMIB {snmpModules 1} defined in rfc1906 (#1 MODULE-IDENTITY)
  - snmpMIBObjects lives under here (start with...
OBJECT-TYPE macro redefined with a few changes

- changes to application types
- changes to access/view
- status now: current | obsolete | deprecated
- table index changes
- optional UNITS clause, text for units associated with object (celsius, whatever)
data types

- integers may now include enumerated values: red(0), green(1), blue(2)
- Counter32 - nonnegative that may be incremented, not decremented
  - initial value not meaningful
- Counter64 - max is $2^{64}-1$
- Unsigned32 == Gauge32
- Gauge32 - may increase/decrease, latch at max
- BITS - enumeration of named bits (bit flags)

Jim Binkley
MAX-ACCESS (was ACCESS)

- no write-only
- from least access to most access
  - not-accessible
  - accessible-for-notify - for traps
  - read-only
  - read-write (can replace write-only too)
  - read-create, read/write/create (for rows)
    » use set to create row instance; for example

Jim Binkley
tables

◆ still:
  – row defined by SEQUENCE {type,type,type}
  – table is SEQUENCE OF rows
◆ tables fall into two categories:
  – agent controlled - manager cannot create/delete
    » likely to be read-only rows
  – manager can create/delete rows
◆ base conceptual row: basically row index
◆ AUGMENTS can be used to add table extensions
  – vendor-specific objects added to normal table
may use object for index that is not part of table
  – text description part must explain how this works

table creation/deletion done via so-called RMON-polka (use RowStatus variable)
  – done with state machine/set and get operations
  – RowStatus has read-create access value
  – called the status column for the row
status value over-simplified

- can have following values:
  - 1. active - manager enables row
  - 2. notInService - not available (manager can set)
  - 3. notReady - not available yet
  - 4. createAndGo - autoset to active
  - 5. createAndWait - create and wait agent to setup/more complex setup (manager needed)
  - 6. destroy - manager deletes row, etc.

Jim Binkley
from rfc1906

- `sysDescr` OBJECT-TYPE
  SYNTAX DisplayString ...
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION "A textual description of the entity. blah blah ..."
  ::= {system 1}
notification-type macro

- used for traps, RFC1573 (son of interfaces)
  - linkUp NOTIFICATION-TYPE
    OBJECTS
    {ifIndex,ifAdminStatus,ifOperStatus}
    STATUS current
    DESCRIPTION "...ifOperStatus not down..."
  - ::= {snmpTraps 4}

- OBJECTS are communicated to manager upon trap
information modules

- 3 kinds of information modules
  - 1. MIB modules which include OBJECT-TYPE and NOTIFICATION-TYPE
  - 2. Compliance statements for said MIB modules, using OBJECT-GROUP and MODULE-COMPLIANCE macros
  - 3. AGENT-CAPABILITIES, state what agents can do

- All info modules must start with MODULE-IDENTIFY macro (documentation)

- OBJECT-IDENTITY used to doc objects as well
MODULE-IDENTITY

* someMIB MODULE-IDENTITY
  LAST-UPDATED string
  ORGANIZATION string
  CONTACT-INFO string
  DESCRIPTION string
  REVISION string
  DESCRIPTION string
  ::= { snmpModules 1492 }
conformance documentation

◆ to define acceptable lower bounds of implementation & actual level of implementation

◆ four macros:
  – OBJECT-GROUP
  – NOTIFICATION-GROUP
  – MODULE-COMPLIANCE
  – AGENT-CAPABILITIES
OBJECT/NOTIFICATION

GROUP

➔ OBJECT-GROUP: vendor lists OBJECTS that are supported.
  – agents may not return a value for objects not implemented (unlike in v1)

➔ NOTIFICATION-GROUP MACRO
  – simply list and describe traps supported
MODULE COMPLIANCE, etc.

- MODULE COMPLIANCE MACRO - statement of modules included in system
- AGENT-CAPABILITIES MACRO - document level of support agent claims for a given MIB group
  - may allow manager to tune itself to agent
protocol

- **SEQUENCE**
  - version (value is 1)
  - community OCTET STRING
  - data ANY

- similar to SNMPv1 although a few new messages

- one LESS PDU format (trap like all the others)
protocol PDU types

- get-request
- get-next-request
- get-bulk-request
- response
- set-request
- inform-request
- snmpV2-trap
- report (never mind) - used in sec documents and then dropped
**encapsulation of PDUs**

Note below nested in (version, community, X)

<table>
<thead>
<tr>
<th>type</th>
<th>id</th>
<th>data1</th>
<th>data2</th>
<th>var. bindings</th>
</tr>
</thead>
</table>

data1, data2 depend on type of PDU
variable bindings may be multiple OID, data
PDU formats, cont.

- Response-PDU returns error-status and error-index for data1/data2
- GetBulkRequest-PDU - nas non-repeaters and max-repetitions for data1/data2
- variable bindings may include:
  - OID and NULL used in retrieval requests
  - noSuchObject, noSuchInstance, endOfMibView
format., cont.

- **noSuchObject** - OID requested does not match
- **noSuchInstance** - OID exists, but no such column object
  - e.g., non-existant row or row is not ready yet
- **endOfMibView** - you walked off the end
get-request, and get-next-request

- similar to V1, but
- in V1 request for all variable bindings must be atomic; i.e., either all or none
- in V2, can be incomplete
  - return noSuchObject, noSuchInstance, endOfMibView possible for individual items
get-bulk-request

- not really possible to say “hey table, come over here”
- however it is possible to essentially ask for objects row by row and optimize the overall transaction (get/response)
- similar to get-next in that it uses lexicographical ordering
- get-bulk includes a list of variables in variable-bindings AND 2 variables
  - non-repeaters - # of variables in beginning of var. binds. which have single lex.. successor
  - max-repetitions - count of lexicographical variables to be returned for rest of var. binding list

Jim Binkley
get-bulk PDU remainder

<table>
<thead>
<tr>
<th></th>
<th>vb(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-rep</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>max-rep</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var. bindings</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

if non-rep == 1, applies to vb(1)
if max-rep == 2, applies to v2(2), on and asks for 2 lexico. variables beyond for each remaining variable
conceptual example:

- assume non-rep == 0, and max-rep == 2
- get-bulk(tcpConnState, tcpConnLocalAddress, tcpConnLocalPort, tcpConnRemAddress, tcpConnRemPort)
- would allow us to fetch tcp connection table two rows at a time
- if request is too big, agent will send back as much data as it can
set-request

- only difference with v1 is with responses
- still atomic (all variables are set or none)
- basically more error codes possible
snmpv2-Trap PDU

- different format from v1 BUT
- same format as other v2 messages
- var.bindings includes
  - sysUpTime.0
  - snmpTrapOID.0 (which trap)
  - OBJECTS of interest as specified
  - agent specific OBJECTS of interest
Inform-Request PDU

- sent by one manager to another to provide it with data (view as download)
- X can tell Y about Z
- OIDs of interest are passed to local app
- the response must be echoed in toto
- I have not seen this used (... yet ...)
transport mappings

- basically SNMPv2 can use more than IP/UDP including:
  - IPX
  - Appletalk
  - OSI
- we don’t care much, but IPX has used SNMP ... sans UDP/IP
interoperability with v1

- agents will speak v1 and/or v2
- managers may speak v2
  - and must talk to v1 agents
- obviously a manager that speaks v1 will not have much luck with a v2 agent ...
SMI interoperability

- key idea: MIB files should be refined to put in SMI/v2 garp BUT
- implementations do not have to change
- Changes should include
  - OBJECT definitions
  - TRAP definitions
  - Compliance definitions
  - Capabilities definitions
protocol interoperability

- manager must speak both kinds
  - can try snmp v2/v1 with agent and simply use whatever works
  - if agent doesn’t speak v2, it won’t respond (other than with error message)
  - manager can internally map v2 operations into v1 equivalents
    » getbulk turned into getnext
snmpv2 MIB

- snmpv2 MIB describes behavior of entity
- includes 3 groups
  - system group extended to allow agent to dynamically describe configurable objects
  - snmp group extensions
  - MIB objects group - deals with traps and allows > 1 manager to cooperate
**system group**

- original plus a few new objects (object resource table mostly)
- `sysORLastChange(8)` - time of last OR change
- `sysORTable(9)`
  - `sysOREntry(1)`
    - `sysORIndex`
    - `sysORID - OID`
    - `sysORDescr - string description`
    - `sysORUptime - time when row was instantiated`

- RO table of objects that can be dynamically configured
snmp group

- additions and deletions
- RO except for snmpEnableAuthenTraps
- basically SMALLER and hopefully more useful
- next slide for revised group
snmp group revised

- snmpInPackets
- snmpInBadVersions - version # wrong
- snmpInBadCommunityNames
- snmpInBadCommunityUses
- snmpInASNParseErrs
- snmpEnableAuthenTraps (RW)
- snmpSilentDrops - no response
- snmpProxyDrops - no response
MIB objects group

- for control of MIB objects (duh)
- `snmpMibObjects {snmpMIB 1}`
  - `snmpTrap`
    » `snmpTrapOID` - current trap OID (being sent)
    » `snmpTrapEnterprise` - enterprise OID
  - `snmpSet`
    » `snmpSerialNo` - basically test and set variable to support concurrent access (global and course and advisory lock ...)

Jim Binkley
interface group MIB

- RFC 1573 improves interfaces group from RFC 1213, uses SMIv2
- fix problems with 1213 and notes problems in some cases
- ifNumber can be problematic in terms of dynamic devices (no kidding)
  - value should not change until reboot
- interfaces may have sub-layers
  - virtual circuits on circuit-oriented devices

Jim Binkley
interaces, cont

- packet-oriented may be nonsense for bit-oriented devices
- interface byte count insufficient - may wrap on fast devices
- speed limited to $2^{32}-1$ (2.2Gbps) insufficient (OC-48 is 2.448 gbps)
- multicast/broadcast need to be counted explicitly
- new ifType values
more

◆ four new tables including:
  – ifXTable - extensions
  – ifStackTable - the stack
  – ifTestTable - tests
  – ifRcvAddressTable
◆ old ifInNUcastPkts, ifOutNUcastPkts deprecated
◆ ifOutQLen and ifSpecific deprecated
ifXEntry

- ifName - string
- ifInMulticastPkts - Counter32
- ifInBroadcastPkts - same
- ifOutMulticastPkts - same
- ifOutBroadcastPkts - same
- ifHCInOctets - Counter64 “high-capacity”
- ifHCInMulticastPkts - 64
- ifHCInBroadcastPkts - 64
cont.

- ifHCOutOctets
- ifHCOutUcastPkts
- ifHCOutMulticastPkts
- ifHCOutBroadcastPkts
- ifLinkUpDownTrapEnable
- ifHighSpeed - Gauge32
- ifPromiscuousMode TruthValue
- ifConnectorPresent TruthValue (true(1), false(2))
if stack table, etc.

- describe architecture of stack
- can theoretically create/destroy entries (and interfaces)
- test table - manager can test objects here if they exist
- rcv address - per interface list of addresses, should include unicast/multicast/bcast
conclusions

◆ how wide spread?
◆ much ado about nothing?
◆ get-bulk is useful
  – HPOV and full Inet/bgp routing tables
◆ more clarity and certain ambiguities removed (a good thing)
◆ not clear much of a win in MIB land -- primarily structural

Jim Binkley