TTM 4128 Exam May 19th 2008

Enclosed :
Shortened version of RFC 1157
Shortened version of RFC 1213
ASN.1 Encoding Scheme

Task 1: SNMPv1 (15%)

1.1 Explain shortly what SNMP can be used for.

SNMP is used for managing networked equipment such as Hosts, Routers, Switches, Line printers, etc. These components contain sub-components such as interface cards, protocol functionality on various layers that can be managed. This management can comprise

- *Observation of state variables related to protocols such as Ethernet, Token Ring, IP, TCP and UDP*
- *Observation, setting values and adding lines in network connectivity tables*

Observation, setting values and changing routing tables

1.2 What is SNMP SMI? List at least 3 important issues defined in SMI for SNMPv1.

SMI (The Structure of Management Information) addresses the generic aspects of the MIB, where MIB (Management Information base) defines the types of the managed objects.

SMI for SNMPv1 is defined in RFC 1155 and 1212 and comprises definition of

- *SNMP specific ASN.1 syntax and semantics*
- *Nodes in the Internet Management tree*
- *OBJECT-TYPE Macro to define managed object types*
- *Mechanism to define Indexes*

1.3 What is a MIB group?

A MIB group is a collection of related MIBs that are implemented as a whole in a managed system.

1.4 Which MIB objects can be accessed to decide if a managed component is a bridge, gateway or a router. What is the MIB group of these objects?

sysServices object in system group can be used to find the OSI level functionality (see RFC 1213) and exercise 2.

ipForwarding object in ip group can be used to determine if gateway or router.

1.5 How are MIB types identified? How are MIB object instances identified?

Entities:	**Identified/addressing**
MIBs	OBJECTIDENTIFIERS
Managed Objects (= MIB instances)	Socket addresses of SNMP entities combined with OBJECTIDENTIFIERS and indexes. Indexes are used for tables. The OBJECTIDENTIFIER is then used to indicate the tabular object while the indexes are used to select the appropriate row

Task 2: SNMPv3 (15%)

2.1 List shortly the security threats that are handled in SNMPv3.

- Modification of information by unauthorized user
- Masquerade: change of originating address by unauthorized user
- Reordering of fragments of message to modify the meaning
- Disclosure

2.2 Which security services are defined to handle the defined threats?

- Authentication module
  Data integrity: Authentication protocols such as HMAC-MD5-96 / HMAC-SHA-96
  Data origin authentication: Append to the message a unique Identifier associated with authoritative SNMP engine
- Privacy / confidentiality module:
  Encryption
- Timeliness Module:
  Certain data to be checked preventing redirection, delay and replay

Task 3: Table traversal by SNMPv1 (40%)

A Manager is managing an instance of a routing table in a router. The manager and agent communicate by SNMP version 1. Shortened versions of RFC 1157 and 1213 are enclosed. We have ip::=OBJCET INDENTIFIER{mgmt(2) mib-2(1) 4}

The following instance of the ipRoutingTable exists in the considered router:

ipRouteDest	ipRouteIfIndex	ipRouteNextHop	ipRouteMetric1	ipRouteMetric2	ipRouteType
10.0.0.99	1	89.1.1.42	5	-1	3
9.1.2.3	2	99.0.0.3	3	-1	4
10.0.0.51	3	89.1.1.42	5	-1	3

3.1 (5%) What are the OBJECT IDENTIFIER and OBJECT-SYNTAX of the columnar objects of the instance of the ipRoutingTable?

Entity	OID	SYNTAX
ipRouteDest	ipRouteEntry 1	IpAddress
ipRouteIfIndex	ipRouteEntry 2	INTEGER
ipRouteNextHop	ipRouteEntry 7	IpAdress
inRouteMetric1	ipRouteEntry 3	INTEGER
inRouteMetric2	ipRouteEntry 4	INTEGER
inRouteType	ipRouteEntry 8	INTEGER

Where ip ::= OBJCETINDENTIFIER{mgmt(2) mib-2(1) 4} and ipRouteEntry::= {ipRoutingTable 1} and ipRoutingTable::= {ip 21}

3.2 (15%) The Manager is interested in the following managed objects.

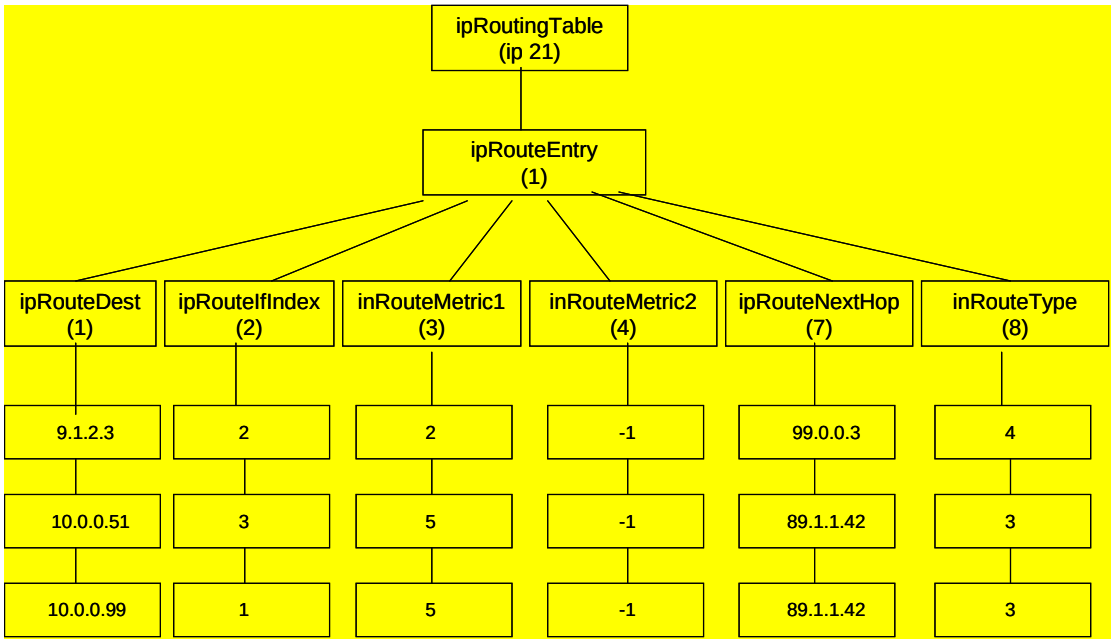
ipRouteDest	ipRouteNextHop	ipRouteMetric1
10.0.0.99	89.1.1.42	5
9.1.2.3	99.0.0.3	3
10.0.0.51	89.1.1.42	5

Show how a manager can traverse the objects {ipRouteDest, ipRouteNextHop, ipRouteMetric1} of this table by using the appropriate sequence of SNMP v1 messages, and when one SNMP message from the manager to the agent is used to access one row of the table.

Write the sequence of messages exchanged between the manager and the agent including the message from the agent that indicates that there are no more instances of the objects {ipRouteDest, ipRouteNextHop, ipRouteMetric1} in the table.

Each message shall be indicated with SNMP message name and with relevant VarBind element as parameters. This is the same convention as used in the book, in the lectures and in the RFCs.

This is the managed objects lexicographically ordered:



The management station sends to the SNMP agent a GetNextRequest-PDU containing the indicated OBJECT IDENTIFIER values as the requested variable names:
GetNextRequest (ipRouteDest, ipRouteNextHop, ipRouteMetric1)

The SNMP agent responds with a GetResponse-PDU:
GetResponse ((ipRouteDest.9.1.2.3 = "9.1.2.3"),(ipRouteNextHop.9.1.2.3 = "99.0.0.3"), (ipRouteMetric1.9.1.2.3 = 3))

The management station continues with:

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

The SNMP agent responds:
GetResponse ((ipRouteDest.10.0.0.51 = "10.0.0.51"),(ipRouteNextHop.10.0.0.51 = "89.1.1.42"), (ipRouteMetric1.10.0.0.51 = 5))

The management station continues with:
GetNextRequest (ipRouteDest.10.0.0.51,ipRouteNextHop.10.0.0.51,ipRouteMetric1.10.0.0.51)

The SNMP agent responds:
GetResponse ((ipRouteDest.10.0.0.99 = "10.0.0.99"),(ipRouteNextHop.10.0.0.99 = "89.1.1.42"), (ipRouteMetric1.10.0.0.99 = 5))

The management station continues with:
GetNextRequest(ipRouteDest.10.0.0.99,ipRouteNextHop.10.0.0.99,ipRouteMetric1.10.0.0.99)

As there are no further entries in the table, the SNMP agent returns those objects that are next in the lexicographical ordering of the known object names. This response signals the end of the routing table to the management station.

The SNMP agent responds:
GetResponse ((ipRouteIfIndex.9.1.2.3 = “2”), (ipRouteType.9.1.2.3= “4”),(ipRouteMetric.2.9.1.2.3 = -1))

3.3 (10%) We are considering the first Message going from the Manager to the Agent.

3.3.1 Define the type of the VarBind elements VarBind1, Varbind2, etc. in VarBindList by using ASN.1.

GetNextRequest (ipRouteDest, ipRouteNextHop, ipRouteMetric1)

VarBindList::=SEQUENCEOF Varbind
VarBind::=SEQUENCE{name ObjectName, value ObjectSyntax}

Data types of the VarbindList:

Varbind1::= SEQUENCE{ OBJECTIDENTIFIER, IpAddress)
Varbind2::= SEQUENCE{ OBJECTIDENTIFIER, IpAddress)
Varbind3::= SEQUENCE{ OBJECTIDENTIFIER, INTEGER)

3.3.2 Define instances of the VarBind elements defined in 3.3.1 with assigned values.

No value of IpAddess and INTEGER is to be carried on theGetNextRequest. We can then use two options:

Option 1:

We are using the generic type definition of Varbind1, Varbind2 and Varbind3

Varbind1::= SEQUENCE{ OBJECTIDENTIFIER, IpAddress)
Varbind2::= SEQUENCE{ OBJECTIDENTIFIER, IpAddress)
Varbind3::= SEQUENCE{ OBJECTIDENTIFIER, INTEGER)

and with values as follows:

varbind1 :=Varbind1{OBJECT IDENTIFIER ::= ipRouteDest, IPAdress::= 0}
varbind2::=Varbind2{ OBJECT IDENTIFIER ::= ipRouteNextHop, IPAdress::= 0}
varbind3::=Varbind3::={ OBJECT IDENTIFIER::= ipRouteMetric1, INTEGER::= 0)

Option 2:
We are using a new type definition for Varbind1, Varbind2, Varbind3:

Varbind1::= Varbind2:= Varbind3:= SEQUENCE{ OBJECTIDENTIFIER::= value, NULL)

(From RFC 1157: Some PDUs are concerned only with the name of a variable and not its value (e.g., the GetRequest-PDU). In this case, the value portion of the binding is ignored by the protocol entity. However, the value portion must still have valid ASN.1 syntax and encoding. It is recommended that the ASN.1 value NULL be used for the value portion of such bindings.)

In the following, the first and most direct (but most comprehensive) alternative is chosen.

Values of the VarbindList:

varbind1 :=Varbind1{OBJECT IDENTIFIER ::= ipRouteDest, IPAdress::= 0}
varbind2::=Varbind2{ OBJECT IDENTIFIER ::= ipRouteNextHop, IPAdress::= 0}
varbind3::=Varbind3::={ OBJECT IDENTIFIER::= ipRouteMetric1, INTEGER::= 0)

3.4 (10%) The encoding of the VarBind elements is denoted as varbind1BER, varbind2BER, etc.

Define varbind1BER

An OBJECTIDENTIFIER is encoded with each sub identifier value encoded as an octet. An exception is the iso(1) and organization (3), which are encoded in one octet as 43.

varbind1 :=Varbind1{OBJECT IDENTIFIER ::= ipRouteDest, IPAdress::= 0}
::= SEQUENCE{ OBJECTIDENTIFIER::= ipRouteDest, IpAddress::=0)

OBJECT IDENTIFIER = Universal 6 = 00000110 = 06 hex
IpAddress = [Application 0] IMPLICIT OCTET STRING (SIZE (4)) =
01000000 = 40 hex
SEQUENCE = Universal 16 constructed = 00110000 = 30 hex

ipRouteDest = 1 3 6 1 2 1 4 21 1 1

So the encoding of the instances of the OBJECT IDENTIFIERS will need 9 octets (10-1). So the length field will be 00010001 = 11 hex

Varbind 1 Encoding

OBJECT IDENTIFIER			IpAddress		
Type	length	Value	Type	length	Value
06hex	11hex	430601020104210101hex	40hex	01	00

The length of varbindBER 1 is 14 = 16 hex
varbind1BER = 3016061143060102010421010140010 0

Task 4. Semantic WEB (10 %)

Explain what RDF is? Also give a simple RDF specification example.

- RDF stands for Resource Description Framework
- RDF is a framework for describing resources on the web
- RDF provides a model for data, and a syntax so that independent parties can exchange and use it
- RDF is designed to be read and understood by computers
- RDF is not designed for being displayed to people
- RDF is written in XML
- RDF is a part of the W3C's Semantic Web Activity
- RDF is a W3C Recommendation

RDF describes resources with properties and property values.

Example :

```
<?xml version="1.0"?><RDF>
  <Description about="http://www.w3schools.com/RDF">
    <author>Jan Egil Refsnes</author>
    <homepage>http://www.w3schools.com</homepage>
  </Description>
</RDF>
```

A Resource is in this example "http://www.w3schools.com/RDF"
A Property is a Resource that has a name, such as "author" or "homepage"
A Property value is the value of a Property, such as "Jan Egil Refsnes" or "http://www.w3schools.com" (note that a property value can be another resource)

Task 5. Web-based Management (20 %)

5.1 What are the components of CIM? What is the purpose and functionality of these CIM components.

CIM consists of three components:
(i) CIM Specification
(ii) CIM Schema
(iii) CIM Extension Schema
CIM Specification defines the details for integration with other management models. CIM Specification uses an object-oriented approach to describe each entity within a schema's area of concern. The specification defines the syntax and rules. The specification defines the CIM metaschema, each of the metaschema elements, and the rules for each element.
CIM Schema provides the actual model descriptions for systems, applications, local area networks, and devices. It is a set of classes and associations that provide a framework within which it is possible to organize the information about the managed environment
CIM Schema consists of the following models:
(i) The Core Model
(ii) The Common Model
Extension schema
Represents technology and platform-specific extensions to the Common Model. Specific to environments such as operating systems. Vendors extend the model for their products by creating subclasses of objects. Applications can then transverse object instances in the standard model to manage different products in a heterogeneous environment.

5.2 Explain the WBEM architecture and the functionality of its components.

Web-Based Enterprise Management (WBEM) is a set of system/network management technologies developed to unify the management of distributed computing

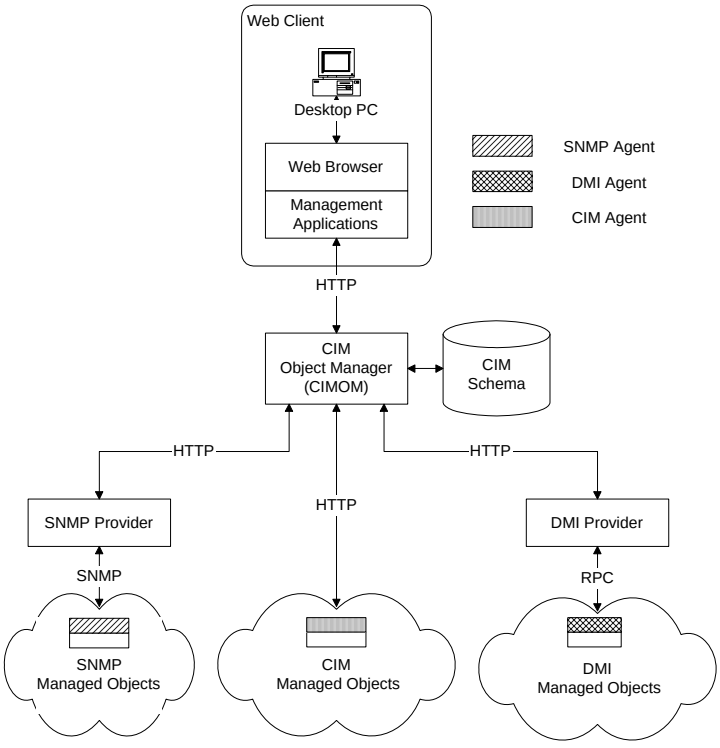


Figure 14.8 WBEM Architecture

environments. WBEM is based on CIM specification and schema, CIM-XML, CIM operations over HTTP, WS-Management.

Web client
Web client is Web browser with management applications
Application requests use CIM schema
Multiple instances of Web clients feasible
CIMOM
CIM object manger mediates between Web clients, managed objects, and CIM schema
Respond to Operations defined in “CIM Operations” spec.
Create, Modify, Delete operations on
Class, Instance, Property, Qualifier
Handle Provider Registration
Forward Requests to Providers, repositories, etc.
Read/Write access to Management Information
Maintain Class/Instance Information
Information Model
CIM Schema (Core, System,…)
Communication Model
CIM Operations over HTTP
Transport Encoding
Cim-xml – CIM/XML mapping