RICM - troisième année -
Administration de réseaux 2002-2003 – 1 heure

Exercice mail

Note: Pour les 3 questions justifier vos réponses, et indiquer s'il y a plusieurs solutions possibles ; décrivez les actions jusqu'à la remise finale du message (s'il y a lieu), mais inutile d'entrer dans le détail des protocoles ou des formats de messages.

- Sur toutes les machines, l'installation de messagerie a été faite de la manière suivante : connectivité par IP, utilisation du transport SMTP, utilisation des enregistrements DNS de type MX seulement.
- Les boites aux lettres des utilisateurs du domaine **essai.fr** sont sur la machine *srvmail.essai.fr*.
- Les seuls enregistrements DNS utiles sont :

essai.fr.	IN	MX	40	msa.service.fr.
essai.fr.	IN	MX	40	msb.service.fr.
essai.fr.	IN	MX	10	srvmail.essai.fr.

mailserv.test.fr va envoyer plusieurs mails vers des adresses du domaine *essai.fr*.

1. On suppose dans cette question qu'il n'y a aucun problème de disponibilité, d'accessibilité ni de délai dans le réseau et les machines. La machine *mailserv.test.fr* a un message à envoyer à l'adresse destination **luc@essai.fr** ; que se passe-t-il ?
2. On suppose maintenant que *srvmail.essai.fr* est en panne. La machine *mailserv.test.fr* a un message à envoyer à l'adresse destination **luc@essai.fr** ; que se passe-t-il ?
3. Que se passe-t-il une fois que *srvmail.essai.fr* est de retour en ligne ?

Exercice SNMP

Note : On trouvera en annexe les extraits utiles de la MIB-II ; on a supprimé certaines variables pour simplifier, considérez que seules les variables indiquées existent.

1. On considère la variable ipInReceives. Quelle est le « Object Identifier »(OID) sous forme numérique qui permet d'accéder à la valeur correspondante de l'agent (argument d'un échange snmpget). Que signifie sa « SYNTAX Counter » ?
2. On s'intéresse à la lecture des tables de routage par SNMP d'un machine.
   Quel est le nom, et l'OID numérique de la variable donnant le « NextHop » pour la route 127.0.0.1 ?
   Même question pour la route par défaut.
3. Proposer un algorithme pour récupérer à distance par SNMP et afficher les routes d'un machines (couple réseau IP et netmask).
4. Il est inutile de décrire les messages SNMP échangés, ne cherchez pas à raffiner l'affichage. Considérer que vous avez un langage qui fournit tous les types utiles (OID, string, …), une bibliothèque de fonctions réalisant l'échange SNMP, de type ***snmpXX(machine, OID) → …*** (la réponse ou une erreur) et une fonction ***print*** universelle.

```
RFC1155-SMI DEFINITIONS ::= BEGIN

EXPORTS -- EVERYTHING
      internet, directory, mgmt,
      experimental, private, enterprises,
      OBJECT-TYPE, ObjectName, ObjectSyntax, SimpleSyntax,
      ApplicationSyntax, NetworkAddress, IpAddress,
      Counter, Gauge, TimeTicks, Opaque;

 -- the path to the root
 internet             OBJECT IDENTIFIER ::= { iso(1) org(3) dod(6) 1 }
 directory            OBJECT IDENTIFIER ::= { internet 1 }
 mgmt                 OBJECT IDENTIFIER ::= { internet 2 }
 experimental         OBJECT IDENTIFIER ::= { internet 3 }
 private              OBJECT IDENTIFIER ::= { internet 4 }
 enterprises          OBJECT IDENTIFIER ::= { private 1 }

  -- names of objects in the MIB
   ObjectName ::= OBJECT IDENTIFIER

   -- syntax of objects in the MIB
   ObjectSyntax ::= CHOICE {
         simple                    SimpleSyntax,
         application-wide          ApplicationSyntax
     }
     SimpleSyntax ::= CHOICE {
           number                  INTEGER,
           string                  OCTET STRING,
           object                  OBJECT IDENTIFIER,
           empty                   NULL
       }
     ApplicationSyntax ::= CHOICE {
           address                 NetworkAddress,
           counter                 Counter,
           gauge                   Gauge,
           ticks                   TimeTicks,
           arbitrary               Opaque
     -- other application-wide types, as they are defined, will be added here
       }

     -- application-wide types
     NetworkAddress ::= CHOICE {
           internet                IpAddress
       }
     IpAddress ::=                 -- in network-byte order
        [APPLICATION 0] IMPLICIT OCTET STRING (SIZE (4))
     Counter ::=
        [APPLICATION 1] IMPLICIT INTEGER (0..4294967295)
     Gauge ::=
        [APPLICATION 2] IMPLICIT INTEGER (0..4294967295)
     TimeTicks ::=
        [APPLICATION 3] IMPLICIT INTEGER (0..4294967295)
     Opaque ::=
        [APPLICATION 4]                 -- arbitrary ASN.1 value,
           IMPLICIT OCTET STRING        --   "double-wrapped"
     END
```

```
RFC 1213                    MIB-II                    March 1991

        RFC1213-MIB DEFINITIONS ::= BEGIN

        IMPORTS
              mgmt, NetworkAddress, IpAddress, Counter, Gauge, TimeTicks
                 FROM RFC1155-SMI
              OBJECT-TYPE
                    FROM RFC-1212;
--  MIB-II (same prefix as MIB-I)
        mib-2      OBJECT IDENTIFIER ::= { mgmt 1 }

-- textual conventions
        DisplayString ::= OCTET STRING
                        -- This data type is used to model textual information taken from the NVT ASCII character set.
                        -- By convention, objects with this syntax are declared as having SIZE (0..255)
        PhysAddress ::= OCTET STRING
                        -- This data type is used to model media addresses.  For many types of media, this will be in a binary
                        -- representation. For example, an ethernet address would be represented as a string of 6 octets.

-- groups in MIB-II
        system              OBJECT IDENTIFIER ::= { mib-2 1 }
        interfaces          OBJECT IDENTIFIER ::= { mib-2 2 }
        ip                  OBJECT IDENTIFIER ::= { mib-2 4 }
................
-- the IP group
-- Implementation of the IP group is mandatory for all systems.

        ipForwarding OBJECT-TYPE
           SYNTAX  INTEGER {
                    forwarding(1),                  -- acting as a gateway
                    not-forwarding(2)               -- NOT acting as a gateway
                }
           ACCESS  read-write
           STATUS  mandatory
           DESCRIPTION          "The indication of whether this entity is acting as an IP gateway in respect to the forwarding
                                of datagrams received by, but not addressed to, this entity.  IP gateways forward datagrams.
                                IP hosts do not (except those source-routed via the host)."
           ::= { ip 1 }

        ipDefaultTTL OBJECT-TYPE
           SYNTAX  INTEGER
           ACCESS  read-write
           STATUS  mandatory
           DESCRIPTION          "The default value inserted into the Time-To-Live field of the IP header of datagrams by the
                                transport layer protocol."
           ::= { ip 2 }

        ipInReceives OBJECT-TYPE
           SYNTAX  Counter
           ACCESS  read-only
           STATUS  mandatory
           DESCRIPTION          "The total number of input datagrams received from interfaces, including those received in
                                error."
           ::= { ip 3 }
.......................
-- the IP routing table
-- The IP routing table contains an entry for each route presently known to this entity.
-- NOTE: plusieurs champs ont été supprimés pour simplifier le texte

        ipRouteTable OBJECT-TYPE
           SYNTAX  SEQUENCE OF IpRouteEntry
           ACCESS  not-accessible
           STATUS  mandatory
           DESCRIPTION          "This entity's IP Routing table."
           ::= { ip 21 }
```

```
ipRouteEntry OBJECT-TYPE
   SYNTAX  IpRouteEntry
   ACCESS  not-accessible
   STATUS  mandatory
   DESCRIPTION          "A route to a particular destination."
   INDEX   { ipRouteDest }
   ::= { ipRouteTable 1 }

IpRouteEntry ::=
   SEQUENCE {
      ipRouteDest                     IpAddress,
      ipRouteIfIndex                  INTEGER,
      ipRouteNextHop                  IpAddress,
      ipRouteType                     INTEGER,
      ipRouteMask                     IpAddress,
   }

ipRouteDest OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION          "The destination IP address of this route.  An entry with a value of 0.0.0.0 is considered a
                        default route."
   ::= { ipRouteEntry 1 }

ipRouteIfIndex OBJECT-TYPE
   SYNTAX  INTEGER
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION          "The index value which uniquely identifies the local interface through which the next hop of
                        this route should be reached.  The interface identified by a particular value of this index is
                        the one identified by the same value of ifIndex."
   ::= { ipRouteEntry 2 }

ipRouteNextHop OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION          "The IP address of the next hop of this route.  (In the case of a route bound to an interface
                        which is realized via a broadcast media, the value of this field is the agent's IP address on
                        that interface.)"
   ::= { ipRouteEntry 7 }

ipRouteType OBJECT-TYPE
   SYNTAX  INTEGER {
            other(1),                 -- none of the following
            invalid(2),               -- an invalidated route
            direct(3),                -- route to directly connected (sub-)network
            indirect(4)               -- route to a non-local host/network/sub-network
         }
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION "The type of route.  Note that the values direct(3) and indirect(4) refer to the notion of direct and
                        indirect routing in the IP architecture."
   ::= { ipRouteEntry 8 }

ipRouteMask OBJECT-TYPE
   SYNTAX  IpAddress
   ACCESS  read-write
   STATUS  mandatory
   DESCRIPTION          "Indicate the mask to be logical-ANDed with the destination address before being compared
                        to the value in the ipRouteDest field.
                        If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also
                        0.0.0.0.  It should be noted that all IP routing subsystems implicitly use this mechanism."
   ::= { ipRouteEntry 11 }
...................
```