DESS GI – 2001/2002

Examen de l'option Réseaux et administration de systèmes

A / Exercice "Routage"

Soit un système S ayant une interface d'adresse 200.200.200.200. À un moment donné, sa table de routage contient les informations suivantes - en résumé car des champs sont omis ci-dessous mais toutes les lignes sont présentes :

Destination	Routeur	Interface
200.200.200.192 / 26	200.200.200.200	eri0
193.25.110.128 / 25	200.200.200.193	
10.0.0.0	200.200.200.193	
193.25.110.224 / 27	200.200.200.194	
200.200.200.96 / 28	200.200.200.194	
192.0.0.0	200.200.200.194	
default	200.200.200.254	
127.0.0.1	127.0.0.1	lo0
224.0.0.0	200.200.200.200	eri0

1. Que signifie la notation « / n » ?
   Dans les 6 premières lignes, ajoutez cette indication lorsqu'elle manque - pour cela, vous supposerez que le masque du réseau est alors celui implicitement attaché à la classe.

2. Expliquez, brièvement et sans détailler d'échanges, où S envoie un paquet IP dans le cas de chacune des adresses destination suivantes :
   - 200.200.200.100
   - 200.200.200.193
   - 200.200.200.115
   - 200.200.200.255
   - 193.25.110.254

3. On ajoute un autre routeur, Rx, ayant deux interfaces d'adresses 200.200.200.194 et 200.200.200.126 sur deux réseaux /28. Ce routeur n'apparaît pas initialement dans la table de routage de S mais il se trouve que, pour l'une des destinations du point 2 ci-dessus, S reçoit un « ICMP Redirect » désignant Rx. Il vous est demandé :
   - de quelle destination il s'agit,
   - en quoi ceci modifie une de vos réponses du point 2 ci-dessus,
   - quel ajout S effectue alors à sa table de routage.

B / Exercice “SNMP”

On s’intéresse à la lecture des tables de routage par SNMP. La structure de la MIB-II est fournie en annexe, certaines variables étant omises pour simplifier. Considérez que seules les variables indiquées existent.
Les valeurs correspondent à la table de routage du système S de l’exercice précédent.
Vous n’avez pas à préciser le détail des messages échangés.
Ces conventions s’appliquent à tout l’exercice.

1. On veut savoir si S est un routeur IP ou non. Quelle variable SNMP fournit cette indication ?
   Donnez son nom et son Object Identifier (OID) numérique.

2. Quelle requête SNMP permet de lire la valeur de cette variable, quel est le résultat ? Donnez la requête et le résultat comme si l’on avait une fonction C avec des types C correspondant aux types SNMP.

.

3. Sachant qu’il existe plusieurs façons de lire une variable, donnez une autre solution à la question précédente.

4. Quelle requête SNMP permet de modifier la valeur TTL par défaut ?

5. Quel est le nom complet sous forme numérique de la variable donnant le “NextHop” pour la route 127.0.0.1 ?
   Quelle requête SNMP permet de lire la valeur de cette variable, quel est le résultat ?

6. Listez dans l’ordre de parcours les variables correspondant à la table de routage, avec leurs valeurs.
   On donne l’information suivante : la valeur IfIndex de l’interface eri0 est 1, la valeur IfIndex de l’interface lo0 est 2.

7. Proposez un algorithme pour récupérer par SNMP et afficher les tables de routage. Ne cherchez pas à raffiner l’affichage : il suffit d’afficher un type numérique ou le numéro d’index de l’interface.

C / Questions

1. Dans l’exercice A, que signifie numériquement l’affichage « default » ?
   Précisez la valeur et la longueur du préfixe. Justifiez.

2. À quoi sert le champ « TTL » des paquets IP ?
   Précisez deux utilisations.

3. Quelle différence faites-vous entre un hub et un commutateur ?
   Par « commutateur » sans autre précision, on entend ici un « commutateur de niveau 2 » Ethernet. Connaissez-vous d’autres types de commutateurs ?

4. Indiquez brièvement l’intérêt d’un protocole de routage « à état des liaisons » par rapport à un protocole « à vecteurs de distances ».

ANNEXE SNMP

```
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

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.
-- NOTA 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 }
.....................
```