MP2GI — SRR
Architecture et Administration des Réseaux

Calculatrice et tout document autorisés

Martin Heusse – Jean-Luc Richier – Vincent Roca

vendredi 25 mars 2005

1 Administration de réseau (1 heure)

Les trames détaillées en annexe A ont été capturées sur un réseau local. Une partie des champs des paquets a été plus détaillée que d'autres.

1.a) Détaillez succinctement les différentes parties qui composent premier paquet.

1.b) Expliquer les quatres lignes de la partie « *Answers* » de la « `Frame 2` ».

1.c) À quoi correspondent ces trames, à quelle action de l'utilisateur ?

1.d) Quelles informations pouvez-vous retirer de la topologie du réseau ? Reporter sur un schéma les adresses ethernet et IP des systèmes en jeu, leur localisation, le nombre de routeurs par lesquels transitent les paquets, le MTU des liens...

1.e) Les paquets TCP portent le drapeau `Don't fragment` (ce qui apparaît dans les captures sous la forme « `Flags : 0x04` »). Que se passe-t-il si une trame portant ce drapeau est trop grande pour passer sur un lien ? Que se passe-t-il si c'est un paquet UDP qui est dans la même situation ?

1.f) Choisissez des adresses IP pour ce routeur, et donnez une table de routage minimale.

1.g) Si on va voir ce routeur de plus près, on s'aperçoit qu'il n'a qu'une interface physique. À quoi ressemble le réseau dans son voisinage ? Faire un schéma.

2 Exercice SNMP ($\frac{1}{2}$ heure)

Note : on trouvera en annexe B les extraits utiles de la MIB-II ; on a supprimé certaines variables pour simplifier, considérez que seules les variables indiquées existent. Pour les Objet Identifier (OID), les noter sous forme numérique, complète à partir de la racine.

2.a) Donner l'appel SNMP (avec ses arguments) forçant à 30 la valeur par défaut du champ « nombre de saut » des paquets IP émis par S.

2.b) On considère pour S la table de routes suivante (cas d'une machine avec un interface Ethernet d'adresse 210.1.1.10). Quelles sont les variables SNMP associées à la route 200.1.128.0/24 (donner les OID et les valeurs associées) ?

Destination	Routeur	Type	Index-Interface
200.1.128.0/24	210.1.1.1	Gateway	1
default	210.1.1.3	Gateway	1
210.1.1.0/28	210.1.1.10	Direct	1

2.c) Pour chaque variable V de la réponse à la question précédente, que répond la requête "snmpgetnext(V)" (envoyée à S) ?

3 Réseaux sans fil ($\frac{1}{2}$ heure)

3.a) Comment est gérée la bidirectionnalité des transmissions dans un réseau sans fils utilisant une couche MAC de type CSMA/CA ? Et dans un réseau GSM ?

3.b) Que peut-on dire de la sécurité des réseaux Wifi ?

3.c)
- On parle d'étalement de spectre avec la norme IEEE 802.11 qui effectivement utilise l'approche DSSS. Or le spectre de puissance pour un canal donné est centré autour de la fréquence de la porteuse et est relativement peu étalé. N'est-ce pas contradictoire ?
- A quoi cet étalement sert-il en définitive ?

3.d)
- Pourquoi doit-on en général utiliser une modulation plutôt que de transmettre un flux d'informations « en bande de base » ? Que module-t-on ?
- Pourquoi parle-t-on de "Shift Keying" en modulation numérique ? Donnez un exemple mettant en évidence le compromis entre efficacité de puissance et efficacité de bande passante.

A Capture

```
Frame 1 (72 bytes on wire, 72 bytes captured)
Ethernet II, Src: 00:03:93:a6:f8:e2, Dst: 00:0e:39:af:54:80
Internet Protocol, Src Addr: 129.88.38.96 (129.88.38.96), Dst Addr: 129.88.30.10 (129.88.30.10)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 58
    Identification: 0x2a26 (10790)
    Flags: 0x00
    Fragment offset: 0
    Time to live: 64
    Protocol: UDP (0x11)
    Header checksum: 0x0973 (correct)
    Source: 129.88.38.96 (129.88.38.96)
    Destination: 129.88.30.10 (129.88.30.10)
User Datagram Protocol, Src Port: 49382 (49382), Dst Port: domain (53)
Domain Name System (query)
    Transaction ID: 0x1f70
    Flags: 0x0100 (Standard query)
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.ieee.org: type A, class inet

Frame 2 (482 bytes on wire, 482 bytes captured)
Ethernet II, Src: 00:0e:39:af:54:80, Dst: 00:03:93:a6:f8:e2
Internet Protocol, Src Addr: 129.88.30.10 (129.88.30.10), Dst Addr: 129.88.38.96 (129.88.38.96)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 464
    Identification: 0x0462 (1122)
    Flags: 0x04
    Fragment offset: 0
    Time to live: 254
    Protocol: UDP (0x11)
    Header checksum: 0x2fa0 (correct)
    Source: 129.88.30.10 (129.88.30.10)
    Destination: 129.88.38.96 (129.88.38.96)
User Datagram Protocol, Src Port: domain (53), Dst Port: 49382 (49382)
Domain Name System (response)
    Transaction ID: 0x1f70
    Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 4
    Authority RRs: 9
    Additional RRs: 9
    Queries
        www.ieee.org: type A, class inet
    Answers
        www.ieee.org: type CNAME, class inet, cname www.ieee.org.edgesuite.net
        www.ieee.org.edgesuite.net: type CNAME, class inet, cname a165.g.akamai.net
        a165.g.akamai.net: type A, class inet, addr 62.225.115.136
        a165.g.akamai.net: type A, class inet, addr 62.225.115.150
    Authoritative nameservers
        g.akamai.net: type NS, class inet, ns n6g.akamai.net
        g.akamai.net: type NS, class inet, ns n7g.akamai.net
        g.akamai.net: type NS, class inet, ns n8g.akamai.net
        g.akamai.net: type NS, class inet, ns n0g.akamai.net
        g.akamai.net: type NS, class inet, ns n1g.akamai.net
        g.akamai.net: type NS, class inet, ns n2g.akamai.net
        g.akamai.net: type NS, class inet, ns n3g.akamai.net
        g.akamai.net: type NS, class inet, ns n4g.akamai.net
        g.akamai.net: type NS, class inet, ns n5g.akamai.net
    Additional records
        n0g.akamai.net: type A, class inet, addr 193.50.203.2
```

```
        n1g.akamai.net: type A, class inet, addr 193.50.203.4
        n2g.akamai.net: type A, class inet, addr 193.50.203.48
        n3g.akamai.net: type A, class inet, addr 193.50.203.50
        n4g.akamai.net: type A, class inet, addr 193.50.203.2
        n5g.akamai.net: type A, class inet, addr 193.50.203.2
        n6g.akamai.net: type A, class inet, addr 193.45.10.92
        n7g.akamai.net: type A, class inet, addr 206.65.174.13
        n8g.akamai.net: type A, class inet, addr 193.50.203.2

Frame 3 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: 00:03:93:a6:f8:e2, Dst: 00:0e:39:af:54:80
Internet Protocol, Src Addr: 129.88.38.96 (129.88.38.96), Dst Addr: 62.225.115.136 (62.225.115.136)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 60
    Identification: 0x2a27 (10791)
    Flags: 0x04
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0xb673 (correct)
    Source: 129.88.38.96 (129.88.38.96)
    Destination: 62.225.115.136 (62.225.115.136)
Transmission Control Protocol, Src Port: 49639 (49639), Dst Port: http (80), Seq: 0, Ack: 0, Len: 0
    Source port: 49639 (49639)
    Destination port: http (80)
    Sequence number: 0
    Header length: 40 bytes
    Flags: 0x0002 (SYN)
    Window size: 65535
    Checksum: 0x5a50 (incorrect, should be 0x8c96)
    Options: (20 bytes)

Frame 4 (78 bytes on wire, 78 bytes captured)
Ethernet II, Src: 00:0e:39:af:54:80, Dst: 00:03:93:a6:f8:e2
Internet Protocol, Src Addr: 62.225.115.136 (62.225.115.136), Dst Addr: 129.88.38.96 (129.88.38.96)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 60
    Identification: 0x0000 (0)
    Flags: 0x04
    Fragment offset: 0
    Time to live: 54
    Protocol: TCP (0x06)
    Header checksum: 0xea9a (correct)
    Source: 62.225.115.136 (62.225.115.136)
    Destination: 129.88.38.96 (129.88.38.96)
Transmission Control Protocol, Src Port: http (80), Dst Port: 49639 (49639), Seq: 0, Ack: 1, Len: 0
    Source port: http (80)
    Destination port: 49639 (49639)
    Sequence number: 0
    Acknowledgement number: 1
    Header length: 40 bytes
    Flags: 0x0012 (SYN, ACK)
    Window size: 5792
    Checksum: 0x50d1 (correct)
    Options: (20 bytes)

Frame 5 (66 bytes on wire, 66 bytes captured)
Ethernet II, Src: 00:03:93:a6:f8:e2, Dst: 00:0e:39:af:54:80
Internet Protocol, Src Addr: 129.88.38.96 (129.88.38.96), Dst Addr: 62.225.115.136 (62.225.115.136)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 52
    Identification: 0x2a28 (10792)
    Flags: 0x04
    Fragment offset: 0
```

```
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0xb67a (correct)
    Source: 129.88.38.96 (129.88.38.96)
    Destination: 62.225.115.136 (62.225.115.136)
Transmission Control Protocol, Src Port: 49639 (49639), Dst Port: http (80), Seq: 1, Ack: 1, Len: 0
    Source port: 49639 (49639)
    Destination port: http (80)
    Sequence number: 1
    Acknowledgement number: 1
    Header length: 32 bytes
    Flags: 0x0010 (ACK)
    Window size: 65535
    Checksum: 0x5a48 (incorrect, should be 0x9335)
    Options: (12 bytes)

Frame 6 (413 bytes on wire, 413 bytes captured)
Ethernet II, Src: 00:03:93:a6:f8:e2, Dst: 00:0e:39:af:54:80
Internet Protocol, Src Addr: 129.88.38.96 (129.88.38.96), Dst Addr: 62.225.115.136 (62.225.115.136)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 399
    Identification: 0x2a29 (10793)
    Flags: 0x04
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (0x06)
    Header checksum: 0xb51e (correct)
    Source: 129.88.38.96 (129.88.38.96)
    Destination: 62.225.115.136 (62.225.115.136)
Transmission Control Protocol, Src Port: 49639 (49639), Dst Port: http (80), Seq: 1, Ack: 1, Len: 347
    Source port: 49639 (49639)
    Destination port: http (80)
    Sequence number: 1
    Next sequence number: 348
    Acknowledgement number: 1
    Header length: 32 bytes
    Flags: 0x0018 (PSH, ACK)
    Window size: 65535
    Checksum: 0x5ba3 (incorrect, should be 0xa3c7)
    Options: (12 bytes)
Hypertext Transfer Protocol

Frame 7 (70 bytes on wire, 70 bytes captured)
Ethernet II, Src: 00:0e:39:af:54:80, Dst: 00:03:93:a6:f8:e2
Internet Protocol, Src Addr: 62.225.115.136 (62.225.115.136), Dst Addr: 129.88.38.96 (129.88.38.96)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 52
    Identification: 0x37f2 (14322)
    Flags: 0x04
    Fragment offset: 0
    Time to live: 54
    Protocol: TCP (0x06)
    Header checksum: 0xb2b0 (correct)
    Source: 62.225.115.136 (62.225.115.136)
    Destination: 129.88.38.96 (129.88.38.96)
Transmission Control Protocol, Src Port: http (80), Dst Port: 49639 (49639), Seq: 1, Ack: 348, Len: 0
    Source port: http (80)
    Destination port: 49639 (49639)
    Sequence number: 1
    Acknowledgement number: 348
    Header length: 32 bytes
    Flags: 0x0010 (ACK)
    Window size: 6432
    Checksum: 0x78b0 (correct)
    Options: (12 bytes)
```

B RFC 1155

```
RFC 1155                        SMI                          May 1990

          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.
-- 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 }
......
-- Le reste est supprimé
```