

**INF3405 – Réseaux informatiques**

**Hiver 2017**

**TP No. 3**

**Groupe 2**

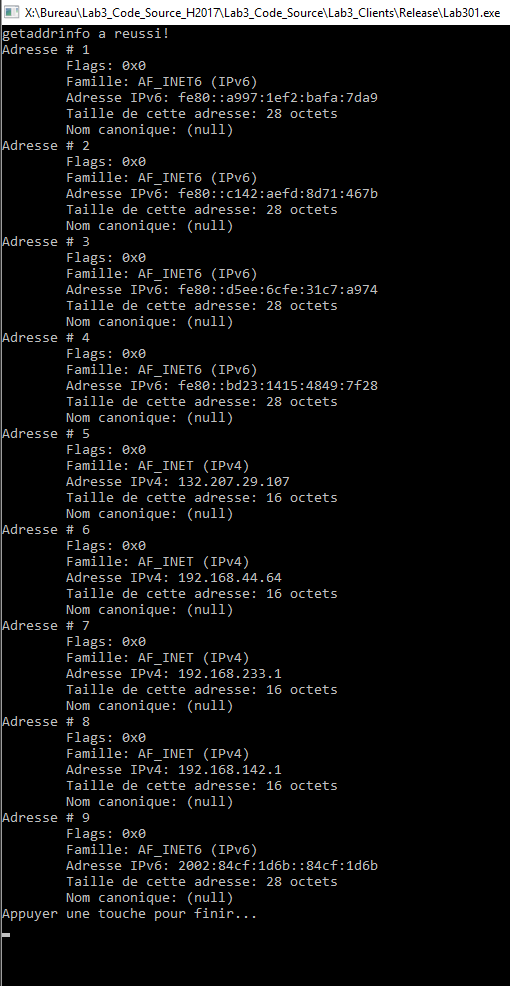
**1792473 –Richer Archambault**

**1794745 – Kevin Pantelakis**

**POSTE : L4708-07**

**Soumis à : Fabien Berquez**

1. **février 2017**

Les résultats de l’exécution du programme sont :

1)

Ce poste détient 9 adresses

2)  
Un NAT donne une adresse IPv4 et IPv6 par interface réseau qui supporte l’a norme.

3)

Il y a 5 classes d’adresse IPv4 :

Classe A : 0.0.0.0 à 127.255.255.255 et le masque : 255.0.0.0  
Classe B :128.0.0.0 à 191.255.255.255 et le masque : 255.255.0.0  
Classe C : 192.0.0.0 à 223.255.255.255 et le masque : 255.255.255.0  
Classe D : 224.0.0.0 à 239.255.255.255 et aucun masque  
Classe E : 240.0.0.0 à 255.255.255.255 et aucun masque

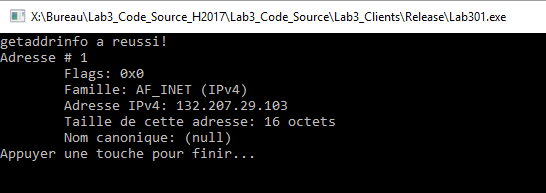
4)

L’adresse 132.207.29.107 correspond à une classe B

5)  
a) L’adresse : 10000100. 11001111. 00011101. 01101011  
Le masque : 11111111. 11111111. 00000000. 00000000

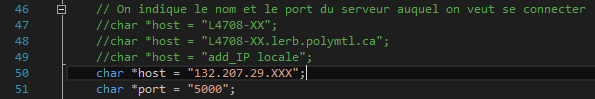
b)   
Binaire : 10000100. 11001111. 0000000. 0000000  
Décimale : 132.207.0.0

6)



7)  
Oui, car il sont dans le même réseau, et partagent le même 3e octet. Donc on peut supposer que tous les ordinateurs qui ont une adresse correspondant au masque 255.255.29.0 seront dans le même sous-réseau.

8)  
La fonction retourne à la fois les informations relatives aux host passés en paramètre (adresse IP, flags, …) ainsi qu’une valeur pour indiquer le bon fonctionnement de la fonction.

9) Les lignes sont :  
  
Le port utilisé est 5000

10)  
La famille AF\_INET

11)  
Le type choisi précise qu’il y aura un connexion en TCP.

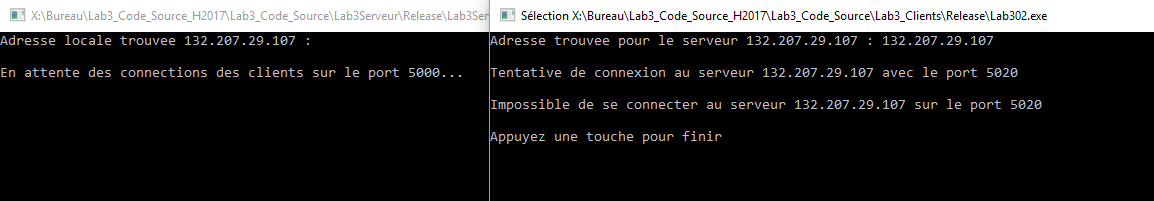
Le résultat obtenu de l’exécution est la transmission d’un mot du client vers le serveur, puis du serveur vers le client

12)  
La fonction send()

13)  
La fonction recv()

14)

Le mot envoyé est 123456 et le mot reçu est 654321 (mot de 6 lettres comme le code ne fonctionne pas pour 7). Le serveur revoit donc le mot à l’envers.

15)  
  
Il est impossible de se connecter au serveur, car le port est différent.

16)  
L’utilité du port est de diriger les flux de donnée entre les différentes applications qui utilisent une même interface réseau.

17)

PROMT\_IP:

printf("Saisir une adresse IP ");

gets\_s(host);

char end[2];

if (strlen(host) == 14) {

end[0] = host[12];

end[1] = host[13];

if (!(atoi(end) > 1 && atoi(end) < 28)) {

printf("Adresse IP invalide\n");

goto PROMT\_IP;

}

}

else {

printf("Adresse IP invalide\n");

goto PROMT\_IP;

}

18)  
Ce mécanisme est les threads sur lesquels sont utilisés des sockets.

19)  
SOCKET sd = accept(ServerSocket, (sockaddr\*)&sinRemote, &nAddrSize);

if (sd != INVALID\_SOCKET) {

cout << "Connection acceptee De : " <<

inet\_ntoa(sinRemote.sin\_addr) << ":" <<

ntohs(sinRemote.sin\_port) << "." <<

endl;

DWORD nThreadID;

CreateThread(0, 0, EchoHandler, (void\*)sd, 0, &nThreadID);

}

else {

cerr << WSAGetLastErrorMessage("Echec d'une connection.") <<

endl;

// return 1;

}

20)

|  |  |
| --- | --- |
| **Protocole** | **Port(s) par défaut** |
| HTTP | 80 |
| FTP | 20 et 21 |
| HTTPS | 443 |
| SSH | 22 |