Responsable Technique : Prieux Pierre Louis

Etude de Trame TCP/IP

Protocole TCP :

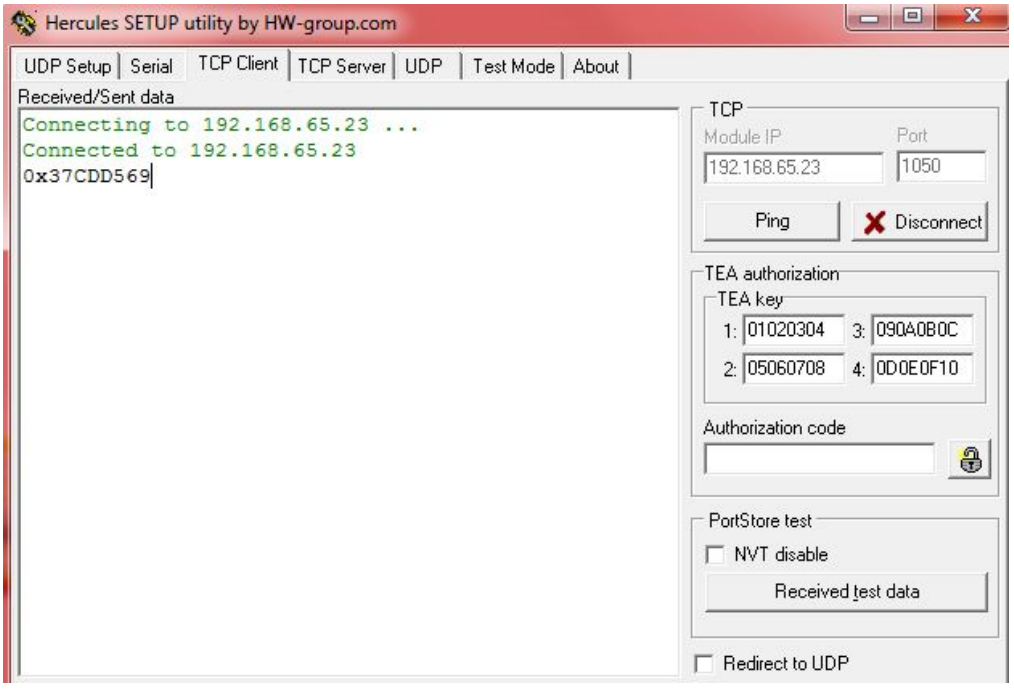
Le Protocole TCP est un moyen plus fiable que le Protocole UDP, cela nous permet de savoir si les trames envoyées ont bien été reçues par la machine concernée. La machine qui va recevoir le paquet va donc renvoyer une donnée qui permettra à la machine émettrice que la machine réceptrice a bien reçu les données. D'ou l'utilité du CRC qui vérifiera l'intégrité des données transmises.

Protocole UDP :

Contrairement au protocole TCP le protocole UDP n'est pas orienté connexion, il ne fournit pas de contrôle d'erreurs. Néanmoins l'UDP va permettre une transmission des données plus rapides que le TCP car il n'y pas de système de question/réponse.

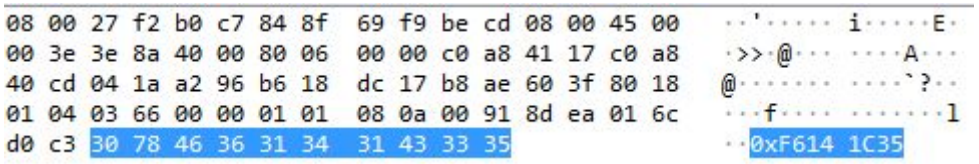
Coté Hercules :

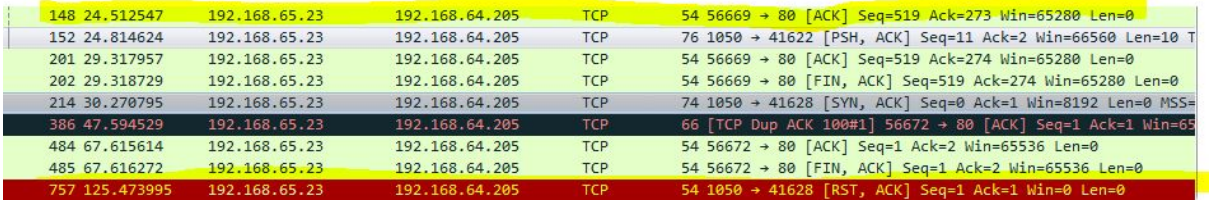
Ici le client se connecte sur l'adresse "192.168.65.23" et sur le Port :"1050", On peut voir la trame envoyer par le serveur dans le Receive/Send Data.



Coté WireShark :

Wireshark est une application qui va nous permettre une analyse approfondie de tous les protocoles du réseau sur lequel on se trouve, grâce à cela on va pouvoir apercevoir la trame que nous avons envoyée, quel protocole et à qui elle a été envoyée !





Coté du Client PHP :

```php
public function RecupBuffer($buf, $db, $id_user, $Nom, $Prenom, $Age, $Classe, $Num_carte, $imageUser)
{
    $service_port = 1050;
    $address = '192.168.65.23';
    $socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
    if ($socket === false) {
        echo "socket_create() failed: reason: " . socket_strerror(socket_last_error()) . "\n";
    }
    $result = socket_connect($socket, $address, $service_port);
    if ($result === false) {
        echo "socket_connect() failed.\($result) " . socket_strerror(socket_last_error($socket)) . "\n";
    }
    $in = "";
    $out = '';

    socket_write($socket, $in, strlen($in));
    $buf = 'This is my buffer';
    if (false !== ($bytes = socket_recv($socket, $buf, 10, MSG_WAITALL))) {
        $id = $db->query('SELECT * FROM `user` WHERE `Num_carte`= "' . $buf . '"');
        while ($Tab = $id->fetch()) {
            $id_user = $Tab["id_user"];
            $Nom = $Tab["Nom"];
            $Prenom = $Tab["Prenom"];
            $Age =  $Tab["Age"];
            $Classe =  $Tab["Classe"];
            $Num_carte = $Tab["Num_carte"];
            $imageUser = $Tab["imageUser"];
            include("perso.php");
        }
        //socket_close($socket);

        return $buf;
    }
```

Ici on a donc une liaison TCP, celle-ci va donc se connecter au serveur puis va attendre l'information que le serveur c++ va lui envoyer, si rien est envoyé rien n'est affiché.

La fonction va nous permettre de récupérer la data qui est envoyé par le serveur puis ensuite afficher ce que l'on demande.

Tout ca sur le port 1050 et à l'adresse du serveur :"192.168.65.23"

Problème rencontré :

L'un des problèmes majeurs que l'on a eu a été la connexion au serveur. Celui-ci ne pouvait que recevoir qu'une seule connexion de base, on était alors obligé de devoir relancer le serveur a chaque fois que l'on voulait changer de carte.
J'ai donc modifier le serveur afin que celui-ci accepte plusieurs connexions et qu'on ne soit pas obligé de le relancer a chaque utilisation.
J'ai donc utilisé des multithread en c++ pour éviter le problème cité juste avant.