Challenger

Mettez votre interlocuteur au défi

- Le Laboratoire aux Araignées -
pedagogie@ecole-89.com

Ce mini-projet consiste à réaliser un système de transmission d'information chiffrées entre deux partis disposant chacun du secret de l'autre.

INDEX

01 – Détails administratifs
02 – Propreté de votre rendu
03 – Norme
04 – Construction

05 – Établissement de la communication
06 – Communication
07 – Interface de fonctionnement

01 – Détails administratifs

Votre travail doit être envoyé via la forge logicielle. Le dépôt s'intitulera 2021_challenger

02 – Propreté de votre rendu

Votre rendu, c'est à dire le contenu de l'archive ou du dépôt que vous entrez sur l'interface du TechnoCentre, doit respecter **strictement** l'ensemble des règles suivantes :

- Il ne doit contenir **aucun** fichier objet. (*.o)
- Il ne doit contenir **aucun** ichier tampon. (*~, #*#)
- Il ne doit pas contenir votre production finale (programme ou bibliothèque)

03 – Règlement quant à la rédaction du code C

Vous devez respecter l'intégralité des tables de norme pour ce projet.

04 – Construction de votre rendu

Le programme de correction va construire une sous-partie déterminée de votre rendu afin d'effectuer des tests dessus. En voici les paramètres :

- Les fichiers qui seront compilés sont ceux qui auront l'extension *.c.
- Seuls les fichiers dans le(s) dossier(s) ./ seront compilés.
- Les fichiers seront compilés avec **-W -Wall -Werror**.

- Tous les fichiers seront compilés **ensemble**, cela signifie donc que chaque fonction doit être unique, et que vous pouvez utiliser les fonctions des autres exercices dans chaque exercice.

05 – Établissement de la communication

La situation est la suivante : vous êtes deux à vouloir communiquer et vous disposez chacun d'un secret partagé. Chaque parti lance un défi à l'autre parti. Ce défi consiste à envoyer une donnée depuis chaque parti à l'autre afin que cet autre la transforme et la renvoie.

Si la donnée est correctement transformée par l'autre parti, alors la confiance peut-être établie. La donnée du défi doit être aléatoire. La transformation consiste en un mélange du défi et du secret, réalisé par une fonction de digestion.

Une fois l'établissement de la communication réalisé, la communication peut avoir lieu. Les deux pairs sont surs de se connaître. Un nouveau défi peut-être renvoyé régulièrement.

L'empreinte issu de la digestion fera toujours la taille de la donnée.

06 – Communication

Le secret va ensuite servir à générer de nouvelles empreintes à l'aide du même algorithme de digestion. Le mécanisme est le suivant : l'empreinte du dernier échange (sa propre réponse au dernier défi en date si il n'y en a pas eu) auquel sera concaténé le secret sera de nouveau digérée pour donner cette nouvelle empreinte qui sera utilisée comme clef.

Cette clef sera donc jetable, utilisable pour un seul et unique envoi. La synchronisation entre les deux partis est donc essentielle : chaque parti doit garder en mémoire le nombre d'échange et générer les mêmes empreintes que son homologue, en profitant du fait que les informations servant à cette génération ont été échangés au départ.

Bien sur, l'envoi d'un nouveau défi stoppe net cette progression et la réinitialise en partant du nouveau défi.

L'algorithme de chiffrement appliquant l'empreinte comme clef à la donnée est un paramètre du programme, de même que la fonction de digestion.

07 – Interface de fonctionnement

Réalisez une bibliothèque dynamique de communication permettant d'exploiter ce mécanisme. Vous devrez fournir les fonctions suivantes :

```
// *outdata est alloué et *outlen établi à sa taille
typedef void          (*t_challenger_digest)(void                *data,
                                             size_t              len,
                                             void                **outdata,
                                             size_t              *outlen);

// data est transformé par cette fonction
typedef void          (*t_challenger_cipher)(void                *data,
                                             size_t              len,
                                             void                *key);

t_challenger          *e89_challenger_server(int                 port,
                                             t_challenger_digest digest,
                                             t_challenger_cipher cipher,
                                             void                *secret,
                                             size_t              secretlen);
t_challenger          *e89_challenger_client(const char          *host,
                                             int                 port,
                                             t_challenger_cipher cipher,
                                             t_challenger_digest digest,
                                             void                *secret,
                                             size_t              secretlen);

void                  e89_challenger_close(t_challenger          *challenger);

void                  e89_challenger_challenge(t_challenger      *challenger,
                                               t_challenger_com  *com);

t_challenger_com      *e89_challenger_poll(t_challenger          *challenger);
bool                  e89_challenger_write(t_challenger          *challenger,
                                           t_challenger_com       *com);
```

Les contenus des structures **t_challenger_com** et t_challenger sont à votre charge. t_challenger est le système général. **t_challenger_com** étant une unique communication. Coté client, com peut-être mis à **NULL** dans la fonction **e89_challenger_write** tout comme dans la fonction **e89_challenger_challenge**.

La reconnaissance des éléments réseaux tel que l'apparition d'une nouvelle connexion coté serveur sont à votre charge. Vous avez toute une structure **t_challenger_com** pour y noter les variations multiples nécessaires.

Vous devez fournir vos propres fonctions de type **t_challenger_digest** et **t_challenger_cipher** dans votre bibliothèque. Elles s'appelleront **e89_challenger_digest** et **e89_challenger_cipher**.