Université Gustave Eiffel (UGE) LIEGEY Armand
Request For Comments : 8747 WADAN Samy
Mars 2020

ChatHack : service de chat complètement légal !

Introduction :

Ce document décrit le protocole ChatHack. Le protocole permet à des clients de communiquer avec un serveur ChatHack. Via ce serveur, les clients peuvent s'échanger des messages et demander à établir des connexions privées avec les autres clients connectés. Sur ces connexions privées, les clients peuvent échanger des messages et des fichiers. Tous les transferts de données se font au-dessus de connexions TCP.
Les entiers sont tous signés en BigEndian.

Les clients se connectent à un serveur. Chaque client connecté est identifié par un pseudonyme.
Le protocole permet deux formes d'accès : un accès authentifié par un mot de passe et un accès sans mot de passe. On suppose que le serveur a accès sous une forme ou une autre à une base de données de pseudonymes et de mots de passe. Le protocole ne s'occupe pas de la création et de la mise à jour de cette base. Le serveur garantit que deux clients ne peuvent pas avoir le même pseudonyme et qu'un client authentifié sans mot de passe n'utilise pas le pseudonyme d'un utilisateur de la base de données. Une fois connectés et identifiés par un pseudonyme, les clients peuvent :

- Envoyer des messages qui seront transmis à tous les clients connectés.
- Envoyer des messages privés et des fichiers à un autre client.

Le protocole permet aux clients de faire des demandes de communication privée et d'accepter/refuser la demande de communication privée.

Une communication privée de fait de la manière suivante :
Un client A effectue une demande de communication privée avec un autre client B.
Si B accepte, B devient hébergeur d’un nouveau serveur (s’il ne l’est pas déjà), et A se connecte au serveur de B.
Les messages (et fichiers) entre A et B passent par le serveur de B.
Si un autre client C souhaite démarrer une nouvelle communication privée avec B, C se connecte au serveur de B et les deux peuvent communiquer entre eux.

Le protocole gère également les logs, les warnings et les erreurs.

Les téléchargements de fichier se font par bloc de 1024 octets.

Les restrictions du type taille maximum des messages, système d’authentification, nombre maximum de clients connectés, contenu des « messages divers »… sont libres à l’implémentation.

Table des matières

Messages provenant du client

1 - Demande de connexion au serveur

Voici comment initialiser une connexion avec le serveur.

Il y a deux façons de se connecter au serveur. Soit le client est enregistré et possède un identifiant et un mot de passe, soit il ne possède pas de mot de passe et peut ainsi se connecter en tant « qu'invité ». Il doit alors saisir un identifiant qui n'est pas déjà dans la base de données et qui soit différent de tous les identifiants des clients déjà connecté au serveur.

Le type d'authentification est déterminé par le deuxième octet de la trame TCP. Il vaut soit 0 (authentification avec mot de passe) soit 1 (authentification sans mot de passe). Dans ce deuxième cas le client ne doit pas renseigner de mot de passe.

Format de la trame – authentification avec mot de passe :

1 byte	1 byte	4 bytes	N bytes	4 bytes	M bytes
0	0	nameSize	name	passwordSize	password

Format de la trame – authentification sans mot de passe :

1 byte	1 byte	4 bytes	N bytes
0	1	nameSize	name

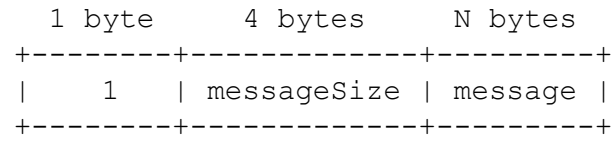
Avec :

- nameSize : Taille N du pseudo du client encodé en UTF-8
- name : Pseudo du client encodé en UTF-8 de taille N
- passwordSize : Taille M du mot de passe du client encodé en UTF-8
- password : Mot de passe du client encodé en UTF-8 de taille M

Après l'avoir envoyé le client doit ensuite attendre la réponse du serveur pour savoir si sa connexion a été acceptée ou non.

2 - Envoi d'un message public à tous les clients connectés au serveur

Pour effectuer cette action ainsi que toutes les autres, le client devra être préalablement connecté au serveur.
Il enverra une trame respectant ce format :



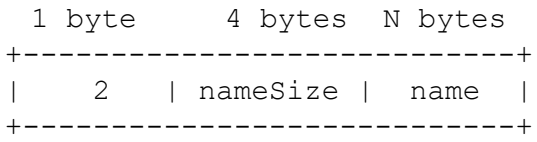
Avec :

- messageSize : Taille N du message encodé en UTF-8
- message : Message encodé en UTF-8 de taille N

Ensuite le serveur fera parvenir le message à tous les utilisateurs.

3 - Demande d'effectuer une communication privée avec un autre client

Pour réaliser une demande de communication privée, le client envoie une trame avec l'opcode = 2 ainsi que le pseudo du destinataire.



Avec :

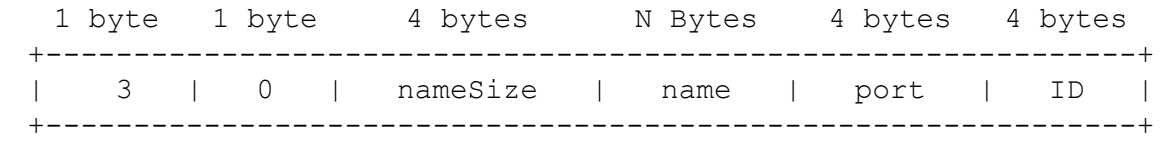
- nameSize : Taille N du pseudo du destinataire encodé en UTF-8
- name : Pseudo du destinataire encodé en UTF-8 de taille N

Le client devra alors attendre la réponse du destinataire, et si ce dernier accepte, un serveur sera créé au niveau du client destinataire de la communication privée (si pas déjà existant).

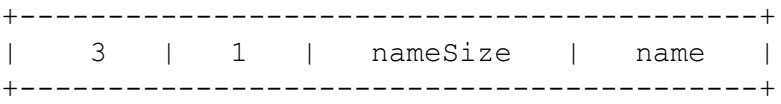
4 - Réponse à la demande de communication privée avec un autre client

Après avoir reçu la demande d'ouverture de communication privée, le client peut renvoyer une trame pour l'accepter ou la refuser. S'il accepte, il créera un serveur (si ce n'est pas déjà fait) puis indiquera son numéro de port et attribuera au demandeur un identifiant unique, qu'il utilisera à chaque communication.

Format de la trame – Acceptation de la communication privée :



Format de la trame – Refus de la communication privée :

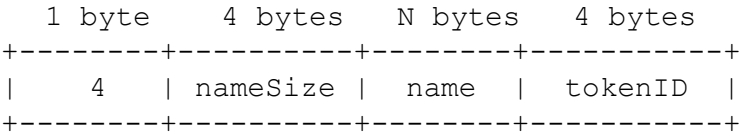


Avec :

- nameSize : Taille N du pseudo du destinataire encodé en UTF-8
- name : Pseudo du destinataire encodé en UTF-8 de taille N
- port : Numéro du port du serveur auquel le client demandeur peut se connecter
- ID : Identifiant unique pour le client demandeur

5 – Authentification du client au serveur privé

Après s'être connecté au serveur privé, le client doit prouver son identité en envoyant une trame contenant ses identifiants.

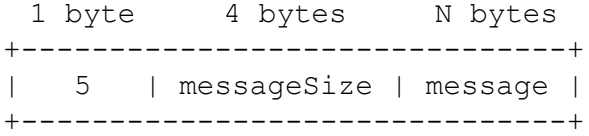


Avec :

- nameSize : Taille N du pseudo du destinataire encodé en UTF-8
- name : Pseudo du destinataire encodé en UTF-8 de taille N
- tokenID : L'identifiant attribué par le propriétaire du serveur

6 - Envoi d’un message privé à un autre client

Une fois la connexion établie entre les deux clients, il sera possible d’envoyer un message entre ces deux utilisateurs uniquement. Il faudra pour cela envoyer une trame de ce type :



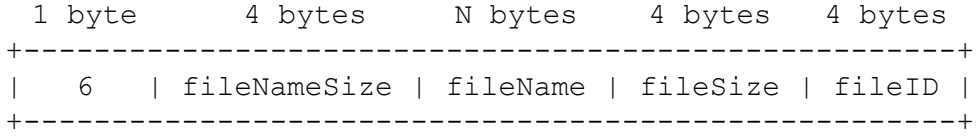
Avec :

- messageSize : Taille N du message encodé en UTF-8
- message : Message encodé en UTF-8 de taille N

7 - Envoi d'un fichier à un autre client – Initialisation

Il sera également possible d'envoyer des fichiers en message privé aux autres clients.

Cette trame va servir à initialiser l'envoi du fichier.
Pour cela il faut définir quelques paramètres.
Le destinataire connaîtra ainsi la taille du fichier et devra attendre d'avoir reçu toutes les données avant de pouvoir finir le téléchargement.

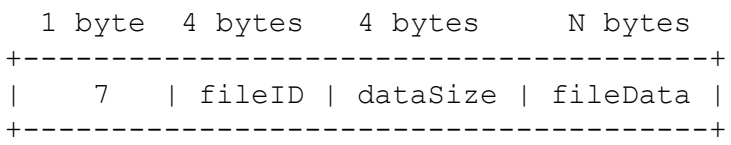


Avec :

- fileNameSize : Taille N du nom du fichier encodé en UTF-8
- fileName : Nom du fichier encodé en UTF-8 de taille N
- fileSize : Taille du fichier
- fileID : Identifiant du fichier

8 - Envoi d'un fichier à autre client – Téléchargement

Cette nouvelle trame sert à envoyer le fichier petit à petit. Chaque trame avec l'opcode = 7 correspond à un « chunk » de fichier, ainsi plusieurs trames avec le même clientId et fileId peuvent se succéder.



Avec :

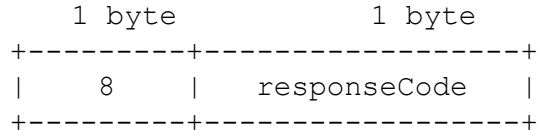
- fileID : Identifiant du fichier
- dataSize : Taille N du chunk
- fileData : Chunk du fichier de taille N

Messages à destination du client

Comme pour le client, chaque trame envoyée par le/les serveur(s) débute par un code d'opération codé sur un octet.

9 - Confirmation ou refus de la connexion du client au serveur

Lorsqu'un client envoie sa demande de connexion, le serveur lui répond avec un opcode = 8 suivi d'un autre code indiquant que la connexion est acceptée ou qu'il y a eu un problème, et que dans ce cas la connexion n'est pas établie.

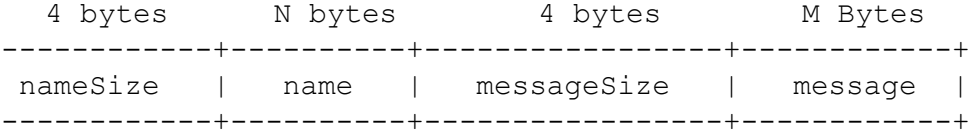


Avec responseCode qui peut avoir les valeurs suivantes :

- 0 : Connexion acceptée
- 1 : Connexion refusée - Identifiants invalides
- 2 : Connexion refusée – Login déjà existant
- 3 : Connexion refusée – Le serveur n'accepte plus de nouveaux clients.

10 - Envoi d'un message public de la part d'un client vers tous les autres clients connectés

Lorsqu'un client émet un message public, le serveur le redistribue à tous les autres clients connectés en leur envoyant un paquet de la forme suivante :

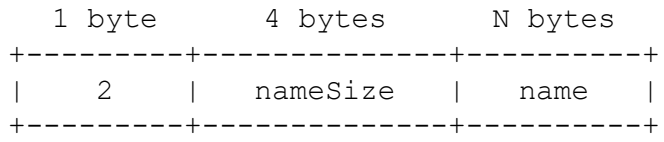


Avec :

- nameSize : Taille N du pseudo de l'émetteur du message encodé en UTF-8
- name : Pseudo de l'émetteur du message encodé en UTF-8 de taille N
- messageSize : Taille M du message encodé en UTF-8
- message : Message encodé en UTF-8 de taille M

11 - Notification d’un client qu’un autre client souhaite effectuer une communication privée

Lorsqu’un client effectue une demande de connexion privée avec un autre client, ce dernier reçoit une trame du serveur avec un opcode = 2 ainsi que le pseudo du demandeur.



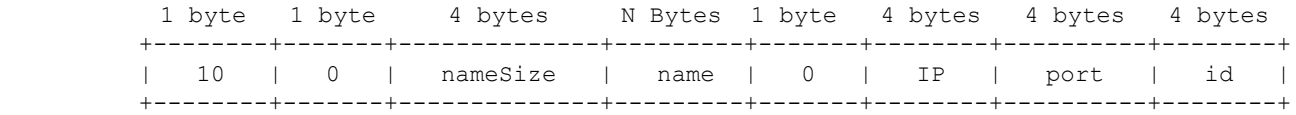
Avec :

- nameSize : Taille N du pseudo du demandeur encodé en UTF-8
- name : Pseudo du demandeur encodé en UTF-8 de taille N

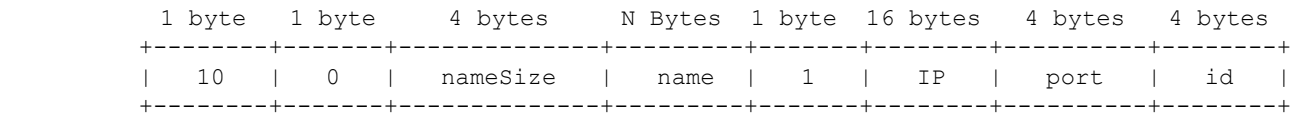
12 - Réponse à la demande de communication privée

Lorsqu'un client effectue une demande de connexion privée avec un autre client, ce dernier reçoit une trame du serveur avec un opcode = 10, suivi d'un autre code spécifiant la réponse.
Ce code possède 3 valeurs : le destinataire a accepté la demande de communication privée, le destinataire l'a refusée, le destinataire demandé n'existe pas ou n'est pas connecté au serveur.
Si le destinataire a accepté, le client reçoit un identifiant et les informations pour se connecter au serveur du destinataire : adresse IP et port.

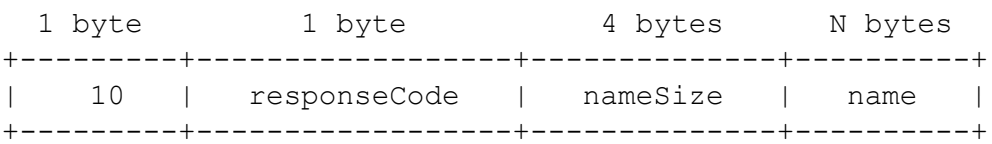
Format de la trame – Demande acceptée et IP version IPv4 :



Format de la trame – Demande acceptée et IP version IPv6 :



Format de la trame – Demande refusée :



Avec :

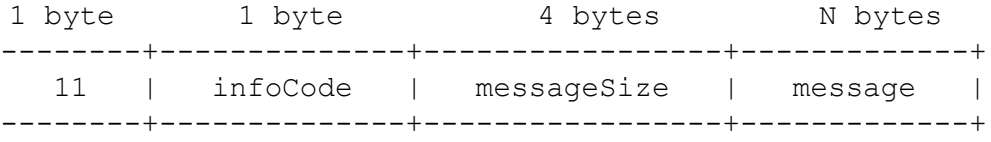
- responseCode : Valeurs possibles :
  - 1 : Le client destinataire a refusé la demande de communication privée
  - 2 : Le client destinataire n'existe pas ou n'est pas connecté au serveur
- nameSize : Taille N du pseudo du client destinataire encodé en UTF-8
- name : Pseudo du client encodé en UTF-8 de taille N
- IP : Adresse IPv4 ou IPv6 du serveur
- Port : Numéro de port du serveur

13 - Messages divers

Le serveur peut à tout moment envoyer un message à un ou plusieurs clients selon un évènement. Ainsi il peut envoyer :

- Des messages à titre informatif, tels que les logs.
- Des messages d'erreurs, tels que l'interruption de la connexion d'un client
- Des messages d'avertissement : message trop long, fichier trop volumineux…

Chacun de ces messages est envoyé dans une trame portant l'opcode 12, suivi d'un autre code indiquant le type d'information, et enfin le texte de l'information.



Avec :

- infoCode : Indique le type de message :
  - 0 : Log
  - 1 : Warning
  - 2 : Erreur
- messageSize : Taille N du message encodé en UTF-8
- message : Message encodé en UTF-8 de taille N

Auteurs :

LIEGEY Armand
aliegey@etud.u-pem.fr

WADAN Samy
swadan@etud.u-pem.fr

Université Gustave Eiffel
Master 1 – Informatique
Groupe Apprentis
2019 - 2020