Tomahawk

v.alpha

Généré par Doxygen 1.8.11

Table des matières

Chapitre 1

Page d'accueil

Le projet Tomahawk est dévelopé et maintenu comme un serveur HTTP open-source pour les systèmes d'exploitation moderne, ce qui inclut UNIX et Windows. Le but étant de procurer un serveur peu sécurisé, peu fiable, moyennement efficace et conforme aux standards HTTP du siècle derni/r.< >

Le serveur Tomahawk

Ce serveur sera bientôt un incontournable pour toutes les entreprises.

L'équipe de développement

L'équipe de développement est fantastique, et mettra tout en oeuvre pour maintenir le développement de Tomahawk.

Doués, passionnés et impliqués ils sont prêts à recevoir toutes remarques et répondront avec plaisir au mail d'admirateur. Comme le dis Dennis MacAlistair Ritchie à leurs propos : "Ces mecs sont formidables".

Chapitre 2

Index des structures de données

2.1 Structures de données

Liste des structures de données avec une brève description :

Chapitre 3

Index des fichiers

3.1 Liste des fichiers

Liste de tous les fichiers documentés avec une brève description :

Chapitre 4

Documentation des structures de données

4.1 Référence de la structure HTTP_Node

Représente un noeud de l'arbre utilisé par le parseur.

`#include <http_node.h>`

Champs de données

- char * name
- int beg
- int end
- HTTP_Node ** childs
- int nb_childs

4.1.1 Description détaillée

Représente un noeud de l'arbre utilisé par le parseur.

4.1.2 Documentation des champs

4.1.2.1 int beg

Début de l'élément dans la requête HTTP.

4.1.2.2 HTTP_Node** childs

Ensemble de ses enfants.

4.1.2.3 int end

Fin de l'élément dans la requête HTTP.

4.1.2.4 char* name

Nom du noeud.

4.1.2.5 int nb_childs

Nombre d'enfant.

La documentation de cette structure a été générée à partir du fichier suivant :
— src/http_node.h

Chapitre 5

Documentation des fichiers

5.1 Référence du fichier src/api.h

API du Parseur HTTP 1.0.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Fonctions

- int parser (char *buf, unsigned int len, char *search, void(*callback)())
  Fonction demande par l'API de test.
- char * copierChaine (char *buf, char *res, int beg, int end)
  Fonction permettant de recopier une sous-chaine.

5.1.1 Description détaillée

API du Parseur HTTP 1.0.

Auteur

R. Vidal / A. Lorber / T. Jacumin

Version

1.0

5.1.2 Documentation des fonctions

5.1.2.1 char* copierChaine (char * *buf,* char * *res,* int *beg,* int *end*)

Fonction permettant de recopier une sous-chaine.

Paramètres

buf	Pointeur vers le premier caractère de la chaine.
res	Pointeur vers le resultat.
beg	Debut de la sous-chaine.
end	Fin de la sous-chaine.

Renvoie

Un pointeur vers res.

5.1.2.2 int parser (char * *buf*, unsigned int *len*, char * *search*, void(*)() *callback*)

Fonction demande par l'API de test.

Paramètres

Buf	pointeur vers le premier caractère du flux.
Len	longueur du flux à traiter.
Search	champs à chercher.
Callback	fonction de callback.

Renvoie

-1 si le message est valide syntaxiquement, sinon un entier correspondant à l'indice dans la chaine buf ou le parsuer à détecter une erreur syntaxique.

5.2 Référence du fichier src/http_node.h

Gestion de l'arbre n-aire utiliser par le parseur HTTP.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Structures de données

— struct HTTP_Node
Représente un noeud de l'arbre utilisé par le parseur.

Définitions de type

— typedef struct HTTP_Node **HTTP_Node**

Fonctions

- void init_HTTP_Node (char *name, HTTP_Node *node)
  Fonction de création d'une nouvelle instance d'un objet HTTP_Node.
- void addChild_HTTP_Node (HTTP_Node *node, HTTP_Node *child)
  Fonction d'ajout d'un fils à un noeud.
- void free_HTTP_Node (HTTP_Node *node)
  Fonction de suppression d'un noeud.
- void free_HTTP_Tree (HTTP_Node *root)
  Fonction de suppression d'un arbre.
- HTTP_Node * found_HTTP_Node (HTTP_Node *root, char *string)
  Fonction de recherche dans un arbre.
- void **foundAll_HTTP_Node** (HTTP_Node *root, char *string, int *nbFound, HTTP_Node **result)
- void foundAll_HTTP_Node_rec (HTTP_Node *root, char *string, HTTP_Node **result, int *nbFound)
  Fonction de recherche dans un arbre.
- void print_HTTP_Node (char *request, HTTP_Node *node)
  Fonction pour afficher le contenu d'un HTTP_Node.
- void print_HTTP_Tree (char *request, HTTP_Node *root, int level)
  Fonction pour afficher le contenu d'un arbre de HTTP_Node.

5.2.1 Description détaillée

Gestion de l'arbre n-aire utiliser par le parseur HTTP.

Auteur

R. Vidal / A. Lorber / T. Jacumin

Version

1.0

5.2.2 Documentation des fonctions

5.2.2.1 void addChild_HTTP_Node (HTTP_Node * *node*, HTTP_Node * *child*)

Fonction d'ajout d'un fils à un noeud.

Paramètres

node	Pointeur vers le noeud parent.
child	Pointeur vers le noeud fils.

5.2.2.2 HTTP_Node* found_HTTP_Node (HTTP_Node * *root*, char * *string*)

Fonction de recherche dans un arbre.

Paramètres

root	Pointeur vers le noeud racine de l'arbre à supprimer.
string	La chaine à rechercher.

Renvoie

Le noeud qui à pour nom la chaine string.

5.2.2.3 void foundAll_HTTP_Node_rec (HTTP_Node * *root*, char * *string*, HTTP_Node ** *result*, int * *nbFound*)

Fonction de recherche dans un arbre.

Paramètres

root	Pointeur vers le noeud racine de l'arbre à supprimer.
string	La chaîne à rechercher.
result	Tableau des noeuds correspondant à la recherche.
nbFound	Le nombre de noeud trouvé.

5.2.2.4 void free_HTTP_Node (HTTP_Node * *node*)

Fonction de suppression d'un noeud.

Paramètres

node	Pointeur vers le noeud à supprimer.

5.2.2.5 void free_HTTP_Tree (HTTP_Node * *root*)

Fonction de suppression d'un arbre.

Paramètres

node	Pointeur vers le noeud racine de l'arbre à supprimer.

5.2.2.6 void init_HTTP_Node (char * *name*, HTTP_Node * *node*)

Fonction de création d'une nouvelle instance d'un objet HTTP_Node.

Paramètres

name	Nom du noeud.
node	Pointeur vers le noeud à initialiser.

5.2.2.7 void print_HTTP_Node (char * *request*, HTTP_Node * *node*)

Fonction pour afficher le contenu d'un HTTP_Node.

Paramètres

request	Chaîne contenant la requête.
node	Noeud à afficher.

5.2.2.8 void print_HTTP_Tree (char * *request*, HTTP_Node * *root*, int *level*)

Fonction pour afficher le contenu d'un arbre de HTTP_Node.

Paramètres

request	Chaîne contenant la requête.
node	Noeud racine.
level	Pour l'indentation (laisser à 0).

5.3 Référence du fichier src/http_parser.h

Parseur HTTP 1.0.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "http_node.h"
```

Macros

- #define **SP** ' '
- #define **DEBUG**
- #define **DEBUG_PRINT**(x) printf(x)

Fonctions

- int parse_HTTP_message (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le HTTP-message.
- int parse_start_line (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le start-line.
- int parse_status_line (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le status-line.
- int parse_header_field (char *request, int *cursor, HTTP_Node *header_field)
  Fonction pour parser le header-field.
- int parse_field_name (char *request, int *cursor, HTTP_Node *header_field, HTTP_Node *field_name, HTTP_Node *field_value)
  Fonction pour parser le field-name.
- int parse_field_value (char *request, int *cursor, HTTP_Node *header_field, HTTP_Node *field_name, HTTP_Node *field_value)
  Fonction pour parser le field-content (utilisée par header-field).
- int parse_field_content (char *request, int *cursor, HTTP_Node *header_field, HTTP_Node *field_name, HTTP_Node *field_value)
  Fonction pour parser le field-content (utilisée par header-field).

- int parse_obs_fold (char *request, int *cursor, HTTP_Node *header_field, HTTP_Node *field_name, HTTP_Node *field_value)
  Fonction pour parser le obs-fold (utilisée par header-field).
- int parse_message_body (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le message-body.
- int parse_request_target (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le request-target.
- int parse_request_line (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le request-line.
- int parse_origin_form (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le origin-form.
- int parse_absolute_form (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le absolute-form.
- int parse_authority_form (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le authority-form.
- int parse_IPv4 (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le IPv4.
- int parse_user_info (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le user-info.
- int parse_regname (char *request, int *cursor, HTTP_Node *node)
  Fonction pour parser le regname.
- int parse_string (char *str, int *cursor, char *cmp_str)
  Fonction pour vérifier l'égalité d'une chaîne et d'une sous-chaîne.
- int isDIGIT (char *request, int *cursor)
  Fonction pour vérifier si on a bien un DIGIT.
- int isTchar (char *request, int *cursor)
  Fonction pour vérifier si on a bien un TCHAR.
- int isVCHAR (char *request, int *cursor)
  Fonction pour vérifier si on a bien un VCHAR.
- int isObstext (char *request, int *cursor)
  Fonction pour vérifier si on a bien un ObsText.
- int isFieldvchar (char *request, int *cursor)
  Fonction pour vérifier si on a bien un Field-VCHAR.
- int isUnreserved (char *request, int *cursor)
  Fonction pour vérifier si on a bien un Unreserved.

5.3.1 Description détaillée

Parseur HTTP 1.0.

Auteur

R. Vidal / A. Lorber / T. Jacumin

Version

1.0

5.3.2 Documentation des fonctions

5.3.2.1 int isDIGIT (char * *request*, int * *cursor*)

Fonction pour vérifier si on a bien un DIGIT.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.

Renvoie

Retourne 1 si le prochain caractere est un DIGIT, 0 sinon.

5.3.2.2 int isFieldvchar (char * *request,* int * *cursor*)

Fonction pour vérifier si on a bien un Field-VCHAR.

Paramètres

str	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.

Renvoie

Retourne 1 si la caractere de la requête est un field-vchar, 0 sinon.

5.3.2.3 int isObstext (char * *request,* int * *cursor*)

Fonction pour vérifier si on a bien un ObsText.

Paramètres

str	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.

Renvoie

Retourne 1 si le caractere de la requete est un ObsText, 0 sinon.

5.3.2.4 int isTchar (char * *request,* int * *cursor*)

Fonction pour vérifier si on a bien un TCHAR.

Paramètres

str	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.

Renvoie

Retourne 1 si le caractere de la requete est un TCHAR, 0 sinon.

5.3.2.5 int isUnreserved (char * *request,* int * *cursor*)

Fonction pour vérifier si on a bien un Unreserved.

Paramètres

str	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.

Renvoie

Retourne 1 si la caractere de la requête est un Unreserved, 0 sinon.

5.3.2.6 int isVCHAR (char * *request,* int * *cursor*)

Fonction pour vérifier si on a bien un VCHAR.

Paramètres

str	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.

Renvoie

Retourne 1 si le caractere de la requete est un VCHAR, 0 sinon.

5.3.2.7 int parse_absolute_form (char * *request,* int * *cursor,* HTTP_Node * *node*)

Fonction pour parser le absolute-form.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.8 int parse_authority_form (char * *request,* int * *cursor,* HTTP_Node * *node*)

Fonction pour parser le authority-form.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.9 int parse_field_content (char * *request*, int * *cursor*, HTTP_Node * *header_field*, HTTP_Node * *field_name*, HTTP_Node * *field_value*)

Fonction pour parser le field-content (utilisée par header-field).

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud header-field utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-name utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-value utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.10 int parse_field_name (char * *request*, int * *cursor*, HTTP_Node * *header_field*, HTTP_Node * *field_name*, HTTP_Node * *field_value*)

Fonction pour parser le field-name.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud header-field utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-name utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-value utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.11 int parse_field_value (char * *request*, int * *cursor*, HTTP_Node * *header_field*, HTTP_Node * *field_name*, HTTP_Node * *field_value*)

Fonction pour parser le field-content (utilisée par header-field).

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud header-field utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-name utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-value utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.12 int parse_header_field (char * *request,* int * *cursor,* HTTP_Node * *header_field*)

Fonction pour parser le header-field.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.13 int parse_HTTP_message (char * *request,* int * *cursor,* HTTP_Node * *node*)

Fonction pour parser le HTTP-message.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.14 int parse_IPv4 (char * *request,* int * *cursor,* HTTP_Node * *node*)

Fonction pour parser le IPv4.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.15 int parse_message_body (char * *request*, int * *cursor*, HTTP_Node * *node*)

Fonction pour parser le message-body.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.16 int parse_obs_fold (char * *request*, int * *cursor*, HTTP_Node * *header_field*, HTTP_Node * *field_name*, HTTP_Node * *field_value*)

Fonction pour parser le obs-fold (utilisée par header-field).

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud header-field utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-name utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.
node	Pointeur vers le noeud field-value utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.17 int parse_origin_form (char * *request*, int * *cursor*, HTTP_Node * *node*)

Fonction pour parser le origin-form.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.18 int parse_regname (char * *request*, int * *cursor*, HTTP_Node * *node*)

Fonction pour parser le regname.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.19 int parse_request_line (char * *request*, int * *cursor*, HTTP_Node * *node*)

Fonction pour parser le request-line.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.20 int parse_request_target (char * *request*, int * *cursor*, HTTP_Node * *node*)

Fonction pour parser le request-target.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.21 int parse_start_line (char * *request,* int * *cursor,* HTTP_Node * *node*)

Fonction pour parser le start-line.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.22 int parse_status_line (char * *request,* int * *cursor,* HTTP_Node * *node*)

Fonction pour parser le status-line.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.23 int parse_string (char * *str,* int * *cursor,* char * *cmp_str*)

Fonction pour vérifier l'égalité d'une chaîne et d'une sous-chaîne.

Paramètres

str	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
cmp_str	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.

5.3.2.24 int parse_user_info (char * *request*, int * *cursor*, HTTP_Node * *node*)

Fonction pour parser le user-info.

Paramètres

request	chaîne contenant la requête.
cursor	Pointeur vers la position de la lecture de la requête.
node	Pointeur vers le noeud utilisé pour sauvegarder la valeur de l'élément correspondant au noeud.

Renvoie

Retourne 1 si la requête est correcte, 0 sinon.