Министерство образования и науки Российской Федерации

Санкт-Петербургский Политехнический Университет Петра Великого

—

Институт кибербезопасности и защиты информации

ЛАБОРАТОРНАЯ РАБОТА № 13

«Защита web-сервера от несанкционированного доступа»

1. по дисциплине «Основы информационной безопасности»
2. Выполнил
3. студент гр. 4851004/10001 Веремейчук Я.Ю.

<подпись>

1. Преподаватель
2. асс. преподавателя Панков И.Д.

<подпись>

1. Санкт-Петербург
2. 2022
3. Цель работы

Приобретение практических навыков по созданию безопасной рабочей конфигурации web-ceрвepa Apache, исследование функций безопасности web-ceрвepa и механизмов контроля и управления доступом к страницам сайта.

1. Постановка задачи

Необходимо изучить работу web-сервера Apache, разработать программу подбора пароля, зашифрованного способом base64, разработать программу проверки пароля.

1. Теоретические исследования

Необходимо рассмотреть какие существуют подходы к решению задачи. Обычно подходов несколько. Необходимо проанализировать какой из подходов лучше подходит к решению задачи (по каким-либо выбранным вами критериям) и обосновать – почему выбран именно он.

Web-приложенпе клиент-серверное приложение, в котором клиентом выступает web-браузер, в котором исполняется активный код или отображается web-контент, а сервером выступает web-сервер, на котором хранится набор программ п данных, передаваемых клиенту. Логика web-приложения реализуется на стороне сервера, клиент лишь формирует запросы и получает ответы. В этой связи безопасность web-сервера становится определяющей для безопасности всей системы в целом и web-приложенпй в частности.

Пример наиболее распространенной серверной web-платформы – свободно распространяемый, полнофункциональный, расширяемый web-сервер Apache. М7еЬ-сервер Apache настраивается путем изменения текстовых конфигурационных файлов. Основные параметры web-ceрвepa уже заданы по умолчанию. В самом простом случае после установки web-сервер Apache сразу же может использоваться для обработки поступающих web-запросов со стороны клиентских web-приложений к HTML-сайтам. Для обеспечения безопасности сайтов и разграничения доступа web-приложений требуется задействовать функции безопасности web-сервера, например, механизм авторизации запросов.

В операционных системах семейства UNIX обычно файлы создаются с правами rv- --- ---, то есть доступны пользователю-владельцу, который создал п запустил web-ceрвep. Только суперпользователь root и пользователь, от имени которого запущен web-ceрвep, имеют права на чтение этих файлов. Такая система не является действительно безопасной, так как от имени этого же пользователя могут быть выполнены скрипты, доступные по протоколу НТТР.

Если на сервере в скрипте имеется какая-либо уязвимость, позволяющая получать содержимое файлов по запросам web-приложений, то в этом случае атакующий сможет узнать учетные записи любого пользователя. Защититься от этого можно, ограничивая доступ на уровне сайта. Для этого настройки сохраняются в файлах .htaccess и .htpasswd, при этом используются функциональные модули – mod\_auth и mod\_ассеss.

Если на одном сервере с установленной операционной системой семейства Unix и web-сервером Apache заведено несколько пользователей, то каждому из них можно создать отдельную директорию. Точнее, она будет создаваться автоматически вместе с созданием псевдонима пользователя. Это делается с помощью модуля mod\_userdir п директивы UserDir.

Метод basic-аутентификации в web-ceрвepax основывается на спецификации протокола НТТР. Аутентификация пользователей работает следующим образом: если пользователь не ввел пароль или пароль неверен, то сервер отвечает заголовком ”401 Unauthorized”. Если web-браузер получает от сервера такое сообщение, то браузер должен отобразить пользователю диалоговое окно с требованием ввести имя пользователя и пароль. Как только имя п пароль введены пользователем, web-браузер должен повторить запрос к серверу и направить серверу имя пользователя и пароль. web-сервер повторно проверяет имя пользователя и пароль п затем либо разрешает доступ, либо вновь отвечает сообщением "401 Unauthorized". В последнем случае можно полагать, что отправлены неверные сведения или имя. В течение всего сеанса браузер каждый раз передает имя и пароль для доступа ко всем документам, расположенным в том же web-каталоге или в других подкаталогах.

1. Описание решения

Работа была выполнена в ОС Xubuntu. Для начала необходимо выяснить IP-адрес. Для этого в консоли нужно выполнить команду «ifconfig» (Рисунок 1). Если такой команды нет, необходимо в консоли выполнить команду «sudo apt install net-tools».

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Поиск IP-адреса

Если зайти в браузер на данном этапе и ввести в поисковике найденный IP-адрес, браузер сообщит, что такой страницы не найдено.

Чтобы установить веб-сервер Apache, нужно выполнить команду «sudo apt install apache2». Результат, который был отражен в браузере, можно увидеть на рисунке 2.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Веб-страница, полученная при вводе IP-адреса в поисковую строку

Далее опытным путем было выяснено, что есть возможность вызвать команду «wget», так как она уже установлена (рисунок 3).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Команда "wget" уже была установлена

Далее необходимо получить исходные файлы веб-сервера путем их загрузки с официального сайта <https://dlcdn.apache.org/httpd/httpd-2.4.53.tar.gz> с помощью команды «wget». Но возникла ошибка поиска. Поэтому в поисковой строке был введен адрес <https://dlcdn.apache.org/httpd/> , который открыл список исходных файлов веб-сервера. Был скачен файл httpd-2.4.53.tar.gz (рисунок 4). Из папки «Загрузки» загруженный архив был перенесен в директорию, с которой происходит работа в консоли (рисунок 5).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Поиск необходимого файла

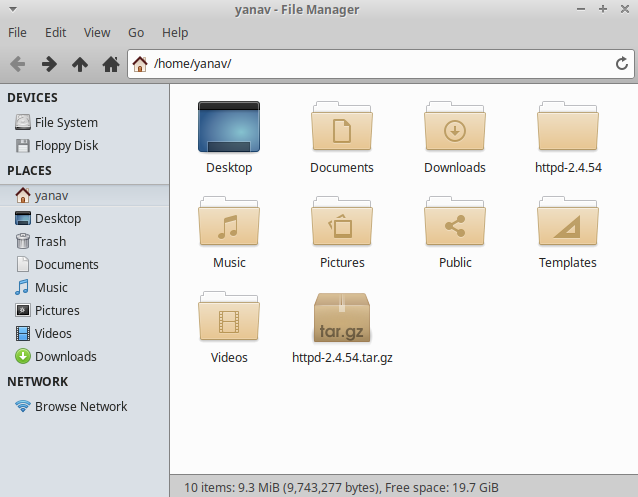


Рисунок – Сохранение архива в директории, с которой происходит работа в консоли

Далее при помощи команды «tar -zxvf httpd-2.4.53.tar.gz» архив был распакован. Затем был выполнен вход в распакованную директорию с помощью команды «cd httpd-2.4.53» (рисунок 6).



Рисунок – Вход в распакованную директорию

Затем была произведена неудачная попытка сформировать файл сборки с помощью команды «./configure» (рисунок 7).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Ошибка при попытке сборки

Консоль предупредила об отсутствии файла pcre(2)-config, необходимо скачать его. Для этого нужно перейти на сайт <http://pcre.org/> , скачать необходимый архив. Но данный способ не привел к успеху – консоль все так же сообщала об отсутствии файла. Было найдено другое решение: выполнить команду «sudo apt-get install libpcre3-dev» (рисунок 8). После этого были установлены необходимые файлы, и успешно была выполнена команда «./configure» (рисунок 9).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Успешное скачивание необходимых файлов

Изображение выглядит как текст

Автоматически созданное описание

Рисунок –Успешное завершение команды "./configure"

Далее была осуществлена сборка проекта с помощью команды «make».

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Как выглядит файл htpasswd в текстовом редакторе

Было создано 5 пользователей с непустыми паролями (рисунок 11). Чтобы просмотреть зашифрованные пароли пользователей, необходимо выполнить команду «cat /etc/apache2/.htpasswd» (рисунок 12).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Создание новых пользователей и присваивание им паролей

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Зашифрованные пароли пользователей

Далее был осуществлен переход в директорию /var/www/html/ и созданы папки secret, not\_secret и some\_secret (рисунок 13).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Создание каталогов

Затем была установлена команда «gedit» и открыт файл /etc/apache2/apache2.conf, чтобы включить поддержку файлов .htaccess.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Файл ДО

Запись в файле была изменена (рисунок 15).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Файл ПОСЛЕ

Затем с помощью команды «sudo systemctl restart apache2.service» веб-сервер был перезапущен. Затем был создан файл .htaccess с помощью команды «sudo gedit /var/www/html/secret/.htaccess». В него необходимо записать текст, представленный на рисунке 16.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Файл .htaccess

Далее в веб-браузере был открыт адрес <http://localhost/secret/>, где была найдена форма для ввода имени пользователя и пароля (рисунок 17).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Форма ввода логина и пароля

Теперь, чтобы разрешить лишь отдельным пользователям заходить в определенные каталоги, в файл из (рисунка наверху) необходимо добавить соответствующую строку. К каталогу secret имеет доступ лишь user1 (рисунок 18), к some\_secret – user1, user3 и user5 (рисунок 19), к каталогу not\_secret имеют доступ все пользователи (рисунок 20). В таблице 1 можно увидеть матрицу доступа.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Прописывание в файле .htaccess пользователей, имеющих доступ к папке secret

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Прописывание в файле .htaccess пользователей, имеющих доступ к папке some\_secret

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Прописывание в файле .htaccess пользователей, имеющих доступ к папке not\_secret

Таблица – Матрица доступа

| Имя пользователя | secret | some\_secret | not\_secret |
| --- | --- | --- | --- |
| User1 | Доступ разрешён | Доступ разрешён | Доступ разрешён |
| User2 | Доступ запрещён | Доступ запрещён | Доступ разрешён |
| User3 | Доступ запрещён | Доступ разрешён | Доступ разрешён |
| User4 | Доступ запрещён | Доступ запрещён | Доступ разрешён |
| User5 | Доступ запрещён | Доступ разрешён | Доступ разрешён |

После этого была произведена попытка попасть на сайт пользователю, который не имеет разрешения просматривать каталог. Попытка завершилась неудачей. Затем был создан файл .htgroup (Рисунок 21).

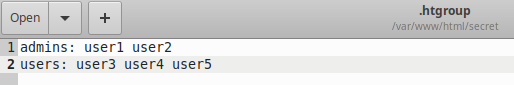


Рисунок – Создание групп пользователей

В файле .htaccess был прописан текст, который можно увидеть на рисунке 22.

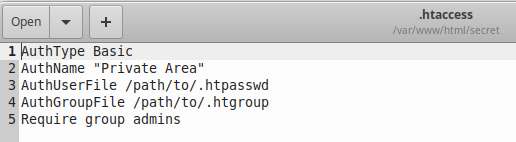


Рисунок – Прописывание группы пользователей, которой будет доступна папка secret

После предоставления группового доступа было установлено, что администраторы имели доступ к сайту, а другие пользователи – нет. Результаты проверки можно увидеть в таблице 2.

Таблица – Матрица доступа к папке secret

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | User1 | User2 | User3 | User4 | User5 |
| Доступ к папке secret | Доступ разрешён | Доступ разрешён | Доступ запрещён | Доступ запрещён | Доступ запрещён |

1. Тестирование и результаты работы программы

На языке программирования C были написаны две программы: подбора пароля и проверки пароля.

Алгоритм проверки пароля был встроен в утилиту htpasswd. Для этого был открыт файл passwd\_common.c, в котором прописан код для взаимодействия разработчика с консолью: ввод пароля, ошибки (рисунок 23).

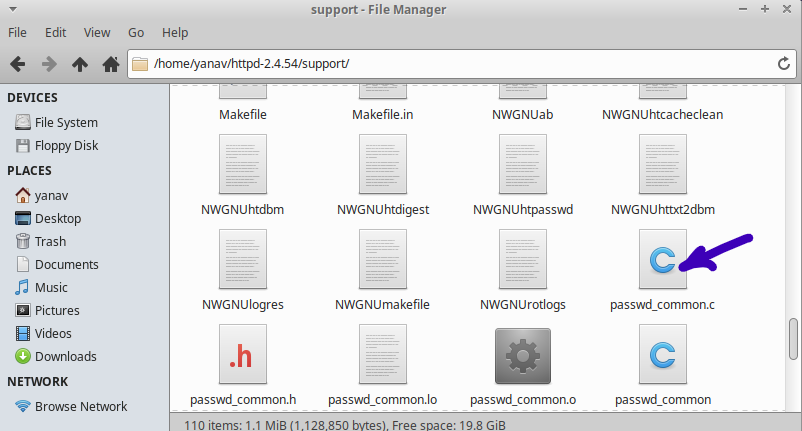


Рисунок – Поиск исходного кода для ввода пароля

Далее было необходимо найти в файле код, который записывает введенный в консоли пароля в переменную (рисунок 24). После данного кода и была вставлена проверка: в файл была записана реализация функции проверки и эта функция была написана после чтения пароля (рисунок 25). После этого утилиту необходимо пересобрать (рисунок 26). Затем в каталоге утилиты необходимо найти файл htpasswd (рисунок 27). Его необходимо скопировать в системный каталог bin. Вручную это сделать не удалось, поэтому копирование было произведено с помощью консоли (рисунок 28). Также были рассмотрены свойства скопированного файла на вопрос того, когда он был изменен, чтобы удостовериться в том, что копирование произведено успешно (рисунок 29).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Поиск кода, отвечающего за ввод пароля

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Вставка проверки пароля в исходный код



Рисунок – Сборка утилиты

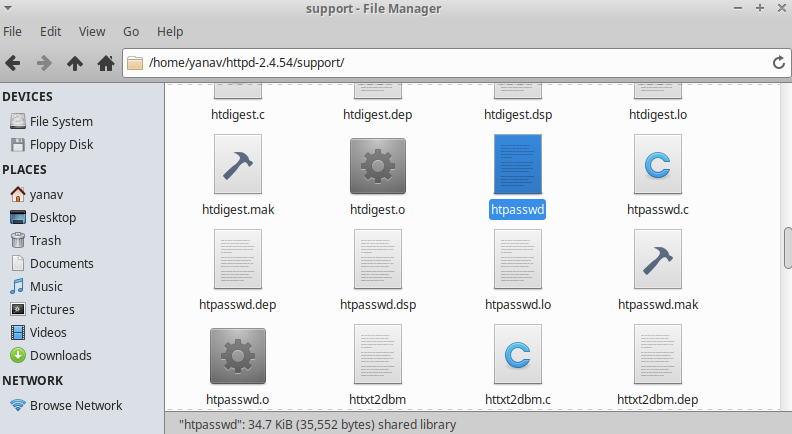


Рисунок – Поиск файла htpasswd



Рисунок – Перемещение файла htpasswd в каталог bin

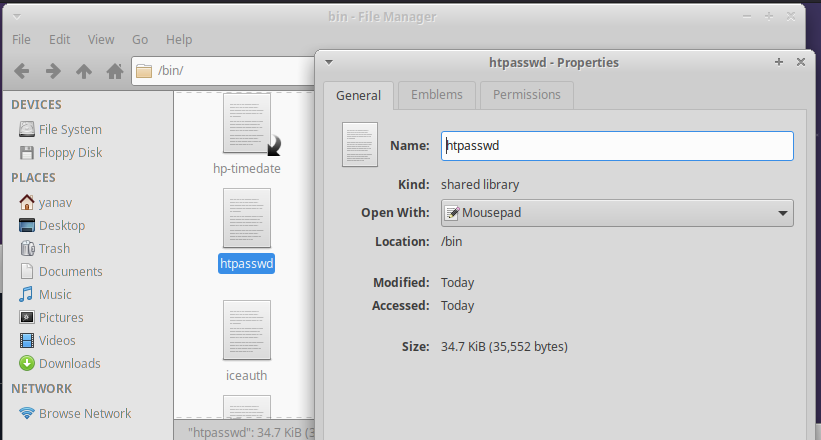


Рисунок – Проверка того, что файл в bin был заменен

Протестировав улучшенную утилиту, можно сделать вывод, что она работает правильно: при некорректном пароле выводится ошибка (рисунок 30), при корректном пароле выводится сообщение об успешной записи пароля (рисунок 31).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Попытка ввода некорректного пароля

Изображение выглядит как текст

Автоматически созданное описание

Рисунок – Успешный ввод корректного пароля

1. Выводы

Были приобретены практические навыки по созданию безопасной рабочей конфигурации web-ceрвepa Apache, исследованы функций безопасности web-ceрвepa и механизмов контроля и управления доступом к страницам сайта. Также была изучена работа web-сервера Apache, разработана программа подбора пароля, зашифрованного способом base64, разработана программа проверки пароля.

Список используемых источников

1. Зегжда П.Д. Основы информационной безопасности. Введение в профессиональную деятельность. Лабораторный практикум : учеб. пособие / П. Д. Зегжда, М. О. Калинин. – СПб. : ПОЛИТЕХ-ПРЕСС, 2019. – 112 с.
2. Apache installation; libpcre error [Электронный ресурс] – Режим доступа: <https://stackoverflow.com/questions/10663180/apache-installation-libpcre-error> , свободный.
3. Как настроить авторизацию по паролю на Apache с Ubuntu 16.04 [Электронный ресурс] – Режим доступа: <https://timeweb.com/ru/community/articles/kak-nastroit-avtorizaciyu-po-parolyu-na-apache-s-ubuntu-16-04-1> , свободный.
4. Авторизация через .htaccess [Электронный ресурс] – Режим доступа: https://snipp.ru/htaccess/basic-auth-htaccess , свободный.

ПРИЛОЖЕНИЕ А

Листинг программы «Подбор пароля»

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define PAS\_SIZE 4

#define CODE\_SIZE 6

#define BYTE 8

void decode\_base64(char\*, char\*);

int main()

{

char encode[CODE\_SIZE+1];

char password[PAS\_SIZE + 1] = { 0 };

decode\_base64(encode, password);

return 0;

}

void decode\_base64(char\* encode, char\* password)

{

printf("Enter code: ");

(void)scanf("%6s", encode); // 6 == CODE\_SIZE

int count = 0;

while (encode[count] != '\0') count++;

printf("Code size = %d\n", count);

int size\_bit\_code = count \* BYTE + 1;

char\* bit\_code = (char\*)calloc(size\_bit\_code, sizeof(char));

if (bit\_code == NULL)

{

printf("No memory in bit\_code!");

exit(-1);

}

//перевод символов в битовое представление и их запись в массив

char tmp;

int tmp\_count;

int bit\_code\_iterator = 0;

for (int i = 0; i < count; i++)

{

//составление соответствия таблицы ascii с таблицей base64

tmp = encode[i];

if (tmp >= 'A' && tmp <= 'Z') tmp -= 65; //65==код 'A' в ASCII

else if (tmp >= 'a' && tmp <= 'z') tmp = tmp - 97 + 26; //97==код 'a' в ASCII, 26==код 'a' в base64

else if (tmp >= '0' && tmp <= '9') tmp = tmp - 48 + 52; //48==код '0' в ASCII, 52==код '0' в base64

else if (tmp == '\_') tmp = 62; //62==код '\_' в base64

else

{

printf("Encorrect value!\n");

exit(-1);

}

tmp\_count = 1;

if (bit\_code)

{

while (tmp\_count <= 6 && bit\_code\_iterator < size\_bit\_code)

{

bit\_code[bit\_code\_iterator] = tmp & 0x20 ? '1' : '0'; //0x20 это 6ой разряд

bit\_code\_iterator++;

tmp\_count++;

tmp = tmp << 1;

}

}

}

if(bit\_code)bit\_code[bit\_code\_iterator] = '\0';

printf("bit\_code = %s\n", bit\_code);

//перевод массива в исходные символы по 8 бит

int pass\_iterator = -1;

int digit;

for (int i = 0; i < bit\_code\_iterator;i++)

{

if (!(i % BYTE))

{

pass\_iterator++;

password[pass\_iterator] = 0;

digit = 0x80; // 0x80 это 8ой разряд

}

password[pass\_iterator] += digit \* (bit\_code[i] - '0');

digit >>= 1;

}

printf("password = %s\n", password);

free(bit\_code);

}

ПРИЛОЖЕНИЕ B

Листинг программы «Проверка пароля»

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_SIZE 127

typedef enum ERROR

{

NOT\_SIZE = 0b00000001,

NO\_S\_REG = 0b00000010,

NO\_B\_REG = 0b00000100,

NO\_SYM = 0b00001000,

NO\_DIGIT = 0b00010000

}ERROR;

char\* enter\_pass(char\*);

int check\_password(char\*);

int print\_errors(int errors);

int main()

{

while (1)

{

int errors = 0;

char\* pass = NULL;

pass = enter\_pass(pass);

printf("%s\n", pass);

errors = check\_password(pass);

free(pass);

if (!print\_errors(errors))

{

printf("Password is correct!");

return 0;

}

}

return 0;

}

char\* enter\_pass(char\*pass)

{

pass = (char\*)calloc(MAX\_SIZE + 1, sizeof(char));

printf("Enter password: ");

(void)scanf("%s", pass);

return pass;

}

int check\_password(char\*pass)

{

int errors = 0;

//размер

int count = strlen(pass);

if (count < 10) errors |= NOT\_SIZE;

//регистры и символы

int flag = 0;

for (int i = 0; i < count; i++)

{

if (pass[i] >= 'a' && pass[i] <= 'z') flag = 1;

}

if (flag == 0) errors |= NO\_S\_REG;

flag = 0;

for (int i = 0; i < count; i++)

{

if (pass[i] >= 'A' && pass[i] <= 'Z') flag = 1;

}

if (flag == 0) errors |= NO\_B\_REG;

flag = 0;

for (int i = 0; i < count; i++)

{

if ((pass[i] >= 32 && pass[i] <= 47)||(pass[i] >= 58 && pass[i] <= 64)||(pass[i] >= 91 && pass[i] <= 96) || (pass[i] >= 123 && pass[i] <= 127)) flag = 1;

}

if (flag == 0) errors |= NO\_SYM;

flag = 0;

for (int i = 0; i < count; i++)

{

if (pass[i] >= '0' && pass[i] <= '9') flag = 1;

}

if (flag == 0) errors |= NO\_DIGIT;

flag = 0;

return errors;

}

int print\_errors(int errors)

{

if (errors & NOT\_SIZE) printf("Error: size of password < 10!\n");

if (errors & NO\_S\_REG) printf("Error: no symbols in small register!\n");

if (errors & NO\_B\_REG) printf("Error: no symbols in big register!\n");

if (errors & NO\_SYM) printf("Error: no special symbols!\n");

if (errors & NO\_DIGIT) printf("Error: no digits!\n");

if (errors != 0)

{

printf("Enter new password!\n\n");

}

else return 0;

return 1;

}