Министерство образования и науки Российской Федерации Санкт-Петербургский Политехнический Университет Петра Великого

[—](https://dl.spbstu.ru/)

[Институт компьютерных наук и кибербезопасности](https://dl.spbstu.ru/)

ЛАБОРАТОРНАЯ РАБОТА № 13

«Защита WEB-сервера от несанкционированного доступа» по дисциплине «Основы информационной безопасности»

Выполнили

студенты гр. 5151001/40001 Кириллов Д.А.

<*подпись*>

Преподаватель

ст. преподавателя Орел Е. М.

<*подпись*>

Санкт-Петербург 2025

## Цель работы

Приобретение практических навыков по созданию безопасной рабочей конфигурации web-сервера и механизмов контроля и управления доступом к страницам сайта.

## Постановка задачи

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

## Теоретические исследования

Web-приложение — клиент-серверное приложение, в котором клиентом выступает web-браузер, в котором исполняется активный код или отображается web-контент, сервером выступает web-сервер, на котором хранится набор программ и 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-приложений в частности.

Пример наиболее распространенной серверной web-платформы — свободно распространяемый, полнофункциональный сервер Apache. Web-сервер Apache настраивается путем изменения текстовых конфигурационных файлов. Основные параметры web-сервера уже заданы по умолчанию. В самом простом случае после установки web-сервер Apache сразу может использоваться для обработки поступающих web-запросов со стороны клиентских web-приложений к HTML-сайтам. Для обеспечения безопасности сайтов иразграничения доступа.

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 и 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.

## Ход работы

Работа была выполнена в ОС Ubuntu.

На виртуальную машину был установлен web – сервер Apache, первоначально файлы .htaccess и .htpasswd не были созданы изначально, поэтому они создавались вручную.

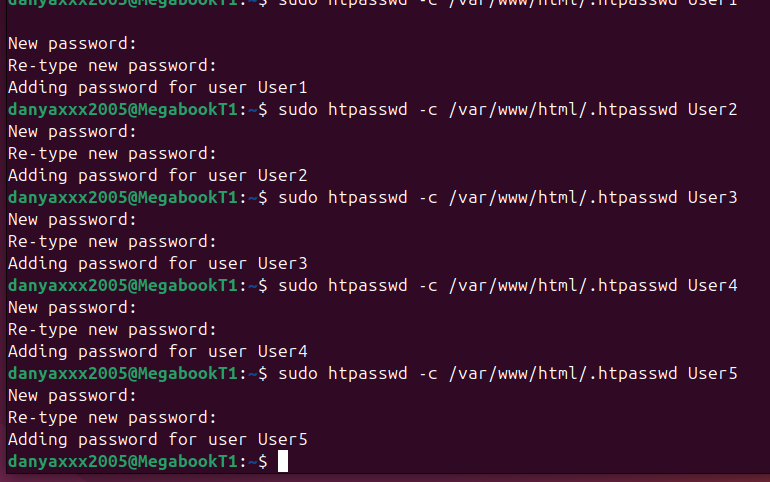


Рисунок 2 – Создание новых пользователей и присваивание им паролей

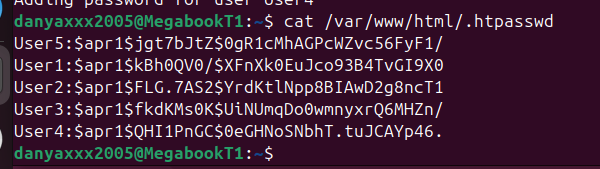


Рисунок 3 – Содержимое файла .htpasswd

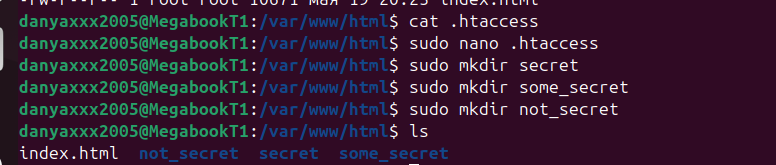


Рисунок 4 – Создание папок secret, not\_secret и some\_secret

Затем был открыт файл /etc/apache2/apache2.conf, чтобы включить поддержку файлов .htaccess.

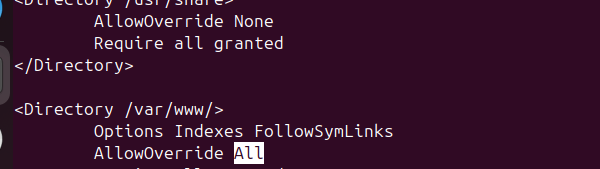


Рисунок 5 – Включения защиты паролем с помощью файла .htaccess (смена None на All)

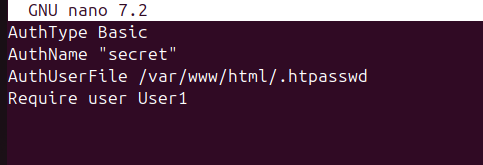


Рисунок 6 – Прописывание в файле .htaccess пользователей, имеющих доступ к папке secret

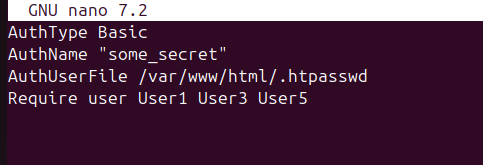


Рисунок 7 – Прописывание в файле .htaccess пользователей, имеющих доступ к папке some\_secret

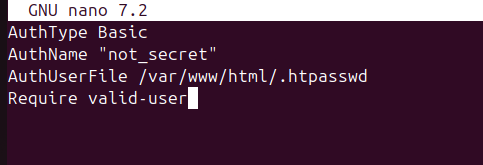


Рисунок 8 – Прописывание в файле .htaccess пользователей, имеющих доступ к папке not\_secret

Матрица доступа

|  |  |  |  |
| --- | --- | --- | --- |
| Пользователи | secret | Some\_secret | Not\_secret |
| User1 | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (12).png  Доступ разрешен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (13).png  Доступ разрешен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (14).png  Доступ разрешен |
| User2 | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (16).png | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (16).png | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (14).png  Доступ разрешен |
| User3 | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (16).png  Доступ запрещен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (18).png  Доступ разрешен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (14).png  Доступ разрешен |
| User4 | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (16).png  Доступ запрещен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (16).png  Доступ запрещен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (14).png  Доступ разрешен |
| User5 | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (16).png  Доступ запрещен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (18).png  Доступ разрешен | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (14).png  Доступ разрешен |

Затем был создан файл .htgroup.

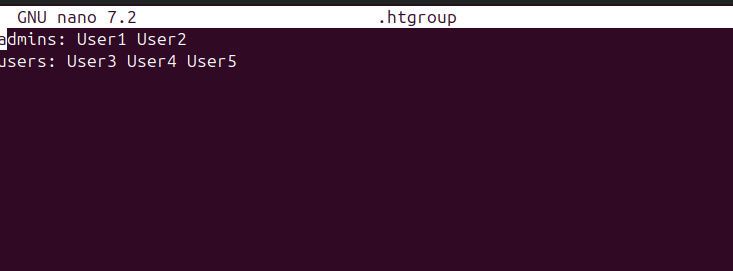


Рисунок 9 – Создание групп пользователей

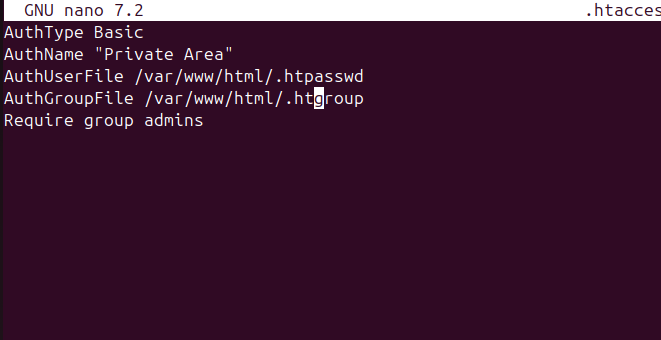


Рисунок 10 – Прописывание группы пользователей, которой будет доступна папка secret

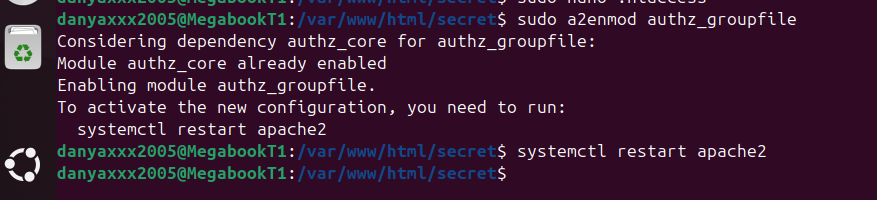


Рисунок 11 – Включение модуля для работы с группами пользователей и перезапуск сервера.

Матрица доступа к папке secret

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | User1 | User2 | User3 | User4 | User5 |
| Доступ к папке secret | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (28).png | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (28).png | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (30).png | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (30).png | C:\Users\Даниил\AppData\Roaming\Telegram Desktop\tdata\temp_data\изображение (30).png |

После была разработана программа по добавлению нового пользователя с паролем. Она должна проверять веденный пароль на:

Длину (не менее 10 символов)

Наличие заглавных и строчных букв

Наличие цифр

Наличие специальных символов

В случае, если все условия выполнены, программа создаёт нового пользователя. (Приложение1)

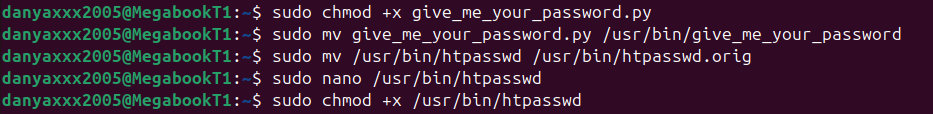


Рисунок 12. Встраивание программы

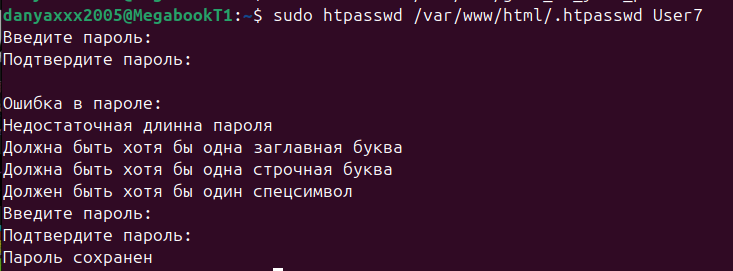


Рисунок 13 — Результат работы программы проверки пароля

Также была разработана программа для подбора пароля пользователя,

зашифрованного способом base64. (Приложение 2) Требования к паролю следующие:

* Длина пароля не более четырех символов.
* Разрешены буквы английского алфавита (верхнего и нижнего регистра), цифры и знак подчеркивания (\_).

Алгоритм шифрования base64:

* Каждому символу пароля сопоставляется его код в таблице ASCII;
* Эти коды представляются в двоичном виде (каждому символу соответствуют 8 бит) и «склеиваются» в одну строку;
* При необходимости добавляется дополнительные нули;
* Полученная строка разделяется на группы по 6 бит и каждой группе ставится в соответствие символ из таблицы кодов Base64.
* Если длина полученной строки не кратно 4, то в конец добавляются ‘=’

## Выводы

В рамках лабораторной работы были освоены основные принципы работы с базовыми алгоритмами web-сервера Apache, изучены методы создания безопасной конфигурации сервера и рассмотрены функции обеспечения безопасности. Проведен анализ механизмов контроля и управления доступом к веб-страницам сайта. Кроме того, исследован алгоритм base64, получены начальные навыки его декодирования и применения на практике.

## Ответы на контрольные вопросы

1. *Какую роль web-сервер выполняет при работе с web-приложениями?*

При работе с web-приложениями web-сервер выполняет функции: хранение набор программ и данных, которые передаются пользователям, осуществление вычислений, предоставление ресурса.

1. *Перечислите отличия web-приложения от обычного приложения.*

Web-приложения кроссплатформенны, независимы от программной и аппаратной среды, могут работать с большим количеством различных клиентов. Логика web-приложения реализуется на стороне сервера. Требуют наличие подключения к Интернету и web-браузера.

1. *Какие конфигурационне файлы используются при работе механизма контроля доступа к web-страницам сайта, размещенного на web-сервере Apache?*

Есть три файла: .htaccess, .htpasswd, .htgroup. Первый файл отвечает за конфигурацию доступа к папке отдельным пользователям(группам). .htpasswd хранит в себе все пароли для каждого пользователя. .htgroup хранит в себе группы с распределенными пользователями.

1. *Опишите процесс basic-футентификации со стороны клиента.*

Процесс basic-аутентификации со стороны клиента: При пропуске ввода пароля или его некорректного ввода сервер отвечает заголовком “401 Unauthorized”. В том случае, когда web-браузер получает от сервера данное сообщение, то он должен будет отобразить пользователю диалоговое окно с требованием ввести имя пользователя и пароль. Затем клиент повторяет ввод. Далее web-браузер должен повторить запрос к серверу и направить серверу имя пользователя и пароль. Web-сервер повторно проверяет имя пользователя и пароль. При корректном вводе разрешает доступ, в противном случае вновь отвечает сообщением “401 Unauthorized”.

1. *Что позволяет нарушить безопасность web-сервера Apache при basic- аутентификации? Предложите и обоснуйте методы защиты от нарушений безопасности web-сервера Apache при basic-аутентификации.* Безопасность сервера позволяет нарушить хранение паролей в файле .htpasswd в открытом виде. Ситуация ухудшается в том случае, когда пароль состоит из малого количества символов, не содержит специальных символов, цифр и букв различного регистра. Избежать данного нарушения можно при помощи предварительной проверки команд, с которыми запускается утилита htpasswd.с

Приложение 1

#!/usr/bin/env python3

import re

import sys

from getpass import getpass

def find\_errors(password):

    errors = []

    if len(password) < 10:

        errors.append("Недостаточная длинна пароля")

    if not re.search(r'[A-Z]', password):

        errors.append("Должна быть хотя бы одна заглавная буква")

    if not re.search(r'[a-z]', password):

        errors.append("Должна быть хотя бы одна строчная буква")

    if not re.search(r'[0-9]', password):

        errors.append("Должна быть хотя бы одна цифра")

    if not re.search(r'[^A-Za-z0-9]', password):

        errors.append("Должен быть хотя бы один спецсимвол")

    return errors

def main():

    max\_try = 5

    Try = 0

    while Try < max\_try:

        password = getpass("Введите пароль: ")

        re\_password = getpass("Подтвердите пароль: ")

        if password != re\_password:

            print("Пароли не совпадают")

            Try += 1

            continue

        errors = find\_errors(password)

        if not errors:

            print("Пароль сохранен")

            sys.exit(0)

        print("\nОшибка в пароле: ")

        for error in errors:

            print(f"{error}  ")

        Try += 1

    print("Превышено максимальное количество попыток")

    sys.exit(1)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Приложение 2

import itertools

CHARS = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789\_"

def base64(string):

    alf = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"

    bytes\_string = string.encode('ascii')

    bin\_string = ''.join(f"{byte:08b}" for byte in bytes\_string)

    while(len(bin\_string)%6!=0):

        bin\_string += '0'

    encoded = []

    for i in range(0,len(bin\_string),6):

        encoded.append(alf[int(bin\_string[i:i+6],2)])

    encoded = "".join(encoded)

    while len(encoded) % 4 != 0:

        encoded += '='

    return encoded

def brutforce(en\_string):

    for len in range(1,5):

        for x in itertools.product(CHARS, repeat=len):

            password = ''.join(x)

            encoded = base64(password)

            if encoded == en\_string:

                return password

    return None

your\_password = input("Введите пароль ")

result = brutforce(your\_password)

if result:

    print(f"Найден пароль: {result}")

else:

    print("Пароль не найден.")