Задание:

1. Установить Nginx и настроить его на работу с PHP-FPM.

Устанавливаем компоненты:



Разрешаем автозапуск php-fpm и запускаем его:



Проверяем открытые порты: 

Открываем порты для TCP и FTP:





Для сохранения правил ставим пакет iptables-persistent:

Сохраняем правила: 

Проверяем работу nginx: 

Проверим работу веб-сервера: 

Настройка связки NGINX + PHP.

Открываем файл для настройки виртуального домена по умолчанию:



В секции  **server** редактируем параметр **index** на значение:

index index.php index.html index.htm;

Внутри секции **server** добавляем:

location ~ \.php$ {

*# корневой путь хранения скриптов*   
    set $root\_path /var/www/html;

*# путь до файла сокета для взаимодействия с php-fpm*

fastcgi\_pass unix:/run/php/php8.1-fpm.sock;

    fastcgi\_index index.php;  
    fastcgi\_param SCRIPT\_FILENAME $root\_path$fastcgi\_script\_name;  
    include fastcgi\_params;  
    fastcgi\_param DOCUMENT\_ROOT $root\_path;  
}

проверяем: , перезагружаем 

Открываем конфигурационный файл PHP-FPM: 

Проверяем, что путь до файла сокета такой же, как мы задали в настройках NGINX:

listen = /run/php/php8.1-fpm.sock

заходим в каталог хранения настроенного сайта: 

Создаем index.php 

Открываем его для редактирования: 

Вносим: <?php phpinfo(); ?>

Для проверки переходим в браузере по IP сервера: <http://10.0.1.12/>, видим сводную информацию по PHP – Ок.

1. Установить Apache. Настроить обработку PHP. Добиться одновременной работы с Nginx.

Устанавливаем Apache: 

Устанавливаем модуль php для apache

настраиваем порты: 

Listen 8080 # порт 80 занят nginx  
  
#<IfModule ssl\_module>  
#       Listen 443 #порт слушает nginx  
#</IfModule>  
#<IfModule mod\_gnutls.c>  
#       Listen 443  
#</IfModule>

Настраиваем обработку сначала для php в модуле м

<IfModule dir\_module>  
    DirectoryIndex index.php index.html ...  
</IfModule>

Настраиваем основной файл конфигурации Apache:

<Directory /var/www/\*/www> # *указывает на путь, для которого мы хотим задать настройки*  
    AllowOverride All # *позволяет переопределить все настройки с помощью файла .htaccess*

*#* **Options***задает некоторые настройки:***Indexes***разрешает списки каталогов,***ExecCGI***разрешает запуск cgi скриптов*  
    Options Indexes ExecCGI FollowSymLinks  
    Require all granted # *предоставляет всем доступ к сайтам в данном каталоге*  
</Directory>

. . . #в конце добавляем:

<IfModule setenvif\_module>  
    SetEnvIf X-Forwarded-Proto https HTTPS=on  
</IfModule>

#этой настройкой мы при получении заголовка **X-Forwarded-Proto** со значением **https** задаем #переменную **$\_SERVER['HTTPS']** равную **on**. Данная настройки критична для функционирования некоторых CMS.

Проверяем конфиг:  - Ok.

Запрещаем mpm\_event: 

(по умолчанию, apache2 может быть установлен с модулем мультипроцессовой обработки **mpm\_event**. Данный модуль не поддерживает php 7 и выше)

Разрешаем модуль мультипроцессовой обработки mpm\_prefork:

Разрешаем модуль php: 

Разрешаем модуль setenvif: 

Разрешаем модуль rewrite: 

Разрешаем автозапуск Apache: 

Перезапускаем Apache: 

Для проверки переходим в браузере по IP сервера: http://10.0.1.12:8080/, видим сводную информацию по PHP, с помощью инструментов разработчика F12 убеждаемся, что сервер – Apache.

1. Настроить схему обратного прокси для Nginx (динамика - на Apache).

Меняем конфигурацию nginx:

в секции location комментируем index.php,

в секции location ~ \.php$ комментируем настройки для работы через php-fpm, добавляем:

        location ~ \.php$ {  
            proxy\_pass http://127.0.0.1:8080;  
            proxy\_redirect off;  
            proxy\_set\_header   Host $host;  
            proxy\_set\_header   X-Forwarded-Proto $scheme;  
            proxy\_set\_header   X-Real-IP        $remote\_addr;  
            proxy\_set\_header   X-Forwarded-For  $proxy\_add\_x\_forwarded\_for;  
        }

проверяем конфиг: sudo nginx –t

перезапускаем nginx: 

Для проверки переходим в браузере по IP сервера: http://10.0.1.12

1. Установить MySQL. Создать новую базу данных и таблицу в ней.

Устанавливаем бесплатный форк MySQL:

Включаем автозапуск 

Устанавливаем пароль root 

Запускаем систему под рутом 

Создаем новую базу**:**  **CREATE DATABASE test\_data\_base DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8\_general\_ci;**

Устанавливаем привилегии: **GRANT ALL PRIVILEGES ON test\_data\_base.\* TO root@localhost IDENTIFIED BY 'test' WITH GRANT OPTION**;

1. \* Установить пакет phpmyadmin и запустить его веб-интерфейс для управления MySQL



Создаем новый файл конфигурации для nginx:

server {

listen 80;

server\_name phpmyadmin.local;

set $root\_path /usr/share/phpmyadmin;

root $root\_path;

location / {

index index.php;

}

location ~ \.php$ {

fastcgi\_pass unix:/run/php/php8.1-fpm.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $root\_path$fastcgi\_script\_name;

include fastcgi\_params;

fastcgi\_param DOCUMENT\_ROOT $root\_path;

}

}

Прописываем в hosts: 127.0.1.1 phpmyadmin.local

Для проверки переходим в браузере по IP сервера: <http://10.0.1.12>

Видим веб-интерфейс phpmyadmin – Ok.

1. \* Настроить схему балансировки трафика между несколькими серверами Apache на стороне Nginx с помощью модуля ngx\_http\_upstream\_module.

Для балансировки нескольких серверов нужно прописать в файл конфигурации nginx все сервера с указанием их портов и веса, то есть приоритета обращений, вида:

upstream backend {

server 127.0.0.1:8080 weight=2;

server 127.0.0.1:8081;