nginx (Русский)

**[Tango-preferences-desktop-locale.png](https://wiki.archlinux.org/index.php/File:Tango-preferences-desktop-locale.png)Эта страница нуждается в сопроводителе[Tango-preferences-desktop-locale.png](https://wiki.archlinux.org/index.php/File:Tango-preferences-desktop-locale.png)**

Статья не гарантирует актуальность информации. Помогите русскоязычному сообществу поддержкой подобных страниц. См. [**Команда переводчиков ArchWiki**](https://wiki.archlinux.org/index.php/%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%B0_%D0%BF%D0%B5%D1%80%D0%B5%D0%B2%D0%BE%D0%B4%D1%87%D0%B8%D0%BA%D0%BE%D0%B2_ArchWiki)

**Состояние перевода:**На этой странице представлен перевод статьи [**nginx**](https://wiki.archlinux.org/index.php/Nginx). Дата последней синхронизации: 2015-07-29. Вы можете [**помочь**](https://wiki.archlinux.org/index.php/ArchWiki_Translation_Team_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)) синхронизировать перевод, если в английской версии произошли [**изменения**](https://wiki.archlinux.org/index.php?title=Nginx&diff=0&oldid=388788).

[**nginx**](https://en.wikipedia.org/wiki/ru:nginx) (произносится "э́нжин-э́кс" или "э́нжин-и́кс") — это свободный высокопроизводительный HTTP-сервер с открытым исходным кодом, а также обратный прокси и IMAP/POP3 прокси-сервер, написанный Игорем Сысоевым в 2005 году. Согласно [**April 2015 Web Server Survey**](http://news.netcraft.com/archives/2015/04/20/april-2015-web-server-survey.html), nginx используется на 14,48% доменов всего мира, в то время как [**Apache**](https://wiki.archlinux.org/index.php/Apache_HTTP_Server_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)) используется примерно на 38,39% доменов. nginx получил широкое распространение благодаря своей стабильности, богатой функциональности, простой настройке и низкому потреблению ресурсов.

**Contents**

 [hide]

* [1Установка](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BA%D0%B0)
* [2Запуск](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%97%D0%B0%D0%BF%D1%83%D1%81%D0%BA)
* [3Настройка](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9D%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0)
  + [3.1Основные настройки](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%8B%D0%B5_%D0%BD%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B8)
    - [3.1.1Процессы и соединения](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D1%8B_%D0%B8_%D1%81%D0%BE%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F)
    - [3.1.2Запуск под другим пользователем](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%97%D0%B0%D0%BF%D1%83%D1%81%D0%BA_%D0%BF%D0%BE%D0%B4_%D0%B4%D1%80%D1%83%D0%B3%D0%B8%D0%BC_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D0%B5%D0%BC)
    - [3.1.3Блоки server](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%91%D0%BB%D0%BE%D0%BA%D0%B8_server)
    - [3.1.4TLS/SSL](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#TLS/SSL)
  + [3.2FastCGI](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#FastCGI)
    - [3.2.1Реализация PHP](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A0%D0%B5%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_PHP)
      * [3.2.1.1Настройка PHP](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9D%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0_PHP)
        + [3.2.1.1.1MariaDB](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#MariaDB)
      * [3.2.1.2Настройка nginx](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9D%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0_nginx)
        + [3.2.1.2.1Добавление к основной конфигурации](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%94%D0%BE%D0%B1%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BA_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D0%BE%D0%B9_%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B8)
        + [3.2.1.2.2Управление несколькими блоками (опционально)](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BD%D0%B5%D1%81%D0%BA%D0%BE%D0%BB%D1%8C%D0%BA%D0%B8%D0%BC%D0%B8_%D0%B1%D0%BB%D0%BE%D0%BA%D0%B0%D0%BC%D0%B8_(%D0%BE%D0%BF%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE))
      * [3.2.1.3Проверка конфигурации](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9F%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%BA%D0%B0_%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B8)
    - [3.2.2Реализация CGI](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A0%D0%B5%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_CGI)
      * [3.2.2.1fcgiwrap](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#fcgiwrap)
        + [3.2.2.1.1Несколько рабочих потоков](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9D%D0%B5%D1%81%D0%BA%D0%BE%D0%BB%D1%8C%D0%BA%D0%BE_%D1%80%D0%B0%D0%B1%D0%BE%D1%87%D0%B8%D1%85_%D0%BF%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%B2)
      * [3.2.2.2Настройка nginx](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9D%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0_nginx_2)
* [4Установка в chroot](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BA%D0%B0_%D0%B2_chroot)
  + [4.1Создание необходимых устройств](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BD%D0%B5%D0%BE%D0%B1%D1%85%D0%BE%D0%B4%D0%B8%D0%BC%D1%8B%D1%85_%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2)
  + [4.2Создание необходимых каталогов](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BD%D0%B5%D0%BE%D0%B1%D1%85%D0%BE%D0%B4%D0%B8%D0%BC%D1%8B%D1%85_%D0%BA%D0%B0%D1%82%D0%B0%D0%BB%D0%BE%D0%B3%D0%BE%D0%B2)
  + [4.3Заполнение chroot](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%97%D0%B0%D0%BF%D0%BE%D0%BB%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_chroot)
  + [4.4Отредактируйте nginx.service для запуска chroot](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9E%D1%82%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%B8%D1%80%D1%83%D0%B9%D1%82%D0%B5_nginx.service_%D0%B4%D0%BB%D1%8F_%D0%B7%D0%B0%D0%BF%D1%83%D1%81%D0%BA%D0%B0_chroot)
* [5Решение проблем](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A0%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B1%D0%BB%D0%B5%D0%BC)
  + [5.1Валидация конфигурации](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%92%D0%B0%D0%BB%D0%B8%D0%B4%D0%B0%D1%86%D0%B8%D1%8F_%D0%BA%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%B8)
  + [5.2При доступе с локального IP перенаправляется на localhost](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9F%D1%80%D0%B8_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%B5_%D1%81_%D0%BB%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B3%D0%BE_IP_%D0%BF%D0%B5%D1%80%D0%B5%D0%BD%D0%B0%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D1%82%D1%81%D1%8F_%D0%BD%D0%B0_localhost)
  + [5.3Ошибка: Страница, которую вы ищите, временно недоступна. Пожалуйста, попробуйте позже. (502 Bad Gateway)](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9E%D1%88%D0%B8%D0%B1%D0%BA%D0%B0:_%D0%A1%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0,_%D0%BA%D0%BE%D1%82%D0%BE%D1%80%D1%83%D1%8E_%D0%B2%D1%8B_%D0%B8%D1%89%D0%B8%D1%82%D0%B5,_%D0%B2%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE_%D0%BD%D0%B5%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%BD%D0%B0._%D0%9F%D0%BE%D0%B6%D0%B0%D0%BB%D1%83%D0%B9%D1%81%D1%82%D0%B0,_%D0%BF%D0%BE%D0%BF%D1%80%D0%BE%D0%B1%D1%83%D0%B9%D1%82%D0%B5_%D0%BF%D0%BE%D0%B7%D0%B6%D0%B5._(502_Bad_Gateway))
  + [5.4Ошибка: No input file specified](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9E%D1%88%D0%B8%D0%B1%D0%BA%D0%B0:_No_input_file_specified)
  + [5.5Ошибка: "File not found" в браузере или "Primary script unknown" в лог-файле](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9E%D1%88%D0%B8%D0%B1%D0%BA%D0%B0:_%22File_not_found%22_%D0%B2_%D0%B1%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80%D0%B5_%D0%B8%D0%BB%D0%B8_%22Primary_script_unknown%22_%D0%B2_%D0%BB%D0%BE%D0%B3-%D1%84%D0%B0%D0%B9%D0%BB%D0%B5)
  + [5.6Ошибка: chroot: '/usr/sbin/nginx' No such file or directory](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%9E%D1%88%D0%B8%D0%B1%D0%BA%D0%B0:_chroot:_'/usr/sbin/nginx'_No_such_file_or_directory)
  + [5.7Альтернативный скрипт для systemd](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%90%D0%BB%D1%8C%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D1%81%D0%BA%D1%80%D0%B8%D0%BF%D1%82_%D0%B4%D0%BB%D1%8F_systemd)
* [6Смотрите также](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A1%D0%BC%D0%BE%D1%82%D1%80%D0%B8%D1%82%D0%B5_%D1%82%D0%B0%D0%BA%D0%B6%D0%B5)

Установка

[**Установите**](https://wiki.archlinux.org/index.php/%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8%D1%82%D0%B5) пакет [**nginx**](https://www.archlinux.org/packages/?name=nginx).

Для установки Ruby on Rails с nginx смотрите раздел [**Ruby on Rails#The Perfect Rails Setup**](https://wiki.archlinux.org/index.php/Ruby_on_Rails#The_Perfect_Rails_Setup).

Если для обеспечения дополнительной безопасности вы хотите установить nginx в chroot-окружении, смотрите раздел [**#Установка в chroot**](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BA%D0%B0_%D0%B2_chroot).

Запуск

Запустите/включите nginx.service [**используя systemd**](https://wiki.archlinux.org/index.php/Systemd_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D1%8E%D0%BD%D0%B8%D1%82%D0%BE%D0%B2).

Страница по умолчанию, доступная по адресу [**http://127.0.0.1**](http://127.0.0.1/) располагается в /usr/share/nginx/html/index.html.

Настройка

Первые шаги по настройке и использованию nginx описаны в руководстве [**Beginner’s Guide**](http://nginx.org/ru/docs/beginners_guide.html). Вы можете настроить сервер, редактируя файлы в /etc/nginx/; главный файл настроек расположен в /etc/nginx/nginx.conf.

Более подробную информацию можно прочитать на странице [**Nginx Configuration Examples**](http://wiki.nginx.org/Configuration) и в [**официальной документации**](http://nginx.org/ru/docs/).

Приведенные далее примеры покрывают большинство типичных потребностей. Предполагается, что вы используете стандартное место расположения веб-документов (/usr/share/nginx/html). Если это не так, замените путь на свой.

**Основные настройки**

**Процессы и соединения**

Вы должны выбрать подходящее значение для worker\_processes. Этот параметр определяет сколько одновременных соединений сможет принимать nginx и сколько процессоров он сможет при этом использовать. Как правило, это значение устанавливают равным количеству аппаратных потоков в системе. Однако, начиная с версий 1.3.8 и 1.2.5, в качестве значения worker\_processes вы также можете задать auto, при этом nginx попытается автоматически подобрать оптимальное значение ([**источник**](http://nginx.org/ru/docs/ngx_core_module.html#worker_processes)).

Максимальное количество одновременных соединений, которое nginx сможет принимать, вычисляется как max\_clients = worker\_processes \* worker\_connections.

**Запуск под другим пользователем**

По умолчанию nginx выполняется от имени пользователя *nobody*. Чтобы запустить его от имени другого пользователя, измените строку user в nginx.conf:

/etc/nginx/nginx.conf

user *пользователь* *группа*; # например http

Теперь Nginx должен работать от указанного имени пользователя *пользователь* и группы *группа*. Если используется группа, имя которой совпадает с именем пользователя, то ее название можно опустить.

**Блоки server**

Посредством добавления блоков server в файл настроек возможно обслуживать сразу несколько доменов одновременно. Эти блоки работают аналогично "VirtualHosts" в [**Apache**](https://wiki.archlinux.org/index.php/Apache_HTTP_Server_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)).

В этом примере сервер принимает запросы для двух доменов: domainname1.dom и domainname2.dom:

/etc/nginx/nginx.conf

...

server {

listen 80;

server\_name domainname1.dom;

root /usr/share/nginx/domainname1.dom/html;

...

}

server {

listen 80;

server\_name domainname2.dom;

listen 443 ssl; # также прослушивать по HTTPS

root /usr/share/nginx/domainname2.dom/html;

...

}

...

[**Перезапустите**](https://wiki.archlinux.org/index.php/%D0%9F%D0%B5%D1%80%D0%B5%D0%B7%D0%B0%D0%BF%D1%83%D1%81%D1%82%D0%B8%D1%82%D0%B5) службу nginx, чтобы изменения вступили в силу.

Следует настроить DNS-сервер, например [**BIND**](https://wiki.archlinux.org/index.php/BIND) или [**dnsmasq**](https://wiki.archlinux.org/index.php/Dnsmasq), чтобы у подключающихся клиентов эти доменные имена разрешались в IP-адрес сервера.

А пока вы можете просто добавить их в ваш файл /etc/hosts, заменив 192.168.0.101 на фактический IP-адрес сервера:

192.168.0.101 domainname1.dom

192.168.0.101 domainname2.dom

**TLS/SSL**

[**openssl**](https://www.archlinux.org/packages/?name=openssl) предоставляет поддержку TLS/SSL и установлен по умолчанию на установленных Arch.

**Совет:**Перед тем как настраивать SSL, вы можете почитать документацию [**ngx\_http\_ssl\_module**](http://nginx.org/en/docs/http/ngx_http_ssl_module.html#ssl_certificate)

Создайте секретный ключ и самоподписанный сертификат. Это подходит для большинства случаев, в которых не требуется [**CSR**](https://en.wikipedia.org/wiki/Certificate_signing_request):

# cd /etc/nginx/

# openssl req -new -x509 -nodes -newkey rsa:4096 -keyout nginx.key -out nginx.crt -days 1095

# chmod 400 nginx.key

# chmod 444 nginx.crt

**Примечание:**Опция -days является необязательной, а RSA keysize можно уменьшить до 2048 (по умолчанию).

Если же вам нужно создать [**CSR**](https://en.wikipedia.org/wiki/Certificate_signing_request), то следуйте данным инструкциям по созданию ключа, вместо приведённых выше:

# openssl genpkey -algorithm RSA -pkeyopt rsa\_keygen\_bits:4096 -out nginx.key

# chmod 400 nginx.key

# openssl req -new -sha256 -key nginx.key -out nginx.csr

# openssl x509 -req -days 1095 -in nginx.csr -signkey nginx.key -out nginx.crt

**Примечание:**Для дополнительных опций openssl, прочтите [**man страницу**](https://www.openssl.org/docs/apps/openssl.html) или изучите [**подробную документацию**](https://www.openssl.org/docs/) по openssl.

**Важно:**Если вы планируете развернуть SSL/TLS, вы должны знать, что некоторые вариации и реализации [**всё ещё**](https://weakdh.org/#affected) [**подвержены атакам**](https://en.wikipedia.org/wiki/Transport_Layer_Security#Attacks_against_TLS.2FSSL). За дополнительной информацией о текущих подверженных версиях этих реализаций SSL/TLS и как применить нужные настройки к nginx посетите [**http://disablessl3.com/**](http://disablessl3.com/) и [**https://weakdh.org/sysadmin.html**](https://weakdh.org/sysadmin.html)

Пример nginx.conf, использующего SSL:

/etc/nginx/nginx.conf

http {

ssl\_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH";

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

ssl\_prefer\_server\_ciphers on;

ssl\_session\_cache shared:SSL:10m;

add\_header Strict-Transport-Security "max-age=63072000; includeSubdomains; preload";

add\_header X-Frame-Options DENY;

add\_header X-Content-Type-Options nosniff;

ssl\_session\_tickets off;

ssl\_stapling on;

ssl\_stapling\_verify on;

resolver 8.8.8.8 8.8.4.4 valid=300s; # Google DNS Servers

resolver\_timeout 5s;

}

server {

#listen 80; # Раскомментируйте, чтобы также слушать HTTP запросы

listen 443 ssl;

server\_name localhost;

ssl\_certificate nginx.crt;

ssl\_certificate\_key nginx.key;

root /usr/share/nginx/html;

location / {

index index.html index.htm index.php;

}

}

**Совет:**У Mozilla есть полезная [**SSL/TLS статья**](https://wiki.mozilla.org/Security/Server_Side_TLS), которая описывает рекомендации по настройке [**специально для nginx**](https://wiki.mozilla.org/Security/Server_Side_TLS#Nginx), а также [**автоматизированный инструмент**](https://mozilla.github.io/server-side-tls/ssl-config-generator/), который поможет вам создать более безопасную конфигурацию.

**Совет:**[**Cipherli.st**](https://cipherli.st/) показывает примеры надёжных настроек SSL и инструкции для наиболее современных веб серверов.

[**Перезапустите**](https://wiki.archlinux.org/index.php/%D0%9F%D0%B5%D1%80%D0%B5%D0%B7%D0%B0%D0%BF%D1%83%D1%81%D1%82%D0%B8%D1%82%D0%B5) службу nginx, чтобы изменения вступили в силу.

**FastCGI**

FastCGI или просто FCGI — это протокол, являющийся интерфейсом между веб-сервером и интерактивными программами. Это модифицированный CGI (*Common Gateway Interface*), главная цель которого — снизить накладные расходы, связанные со взаимодействием веб сервера и CGI программ, тем самым позволяя серверу обрабатывать большее количество запросов одновременно.

Технология FastCGI встроена в nginx для работы со многими внешними инструментами, например, Perl, [**PHP**](https://wiki.archlinux.org/index.php/PHP) и [**Python**](https://wiki.archlinux.org/index.php/Python_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)).

**Реализация PHP**

В качестве FastCGI-сервера для PHP рекомендуется использовать [**PHP-FPM**](http://php-fpm.org/).

**Настройка PHP**

[**Установите**](https://wiki.archlinux.org/index.php/%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8%D1%82%D0%B5) пакеты [**php**](https://www.archlinux.org/packages/?name=php) и [**php-fpm**](https://www.archlinux.org/packages/?name=php-fpm).

Опция open\_basedir в /etc/php/php.ini должна содержать список всех каталогов, с файлами PHP, которые должны быть доступны серверу. Например, для /usr/share/nginx/html/ и /usr/share/webapps/:

open\_basedir = /usr/share/webapps/:/srv/http/:/usr/share/nginx/html/:/home/:/tmp/:/usr/share/pear/

После этого настройте нужные вам модули. Например, для использования sqlite3 нужно установить [**php-sqlite**](https://www.archlinux.org/packages/?name=php-sqlite). Затем включите этот модуль в файле /etc/php/php.ini, раскомментировав следующую строку:

extension=sqlite3.so

Основным конфигурационным файлом PHP-FPM является /etc/php/php-fpm.conf. [**Включите**](https://wiki.archlinux.org/index.php/%D0%92%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D0%B5) и [**запустите**](https://wiki.archlinux.org/index.php/%D0%97%D0%B0%D0%BF%D1%83%D1%81%D1%82%D0%B8%D1%82%D0%B5) [**systemd**](https://wiki.archlinux.org/index.php/Systemd) службу php-fpm.

**Примечание:**Если вы запускаете nginx в изолированном окружении (к примеру, chroot находится в /srv/nginx-jail, веб-документы расположены в /srv/nginx-jail/www), то вы должны в /etc/php/php-fpm.conf добавить опции chroot /srv/nginx-jail и listen = /srv/nginx-jail/run/php-fpm/php-fpm.sock внутри секции пула (по умолчанию это [www]). Создайте каталог для файла сокета, если его нет.

**MariaDB**

Настройте MySQL/MariaDB как описано в [**MariaDB**](https://wiki.archlinux.org/index.php/MariaDB).

Раскомментируйте [**хотя бы одну**](http://www.php.net/manual/en/mysqlinfo.api.choosing.php) из следующих строк в /etc/php/php.ini:

extension=pdo\_mysql.so

extension=mysqli.so

**Важно:**Начиная с PHP 5.5, mysql.so объявлен [**устаревшим**](http://www.php.net/manual/de/migration55.deprecated.php), ваши лог файлы будут переполнены.

Вы можете добавить менее привилегированных MySQL пользователей для ваших веб скриптов. Вы можете также захотеть отредактировать /etc/mysql/my.cnf и раскомментировать строку skip-networking, чтобы MySQL сервер был доступен только из localhost. Вы должны перезапустить MySQL, чтобы изменения вступили в силу.

**Совет:**Вы можете захотеть установить инструменты вроде [**phpMyAdmin**](https://wiki.archlinux.org/index.php/PhpMyAdmin), [**Adminer**](https://wiki.archlinux.org/index.php/Adminer) или [**mysql-workbench**](https://www.archlinux.org/packages/?name=mysql-workbench), чтобы работать с вашими базами данных.

**Настройка nginx**

**Добавление к основной конфигурации**

Внутри каждого блока server, который обслуживает веб-приложение PHP должен находиться блок location:

location ~ \.php$ {

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

fastcgi\_index index.php;

include fastcgi.conf;

}

Если требуется обрабатывать другие расширения наряду с PHP (например *.html* и *.htm*):

location ~ \.(php**|html|htm**)$ {

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

fastcgi\_index index.php;

include fastcgi.conf;

}

Все расширения, обрабатываемые в php-fpm должны быть также явно добавлены в /etc/php/php-fpm.conf:

security.limit\_extensions = .php .html .htm

**Примечание:**Аргумент fastcgi\_pass должен быть определен как TCP-сокет или сокет Unix выбранным FastCGI сервером в его конфигурационном файле. **По умолчанию** для php-fpm используется сокет

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

Вы можете использовать также общий TCP-сокет:

fastcgi\_pass 127.0.0.1:9000;

Однако, доменные сокеты Unix должны работать быстрее.

Пример, показанный ниже, является копией рабочей конфигурации. Заметьте, что в этом примере путь к root определен непосредственно в server, а не внутри location(как это сделано в конфигурации по умолчанию).

server {

listen 80;

server\_name localhost;

root /usr/share/nginx/html;

location / {

index index.html index.htm index.php;

}

location ~ \.php$ {

#fastcgi\_pass 127.0.0.1:9000; (depending on your php-fpm socket configuration)

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

fastcgi\_index index.php;

include fastcgi.conf;

}

}

**Управление несколькими блоками (опционально)**

Если вы добавляете однотипную конфигурацию для PHP сразу во множество блоков server, может оказаться удобнее использовать для этого внешний файл:

/etc/nginx/php.conf

location ~ \.php$ {

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

fastcgi\_index index.php;

include fastcgi.conf;

}

Теперь включите файл php.conf в каждый из блоков server:

/etc/nginx/nginx.conf

server = {

...

include php.conf;

}

**Проверка конфигурации**

[**Перезапустите**](https://wiki.archlinux.org/index.php/%D0%9F%D0%B5%D1%80%D0%B5%D0%B7%D0%B0%D0%BF%D1%83%D1%81%D1%82%D0%B8%D1%82%D0%B5) службы php-fpm и nginx после изменения настроек, чтобы изменения вступили в силу.

Чтобы проверить работу FastCGI, создайте новый файл *.php* внутри каталога веб-документов, содержащий:

<?php

phpinfo();

?>

При открытии файла в браузере должна отобразиться информационная страница с текущими настройками PHP.

Смотрите [**#Решение проблем**](https://wiki.archlinux.org/index.php/Nginx_%28%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9%29#%D0%A0%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B1%D0%BB%D0%B5%D0%BC), если новая конфигурация не работает.

**Реализация CGI**

Эта реализация нужна для CGI-приложений.

**fcgiwrap**

[**Установите**](https://wiki.archlinux.org/index.php/%D0%A3%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8%D1%82%D0%B5) [**fcgiwrap**](https://www.archlinux.org/packages/?name=fcgiwrap). Файл настроек находится в /usr/lib/systemd/system/fcgiwrap.socket. [**Включите**](https://wiki.archlinux.org/index.php/%D0%92%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D0%B5) и [**запустите**](https://wiki.archlinux.org/index.php/%D0%97%D0%B0%D0%BF%D1%83%D1%81%D1%82%D0%B8%D1%82%D0%B5) fcgiwrap.socket.

**Несколько рабочих потоков**

Если вы хотите породить несколько рабочих потоков, вам рекомендуется использовать [**multiwatch**](https://aur.archlinux.org/packages/multiwatch/)AUR, который умеет отслеживать упавшие подпроцессы и перезапускать их. Вам нужно использовать spawn-fcgi, чтобы создать доменный сокет Unix, так как multiwatch не может обрабатывать сокеты, созданные [**systemd**](https://wiki.archlinux.org/index.php/Systemd_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)), однако, *fcgiwrap* сама по себе не вызывает никаких проблем, если вызывается непосредственно из юнит-файла.

Скопируйте юнит-файл из /usr/lib/systemd/system/fcgiwrap.service в /etc/systemd/system/fcgiwrap.service (и юнит fcgiwrap.socket, если он есть), и отредактируйте строку ExecStart в соответствии с вашими нуждами. В примере показан юнит файл, который использует [**multiwatch**](https://aur.archlinux.org/packages/multiwatch/)AUR. Убедитесь, что fcgiwrap.socket не включен и не запущен, потому что он будет конфликтовать с этим юнитом:

/etc/systemd/system/fcgiwrap.service

[Unit]

Description=Simple CGI Server

After=nss-user-lookup.target

[Service]

ExecStartPre=/bin/rm -f /run/fcgiwrap.socket

ExecStart=/usr/bin/spawn-fcgi -u http -g http -s /run/fcgiwrap.sock -n -- /usr/bin/multiwatch -f 10 -- /usr/sbin/fcgiwrap

ExecStartPost=/usr/bin/chmod 660 /run/fcgiwrap.sock

PrivateTmp=true

Restart=on-failure

[Install]

WantedBy=multi-user.target

Выберите подходящий -f 10, чтобы изменить количество порождаемых подпроцессов.

**Важно:**Строка ExecStartPost требуется из-за странного поведения, которое я наблюдаю при использовании опции -M 660 для spawn-fcgi. Устанавливается неправильный режим. Может это баг?

**Настройка nginx**

Внутри каждого блока server CGI-приложения должен находиться вложенный блок location:

location ~ \.cgi$ {

root /path/to/server/cgi-bin;

fastcgi\_pass unix:/run/fcgiwrap.sock;

include fastcgi.conf;

}

Стандартным сокетом для fcgiwrap является /run/fcgiwrap.sock.

Установка в chroot

Установка nginx в [**chroot**](https://wiki.archlinux.org/index.php/Change_root_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)) добавляет дополнительный уровень безопасности. Для максимальной безопасности chroot должен включать только файлы, необходимые для запуска сервера nginx, при этом все файлы должны иметь по возможности максимально ограниченные права доступа. Например, как можно больше файлов должно принадлежать пользователю root, а таким каталогам, как /usr/bin должен быть установлен запрет на чтение и запись.

Arch поставляется с пользователем http и группой по умолчанию, от имени которых запускается сервер. Измененный корневой каталог будет находиться в /srv/http.

Существует perl-скрипт для создания chroot-окружения, который доступен в [**jail.pl gist**](https://gist.github.com/4365696). Вы можете либо использовать его, либо следовать дальнейшим инструкциям из этой статьи. Скрипт требует прав суперпользователя для работы. Вам нужно будет раскомментировать строку, перед тем, как он сможет выполнять какие-либо изменения.

**Создание необходимых устройств**

Для nginx нужны /dev/null, /dev/random и /dev/urandom. Чтобы установить их в chroot мы создадим каталог /dev и добавим устройства с помощью *mknod*. Избегайте монтирования всех устройств в /dev: тогда, даже если chroot будет скомпрометирован, атакующий должен будет выбраться из chroot-окружения чтобы добраться до важных устройств, например /dev/sda1.

**Совет:**Убедитесь, что /src/http примонтирован без опции no-dev

**Совет:**Смотрите [**mknod(1)**](https://jlk.fjfi.cvut.cz/arch/manpages/man/mknod.1) и ls -l /dev/{null,random,urandom}, чтобы лучше понять опции *mknod*.

# export JAIL=/srv/http

# mkdir $JAIL/dev

# mknod -m 0666 $JAIL/dev/null c 1 3

# mknod -m 0666 $JAIL/dev/random c 1 8

# mknod -m 0444 $JAIL/dev/urandom c 1 9

**Создание необходимых каталогов**

Для работы nginx требует определенный набор файлов. Перед тем, как их копировать, создайте для них соответствующие каталоги. Предполагается, что ваш корневой каталог веб-документов nginx находится в /srv/http/www.

# mkdir -p $JAIL/etc/nginx/logs

# mkdir -p $JAIL/usr/{lib,bin}

# mkdir -p $JAIL/usr/share/nginx

# mkdir -p $JAIL/var/{log,lib}/nginx

# mkdir -p $JAIL/www/cgi-bin

# mkdir -p $JAIL/{run,tmp}

# cd $JAIL; ln -s usr/lib lib

**Примечание:**Если вы используете 64-битное ядро, вам нужно создать символические ссылки для lib64 и usr/lib64 в usr/lib: cd $JAIL; ln -s usr/lib lib64 и cd $JAIL/usr; ln -s lib lib64.

Затем смонтируйте $JAIL/tmp и $JAIL/run как tmpfs-ы. Размер должен быть ограничен, чтобы быть уверенным, что атакующий не сможет занять всю доступную RAM.

# mount -t tmpfs none $JAIL/run -o 'noexec,size=1M'

# mount -t tmpfs none $JAIL/tmp -o 'noexec,size=100M'

Для того, чтобы монтирование выполнялось автоматически при загрузке системы, добавьте следующие записи в /etc/fstab:

/etc/fstab

tmpfs /srv/http/run tmpfs rw,noexec,relatime,size=1024k 0 0

tmpfs /srv/http/tmp tmpfs rw,noexec,relatime,size=102400k 0 0

**Заполнение chroot**

Сначала скопируйте простые файлы.

# cp -r /usr/share/nginx/\* $JAIL/usr/share/nginx

# cp -r /usr/share/nginx/html/\* $JAIL/www

# cp /usr/bin/nginx $JAIL/usr/bin/

# cp -r /var/lib/nginx $JAIL/var/lib/nginx

Теперь скопируйте нужные библиотеки. Используйте *ldd*, чтобы отобразить их и скопируйте все файлы в правильное место. Копирование предпочтительнее, чем создание жестких ссылок, потому, что даже если атакующий получит права записи в файлы, они не смогут уничтожить или изменить системные файлы вне chroot-окружения.

$ ldd /usr/bin/nginx

linux-vdso.so.1 (0x00007fffc41fe000)

libpthread.so.0 => /usr/lib/libpthread.so.0 (0x00007f57ec3e8000)

libcrypt.so.1 => /usr/lib/libcrypt.so.1 (0x00007f57ec1b1000)

libstdc++.so.6 => /usr/lib/libstdc++.so.6 (0x00007f57ebead000)

libm.so.6 => /usr/lib/libm.so.6 (0x00007f57ebbaf000)

libpcre.so.1 => /usr/lib/libpcre.so.1 (0x00007f57eb94c000)

libssl.so.1.0.0 => /usr/lib/libssl.so.1.0.0 (0x00007f57eb6e0000)

libcrypto.so.1.0.0 => /usr/lib/libcrypto.so.1.0.0 (0x00007f57eb2d6000)

libdl.so.2 => /usr/lib/libdl.so.2 (0x00007f57eb0d2000)

libz.so.1 => /usr/lib/libz.so.1 (0x00007f57eaebc000)

libGeoIP.so.1 => /usr/lib/libGeoIP.so.1 (0x00007f57eac8d000)

libgcc\_s.so.1 => /usr/lib/libgcc\_s.so.1 (0x00007f57eaa77000)

libc.so.6 => /usr/lib/libc.so.6 (0x00007f57ea6ca000)

/lib64/ld-linux-x86-64.so.2 (0x00007f57ec604000)

# cp /lib64/ld-linux-x86-64.so.2 $JAIL/lib

Для файлов, находящихся в /usr/lib, вы можете воспользоваться следующей командой:

# cp $(ldd /usr/bin/nginx | grep /usr/lib | sed -sre 's/(.+)(\/usr\/lib\/\S+).+/\2/g') $JAIL/usr/lib

**Примечание:**Не пытайтесь скопировать linux-vdso.so — это не настоящая библиотека и ее не существует в /usr/lib. Аналогично ld-linux-x86-64.so также будет отображена в /lib64 для 64-битной системы.

Копируйте другие необходимые библиотеки и системные файлы.

# cp /usr/lib/libnss\_\* $JAIL/usr/lib

# cp -rfvL /etc/{services,localtime,nsswitch.conf,nscd.conf,protocols,hosts,ld.so.cache,ld.so.conf,resolv.conf,host.conf,nginx} $JAIL/etc

Создайте файлы пользователей и групп в chroot-окружении. Таким образом, в chroot-окружении будут доступны только указанные пользователи, и никакая информация о пользователях из основной системы не будет доступна атакующему, получившему доступ в chroot-окружение.

$JAIL/etc/group

http:x:33:

nobody:x:99:

$JAIL/etc/passwd

http:x:33:33:http:/:/bin/false

nobody:x:99:99:nobody:/:/bin/false

$JAIL/etc/shadow

http:x:14871::::::

nobody:x:14871::::::

$JAIL/etc/gshadow

http:::

nobody:::

# touch $JAIL/etc/shells

# touch $JAIL/run/nginx.pid

Наконец, сделайте права доступа максимально ограниченными. Как можно больше должно принадлежать суперпользователю и быть закрытым для записи.

# chown -R root:root $JAIL/

# chown -R http:http $JAIL/www

# chown -R http:http $JAIL/etc/nginx

# chown -R http:http $JAIL/var/{log,lib}/nginx

# chown http:http $JAIL/run/nginx.pid

# find $JAIL/ -gid 0 -uid 0 -type d -print | xargs sudo chmod -rw

# find $JAIL/ -gid 0 -uid 0 -type d -print | xargs sudo chmod +x

# find $JAIL/etc -gid 0 -uid 0 -type f -print | xargs sudo chmod -x

# find $JAIL/usr/bin -type f -print | xargs sudo chmod ug+rx

# find $JAIL/ -group http -user http -print | xargs sudo chmod o-rwx

# chmod +rw $JAIL/tmp

# chmod +rw $JAIL/run

Если ваш сервер будет принимать входящие соединения на 80 порту (или любому другому порту в диапазоне [1-1023]), дайте исполнителю chroot права на соединение с этими портами без необходимости прав суперпользователя.

# setcap 'cap\_net\_bind\_service=+ep' $JAIL/usr/bin/nginx

**Отредактируйте nginx.service для запуска chroot**

Перед редактированием юнит-файла nginx.service неплохо будет скопировать его в /etc/systemd/system/, так как там юнит файлы имеют приоритет над теми, что в /usr/lib/systemd/system/. Это значит, что обновление nginx не перезапишет ваш собственный файл *.service*.

# cp /usr/lib/systemd/system/nginx.service /etc/systemd/system/nginx.service

Юнит systemd должен быть настроен так, чтобы запускать nginx в chroot от имени пользователя http и хранить pid-файл в chroot.

**Примечание:**Я не уверен, нужно ли хранить pid-файл в chroot.

/etc/systemd/system/nginx.service

[Unit]

Description=A high performance web server and a reverse proxy server

After=syslog.target network.target

[Service]

Type=forking

PIDFile=/srv/http/run/nginx.pid

ExecStartPre=/usr/bin/chroot --userspec=http:http /srv/http /usr/bin/nginx -t -q -g 'pid /run/nginx.pid; daemon on; master\_process on;'

ExecStart=/usr/bin/chroot --userspec=http:http /srv/http /usr/bin/nginx -g 'pid /run/nginx.pid; daemon on; master\_process on;'

ExecReload=/usr/bin/chroot --userspec=http:http /srv/http /usr/bin/nginx -g 'pid /run/nginx.pid; daemon on; master\_process on;' -s reload

ExecStop=/usr/bin/chroot --userspec=http:http /srv/http /usr/bin/nginx -g 'pid /run/nginx.pid;' -s quit

[Install]

WantedBy=multi-user.target

**Примечание:**Обновление nginx с помощью pacman не обновит установленную в chroot копию. Вы должны вручную выполнять обновления, повторяя указанные выше шаги по переносу файлов. Не забудьте также обновить библиотеки, которые использует nginx.

Теперь вы можете спокойно удалить установленный вне chroot nginx.

# pacman -Rsc nginx

Если вы не удалили установленный вне chroot nginx, проверьте, что работающий процесс nginx — это действительно именно тот, что в находится chroot. Для этого посмотрите, куда указывает символическая ссылка /proc/{PID}/root: она должен указывать на /srv/http, а не на /.

# ps -C nginx | awk '{print $1}' | sed 1d | while read -r PID; do ls -l /proc/$PID/root; done

Решение проблем

**Валидация конфигурации**

# nginx -t

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok

nginx: configuration file /etc/nginx/nginx.conf test is successful

**При доступе с локального IP перенаправляется на localhost**

Решение с [**форума Arch Linux**](https://bbs.archlinux.org/viewtopic.php?pid=780561#p780561).

В файле /etc/nginx/nginx.conf найдите незакомментированную строку server\_name localhost (без # вначале) и добавьте под ней:

server\_name\_in\_redirect off;

По умолчанию, nginx перенаправляет любые запросы на указанное в опции server\_name имя.

**Ошибка: Страница, которую вы ищите, временно недоступна. Пожалуйста, попробуйте позже. (502 Bad Gateway)**

Это из-за того, что сервер FastCGI не запущен или используемый сокет имеет неправильные права доступа.

Попробуйте [**этот ответ**](https://stackoverflow.com/questions/4252368/nginx-502-bad-gateway/16497957#16497957), чтобы исправить 502 ошибку.

В Archlinux, файлом настройки, упомянутом по ссылке выше, является /etc/php/php-fpm.conf.

При определённых обстоятельствах, fcgiwrap.socket может не запуститься правильно и создать бесполезный сокет юникс домена /run/fcgiwrap.sock.

Попробуйте [**остановить**](https://wiki.archlinux.org/index.php/%D0%9E%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8%D1%82%D1%8C) службу fcgiwrap.socket и удалить файл доменного юникс сокета по умолчанию.

# rm /run/fcgiwrap.sock

Затем [**запустите**](https://wiki.archlinux.org/index.php/%D0%97%D0%B0%D0%BF%D1%83%D1%81%D1%82%D0%B8%D1%82%D0%B5) fcgiwrap.service вместо него. Проверьте статус fcgiwrap.service и нового доменного юникс сокета /run/fcgiwrap.sock:

$ systemctl status fcgiwrap.service

$ ls /run/fcgiwrap.sock

Если это сработало, [**отключите**](https://wiki.archlinux.org/index.php/%D0%9E%D1%82%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D0%B5) fcgiwrap.socket и [**включите**](https://wiki.archlinux.org/index.php/%D0%92%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D0%B5) fcgiwrap.service.

**Ошибка: No input file specified**

1. Скорее всего у вас не установлена переменная SCRIPT\_FILENAME, содержащая полный путь до ваших скриптов. Если конфигурация nginx (fastcgi\_param SCRIPT\_FILENAME) правильная, то эта ошибка означает, что php не смог загрузить запрашиваемый скрипт. Часто это просто оказывается ошибкой прав доступа, и вы можете запустить php-cgi с правами root:

# spawn-fcgi -a 127.0.0.1 -p 9000 -f /usr/bin/php-cgi

или вам следует создать группу и пользователя для запуска php-cgi:

# groupadd www

# useradd -g www www

# chmod +w /srv/www/nginx/html

# chown -R www:www /srv/www/nginx/html

# spawn-fcgi -a 127.0.0.1 -p 9000 -u www -g www -f /usr/bin/php-cgi

2. Другой причиной может быть то, что задан неправильный аргумент root в секции location ~ \.php$ в nginx.conf. Убедитесь, что root указывает на ту же директорию, что и в location / на том же сервере. Либо вы можете просто задать абсолютный путь до корневого каталога, не определяя его в каких-либо location секциях.

3. Убедитесь, что переменная open\_basedir в /etc/php/php.ini также содержит путь, который соответствует аргументу root в nginx.conf.

4. Также обратите внимание, что не только php-скрипты должны иметь права на чтение, но также и вся структура каталогов должна иметь право на исполнение, чтобы пользователь PHP мог добраться до этого каталога.

**Ошибка: "File not found" в браузере или "Primary script unknown" в лог-файле**

Убедитесь, что вы определили root и index в ваших директивах server или location:

location ~ \.php$ {

root /srv/http/root\_dir;

index index.php;

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

include fastcgi.conf;

}

Также убедитесь, что запрашиваемый файл существует на сервере.

**Ошибка: chroot: '/usr/sbin/nginx' No such file or directory**

Если у вас возникает эта ошибка при запуске демона *nginx* в chroot, скорее всего, это происходит из-за отсутствующих 64-битных библиотек в изолированном окружении.

Если ваш chroot запущен в /srv/http, вам нужно добавить требуемые 64-битные библиотеки.

Сначала создайте каталоги:

# mkdir /srv/http/usr/lib64

# cd /srv/http; ln -s usr/lib64 lib64

Затем скопируйте требуемые 64-битные библиотеки, перечисленные командой ldd /usr/sbin/nginx в /srv/http/usr/lib64.

При запуске от root, на библиотеки должны быть права чтения и исполнения для всех пользователей, так что изменения не требуются.

**Альтернативный скрипт для systemd**

На чистой systemd вы можете получить преимущества при использовании связки chroot и systemd [**[1]**](http://0pointer.de/blog/projects/changing-roots.html). На основе заданных [**пользователя и группы**](http://wiki.nginx.org/CoreModule#user) и pid:

/etc/nginx/nginx.conf

user http;

pid /run/nginx.pid;

абсолютным путем к файлу является /srv/http/etc/nginx/nginx.conf.3

/etc/systemd/system/nginx.service

[Unit]

Description=nginx (Chroot)

After=syslog.target network.target

[Service]

Type=forking

PIDFile=/srv/http/run/nginx.pid

RootDirectory=/srv/http

ExecStartPre=/usr/sbin/nginx -t -c /etc/nginx/nginx.conf

ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf

ExecReload=/usr/sbin/nginx -c /etc/nginx/nginx.conf -s reload

ExecStop=/usr/sbin/nginx -c /etc/nginx/nginx.conf -s stop

[Install]

WantedBy=multi-user.target

Нет необходимости задавать расположение по умолчанию, nginx по умолчанию загружает -c /etc/nginx/nginx.conf, хотя вообще это хорошая идея.

Также можно запускать **только** ExecStart как chroot с параметром RootDirectoryStartOnly заданным как yes [**man systemd service**](http://www.freedesktop.org/software/systemd/man/systemd.service.html) или запустить его до точки монтирования в качестве эффективного или [**пути systemd**](http://www.freedesktop.org/software/systemd/man/systemd.path.html).

/etc/systemd/system/nginx.path

[Unit]

Description=nginx (Chroot) path

[Path]

PathExists=/srv/http/site/Public\_html

[Install]

WantedBy=default.target

[**Включите**](https://wiki.archlinux.org/index.php/%D0%92%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D0%B5) nginx.path и замените WantedBy=default.target на WantedBy=nginx.path in /etc/systemd/system/nginx.service.

Ссылка PIDFile в файле юнита позволяет systemd следить за процессом (необходим абсолютный путь). Если это нежелательно, вы можете изменить тип one-shoot по умолчанию и удалить ссылку из файла юнита.

Смотрите также

* [**Very good in-depth 2014 look at Nginx security and Reverse Proxying**](https://calomel.org/nginx.html)
* [**Nginx Official Site**](http://nginx.org/)
* [**Nginx HowTo**](http://calomel.org/nginx.html)
* [**Custom Nginx Indexer**](http://blog.gotux.net/tutorial/custom-nginx-indexer/)
* [**Installing LEMP (Nginx, PHP, MySQL with MariaDB engine and PhpMyAdmin) in Arch Linux**](http://www.tecmint.com/install-nginx-php-mysql-with-mariadb-engine-and-phpmyadmin-in-arch-linux/)

[Categories](https://wiki.archlinux.org/index.php/Special:Categories):

* [Web server (Русский)](https://wiki.archlinux.org/index.php/Category:Web_server_(%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9))
* [Русский](https://wiki.archlinux.org/index.php/Category:%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%B8%D0%B9)