Последнее время я немного занимаюсь ускорением сайтов и сегодня напишу на эту тему. Я расскажу, как собрать свой rpm пакет nginx с поддержкой tls 1.3 и шифрованием brotli. Зачем все это нужно и какие сулит преимущества, читайте дальше.

Содержание:

Введение

Сразу перейду к сути, так как заголовок немного сбивает с толку. На самом деле nginx уже давно поддерживает tls 1.3, достаточно в параметрах указать его поддержку. Но это будет работать только на системах, с версией **openssl версии 1.1.1 и выше**. В Centos 7 на текущий момент с последними обновлениями у вас будет версия 1.0.2k-fips.

```
# openssl version
OpenSSL 1.0.2k-fips  26 Jan 2017
```

Для того, чтобы tls 1.3 заработал, nginx должен быть собран с версией openssl 1.1.1 или выше. Я как раз и хочу рассказать, как это сделать. Если вы не хотите сами разбираться во всем этом, то можете воспользоваться репозиторием, где все уже сделали за вас. Вам достаточно будет только подключить его и выполнить установку. Пример такого репозитория — https://repo.codeit.guru Я не знаю, что это за люди и какие точно изменения вносят в стандартные пакеты. Можете сами посмотреть их сайт и решить, можно им доверять или нет.

Я расскажу, как собрать свой пакет самостоятельно и добавить туда те функции, которые вам нужны. Помимо поддержки tls 1.3 я добавлю модуль для поддержки **шифрования brotli**, вместо стандартного gzip.

Не хочется подробно описывать, в чем смысл tls 1.3 и brotli. Если кратко — они ускоряют работу сайта. Но ускоряют не значительно. Не нужно очень сильно рассчитывать на подобное ускорение Это имеет смысл, если у вас все остальное идеально и вы хотите еще немного ускориться. Более подробно об этом можно прочитать у cloudflare:

- tls 1.3 — https://blog.cloudflare.com/rfc-8446-aka-tls-1-3/
- brotli — https://blog.cloudflare.com/results-experimenting-brotli/

Подводя итог, еще раз поясню, что я буду собирать rpm пакет с последней версией openssl и модулем brotli.

Подготовка к сборке своего rpm пакета

Для начала поставим все, что нам понадобится для самостоятельной сборки своего rpm пакета.

```
# yum groupinstall "Development Tools" && yum install rpmdevtools yum-utils wget git
```

Подключим репозитории nginx **mainline** ветки для CentOS 7. Обращаю внимание, что используется не стабильная версия stable, а основная — mainline. Она достаточно надежная, в ней все свежие обновления.

```
# mcedit /etc/yum.repos.d/nginx.repo
```

```
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/mainline/centos/7/$basearch/
gpgcheck=0
```

```
enabled=1

[nginx-source]
name=nginx source repo
baseurl=http://nginx.org/packages/mainline/centos/7/SRPMS/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```

Обновите репозитории:

```
# yum update
```

Перейдем в домашний каталог и создадим там структуру каталогов.

```
# cd ~
# rpmdev-setuptree
```

У nginx в репозитории уже есть все в готовом виде для сборки из исходников. Загрузим пакет с исходниками и установим его.

```
# yumdownloader --source nginx
# rpm -Uvh nginx*.src.rpm
```

Если работаете от root, то получите пачку предупреждений.

Для сборки пакетов рекомендуется использовать отдельного пользователя. Но это не критично.

Устанавливаем зависимости, необходимые для сборки.

```
# yum-builddep nginx
```

Сборка rpm пакета nginx с поддержкой brotli и tls 1.3

Для сборки rpm пакета все готово. Нам нужно скачать исходники openssl и модуля brotli, которые мы будем использовать. На момент написания статьи, последняя версия openssl — 1.1.1a. Ее и будем использовать. Для этого идем на сайт https://www.openssl.org и копируем ссылку на загрузку из раздела Download.

```
# cd /usr/src
# wget https://www.openssl.org/source/openssl-1.1.1a.tar.gz
```

Сразу же распакуем:

```
# tar xzvf openssl-*.tar.gz
```

Скачиваем модуль brotli через git.

```
# git clone https://github.com/eustas/ngx_brotli.git
# cd ngx_brotli
# git submodule update --init
```

Для сборки rpm все готово. Теперь укажем в параметрах сборки нашу версию openssl и модуль brotli.

```
# mcedit ~/rpmbuild/SPECS/nginx.spec
```

Добавляем в строку, начинающуюся с %define BASE_CONFIGURE_ARGS в самый конец к списку параметров:

```
--add-module=/usr/src/ngx_brotli --with-openssl=/usr/src/openssl-1.1.1a --with-openssl-opt=enable-tls1_3
```

Запускаем сборку rpm:

```
# cd ~/rpmbuild/SPECS/
# rpmbuild -ba nginx.spec
```

Устанавливаем собранный пакет:

```
cd ~/rpmbuild/RPM/
rpm -Uvh nginx-1.15.7-1.el7_4.ngx.x86_64.rpm
```

Проверка работы tls 1.3 и brotli в nginx

Запускаем nginx:

```
# systemctl start nginx
```

Проверяем версию openssl и наличие brotli модуля:

```
nginx -V
```

Для работы нового функционала, добавляем параметры в */etc/nginx/nginx.conf*:

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3;
ssl_ciphers TLS13-CHACHA20-POLY1305-SHA256:TLS13-AES-128-GCM-SHA256:TLS13-AES-256-GCM-SHA384:ECDHE:!COMPLEMENTOFDEFAULT;
ssl_prefer_server_ciphers on;
ssl_stapling on;
add_header Strict-Transport-Security max-age=15768000;
```

```
brotli_static on;
brotli on;
brotli_comp_level 6;
brotli_types text/plain text/css text/xml application/javascript image/x-icon image/svg+xml;
```

Обращаю внимание на настройки nginx. Если у вас в виртуальных хостах стоят разные настройки ssl, то могут возникнуть проблемы с работой tls 1.3. Изначально я тестировал все на отдельном виртуальном хосте, а остальные не трогал. Но у меня ничего не работало. Какие бы настройки ssl я не ставил, tls 1.3 не работал, только 1.2. Что я только не перепробовал — раз 10 пересобирал nginx с разными параметрами и версиями openssl.

Заработало все только после того, как я у всех виртуальных хостов убрал настройки ssl, кроме путей до сертификатов, и прописал глобальные настройки для всех в nginx.conf. После этого tls 1.3 заработал.

Проверяем конфигурацию на ошибки и перезапускаем nginx:

```
# nginx -t
# nginx -s reload
```

Открываем тестовый сайт в последней версии chrome и проверяем с помощью dev tools.

Сравните с остальными сайтами. Сжатие в основном будет gzip, а версия tls -1.2. А мы теперь молодцы, на острие прогресса — у нас все самое новое.

> Обращаю внимание на то, что проверять нужно в свежей версии браузера. Не все еще поддерживают работу с tls 1.3. К примеру, на момент написания статьи Chrome уже поддерживал, а Яндекс.Браузер нет.

Запрет обновления пакета через yum

В завершении рекомендую заблокировать обновление nginx через yum, иначе вы замените свою сборку очередной новой версией из официальной репы. Для блокировки, установите пакет **yum-plugin-versionlock**:

```
# yum install yum-plugin-versionlock
```

Теперь блокируем пакет nginx:

```
# yum versionlock nginx
```

Посмотреть список заблокированных пакетов можно командой:

```
# yum versionlock list
```

Больше nginx не будет автоматически обновляться через yum. Вы сможете сами по мере необходимости готовить свои пакеты с ним и устанавливать вручную. В идеале, конечно, лучше настроить свой репозиторий. Но это уже тема отдельного разговора.

Заключение

> Не понравилась статья и хочешь научить меня администрировать? Пожалуйста, я люблю учиться. Комментарии в твоем распоряжении. Расскажи, как сделать правильно!

Я описал один маленький элемент в ускорении сайта. Планирую написать цикл статей на эту тему. Материал уже накопился. Серверная часть дает незначительный прирост, по сравнению с оптимизацией кода и базы, но тем не менее, ей тоже стоит заниматься.

Так же рекомендую свою статью для тех, кто интересуется более тонкой и осмысленной настройкой nginx. В статье поделился своим опытом и личными примерами.