GeekBrains

Основные сервисы на Linux для предприятия

# PKI и OpenVPN

Рассмотрим инфраструктуру PKI и базовую настройку OpenVPN-сервера

## Оглавление

# Криптография и PKI

Криптография используется, чтобы защищать сообщения от посторонних читателей и хранить секреты. С этой целью применялись различные методы, которые зависели от доступных в то время технологий. Сейчас постоянно растущие высокопроизводительные вычислительные системы, объединённые в сеть, создали новую парадигму. Это усложнило обеспечение безопасности данных и потребовало новой парадигмы информационной безопасности.

Основная концепция шифрования проста. Криптография пытается взять незашифрованные (открытый текст) данные и математически манипулировать ими для создания зашифрованного текста. Использование новых методов шифрования обеспечивает высокие шансы сохранить конфиденциальность и целостность данных.

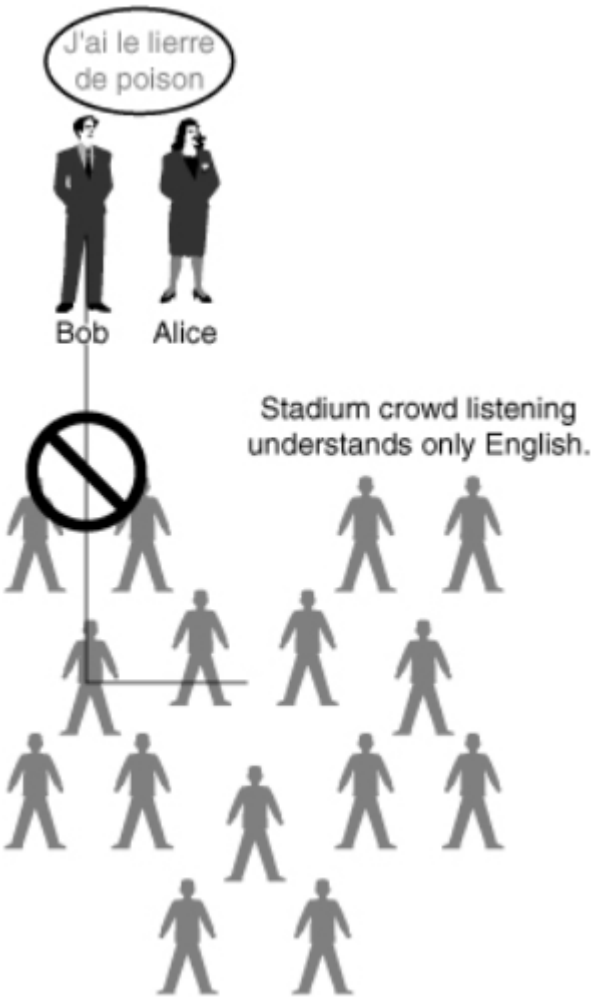
## Конфиденциальность (Confidentiality)

У компаний, военных и финансовых учреждений и даже конечного пользователя, сидящего за компьютером, есть данные, которые должны оставаться конфиденциальными. Обеспечение их конфиденциальности позволяет компаниям сохранить корпоративную тайну, военным — выиграть сражение или войну, а финансовым учреждениям — сохранить деньги либо обезопасить персональные данные пользователей.

Конфиденциальность — часть вашей повседневной жизни. Каждый раз, когда вы используете банкомат для снятия денег, вы, вероятно, оглядываетесь через плечо, чтобы убедиться, что никто не смотрит, когда вы вводите PIN-код. Даже в нетехнических сферах жизни существуют личные вещи, которые мы не хотим обнародовать.

Мы можем захотеть поделиться этими личными подробностями о жизни только с кем-то вроде близкого друга или члена семьи. Это подводит нас к цели конфиденциальности — предоставить информацию только тем людям, с которыми вы хотите поделиться ею, и сделать её недоступной для других. Хороший пример — разговор двух иностранцев на их родном языке за рубежом. Насколько бы громко они ни говорили, если люди вокруг не понимают их язык, конфиденциальность их разговора сохраняется.

Предположим, Алиса и Боб (только два человека на стадионе) говорят по-французски. Любое объявление на французском поймут только они, если другие люди попытаются слушать, они, вероятно, не смогут перевести, что было сказано. Конфиденциальность разговора сохраняется.




Сохранение конфиденциальности для некоторых компаний имеет юридические и финансовые последствия. Если финансовое учреждение не выполнит обозначенные регулятором условия, например, обозначенные в стандарте PCI-DSS (Payment Card Industry Data Security Standard), то у него могут отозвать разрешение на обработку платежей кредитных карт.

## Целостность данных (Integrity)

Цель сохранения и проверки целостности данных в криптографии проста — сохранить исходное сообщение не модифицированным. Пока сообщение находится в пути, оно может быть изменено, даже оставаясь при этом зашифрованным.

Проверка целостности позволяет обнаруживать попытку или изменение исходного сообщения. Вы можете использовать различные методы для проверки целостности, но один из самых часто используемых методов — хеширование исходного сообщения, а затем шифрование хеша вместе с сообщением. Если сообщение было каким-либо образом изменено, дешифрованный хеш не будет соответствовать новому. Следовательно, нарушение целостности исходного сообщения будет обнаружено.

## Подлинность и неопровержимость (Authenticity and Nonrepudiation)

Подлинность и неопровержимость включают в себя идентификацию отправителя как способ защиты данных. То есть, каким-то образом нужно доказать, что отправитель — именно тот, кого мы ожидаем увидеть. Это может быть сделано при помощи аутентификации отправителя, например паролем, либо при помощи PKI. В PKI это достигается с помощью сертификатов.

Подлинность и неопровержимость — одна из ключевых задач большинства систем сертификатов, для её решения может быть использована цифровая подпись доверенной третьей стороны. Как правило, цифровая подпись создается с использованием пар асимметричных ключей. Отправитель «подписывает» сообщение. Сертификат предоставляет доверенная третья сторона, которая ручается за подлинность, «заверяя» подпись отправителя.

## Симметричное шифрование

Симметричное шифрование — форма изменения обычного текста для шифрования при использовании общего ключа (секрета). Общий секрет известен только отправителю и получателю, и обе стороны используют один и тот же ключ для шифрования и дешифрования.

Эта форма шифрования имеет наибольшую историю применения и широкое распространение. Вы можете использовать несколько шифров, включая прямые замены и преобразования, но у всех подходов к симметричному шифрованию один общий элемент: использование общего секрета, который и место уязвимости. Цель взлома этой формы шифрования — получение общего ключа или общего секрета. Кроме того, в интернете участники сетевого взаимодействия не знают друг о друге до момента его начала. Следовательно, они должны создать общий секрет через небезопасную среду. Для этого используются безопасные механизмы обмена общим секретом, один из них — механизм Диффи-Хеллмана.

Преимущество использования симметричного шифрования — в его фундаментальной простоте, а, следовательно, и производительности. Сила шифрования зависит от длины ключа для любого данного алгоритма, поэтому чем длиннее общий секрет, тем дольше злоумышленник может его расшифровывать.

Основной недостаток симметричного шифрования — распространение общего секрета. Чтобы эта форма шифрования работала, все стороны должны знать общий ключ. То есть, во-первых, требуется безопасный способ доставки общего секрета. Без предоставления общего секрета всем сторонам шифрование или дешифрование невозможно. Вторая проблема связана с безопасностью, которая зависит от знания общего секрета. Если общий секретный код скомпрометирован, вся дальнейшая

передача данных будет скомпрометирована. Один из подходов к решению этой проблемы — использование механизма Диффи-Хеллмана.

Примеры современных механизмов симметричного шифрования: AES, CHACHA20, Кузнечик.

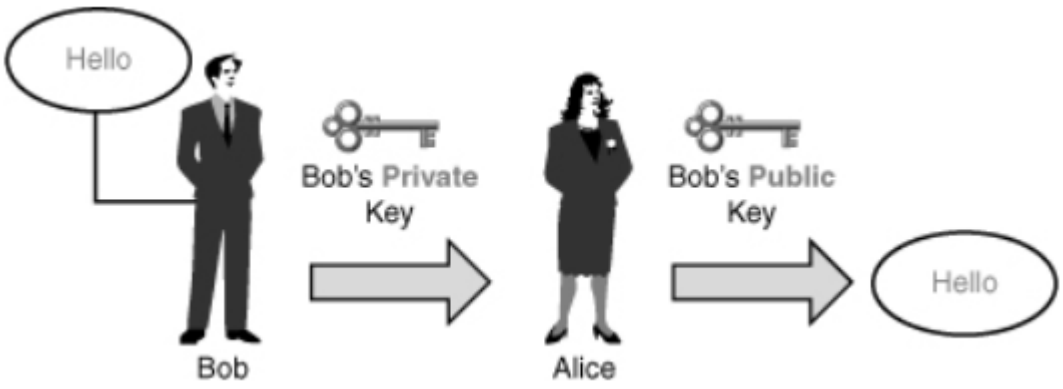
# Асимметричное шифрование

В отличие от симметричного шифрования, в котором используется общий секрет, асимметричное шифрование задействует сразу несколько ключей. У участника взаимодействия есть пара ключей: открытый и закрытый. Открытый ключ доступен каждому в мире и распространяется открыто. Закрытый ключ скрыт и известен только создателю пары ключей. Закрытый ключ, который известен только отправителю, — критически важное условие безопасного и надёжного асимметричного шифрования.

Преобразование простого текста в зашифрованный в ассиметричном шифровании использует другую парадигму, в отличие от симметричного. В зависимости от цели передачи данных, открытый или закрытый ключ может шифровать данные. Получатель использует противоположный ключ для расшифровки данных. Рассмотрим два основных направления, в которых используют ассиметричное шифрование.

## Аутентификация

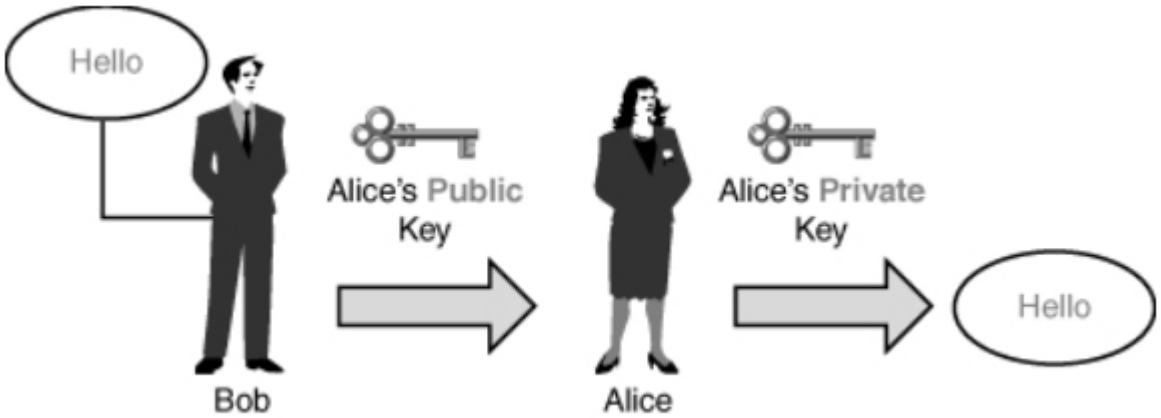
Аутентификация — основное применение большинства решений PKI. Цель аутентификации — не обеспечение конфиденциальности информации, а проверка подлинности (личности) отправителя. Аутентификация в асимметричном шифровании использует закрытый и открытый ключи отправителя. Если какое-либо сообщение зашифровано с использованием личного ключа отправителя (подписано отправителем), получатель может использовать открытый ключ отправителя для его расшифровки. Предполагается, что единственный пользователь с закрытым ключом — отправитель. Если сообщение успешно расшифровано получателем с открытым ключом, значит, его прислал именно этот отправитель и проверка подлинности прошла успешно. Цифровые подписи используют такую схему аутентификации:



Например,  Боб шифрует сообщение «Hello» своим закрытым ключом. Поскольку Алиса может расшифровать сообщение с помощью открытого ключа Боба, только он и мог отправить это сообщение. Таким образом, сообщение и его отправитель были аутентифицированы.

### Шифрование

Шифрование с использованием асимметричной схемы, как и аутентификация, использует открытый и закрытый ключи. Отправитель получает открытый ключ получателя. Затем отправитель шифрует сообщение с помощью открытого ключа получателя. Предполагается, что закрытый ключ стороны известен только получателю; то есть получатель — единственный участник взаимодействия — обладает закрытым ключом. Следовательно, когда он получает копию зашифрованного сообщения, он и только он может расшифровать его, используя этот закрытый ключ из пары.




Например, Боб отправляет Алисе сообщение «Hello». Он использует открытый ключ Алисы для шифрования сообщения. Поскольку только у Алисы есть закрытый ключ, она единственная, кто может расшифровать сообщение. Конфиденциальность сообщения между Алисой и Бобом обеспечивается.

Преимущество такого шифрования — его доступность всем участникам. Вы можете опубликовать открытую часть ключа где угодно, и вам не нужны дополнительные механизмы (например, механизм Диффи-Хеллмана) для обмена ключами.

Недостаток — сложность самих алгоритмов шифрования и, следовательно, более активное потребление ресурсов по сравнению с симметричным шифрованием.

Пример механизмов асимметричного шифрования: RSA, DSA.

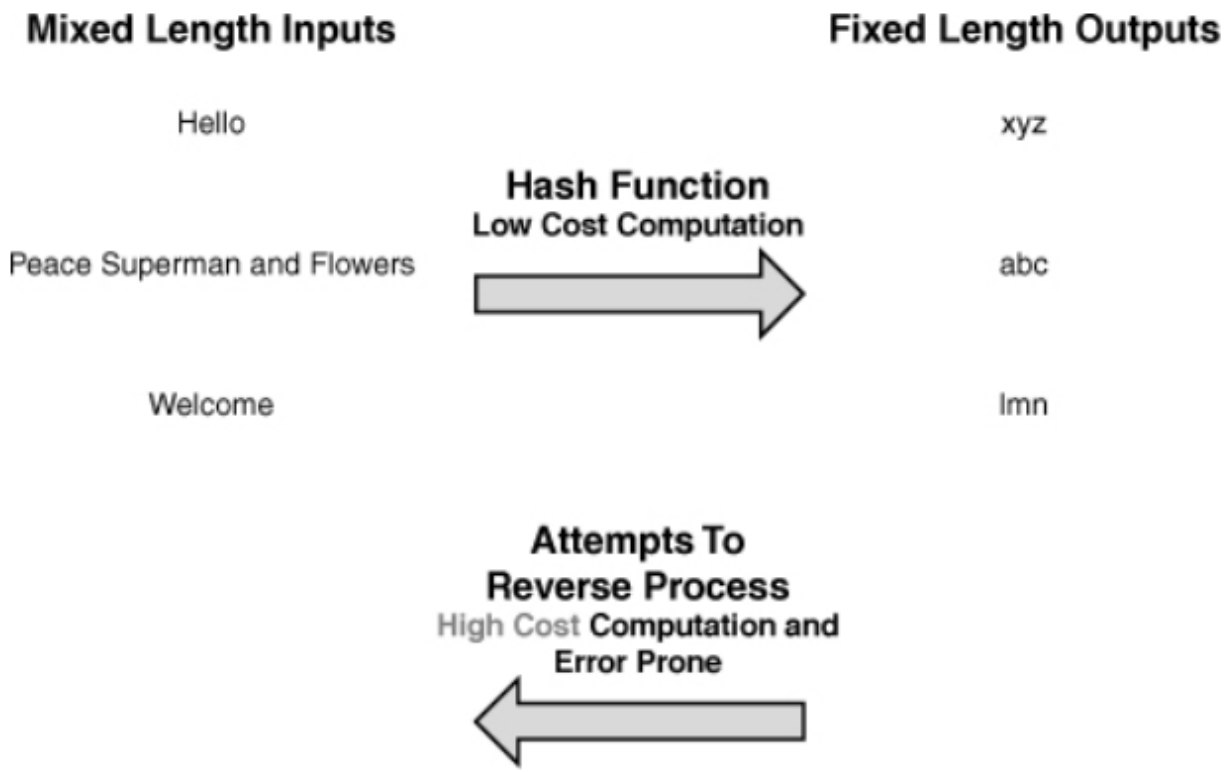
## Хеш-функция

Две важнейшие вспомогательные функции для криптографии — хеши и цифровые подписи.

## Хеш (Hash)

Возможность взять строку любой длины и преобразовывать её в уникальную строку фиксированной длины — полезный механизм в криптографии. Смысл механизма в том, что создание такой строки — очень простое вычисление, а вот обратное преобразование — крайне ресурсоемкая задача. Кроме того, получившийся в результатах расчёта хеш необратим. То есть, зная само значение хеша, невозможно из него получить исходные данные.

Например, на рисунке хеши принимают входные данные переменной длины и создают псевдослучайные выходные данные фиксированной длины.




Из-за необратимости и фиксированной длины хеш-функции используются в качестве методов проверки целостности сообщения

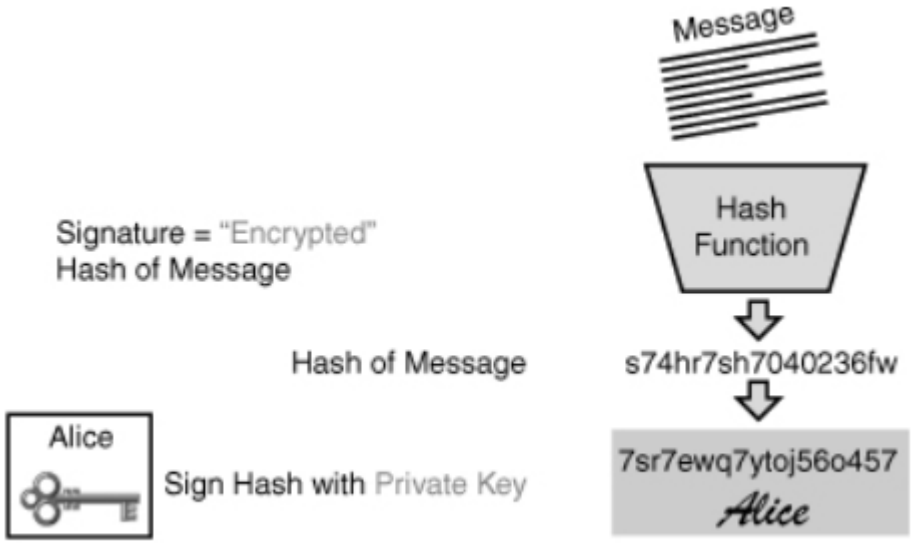
## Цифровая подпись (Digital Signature)

Отправитель использует цифровые подписи для аутентификации сообщения. Цифровые подписи используют асимметричную криптографию; в частности, они основаны на методе аутентификации в асимметричной криптографии и работают в двух разных направлениях: создание цифровой подписи и проверка цифровой подписи. Ниже приведены шаги в создании подписи:

1. Отправитель создаёт сообщение.

2. Берётся хеш  от этого сообщения.
3. Этот хеш зашифрован с помощью закрытого ключа отправителя.
4. Зашифрованный хеш, цифровая подпись, отправляется с исходным сообщением.

Например, цифровая подпись берёт копию сообщения, хеширует его и затем шифрует с помощью закрытого ключа отправителя. Закрытый ключ  используется не для конфиденциальности, а для аутентификации.




Получатель такого сообщения производит проверку подписи:

- Зашифрованный хеш отделяется от исходного сообщения.
- Берётся  хеш из исходного сообщения.
- Зашифрованный хеш расшифровывается с помощью открытого ключа отправителя.
- Расшифрованный хеш сравнивается с хешем исходного сообщения.
- Если оба хеша одинаковы, проверяется подпись и, следовательно, личность отправителя.

Таким образом, получается, что цифровая подпись — это хеш сообщения, зашифрованный с помощью закрытого ключа отправителя. Подпись должна быть проверена для подтверждения личности отправителя — это и делает получатель, который расшифровывает подпись с помощью открытого ключа отправителя, создает хеш исходного сообщения и сравнивает оба. Если они одинаковы, личность отправителя подтверждена.

# Public Key Infrastructure

## Цифровой сертификат

В цифровом мире сертификаты связывают  айдентити (отдельное лицо или устройство любого типа) и соответствующим электронным материалом. В нашем случае материал представляет собой

криптографическую информацию, то есть ключи шифрования. Получается, что цифровой сертификат представляет собой набор изначально независимой цифровой информации, объединённой и подписанной уполномоченным органом. Этот орган имеет определённый уровень доверия со стороны пользователей, например, это правительственные организации или некоторые компании.

## Структура и содержание цифрового сертификата.

Основная информация, содержащаяся в сертификате, следующая:

- Айдентити — имя или ссылка на объект, человека или материал.
- Некоторые атрибуты, прикреплённые либо к самой личности, либо к разрешённому использованию сертификата.
- Открытый ключ выполняет заданный набор криптографических операций. Он включён в сертификат, так что он публикуется одновременно и, следовательно, становится доступным для третьих лиц. Как мы помним, в открытой части ключа нет секрета, поэтому такой подход безопасен.
- Подпись от удостоверяющего центра (CA), охватывающая все предшествующие информационные поля. Подпись аутентифицирует связь между информацией, содержащейся в пакете.

Наряду с этими четырьмя элементами, в сертификат может быть включена дополнительная информация — для повышения безопасности либо автоматизации операций внутри PKI.

Типичный сертификат имеет следующую структуру:

- Сертификат.
- Версия.
- Серийный номер.
- Идентификатор алгоритма.
- Эмитент.
- Срок годности.
    - Не раньше, чем...
    - Не после...
- Имя субъекта.
- Информация об открытом ключе субъекта.
    - Алгоритм с открытым ключом.
    - Открытый публичный ключ.
- Уникальный идентификатор эмитента (необязательно, версия 2 и выше).
- Уникальный идентификатор субъекта (необязательно, версия 2 и выше).
- Расширения (необязательно, версия 3 и выше).
    - ...
- Алгоритм подписи сертификата.

- Сертификат подписи.

Библиотека OpenSSL предлагают удобные инструменты для работы и просмотра сертификатов. Например, мы можем проверить сертификат любого ресурса:

```
# openssl s_client -showcerts -connect yandex.ru:443
CONNECTED(00000003)
depth=2 C = PL, O = Unizeto Technologies S.A., OU = Certum Certification
Authority, CN = Certum Trusted Network CA
verify return:1
depth=1 C = RU, O = Yandex LLC, OU = Yandex Certification Authority, CN = Yandex
CA
verify return:1
depth=0 CN = yandex.ru, O = Yandex LLC, OU = ITO, L = Moscow, ST = Russia, C =
RU
verify return:1
---
Certificate chain
 0 s:/CN=yandex.ru/O=Yandex LLC/OU=ITO/L=Moscow/ST=Russia/C=RU
   i:/C=RU/O=Yandex LLC/OU=Yandex Certification Authority/CN=Yandex CA
-----BEGIN CERTIFICATE-----
MIILlzCCCn+gAwIBAgIQEzNmdWqsjo6ZWBhxvbod4DANBgkqhkiG9w0BAQsFADBf
MQswCQYDVQQGEwJSVTETMBEGA1UEChMKWWFuZGV4IExMQzEnMCUGA1UECxMeWWFu
ZGV4IENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRIwEAYDVQQDEwlZYW5kZXggQ0Ew
HhcNMTkwOTA1MTExMjM0WhcNMjAwOTA0MTExMjM0WjBmMRIwEAYDVQQDDAl5YW5k
ZXgucnUxEzARBgNVBAoMCllhbmRleCBMTEMxDDAKBgNVBAsMA0lUTzEPMA0GA1UE
BwwGTW9zY293MQ8wDQYDVQQIDAZSdXNzaWExCzAJBgNVBAYTAlJVMFkwEwYHKoZI
zj0CAQYIKoZIzj0DAQcDQgAEI82TnU+s6jNrxuWTPm67XwjFuXS1I6I5vjO1Jhok
n0/jWrD7l4/OMpIf36URXqAmQgsGgy2IjLj3SK1AoV4u3aOCCREwggkNMAwGA1Ud
EwEB/wQCMAAwaQYDVR0fBGIwYDAvoC2gK4YpaHR0cDovL2NybHMueWFuZGV4Lm5l
dC9jZXJ0dW0veWNhc2hhMi5jcmwwLaAroCmGJ2h0dHA6Ly95YW5kZXguY3JsLmNl
cnR1bS5wbC95Y2FzaGEyLmNybDBxBggrBgEFBQcBAQRlMGMwLAYIKwYBBQUHMAGG
IGh0dHA6Ly95YW5kZXgub2NzcC1yZXNwb25kZXIuY29tMDMGCCsGAQUFBzAChido
dHRwOi8vcmVwb3NpdG9yeS5jZXJ0dW0ucGwveWNhc2hhMi5jZXIwHwYDVR0jBBgw
FoAUN1zjGeCyjqGoTtLPq9Dc4wtcNU0wHQYDVR0OBBYEFIhadSmxLCL8t9xzmoJl
VEhPlgmwMEwGA1UdIARFMEMwCAYGZ4EMAQICMDcGDCqEaAGG9ncCBQEKAjAnMCUG
CCsGAQUFBwIBFhlodHRwczovL3d3dy5jZXJ0dW0ucGwvQ1BTMB0GA1UdJQQWMBQG
CCsGAQUFBwMBBggrBgEFBQcDAjAOBgNVHQ8BAf8EBAMCB4AwggXdBgNVHREEggXU
MIIF0IIKeWFuZGV4LmNvbYIJeWFuZGV4LmJ5ggp5YW5kZXgubmV0ggl5YW5kZXgu
a3qCCXlhbmRleC51YYIJeWFuZGV4LnJ1gg15YW5kZXguY29tLnRyghp3d3cueG4t
LWQxYWNwangzZi54bi0tcDFhaYIYbS54bi0tZDFhY3BqeDNmLnhuLS1wMWFpghZ4
bi0tZDFhY3BqeDNmLnhuLS1wMWFpghN4bWxzZWFyY2gueWFuZGV4LmJ5ghBwZW9w
bGUueWFuZGV4LmJ5gg92aWRlby55YW5kZXguYnmCEGltYWdlcy55YW5kZXguYnmC
DXd3dy55YW5kZXguYnmCDXd3dy55YW5kZXguZnKCC20ueWFuZGV4LmZyggl5YW5k
ZXguZnKCC20ueWFuZGV4LmVlggl5YW5kZXguZWWCEHBlb3BsZS55YW5kZXgua3qC
C20ueWFuZGV4Lmt6ghN4bWxzZWFyY2gueWFuZGV4Lmt6gg13d3cueWFuZGV4Lmt6
gg92aWRlby55YW5kZXgua3qCFGZhbWlseS55YW5kZXguY29tLnRygg9tLnlhbmRl
eC5jb20udHKCFHBlb3BsZS55YW5kZXguY29tLnRyghJhaWxlLnlhbmRleC5jb20u
dHKCFGltYWdlcy55YW5kZXguY29tLnRygg13d3cueWFuZGV4LmtnghF3d3cueWFu
ZGV4LmNvbS5hbYILbS55YW5kZXgubWSCCXlhbmRleC5tZIILbS55YW5kZXgubHaC
EGltYWdlcy55YW5kZXgudWGCEmdhbWUueWFuZGV4LmNvbS50coITdmlkZW8ueWFu
ZGV4LmNvbS50coINd3d3LnlhbmRleC51YYINd3d3LnlhbmRleC51eoIPbS55YW5k
```

```
ZXguY29tLmdlgg13d3cueWFuZGV4Lm1kggttLnlhbmRleC5ydYILbS55YW5kZXgu
dG2CDm0ueWFuZGV4LmNvLmlsggx5YW5kZXguY28uaWyCDXd3dy55YW5kZXguZWWC
CXlhbmRleC5sdIILbS55YW5kZXgudXqCEXd3dy55YW5kZXguY29tLmdlgg13d3cu
eWFuZGV4LmF6ghB3d3cueWFuZGV4LmNvLmlsggttLnlhbmRleC5rZ4IRd3d3Lnlh
bmRleC5jb20udHKCEnBsYXkueWFuZGV4LmNvbS50coIPbS55YW5kZXguY29tLmFt
ghRnb3JzZWwueWFuZGV4LmNvbS50coIJeWFuZGV4Lmx2ghBwZW9wbGUueWFuZGV4
LnJ1ggttLnlhbmRleC51YYIQcGVvcGxlLnlhbmRleC51YYIJeWFuZGV4LmF6gg92
aWRlby55YW5kZXgucnWCDG0ueWFuZGV4LmNvbYIXeG1sc2VhcmNoLnlhbmRleC5j
b20udHKCDXd3dy55YW5kZXgubHaCEGltYWdlcy55YW5kZXgucnWCE3htbHNlYXJj
aC55YW5kZXgudWGCDXd3dy55YW5kZXgudG2CDnd3dy55YW5kZXguY29tgg13d3cu
eWFuZGV4Lmx0ggl5YW5kZXgudG2CDXd3dy55YW5kZXgucnWCFHhtbHNlYXJjaC55
YW5kZXguY29tgg15YW5kZXguY29tLmdlghNnYW1lcy55YW5kZXguY29tLnRyghN4
bWxzZWFyY2gueWFuZGV4LnJ1ggttLnlhbmRleC5ieYIJeWFuZGV4LnRqgg92aWRl
by55YW5kZXgudWGCDXd3dy55YW5kZXgudGqCC20ueWFuZGV4Lmx0ghBpbWFnZXMu
eWFuZGV4Lmt6ghFwZW9wbGUueWFuZGV4LmNvbYISb3l1bi55YW5kZXguY29tLnRy
ggttLnlhbmRleC5heoINeWFuZGV4LmNvbS5hbYIQdmlkZW8ueWFuZGV4LmNvbYIR
aW1hZ2VzLnlhbmRleC5jb22CCXlhbmRleC5rZ4IJeWFuZGV4LnV6ggttLnlhbmRl
eC50ajCCAX8GCisGAQQB1nkCBAIEggFvBIIBawFpAHcApLkJkLQYWBSHuxOizGdw
Cjw1mAT5G9+443fNDsgN3BAAAAFtARvLPgAABAMASDBGAiEA2eu2MHUwKhvlVzVq
nzCD7xnZRbxUDLl/aW3ESd9lb/wCIQDT+VwK5G3+0/8j8TXsWdoDbC1uyEWpoqrX
MiksHnxJXAB2AFWB1MIWkDYBSuoLm1c8U/DA5Dh4cCUIFy+jqh0HE9MMAAABbQEb
zNQAAAQDAEcwRQIhAOpKP3IEWNRzIvltBAT7xmd9EqpKr+vFMCSxMsLJ5dX4AiBz
A7PGv8+O3rfhxh4JmBXoJNNW3fDGAajOPRTqZNr8+AB2AF6nc/nfVsDntTZIfdBJ
4DJ6kZoMhKESEoQYdZaBcUVYAAABbQEby5QAAAQDAEcwRQIhAKnrUMyjFrtWPvTs
CCPHyosSxuNmLk0MVfEbLDqZ9YHBAiBdttwYKFDN1ZxtMxqo1tLhtCfp0dhRh9/X
PHZ8O+4xHzANBgkqhkiG9w0BAQsFAAOCAQEAK0w3Vs5KCWVERxFwuiMALR5n3r5e
kJA3OmwFjUINqa3hkpL8qjMKgLMQpdnkQSJoAWGLvXbOJ9ypJRv14qeN55lE55rI
T53Y5N1CPyqy1pUqXRMKZLwcXUkao1ZaRtg6wQh+8sOYg7/9psgNNXTdQOI5RPs7
1hyx+N7b3Gh+YAe82aC5WdLJU2LvGSSgPaqrcZS91xlEgVlNx0TI0RLuhPXunytN
DhtLXBAI1tZzs5mcAMTGqMAb/2c2EpXUWz00B3hi9fvwWIVHXkrjCsj4aEAa2Xp8
mFPRVsOLaEF3vdiSB1J9vnPTAG2mjsoRv1RpMqg8jL8enL1BLGW4cd13kw==
-----END CERTIFICATE-----
 1 s:/C=RU/O=Yandex LLC/OU=Yandex Certification Authority/CN=Yandex CA
            i:/C=PL/O=Unizeto    Technologies    S.A./OU=Certum    Certification
Authority/CN=Certum Trusted Network CA
-----BEGIN CERTIFICATE-----
MIIEqDCCA5CgAwIBAgIRAOQFR4MODGRSl296NUnA3UgwDQYJKoZIhvcNAQELBQAw
fjELMAkGA1UEBhMCUEwxIjAgBgNVBAoTGVVuaXpldG8gVGVjaG5vbG9naWVzIFMu
QS4xJzAlBgNVBAsTHkNlcnR1bSBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0eTEiMCAG
A1UEAxMZQ2VydHVtIFRydXN0ZWQgTmV0d29yayBDQTAeFw0xNTAxMjExMjAwMDBa
Fw0yNTAxMTgxMjAwMDBaMF8xCzAJBgNVBAYTAlJVMRMwEQYDVQQKEwpZYW5kZXgg
TExDMScwJQYDVQQLEx5ZYW5kZXggQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkxEjAQ
BgNVBAMTCVlhbmRleCBDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AKYFJHZhuZ5CYCJjhVnlnYgN3+8hZFomlHE6pH8rU8Ose7qVQm1qW9Z+eAxnQJgv
ai3Qtxg6fplgAeUnv/9J9c3EWMNM4XDV/QioeZV2HA4FQfq9gDgqh0/BZ0KqF6bu
p4yO7y1/eh0FF49+O5I19WjtkwNVI09LogCGZZEP6/Y81dttDu3ofDrIurdTwaTY
QALltaLKv9qclA38xRwqWYhiV5MuEfA4LHqBKvIlFRc1cCxL9yNMgu8zn8KaC6Pi
XWs4d/lgM88ue1a3E5MfNJdxmXYCRjUUfNzKSIoKckt4bYI0lhNFzwIvUBM5Q4nA
4XTXKHEh5aqXDu5G7JP3I30CAwEAAaOCAT4wggE6MA8GA1UdEwEB/wQFMAMBAf8w
HQYDVR0OBBYEFDdc4xngso6hqE7Sz6vQ3OMLXDVNMB8GA1UdIwQYMBaAFAh2zcsH
/yT2xc3tu5C84oQ3RnX3MA4GA1UdDwEB/wQEAwIBBjAvBgNVHR8EKDAmMCSgIqAg
hh5odHRwOi8vY3JsLmNlcnR1bS5wbC9jdG5jYS5jcmwwawYIKwYBBQUHAQEEXzBd
```

```
MCgGCCsGAQUFBzABhhxodHRwOi8vc3ViY2Eub2NzcC1jZXJ0dW0uY29tMDEGCCsG
AQUFBzAChiVodHRwOi8vcmVwb3NpdG9yeS5jZXJ0dW0ucGwvY3RuY2EuY2VyMDkG
A1UdIAQyMDAwLgYEVR0gADAmMCQGCCsGAQUFBwIBFhhodHRwOi8vd3d3LmNlcnR1
bS5wbC9DUFMwDQYJKoZIhvcNAQELBQADggEBAAJejnvgZqHGq4sYHw65xM1x20Rc
A31l6rhHtR7OJHCgf9PfZkuMkOKl7ZuUNrSovvB0jCaSdZ1WUJ6t0Bqg36QUVhB1
k3rB9FOgdnQscrq10cni3EaGPx32M4dZ7JzcLR5NQxrOutmHfuJHRXI9KAPJCk3g
V6Nebn7MWsjEeAFXaHo4O1M255Jtiiwv14u2NKjRtvheO6vtpY85b0Wty2PtamTJ
EKcDCBJTsRyvyvdT/NgpSxv7OM3AY/9f5LmNXqor0sMiNTH2MA5TMvSTxUPLyPAV
Vo8AGYfKeCKNoC7bL6DDfildkSWEHR05qxvF1pH+aQ5GgLxFezVTKt8Atnc=
-----END CERTIFICATE-----
    2    s:/C=PL/O=Unizeto    Technologies    S.A./OU=Certum    Certification
Authority/CN=Certum Trusted Network CA
   i:/C=PL/O=Unizeto Sp. z o.o./CN=Certum CA
-----BEGIN CERTIFICATE-----
MIIEtDCCA5ygAwIBAgIRAJOShUABZXFflH8oj+/JmygwDQYJKoZIhvcNAQELBQAw
PjELMAkGA1UEBhMCUEwxGzAZBgNVBAoTElVuaXpldG8gU3AuIHogby5vLjESMBAG
A1UEAxMJQ2VydHVtIENBMB4XDTA4MTAyMjEyMDczN1oXDTI3MDYxMDEwNDYzOVow
fjELMAkGA1UEBhMCUEwxIjAgBgNVBAoTGVVuaXpldG8gVGVjaG5vbG9naWVzIFMu
QS4xJzAlBgNVBAsTHkNlcnR1bSBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0eTEiMCAG
A1UEAxMZQ2VydHVtIFRydXN0ZWQgTmV0d29yayBDQTCCASIwDQYJKoZIhvcNAQEB
BQADggEPADCCAQoCggEBAOP7faNyusLwyRSH9WsBTuFuQAe6bSddf/dbLbNax1Ff
q6QypmGHtm4PhtIwApf412lXoRg5XWpkecYBWaw8MUo4fNIE0kso6CBfOweizE1z
2/OuT8dW1Vqnlon686to1COGWSfPCSe8rG5ygxwwct/gounS4XR1Gb0qnnsVVAQb
10M5rVUoxeIau/TA5K44STPMdoWfOUXSpJ7yEoxR+HzkLX/1rF/rFp+xLdG6zJFC
d0wlyZA4b9vwzPuOHpdZPtVgTuYFKO1JeRNLukjbL/ly0znK/h/YNHL1tEDPMQHD
7N4RLRddH7hQ0V4Zp2neBzMoylCV+adUy1SGUEWp+UkCAwEAAaOCAWswggFnMA8G
A1UdEwEB/wQFMAMBAf8wHQYDVR0OBBYEFAh2zcsH/yT2xc3tu5C84oQ3RnX3MFIG
A1UdIwRLMEmhQqRAMD4xCzAJBgNVBAYTAlBMMRswGQYDVQQKExJVbml6ZXRvIFNw
LiB6IG8uby4xEjAQBgNVBAMTCUNlcnR1bSBDQYIDAQAgMA4GA1UdDwEB/wQEAwIB
BjAsBgNVHR8EJTAjMCGgH6AdhhtodHRwOi8vY3JsLmNlcnR1bS5wbC9jYS5jcmww
aAYIKwYBBQUHAQEEXDBaMCgGCCsGAQUFBzABhhxodHRwOi8vc3ViY2Eub2NzcC1j
ZXJ0dW0uY29tMC4GCCsGAQUFBzAChiJodHRwOi8vcmVwb3NpdG9yeS5jZXJ0dW0u
cGwvY2EuY2VyMDkGA1UdIAQyMDAwLgYEVR0gADAmMCQGCCsGAQUFBwIBFhhodHRw
Oi8vd3d3LmNlcnR1bS5wbC9DUFMwDQYJKoZIhvcNAQELBQADggEBAI3m/UBmo0yc
p6uh2oTdHDAH5tvHLeyDoVbkHTwmoaUJK+h9Yr6ydZTdCPJ/KEHkgGcCToqPwzXQ
1aknKOrS9KsGhkOujOP5iH3g271CgYACEnWy6BdxqyGVMUZCDYgQOdNv7C9C6kBT
Yr/rynieq6LVLgXqM6vp1peUQl4E7Sztapx6lX0FKgV/CF1mrWHUdqx1lpdzY70a
QVkppV4ig8OLWfqaova9ML9yHRyZhpzyhTwd9yaWLy75ArG1qVDoOPqbCl60BMDO
TjksygtbYvBNWFA0meaaLNKQ1wmB1sCqXs7+0vehukvZ1oaOGR+mBkdCcuBWCgAc
eLmNzJkEN0k=
-----END CERTIFICATE-----
---
Server certificate
subject=/CN=yandex.ru/O=Yandex LLC/OU=ITO/L=Moscow/ST=Russia/C=RU
issuer=/C=RU/O=Yandex LLC/OU=Yandex Certification Authority/CN=Yandex CA
---
No client certificate CA names sent
Peer signing digest: SHA256
Server Temp Key: ECDH, P-256, 256 bits
---
SSL handshake has read 5851 bytes and written 415 bytes
---
```

```
New, TLSv1/SSLv3, Cipher is ECDHE-ECDSA-AES128-GCM-SHA256
Server public key is 256 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-ECDSA-AES128-GCM-SHA256
    Session-ID: 069E82EC1DB7513A5617846995258C916DDA151563CA25B56D0D359CE674C182
    Session-ID-ctx:
                                                                    Master-Key:
23D28DA1E4CAD69642C403DD4D812463B9C82E0AA375F8B444B9E51D9082DAB710870AFB5110612B
C9B1641E1D13D8FC
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    TLS session ticket lifetime hint: 100800 (seconds)
    TLS session ticket:
    0000 - 23 13 61 be 89 61 da bb-05 35 53 59 57 3a e5 44   #.a..a...5SYW:.D
    0010 - 67 4d 4a 3e 50 d4 69 5b-ac 03 35 c7 d2 35 5c b2   gMJ>P.i[..5..5\.
    0020 - 4f 87 f6 5f 4e 52 dd be-c0 83 d6 c8 24 bd ff b2   O.._NR......$...
    0030 - 98 82 d8 9b 57 6b c1 c9-a8 8b 0f cb e7 4a 99 2d   ....Wk.......J.-
    0040 - 0e 5d 68 0d a7 9f 35 7b-5e d4 85 76 7c f7 f4 7e   .]h...5{^..v|..~
    0050 - 5f 09 69 6f a7 cc 3e 4c-49 c8 b6 b2 9a c6 6c f4   _.io..>LI.....l.
    0060 - 23 2e bc 26 4b 9f 5c 67-ee e0 a3 70 51 d6 21 02   #..&K.\g...pQ.!.
    0070 - 1e b5 13 c0 ab aa 1b 9b-4c 87 a3 eb 83 55 15 fc   ........L....U..
    0080 - 52 6a 12 74 1d 40 0a e8-cf 19 4b a5 8b 77 07 bd   Rj.t.@....K..w..
    0090 - e7 b3 ff c2 33 ea 60 d2-bd 47 3a 29 60 7f 7b 48   ....3.`..G:)`.{H

    Start Time: 1572776432
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
---
```

Состав и поля сертификатов в формате X.509 описан в RFC-5280.

- Version. На сегодняшний день определены три версии стандарта X.509. С 1996 года последняя и наиболее часто используемая — v3. Различия между версиями заключаются в том, что некоторые дополнительные поля оказались необходимыми с течением времени, вызывая обновления в определённых стандартах.
- Serial Number уникально идентифицирует сертификат среди всех выданных данным центром сертификации (CA).
- Algorithm ID — идентификатор алгоритма, используемого CA для подписи сертификата.
- Issuer — информация о конкретном центре сертификации, который выпустил и подписал сертификат.

- Validity Period. Определяет интервал времени, в течение которого CA гарантирует, что он будет хранить информацию о статусе сертификата (действительный он или отозванный). После истечения срока действия сертификат недействителен. Это поле содержит две даты: not before и not after (они определяют временной интервал), выраженные в универсальном времени (UTC / GMT).
- Subject Name. Идентифицирует объект, которому принадлежит открытый ключ. Имя субъекта принимает форму отличительного имени X.500 (Distinguished Name — DN). Типичные структуры DN включают страну (Country — C), организацию (Organisation — O), организационную единицу (Organizational Unit — OU) и общее имя (Common Name — CN), и могут быть организованы иерархическим образом.
- Public Key Algorithm — алгоритм, с помощью которого может быть использован открытый ключ субъекта (то есть RSA или DSA).
- Subject Public Key — открытый ключ, связанный с объектом, указанным в поле «Имя субъекта». Это один из ключей, используемых для криптографических операций.
- Subject Unique Identifier — уникальный идентификатор, связанный с субъектом. Это необязательное поле можно использовать в случае, если несколько объектов (пользователей или устройств) имеют одинаковые имена субъектов.
- Issuer Unique Identifier — ссылка на уникальный идентификатор субъекта, используемый в сертификате CA, если несколько CA имеют схожие имена субъектов.
- Certificate Signature Algorithm — идентификатор алгоритма, используемого CA для подписи сертификата. Примером такого алгоритма может быть SHA256 с шифрованием RSA. Это поле должно содержать тот же идентификатор алгоритма, что и Algorithm ID.
- Certificate Signature — цифровая подпись, вычисленная в полях сертификата, добавленная центром сертификации при его создании. Генерируя эту подпись, CA удостоверяет достоверность информации в полях, в частности, сертифицирует связь между материалом открытого ключа и предметом сертификата. При получении сертификата от пира один из первых шагов — проверка подписи, чтобы убедиться, что он не был подделан.
- Extensions представляет собой способ связать различные атрибуты с пользователем или открытым ключом. Частные расширения могут быть определены в соответствии с требованиями реализации. Каждое расширение помечается как критическое либо некритическое. Если система не может понять или обработать критическое расширение, она должна отклонить сертификат. Между тем некритические расширения могут игнорироваться в подобных ситуациях. Поэтому следует соблюдать осторожность, прежде чем помечать расширение как критическое в развертывании PKI.

Наиболее распространённые стандартные расширения:

- Authority Key Identifier предоставляет способ уникальной идентификации пары ключей, используемой для подписи сертификата. Это полезно, если CA имеет несколько сертификатов.
- Subject Key Identifier предоставляет способ уникальной идентификации сертификатов, которые содержат определенный открытый ключ. Это расширение обязательно для сертификатов CA и необязательно для других.
- Key Usage определяет использование ключа, содержащегося в сертификате. Типичные случаи использования — шифрование, цифровые подписи, подпись CRL и подпись сертификата.
- Subject Alternative Name привязывает айдентити к предмету сертификата. Такими идентификационными данными могут быть адрес электронной почты, DNS-имя или IP-адрес.
- Basic Constraints определяет, является ли предмет сертификата Центром Сертификации (разрешено выдавать дочерние сертификаты) и максимальную глубину действующей цепочки сертификации (включая этот сертификат).
- Extended Key Usage указывает цели, для которых может использоваться сертифицированный открытый ключ, в дополнение или вместо основных целей, указанных в расширении использования ключа. Примеры использования ключей включают проверку подлинности веб-сервера TLS, проверку подлинности веб-клиента TLS, подписание загружаемого исполняемого кода, защиту электронной почты или подпись ответов OCSP.
- CRL Distribution Points определяет, как следует получать информацию CRL. Это поле содержит чаще всего HTTP или LDAP URI.

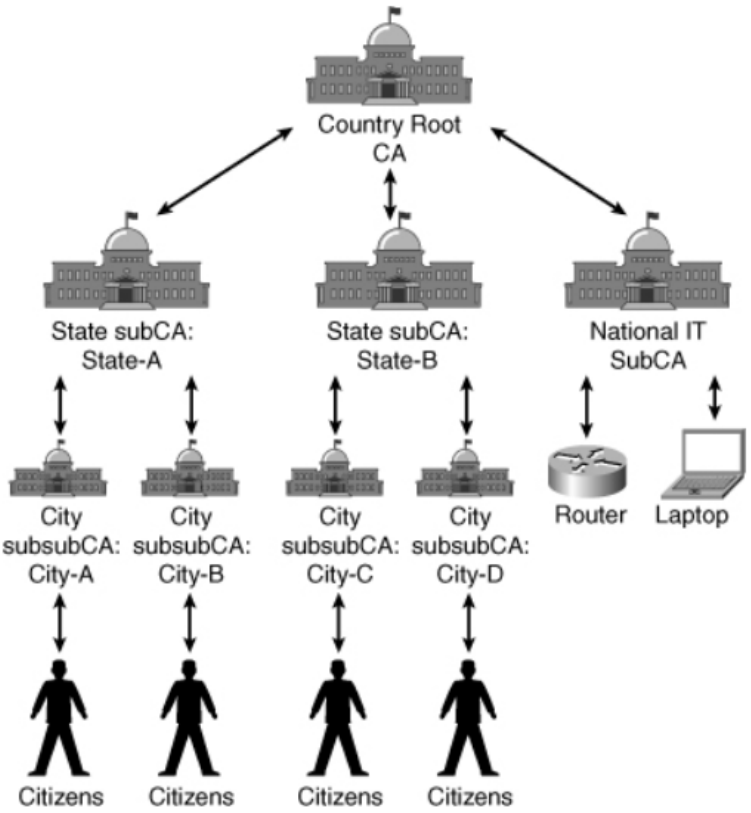
## Удостоверяющий центр (Certification Authority — CA)

Сертификат создаёт связь между айдентити и парой ключей. Но этого недостаточно, чтобы подтвердить валидность отправителя. Поэтому существует третья, внешняя сущность, чтобы официально заявить и гарантировать существование этой связи. Идея состоит в том, чтобы построить доверительные отношения с Удостоверяющим центром (CA) между всеми участниками взаимодействия. Этот третий орган отвечает за проверку подлинности запросов сертификатов, а также за их выдачу.

Например, если кто-то запрашивает сертификат с именем «Алиса», центр сертификации отвечает за проверку того, что на самом деле его запрашивает Алиса (и владеет соответствующими ключами), а не Боб. Чтобы документально подтвердить участие центра сертификации, цифровая подпись CA (созданная с помощью закрытого ключа) добавляется в сертификат в качестве доказательства. Поскольку центр сертификации также доверяет другим членам PKI, проверка подписи CA на сертификате гарантирует его подлинность и, следовательно, личность владельца сертификата.

CA — наиболее важное звено в системе PKI. Если CA был когда-либо скомпрометирован, вся реализация PKI больше не может быть доверенной.

Чтобы обеспечить работоспособность этой системы, CA могут организовывать иерархию:



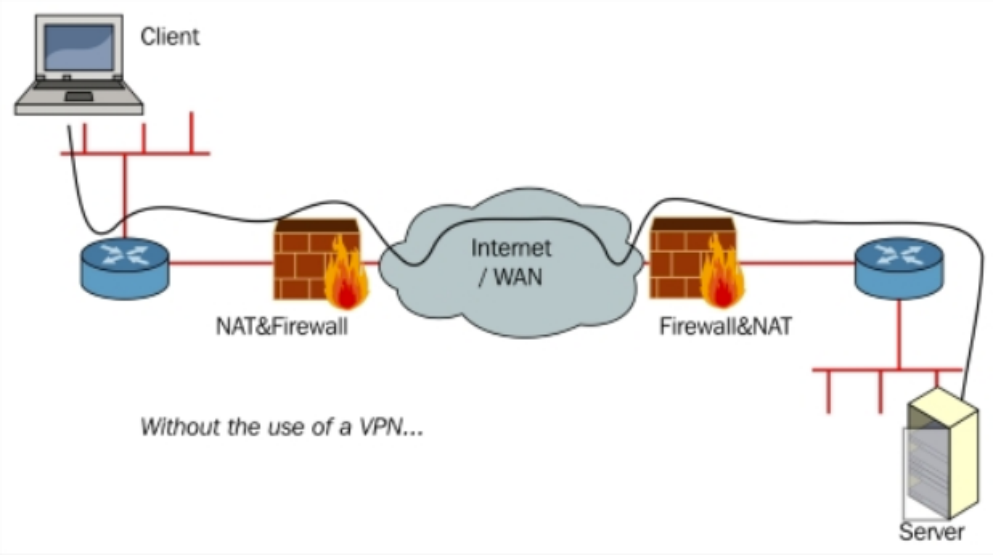

# OpenVPN

Виртуальные частные сети (Virtual Private Network — VPN) были созданы из-за большой потребности в защищённой связи через незащищённую инфраструктуру (интернет). Как вы знаете, сеть Интернет общедоступна и небезопасна. Технически кто угодно может иметь доступ к вашему сетевому взаимодействию с каким-либо ресурсом в интернете.

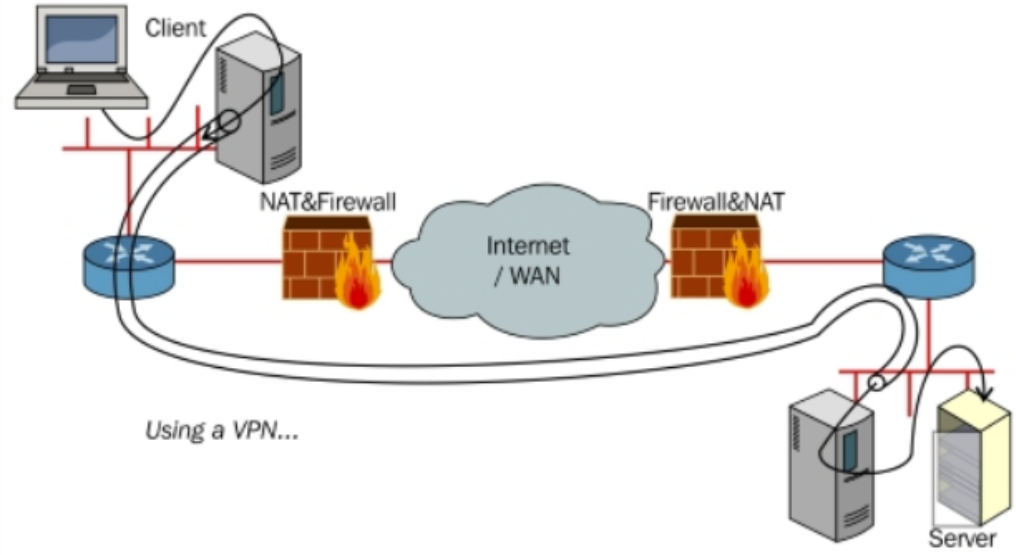
Каждый день большинство людей используют VPN постоянно и целенаправленно, либо неосознанно. Есть много различных технологий VPN — от коммерческих до проектов с открытым исходным кодом. Одна из самых популярных реализаций программного обеспечения VPN с открытым исходным кодом — OpenVPN.

VPN позволяет администратору создавать локальную сеть между несколькими компьютерами в разных сегментах сети. В некоторых случаях эти машины могут находиться в одной и той же локальной сети, они могут быть удалены друг от друга через Интернет. V — значит виртуальная, то есть над физической сетью с коммутаторами, маршрутизаторами и другими устройствами создается виртуальная сеть, которая связывает устройства вместе, как если бы они были подключены друг к другу напрямую или находились в одной локальной сети. Сетевой трафик, проходящий через VPN, часто называют трафиком внутри туннеля (VPN) по сравнению с тем, что за пределами туннеля, так как этот трафик абстрагирован и скрывает некоторые детали взаимодействия от остальных участников сети Интернет.

Без VPN взаимодействие выглядело бы так:




Все устройства в сети знают обо всех деталях взаимодействия между клиентом и сервером. При использовании VPN клиент как бы подключается виртуальным кабелем к VPN серверу. Что логически делает его подключенным к серверу напрямую:




То есть всё взаимодействие изолированно и, зачастую, зашифровано от остальных устройств в сети.

Для установки OpenVPN нам также потребуется утилита easy-rsa для поднятия CA и выписывания сертификатов для клиента и сервера. Сначала обновляем операционную систему и добавляем epel-репозиторий:

```
[root@vpn ~]# sudo yum install epel-release -y
Dependencies Resolved

================================================================================
```

```
 Package             Arch          Version      Repository     Size
================================================================================
Installing:
 epel-release        noarch        7-11         extras         15 k

Transaction Summary
================================================================================
```

Устанавливаем OpenVPN 2.4 и easy-rsa3:

```
[root@vpn ~]# yum install openvpn easy-rsa -y
Dependencies Resolved

================================================================================
 Package            Arch        Version             Repository
                                                                          Size
================================================================================
Installing:
 easy-rsa           noarch      3.0.6-1.el7         epel       36 k
 openvpn            x86_64      2.4.7-1.el7         epel      522 k
Installing for dependencies:
 pkcs11-helper      x86_64      1.11-3.el7          epel       56 k

Transaction Summary
================================================================================
```

Копируем стандартную директорию для easy-rsa в директорию с OpenVPN:

```
[root@vpn ~]# cd /etc/openvpn/
[root@vpn openvpn]# cp -r /usr/share/easy-rsa /etc/openvpn/
[root@vpn openvpn]# cd /etc/openvpn/easy-rsa/3
[root@vpn 3]# ll
total 56
-rwxr-xr-x. 1 root root 48730 Nov  3 14:31 easyrsa
-rw-r--r--. 1 root root  4651 Nov  3 14:31 openssl-easyrsa.cnf
drwxr-xr-x. 2 root root    98 Nov  3 14:31 x509-types
```

Создаём файл vars с настройками для выдачи сертификатов:

```
[root@vpn 3]# vim vars
set_var EASYRSA                 "$PWD"
set_var EASYRSA_PKI             "$EASYRSA/pki"
set_var EASYRSA_DN              "cn_only"
set_var EASYRSA_REQ_COUNTRY     "RU"
set_var EASYRSA_REQ_PROVINCE    "Moscow"
set_var EASYRSA_REQ_CITY        "Moscow"
```

```
set_var EASYRSA_REQ_ORG         "EXAMPLE CERTIFICATE AUTHORITY"
set_var EASYRSA_REQ_EMAIL       "openvpn@example.com"
set_var EASYRSA_REQ_OU          "Example.com EASY CA"
set_var EASYRSA_KEY_SIZE        2048
set_var EASYRSA_ALGO            rsa
set_var EASYRSA_CA_EXPIRE       7500
set_var EASYRSA_CERT_EXPIRE     365
set_var EASYRSA_NS_SUPPORT      "no"
set_var EASYRSA_NS_COMMENT      "EXAMPLE CERTIFICATE AUTHORITY"
set_var EASYRSA_EXT_DIR         "$EASYRSA/x509-types"
set_var EASYRSA_SSL_CONF        "$EASYRSA/openssl-1.0.cnf"
set_var EASYRSA_DIGEST          "sha256"
```

Делаем vars исполняемым файлом:

```
[root@vpn 3]# chmod +x vars
```

Далее, используя файл с переменными, мы можем построить свою собственную инфраструктуру PKI. Ключ nopass генерирует приватные ключи, которые не требуют пароля при обращении с ними. Это хорошая идея для тестирования настроек, но лучше так не делать в реальных имплементациях.

```
[root@vpn 3]# ./easyrsa init-pki

Note: using Easy-RSA configuration from: ./vars

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /etc/openvpn/easy-rsa/3/pki

[root@vpn 3]# ./easyrsa build-ca nopass

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating RSA private key, 2048 bit long modulus
.............+++
.....................................................................................
.....................................................................................
.............................+++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:

CA creation complete and you may now import and sign cert requests.
Your new CA certificate file for publishing is at:
```

```
/etc/openvpn/easy-rsa/3/pki/ca.crt
```

Создаём ключи для сервера:

```
[root@vpn 3]# ./easyrsa gen-req server nopass

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating a 2048 bit RSA private key
..........+++
.............................................+++
writing new private key to
'/etc/openvpn/easy-rsa/3/pki/private/server.key.M4bGp3H93j'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [server]:

Keypair and certificate request completed. Your files are:
req: /etc/openvpn/easy-rsa/3/pki/reqs/server.req
key: /etc/openvpn/easy-rsa/3/pki/private/server.key
```

Подписываем сертификат сервера у удостоверяющего центра:

```
[root@vpn 3]# ./easyrsa sign-req server server

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017


You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 365 days:

subject=
    commonName                = server
```

```
Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /etc/openvpn/easy-rsa/3/pki/safessl-easyrsa.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'server'
Certificate is to be certified until Nov  2 11:42:41 2020 GMT (365 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /etc/openvpn/easy-rsa/3/pki/issued/server.crt
```

Проверяем валидность выписанного сертификата:

```
[root@vpn 3]# openssl verify -CAfile pki/ca.crt pki/issued/server.crt
pki/issued/server.crt: OK
```

Теперь настало время выписать сертификат для клиента:

```
[root@vpn 3]# ./easyrsa gen-req client01 nopass

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating a 2048 bit RSA private key
.....................................................................................
.................................+++
.....................+++
writing               new               private               key               to
'/etc/openvpn/easy-rsa/3/pki/private/client01.key.4tvdUhcrep'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [client01]:

Keypair and certificate request completed. Your files are:
req: /etc/openvpn/easy-rsa/3/pki/reqs/client01.req
key: /etc/openvpn/easy-rsa/3/pki/private/client01.key
```

Подписываем выписанный сертификат у CA:

```
[root@vpn 3]# ./easyrsa sign-req client client01

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017


You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a client certificate for 365 days:

subject=
    commonName                = client01


Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /etc/openvpn/easy-rsa/3/pki/safessl-easyrsa.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'client01'
Certificate is to be certified until Nov  2 11:45:43 2020 GMT (365 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /etc/openvpn/easy-rsa/3/pki/issued/client01.crt
```

Проверяем валидность сертификата:

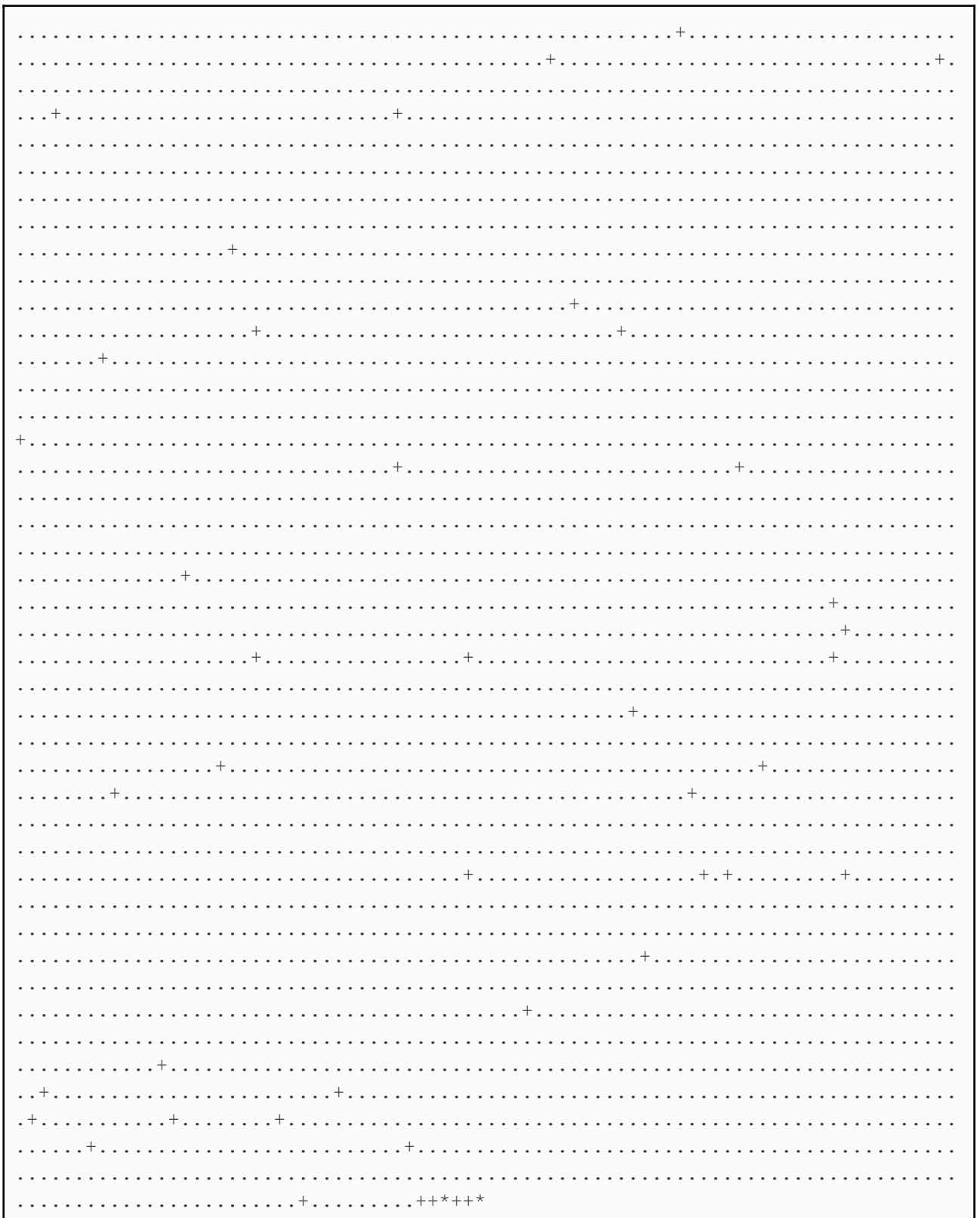
```
[root@vpn 3]# openssl verify -CAfile pki/ca.crt pki/issued/client01.crt
pki/issued/client01.crt: OK
```

Создаём Diffie-Hellman-ключ:

```
[root@centos7 3]# ./easyrsa gen-dh

Note: using Easy-RSA configuration from: ./vars

Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
```



```
DH parameters of size 2048 created at /etc/openvpn/easy-rsa/3/pki/dh.pem
```

По желанию можно создать Certificate Revoking List (CRL) для отзыва сертификатов, но для теста этого не требуется.
Копируем выписанные сертификаты в папку /etc/openvpn/server и /etc/openvpn/client:

```
[root@vpn 3]# pwd
/etc/openvpn/easy-rsa/3
[root@vpn 3]# cp pki/ca.crt /etc/openvpn/server/
[root@vpn 3]# cp pki/issued/server.crt /etc/openvpn/server/
[root@vpn 3]# cp pki/private/server.key /etc/openvpn/server/
[root@vpn 3]# cp pki/ca.crt /etc/openvpn/client/
[root@vpn 3]# cp pki/issued/client01.crt /etc/openvpn/client/
[root@vpn 3]# cp pki/private/client01.key /etc/openvpn/client/
[root@vpn 3]# cp pki/dh.pem /etc/openvpn/server/
```

Создаём конфиг для сервера при помощи файла /etc/openvpn/server.conf:

```
[root@vpn 3]# vim /etc/openvpn/server.conf
# OpenVPN Port, Protocol and the Tun
port 1194
proto udp
dev tun

# OpenVPN Server Certificate - CA, server key and certificate
ca /etc/openvpn/server/ca.crt
cert /etc/openvpn/server/server.crt
key /etc/openvpn/server/server.key

#DH key
dh /etc/openvpn/server/dh.pem

# Network Configuration - Internal network
# Redirect all Connection through OpenVPN Server
server 10.8.1.0 255.255.255.0
push "redirect-gateway def1"

# Using the DNS from https://dns.watch
push "dhcp-option DNS 8.8.8.8"

#Enable multiple client to connect with same Certificate key
duplicate-cn

# TLS Security
cipher AES-256-CBC
tls-version-min 1.2
tls-cipher
TLS-DHE-RSA-WITH-AES-256-GCM-SHA384:TLS-DHE-RSA-WITH-AES-256-CBC-SHA256:TLS-DHE-
RSA-WITH-AES-128-GCM-SHA256:TLS-DHE-RSA-WITH-AES-128-CBC-SHA256
auth SHA512
auth-nocache

# Other Configuration
keepalive 20 60
persist-key
persist-tun
```

```
comp-lzo yes
daemon
user nobody
group nobody

# OpenVPN Log
log-append /var/log/openvpn.log
verb 3
```

Включаем форвардинг и разрешаем OpenVPN в файерволе:

```
[root@vpn openvpn]# echo 'net.ipv4.ip_forward = 1' >> /etc/sysctl.conf
[root@vpn openvpn]# sysctl -p
net.ipv4.ip_forward = 1
[root@vpn openvpn]# firewall-cmd --permanent --add-service=openvpn
[root@vpn openvpn]# firewall-cmd --permanent --zone=trusted --add-interface=tun0
[root@vpn openvpn]# firewall-cmd --reload
success
```

Запускаем сервис, проверяем состояние OpenVPN-юнита и наличие сокета на UDP 1194:

```
[root@vpn openvpn]# systemctl start openvpn@server
[root@vpn openvpn]# systemctl enable openvpn@server
[root@vpn openvpn]# systemctl status  openvpn@server
● openvpn@server.service - OpenVPN Robust And Highly Flexible Tunneling
Application On server
    Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; disabled; vendor
preset: disabled)
   Active: active (running) since Sun 2019-11-03 15:01:27 MSK; 58s ago
 Main PID: 20195 (openvpn)
   Status: "Initialization Sequence Completed"
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
           └─20195 /usr/sbin/openvpn --cd /etc/openvpn/ --config ...

Nov 03 15:01:27 centos7.localdomain systemd[1]: Starting OpenVPN ...
Nov 03 15:01:27 centos7.localdomain systemd[1]: Started OpenVPN R...
Hint: Some lines were ellipsized, use -l to show in full.
[root@vpn openvpn]#[root@centos7 openvpn]# ss -tulpan | grep 1194
udp      UNCONN       0         0             *:1194                          *:*
users:(("openvpn",pid=20195,fd=4))
```

Теперь создаём конфиг клиента и указываем там IP-адрес VPN-сервера:

```
[root@vpn openvpn]# cd /etc/openvpn/client
[root@vpn client]# vim client01.ovpn
```

```
client
dev tun
proto udp

remote xx.xx.xx.xx 1194 # IP адрес сервера

ca ca.crt
cert client01.crt
key client01.key

cipher AES-256-CBC
auth SHA512
auth-nocache
tls-version-min 1.2
tls-cipher
TLS-DHE-RSA-WITH-AES-256-GCM-SHA384:TLS-DHE-RSA-WITH-AES-256-CBC-SHA256:TLS-DHE-
RSA-WITH-AES-128-GCM-SHA256:TLS-DHE-RSA-WITH-AES-128-CBC-SHA256

resolv-retry infinite
compress lzo
nobind
persist-key
persist-tun
mute-replay-warnings
verb 3
```

Упаковываем сертификаты клиента, сертификат CA и конфиг клиента в архив:

```
[root@vpn client]# cd /etc/openvpn/
[root@vpn openvpn]# tar -czvf client01.tar.gz client/*
client/ca.crt
client/client01.crt
client/client01.key
client/client01.ovpn
```

Скачиваем архив на машину клиента и распаковываем. Кроме этого, скачиваем пакеты для использования OpenVPN-клиента:

```
[root@centos7 ~]# sudo yum install openvpn network-manager-openvpn -y
================================================================================
 Package            Arch        Version             Repository
                                                                          Size
================================================================================
Installing:
 openvpn            x86_64      2.4.7-1.el7         epel      522 k
Installing for dependencies:
 pkcs11-helper      x86_64      1.11-3.el7          epel       56 k

Transaction Summary
```

```
==========================================================================
[root@centos7 ~]# scp root@192.168.1.41:/etc/openvpn/client01.tar.gz .
The authenticity of host '192.168.1.41 (192.168.1.41)' can't be established.
ECDSA key fingerprint is SHA256:Ro2rSQq+/bZVFP/5FLLpHMK1qvUbrXPVH9B/W2m5WFI.
ECDSA key fingerprint is MD5:47:62:08:27:1e:b2:83:f8:6f:b1:40:49:6c:9f:04:a0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.41' (ECDSA) to the list of known hosts.
root@192.168.1.41's password:
client01.tar.gz                   100% 4987     5.0MB/s   00:00
```

Разархивируем файл и подключимся к серверу:

```
[root@centos7 ~]# tar -xzvf client01.tar.gz
client/ca.crt
client/client01.crt
client/client01.key
client/client01.ovpn
```

Переходим в папку, где хранятся ключи и запускаем OpenVPN-клиента:

```
[root@centos7 ~]# cd client
[root@centos7 client]# openvpn --config client01.ovpn
Sun Nov  3 15:18:22 2019 OpenVPN 2.4.7 x86_64-redhat-linux-gnu [Fedora EPEL patched] [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built on Feb 20 2019
Sun Nov  3 15:18:22 2019 library versions: OpenSSL 1.0.2k-fips  26 Jan 2017, LZO 2.06
Sun Nov  3 15:18:22 2019 WARNING: No server certificate verification method has been enabled.  See http://openvpn.net/howto.html#mitm for more info.
Sun Nov  3 15:18:22 2019 TCP/UDP: Preserving recently used remote address: [AF_INET]192.168.1.41:1194
Sun Nov  3 15:18:22 2019 Socket Buffers: R=[212992->212992] S=[212992->212992]
Sun Nov  3 15:18:22 2019 UDP link local: (not bound)
Sun Nov  3 15:18:22 2019 UDP link remote: [AF_INET]192.168.1.41:1194
Sun Nov  3 15:18:22 2019 TLS: Initial packet from [AF_INET]192.168.1.41:1194, sid=a6db8667 c4a635e8
Sun Nov  3 15:18:22 2019 VERIFY OK: depth=1, CN=Easy-RSA CA
Sun Nov  3 15:18:22 2019 VERIFY OK: depth=0, CN=server
Sun Nov  3 15:18:22 2019 Control Channel: TLSv1.2, cipher TLSv1/SSLv3 DHE-RSA-AES256-GCM-SHA384, 2048 bit RSA
Sun Nov  3 15:18:22 2019 [server] Peer Connection Initiated with [AF_INET]192.168.1.41:1194
Sun Nov  3 15:18:23 2019 SENT CONTROL [server]: 'PUSH_REQUEST' (status=1)
Sun Nov  3 15:18:23 2019 PUSH: Received control message: 'PUSH_REPLY,redirect-gateway def1,dhcp-option DNS 8.8.8.8,route 10.8.1.1,topology net30,ping 20,ping-restart 60,ifconfig 10.8.1.6 10.8.1.5,peer-id 0,cipher AES-256-GCM'
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: timers and/or timeouts modified
```

```
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: --ifconfig/up options modified
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: route options modified
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: --ip-win32 and/or --dhcp-option options
modified
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: peer-id set
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: adjusting link_mtu to 1625
Sun Nov  3 15:18:23 2019 OPTIONS IMPORT: data channel crypto options modified
Sun Nov  3 15:18:23 2019 Data Channel: using negotiated cipher 'AES-256-GCM'
Sun Nov  3 15:18:23 2019 Outgoing Data Channel: Cipher 'AES-256-GCM' initialized
with 256 bit key
Sun Nov  3 15:18:23 2019 Incoming Data Channel: Cipher 'AES-256-GCM' initialized
with 256 bit key
Sun Nov  3 15:18:23 2019 ROUTE_GATEWAY 192.168.1.1/255.255.255.0 IFACE=ens33
HWADDR=00:0c:29:4b:c3:58
Sun Nov  3 15:18:23 2019 TUN/TAP device tun0 opened
Sun Nov  3 15:18:23 2019 TUN/TAP TX queue length set to 100
Sun Nov  3 15:18:23 2019 /sbin/ip link set dev tun0 up mtu 1500
Sun Nov  3 15:18:23 2019 /sbin/ip addr add dev tun0 local 10.8.1.6 peer 10.8.1.5
Sun Nov  3 15:18:23 2019 /sbin/ip route add 192.168.1.41/32 dev ens33
Sun Nov  3 15:18:23 2019 /sbin/ip route add 0.0.0.0/1 via 10.8.1.5
Sun Nov  3 15:18:23 2019 /sbin/ip route add 128.0.0.0/1 via 10.8.1.5
Sun Nov  3 15:18:23 2019 /sbin/ip route add 10.8.1.1/32 via 10.8.1.5
Sun Nov  3 15:18:23 2019 Initialization Sequence Completed
```

# Домашнее задание

1. На сервере server3 добавить еще один интерфейс — dummy с IP-адресом 33.33.33.33/32.
2. **НЕ** анонсировать этот интерфейс в OSPF.
3. Поднять openvpn-сервер на server3 и обеспечить возможность подключения клиента server1, используя сертификаты.
4. Убедиться, что server1 может пропинговать 33.33.33.33, когда VPN подключен, и не может этого сделать, когда VPN не подключен.

# Дополнительные материалы

1. Установка центра сертификации на предприятии. Часть 1.
2. Установка центра сертификации на предприятии. Часть 2.

# Используемая литература

1. Product Documentation for Red Hat Enterprise Linux 7.

2. Bulletproof SSL and TLS: Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications by Ivan Ristic. ISBN 978-1907117046