

Linux and more. SSH with keys

Artem Trunov



На прошлом занятии

- Узнали про Linux
- Узнали про ssh
- Залогинились на сервер с гостевым аккаунтом
- Посмотрели характеристики сервера
- Поработали с файловой системой

На этом занятии

- SSH с ключами.
- Генерируем пару ключей
- Создаем аккаунт на Github
- Загружаем публичный ключ на Github
- Проверка и создание собственного аккаунта на сервере
- Работа с командной строкой
 - Pipes, io redirection
 - Поиск в файлах - grep

Логинимся. SSH (рассматривали на Занятии 1)

- Transport Layer Protocol
 - Аутентификация сервера (сервер посылает свой публичный ключ)
 - Создание безопасного соединения для аутентификации пользователя (Диффи-Хеллман)
- User Authentication Protocol
 - Подтверждение аутентичности пользователя (пароль или сертификат)
- Connection Protocol
 - Создание канала для передачи данных в течении сессии ssh

Протокол передачи данных — набор соглашений интерфейса логического уровня, которые определяют обмен данными между различными программами. Эти соглашения задают единообразный способ передачи сообщений и обработки ошибок.

Еще немного о Transport Layer Protocol

- Используется для создания безопасного соединения.
- Соединение шифруется с помощью симметричного ключа
 - Один ключ используется для шифрования и дешифрования сообщений.
 - Проблема? Нет - ключ не передается между клиентом и сервером
 - Клиент и сервер генерируют этот ключ независимо по определенному протоколу (Диффи-Хелман)
 - Этот ключ используется только для установки соединения и аутентификации пользователя.
- Шифр - алгоритм(ы) преобразования открытых данных в зашифрованные
- Ключ - параметр алгоритма, который выбирает конкретное преобразование из множества возможных для данного алгоритма.

Симметричное и асимметричное шифрование

- Симметричное
 - Обе стороны используют один и тот же ключ для шифрования и расшифрования сообщений.
- Асимметричное
 - Используется пара ключей
 - Публичный ключ используется для шифровки сообщения
 - Приватный ключ используется для расшифровки.
 - Приватный ключ должен оставаться в руках владельца и оберегаться
 - Публичный ключ распространяется среди тех, кто хочет послать вам зашифрованное сообщение

Аутентикация

- User Authentication Protocol
 - Подтверждение аутентичности пользователя (пароль или сертификат)
- Можно ли использовать симметричный ключ для аутентификации?
 - Нет. Проблема пересылки ключа
 - И его дальнейшего использования - тот у кого есть ваш ключ может от вашего имени что-то зашифровать

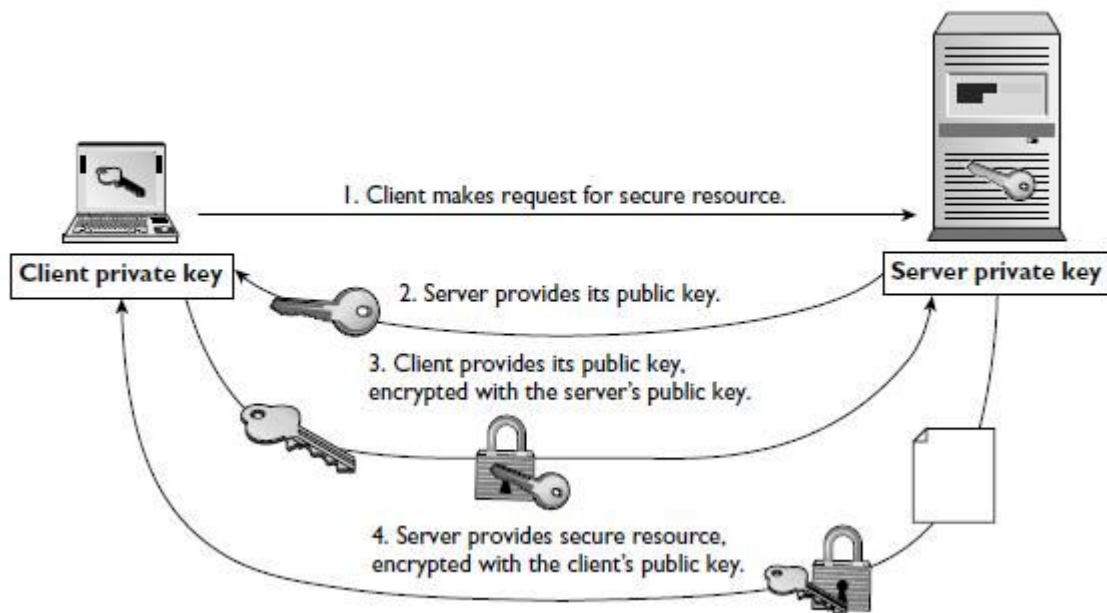
SSH-login с ключами (User Authentication).

- Сервер шифрует “challenge message” с помощью публичного ключа пользователя, и посылает его пользователю
- Если пользователь может расшифровать его, то он подтвердил, что владеет приватным ключем.
- Устанавливается сессия

Таким образом, для логина с ключом требуется:

- Приватный ключ
- Публичный ключ

Логин с ключем. Схема.



+Клиент расшифровывает secure resource (challenge message) и отправляет серверу для сверки

SSH-login с ключами

| Как выглядят ключи

```
artem@artem-ubuntu2:~$ head .ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,31042FCC3637A25D710545D33331E2BD

nyebHaG1PnpqsimRMuuUFudisxnQWw/kJcQ92YPuzpXPeGort7rBdgowJvliX0Sh
fAJmD7it7LPISaGnPWEA70EZKnvq5EBnATa31q9qxgeaM88w5Ne/QAbYV7MLQZ0k
wnTXFkhjGuaD/1WrJRNHB9sc4rWMnde7RZiRdpTEcv85iTMrc1KrZSZRTB7wza4k
wk4b4UMJaER02fPG9h1RqVLzdXT++WABgnoYbCsaenNS8F7dBDXqUdgSA05m+pBb
Wa/ML74Nmm6SGTG+4B5CLUFsHE6DjbnimGDIjjkobbrcpLpCEKoFwQ4dHBQui1I9
e9oUel2Ubu8Inq5KA91+cH0SBCJLn/zSDvd2HubGfUr2dP+cl4ULCLtSvsUe1Pdf
```

```
artem@artem-ubuntu2:~$ head .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCe2J8KngRKtVHjjFzTW9qcd7f29vXZjuz9Q+Lzf408
NiY/E2CDD7ZkuNY4Da6vqFjMw9bm0JSQD6rBV2Ek+2+quDEmTvZTSJWycjtP7t7j8k168rKVAVQv0C2K
dG4WA45Tzb/ygY7gfGf/tK3o10+ecyvFdYHPTeVe27fv2InQty4RqamIszEyJJW5UAA/EzGXczfg6JfA
Cmt3W+bcyIxQumPHA07FLTcxT9pUkCLCUB3DXjrGMAv/WeY6olvf4QKNBzb002VJu8Sc2Q3XkVYHI41g
Fb/M7Qe4FoMyr0DMOMCT+KyMsqmqI2cwpK/1KTEn3Xd/1mWZv7F+4bdQwLSh artem@artem-ubuntu0
```

SSH-login с ключами

~/.ssh - настройки и ключи

- `id_rsa` - приватный ключ
- `id_rsa.pub` - публичный ключ
- `config` - опции и алиасы для команды `ssh`
- `known_hosts` - цифровые “отпечатки” серверов, с которыми было установлено ssh-соединение.

```
artem@artem-ubuntu2:~$ ls -l .ssh/{id*,config,known_hosts}
-rw----- 1 artem artem 2097 Jan 24 14:40 .ssh/config
-rw----- 1 artem artem 1766 Jul  2 2018 .ssh/id_rsa
-rw-r--r-- 1 artem artem  401 Jul  2 2018 .ssh/id_rsa.pub
-rw----- 1 artem artem 25518 Oct 16 11:05 .ssh/known_hosts
```

Практика 1 (выполняется на своем ПК)

- Этап 1. Сгенерируем пару ключей на своем компьютере
 - Знакомство с командой **ssh-keygen**
 - Внимание! Если у вас уже есть ключ `~/id_rsa` и вам надо его сохранить, не генерируйте ключ, а переходите к следующему шагу
- Этап 2. Загрузим публичный ключ на наш сервер
 - Исполнение команд через **ssh**
 - Команда **scp**
- Этап 3. Залогинимся, используя ключ.
 - Пароль больше вводить не придется

Этап 1. Генерация пары ключей

- Внимание! Если у вас уже есть ключ в `~/.ssh/id_rsa` и вам надо его сохранить, не генерируйте ключ
 - Как проверить
 - **type** `C:\Users\USERNAME\.ssh\id_rsa.pub` (Windows)
 - **cat** `~/.ssh/id_rsa.pub` (MacOS, Linux)
- Обратите внимание на атрибуты доступа - приватный ключ не должен быть доступен для чтения никому, кроме владельца
 - **chmod** `600 ~/.ssh/id_rsa`
 - `ssh-keygen` ставит правильные пермишены при генерации самостоятельно

ssh-keygen - создание пары ключей

- Запустите без опций и аргументов
 - **ssh-keygen**
- Посмотрите, что там создалось:
 - **cat .ssh/id_rsa; cat .ssh/id_rsa.pub** - MAC, Linux
 - **dir .ssh; type .ssh\id_rsa.pub** - Windows
- Подробнее:
 - **man ssh-keygen** - подробное описание, на Mac, Linux
 - **ssh-keygen /h** - только опции, на Windows.
 - -q -без лишнего вывода и вопросов
 - -b bits - битность ключа (2048 достаточно)
 - -t algo - выбор семейства алгоритмов шифрования
 - -N password - назначение пароля на ключ
 - -C комментарий к ключу
 - -f имя файла приватного ключа (+ .pub для публичного)

ssh-keygen - пример на Windows

```
C:\Users\Admin>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\Admin\.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\Admin\.ssh/id_rsa.
Your public key has been saved in C:\Users\Admin\.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:L7lYL1bL173EacBGVFv8G1qaNg6g3mA+oDv7uZsDz/A admin@Admin-PC
The key's randomart image is:
+---[RSA 2048]---+
|                ..O.|
|                .  +|
|                .  ..|
|               .  o  o.|
|              S .  += o|
|             o .  + o...*o..|
|            B = *o..+..*|
|           o Eo*o=o  ..+ .|
|          o=*=.o...  ..|
+-----[SHA256]-----+
```

ssh-keygen - ключ на Windows

```
C:\Users\Admin>dir .ssh
Том в устройстве C имеет метку Acer
Серийный номер тома: 0AC3-785D

Содержимое папки C:\Users\Admin\.ssh

16.09.2020  16:33      <DIR>          .
16.09.2020  16:33      <DIR>          ..
16.09.2020  16:33                1 679 id_rsa
16.09.2020  16:33                397 id_rsa.pub
11.09.2020  12:41                721 known_hosts
          3 файлов                2 797 байт
          2 папок   337 421 303 808 байт свободно

C:\Users\Admin>type .ssh\id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQAcjdeIuI00nCTfl44DwLotGhKrtWy/5QZK
eC8T0+ujScKn4SXf5yiYaiH9LYRpf/KqCl9Re7BGrk2j1hUxacOrNU9I12Qp064+ymMAufJ
2hBRUEmjv1cFIuhBg0ZfSeOJlJrTtpM3IiIJQh5KX7x1MT4ImjzQu1MhRS7qyKE1Au/z4tg
D1vfgjngR4IUmtXERouJ admin@Admin-PC

C:\Users\Admin>
```


Этап 2. Загружаем публичный ключ на сервер

Все команды запускаются на вашем лаптопе/desktope

1. Создаем директорию `.ssh` на сервере в своей домашней папке
 - a. `ssh lquestxxx@bigdatamasters.ml mkdir .ssh`
2. Убеждаемся, что она создана
 - a. `ssh lquestxxx@bigdatamasters.ml ls -al .ssh`
3. Запускаем команду копирования публичного ключа с лаптопа в файл `authorized_keys` в директории `.ssh` на сервере:
 - a. `scp .ssh/id_rsa.pub lquestxxx@bigdatamasters.mlu:.ssh/authorized_keys`
 - i. опция `-l` в `scp` делает не то.
 - ii. Если директории `.ssh` нет у вас в домашней директории, будет ошибка
4. Если все в порядке, то команда с шага 2 теперь должна отработать без ввода пароля (это и есть Этап 3)

Загружаем - пример на windows

```
C:\Users\Admin>ssh lguest217@bigdatamasters.ml mkdir .ssh
lguest217@bigdatamasters.ml's password:

C:\Users\Admin>ssh lguest217@bigdatamasters.ml ls -al .ssh
lguest217@bigdatamasters.ml's password:
total 8
drwxr-xr-x 2 lguest217 students 4096 Sep 21 10:02 .
drwxr-xr-x 5 lguest217 students 4096 Sep 21 10:02 ..

C:\Users\Admin>scp .ssh/id_rsa.pub lguest217@bigdatamasters.ml:~/.ssh/authorized_keys
lguest217@bigdatamasters.ml's password:
id_rsa.pub

C:\Users\Admin>ssh lguest217@bigdatamasters.ml ls -al .ssh
total 12
drwxr-xr-x 2 lguest217 students 4096 Sep 21 10:06 .
drwxr-xr-x 5 lguest217 students 4096 Sep 21 10:02 ..
-rw-r--r-- 1 lguest217 students  397 Sep 21 10:06 authorized_keys

C:\Users\Admin>_
```

Практика 2 (на сервере под гостевым акк)

- Сгенерируем пару ключей на сервере
 - Загрузим публичный ключ на Github.com
 - Я зашифрую вашим публичным ключем сообщение
 - Вы расшифруете его вашим приватным ключем
-
- Создадим ваш персональный аккаунт на сервере, с названием как ваш ник на гитхабе.
 - Вы залогинитесь на него с ключем.

Задание

- Зайдите на сервер под гостевым аккаунтом и сгенерируйте пару из приватного и публичного ключа используя `ssh-keygen`
 - `ssh-keygen`
 - Enter на все вопросы (путь, пустой пароль)
- Проверьте наличие и содержимое файлов вашей пары `id_rsa`, `id_rsa.pub`
 - `ls`, `cat`
- Попробуйте зашифровать/расшифровать какой-нибудь (очень короткий < 200 B) текст с помощью ключей (см. следующий слайд)

Шифрование с помощью ключа ssh

- Мы используем утилиты **openssl**, так как утилиты **openssh** не предусматривают такой работы с сообщениями.
- Подготовка публичного ключа
 - Его необходимо конвертировать в pem-формат (см слайд в конце):
 - `ssh-keygen -f ~/.ssh/id_rsa.pub -e -m PKCS8 > ~/.ssh/id_rsa.pub.pem`
- Зашифровка
 - `openssl rsautl -encrypt -pubin -inkey ~/.ssh/id_rsa.pub.pem -ssl -in test -out encrypted`
- Расшифровка
 - `openssl rsautl -decrypt -inkey ~/.ssh/id_rsa -in encrypted -out decrypted`

Что-то не так: не можете расшифровать

- “Unable to load private key”

```
lguest217@linux1:~$ openssl rsautl -decrypt -inkey ~/.ssh/id_rsa.pem -in /tmp/lguest217-enc  
rypted -out decrypted  
unable to load Private Key  
140136360441152:error:0909006C:PEM routines:get_name:no start line:../crypto/pem/pem_lib.c  
:745:Expecting: ANY PRIVATE KEY
```

- Проверьте с помощью `cat .ssh/id_rsa`, что ваш приватный ключ типа RSA, а не OPENSSH:
-----BEGIN RSA PUBLIC KEY-----

- “Padding check failed” или что-то подобное

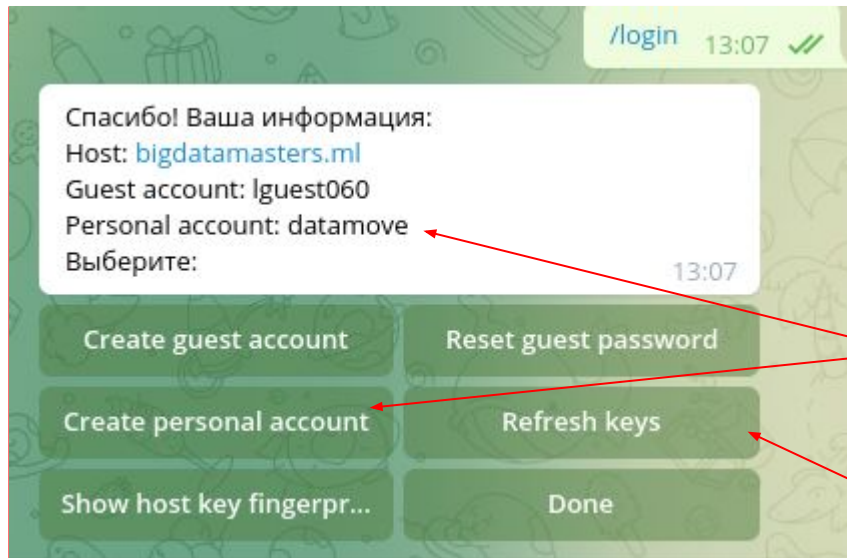
```
lguest217@linux1:~$ openssl rsautl -decrypt -inkey ~/.ssh/id_rsa -in /tmp/lguest217-encryp  
ted -out decrypted  
RSA operation error  
140246635259200:error:0407109F:rsa routines:RSA_padding_check_PKCS1_type_2:pkcs decoding e  
rror:../crypto/rsa/rsa_pk1.c:251:  
140246635259200:error:04065072:rsa routines:rsa_ossl_private_decrypt:padding check failed:  
../crypto/rsa/rsa_ossl.c:491:
```

- Скорее всего несоответствие ключей. Проверьте, что публичный ключ из вашей директории действительно загружен на гитхаб и там он самый последний в списке.

Задание

- Загрузите ваш публичный ключ на Github.com
 - Инструкция: <https://docs.github.com/en/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account>
 - Используйте copy/paste содержимого ключа.
 - Добавьте сгенерированный ключ, даже если у вас уже есть на гитхабе свои. Ваш ключ должен быть последним в списке. Как он будет называться на гитхабе - не важно.
- В **@aim_big_data_bot** запустите команду **/login**
 - Нажмите **“Create personal account”**, введите ваш ник на Github
 - Будет предложено расшифровать некий файл под вашим гостевым аккаунтом приватным ключем, которому соответствует публичный, загруженный на гитхаб. Сделайте это.
 - Опять в боте запустите **/login** и снова нажмите **“Create personal account”**. Если проверка прошла, будет создан аккаунт с названием, как ваш ник на Github и всеми ключами с гитхаба в `authorized_keys` этого аккаунта

Работа с ботом (персональный акк)



“Персональный” аккаунт для логина с ключем
(такой же как ваш ник на github.com)

Добавьте на github.com любые ключи и нажмите
Refresh - бот скопирует ключи в `authorized_keys`

Задание

- Из-под гостевого аккаунта, залогиньтесь на ваш персональный аккаунт, используя приватный ключ
 - `ssh gitnick@bigdatamasters.ml`
- Теперь вы можете использовать любой ключ из тех, что вы используете на гитхабе для логина на ваш персональный аккаунт с любого устройства.
 - Вы также сможете пользоваться командами **git clone, pull, push** без пароля, но только под теми аккаунтами, где у вас есть соответствующий приватный ключ `~/.ssh/id_rsa`
- Самостоятельно
 - Чтобы иметь возможность логиниться в персональный акк с лаптопа, надо загрузить в гитхаб публичный ключ с лаптопа и в боте нажать `/login, refresh keys`. Эта команда синхронизирует ключи с гитхаба под вашим персональным аккаунтом на сервере

ssh config - для удобства

- Полная команда
 - `ssh -l lguest060 -i /home/artem/.ssh/id_rsa login.linuxcourse.ru`
- `~/.ssh/config`

```
Host linux1
Hostname bigdatamasters.ml
User lguest060
IdentityFile /home/artem/.ssh/id_rsa
```

- Теперь логиниться можно командой
 - `ssh linux1`

Самостоятельно: создайте себе ssh config с записью для входа на bigdatamasters.ml

Практический совет

Ключи для логина по ssh

- Поставьте на приватный ключ пароль
- Берегите ключ так же, как пароль.
- Сделайте резервную копию ключей.
- Не используйте ключи для логина совместно. Каждый член команды должен иметь свой ключ

Еще о шифровании сообщений

- Openssl не подходит для шифрования писем
- PGP, OpenPGP, GPG - с 1991 года
 - Не взлетело, сложно использовать простому юзеру.
 - Даже Google и Yahoo не смогли встроить в почту.
 - Или не захотели
- Пользователям удобнее мессенджеры с шифрованием
 - Или с “шифрованием” - вы точно уверены в WA, TG, Signal?
 - Вы уверены в конечном устройстве (телефоне)?
- “Безопасная” почта (protonmail etc)
 - Кем контролируется? А если ее взломают?
- Если вы действительно заботитесь о своей связи, придется делать все самостоятельно и использовать нестандартные решения.
 - В стандартных сервисах вас накроют заодно со всеми
 -

Ssh key forwarding

- У вас на персональном аккаунте нет приватного ключа - он под гостевым.
- А без него вы не сможете сделать git clone приватного репо.

Решение - аутентификационный агент. Под гостевым акк:

ssh-add #добавляет приватный ключ в аутентификационный агент
ssh -A gitnick@login.linuxcourse.ru # чтобы иметь возможность коммуницировать с агентом

Теперь под персональным аккаунтом можно общаться с гитом.

P.S. Естественно, можно также скопировать приватный ключ под персональный аккаунт, но это менее безопасно. В идеале, ваш ценный ПК не должен покидать вашего устройства.

P.P.S. Используйте **ForwardAgent yes** в ssh config

Когда ключ не на “своем” месте

- Свое место (по умолчанию) ~/.ssh/id-rsa
- Но при генерации ключа можно указать путь к новому ключу.
- Тогда нужно использовать **ssh -i /path/to/pk**
 - Или scp -i
- В ssh config:
 - IdentityFile /path/to/pk
- Добавить в агента
 - ssh-add /path/to/pk

Разное. Работа с ключами из питона

```
import rsa
```

```
#Боб формирует публичный и секретный ключ  
(bob_pub, bob_priv) = rsa.newkeys(512)
```

```
#Алиса формирует сообщение Бобу и кодирует его в UTF8,  
#поскольку RSA работает только с байтами  
message = 'hello Bob!'.encode('utf8')
```

```
#Алиса шифрует сообщение публичным ключом Боба  
crypto = rsa.encrypt(message, bob_pub)
```

```
#Боб расшифровывает сообщение своим секретным ключом  
message = rsa.decrypt(crypto, bob_priv)  
print(message.decode('utf8'))
```

Разное. Формат PEM

- PEM PKCS8-формат:

-----BEGIN PUBLIC KEY-----

содержимое ключа

-----END PUBLIC KEY-----

- Но он неудобен для openssh, так как предпочтительнее хранить ключ в одной строке, как в файле ~/.ssh/authorized_keys:

ssh-rsa <*содержимое ключа*> <*id ключа(comment)*>