Удаленное администрирование серверов.
Настройка, оптимизация, поддержка.

УСЛУГИ КОНТАКТЫ

СТАТЬИ
Backup (3)
FreeBSD (13)
HTTP (1)
Mail (1)
MySQL (2)
Networks (1)
Nginx (10)
Other (4)
Security (2)
SSH (5)
Unix (Linux) (10)

MANUALS
Shell Scripting
Unix ToolBox
Команды Freebsd

SSH, аутентификация по ключам, ssh-keygen, ssh-agent

Версия для печати

В нескольких предыдущих статьях данного раздела, мы более-менее раскрыли тему протокола **SSH**, настройку и использование *SSH* сервера и *SSH* клиента в операционной системе FreeBSD. В данной статье хотелось-бы рассказать об *SSH* аутентификации на основе пар ключей, заодно рассмотреть остальные программы из пакета программного обеспечения, протокола *SSH*, под FreeBSD.

Итак, нужно это в первую очередь для удобства, при удаленном администрировании серверов, не нужно вводить пароль при каждом подключении, и в определенной степени более безопасно, нежели подключаться к удаленной машине только по паролю. Общий принцип для аутентификации на основе публичного ключа, в протоколе *SSH*, таков:

- С помощью программы **ssh-keygen**, должна быть сгенерирована пара ключей, публичный ключ (*public key*) и приватный ключ (*private key*)
- Секретный ключ, всегда остается у клиента и никому никогда не показывается.
- Публичный ключ копируется на удаленный *SSH* сервер (говорим опять-же в контексте операционной системы FreeBSD) и кладется в специальный файл, известный *SSH* серверу. По-умолчанию, для хранения публичных ключей, используется файл *~/.ssh /authorized_keys*. Файл для хранения ключей назначается в файле конфигурации *SSH* сервера, директивой **AuthorizedKeysFile**
- Клиент, отправляет *SSH* серверу свой публичный ключ и запрашивает аутентификацию по данному ключу.
- Сервер проверяет файл *~/.ssh/authorized_keys*, если такой ключ найден, *SSH* сервер отправляет клиенту сообщение, зашифрованное найденным публичным ключом пользователя.
- Клиент должен расшифровать сообщение с помощью своего приватного ключа, если приватный ключ защищен паролем (а так и должно быть всегда, в целях безопасности), программа *ssh*, попросит пользователя ввести пароль, что-бы сначала расшифровать сам ключ.
- Если сообщение расшифровано, правильность публичного и приватного, ключей, считается подтвержденной и пользователю предоставляется доступ в систему.

Весь процесс аутентификации можно посмотреть, с помощью опции `-v` (*verbose*), программы `ssh` , очень полезная штука, особенно на стадии настройки серверной и клиентской частей протокола *SSH*.

Генерация ключей с помощью программы ssh-keygen.

Для создания и управления ключами, предназначена программа `ssh-keygen` , так-же входящая в пакет программного обеспечения *OpenSSH*. Полный список опций можно как всегда посмотреть командой `man ssh-keygen` . Здесь приведу лишь несколько из них:

-t type
ssh-keygen, работает с тремя типами ключей. Возможные значения:
RSA 1 - для протокола SSH версии 1.
RSA - для протокола SSH версии 2.
DSA - для протокола SSH версии 2.

-b
Длина ключа в битах.
RSA - минимальная длина, 768 бит, длина ключа по-умолчанию, 2048 бит.
DSA - длина 1024 бита.

-i
Данная опция используется для импорта ключей из одного формата (например ключи сгенерированные программой `PuTTYgen` , для Windows), в формат OpenSSH.

-l
Посмотреть отпечаток секретного ключа (fingerprint).

-p
Изменить секретную фразу приватного ключа.

Сгенерируем пару *RSA* ключей, это рекомендуемый формат, как наиболее устойчивый к взлому. По-умолчанию, ключи, сохраняются в домашнюю директорию пользователя, в файлы *~/.ssh/id_rsa* - приватный (секретный) ключ, и *~/.ssh/id_rsa.pub* - публичный ключ.

```
vds-admin /root# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Здесь предлагается ввести путь для сохранения файлов ключей. Имейте в виду, если просто указать свое имя файла, например: putnev_id_rsa, то ключи будут сохранены в корне домашней директории пользователя, в моем случае в директории /root, с именами: pu
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Здесь вас попросят ввести секретную фразу, настоятельно рекомендую делать это всегда. )
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
56:79:b5:61:ea:19:70:13:a4:67:a2:af:15:11:db:b5 root@vds-admin.
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|E                |
| .               |
|  o o ..         |
| . B +.So        |
|  o =.. o .      |
| . o. .o         |
|  =..+o          |
| o.+*o           |
+-----------------+
```

Вот собственно и все, сгенерирована пара ключей *RSA*, с длиной 4096 бит и сохранены в файлы */root/.ssh/id_rsa* и */root/.ssh/id_rsa.pub* .

Настройка SSH сервера на аутентификацию по открытому ключу.

SSH сервер естественно должен быть настроен на аутентификацию по ключам, приведу кусок, касающийся аутентификации, своего файла конфигурации, *SSH* сервера. Все что закомментировано в файле конфигурации *SSH*, отсюда убрал, для простоты восприятия:

```
PermitRootLogin yes
Данная директива нужна, если вы планируете работать под учетной записью root.
PubkeyAuthentication yes
AuthorizedKeysFile .ssh/authorized_keys
RhostsRSAAuthentication no
HostbasedAuthentication no
PermitEmptyPasswords no
UseLogin no
```

Теперь копируем публичный ключ на удаленный *SSH* сервер:

```
vds-admin /root/.ssh# cat id_rsa.pub | ssh 192.168.50.50 "cat >> ~/.ssh/authorized_keys"
```

Так как для копирования вы подключаетесь к *SSH* серверу, он запросит пароль, авторизацию по ключам-то мы еще не настроили.) Я работал под учетной записью *root*, а без явного указания имени пользователя в командной строке или в конфигурационном файле *SSH* клиента, подключение происходит с именем текущего пользователя, то есть мне нужно было ввести пароль пользователя root, удаленной системы. После ввода пароля, публичный ключ будет добавлен в конец файла *~/.ssh/authorized_keys*, так как мы подключаемся пользователем root, то путь *~/.ssh/authorized_keys* указывает на директорию */root/.ssh/authorized_keys* .

Ключи сгенерированы и скопированы на *SSH* сервер, сервер соответствующим образом настроен, пробуем подключится:

```
vds-admin /root/.ssh# ssh 192.168.50.50
Enter passphrase for key '/root/.ssh/id_rsa':  Здесь вводим нашу секретную фразу, указанную при генерации ключей.
```

Если пароль на ключ введен верно, получаем доступ в удаленную систему.

> Обратите внимание на следующий момент, с приведенным выше вариантом конфигурации *SSH* сервера, при неудачной аутентификации по ключам, например если неправильно ввести секретную фразу ключей, будет предложена аутентификация по паролю. Что-бы изменить это поведение и например вообще не пускать пользователя root иначе, как по ключам, можно изменить в конфигурационном файле сервера, значение директивы **PermitRootLogin** с *yes* на *without-password*.

Использование программы ssh-agent

Как было сказано выше, в целях безопасности, приватный ключ, всегда должен быть защищен секретной фразой (паролем), но это вызывает некоторые неудобства, вам придется вводить секретную фразу, каждый раз когда вы подключаетесь к удаленному серверу по ключу, было-бы гораздо проще ввести пароль на ключ один раз и пользоваться им сколько потребуется. На этот случай в пакете *OpenSSH*, существуют специальные программы `ssh-agent` и `ssh-add` , в общем-то вторая является дополнением первой.

Как это работает. Поле запуска программы `ssh-agent` , в нее добавляются расшифрованные ключи, то есть при добавлении она запросит секретную фразу ключа, для его дешифровки, и далее `ssh-agent` , будет выдавать уже расшифрованные ключи по запросу, например программе SSH.

Запускать `ssh-agent` , можно двумя способами, со специальной опцией, говорящей, какой тип оболочки используется, или с помощью команды `eval` . Принципиальной разницы как его запускать, нет, просто в случае с опцией, вы должны точно знать, какую оболочку вы используете.

ssh-agent -c
Если в качестве оболочки используется C - Shell

ssh-agent -s
Если в качестве оболочки используется Bourne Shell

eval `ssh-agent`
В таком варианте запущенный ssh-agent, будет передан команде eval, которая выполнит его в текущей оболочке. Обратите внимание, используются **обратные кавычки** а не обычные !

Итак, запускаем `ssh-agent` :

```
vds-admin /root/.ssh# eval `ssh-agent`
Agent pid 1982
```

При запуске, `ssh-agent` создает переменные окружения, проверим какие:

```
vds-admin /root/.ssh# env | grep SSH_A
SSH_AUTH_SOCK=/tmp/ssh-TEwitdI5mr/agent.1981
В этой переменной хранится сокет, через который программа ssh, будет связываться с ssh-agent.
SSH_AGENT_PID=1982
Это PID процесса ssh-agent
```

Теперь нужно поместить в него расшифрованные ключи, делается это с помощью программы `ssh-add` .

Если запустить ее без аргументов, будут добавлены все ключи, найденные в стандартных местах их обитания, то есть будут просканированы следующие файлы, *~/.ssh/identify* , *~/.ssh/id_rsa* и */.ssh/id_dsa* . Если ключ защищен парольной фразой, программа попросит ввести ее, что-бы расшифровать ключ и загрузить уже готовый к применению.

```
vds-admin /root/.ssh# ssh-add
Enter passphrase for /root/.ssh/id_rsa: Запрос пароля на расшифровку
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
```

Теперь пробуем подключиться к удаленному SSH серверу:

```
vds-admin /root/.ssh# ssh 192.168.50.50
```

```
Last login: Tue Jul 7 18:45:27 2009 from .host.
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
The Regents of the University of California. All rights reserved.

FreeBSD 7.1-STABLE (SMP_KERNEL) #1: Tue Mar 10 18:14:59 UTC 2009

Welcome to FreeBSD!
```

Как видите пароль у нас больше никто не спрашивает, программа *SSH*, получает уже расшифрованный ключ от `ssh-agent` и мы успешно подключаемся к удаленному *SSH* серверу.

Посмотреть отпечатки загруженных в `ssh-agent` ключей, можно той-же командой `ssh-add` с опцией **-l**, или целиком ключи, опцией **-L**.

```
vds-admin /root/.ssh# ssh-add -l
4096 56:79:b5:61:ea:19:70:13:a4:67:a2:af:15:11:db:b5 /root/.ssh/id_rsa (RSA)
vds-admin /root/.ssh# ssh-add -L
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAgEArL0hIMmhw8rXeg0p72+EJXnC4iAY2XTkPAdTb3LnQb9bc0E5wvd
cwCdNEtLlDIDCH+z0I1FaP3TfpvgVkv59X15TaNIeoB7uydqXvlLMOxpOJkfbc3eiA6a07PvZHMKXcIA0ZZ9+j12u
l+HsGOK2qMQ5g52mOc6BOF1PVuoHfTR1C9nExv5UCA6h7e/v2wxq79pMW07nx7nshB5/1n5Gnyx+toQEzRiFbf
zOJBB1ry/9NUF1DiBwOhKJVdEJBTUi0hyh/e77UAmVtkguEtjrsDEdxJ31sV21SL97EZHymMjRPjwU2nWjRkHf0Pi7
dlXBoCKRj3dQps38kwFd3m9Tu4+hXSnsF8FdxkX5y9XmN8Uz8UWR6O2zslr7xZubkDR3aCq1dtcbu2nkvC4+Vy
TOxEdnaNqDlC6U6G6aUVKFc0Rb5dcPnqpKqUHWE8MlXq/obKMRjuSz+GOr1VgRe/wZM7/0GoO1Xrv2MDMhS+
S1uR+XkHkQr/EjTSxPiDZ92snZhtiyPIzTUZDOmclWHbe4gyvxDtU3Lxqzl3t1+Murg4sN1NrkZIHefMq2xeCOS8P
bI89b3zJG2PJ3i2PSsOMviqIBOL3BBskGSWksJKi/YvvKwrlKaSM10wMZTbXHomgu+6jRd7cZtUOmU/FO0IoKejB
MwuYbcPC+TCWBks0phU= /root/.ssh/id_rsa
```

Загружен один ключ, по которому мы подключались к удаленной машине. Кроме этого, при запуске `ssh-add`, можно указать путь до конкретного ключа, который необходимо загрузить, например:

```
vds-admin /root/.ssh# ssh-add /root/.ssh/id_rsa.new1
Enter passphrase for /root/.ssh/id_rsa.new1:
Identity added: /root/.ssh/id_rsa.new1 (/root/.ssh/id_rsa.new1)

Проверяем, что у нас теперь в ssh-agent:
vds-admin /root/.ssh# ssh-add -l
4096 56:79:b5:61:ea:19:70:13:a4:67:a2:af:15:11:db:b5 /root/.ssh/id_rsa (RSA)
2048 68:81:38:fe:66:e8:05:88:8b:49:80:d2:d1:8b:bf:99 /root/.ssh/id_rsa.new1 (RSA)
Загружено уже 2 ключа
```

Удаляются ключи из `ssh-agent`, так-же просто как и добавляются, для этого используется опция **-d**, без параметров, для удаления стандартных ключей, опция **-d** *файл_ключа*, если нужно удалить конкретный ключ, или опция **-D**, для удаления всех ключей, например:

```
vds-admin /root/.ssh# ssh-add -l
4096 56:79:b5:61:ea:19:70:13:a4:67:a2:af:15:11:db:b5 id_rsa (RSA)
2048 68:81:38:fe:66:e8:05:88:8b:49:80:d2:d1:8b:bf:99 id_rsa.new1 (RSA)
2048 c7:9f:b1:3b:c1:d0:61:15:38:27:d1:36:a7:49:55:cd id_rsa.new2 (RSA)

vds-admin /root/.ssh# ssh-add -d id_rsa.new2
Identity removed: id_rsa.new2 (id_rsa.new2.pub)

vds-admin /root/.ssh# ssh-add -l
4096 56:79:b5:61:ea:19:70:13:a4:67:a2:af:15:11:db:b5 id_rsa (RSA)
2048 68:81:38:fe:66:e8:05:88:8b:49:80:d2:d1:8b:bf:99 id_rsa.new1 (RSA)

vds-admin /root/.ssh# ssh-add -D
All identities removed.

vds-admin /root/.ssh# ssh-add -l
The agent has no identities.
```

Приведу список самых используемых опций программы `ssh-add`:

ssh-add
: Без опций, добавляются стандартные ключи

ssh-add имя файла
: Добавляются конкретный ключ

-l
: Показывает отпечатки всех загруженных в данный момент ключей

-L
: Посмотреть список самих ключей

-D
: Из *ssh-agent*, будут удалены все ключи

-d имя файла
: Удаляет конкретный ключ

-t
: Установить время жизни ключей, через данный промежуток времени ключи будут выгружены.

-x
: Заблокировать *ssh-agent* паролем

-X
: Разблокировать *ssh-agent*

Что-бы закрыть `ssh-agent`, можно вызвать его с опцией **-k**, ну или на крайний случай прибить сигналом, например `kill -QUIT PID`, но это крайняя мера и при корректном запуске, это не потребуется:

```
vds-admin /root/.ssh# ssh-agent -k
unsetenv SSH_AUTH_SOCK;
unsetenv SSH_AGENT_PID;
echo Agent pid 1982 killed;
```

Как видите произошел обратный процесс, переменные очищены, процесс убит.

Форвардинг ssh-agent

Форвардинг агента включается в файле конфигурации клиента *SSH*, директивой **ForwardAgent yes**. Как это работает. Вы запускаете *ssh-agent* на локальной машине, загружаете ключи, подключаетесь к удаленному *SSH* серверу, сервер создает обратное перенаправление через созданный *SSH* туннель к вашему `ssh-agent` и вы можете использовать загруженные в него ключи для последующих соединений.
Для примера, с локального хоста, **Local_host**, подключаемся к удаленной машине **Remote_host**, по каким-то причинам, нам понадобилось что-то посмотреть на еще одном хосте, **Next_remote_host**, что происходит в таком случае:

- Клиент *ssh* с **Local_host**, подключается к *SSH* серверу, **Remote_host**, и запрашивает форвардинг для *ssh-agent*
- Сервер *SSH*, */usr/sbin/sshd*, хоста **Remote_host**, создает сокет в */tmp/ssh-XXXXXXX/agent.#####* и устанавливает переменную окружения *SSH_AUTH_SOCK*, присваивая ей путь к сокету.
- Когда нам понадобится подключиться к следующему серверу, (мы сейчас на сервере **Remote_host**), *SSH* клиент хоста **Remote_host**, обращается по пути, лежащему в переменной *SSH_AUTH_SOCK*, то есть к сокету.
- *SSH* сервер, находящийся на другом конце сокета */tmp/ssh-XXXXXXX/agent.#####*, передает данные из *ssh*, сервера **Remote_host**, на *ssh-agent*, запущенный на хосте **Local_host**. Вся работа с ключами происходит на машине **Local_host** а не на машинах, на которых вы регистрируетесь в процессе работы.
- Теперь с хоста **Remote_host**, вы можете подключиться к хосту **Next_remote_host**, используя ключи, загруженные в *ssh-agent*, на хосте **Local_host**.

Это только на первый взгляд сложно выглядит, вся эта схема работает абсолютно прозрачно для пользователя, от него требуется только соответствующим образом настроить */etc/ssh/ssh_config* а дплее все просто. Собственно тут даже показывать нечего в качестве примера.

Программа PuTTy, клиент SSH под Windows.

Все о чем было рассказано выше, говорилось и делалось в контексте операционной системы FreeBSD, теперь пару слов о клиентах под Windows. Самым популярным, и на мой взгляд, самым функциональным и удобным клиентом под операционные системы Windows, является программный пакет **PuTTy**. В него входят следующие утилиты:

- **putty.exe** - Программа, SSH, Telnet, Rlogin клиент;
- **puttygen.exe** - Утилита для генерации и конвертации ключей;
- **pageant.exe** - Аналог программы ssh-agent;
- **plink.exe** - Клиент SSH, Telnet и Rlogin для командной строки;
- **pscp.exe** - Программа командной строки для безопасного копирования SCP/SFTP
- **psftp.exe** - Интерактивный SFTP клиент для командной строки;
-

Не думаю что у вас возникнут трудности при использовании данной программы, тем не менее, всегда можно нажать кнопку Help).

Комментарии

Опубликовано Анонимный (не проверено) в Срд, 01/19/2011 - 23:36.

Логин от рута - это не есть хорошо.

ответить

> *Опубликовано Анонимус (не проверено) в Чтв, 01/20/2011 - 00:15.*
>
> есть такой грешок, предпочитаю под рутом работать
>
> ответить

Опубликовано Анонимный (не проверено) в Чтв, 06/02/2011 - 13:38.

Вы пишите очень грамотно и толково...
Правда - именно такой документации и не хватает...
Какой options что именно делает, как происходит то или иное действие как отрабатывает тот или иной демон

Конечно - все есть на английском, но если не силн в иностранном языке... спасибо вам.

ответить

> *Опубликовано Sergey (не проверено) в Вск, 06/05/2011 - 12:51.*
>
> Рад что пригодилось)
>
> ответить

Опубликовано Анонимный (не проверено) в Срд, 09/14/2011 - 10:16.

Как можно узнать пароль доступа к приватному ключу ? Есть пара ключей , но пароль безвозвратно забыт

ответить

> *Опубликовано sergey (не проверено) в Срд, 09/14/2011 - 11:48.*
>
> никак
>
> ответить

Опубликовано Павел (не проверено) в Срд, 09/28/2011 - 03:39.

Очень хорошая статься, спасибо

ответить

Опубликовано oleg (не проверено) в Срд, 10/26/2011 - 10:09.

> Публичный ключ копируется на удаленный SSH сервер (говорим опять-же в контексте операционной системы FreeBSD) и кладется в специальный файл, известный SSH серверу. По-умолчанию, для хранения публичных ключей, используется файл ~/.ssh/authorized_keys. Файл для хранения ключей назначается в файле конфигурации SSH сервера, директивой AuthorizedKeysFile

Мне кажется тут ошибочка "~/.ssh/authorized_keys" может так должно быть " .ssh/authorized_keys" ?

ответить

Опубликовано sergey (не проверено) в Птн, 10/28/2011 - 13:29.

не, ошибки нет, можно написать и так
"/root/.ssh/authorized_keys"

ответить

Опубликовано oleg (не проверено) в Птн, 10/28/2011 - 14:10.

Да вы правы, смутила меня строчка в sshd_config
AuthorizedKeysFile .ssh/authorized_keys

ответить

Опубликовано garrotte в Птн, 10/28/2011 - 15:05.

аа, ну он относительно домашней папки root'а считает, поэтому и сбило с толку :)

ответить

Опубликовано Igor (не проверено) в Птн, 12/16/2011 - 15:58.

Громадное спасибо за труд. Наконец-то нашел все что искал и в одном документе. Продолжайте в томже духе)

ответить

Опубликовано Аноним (не проверено) в Суб, 01/21/2012 - 10:37.

Thank You, it very helpful documentation on mother language! It better than it same on opennet (for example) You go true way!)))

ответить

Опубликовано sergey (не проверено) в Суб, 01/21/2012 - 12:33.

thanks :)

ответить

Опубликовано Vladimir (не проверено) в Чтв, 05/24/2012 - 17:29.

Вроде все получилось, но запрос остался

$ ssh client@192.168.2.15
Enter passphrase for key '/usr/home/client/.ssh/id_rsa':

Куда копать?

ответить

Опубликовано sergey (не проверено) в Птн, 05/25/2012 - 01:09.

что говорит

```
ssh-add -l
```

?

ответить

Опубликовано Vladimir (не проверено) в Птн, 05/25/2012 - 11:10.

ssh-add -l
Could not open a connection to your authentication agent.

ответить

Опубликовано Vladimir (не проверено) в Птн, 05/25/2012 - 11:28.

Большое спасибо! Все получилось и работает!

ответить

Опубликовано Vladimir (не проверено) в Птн, 05/25/2012 - 12:18.

Каким образом решить это все через скрип, т.к. время жизни ключа ограничено сеансом н-р: root иначе получим сообщение ?

ssh-add -l
Could not open a connection to your authentication agent.

И каким образом передать параметр passphrase for /root/.ssh/id_rsa в этот скрип?

Задача запуск этого скрипта в crop. Спасибо!

ответить

Опубликовано sergey (не проверено) в Птн, 05/25/2012 - 13:03.

сомневаюсь что это возможно, во всяком случае не пробовал
разве что генерить ключи без пароля

ответить

Опубликовано Vladimir (не проверено) в Птн, 05/25/2012 - 18:01.

Сделал скрипт на Expect. Может пригодится! :)

1. Ставим Expect из портов /usr/ports/lang/expect

2. Создаем скрипт /home/Client_ssh/my_ssh_client.sh
Содержимое my_ssh_client.sh
#!/usr/local/bin/expect

#доменное имя или IP сервера куда конектимся
set ip_server 192.168.xx.xx

set date [timestamp -format "%C%y-%m-%d %X"]

Наш пароль passphrase созданный ssh-keygen -t rsa -b 4096

set ssh_passphrase xxxxx

путь к файлу который хотите скопировать /путь куда копировать
spawn scp user_ssh@$cp_server:/home/user/dump.sql /home/Client_ssh/dump-$date.sql

Ждем сообщение
expect "Enter passphrase for key '/home/Client_ssh/.ssh/id_rsa':"

Посылаем Наш passphrase
send $ssh_passphrase\r

expect eof
Конец скрипта

3. пихаем это все в crop и запускаем от пользователя Client_ssh

Проверенно работает!

ответить

Опубликовано Vladimir (не проверено) в Птн, 05/25/2012 - 18:04.

только вместо $cp_server -> ip_server

spawn scp user_ssh@$ip_server:/home/user/dump.sql /home/Client_ssh/dump-$date.sql

ответить

Опубликовано sergey (не проверено) в Птн, 05/25/2012 - 18:35.

хм, спасибо, может и пригодится :)

ответить

Опубликовано ilyouha (не проверено) в Пнд, 12/03/2012 - 17:26.

Хочу сказать **спасибо** за эту статью и за вот эту - http://vds-admin.ru/unix-toolbox/rsync

там нет возможности коментировать

ответить

Опубликовано garrotte в Втр, 12/04/2012 - 11:45.

пожалуйста)
там я комменты решил не включать)

ответить

Опубликовано Макс (не проверено) в Чтв, 08/28/2014 - 18:41.

Помогите, пожалуйста!
Работаю под виндой, в качестве консоли использую MINGW32 который поставляется вместе с Git. И при каждом новом сеансе приходится делать eval `ssh-agent` и ssh-add. Ключи храню в стандартной директории ~/.ssh/id_rsa ...
Уверен что можно настроить так, чтобы эти команды можно было не вводить каждый раз. Но как настроить?
Ведь ключи предназначены в том числе и для удобства (не вводить каждый раз пароль), а всё равно приходится вводить каждый раз одно и тоже (eval...). =/

ответить

Опубликовано Анонимный (не проверено) в Чтв, 08/28/2014 - 18:59.

не сталкивался с MINGW32, но думаю можно набросать скрипт в пару строк, но пароль для ключа все равно вводить придется (если конечно запаролен)

ответить

Опубликовано Макс (не проверено) в Чтв, 08/28/2014 - 19:01.

Конечно запаролен ;)

Капча убивает =)

ответить

Опубликовано Макс (не проверено) в Чтв, 08/28/2014 - 19:00.

нашел ответ.
Работает следующее:

```
$ cd ~
$ cat > .bashrc
#! /bin/bash
eval `ssh-agent -s`
ssh-add ~/.ssh/*_rsa
```

ответить

Опубликовано Анонимный (не проверено) в Чтв, 08/28/2014 - 19:19.

а, ну логично)

ответить

Опубликовано Анонимный (не проверено) в Вск, 10/19/2014 - 01:36.

И все таки, где должны хранится ключи? в /root/.ssh/authorized_keys или же /home/USER/.ssh/authorized_keys? И нужно ли удалять id_rsa после генерирования путти в ppk? Должны ли совпадать имена файлов *.pub и *.ppk? В /etc/ssh/sshd_config AuthorizedKeysFile указывать %u/.ssh/authorized_keys - по умолчанию там так написано или .ssh/authorized_keys или /home/USER/.ssh/authorized_keys/id_rsa.ppk - как я год назад на дебиане делал и оно работало???

ответить

Опубликовано Анонимный (не проверено) в Вск, 10/19/2014 - 01:37.

ой, не /home/USER/.ssh/authorized_keys/id_rsa.ppk, а /home/USER/.ssh/authorized_keys/id_rsa.pub

ответить

Опубликовано Анонимный (не проверено) в Вск, 10/19/2014 - 02:02.

ключи хранятся в домашней папке юзера, для которого они предназначены, то есть /home/USER/.ssh/authorized_keys
по поводу имен файлов, это условность, обзывать их можно как угодно
по поводу хранить или нет, смотря для чего их генерили и пригодятся-ли они еще
или я что-то не так понял ?

ответить

Опубликовано Анонимный (не проверено) в Вск, 10/19/2014 - 02:03.

в случае /root/.ssh/authorized_keys - это для рута

ответить

Опубликовано Анонимный (не проверено) в Вск, 10/19/2014 - 11:27.

Вот, пишу, что я делаю по порядку:
1. Запустил VBox(ubuntu Server 14.04.1) - открытый доступ к серверу из хостовой машины на VBox - норм заходит, без проблем.
2. Ключи будут храниться в /home/nehalem/.shh/authorized_keys - создал папки
3. Генерю ключ ssh-keygen -t rsa
4. Ключи создались в /home/nehalem/.ssh (id_rsa id_rsa.pub)
5. Перемещаю ключи в /home/nehalem/.shh/authorized_keys (sudo mv id_rsa authorized_keys sudo mv id_rsa.pub authorized_keys)
6. sudo nano /etc/ssh/sshd_config:

Authentication:
LoginGraceTime 120
PermitRootLogin without-password
#PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile ~/.ssh/authorized_keys

ChallengeResponseAuthentication yes
PasswordAuthentication yes

7. Запускаю psftp.exe:
- open 192.168.56.10 (авторизовуюсь - логин, пароль)
- качаю себе id_rsa (get id_rsa)
- В puttygen.exe загружаю id_rsa - сохраняю private key в формате id_rsa.ppk
8. Перезагружаю сервер
9. Открываю putty.exe, в Connection - SSH - Auth указываю файл id_rsa.ppk - подключаюсь:
Просит ввести логин (хотя я указал в Connection Rloagin) ввожу
Ответ:
Server refused our key
Using keyboard-interactive authentication.
Просит ввести пароль, ввожу - вхожу, но не по ключу...

ответить

Опубликовано VladimirT (не проверено) в Втр, 12/02/2014 - 09:04.

"5. Перемещаю ключи в /home/nehalem/.shh/authorized_keys (sudo mv id_rsa authorized_keys sudo mv id_rsa.pub authorized_keys)"

Надо добавить публичный ключ командойS sudo cat id_rsa.pub >> authorized_keys , т.к. командой "mv" ты заменяешь один файл на другой.

ответить

Опубликовано Игорь (не проверено) в Срд, 04/01/2015 - 19:05.

Здравствуйте, замечательная статья, только вот по какой причине вы не отключаете авторизацию по паролю? ведь это снижает безопасность, так как пароль можно будет сбрутить.

ответить

Опубликовано Анонимный (не проверено) в Срд, 04/01/2015 - 22:02.

когда все зависит от меня, отключаю, но клиентам как правило лень париться насчет ключей и проще ходить по паролю)

ответить

Опубликовано Игорь (не проверено) в Чтв, 04/02/2015 - 15:53.

эм а чего в статье не описано как? (я знаю как просто за державу обидно :-D)

ответить

Опубликовано Анонимный (не проверено) в Чтв, 04/02/2015 - 16:35.

да я как-то "по реальным событиям" строчил, поэтому не упомянул)

ответить

Опубликовано Анонимный (не проверено) в Втр, 07/14/2015 - 12:49.

Отличная статья! Все поехало!
Но есть небольшая путаница: Все ключи нужно создавать на удаленной машине(потом копировать их на сервер) и агента тоже запускать на удаленной машине. Автору респект!

ответить

Отправить комментарий

Ваше имя: *
Анонимный

E-mail: *

Содержание этого поля является приватным и не предназначено к показу.

Домашняя страница:

Комментарий: *

Уведомлять меня о новых комментариях

Регистр имеет значение

```
            oooo                    .oooooo..o  oooooo     oooo   .o88o.
             `888                  dP'    `Y8   `888.     .8'    888  "
 oooooooo    888 .oo.    oooo    ooo Y88bo.        `888.   .8'    o888oo
d'""7d8P     888P"Y88b    `88.  .8'   `"Y8888o.     `888. .8'      888
   .d8P'     888   888     `88..8'        "Y88b     `888.8'       888
 .d8P'  .P   888   888      `888'      oo     .d8P    `888'       888
d8888888P   o888o o888o      `8'       8""88888P'      `8'       o888o
```

Введите код проверки: *

Введите код, изображенный в стиле ASCII-арт.