**110.3 Защита данных при помощи шифрования**

Студент должен уметь пользоваться технологиями публичных ключей для защиты данных и обмена информацией.

**Изучаем**:

* использование пары ключей для шифрования данных;
* настройку клиента openssh;
* создание туннелей при помощи openssh;
* шифрование данных при помощи gnupg.

В данном уроке рассматриваются принципы шифрования и подписи с использованием открытого и закрытого ключа. Закрытый (частный) ключ находится только у владельца, при помощи него можно подписать документы и расшифровать файлы. Открытый (публичный) ключ находится в общем доступе, при помощи него можно зашифровать файлы или проверить подпись.

Для управления удаленным сервером Linux часто используется безопасная оболочка SSH. Для подключения к серверу можно воспользоваться командой **ssh**, например:

***ssh admin@10.0.1.6*** *(подключиться к серверу 192.168.0.3 от имени admin);*

Для возможности подключения на удаленной стороне должен быть запущен демон SSH, настроенный на прием входящих запросов. Первое подключение к незнакомому серверу дает нам возможность скачать его публичный ключ для проверки подлинности сервера в дальнейшем.

Сохраненные доверенные открытые ключи хранятся в файле **~/.ssh/known\_hosts**. Если при подключении к удаленному серверу его публичный ключ уже находится в файле ***known\_hosts***, подключение пройдет без предупреждений. Если ключа нет, или сервер был подменен – появится предупреждение.

Файл **~/.ssh/known\_hosts** используется для проверки ключей пользователем, в домашнем каталоге которого он находится. Для проверки ключей на уровне всей системы, вне зависимости от пользователя, используется файл **/etc/ssh/ssh\_known\_hosts**

**\_\_\_**

При работе с ключами чаще всего используются два криптографических алгоритма: RSA (используется для подписи и шифрования) и DSA (только для подписи).

Для того чтобы создать собственную пару ключей используется команда **ssh-keygen -t rsa**(везде далее по тексту можно использовать *dsa* вместо *rsa*). После ее выполнения в каталоге **~/.ssh/** создается пара: открытый и закрытый ключ.

Для того чтобы отправить открытый ключ на удаленный сервер (чтобы он начал доверять нашей системе во время подключения и не запрашивал логин и пароль для входа) используется команда **ssh-copy-id**, например:

***ssh-copy-id -i .ssh/id\_rsa.pub admin@10.0.1.6*** *(послать открытый ключ id\_rsa.pub на машину 10.0.1.6 пользователю admin );*

После отправки открытого ключа авторизация на удаленном сервере происходит автоматически (или после ввода парольной фразы, если таковая была создана в момент генерации пары ключей).

Открытые ключи, по которым удаленная система доверяет входящим подключениям, находится на удаленной системе в каталоге **~/.ssh/authorized\_keys**удаленного пользователя.

Для генерации пары ключей для всей системы, а не отдельных пользователей, можно воспользоваться командой **sudo** **ssh-keygen -t dsa -f /etc/ssh/ssh\_host\_dsa\_key**

Современные графические среды хранят в памяти используемые секретные ключи. Для того чтобы не вводить ключ постоянно в течении одного сеанса в консоли используется конструкция:

***ssh-agent bash*** *(запустить консоль родителем ssh-agent)****;***

***ssh-add .ssh/id\_rsa*** *(добавить в память текущего процесса закрытый ключ).*

\_\_

SSH позволяет организовать туннели, передавая по ним данные в зашифрованном виде. Существует несколько вариантов использования туннелей, например проброс локального порта на удаленную машину или удаленного порта на локальную.

Для запуска графических приложений с удаленного сервера используется ключ **–X**, например:

***ssh –X 10.0.1.6*** *(запустить сеанс ssh с удаленным сервером 10.0.1.6 перенаправляя X11);*

***firefox*** *(запустить на локальной машине окно firefox с удаленного сервера).*

Для проброса удаленного порта на локальную машину используется ключ **–L**, например:

***ssh 10.0.1.6 –N –L 12345:213.180.204.3:80*** *(установить соединение с удаленного сервера 10.0.1.6 на удаленный сервер 213.180.204.3 по порту 80 и пробросить его на локальный порт 12345).*

Для копирования файлов по ssh используется утилита **scp**, например:

***scp text.txt admin@10.0.1.6:/home/admin*** *(скопировать из текущего каталога файл text.txt на удаленный сервер 10.0.1.6 в директорию /home/admin);*

***scp admin@10.0.1.6:/home/admin/text.txt /home/semaev*** *(скопировать с удаленного сервера 10.0.1.6 файл /home/admin/text.txt в локальный каталог /home/semaev);*

***\_\_***

Для шифрования документов также используется пара ключей. Существует несколько реализаций программ шифрования информации, в данном случае рассматривается свободно распространяемое ПО GPG.

Для управления шифрованием используется утилита **gpg**, например:

***gpg --gen-key*** *(сгенерировать пару ключей);*

***gpg --list-key*** *(показать доступные ключи);*

***gpg --export semaevkey > gpg.public*** *(экспортировать открытый ключ в двоичный формат);*

Для того чтобы использовать открытый ключ для шифрования информации, его нужно скачать в необходимую систему, например, при помощи утилиты **scp**. Владея открытым в двоичном формате, можно начать его использования для шифрования информации, например:

***gpg --import gpg.public*** *(импортировать полученный ключ);*

***gpg --list-key*** *(проверить доступные ключи);*

***gpg --out secret.file --recipient ‘Kirill Semaev’ --encrypt file.txt*** *(зашифровать файл file.txt в файл secret.file используя открытый ключ ‘Kirill Semaev’);*

Если отправить созданный зашифрованный файл владельцу закрытого ключа, он сможет расшифровать его, например:

***gpg --out some.file --decrypt secret.file*** *(расшифровать файл secret.file в файл some.file).*