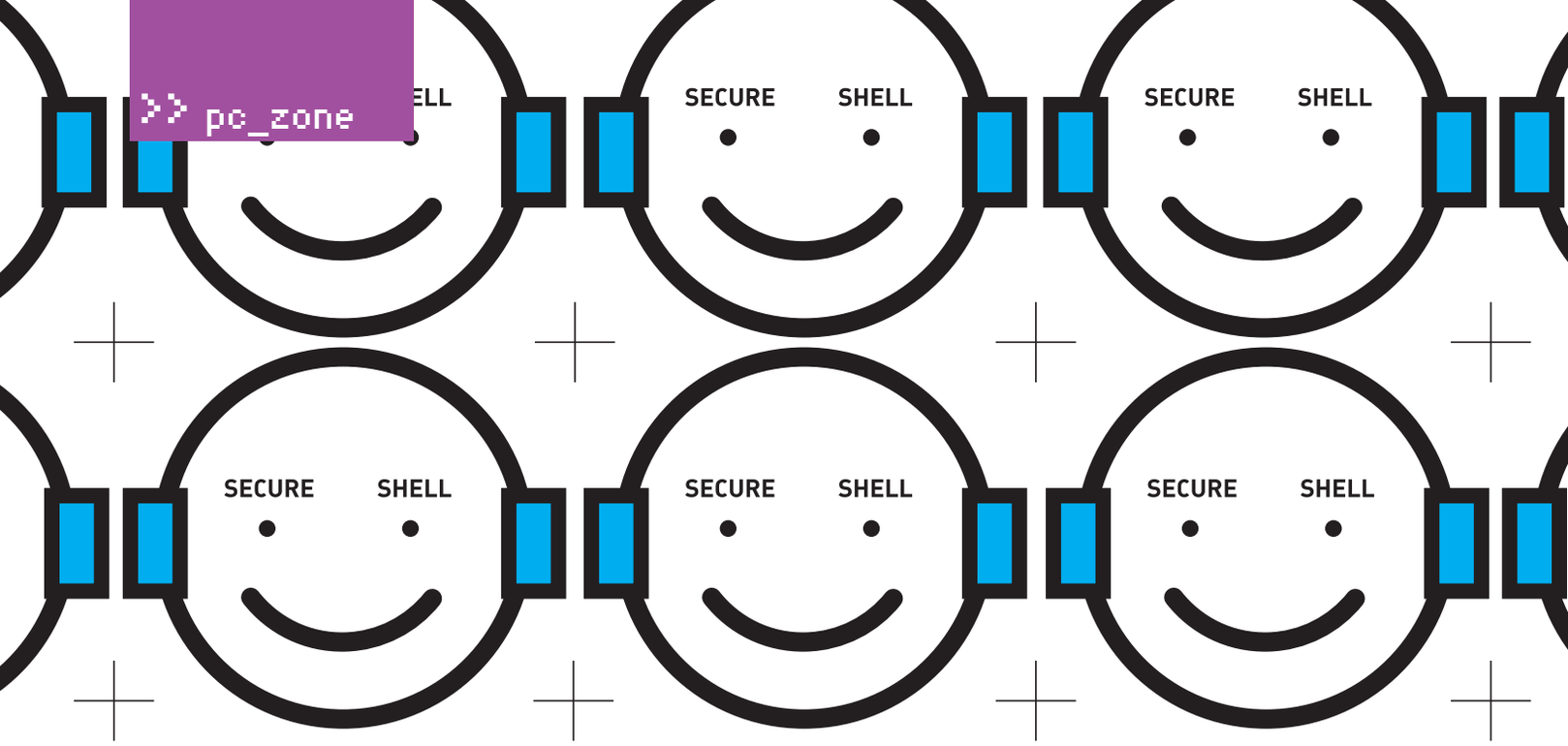


СТЕПАН «STEP» ИЛЬИН
/ STEP@GAMELAND.RU /

КАК СТАТЬ SSH'АСТЛИВЫМ

FULL-GUIDE ПО ИСПОЛЬЗОВАНИЮ SECURE SHELL

Стой! Не листай дальше. Если ты до сих пор воспринимаешь SSH исключительно как безопасную альтернативу устаревшему Telnet, не рискуй вызывать гнев богов, тьфу, разработчиков протокола. Ниже мы собрали самый полный мануал по правильному использованию Secure Shell на полную катушку.

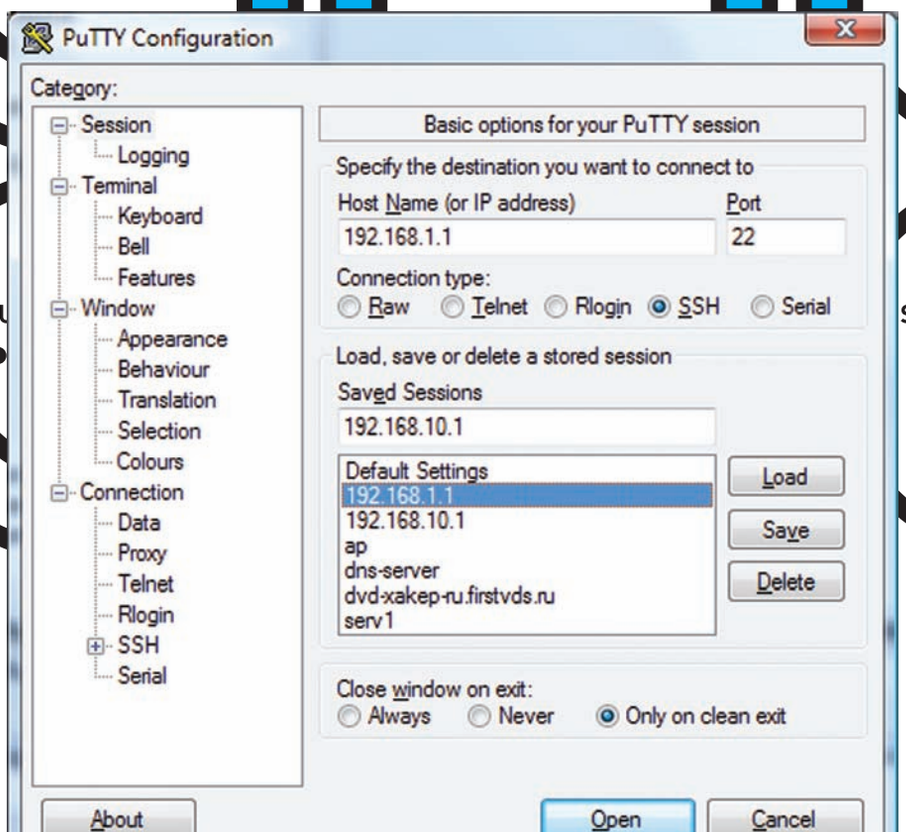
ТРИК 1: ПРОКАЧИВАЕМ SSH-КЛИЕНТ

Несмотря на большое разнообразие SSH-клиентов, особой проблемы с выбором не возникает. Общепризнанных всего два — **PuTTY** (www.chiark.greenend.org.uk) и **SecureCRT** (www.vandyke.com), и оба действительно хороши. Но если за «цитрамон» разработчики просят денежки, то PuTTY распространяется прямо в открытых исходниках. По этой причине выбор зачастую остается именно за ним. Более того, несмотря на то, что многие воспринимают путти как виндовый клиент, у него есть версии и для UNIX. Саму прогу ты видел в действии, когда смотрел ролики Visualhack++. С помощью него ты можешь коннектиться к своим сервакам через: Raw, Telnet, Rlogin, FTP (SFTP), SSH1, SSH2. В общем смысле, PuTTY — это комплект утилит, куда помимо непосредственно клиента (putty.exe) входят тулзы:

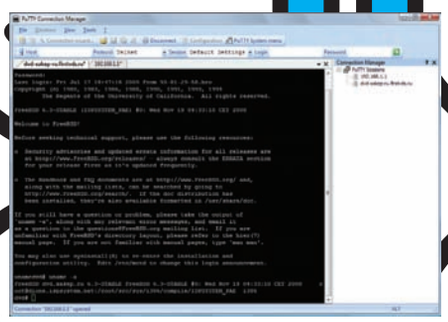
- **puttygen** — генератор rsa/dsa ключей, используемых для авторизации;
- **pagent** — агент аутентификации, который хранит ключи в памяти, благодаря чему ты освобождаешься от ввода паролей ручками;
- **plink** — интерфейс командной строки для putty;
- **pscp** — утилита, обеспечивающая безопасное копирование файлов;
- **psftp** — безопасный ftp-клиент для копирования, просмотра, переименования файлов и т.д.

С некоторыми из этих утилит мы еще познакомимся далее.

Невзирая на личную симпатию к PuTTY, долгое время я отдавал предпочтение SecureCRT. Почему? По большому счету — за одну маленькую, но очень полезную опцию, не реализованную в патти — поддержку табов для разных сессий. Если у тебя когда-нибудь было открыто пять, а то и больше окошек PuTTY для разных серверов, ты знаешь, насколько тяжело ориентироваться среди них. Разработчики не спешат добавлять поддержку табов в утилиту, но зато с этим справилась группа французских энтузиастов, выпустив замечательную тулзу **PuTTY Connection Manager** (puttycm.free.fr). Что важно, это не какая-то там переделка исходников оригинального PuTTY, которая могла повлечь за собой новые баги, в том числе и безопасности. Напротив, за SSH-сесии по-прежнему отвечает исходный бинарник (putty.exe), а PuTTY Connection Manager лишь объединяет открытые окна в удобный интерфейс с табами, а также предоставляет продвинутый интерфейс для настроек подключения. Поддержка табов — это не единственный конек написанной на C# надстройки над PuTTY. После недели использования с трудом можешь представить жизнь без полезных опций:



ДЛЯ ПОДКЛЮЧЕНИЯ УКАЗЫВАЕМ ПАРАМЕТРЫ СОЕДИНЕНИЯ С СЕРВЕРОМ ИЛИ ВЫБИРАЕМ НУЖНЫЙ ПРОФИЛЬ



PUTTY CONNECTION MANAGER

ГЕНЕРИРУЕМ ПРИВАТНЫЙ И ОТКРЫТЫЙ КЛЮЧИ

- сворачивание в трей;
- автоматический логин без необходимости ввода пароля. Надо заметить, что стандарт не позволяет производить подобные действия, но в обход используется эмуляция ввода с клавиатуры пользователем;
- выполнение произвольных команд после успешной авторизации в системе;
- менеджер соединений, позволяющий задать для каждого из серверов отдельные параметры;
- шифрование файла с настройками с помощью AES; правда, для этого требуется установить дополнительную DLL-библиотеку.

ТРИК 2: ПОСТИГАЕМ ПРЕМУДРОСТИ АВТОРИЗАЦИИ

Самый простой способ авторизироваться на удаленном сервере — использовать связку логин/пароль. Понятно, что если к серверу коннетишься раз в день, то набрать связку вручную не составит труда (при условии, что помнишь их). PuTTY для каждого подключения позволяет сохранить настройки. В PuTTY ты можешь создавать профили для различных SSH-серверов, так что не придется вбивать настройки для конкретного сервера, когда ты захочешь к нему очередной раз подсоединиться. В таком профиле, например, можно ввести логин, который будет использоваться для входа. Давай попробуем создать профиль для сервера. Для этого переходим в категорию Sessions. Здесь вводится IP-адрес или имя хоста, порт, а также протокол. Можно указать имя пользователя для подключения, под которым ты хочешь заходить в систему. Перейди в «Connection → Data» и укажи в «Auto-login username» имя пользователя (например, UserAcc). Затем снова иди в категорию Sessions. Под надписью Saved Sessions (сохраненные сессии) введи имя профиля, например, session1, после чего кликай на Save. В следующий раз, когда будешь запускать PuTTY, просто выбери подходящий профиль из Saved Sessions, кликни Load и Open. Причем, имя пользователя введется автоматически. Стандарт на протокол SSH запрещает сохранять пароль, но позже мы научимся обходить это ограничение.

Для авторизации на удаленной системе можно сгенерировать и использовать пару ключей (открытый/закрытый) для SSH-подключения к удаленной системе. В архив с программой входит дополнительная утилита PuTTYgen, которая поможет сгенерировать открытый и приватный ключ. Открытый ключ, как уже говорилось, необходимо передать на удаленный сервер. В случае виндового сервера — путь к нему достаточно указать в настройках аккаунта. Под никсами и OpenSSH необходимо вставить в файл /.ssh/authorized_keys2 ключ в одну строку:

```
mkdir ~/.ssh
chmod 700 ~/.ssh
vi ~/.ssh/authorized_keys2
ssh-dss AAAAB3NzaC1kc3MAAAE [.
. .] Huw2FekFNM7pMgEQi57k= dsa-
key-20061205
chmod 600 ~/.ssh/authorized_keys2
```

Файл должен читаться/правиться только данным пользователем, поэтому последней командой мы устанавливаем нужные права доступа.

Что касается закрытого ключа, путь к нему требуется указать в настройках нужной сессии клиента (SSH → Auth → Private key file for authentification). Добавлю, что даже при использовании пары ключей приходится каждый раз вводиться секретную фразу. Это сильно раздражает при частых коннектах. От проблемы может избавить утилита Pageant, которая также входит в стандартный комплект PuTTY.

Помимо этого, можно воспользоваться так называемым **sshproxy** (sshproxy-project.org/about), написанным на Python. Тулза позволяет подключаться к удаленным хостам без необходимости ввода паролей или ключей. По сути, это маленький демон, который сидит в локалке или DMZ-зоне. Когда пользователь коннектится к нему с помощью SSH-клиента, то sshproxy авторизирует его и проверяет права для доступа к нужному сайту. Если клиенту это разрешено, прокси выполняет соединение на удаленный сайт, используя пароль или ключ, сохраненные в его базе данных.

ТРИК 3: ПРОБРАСЫВАЕМ ТУННЕЛИ

Помимо доступа к удаленной командной строке, SSH предоставляет ряд других возможностей. Первая — это туннелирование. После того, как установлено SSH-соединение, можно безопасно роутить через туннель трафик одного или сразу нескольких приложений. Это

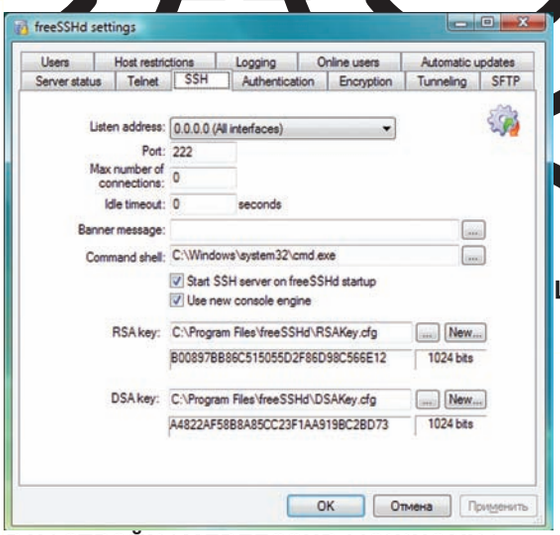
info

Утилиты для брута SSH:

- SSH Brute Forcer (www.securiteam.com/tools/5QP0L2K60E.html)
- SSHatter (freshmeat.net/projects/sshatter)
- SSH BruteForcer (www.darkc0de.com/bruteforce)
- THC Hydra (www.thc.org/thc-hydra)

dvd

Все утилиты для реализации триксов ты найдешь на нашем DVD.



В НАСТРОЙКАХ FREESSHD МОЖНО УКАЗАТЬ ТЕКСТ ПРИВЕТСТВЕННОГО БАННЕРА

не только позволяет обойти файрвол, но еще и гарантированно скроет данные от прослушивания. Туннелинг сейчас поддерживают любые клиенты и серверы. Объясню смысл на примере нашего любимого PuTTY.

Для конфигурации туннеля с помощью PuTTY нужно:

- в окне конфигурации подключения в категории «Session» указать Host Name (твой_ssh_сервер), Port (22), Protocol (SSH);
- в категории «Connection/SSH/Tunnels», в секции «Add new forwarded port», указать «Source port» (локальный_порт, например, 666), Destination (адрес_прокси_или_сервера:3306);
- выбрать пункт Local и нажать кнопку «Добавить».

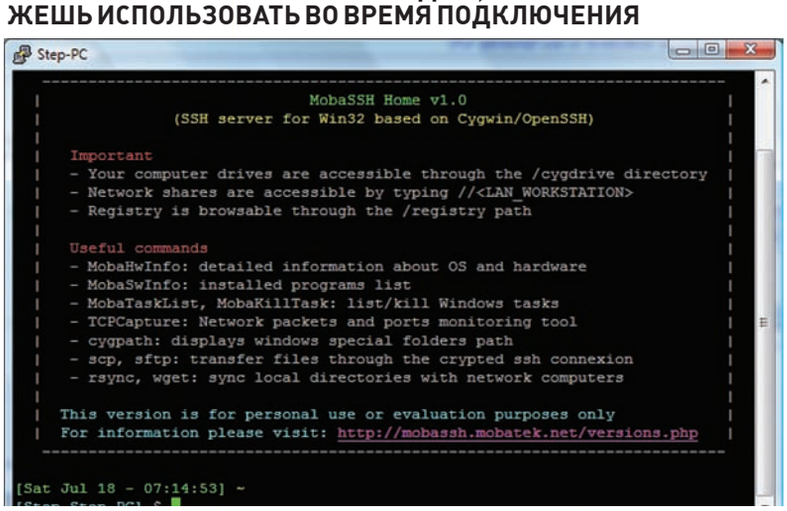
После установления соединения можно запускать браузер, указав в качестве прокси 127.0.0.1 и порт, указанный в качестве Source Port (например, 666).

В unix-системе достаточно набрать команду:

```
ssh -L666:адрес_прокси_или_сервера:порт -n
имяпользователя@адрес_ssh_сервера
```

Аналогичным образом можно поднять туннель до MySQL-сервера, пробросить VNC-сессию до удаленного рабочего стола и т.д.

СПРАВКА ПО ПОЛЕЗНЫМ КОМАНДАМ, КОТОРЫЕ ТЫ МОЖЕШЬ ИСПОЛЬЗОВАТЬ ВО ВРЕМЯ ПОДКЛЮЧЕНИЯ



ДЛЯ ЗАПУСКА И СТАРТА SSH-СЕРВЕРА С ПОМОЩЬЮ MOBASSH ДОСТАТОЧНО ОДНОГО КЛИКА МЫШИ

ТРИК 4: БЕРЕМ НА ВООРУЖЕНИЕ 2-HOP TUNNEL

Что такое «поднял 2-хоповый ssh тунель (2-hop ssh tunnel)»? SSH часто используется как транспортный протокол для безопасной передачи данных между другими приложениями, например, небезопасного VNC (удаленный рабочий стол). Однако бывают ситуации, когда установить туннель невозможно: например, между двумя хостами нет возможности прямого подключения (банально из-за ограничений файрвола). Если ввести некоторый хост, с которым подключение может установить каждая из сторон, то его реально использовать как посредника, прибегнув к приему two hop tunneling (или, проще говоря, — туннель через дополнительный гейт). Достигается это так: сперва мы используем ssh, чтобы переадресовать трафик на порт той машины, с которой возможно установить соединение, и далее заставляем ее переадресовывать трафик на нужный нам хост (с которым для нее также возможен коннект). В следующем примере мы будем осуществлять подключения с машины «myhome.example.org», в качестве промежуточного хоста будет выступать «gateway.example.com», а в роли желанной машины будет недоступный напрямую SSH-демон на «server.example.com».

Наша задача — создать двух-хоповый туннель. Для этого на машине «myhome.example.org» запускаем команду:

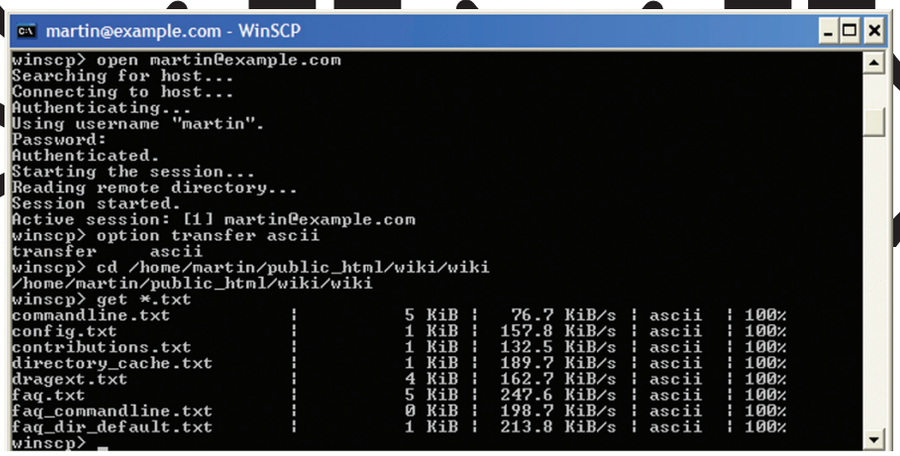
```
ssh -f -N -L 51526:server.example.com:22 -2
gateway.example.com
```

Вот и все! В результате, SSH-подключения на 51526 порт на машине myhome.example.org будут туннелироваться на нужный хост (server.example.com). Другими словами, вместо невозможного напрямую соединения на server.example.com:22, мы просто подключаемся на локальный хост и порт 51526, а все заморочки возьмет на себя механизм SSH.

Клиенты для мобильных устройств

Symbian:
PuTTY for Symbian OS (s2putty.sourceforge.net)
Windows Mobile:
PocketPuTTY (www.pocketputty.net)
Java:
MidpSSH (www.xk72.com/midpssh)
iPhone:
iSSH (www.zinger-soft.com)

ТУЛЗА ДЛЯ ДОСТУПА К REMOTE DESKTOP'У ЧЕРЕЗ SSH



WINSCP ПОЗВОЛЯЕТ АВТОМАТИЗИРОВАТЬ ЧАСТЬ РУТИННОЙ РАБОТЫ

Кстати, в качестве порта можно использовать и любой другой, но желательно из диапазона 49152-65535.

ТРИК 5: ПОДНИМАЕМ SSH-СЕРВЕР ПОД ВИНДОЙ

С никсами все просто. Чуть ли не стандартом де-факто является всем известный OpenSSH, да и практически в любом дистрибе он установлен по умолчанию. В условиях ограниченных ресурсов (на старых компьютерах, аппаратных роутерах, точках доступа и т.д.) зачастую устанавливают **DropBear** (matt.ucc.asn.au/dropbear/dropbear.html). Под виндой, впрочем, поднять SSH-сервер — тоже не бог весть, какая проблема. Для тех же самых OpenSSH и DropBear есть полноценные порты, но их трогать не будем. Обходим стороной и продвинутый, но платный **WinSSHD** (www.bitvise.com/winsshd). В сравнении с WinSSHD программа MobaSSH (mobassh.mobatek.net) чарует своей простой. Все, что требуется для запуска полноценного SSH-сервера с авторизацией, используя аккаунты пользователей в системе — это нажать одну кнопку «Install». В системе тут же появится новая служба.

После соединения с MobaSSH и получения приветствия демона становится понятно, что это не что иное, как сильно переработанный порт OpenSSH, собранный с помощью компилятора Cygwin. Для навигации по системе используются никсовые команды (ls вместо dir для отображения содержимого текущего каталога и т.д.). Имей в виду некоторую специфику демона. Все локальные диски доступны через директорию /cygdrive. Достучаться до сетевых ресурсов можно, используя привычный адрес в UNC-формате: //<LAN_WORKSTATION>, а вносить изменения в реестр — через директорию /registry.

Помимо этого есть ряд полезных команд:

```
MobaHwInfo: детальная информация об ОС и железе
MobaSwInfo: список установлено в системе софта
MobaTaskList, MobaKillTask: список процессов и удаление нужного
TCPCapture: сетевой монитор
scp, sftp: передача данных по криптованному ssh-соединению
rsync, wget: синхронизация локальных папок с сетевыми ресурсами
```

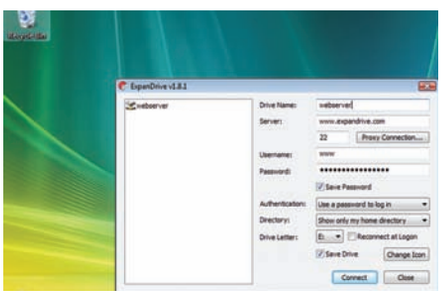
MobaSSH на 100% совместим со всеми никсовыми и виндовыми клиентами. Полностью с нуля, а поэтому и без всяких юниксовых замашек написан бесплатный **freeSSHd** (www.freesshd.com). С установкой также не возникнет проблем; причем, как будет работать демон, в виде сервиса системы или обычного приложения, предоставляется на выбор пользователю. Точно так же можно выбрать и оболочку — по умолчанию выбирается стандартный cmd.exe. Вообще, настроек не то, чтобы много, но как раз достаточно, чтобы все настроить под себя, включая авторизацию пользователя, приветственный баннер, параметры туннелирования, SFT и т.д.

ТРИК 6: НАЛАЖИВАЕМ НЕПРОБИВАЕМЫЙ КОННЕКТ

Все то же самое можно сделать и при помощи консольной утилиты Plink, которая входит в комплект с PuTTY. Любые параметры реально задать через командную строку с помощью различных ключей, а можно использовать настройки, сохраненные в конкретной сессии. Так и поступим:

```
plink my-ssh-session
```

По разным причинам соединение может иногда «падать». Будь уверен, упадет оно в самый неподходящий момент. Чтобы этого не произошло, отслеживай состояние подключения и вновь устанавливай его при необходимости. Когда-то подобные скрипты я писал вручную, но сейчас есть отличная утилита **MyEnTunnel** (nemesis2.qx.net/pages/MyEnTunnel). Она незаметно сидит в трее и поддерживает активными все необходимые SSH-туннели. Принцип прост: тулза отслеживает процесс Plink. Если процесс умирает (соединение оборвано, сервер перегрузился или по какой-то еще причине удаленный хост стал недоступен), MyEnTunnel автоматически перезапустит Plink. Системные ресурсы при этом используются по минимуму. Юзер вправе сам указать, как часто нужно проверять наличие коннекта: в самом скромном режиме «Slow Polling» MyEnTunnel проверяет соединение раз в секунду. Несмотря на то, что тулза написана под винду, она отлично чувствует себя с Wine'ом и под никсами.



МОНТИРУЕМ ФАЙЛЫ С УДАЛЕННОГО ХОСТА В ВИДЕ ЛОГИЧЕСКОГО ДИСКА В СИСТЕМЕ

ТРИК 7: ИСПОЛЬЗУЕМ БЕЗОПАСНУЮ ПЕРЕДАЧУ ФАЙЛОВ

Когда мы говорим об SSH, не стоит забывать о безопасной передаче файлов (Secure file transfer), реализуемой на базе протокола SFTP (SSH File Transfer Protocol) и уже устаревшего протокола SCP (Secure CoPy). Подключившись к серверу по SSH, с помощью специального клиента можно выполнять все основные операции с файлами: загружать их на сервер, переименовывать файлы и папки, изменять свойства файлов, а также создавать символические ссылки и ярлыки. Одним из

самых известных клиентов под винду является **WinSCP** (www.winscp.net). Помимо самых стандартных опций, тут есть и ряд бонусов. Для обновления некоторых сайтов я нередко использую функцию по синхронизации директорий, а автоматизировать часть рутинной работы на сервере, где у меня хранятся бэкапы, помогает возможность написания простых скриптов. Вдвойне приятно, что WinSCP интегрируется с Pageant и позволяет использовать уже сохраненные публичные ключи и сохраненные парольные фразы для подключения.

Впрочем, зачем вообще заморачиваться с запуском каких-то программ? Файлы с удаленного сервера можно примонтировать прямо в систему, и, все равно, операции с ними буду осуществляться через SSH. **ExpanDrive** (www.expandrive.com), которая раньше называлась SFtpDrive, позволяет прозрачно примонтировать новый логический диск и работать с ним, как если бы это была, к примеру, флешка. Я использую эту прогу в достаточно странном ключе, а именно — для доступа к файлам некоторых никсовых систем из-под винды :).

ТРИК 8: КЛИЕНТ С ДОСТУПОМ ЧЕРЕЗ ВЕБ

Ситуация: дома у тебя есть настроенный клиент, с параметрами сессий, ключами для доступа, сохраненными логинами и паролями. Приятно воспользоваться всеми этими благами удаленно. В этом плане интересной разработкой стал бесплатный Telnet/SSH клиент **Tera Term** (http://www.ayera.com/teraterm). Фишка в том, что тулза имеет встроенный веб-сервер, который включается через меню «Web — Accept HTTP Connections». После этого ты получаешь практически полноценный клиент через обычный браузер, просто набрав адрес машины и порт, на котором он принимает соединения. «А ведь наверняка же есть реализации SSH-клиента, полностью написанные для веб», — задумался я. В результате несложных поисков попалось сразу несколько реализаций, но самой качественной оказался **WebShell** (www-personal.umich.edu/~mressl/webshell). Единственная трабла — он написан на Python, а потому установить его на простой хостинг не получится. Но зато он полностью сделан на Ajax, а использовать его удобно не только с обычного компа, но и с телефона (кстати, ссылки на клиенты под различные мобильные платформы ты найдешь во врезке).

ТРИК 9: ПОДКЛЮЧЕНИЕ К RDP ЧЕРЕЗ SSH

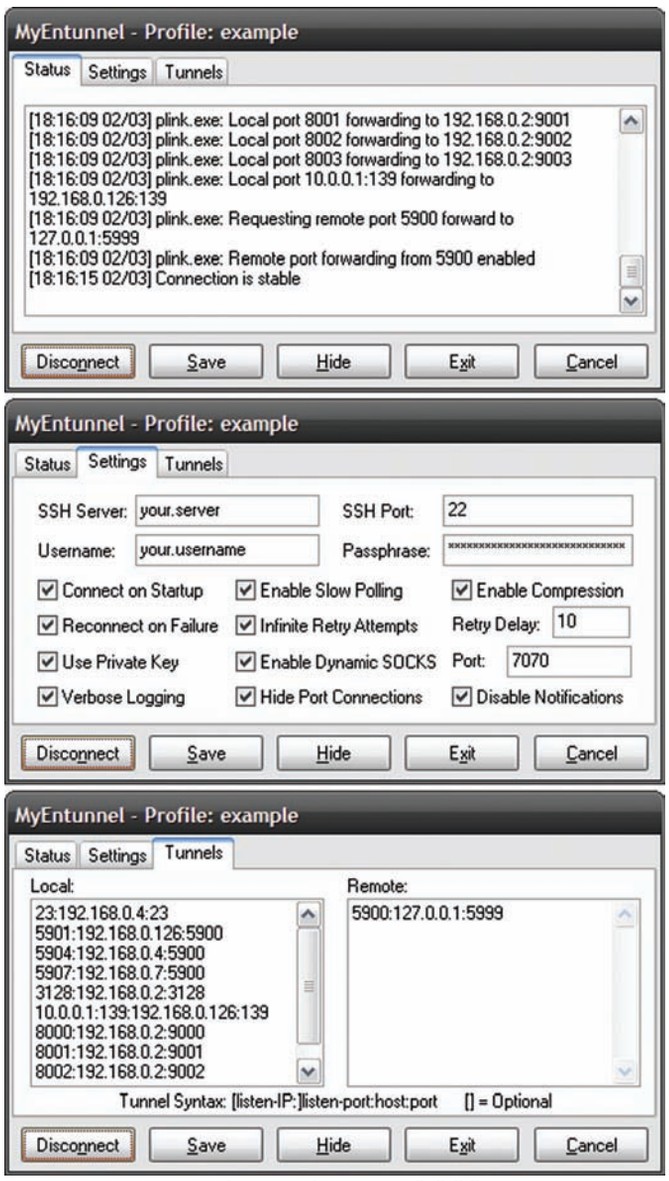
Поднять SSH-туннель и пустить через него VNC или RPD сессию проще простого. Однако для доступа к удаленным хостам по RPD-протоколу есть специальная утилита **WiSSH** (www.wissh.com). WiSSH позволяет осуществлять доступ через Gateway SSH-сервер к компьютерам со следующими системами: Windows 2000 Terminal Servers; Windows 2003 Terminal Servers; Windows NT Terminal Server Edition; Windows XP и Windows 2000/2003 с включенным Remote Desktop. Пользователи смогут работать на удаленной машине так же, как если бы находились непосредственно рядом с ней.

ТРИК 10: АВТОМАТИЗАЦИЯ

Об одном из видов автоматизации мы уже говорили. В случае использования PuTTY Connection Manager для любого соединения можно задать последовательность команд, которые будут выполняться после успешного входа на удаленной системе. Вот еще один способ упростить жизнь админу, в распоряжении которого имеются несколько серверов с одинаковой конфигурацией. С помощью утилиты **ClusterSSH** (clusterssh.sourceforge.net) можно администрировать сразу несколько удаленных хостов. Прога открывает несколько SSH-соединений с различным узлами, а также одну общую администраторскую консоль. Любая команда, набранная в этой консоли, реплицируется, т.е. передается по всем SSH-соединением. Это избавляет тебя от повторения монотонной работы. К сожалению, подобное решение есть только под никсы. ClusterSSH управляет несколькими окнами xterm через единый интерфейс, причем сам он частично написан Perl/TK.

ТРИК 11: ЗАЩИТА ОТ БРУТФОРСА

Авторизация при помощи логина и пароля считается самой небезопасной. В большинстве случаев рекомендуется вообще отключать ее



MYENTUNNEL ГАРАНТИРУЕТ, ЧТО С SSH-ТУННЕЛЕМ БУДЕТ ВСЕ В ПОРЯДКЕ

на сервере, а заодно деактивировать поддержку устаревшего протокола SSH1. Чтобы сделать это в OpenSSH — а он наиболее распространен — необходимо внести поправки в конфиг:

```
vi /etc/ssh/sshd_config
[...]
Protocol 2
PasswordAuthentication no
UsePAM no
[...]
```

Если отключать авторизацию не хочешь, то надо установить примитивную систему предотвращения вторжений. Например, **Sshguard** (sshguard.sourceforge.net). Простой демон проверяет записи в журналах (syslog, syslog-ng, metalog, multilog, raw) и способен вычислять подозрительную активность вроде попыток подбора паролей. Для блокировки таких IP-адресов используется локальный фильтр пакетов (pf, ipfw, netfilter/iptables или файл hosts.allow). Поддерживаются сервисы sshd, dovecot, proftpd, pure-ftpd, FreeBSD ftpd, UWimap (imap, pop). Аналогично работают **Fail2ban** (www.fail2ban.org) и **Sshdfilter** (http://www.csc.liv.ac.uk/~greg/sshdfilter).