Previous | Contents | Index | Next

Chapter 8: Using public keys for SSH authentication

8.1 Public key authentication - an introduction

Public key authentication is an alternative means of identifying yourself to a login server, instead of typing a password. It is more secure and more flexible, but more difficult to set up.

In conventional password authentication, you prove you are who you claim to be by proving that you know the correct password. The only way to prove you know the password is to tell the server what you think the password is. This means that if the server has been hacked, or *spoofed* (see section 2.2), an attacker can learn your password.

Public key authentication solves this problem. You generate a *key pair*, consisting of a public key (which everybody is allowed to know) and a private key (which you keep secret and do not give to anybody). The private key is able to generate *signatures*. A signature created using your private key cannot be forged by anybody who does not have that key; but anybody who has your public key can verify that a particular signature is genuine.

So you generate a key pair on your own computer, and you copy the public key to the server. Then, when the server asks you to prove who you are, PuTTY can generate a signature using your private key. The server can verify that signature (since it has your public key) and allow you to log in. Now if the server is hacked or spoofed, the attacker does not gain your private key or password; they only gain one signature. And signatures cannot be re-used, so they have gained nothing.

There is a problem with this: if your private key is stored unprotected on your own computer, then anybody who gains access to *that* will be able to generate signatures as if they were you. So they will be able to log in to your server under your account. For this reason, your private key is usually *encrypted* when it is stored on your local machine, using a passphrase of your choice. In order to generate a signature, PuTTY must decrypt the key, so you have to type your passphrase.

This can make public-key authentication less convenient than password authentication: every time you log in to the server, instead of typing a short password, you have to type a longer passphrase. One solution to this is to use an *authentication agent*, a separate program which holds decrypted private keys and generates signatures on

request. PuTTY's authentication agent is called Pageant. When you begin a Windows session, you start Pageant and load your private key into it (typing your passphrase once). For the rest of your session, you can start PuTTY any number of times and Pageant will automatically generate signatures without you having to do anything. When you close your Windows session, Pageant shuts down, without ever having stored your decrypted private key on disk. Many people feel this is a good compromise between security and convenience. See chapter 9 for further details.

There is more than one public-key algorithm available. The most common is RSA, but others exist, notably DSA (otherwise known as DSS), the USA's federal Digital Signature Standard. The key types supported by PuTTY are described in section 8.2.2.

8.2 Using PuTTYgen, the PuTTY key generator

PuTTYgen is a key generator. It generates pairs of public and private keys to be used with PuTTY, PSCP, and Plink, as well as the PuTTY authentication agent, Pageant (see chapter 9). PuTTYgen generates RSA and DSA keys.

When you run PuTTYgen you will see a window where you have two choices: ‘Generate’, to generate a new public/private key pair, or ‘Load’ to load in an existing private key.

8.2.1 Generating a new key

This is a general outline of the procedure for generating a new key pair. The following sections describe the process in more detail.

- First, you need to select which type of key you want to generate, and also select the strength of the key. This is described in more detail in section 8.2.2 and section 8.2.3.
- Then press the ‘Generate’ button, to actually generate the key. Section 8.2.4 describes this step.
- Once you have generated the key, select a comment field (section 8.2.6) and a passphrase (section 8.2.7).
- Now you're ready to save the private key to disk; press the ‘Save private key’ button. (See section 8.2.8).

Your key pair is now ready for use. You may also want to copy the public key to your server, either by copying it out of the ‘Public key for pasting into authorized_keys file’ box (see section 8.2.10), or by using the ‘Save public key’ button (section 8.2.9). However, you don't need to do this immediately; if you want, you can load the private key back into PuTTYgen later (see section 8.2.11) and the public key will be available for copying and pasting again.

Section 8.3 describes the typical process of configuring PuTTY to attempt public-key authentication, and configuring your SSH server to accept it.

8.2.2 Selecting the type of key

Before generating a key pair using PuTTYgen, you need to select which type of key you need. PuTTYgen currently supports three types of key:

- An RSA key for use with the SSH-1 protocol.
- An RSA key for use with the SSH-2 protocol.
- A DSA key for use with the SSH-2 protocol.

The SSH-1 protocol only supports RSA keys; if you will be connecting using the SSH-1 protocol, you must select the first key type or your key will be completely useless.

The SSH-2 protocol supports more than one key type. The two types supported by PuTTY are RSA and DSA.

The PuTTY developers *strongly* recommend you use RSA. DSA has an intrinsic weakness which makes it very easy to create a signature which contains enough information to give away the *private* key! This would allow an attacker to pretend to be you for any number of future sessions. PuTTY's implementation has taken very careful precautions to avoid this weakness, but we cannot be 100% certain we have managed it, and if you have the choice we strongly recommend using RSA keys instead.

If you really need to connect to an SSH server which only supports DSA, then you probably have no choice but to use DSA. If you do use DSA, we recommend you do not use the same key to authenticate with more than one server.

8.2.3 Selecting the size (strength) of the key

The 'Number of bits' input box allows you to choose the strength of the key PuTTYgen will generate.

Currently 1024 bits should be sufficient for most purposes.

Note that an RSA key is generated by finding two primes of half the length requested, and then multiplying them together. For example, if you ask PuTTYgen for a 1024-bit RSA key, it will create two 512-bit primes and multiply them. The result of this multiplication might be 1024 bits long, or it might be only 1023; so you may not get the exact length of key you asked for. This is perfectly normal, and you do not need to worry. The lengths should only ever differ by one, and there is no perceptible drop in security as a result.

DSA keys are not created by multiplying primes together, so they should always be exactly the length you asked for.

8.2.4 The 'Generate' button

Once you have chosen the type of key you want, and the strength of the key, press the 'Generate' button and PuTTYgen will begin the process of actually generating the key.

First, a progress bar will appear and PuTTYgen will ask you to move the mouse around to generate randomness. Wave the mouse in circles over the blank area in the PuTTYgen window, and the progress bar will gradually fill up as PuTTYgen collects enough randomness. You don't need to wave the mouse in particularly imaginative patterns (although it can't hurt); PuTTYgen will collect enough randomness just from the fine detail of *exactly* how far the mouse has moved each time Windows samples its position.

When the progress bar reaches the end, PuTTYgen will begin creating the key. The progress bar will reset to the start, and gradually move up again to track the progress of the key generation. It will not move evenly, and may occasionally slow down to a stop; this is unfortunately unavoidable, because key generation is a random process and it is impossible to reliably predict how long it will take.

When the key generation is complete, a new set of controls will appear in the window to indicate this.

8.2.5 The 'Key fingerprint' box

The 'Key fingerprint' box shows you a fingerprint value for the generated key. This is derived cryptographically from the *public* key value, so it doesn't need to be kept secret.

The fingerprint value is intended to be cryptographically secure, in the sense that it is computationally infeasible for someone to invent a second key with the same fingerprint, or to find a key with a particular fingerprint. So some utilities, such as the Pageant key list box (see section 9.2.1) and the Unix `ssh-add` utility, will list key fingerprints rather than the whole public key.

8.2.6 Setting a comment for your key

If you have more than one key and use them for different purposes, you don't need to memorise the key fingerprints in order to tell them apart. PuTTYgen allows you to enter a *comment* for your key, which will be displayed whenever PuTTY or Pageant asks you for the passphrase.

The default comment format, if you don't specify one, contains the key type and the date of generation, such as `rsa-key-20011212`. Another commonly used approach is to use your name and the name of the computer the key will be used on, such as `simon@simons-pc`.

To alter the key comment, just type your comment text into the 'Key comment' box before saving the private key. If you want to change the comment later, you can load the private key back into PuTTYgen, change the comment, and save it again.

8.2.7 Setting a passphrase for your key

The 'Key passphrase' and 'Confirm passphrase' boxes allow you to choose a passphrase for your key. The passphrase will be used to encrypt the key on disk, so you will not be able to use the key without first entering the passphrase.

When you save the key, PuTTYgen will check that the 'Key passphrase' and 'Confirm passphrase' boxes both contain exactly the same passphrase, and will refuse to save the key otherwise.

If you leave the passphrase fields blank, the key will be saved unencrypted. You should *not* do this without good reason; if you do, your private key file on disk will be all an attacker needs to gain access to any machine configured to accept that key. If you want to be able to passwordless loginlog in without having to type a passphrase every time, you should consider using Pageant (chapter 9) so that your decrypted key is only held in memory rather than on disk.

Under special circumstances you may genuinely *need* to use a key with no passphrase; for example, if you need to run an automated batch script that needs to make an SSH connection, you can't be there to type the passphrase. In this case we recommend you generate a special key for each specific batch script (or whatever) that needs one, and on the server side you should arrange that each key is *restricted* so that it can only be used for that specific purpose. The documentation for your SSH server should explain how to do this (it will probably vary between servers).

Choosing a good passphrase is difficult. Just as you shouldn't use a dictionary word as a password because it's easy for an attacker to run through a whole dictionary, you should not use a song lyric, quotation or other well-known sentence as a passphrase. DiceWare (`www.diceware.com`) recommends using at least five words each generated randomly by rolling five dice, which gives over 2^64 possible passphrases and is probably not a bad scheme. If you want your passphrase to make grammatical sense, this cuts down the possibilities a lot and you should use a longer one as a result.

Do not forget your passphrase. There is no way to recover it.

8.2.8 Saving your private key to a disk file

Once you have generated a key, set a comment field and set a passphrase, you are ready to save your private key to disk.

Press the 'Save private key' button. PuTTYgen will put up a dialog box asking you where to save the file. Select a directory, type in a file name, and press 'Save'.

This file is in PuTTY's native format (`*.PPK`); it is the one you will need to tell PuTTY to use for authentication (see section 4.20.7) or tell Pageant to load (see section 9.2.2).

8.2.9 Saving your public key to a disk file

RFC 4716 specifies a standard format for storing SSH-2 public keys on disk. Some SSH servers (such as `ssh.com`'s) require a public key in this format in order to accept authentication with the corresponding private key. (Others, such as OpenSSH, use a different format; see section 8.2.10.)

To save your public key in the SSH-2 standard format, press the 'Save public key' button in PuTTYgen. PuTTYgen will put up a dialog box asking you where to save the file. Select a directory, type in a file name, and press 'Save'.

You will then probably want to copy the public key file to your SSH server machine. See section 8.3 for general instructions on configuring public-key authentication once you have generated a key.

If you use this option with an SSH-1 key, the file PuTTYgen saves will contain exactly the same text that appears in the 'Public key for pasting' box. This is the only existing standard for SSH-1 public keys.

8.2.10 'Public key for pasting into authorized_keys file'

All SSH-1 servers require your public key to be given to it in a one-line format before it will accept authentication with your private key. The OpenSSH server also requires this for SSH-2.

The 'Public key for pasting into authorized_keys file' gives the public-key data in the correct one-line format. Typically you will want to select the entire contents of the box using the mouse, press Ctrl+C to copy it to the clipboard, and then paste the data into a PuTTY session which is already connected to the server.

See section 8.3 for general instructions on configuring public-key authentication once you have generated a key.

8.2.11 Reloading a private key

PuTTYgen allows you to load an existing private key file into memory. If you do this, you can then change the passphrase and comment before saving it again; you can also make extra copies of the public key.

To load an existing key, press the 'Load' button. PuTTYgen will put up a dialog box where you can browse around the file system and find your key file. Once you select the file, PuTTYgen will ask you for a passphrase (if necessary) and will then display the key details in the same way as if it had just generated the key.

If you use the Load command to load a foreign key format, it will work, but you will see a message box warning you that the key you have loaded is not a PuTTY native key. See section 8.2.12 for information about importing foreign key formats.

8.2.12 Dealing with private keys in other formats

Most SSH-1 clients use a standard format for storing private keys on disk. PuTTY uses this format as well; so if you have generated an SSH-1 private key using OpenSSH or `ssh.com`'s client, you can use it with PuTTY, and vice versa.

However, SSH-2 private keys have no standard format. OpenSSH and `ssh.com` have different formats, and PuTTY's is different again. So a key generated with one client cannot immediately be used with another.

Using the 'Import' command from the 'Conversions' menu, PuTTYgen can load SSH-2 private keys in OpenSSH's format and `ssh.com`'s format. Once you have loaded one of these key types, you can then save it back out as a PuTTY-format key (`*.PPK`) so that you can use it with the PuTTY suite. The passphrase will be unchanged by this process (unless you deliberately change it). You may want to change the key comment before

you save the key, since OpenSSH's SSH-2 key format contains no space for a comment and `ssh.com`'s default comment format is long and verbose.

PuTTYgen can also export private keys in OpenSSH format and in `ssh.com` format. To do so, select one of the 'Export' options from the 'Conversions' menu. Exporting a key works exactly like saving it (see section 8.2.8) - you need to have typed your passphrase in beforehand, and you will be warned if you are about to save a key without a passphrase.

Note that since only SSH-2 keys come in different formats, the export options are not available if you have generated an SSH-1 key.

8.3 Getting ready for public key authentication

Connect to your SSH server using PuTTY with the SSH protocol. When the connection succeeds you will be prompted for your user name and password to login. Once logged in, you must configure the server to accept your public key for authentication:

- If your server is using the SSH-1 protocol, you should change into the `.ssh` directory and open the file `authorized_keys` with your favourite editor. (You may have to create this file if this is the first key you have put in it). Then switch to the PuTTYgen window, select all of the text in the 'Public key for pasting into authorized_keys file' box (see section 8.2.10), and copy it to the clipboard (`Ctrl+C`). Then, switch back to the PuTTY window and insert the data into the open file, making sure it ends up all on one line. Save the file.
- If your server is OpenSSH and is using the SSH-2 protocol, you should follow the same instructions, except that in earlier versions of OpenSSH 2 the file might be called `authorized_keys2`. (In modern versions the same `authorized_keys` file is used for both SSH-1 and SSH-2 keys.)
- If your server is `ssh.com`'s product and is using SSH-2, you need to save a *public* key file from PuTTYgen (see section 8.2.9), and copy that into the `.ssh2` directory on the server. Then you should go into that `.ssh2` directory, and edit (or create) a file called `authorization`. In this file you should put a line like `Key mykey.pub`, with `mykey.pub` replaced by the name of your key file.
- For other SSH server software, you should refer to the manual for that server.

You may also need to ensure that your home directory, your `.ssh` directory, and any other files involved (such as `authorized_keys`, `authorized_keys2` or `authorization`) are not group-writable or world-writable. You can typically do this by using a command such as

```
chmod go-w $HOME $HOME/.ssh $HOME/.ssh/authorized_keys
```

Your server should now be configured to accept authentication using your private key. Now you need to configure PuTTY to *attempt* authentication using your private key. You can do this in any of three ways:

- Select the private key in PuTTY's configuration. See section 4.20.7 for details.
- Specify the key file on the command line with the `-i` option. See section 3.8.3.18 for details.
- Load the private key into Pageant (see chapter 9). In this case PuTTY will automatically try to use it for authentication if it can.

If you want to provide feedback on this manual or on the PuTTY tools themselves, see the Feedback page.

[PuTTY release 0.60]