

24 October 2006

By: Mihai Marinof, Linux Editor



[How to Use RSA Key for SSH Authentication](#)

Automatically log-in to Linux systems even without a password.

If your daily activity requires logging in a lot of Linux systems through SSH, you will be happy to know (if you don't already) that there's a way to allow secure, authenticated remote access, file transfer, and command execution without having to remember passwords for each individual host you connect. The `$HOME/.ssh/authorized_keys` file contains the RSA keys allowed for RSA authentication. Each line contains one key, which consists of the following fields: *options*, *bits*, *exponent*, *modulus* and *comment*. The first field is optional, bits, exponent and modulus fields give the RSA key and the last field isn't used at all in the authentication process, but it will be somewhat convenient to the user, for instance to know which key is for which machine. Before we start, make sure your computer has a ssh client installed and the remote Linux system has ssh installed and sshd running, with RSA authentication enabled (`RSAAuthentication yes` in `/etc/ssh/sshd_config`). First, you will need to generate the local RSA key: **# ssh-keygen -t rsa** *Generating public/private rsa key pair. Enter file in which to save the key (/root/.ssh/id_rsa):* (It's safe to press enter here, as the `/root/.ssh` is the default and recommended directory to hold the RSA file.) *Enter passphrase (empty for no passphrase):* *Enter same passphrase again:* (The password you enter here will need to be entered every time you use the RSA key but fortunately, you can set NO passphrase by pressing Enter. However, the upside is that you only have to remember this one passphrase for all the systems you access via RSA authentication and you can change the passphrase later with "ssh-keygen -p".) *Your identification has been saved in /root/.ssh/id_rsa. Your public key has been saved in /root/.ssh/id_rsa.pub.* Once the public key has been generated, it's time to upload it on any Linux systems you usually log into. It's recommended you use scp as the file transfer utility: **# scp .ssh/id_rsa.pub username@hostname.com:~** This command will copy the `id_rsa.pub` file in the `$HOME` directory. For instance, if you used root as the username, the file will be found in the `/root` directory and if you used a normal user, the file will be in the `/home/that.user/` directory. Next, connect to the remote host through SSH, with the username you used in the step above. RSA authentication won't be available just yet, so you'll have to use the old method to login. Once you are connected, add the new hostkey to the file `/root/.ssh/authorized_keys` or `/home/user/.ssh/authorized_keys`. If the `.ssh` directory doesn't exist, create it. **# cd \$HOME# cat id_rsa.pub >> .ssh/authorized_keys** The two right-angles will *add* the contents of `id_rsa.pub` file to the `authorized_keys` file, so in case the file already exists, you won't have to worry about the existing content being modified. You are all set. To test the RSA authentication, initiate a ssh connection from your PC to one of the Linux systems: **# ssh username@remote.hostname.com** If everything worked out well, you should be either asked for the passphrase (if you entered one), or get directly logged in. If you are prompted for the ssh password or get an error message, retry the above command using **-v** in order to turn verbose mode on and to be able to track down and correct the problem.