Updating a Host Key in SSH

SSH and Host Keys

When you connect to a computer using SSH, the intention is for a secure connection, where your data cannot be seen by others. If you try to connect to computer A, but instead connect to computer B (through accident or someone's malicious intent), then entering a password or other information *intended for computer A* could give whoever controls computer B information they should not have.

A **host key** is a specific number stored on a computer. No two computers have the same host key, so the SSH program uses it to identify the computer you are connecting to, to ensure it is not an imposter.

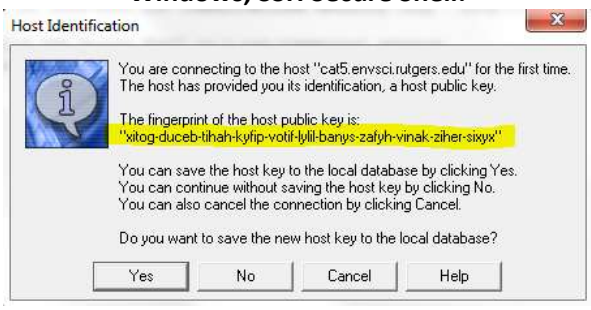
A host key can be over 600 digits long, so for readability it is represented by a shorter **fingerprint** of letters and numbers. Different SSH programs might show the fingerprint differently (see examples below). Two computers *could* possibly have the same fingerprint, but it is very unlikely.

When asked by your SSH program, ***always verify*** the host key fingerprint of a computer to ensure the connection is valid and secure!

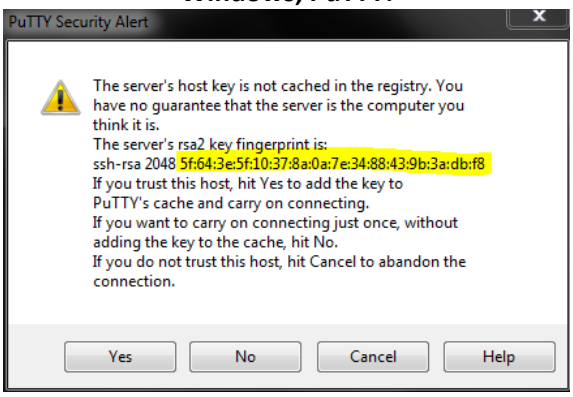
First Connection

The first time you connect to a certain computer, the SSH program doesn't know the correct host key, so it asks for confirmation. Here are examples of such prompts from common SSH programs. They are asking you to **verify the host key fingerprint** (highlighted in yellow). See the next page for how to verify that.

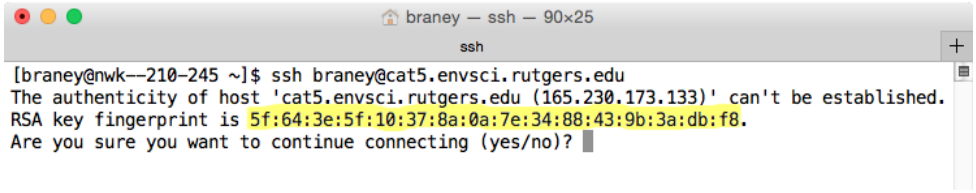
Windows, SSH Secure Shell:



Windows, PuTTY:



Mac OS X or Linux, `ssh` command run from the Terminal:



(Continued on next page)

Verifying a Host Key Fingerprint

To verify a host key fingerprint, you must compare the one presented by the SSH program to one that is provided by the administrator of the computer (i.e. Computing Services). For the **Cat5** computer used in some classes, you can find the valid fingerprints in the table below. The fingerprint to compare with depends on the one selected by your SSH program.

Valid Cat5 fingerprints (as of September 1, 2016)

Computer	Fingerprint	FP Type
Cat5	ee:05:33:ed:1e:5c:31:84:86:b9:8c:11:15:1a:9d:6f	RSA/RSA2 (hex)
Cat5	xodop-gybuk-bytas-kufut-laven-nuvyl-vegiz-zukyg-rycun-cubib-suxex	RSA/RSA2 (babble)
Cat5	25:ea:b6:1d:75:17:9e:ea:48:61:50:09:4d:80:57:17	ECDSA (hex)
Cat5	xitim-fycyf-gucuz-pusev-riguv-nosuf-putic-lobyl-botos-rebaz-cexyx	ECDSA (babble)
Cat5	55:87:cd:ef:80:dc:9d:e8:1d:14:87:27:40:00:01:4a	Ed25519 (hex)
Cat5	xilis-cukal-pazob-losak-gygaz-sizab-pufus-hufas-mugun-dipub-paxax	Ed25519 (babble)

NOTE: The fingerprints shown in the screenshots in this document are for the *old* Cat5, which was replaced at the beginning of the Fall 2016 semester. The table shows the keys for the *new* Cat5.

✶When connecting to cat5.envsci.rutgers.edu, if your SSH program asks for verification, compare the displayed fingerprint to those in the list above:

- If it matches one **exactly**, answer "yes" at the verification prompt. In that case, the SSH program will remember the host key so the computer can be recognized next time.
- If the fingerprint **does not** match any of those in the list, **do not accept it**. Instead, please contact Computing Services.

For other computers, please ask Computing Services for the valid fingerprints. The procedure is otherwise identical.

Host Key Changes: Beware!

Under normal operation, a computer's host key stays constant. However, it is possible for it to change if the computer undergoes a major system upgrade (such as the recent one on Cat5). The host key can also *appear* to change if someone attempts to redirect your connection to an imposter computer. In that case, the *real* host key did not change, but the one presented to you is **incorrect**.

To avoid malicious activity, your SSH program verifies the host key every time you connect to a computer. If the host key has changed, the SSH program will alert you and request that you verify the altered key. The next page shows examples of the ways SSH programs alert you to this fact. Note that they are very suspicious of the host key change. You should be suspicious, too, and **verify the key change before proceeding! If in doubt, choose "no"!**

To proceed, you must verify the new host key fingerprint (see previous section).

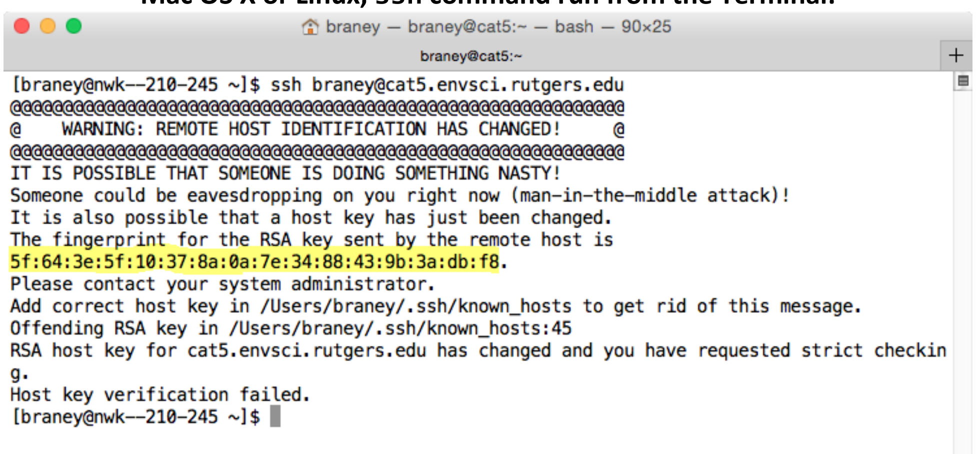
✶Since cat5 was recently updated, its host key has actually changed compared to last semester. Be sure to verify against the table above.

(Continued on next page)

Windows, SSH Secure Shell:

Windows, PuTTY:

Mac OS X or Linux, `ssh` command run from the Terminal:



If the presented fingerprint is correct, then proceed as follows:

In Windows: Click the "Yes" button, to the alert dialog and the program will remember the new host key as the correct one (forgetting the old one).

In Mac or Linux: You must manually remove the old host key from the program's memorized list (see instructions below). Then connect again, and it will be like connecting for the first time and you can have `ssh` remember the new key.

(Continued on next page)

Removing an Old Host Key (Mac/Linux only)

If you know what you're doing, you can edit the ~/.ssh/known_hosts file with a text editor and remove the line for the server in question. If you're not sure, follow the steps below.

Open up the Terminal (i.e. command prompt) and type the commands shown in `console` font.

1. Change to the directory containing the SSH configuration.

   ```
   cd ~/.ssh
   ```

   The file "known_hosts" in this directory contains the list of host keys you've accepted.

2. Store the name of the computer you're trying to update the host key for. For example, cat5:

   ```
   server="cat5.envsci.rutgers.edu"
   ```

3. Verify that you have the correct host name:

   ```
   grep -F $server known_hosts
   ```

   The above command should output one (long) line beginning with the server name, such as:

   ```
   cat5.envsci.rutgers.edu,165.230.173.133 ssh-rsa AAAAB3NzaC1yc2EA\
   AAABIwAAAQEA7wynAU/u5+BYMw2UNQEJRYG94DfgIh6CRtyg15STSMZIV2p0WxaO\
   KXD4Z1h+uVy+tG+MZgkC0b2J7hnh0JEOE2nh57qySVTCmGKIE+2lWL6iY/X9Rt9y\
   sBT9H9FXkdceuM0IqF1S66HTYwKZzbGQAFSJmE7SpUdNyeotUwXkdeF8RDjm4iLO\
   +dJ88y3HMc0jgbWQqKHGqWOVQpB5OMrorcKW5Xt71fhaUzpvLSGs5LGEawvhJrql\
   orjdS4dBFQ3cM0upaZEtBwljHQV5MtWiZzYwYLjwOtHAymdBxXHAK7yMdP4Q0uzH\
   ZrHHOPc17x2Lx9Hb/zXE6eB/yUlL8a8EfQ==
   ```

   If the above command spits out more than one line (i.e. more than one server name), then double-check your server setting in step 2. You don't want to delete lines for the wrong server!

4. Backup the known_hosts file and remove the old host key:

   ```
   mv known_hosts known_hosts.bak
   grep -v -F $server known_hosts.bak > known_hosts
   ```

5. That's it!