How to Symlink a Directory in Linux

Symlink, also known as a symbolic link in Linux, creates a link to a file or a directory for easier access. To put it in another way, symlinks are links that points to another file or folder in your system, quite similar to the shortcuts in Windows. Some users refer to symlinks as soft-links. Before moving forward, let’s elaborate soft-links and hard-links.

Hard-links: Hard-links are the links that mirror or copy the original file. Hard-links have the same inode numbers.

Soft-links: Soft-links are simple links that points to the original file. You can access the original file through soft links. Soft-links can point to a file or folder in any partition and have different inode numbers.

Learning about creating symlink in Linux is a great way to improve your grip on the Linux terminal. So, let’s learn the steps involved in making the soft-links in Linux.

How to Create Symlink (soft-link) in Linux

To make symlink or soft link, we use the “**ln**” command. The syntax to follow to create symlink is mentioned below:

$ ln -s [path of the target file/directory] [symbolic name]

In the first argument after the “-s” option, you will be giving the path of the file of a folder you want to create the symlink of. While in the second argument, pass the name you want to give that symlink. To check the created links, use the following command:

$ ls -l

To check inode numbers, use the command mentioned below:

$ ls -i

How to Create a Symlink (soft link) to a File

Creating a soft link to a file is simple; use the syntax mentioned below:

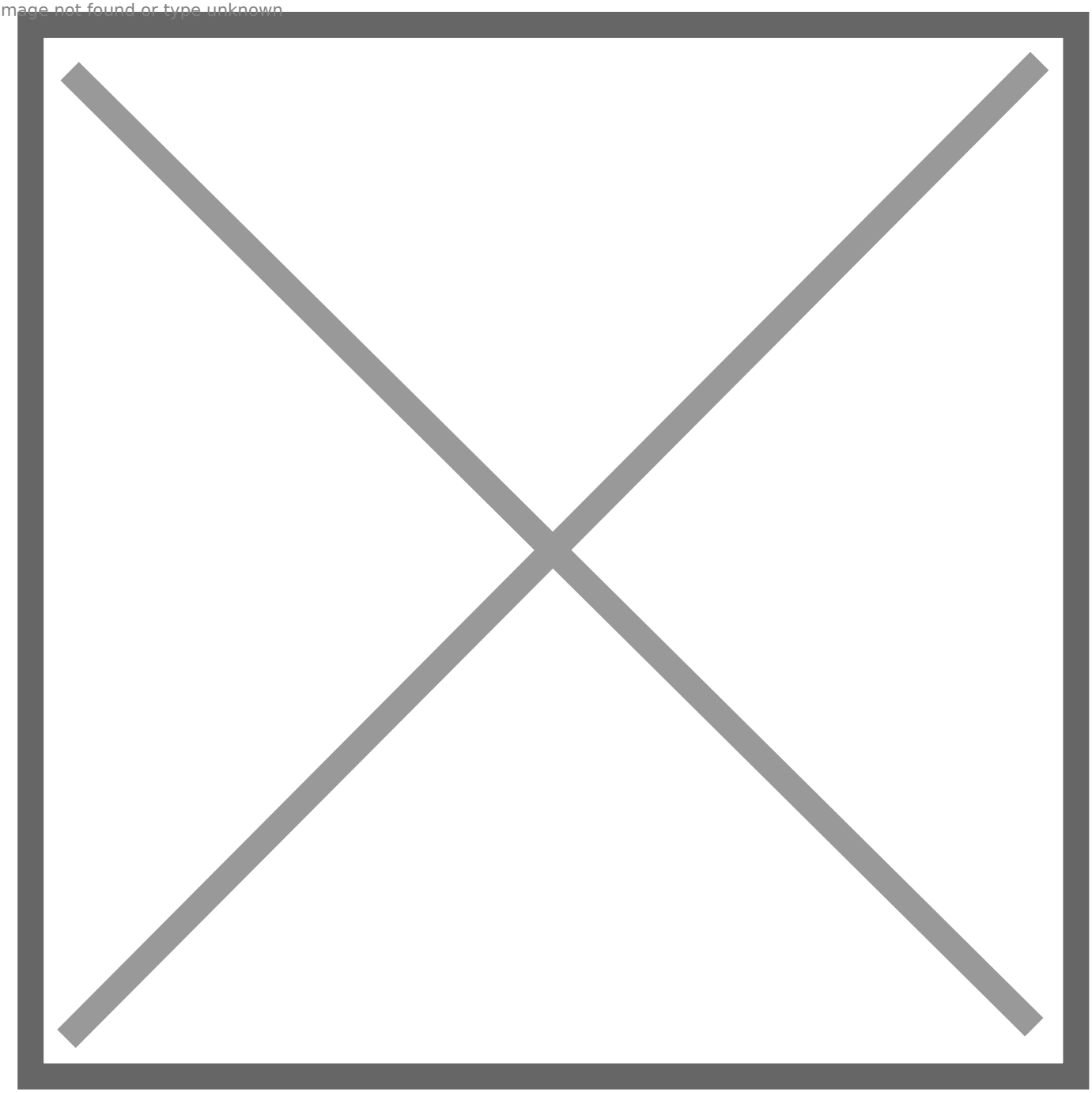
$ ln -s [path of the target file] [symbolic name]

Important to note that if you do not specify the “[symbolic name]”, then the command will create a symlink by the original file’s name. Let’s understand it through an example.

I have created a directory “my_folder” that contains a text file “my_doc.txt”. Now, to create symlink to “my_doc.txt” file, I will use:

$ ln -s my_folder/my_doc.txt my_document

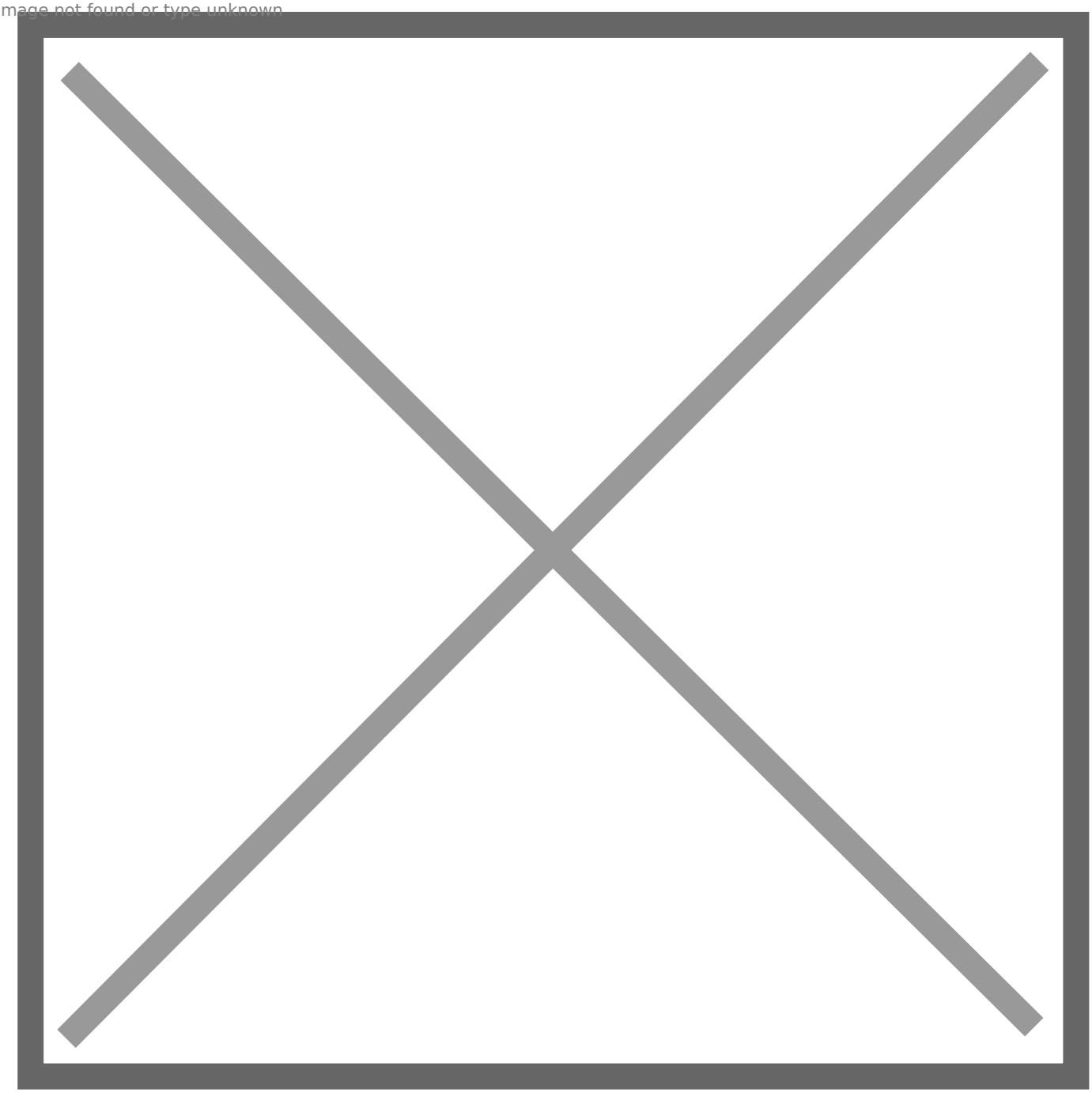
Image not found or type unknown



To verify it, use:

$ ls -l

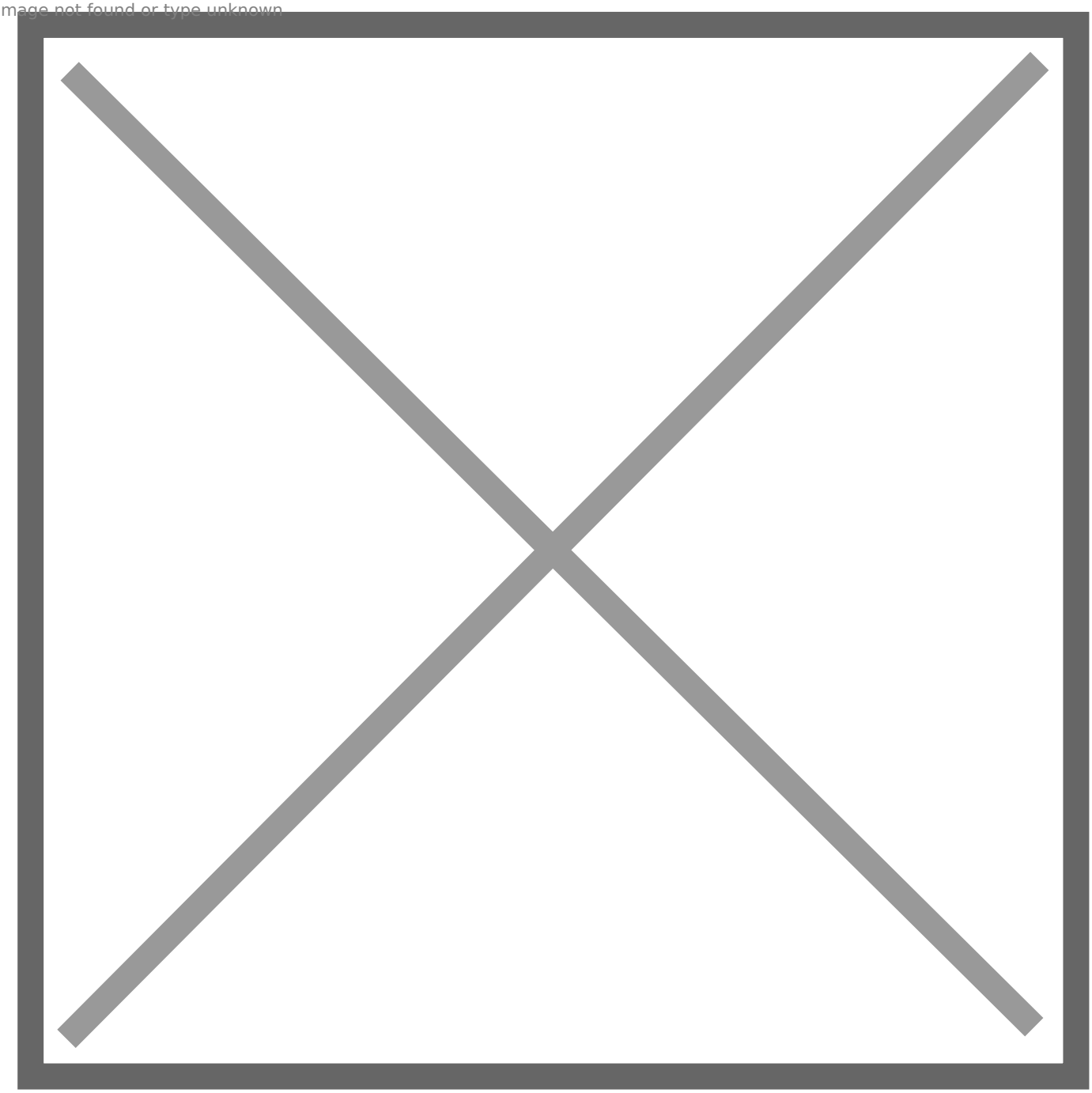
Image not found or type unknown



As it can be seen in the above output, “**my_document**” is pointing to “**my_folder/my_doc.txt**” file. Both the symlink and the original file would have different inode number. To check inode numbers used:

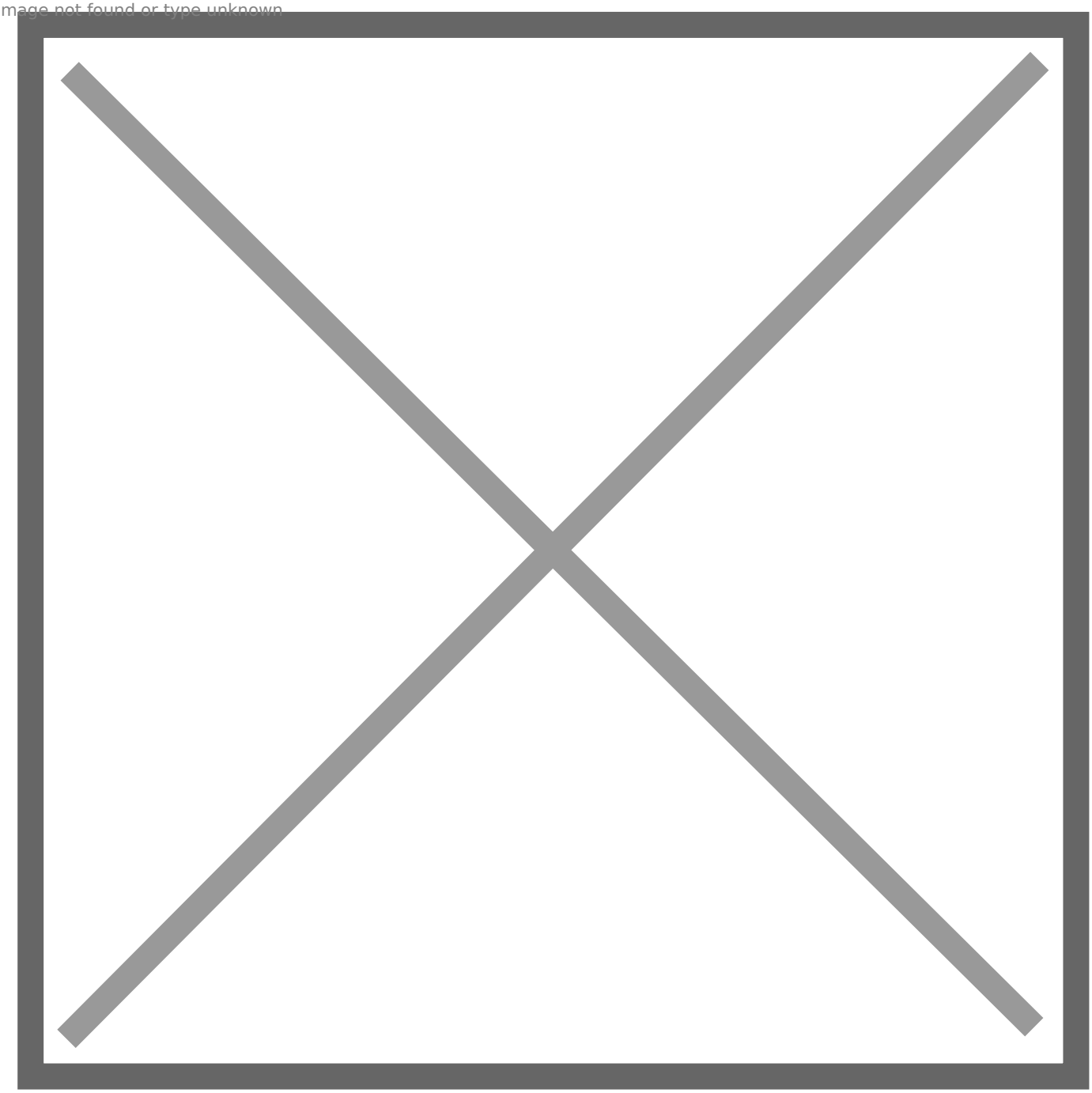
$ ls -i

Image not found or type unknown



Hard links will always have same inode numbers. To verify, I created a hard link of “**my_doc.txt**” file and name it “**my_document_2**”:

Image not found or type unknown



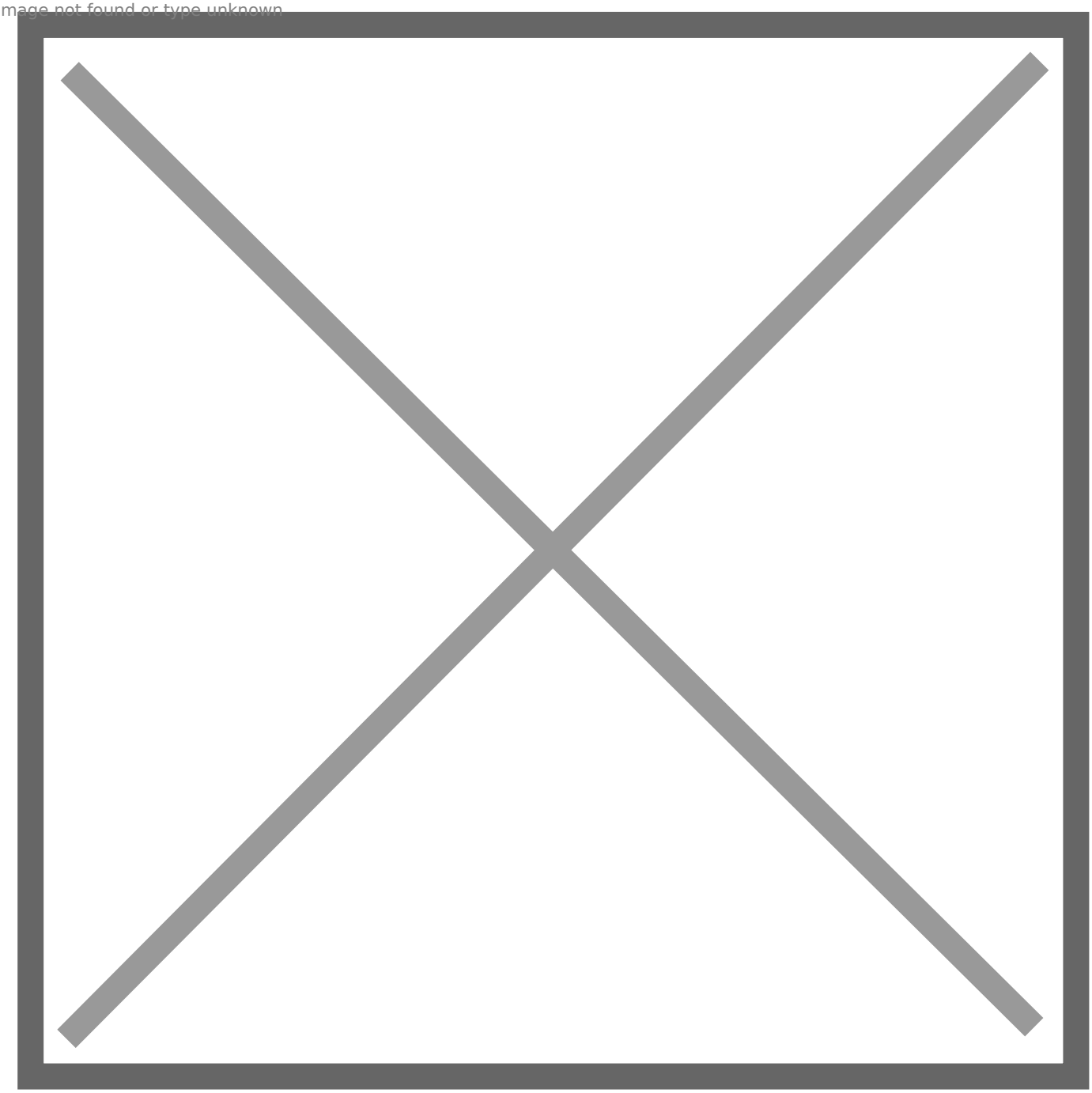
It can be seen in the output that the original file and the hard link have same inode numbers.

How to Create a Symlink (Soft Link) of the Folder/Directory

To create a soft-link or symlink to a directory is quite similar to creating a symlink to a file. For instance, I am creating the symlink of the "**my_folder**" directory using:

$ ln -s my_folder my_doc_folder

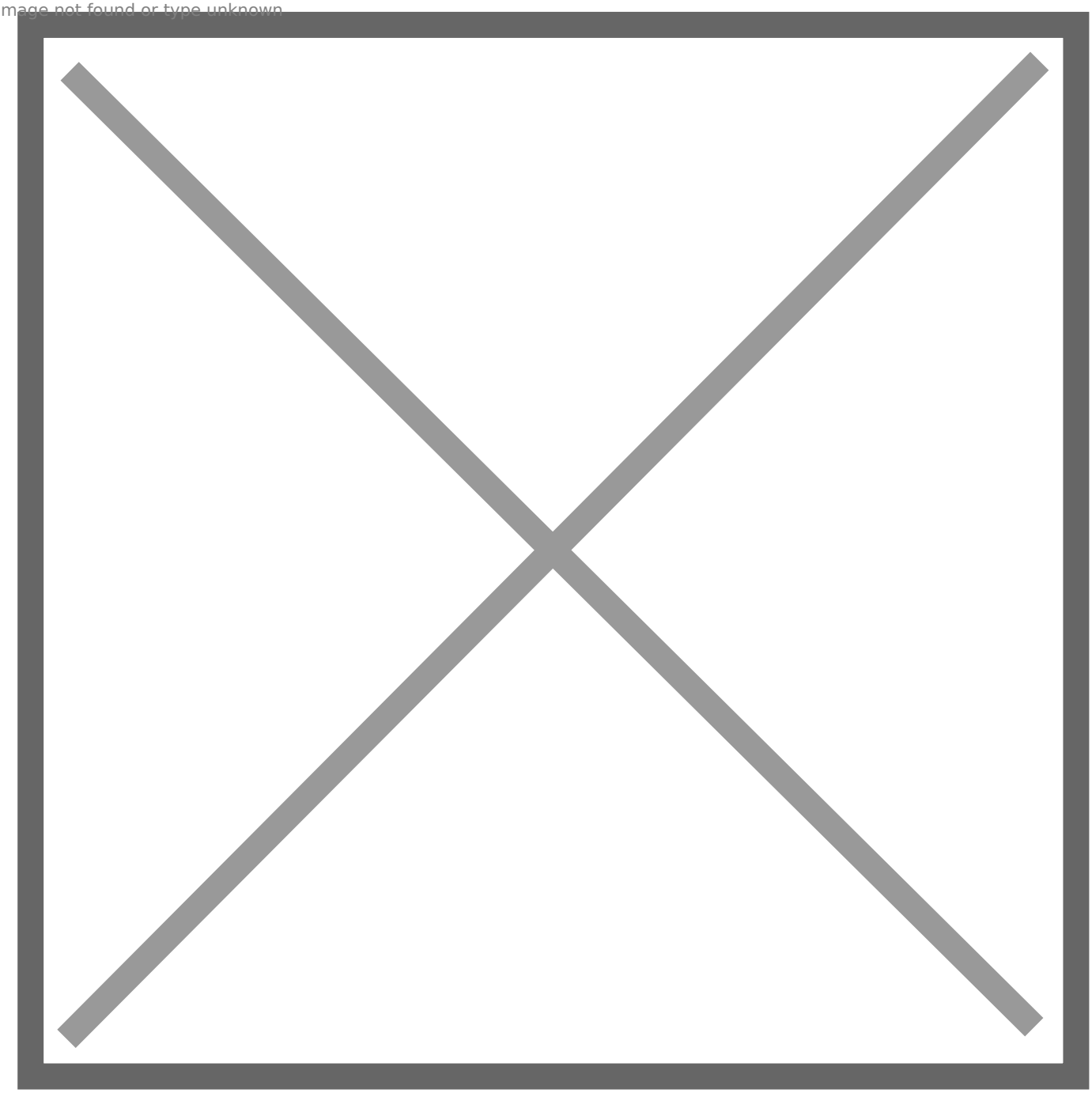
Image not found or type unknown



The above command will create a symlinked folder in the current directory. To verify it, use:

$ ls -l

Image not found or type unknown



Now, check inode numbers:

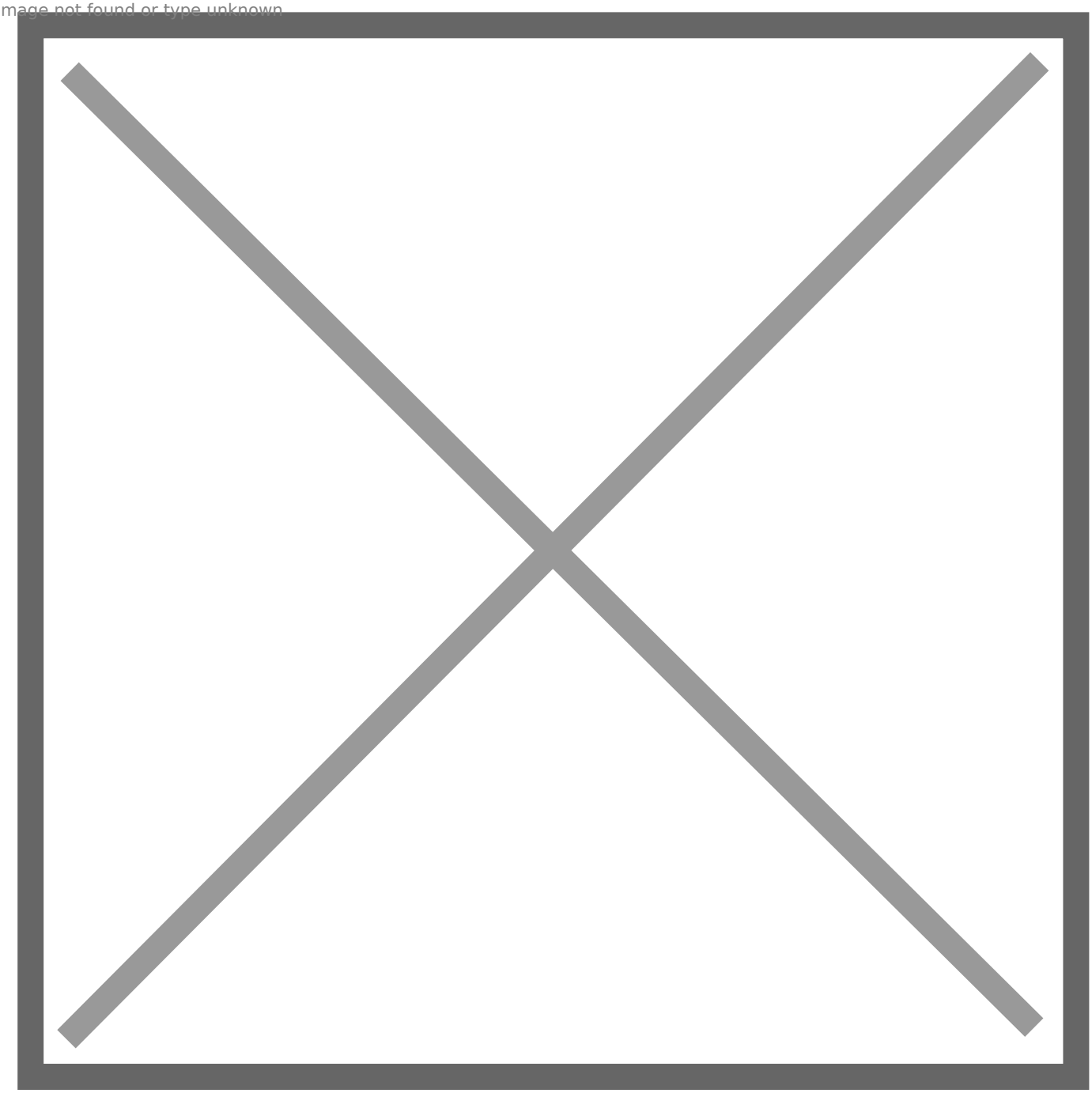
$ ls -i

How to Overwrite the Symlink (Soft Link) in Linux:

If you try to update a symlink with the same name that already exist, then you will get an error:

$ ln -s my_folder_2/my_doc_2.txt my_document

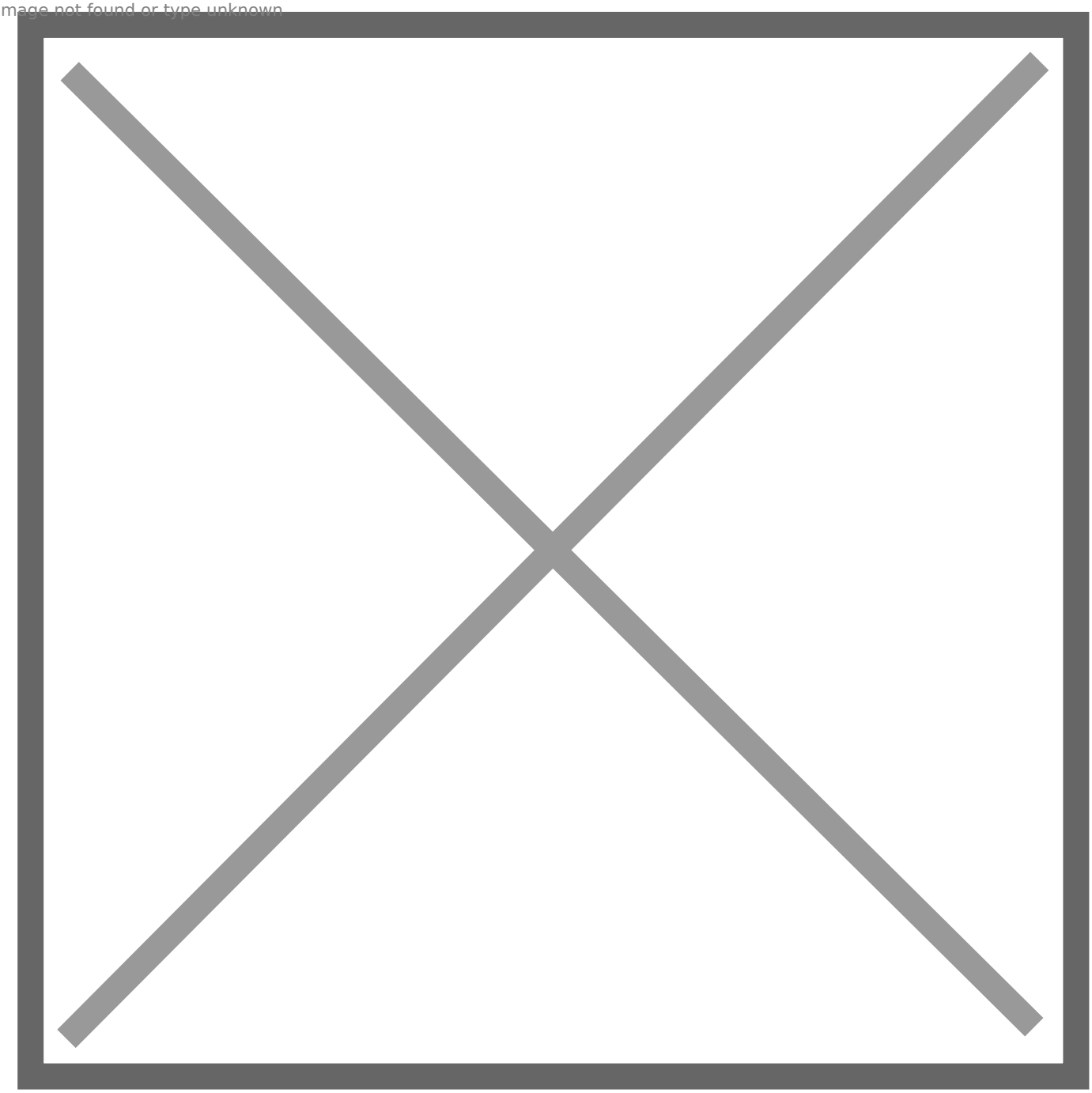
Image not found or type unknown



We will have to use the force flag “**-f**” to overwrite the new path to the existing symlink.

$ ln -sf my_folder_2/my_doc_2.txt my_document

Image not found or type unknown



How to Remove Symlink (Soft Link) in Linux:

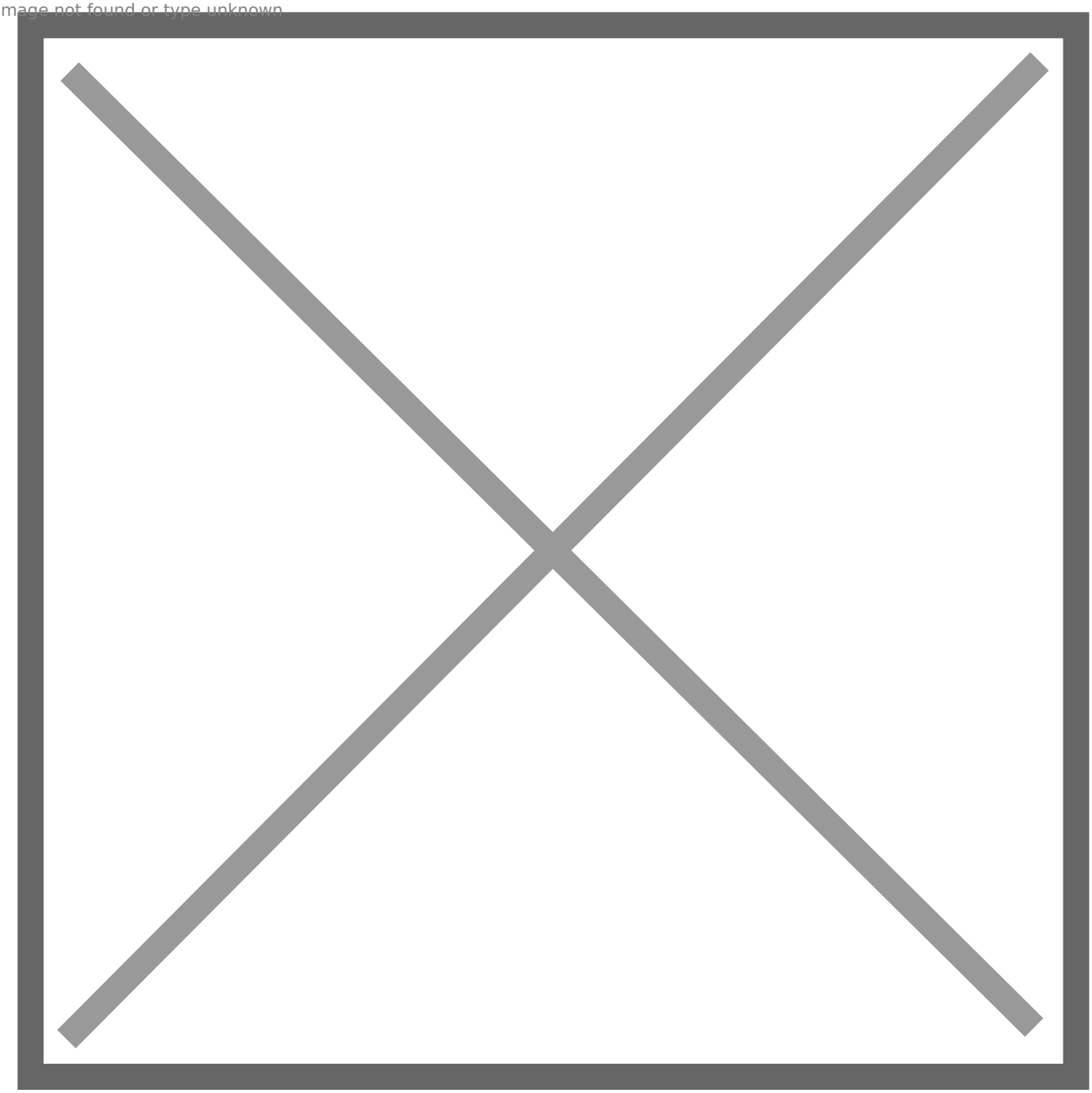
In many situation, you need to remove the unnecessary symlinks from your system. To delete symlink, we use the “**unlink**” command, and the syntax is given below:

$ unlink [symlink name]

Let’s remove the symlinks we created in the above examples. To unlink a symlink of a file, use:

$ unlink my_document

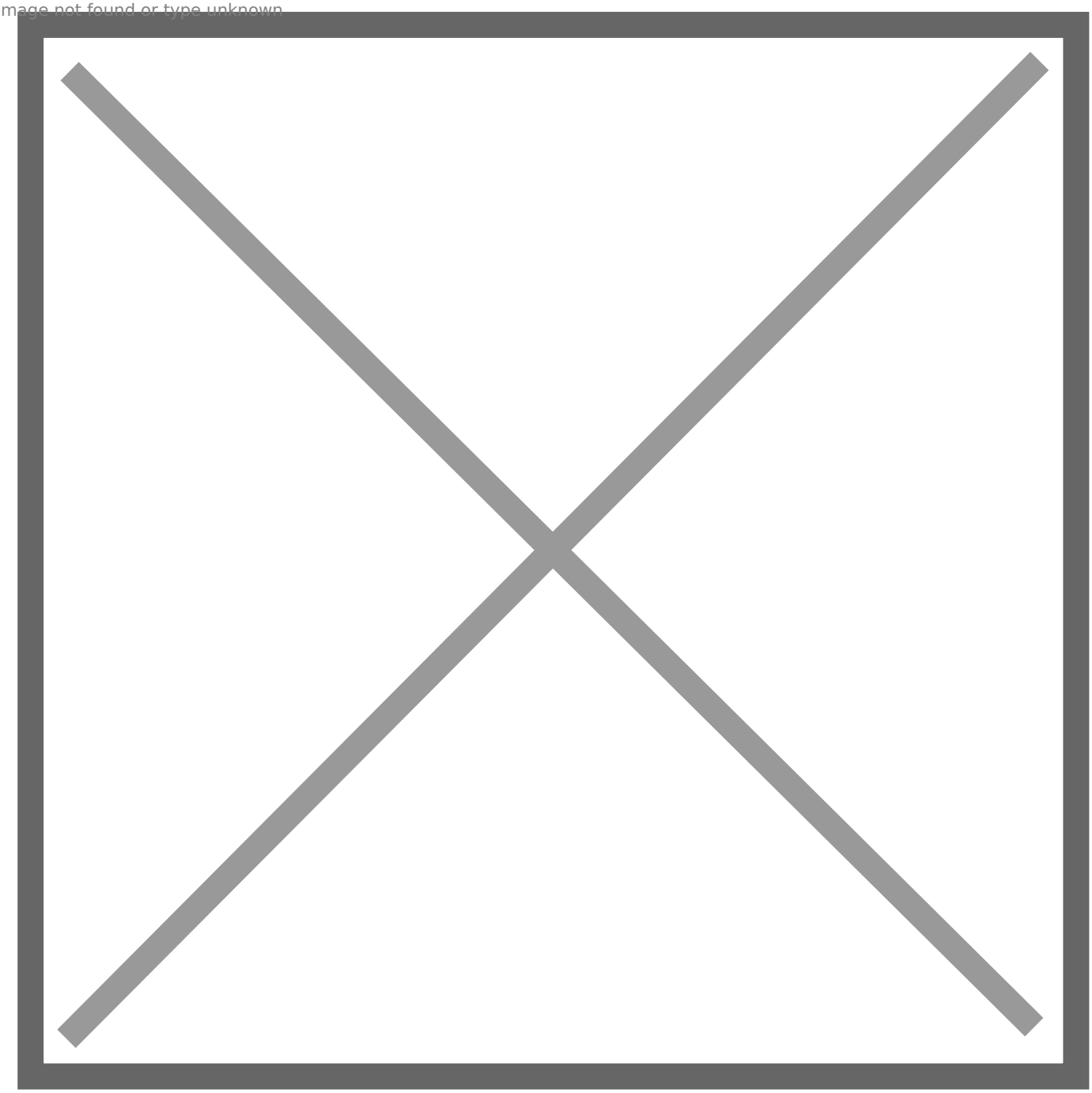
Image not found or type unknown



And to unlink the symlink of a directory:

$ unlink my_doc_folder

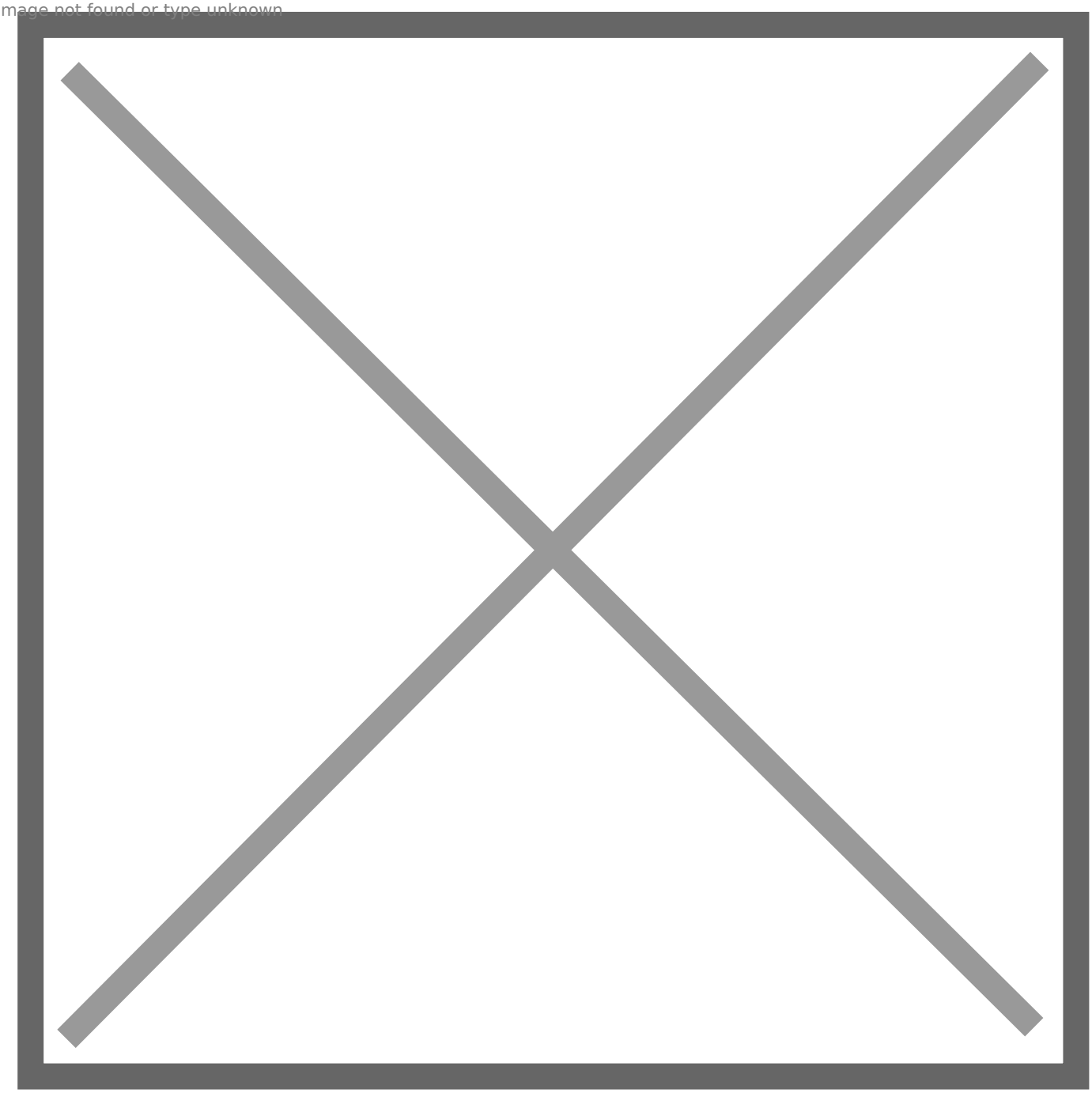
Image not found or type unknown



We can also use the “**rm**” command to remove symlinks.

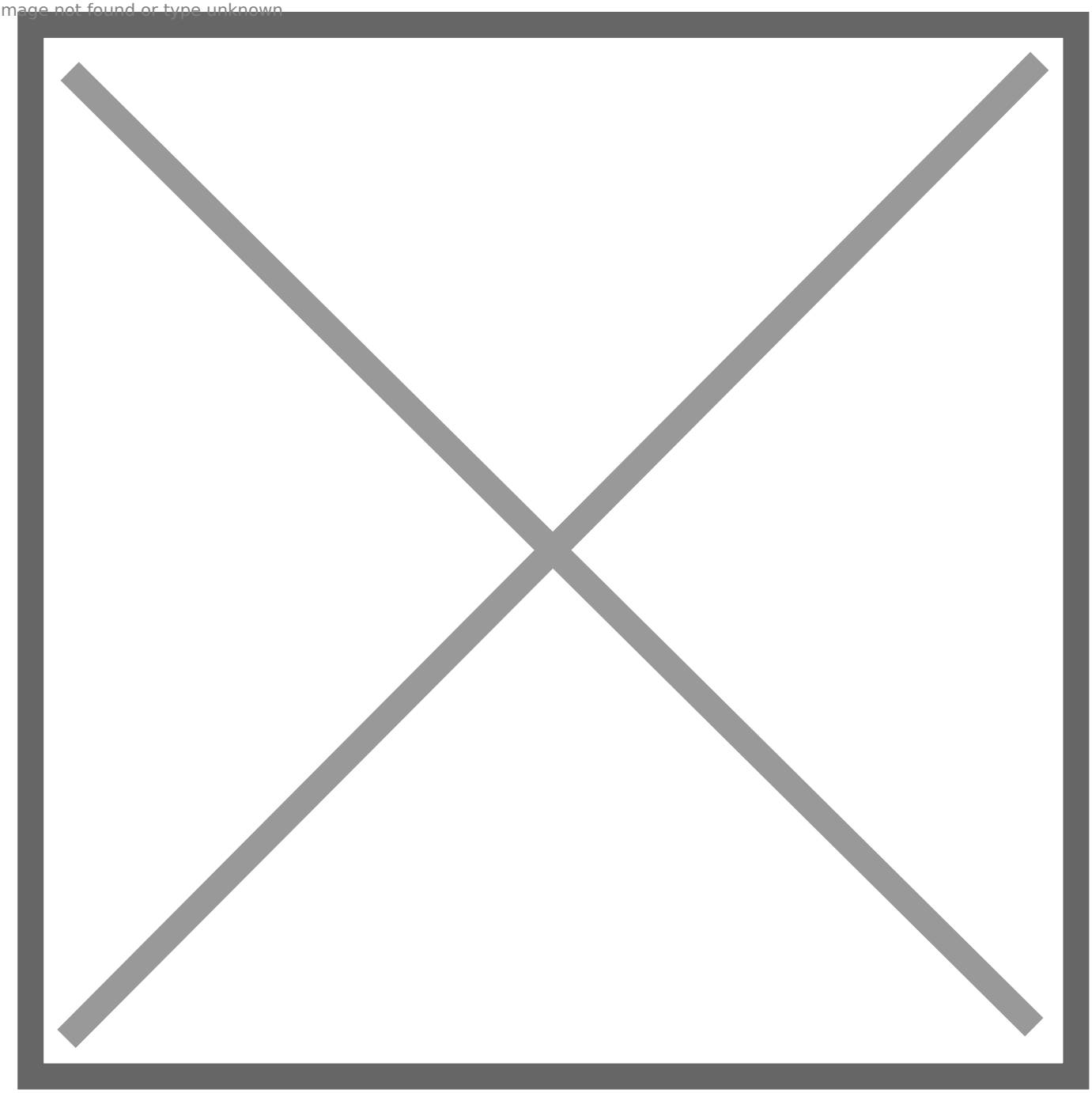
$ rm my_document my_doc_folder

Image not found or type unknown



The advantage of “**rm**” over “**unlink**” is that you can remove multiple symlinks with the “**rm**” command, which is not possible with the “**unlink**” command as shown in the following image:

Image not found or type unknown



Note that whether you use the “**unlink**” or “**rm**” command, do not use trailing slash “**/**” even if it is a directory.

Revision #1
Created 23 December 2023 15:14:06 by ColtM
Updated 7 August 2024 23:24:39 by ColtM