HOW-TO Use the Subversion svn+ssh Protocol

Created by Peter Jones, last modified on Feb 03, 2020

Purpose

As of Jan 27, 2020, there is a new protocol for accessing svn.maxim-ic.com repositories. This page expands on the ideas discussed in the original instructions that were sent out when the changeover happened.

Table of Contents

Original Installation Instructions

The original instructions were sent out to known Subversion users in the weekend of Jan 25-26, 2020; they included a private and public key pair, a registry file, and instructions on how to install everything.

Those instructions had the user place the key pair files in the Documents folder, though really, they can go anywhere.

They also had you download a copy of `putty.exe` from https://intranet.maxim-ic.com/cfointernal/itinternal/itss/IT_BRC/Lists/DepartmentDocuments/Putty .

The registry fix (they had you rename their attached file from `svn_putty.txt` to `svn_putty.reg`) adds in the `sgsvn` session for PuTTY; however, since they then give manual instructions for adding the real PuTTY session they want, and never used the `sgsvn` session, I don't know that it's really necessary.

They have you create a session called `svn.maxim-ic.com`, with the username of `apache`, and setting the location of the private key that it will use to the private key file they sent you. If you didn't choose your Documents folder, make sure you browse to the real location, rather than the location shown.

They show the example of using TortoiseSVN's Repo-browser to navigate to `svn+ssh://apache@svn.maxim-ic.com/layout_wip` to prove that your setup works.

Additional Information

Private Key File Location

As was mentioned above, the private key file can go anywhere (including the Documents folder where they recommended).

However, as a word of caution: if you place those files in any folder that can be accessed through OneDriveForBusiness or is backed up by Crash Plan, then anyone – like IT – who have access to those accounts would theoretically be able to grab those files and spoof you; then again, since they were emailed unencrypted, anyone in IT who has access to the Outlook email server already has access to those files (and IT has access to anyplace on our laptop drives as well).

PuTTY

IT provided an intranet download location for `putty.exe`. Personally, I would recommend going to the official website (https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html), and checking for the newest, in case CFO/IT's site is out of date. I would also suggest downloading `puttygen.exe` from the official location (making sure to grab the 64-bit version), and placing it in the same folder as `putty.exe`.

Username: apache

They didn't explain *why* the username must be `apache`, rather than not specifying a username in the URL. I am not an expert, but from what little I know of the `svn+ssh://` protocol, it can use different usernames on the URL (ssh) side compared to the backend (svn repo). To make it easy to maintain, they created one ssh user, apache; when that apache user logs in over ssh, they compare the private key to their database, and map each *key* to a specific SSO username, and that's the username that svn (and thus your commit history, etc) sees. (There is probably more to it than that.)

Repository Access and FAQs

Checking out repositories / appropriate URLs

Using the syntax they showed, you can use the Repo-browser to find your repository, and then initiate the checkout from there. However, I think they didn't explain the new repo URL structure sufficiently.

If you previously connected to a repository using

```
https://svn.maxim-ic.com/svn/<reponame>
```

you would use

```
svn+ssh://apache@svn.maxim-ic.com/<reponame>
```

The changes made:

- change the protocol from `https` to `svn+ssh`
- add `apache@` before the hostname to indicate ssh username
- remove the svn/ from the URL path: the repository name comes directly after the host now

Other than the URL for checkout, everything else in TortoiseSVN should work as you are already accustomed to.

apache@ may be optional: Per 2020-Feb-03 experiments, assuming you correctly setup PuTTY per their instructions, with the **PuTTY > Connection > Data > Auto-login username** set to `apache`, you shouldn't need to include `apache@` in all your URLs.

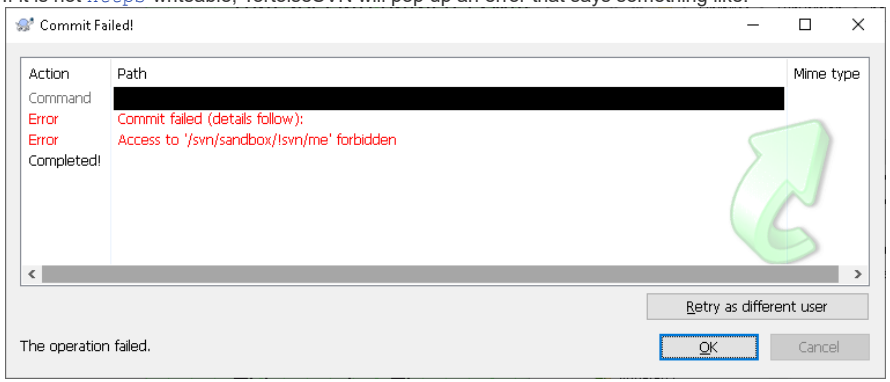
Is it required? What about for other (non-layout) repos?

This change was spawned by the more-efficient transfer capabilities, which will help with huge layout projects.

As of 2020-Jan-30, it appears that they *have* started disabling `https`-access for certain repositories; not all repositories have had `https`-access disabled yet (for example, the `ltx_mx/test_programs` repo is still `https`-writeable).

If you have a checkout on your laptop that was using `https` for one of the affected repos, you will have to either do a new checkout using `svn+ssh` , or use **TortoiseSVN > Relocate** (as described below) to easily switch your checkout from one protocol to another.

If it is not `https`-writeable, TortoiseSVN will pop up an error that says something like:



If you get that error, you will have to do a new checkout with the new `svn+ssh` protocol.

Compatibility

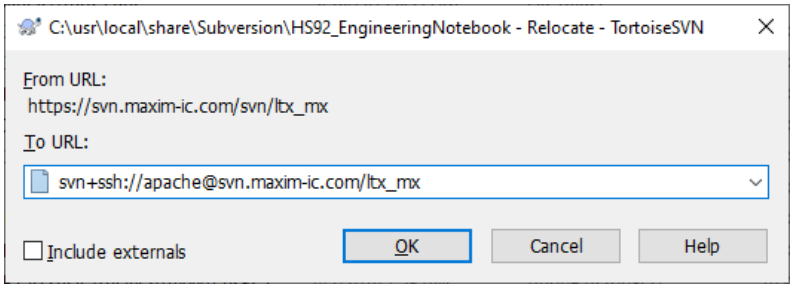
This protocol change doesn't change anything in your projects themselves.

If you have a checkout on one machine (like your laptop) using `svn+ssh`, and on another machine or folder (like on the LTX-MX tester) using `https`, **both** will should work okay, as long as they don't disable `https`-access for that repository.

Changing the URL for Existing Checkouts

If you have a checkout ("working copy") on your machine using the `https` protocol, and you want (or need) to change it over to the `svn+ssh` protocol, you can use the svn relocate command.

In Windows, you can use **TortoiseSVN > Relocate** from the top of your checkout. For example, I had a checkout of `https://svn.maxim-ic.com/svn/ltx_mx/test_programs/hs92_t48_ft/EngineeringNotebook`, and ran **TortoiseSVN > Relocate** ; it gave me this dialog:



So you can see that it ignores the path within the repository; it will just give you the top-level repository URL, which you need to convert to the new syntax in the **To URL** entry box.

From a command line, you can use `svn relocate` (see the svnbook page on `svn relocate` for correct syntax) to accomplish the same thing.

(Thanks to @Tim Severance for pointing this `svn relocate` feature out. I had tried and failed with `svn switch`.)

Using in a Web Browser

FAQ: "I used to use my web browser to navigate through the files. That doesn't work with `svn+ssh://apache@svn.maxim-ic.com/<reponame>`".

The `https` protocol still works for read access on all repositories. Use the `https` URL if you want to navigate your project using your web browser.

```
https://svn.maxim-ic.com/svn/<reponame>
```

TortoiseSVN Command Line Client

If you installed and use the TortoiseSVN Command Line client (for example, if you like running svn from cmd.exe or PowerShell, or if you want to be able to make commits through a batch file or other automated process), then it *might not* work out-of-the box for you for the `svn+ssh` protocol at the command line.

When TortoiseSVN does a commit or other write-access command from the GUI, Tortoise knows which helper it needs to be able to use the `svn+ssh` protocol. However, from the command-line environment, it cannot just "look it up"; instead, it's looking for the value of the `SVN_SSH` environment variable, and will use that. That variable defaults to empty. If it is empty, the svn client will prompt you for the ssh password, which we don't have, and cannot find your private key file. If you `Ctrl+Break` out, you might get an error similar to

```
password: ^C
svn: E170013: Commit failed (details follow):
svn: E170013: Unable to connect to a repository at URL 'svn+ssh://apache@svn.maxim-ic.com/sandbox/petercj/develop/scripts'
svn: E210002: To better debug SSH connection problems, remove the -q option from 'ssh' in the [tunnels] section of your Subversion configuration file.
svn: E210002: Network connection closed unexpectedly
```

What you need to do is open up your Windows user environment variables editor: click the Windows button, then type "environment", and select either "Edit environment variables for your account" or "edit the system environment variables"; both will open the **System Properties > Advanced** dialog box (which you can get to from a variety of other methods as well) – the second will prompt for UAC and maybe an Avecto Defendpoint "are you sure" box. Click on **Environment Variables** tab, button. In the **Environment Variables** dialog, add a **New** variable (doesn't really matter whether it's User Variables or System Variables, though System Variables

requires UAC elevation before entering the dialog). Set the `SVN_SSH` variable to `C:\\Program Files\\TortoiseSVN\\bin\\TortoisePlink.exe`, and click **OK** until all those dialogs are gone. The next time you open a command prompt or run your batch file, subversion should be able to commit properly.

Caveat: If you have 32-bit TortoiseSVN on a 64-bit OS, you will need to set it to `C:\\Program Files\\TortoiseSVN\\bin\\TortoisePlink.exe,` instead.

Warning: If you use single backslashes, like `C:\Program Files\TortoiseSVN\bin\TortoisePlink.exe`, you will get an error (below), because Tortoise will read the environment variable into a C-style string, which will use the backslash as a meta-character rather than a backslash, and then pass that corrupted string to on to try to execute the command, which it won't be able to find. The error will look something like:

```
svn: E170013: Commit failed (details follow):
svn: E170013: Unable to connect to a repository at URL 'svn+ssh://apache@svn.maxim-ic.com/sandbox/petercj/notebooks'
svn: E170012: Can't create tunnel
svn: E720002: Can't create tunnel: The system cannot find the file specified.
```

(If you have a full installation of PuTTY, rather than just putty.exe from the IT / CFO site, you already have a compatible copy of plink. You could set SVN_SSH to point to that plink.exe, using double-backslashes in the path, if you desire; both will work.)

Linux

This guide for using the new `svn+ssh` protocol on linux was not vetted by IT. However, these instructions worked for me.

The svn client on your linux box must be compiled with `svn+ssh` support.

You will need to make a copy of your private key in a format that svn+ssh on the tester will understand (OpenSSH private key format). You can use the `puttygen.exe` tool (mentioned above) to convert the PuTTY private key to an OpenSSH private key. Run puttygen.exe, **File > Load Private Key**, and browse to your private key file. Use **Conversions > Export OpenSSH Key** (not **Export OpenSSH Key (force new file format)**), and give it a name like `svnssh_private_key_linux`. Copy that file to your linux account (use WinSCP, or PuTTY's psftp or pscp, or however else you normally copy files to the linux box – see the tester-specific LTX-MX section below for more ideas for how to transfer files). Place that new file in `~/.ssh` .

In Linux command prompt, change to `~/.ssh` directory. Make sure that the private key file has no "group" or "other" permissions (`chmod 0600 svnssh_private_key_linux` to make sure). Next, edit ~/.ssh/config (creating it if it doesn't already exist). It should contain at least:

```
Host svn.maxim-ic.com
    User apache
    IdentityFile ~/.ssh/svnssh_private_key_linux
```

This tells svn+ssh that you will want to use the ssh-user-name of apache when connecting to `svn.maxim-ic.com`, and tell it where to find your private key.

For the URLs, since you have defined the username in the config file, you don't need the apache@ in the URL, though it doesn't hurt. Thus, either of these would work:

```
svn+ssh://apache@svn.maxim-ic.com/ltx_mx/test_programs/...
svn+ssh://svn.maxim-ic.com/ltx_mx/test_programs/...
```

Testers / Test Program Repositories

There are a few tester-specific things that should be clarified.

Windows: multi-user tester accounts

On a tester like the SPEA DOT400, where we perform offline development at our desks with our own Windows accounts, but are all sharing the single Windows account on the testers, we do *not* want to all copy our private keys to the tester; that's bad from a security standpoint (anyone with access to the tester can pretend to be you); it's also bad from a confusion standpoint ("why is my commit showing up under another username?"). Because the two protocols are *not* mutually exclusive, I recommend you use `svn+ssh` at your desk, but `https` from the tester.

Unfortunately, with some repos disabling `https`-access, that may not be an option for much longer. This page will be updated if it's discovered that's the case already.

LTX-MX

This guide for Linux (above) was developed on the LTX-MX using the svn v1.8.4 client which was loaded on the ltx-mx for handling svn 1.8 repositories, so I know that version **does** include the `svn+ssh` support. I make no guarantees about the ancient svn v1.4.4 or other ancient version that the ltx-mx tester defaulted to. Running `svn --version` will tell you what version of `svn` you have.

As with the instructions above, you can copy the private key to your LTX-MX linux account using WinSCP or psftp or pscp; alternately, if you are in a location with the "ontap" mapping to the linux home directory, you can open Windows Explorer to a UNC host like `\\hlbontapla.maxim-ic.com\<username>` (which is an example for Beaverton).

LTX-MX: mxsvn

mxsvn v1.05.001 is compatible, once you have installed your private key per the Linux instructions, above.

mxsvn has been updated to v1.05.001, and is compatible with svn+ssh repositories. With the new version, it will still default to `https` repositories, but will work (for tagging, branching, etc) if you've checked out an `svn+ssh` working copy, and can make your initial checkout use `svn+ssh` if you create the repository using `mxsvn create --svnssh`. See the mxsvn Confluence page for more details.

If you have setup your paths as the mxsvn Confluence page indicated, the newest mxsvn should already be distributed to your testers and dev machines via EZRev.

No labels