Generate your PAT in git

PAT: glpat-xxxxxxxyyyyyyzzzzzzTOKEN

SSH KEY: ssh-yourSSHKEY-username@server-024fe

**SVN 2 Git migration steps:**

**Generate author list:**

svn log -q <SVN URL> | awk -F '|' '/^r/{sub("^ ", "", $2); sub(" $", "", $2); print $2"="$2" <"$2">"}' | sort -u > svn\_authorList.txt

**A script to write authors list in file:**

svn log "$SVN\_URL" --quiet | grep -E '^r[0-9]+' | awk '{print $3}' | sort | uniq > "$TEMP\_FILE"

> "$OUTPUT\_FILE" #empty/create file

while read -r AUTHOR; do

if [ -n "$AUTHOR" ]; then

echo "$AUTHOR = $AUTHOR <${AUTHOR}@company.com>" >> "$OUTPUT\_FILE"

fi

done < "$TEMP\_FILE"

**List all the folders under a specific SVN URL:**

svn list <SVN URL>

**Migration Steps**

**Do either 1.a, 1.b or 1.c as relevant**

**1.a) Copy history and tags from svn modules and not worried about branches, tags etc.**

**Create a folder** and navigate into that folder.

**Initalize git:** git init .

**Initialize svn repo:** git svn init <SVN\_URL> --no-metadata

**Send the author configuration:** git config svn.authorsfile ../svn\_authorList.txt

**Pull data from SVN:** git svn fetch

**1.b) If the svn module have standard SVN layout ('tags', 'branches', 'trunk')**

Here in git, you will map:

* trunk --> main or master branch
* branches --> Git branches
* tags/ --> Git tags
* deployTags/ --> Git tags (I think the deployTags also a tag in the project).

**Create a folder** and navigate into that folder.

**Git SVN clone:** git svn clone <SVN\_URL> --stdlayout --prefix=svn/ --no-metadata .

Note:

* --***stdlayout*** flag will automatically detects SVN standard layout (trunk, branches, tags).
* The **.** at the end of above command denotes current directory.

**Send the author configuration:** git config svn.authorsfile ../svn\_authorList.txt

**Pull data from SVN:** git svn fetch

**1.c) If the svn modules have custom tags, custom branches etc**

**Create a folder** and navigate into that folder.

**Git SVN clone:** git svn clone <SVN\_URL> --trunk=trunk --branches=branches/\* --tags=tags --tags=deployTags --prefix=svn/ --no-metadata .

Note:

* If you have custom tags (ex: QA, STAGE, RELEASE etc), use branches=branches/\*
* If there are custom tags (ex: ***deployTags***) include them in --tags=<tag name>
* You can combine --stdlayout swith with other switches like --tags, --branches etc.

**Send the author configuration:** git config svn.authorsfile ../svn\_authorList.txt

**Pull data from SVN:** git svn fetch

**2) Convert SVN branches to Git branches**

for branch in $(git branch -r | grep "svn/branches/" | sed 's/svn\/branches\///');do

git branch "$branch" "svn/branches/$branch"

done

**3) Convert SVN tags to Git tags**

for tag in $(git branch -r | grep "svn/tags/" | sed 's/svn\/tags\///');do

git tag "$tag" "svn/branches/$tag"

done

for tag in $(git branch -r | grep "svn/deployTags/" | sed 's/svn\/deployTags\///');do

git tag "$tag" "svn/branches/$tag"

done

**4) Cleanup**

After creating Git branches and tags, you can remove svn/ prefixes:

git branch -r | grep "svn/" | while read ref; do

git branch -d -r "$ref"

done

**5) Verification**

Run **git branch** to list all brances

Run **git tag** to list all branches

Verify that commit history and file content have been preserved.

**6) Push to Remote Git Repository**

**git remote add origin <GIT\_REMOTE\_URL>**

**git push -u origin --all**

**git push --tags**

**NOTES**

**a. UNIX command to find all the files:** $ find . -type f -name "\*.pem"

**b. UNIX command to find \*.pem files and replace it with something else (qfn\_1203):** $ find . -type f -name "\*.pem" -exec bash -c 'mv "$0" "${0%.pem}.qfn\_1203"' {} \;

**c. Find a specific file/files-with-extensions (ex: .pem) in git's branch history:** git log --name-only --oneine | grep '\.pem'

**d. Remove file/files-with-extension (ex: .pem) files from every commit in the history:**

i) Filter branch:

git filter-branch --force --index-filter 'git rm --cached --ignore-unmatch \*.pem' --prune-empty --tag-name-filter cat -- --all

ii) Then clenaup the backup references created by git filter-branch

*rm -rf .git/refs/original/*

*git reflog expire --expire=now --all*

*git gc --prune=now --aggressive*

**e. List recent push in SVN:** svn log --stop-on-copy <https://svnserver.com/infrastructure/svnrepos/apache_haproxy>

**f. Clone svn without tags and you need to create tags manually:** git svn clone -r1:HEAD --no-minimize-url --stdlayout --nometadata --authors-file authors.txt

**g. Convert svn branches and tags into local git:**

*git tag tagname tags/tagname*

*git branch branchname origin/branchname*

**h. Get details of SVN repository:** svn info <https://svnserver.com/infrastructure/svnrepos/apache_haproxy>

**g. Git fetch svn by chunks:** git svn fetch --revision checkpoint#1:checkpoint#2

ex: git svn fetch --revision 0:500

**Troubleshooting**

**a. Git push was getting rejected because master is a protected branch!**

solution: navigate to git project/-/settings/repository#js-push-rules and click 'unprotect'.

**b. Git push was getting rejected because of .pem files!**

solution: navigate to git project/-/settings/repositorys and expand 'Push rules' and uncheck 'prevent pushing secret files'.