Git Repository Access

The Shibboleth's project's Git repositories are hosted at git.shibboleth.net, and may be accessed in a number of different ways.

Anonymous Access

Public access to most repositories may be performed anonymously in one of three ways:

- git clone using the git protocol
- git clone using "smart HTTPS"
- using a web browser to browse the repository

The decentralised nature of Git means that as a non-committer, you can still work with a repository and develop on a local branch. Use techniques such as git rebase to keep your branch up to date with the upstream repository. See this section of the Pro Git book for tips.

If you develop code as a non-committer and would like to contribute it back to the project, please start a discussion on the development list. You will need to agree to the contribution policy before we can accept a contribution. You can then git rebase your branch one final time and submit changes using git request-pull or git format-patch/send-email.

Git Protocol Access

You can clone a repository using the git protocol as follows:

```
$ git clone git://git.shibboleth.net/repository-name
```

A trailing .git is permitted, but not required.

This access method supports git clone, git fetch and git pull. Making changes to the repository is not permitted.

This protocol is very fast, but your client can not authenticate the server. If you require that the server authenticate itself, you can use Smart HTTPS access.

Smart HTTPS Access

You can clone a repository using Smart HTTPS as follows:

```
$ git clone https://git.shibboleth.net/git/repository-name
```

Note that the URL includes a "/git" component to distinguish Git client access from browser access. A trailing .git is also permitted, but not required.

This access method supports git clone, git fetch and git pull. Making changes to the repository is not permitted.

gitweb Browser Access

Most repositories are available for browsing at https://git.shibboleth.net/view/?p=repository-name.git

Note that the .git at the end of the repository name is required in this case.

Accessing https://git.shibboleth.net/view/ will present you with a list of the available repositories.

Committer Access

Making changes directly to any of the Shibboleth project Git repositories can only be done by project committers. Full repository access is performed using the ssh protocol:

```
$ git clone git@git.shibboleth.net:repository-name
```

SSH host key fingerprints are:


We use the gitolite software to authenticate users using their SSH keys and to give each user appropriate permissions.
The current gitolite administrators are Ian Young and Scott Cantor. Contact a gitolite administrator if you need to add or remove an SSH key (you may have more than one, for example if you use separate keys for each workstation you use) or if your permissions appear to be incorrect.

You can find out which repositories you have access to, and the permissions you have over each, using the `info` subcommand, as follows:

```
$ ssh git@git.shibboleth.net info
hello iay, this is git@shibboleth running gitolite3 v3.6.2-18-gb6050a8 on git 2.3.4
  C    CREATOR/..*
  R W  some-repository
  R W  testing
```

This shows that the user `iay` has read and write access to the repositories `some-repository` and `testing`, as well as the ability to create new repositories using their `gitolite` user ID as a prefix, such as `iay/another-repo` or `iay/folder/yet-another`. You can read more about the facilities available from `gitolite` in this summary.

The `testing` repository is available to all authenticated users, as well as being visible through anonymous access.

**Personal Repositories**

Committers may also create personal repositories in a directory named after their `gitolite` identifier. These are initially only visible to their creator.

Personal repositories are never visible through `gitweb`, but can be made available to other committers or the general public by setting their permissions appropriately. Anonymous access is enabled by granting read permission to the user `daemon`.

One use of personal repositories is for long-running feature development which for one reason or another you don't want to make public. For example, you may wish the final feature commit to compress the work you have done on a private feature branch, so that the public repository only shows the final changes required. To do this, you'd can use a local repository and tools such as `git rebase` and `git merge --squash`, but using a personal repository allows that workflow to be extended with limited sharing and, of course, security against loss of the local repository.

To create a personal repository, just perform a clone operation and the repository will be created on the server side at the same time:

```
$ git clone git@git.shibboleth.net:iay/my-personal-repo
```

For more details about this facility, see the `gitolite` documentation.

**Personal Repositories and Jenkins**

To grant Jenkins permission to read a personal repository:

```
$ ssh git@git.shibboleth.net perms iay/my-personal-repo + READERS jenkins
```

The Repository URL in the Jenkins job configuration should then be: `git@git.shibboleth.net:iay/my-personal-repo`