In-Repo Authz

Design: Authz Stored Inside the Repository

It is desirable to store Authz files inside a Subversion Repository (potentially the same repository the file is written to protect). By placing the file in Subversion you gain versioning and the audit trail that comes with it. Additionally, it simplifies replication since your Authz file is no longer out of band.

This can already be achieved by writing hook scripts that export the Authz file from Subversion on commit. However, it is more difficult to install a hook script setup than it is to simply set a configuration parameter. The hook script solution is also very difficult to validate proper installation.

Behavioral Specification

The Authz file may be specified in one of four forms a location inside a repository, a relative path within the repository being accessed, a absolute file path outside of the repository or a relative file path outside of the repository. The file would then be read from the location specified, parsed and the results cached for a single connection as is already the case with the existing external file path implementation. There is one exception to the current behavior which is the --config-file option to svnserve which causes the Authz file specified in the config file to be cached for the entire time svnserve runs, this one case is troublesome.

These four forms allow for Authz to be used out of the same or a different repository and in the case of the SVNParentPath mod_dav_svn option or -d option to svnserve allows for each repository to contain its Authz file.

Within our layered design the changes being described here are being made to the server layers (mod_authz_svn which though not pictured at the linked diagram would be above mod_dav in the Apache box) and svnserve. The code for reading and parsing the authz file are in the repos layer, but is only used by the server layer if it decides to implement authz. Where we need to read the data out of the repository we would use the repos layer.

Format of the Authz Path

The four forms would take the following formats:

- `file:///repo/authz` : A string starting with `file://` followed by the absolute path to the repo followed by the path within the repo to the authz file, identical to what you'd provide to the svn client's cat command if you wanted to cat the file.
- `^/authz` : A string starting with `^/` and followed by the relative path within the repo. `^/` would be replaced by the full path to the current repo's root. This is the same as the path style used within the Subversion client itself to reference relative URL paths.
- `/path/authz` : A fully qualified path to the AuthZ file. On Unix systems it will likely start with `/` and on Windows would start with a drive letter.
- `path/authz` : A relative path to the AuthZ file. Does not start with a `/` or a drive letter. When used with mod_authz_svn the path would be resolved relative to the ServerRoot of the httpd server (AuthzSVNAccessFile) or the conf dir of the repos being accessed (AuthzSVNReposRelativeAccessFile). When used with svnserve it would be resolved relative to the conf dir of the repository. If you wished to reference a relative path which started with `^` you could preceed it with a `.` e.g.: `./^/authz`

SVNParentPath Operation

mod_authz_svn uses two different configuration options with different roots. AuthzSVNAccessFile which is relative to the ServerRoot and AuthzSVNReposRelativeAccessFile which is relative to the conf directory of the repos being accessed, in order to allow different Authz files for different repos with SVNParentPath.

Both of these commands should take the two new formats. Neither of the two new formats (`file://` or `^/`) care about the root for relative file paths. There is a small amount of extra overhead for AuthzSVNReposRelativeAccessFile in finding the conf dir of the repo, but we can shortcut it when we see `file://` or `^/` values.

svnserve --config-file Operation

--config-file presents a special problem for this change. Since the Authz is loaded and cached at startup the `^/` (in repo relative) format can't be cached and must be loaded on each connection (similar to the way svnserve behaves without --config-file).

Security

One obvious question is the security of the Authz file once it is stored within the repository. This would be left up to the user to implement via the Authz file themselves. Appropriate documentation would be written.

Another common question is what happens if you commit and Authz file that disallows access to everyone. RA local access `file://` does not implement authz. So an admin with access to the filesystem that the repository can commit a fixed authz file.

Performance

This has been implemented on the ^/subversion/branches/in-repo-authz branch. At current no optimization has been made to use the same repos object for reading the authz file and for servicing the request since at least in the DAV case separate httpd modules are handling authz and servicing the request.

Testing has shown very tiny decreases in performance when using the authz configuration stored in the repo. The following test results were conducted against a repo containing an authz file that had 1MB of random data in it with 1k changed every rev for 98 revs after the initial rev and then a simple authz config committed in revision 100. Timings were taking from executing svn-bench null-list against the repo 10,000 times.

test	trunk@1423201	in-repo-authz@1423201
httpd AuthzSVNAccessFile repos relative URL	n/a	5m43.718s
httpd AuthzSVNAccessFile absolute URL	n/a	5m43.872s
httpd AuthzSVNAccessFile absolute OS path	5m40.697s	5m41.442s
httpd AuthzSVNAccessFile relative OS path	5m39.892s	5m40.889s
httpd AuthzSVNReposRelativeAccessFile repos relative URL	n/a	5m41.919s
httpd AuthzSVNReposRelativeAccessFile absolute URL	n/a	5m41.805s
httpd AuthzSVNReposRelativeAccessFile absolute OS path	5m39.252s	5m40.966s
httpd AuthzSVNReposRelativeAccessFile relative OS path	5m39.811s	5m40.257s
svnserve repos relative URL	n/a	4m17.670s
svnserve absolute URL	n/a	4m17.994s
svnserve absolute OS path	4m10.292s	4m11.170s
svnserve relative OS path	4m10.514s	4m12.479s

What this is not

This design is not a long term solution to ACLs. Building in ACLs as we might choose to do with FS2 would still be highly desirable. This is nothing more than an expansion of the existing Authz capabilities to allow easier administration.

It is also not a good general model to move things like hook scripts into the repository. While the Authz file is only used by mod_authz_svn and svnserve (both of which have direct access to the repository) the hook scripts are executed by the operating system on behalf of the Subversion server and would have to be written to the file system.