

CVS Repository at Berkeley

by Steve Masover

Table of contents

1 CVS Repository at Berkeley.....	2
1.1 Overview.....	2
1.2 Process for Development and Review (of this document).....	2
1.3 Summary.....	3
1.4 Structure of CVS Repositories.....	3
1.5 Managing Read-Permissions.....	5
1.6 Managing Write-permissions.....	7
1.7 Managing Passwords in the Repositories.....	7
1.8 Automated Deployment of Applications from CVS.....	8

1. CVS Repository at Berkeley

DRAFT - PROPOSAL

Last modified: 18 April 2005

1.1. Overview

This document is a proposal for configuration and use of a CVS instance as a repository of code and other files for the UC Berkeley campus. It is being developed by staff in IS&T-SIS and will be brought to a broader audience in IS&T for negotiation and implementation of a CVS instance that will serve the campus as a whole.

The structure and permissions scheme described in this document are intended to balance the following goals:

- Conformity to the [UC Berkeley Campus IT Security Policy](#)
- Sharing code among campus developers to facilitate IT staff development, best practices, and code re-use
- Adherence to intellectual property, licensure, and security-related requirements regarding source-code access and distribution
- Appropriate preservation of revenue-stream potential for campus IT units funded on a recharge model

1.2. Process for Development and Review (of this document)

This document is being developed and reviewed in the following stages. The next stage of review to be undertaken or completed is listed in **strong** text.

- SIS unit directors: George Suennen, Helen Lee, Randy Ballew
- SIS Object-Oriented Application Development Team
- **SIS managers: all SIS managers not included in above groups**
- CCS system administration staff (re: implementation issues)
- CCS Systems and Data Administration Associate Director (re: process management)
- Additional IS&T units interested in using CVS

1.3. Summary

This substance of this document is divided into several sections:

- [Structure of CVS Repositories](#)
- [Managing Read-Permissions](#)
- [Managing Write-permissions](#)
- [Managing Passwords in the Repositories](#)
- [Automated Deployment of Applications from CVS](#)

The repository structure is designed to facilitate code reuse among UC Berkeley staff, without compromising appropriate organizational security.

Read permissions on the repository files are effected through controlled membership in [UNIX groups](#) on the server hosting CVS.

Write (a.k.a. "commit" or "update") permissions are effected at any level of a repository's directory tree via management of a configuration file.

Code committed to the repository is expected to utilize appropriate procedures to either separate passwords from the codebase or very tightly restrict read-access to the codebase; the former is preferred, and a set of model processes for effecting this goal are under development.

Automated deployment of applications is very strongly preferred to manual deployment, in order to achieve reliable replication of deployment steps and to archive for the organization deployment intricacies that may otherwise exist only in the memories of its staff.

1.4. Structure of CVS Repositories

In outline, the repositories will be implemented as follows:

- Projects (source code and/or released binaries) will reside in a projects directory that is subdivided into organization-specific directories:
 - the org-specific directories' read permissions will be regulated via UNIX groups with relatively broad (but controlled) membership; while,
 - more finely grained restriction on read permission will be regulated via UNIX groups with more restricted membership, as described [below](#).
- Organization-files that are not source code, per se, may reside in organization-specific, non-project repositories should these be desired. Such files could include HTML source for an organization's website(s), administrative data for which version control and centralized backups are desirable, etc.
- Third-party software utilized by deployed applications will reside in third-party

repositories, to which access will be broad (except as required by licensure or security restrictions).

- Data used for accessing secure resources (such as passwords) will be protected by storing it in a repository to which access is highly restricted.

The repository structure is a directory tree on the CVS host's filesystem. The following representation of the proposed tree may clarify the preceding narrative description. Note that the existence of a CVSROOT directory implies a separate repository; each repository has separately configurable write (commit) permissions, as described [below](#)

```

drwxrwsr-x [...] userid cvsasd [...]
/berkeley/projects/ist/adminproj01/CVSROOT
drwxrwsr-x [...] userid cvsasd [...]
/berkeley/projects/ist/adminproj01/...
drwxrwsr-x [...] userid cvsasd [...]
/berkeley/projects/ist/adminproj02/CVSROOT
drwxrwsr-x [...] userid cvsasd [...]
/berkeley/projects/ist/adminproj02/...
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/adminproj03/CVSROOT
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/adminproj03/...
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/servant/CVSROOT
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/servant/...
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/streak/CVSROOT
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/streak/...
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/CVSROOT
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/stusubproj01a
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/stusubproj01b
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/stusubproj01c/src
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/stusubproj01c/doc
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/stusubproj01c/...
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj01/...
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj02/CVSROOT
drwxrwsr-x [...] userid cvssis [...]
/berkeley/projects/ist/stuproj02/...
drwxrwsr-x [...] userid cvsist [...]
/berkeley/projects/ist/stuproj03CVSROOT
drwxrwsr-x [...] userid cvsist [...]

```

```

/berkeley/projects/ist/stuproj03/...
    drwxrwsr-x [...] userid  cvssxx  [...]
/berkeley/projects/ist/stuproj04/CVSRROOT
    drwxrwsr-x [...] userid  cvssxx  [...]
/berkeley/projects/ist/stuproj04/...
    drwxrwsr-x [...] userid  cvsabc  [...]
/berkeley/projects/ist/...
                                {insert other locally-developed IS&T
projects here}
    drwxrwsr-x [...] userid  cvsucb  [...]
/berkeley/projects/ucb/...
                                {insert locally-developed projects
appropriate
                                to campuswide sharing here}
    drwxrwsr-x [...] userid  cvsdef  [...] /berkeley/projects/...
                                {insert other Berkeley organizations
here}
    drwxrwsr-x [...] userid  cvssecur [...]
/berkeley/projects/ist/streek-secure/CVSRROOT
    drwxrwsr-x [...] userid  cvssecur [...]
/berkeley/projects/ist/streek-secure/...
    drwxrwsr-x [...] userid  cvssis  [...] /berkeley/sis/CVSRROOT
    drwxrwsr-x [...] userid  cvssis  [...] /berkeley/sis/website
    drwxrwsr-x [...] userid  cvssis  [...] /berkeley/sis/...
    drwxrwsr-x [...] userid  cvsasd  [...] /berkeley/asd/CVSRROOT
    drwxrwsr-x [...] userid  cvsasd  [...] /berkeley/asd/...
    drwxrwsr-x [...] userid  cvsxyz  [...] /berkeley/{insert other
UCB organizations here}
    drwxrwsr-x [...] userid  cvsucb  [...]
/third-party/jboss/CVSRROOT
    drwxrwsr-x [...] userid  cvsucb  [...] /third-party/jboss/...
    drwxrwsr-x [...] userid  cvsucb  [...]
/third-party/apache/CVSRROOT
    drwxrwsr-x [...] userid  cvsucb  [...] /third-party/apache...
    drwxrwsr-x [...] userid  cvsucb  [...]
/third-party/apache-jakarta/CVSRROOT
    drwxrwsr-x [...] userid  cvsucb  [...]
/third-party/apache-jakarta...
    drwxrwsr-x [...] userid  cvsucb  [...] /third-party/...
                                {insert other third-party software
here}

```

1.5. Managing Read-Permissions

Read permissions are effected via ownership of repository-level directory trees by UNIX groups, and appropriately controlled membership in those groups.

Unless there is a business reason that necessitates more finely grained control, read-access to projects will be granted to a broad spectrum of users (e.g., to IS&T developers in the `berkeley/projects/ist` repository); and an even broader spectrum of developers in the `third-party/` repositories. More finely grained control of read access to project repositories will be predicated on business reasons, such as any one or more of the following:

- restrictions on intellectual property contained in the repository's files;
- licensure issues that restrict access to or distribution of the repository's files;
- passwords in source code where it is impossible or it is deemed too resource-intensive an effort to implement a password model such as that described [below](#)
- other security-related issues that are impossible or that are deemed too resource-intensive to resolve except by restriction of read-access; or,
- revenue-stream potential in the repository, such that access to all cvs users would materially compromise the value of that potential.

An example of the variety of read-access that may be controlled in this way may help illustrate these ideas. The example is consistent with the [filesystem example](#), above.

restrict read-access to medium-scope group (e.g., IS&T-SIS or IS&T-ASD)

An organization might determine it necessary to restrict read access to certain of its project source code due to revenue-stream potential, and therefore a UNIX group to which only that organization's staff may belong would own the repositories of concern (e.g., `/berkeley/projects/ist/*` directories owned by `cvssas` or `cvssis` in the [example](#);

restrict read-access to broader-scope group (e.g., IS&T)

the same organization might be responsible for code that does not require "extra" restriction for any of the listed reasons, and that code would reside in repositories owned by a UNIX group whose membership has broader scope (e.g., `/berkeley/projects/ist/*` directories owned by `cvssist` in the [example](#);

restrict read-access to narrow-scope group (e.g., project team)

a last category of project maintained by the same group might require that passwords be stored in the same files as the codebase, and so the codebase would reside in a repository owned by a UNIX group whose membership was more narrowly restricted - e.g., to only those staff within the organization whose work included maintenance of the codebase compromised by embedded passwords (e.g., `/berkeley/projects/ist/stuproj04/*` directories owned by `cvssxx` in the [example](#).

1.5.1. Governing membership in UNIX groups

Membership to UNIX groups, which governs read access to repositories, will require a coordinated and efficient mechanism for submission and approval; details must be negotiated

with CCS. Development and implementation of processes and mechanisms to effect appropriately approved assignment of UNIX group membership is a pre-condition for re-organizing the CVS repository and broadening access to it. The following rules are expected to be implemented:

- Creation of new projects, designation of the UNIX groups associated with that project's repository (and, implicitly, whether the project falls into a broad-spectrum or a read-restricted category described above), as well as assignment of project-lead-group membership pertaining to that project must be approved by at least one of a small number of appropriate staff, e.g., a designated individual or her/his backup for each of the major IS&T departments using repositories in the `/berkeley/projects/ist` tree shown in the [example](#), above.
- Project leads identified by the process described in the prior bullet-point must approve assignment of project(s)-specific UNIX group membership to users with logins on the CVS host. For example:
 - any one of a small number of designated individual in each major IS&T department must approve assignment of "cvsist" group membership for projects that are widely available ("broad spectrum" as described [above](#));
 - an individual or individuals designated for a major IS&T department must approve assignment of group membership specific to that department (e.g., "cvsasd" or "cvssis") for projects appropriate to share among all programmers in an organizational unit such as IS&T-ASD or IS&T-SIS;
 - leads for a specific project must approve assignment of the narrowest category of read-restricted-project group membership (those projects that meet business criteria that necessitate the narrowest read-access restriction such as that described in the last of the [read access examples](#), above).

1.6. Managing Write-permissions

Commit permissions are configurable via the `avail` file in the CVSROOT directory of each repository. Commit permissions may be set on an entire repository and/or on any part of a directory tree, down to the level of individual files, as deemed appropriate by project leads. Control of the `avail` file can be configured as appropriate (e.g., to project-leads, to a CVS administrator for multiple projects, etc.).

More information on the `avail` file and how it works may be found [here](#)

1.7. Managing Passwords in the Repositories

Without reaching full definition on the precise model to be used, it was agreed that some method(s) would be employed to gather restricted password data (such as database

passwords, LDAP bind passwords, etc.) in a highly restricted repository in CVS, where they will be stored in encrypted form. Access to this repository would be restricted to system administrators (who run deployments to QA and PROD servers), and a small set of IS&T staff who will be responsible for adding to and maintaining files in this repository (i.e., those who may appropriately have knowledge of QA and PROD passwords). This arrangement corresponds to the directory `/berkeley/projects/ist/streek-secure`, owned by the UNIX group `cvssecur` in the [filesystem example](#), above.

The method(s) to be used will allow developers to supply passwords known to them - but NOT committed to repositories to which they have access - for testing purposes (e.g., in order to run an application from a development machine).

It was agreed that development and implementation of method(s) to secure passwords in this way are pre-conditions for re-organizing the CVS repository and broadening access to it.

1.8. Automated Deployment of Applications from CVS

Deployments based on the `hydrant` and `servant` projects currently in use for CVS-archived projects are [Apache-Ant](#) driven processes that are strongly preferred to unscripted (manual) compilation and deployment.

It is expected that [Java](#) applications whose code and dependencies are stored in the CVS repository will be deployed using Ant scripts consistent with the `hydrant` and `servant` model.

For deployment of non-Java applications, it is expected that cost-benefit analysis will be applied to determine whether the benefits of Ant-driven automation outweigh the costs of implementing the deployment, and that an appropriate deployment method will be selected on the basis of that analysis.

`Hydrant` is documented [here](#), and `servant` is documented [here](#)