4-6-2008

The Homework Server

John Beckett
Southern Adventist University, jbeckett@southern.edu

The Homework Server:
a Learning Space for Introductory Web Classes

2008 Instructional Technology Conference at MTSU April 6-8, 2008

Author: John Beckett, DBA, Associate Professor of Computing, Southern Adventist University, jbeckett/at/southern/dot/edu

Abstract

The Homework Server is a true Web hosting platform which provides the student a realistic Web environment for project testing and turn-in while preventing other students from stealing their code. The Homework Server is a comprehensive environment based on freely available software. It is scripted in PHP/XHTML, integrates easily with university systems using IMAP authentication, and requires very little maintenance since most routine tasks are automated. The server described supports student applications that use XHTML, CSS, JavaScript, PHP5, and MySQL.

Purpose

The technology usually available is missing an important component: a learning space that meets the needs of both student and instructor. Commonly-used technologies have serious shortcomings.

- Submitting homework as .zip files or multiple attached files: Students may not be using a realistic server environment, and that the instructor/TA must implement the Web assignment themselves in order to grade it.
- Using existing personal Webspace: Hides components from the grader, raises significant privacy and intellectual property issues, and exposes assignments to pilfering by other students.
- Course Management Systems: No appropriate space for student Web projects that includes server-side functionality.

We have implemented a Homework Server for several years, with the following goals:

- Student projects are “turned in” by loading them on an industry-standard Web server, and may be tested by students in the same environment as the grader will use.
- Server side technology is supported including PHP5 and MySQL with phpMyAdmin.
- Access to a student project is limited to the student submitting it, and the instructor/TA.
- The instructor/TA may view the project either as a Web application per se, or as components.
- Institutional IT security is preserved, without the overhead of a new password-support contact point and without restricting security zones from which the server can be accessed.

- Zero ongoing management effort, and minimal configuration effort.
- Minimal cost for hardware and server software.
- Minimal coordination required for integration with campus IT infrastructure.
- Student projects are arranged in a central location for easy archival.

Methodology

In selecting a hardware platform candidate we look in the closet for a PC that is not fast enough to be interesting to anybody else, but refuses to die so we can't bring ourselves to place it into the waste stream. We have successfully used PC hardware in the range of 200mHz to 800mHz. Currently the Homework Server is implemented on a Macintosh G3 whose only enhancement is a 40 gigabyte hard drive. The only dollar cost we have expended for servers has been a new hard drive, since older drives tend to be unreliable.

The operating system used is the current "stable" release of Debian Linux. We do a minimal install (not including any GUI), plus the packages needed to support this application: apache2, php5, php5-imap, php5-cgi, php5-cli, mysql-server, ssh, and vsftpd. The server is run in a "headless" configuration, in which management interaction is through Web and SSH sessions.

Glue

A small set of scripts ties the technologies together. PHP was selected as the scripting language because it can be used for both server-side code and command-line scripting. The cron function of UNIX/Linux is used to connect Web-based account management activity with account changes in Linux.

Security Design

Since a major purpose of the Homework Server is separation of student efforts, it was necessary to establish a separate username/password namespace. Scripts are used to allow students to create their own accounts on the Homework Server, provided all of the following are true:

- They are using https:// access to create the account.
- They have a valid username/password on the institutional email server. (This link does not require cooperation of the IT department, although they were consulted in the design.)
- Their username is listed on the Homework Server in the file /home/hw/users.
- They enter a password different from that used for email.

Subsequent access by students uses ftp and http, and is unencrypted. The ftp server is VSFTPD, and uses chroot functionality to establish a "jail" for each user. The http server is Apache, which imposes password control and directory restriction using the .htaccess method. Instructors (which could include TAs and readers) can access any students' work on the server, but a student can only access his/her own work.

Operational Cycle

The Homework Server is reincarnated each summer. An incarnation begins with a trip to the closet. We are looking for a reliable computer that is not fast enough to merit real estate on somebody ‘s desk. It receives a new hard drive and perhaps some memory, hw.*domain* is set up for its Ethernet hardware address, and installation commences. (hw_old.*domain* is pointed to the former server, which is left online for several months to smooth the transition.)

Inevitably there are changes since the last year. For this reason, installation should be done by a Linux specialist. The end product of this phase is a functional server with a login for the instructor/TA that allows editing of /home/hw/users. The usernames of the authorized users are placed in that file manually. Once this is done, students can build their own accounts and change their passwords as needed.
During the academic term, students can upload projects to their space any time.

At the end of the year, the server's data is archived – and since the server itself is not valuable, we maintain it online as hw-old.*domain* for several months. This allows any pieces not captured by the archival to be recovered, and permits students to retrieve old projects they had failed to save.

While MySQL is not used for server management per se, it is available for student projects. At this point we do this manually, although it could easily be scripted. A MySQL database is created for the student in question, a user is created in the MySQL server configuration, and that student is granted rights to manipulate that database.

Results

Student Interaction

Students interact with the Homework Server in four ways:

- Access instructional materials such as videos specific to the Homework Server
- Build an account or change their password.
- Upload homework using ftp
- Test homework using http

Shell access is not extended to students for security reasons.

The Homework Server generates consistently positive feedback on student evaluations. It also provides for rapid-fire interaction between student and instructor/TA because only the artifact being corrected need be uploaded for it to be visible to the instructor/TA. This is particularly important in a distance learning environment.

Instructor Interaction

An instructor can do any of the functions permitted for students. Instructors familiar with UNIX/Linux may be granted shell access on request. Typical interactions include:

- Accessing student Websites using http.
- Downloading the day's backup to their PC for more intense scrutiny.
- Enabling a student to use the Homework Server (HW) by adding their username to the authorized user list.
- Providing a restorable backup of a student's work by ZIPping that student's portion of a given day's backup (something we have not yet done, but which would be easy to do).
- Archiving the backup of HW as of the end of a grading purpose, to be filed with class records.

Offline Grading

The backup script generates ISO files which can be easily mounted on either a Windows PC using Virtual Daemon Manager or a Macintosh directly using OS X. This capability allows for easy comparison of different stages of project development, and direct inspection of student Website artifacts. Installing XAMPP on the instructor/TA computer provides for a complete offline grading solution, especially if one disconnects the Ethernet plug to detect absolute URLs.

Conclusions

Limitations

Scalability is always a question on new projects. This has not been a problem while using HW at our location, but if we had larger numbers of students (perhaps in the range of hundreds), a faster server might be necessary. The HW technology can be easily split apart per class, however, so scalability is not a significant issue. Scalability issues involve class size and security between classes.

HW lacks a major function of a Course Management System: the ability to restrict submissions by date. The ease of looking into the past by mounting backups of a given date, however, allows the instructor to enforce due dates.

Since HW was developed using a Linux platform, ASP.NET is not supported. This shortcoming could be addressed in either of two ways: 1) Implementing cross-platform tools such as Mono, or 2) Re-implementing the server on a Windows Server platform using the WMI API.

Accomplishments

HW provides the environment needed for introductory Web classes, and is used at our institution for such classes in two different departments. Its use is often mentioned as a positive attribute in course evaluations, and greatly reduces the burden on instructors.

Recommendations

MySQL scripting: The existing scripts can be expanded to embrace automatic creation of MySQL databases and usernames, including additional usernames for read-only access to those databases.

Debian Packaging: The entire configuration could be set up as a Debian package, easing setup and installation.

FTPS: Previous attempts to implement secure FTP for student uploads have failed, because the system depends on "chroot" functionality to prevent cheating and a secure FTP server that implemented chroot seemed to be unavailable. This issue should be re-visited.

Quota management: I have considered imposing disk space quotas on students, but so far have not done so. I ask students to avoid having large amounts of unneeded materials on the server at backup time (5am), and this simple request has sufficed to keep the quota problem at bay. It should be noted that quotas add overhead, so a faster server might be required.

IIS: Previous to the Linux version, we used Microsoft IIS to implement a similar project. It is probably possible to script an IIS-based server to accomplish the same functionality. The choice of Linux was largely one of familiarity with the environment, and frustration with the cumbersome GUI required for IIS in this context.

VMWare: Using VMWare or a similar product, the entire approach to HW could change – allowing each student to have and manage their own virtual server, which in turn would enable use of more advanced technologies.

Resource Links

(Free unless flagged with "$")

CamStudio	http://camstudio.org/
Camtasia ($)	http://www.techsmith.com/camtasia.asp
Debian Linux	http://www.debian.org
Mono	http://www.mono-project.com/ASP.NET
Virtual Daemon Manager	http://www.daemon-tools.cc
XAMPP	http://www.apachefriends.org/en/xampp.html

This file and other materials presented at the session will be available at http://www.cs.southern.edu/jbeckett/itconf/