1 ColbyChem: a free web server for ISIS/Host

1.1 jwkuehne (at) colby.edu (John Kuehne) exclaimed:

- Date: Wed, 13 May 1998 10:23:31 -0400 (EDT)

```
Dear mod_perl gang,

The following is somewhat late in the "success story" thread of a few months
ago, but I think there might be some interest for the database crowd. Below is
a brief summary of a talk that I gave at a meeting in Philadelphia last week.
Sponsored by Molecular Designs Limited (MDL), the meeting was attended by
several hundred representatives of industry and government, and was concerned
with the problems related to large molecular and reaction databases, and their
use in combinatorial chemistry, drug discovery, etc.  (These are databases
consisting of molecular structures and their models, and reactions. A database
user can pose an sql in the language of chemistry - molecular structures
drawn with ISIS/Draw or ChemDraw - to find data that have substructure
similarity, conformationally flexible similarity, reaction similarity,
and much more. The structures, models, and reactions are displayed using
MDL's chime plugin, itself based on RASMOL, which renders 'live' 3-D drawings
that can be rotated and displayed in a number of ways from within the web page.)

*****************************************************************************



Last November, Dr. Shattuck proposed that we build a reaction database of
reaction mechanisms studied by Dr. Mundy and his colleagues, using MDL's
reaction database software. Furthermore, it was his idea that we make
this a web project open to all. Our first idea was to buy a license for MDL's
ChemScape server, which links NetScape Enterprise server to MDL's database
library. Unfortunately, the upgrade from our current MDL license to include
ChemScape server was too expensive, not to mention NetScape Enterprise server.

I started working on a web server based on Apache and mod_perl that would act
as a gateway to MDL's database software.

Although MDL's database server protocol is not public, they do provide a
command line interface called hostcli, which has most of the functionality
of the proprietary server. The use of hostcli is restricted to one machine,
but within that machine one may run any number of hostcli processes.

ColbyChem, the project that I presented at the meeting, makes use of hostcli
by opening it on a pseudoterminal for each database user. The novel aspect
of ColbyChem is its use of the integrated Apache/perl server running in

single user (-X) mode for each database user.

Because perl is embedded in Apache, dynamic variables are retained between
calls to the server children. Certain Apache packages use this to open a
persistent database connection to industry standard databases such as Oracle,
but this is not an option with proprietary interfaces, such as MDL's.

In order to adapt this to the idea of opening hostcli on a pty for each user,
I run a dedicated Apache/perl daemon for each user, in single-mode (-X), on a
separate port.  That way, each Apache daemon caches the perl program and
retains dynamic variables between calls. In essence, it becomes a new
```

application, composed of Apache and perl, running under my program. The effect is similar to an X client. The browser is like the X server.

Entrance to ColbyChem is through a dedicated login daemon running on port 9000. Upon receiving a valid login name, the daemon forks an Apache/perl daemon on a port specified in a password-like file, and transfers the browser to this new port. Authentication, which is very important here, is carried out entirely on this new daemon. The user supplies a password. ColbyChem encrypts it and compares with the encrypted password assigned to the user. If successful, ColbyChem forks and execs hostcli on the pty. It then records the IP number and sends back a cookie for secondary authentication upon browser reconnect. The cookie is different for each session, is not based only on an easily guessed system parameters like time or checksums, and does not reveal, to within the limitations of crypt(), the original or encrypted password. My solution for the cookie is to take the password, which is secret, and permute it using rand() seeded by time. The permuted cleartext password is then encrypted and sent back as the cookie. Thus, even if one knew the permutation order and cookie, it would still be impossible to recover the original password.

ColbyChem presents side-by-side frames. The left frame contains a query builder and controls for hit-list logic and display. The right frame displays the data indented in the natural hierarchy of the database. Models, structures, and reactions are displayed using MDL's chime plugin.

Essentially, ColbyChem is nothing more than a graphical front-end for hostcli, written in 1200 lines of perl. The heart of ColbyChem is two routines, each a page of code. The first routine, rd2perl, translates an export file from hostcli into a perl data structure that has the hierarchy of the original database, i.e. it imports the database into perl. The second routine

recursively descends the branches of this structure until it reaches the tips, whereupon it prints out the data indented to reflect the database hierarchy.

MDL has just delivered an Oracle interface to its molecular and reaction databases. This opens the possibility of using established packages for persistent database connnections that offer the flexibility of ChemScape server from within Apache/perl, without the novel hack of running dedicated daemons on separate ports for each user.

John Kuehne, Ph.D.
Information Technology Services
Colby College
4200 Mayflower Hill Drive
Waterville ME 04901

jwkuehne@colby.edu
207-872-3652

Table of Contents: