HOWTO: debug Perl (on AIX)

Contributed by Michael Felt

The title is quite broad - but I expect to gather quite a few bits of information regarding debugging perl (on AIX) as I continue resolving a problem with spamassassin.

I have been stuck for quite a while with trying to get spamassassin running on AIX. (See article "Part 1" for when I actually started on this.) The error I have been getting is not unknown to the community - but I have not found a ready solution anywhere. So I am starting to learn a bit more about perl and debugging perl.

The problem

Spamassassin is broken on AIX since an update above the 3.0.X level - which is what I was running before. The current version is 3.2.23 and gives the following error on many platforms.

```
# pwd
/data/prj/cpan/builds/Mail-SpamAssassin-3.2.3/t/log/d.spamd_allow_user_rules
# grep error * | head -1
spamd.err.1:[299250] error: setruid() not implemented at ../spamd/spamd.raw line 1027.

#
```

I have not yet located the exact location of the error statement but I have used the following perl program to start verifing the status and configuration of perl on AIX.

```
#!/usr/bin/perl -w

use Config qw(myconfig config_sh config_vars config_re);
use strict;

print config_vars(('d_setruid','d_setuid','d_seteuid'));
print config_vars(('d_dosuid','d_suidsafe'));
```

```
print config_vars(('d_setresuid','d_setreuid'));
```

The result is:

```
d_setruid='undef';
d_setuid='UNKNOWN';
d_seteuid='define';
d_dosuid='undef';
d_suidsafe='undef';
d_setresuid='undef';
d_setreuid='undef';
```

A command useful for more detailed information is:

```
$ perldoc Config
```

By the way, you can also use perl -V for a quick statement on configuration and compilation of perl

More is coming as I find it. Personally, not being an avid perl user I am finding it difficult to get the level and detail of information I want to resolve this problem. Most answers are either too broad, too detailed, or something to the affect of - if you dont understand that I cannot help you. Thus - self-study begins....

The code being executed is:

```
+1011     # support non-root use
 +1012     if ( $opt{'username'} ) {
 +1013       my ( $uuid, $ugid ) = ( getpwnam( $opt{'username'} ) )[ 2, 3 ];
 +1014       if ( !defined $uuid || $uuid == 0 ) {
 +1015         die "spamd: cannot run as nonexistent user or root with -u option\n";
 +1016       }
 +1017
```

```
+1018       if ( $opt{'groupname'} ) {
+1019         $ugid = getgrnam( $opt{'groupname'} ) || $ugid;
+1020       }
+1021
+1022       # bug 5518: assignments to $) and $( don't always work on all platforms
+1023       # bug 3900: assignments to $> and $< problems with BSD perl bug
+1024       # use the POSIX functions to hide the platform specific workarounds
+1025       POSIX::setgid($ugid);  # set effective and real gid
+1026       POSIX::setuid($uuid);  # set effective and real UID
+1027       $< = $uuid; $> = $uuid;   # bug 5574
+1028
+1029       # keep the sanity check to catch problems like bug 3900 just in case
+1030       if ( $> != $uuid and $> != ( $uuid - 2**32 ) ) {
+1031         die "spamd: setuid to uid $uuid failed (> = $>, < = $<)\n";
+1032       }
+1033     }
```

POSIX::setuid() should just call setuid() from the kernel. And maybe this is the problem. From a README file there is the following text:

```
 There is an issue if you run spamd using the standard perl installation
on Mac OS X and certain *BSD-flavored UNIX platforms.  spamd will change
effective uid to the user calling spamd for security reasons.  Before
calling out to any external programs (DCC and Pyzor, as of 3.0.0,) spamd
will fork() and change the real uid to the same as the effective uid.
Unfortunately, the default perl in at least Mac OS X, does not allow perl
programs to change the real uid so for security reasons the spamd child
will die.  To fix this issue, either disable the DCC and Pyzor rules,
or install a different version of perl which supports setuid() calls.

The default perl binary in FreeBSD had a similar issue when attempting
to change the real uid.  This has been worked around, but there could
be an issue such as the one in Mac OS X that we have not yet heard about.
```

I read the above text to mean that they expect POSIX:setuid to actually do a setruid() rather than a seteuid(). Several years ago - afaik - POSIX decided that setruid() was to be removed, and only seteuid() would be used. Permissions are based on your seteuid() value while your RUID (Real User ID) could be used for accountability and/or audits.

Next step: figure out how to debug spamassassin as it tests itself....