

# ports system woes

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/hackers/2008-03/msg00358.html>

---

- *From:* [soralx@xxxxxxxxxx](mailto:soralx@xxxxxxxxxx)
  - *Date:* Wed, 26 Mar 2008 02:01:31 -0700
- 

Folks,

are there any plans to rewrite the ports/packages system? Maybe someone started work on improving things in this area already?

The thought that pkg\_\* tools and Mk/\* scripts might be somewhat inefficient had crossed my mind before, when at last modular Xorg exposed all the inefficiencies in these tools. Basically, pkg\_\* and portupgrade seem to use very inefficient algorithms all over the place; they've become nearly useless on my system these days.

Consider pkg\_delete, for example:

```
`time pkg_delete /var/db/pkg/rxvt-unicode-9.02/  
real 7m4.207s  
user 0m4.083s  
sys 0m16.348s
```

This one was rather benign. Many of bigger packages take 15 minutes to get removed. Now, all the test were conducted on Pentium 4-M 1.2GHz notebook with 256M RAM and MHT2040AH hard drive (5400RPM, 8M buffer). Notice the amount of RAM. Also:

```
[root@freen0de /var/db/pkg]# ll|wc -l  
959
```

For the rxvt case, the offending function is matchbyorigin() inside src/usr.sbin/pkg\_install/lib/match.c, called from delete/perform.c function pkg\_do(). Here's a snippet from pkg\_do():

```
for (p = Plist.head; p ; p = p->next) {  
    if (p->type != PLIST_PKGDEP)  
        continue;  
    deporigin = \  
(p->next != NULL && p->next->type == PLIST_DEPORIGIN) ?p->next->name :  
    NULL;  
    if (Verbose) {  
        printf("Trying to remove dependency on package '%s'", p->name);  
        if (deporigin != NULL)
```

## ports system woes

```
printf(" with '%s' origin", deporigin);
printf("\n");
}
if (!Fake) {
depnames = (deporigin != NULL) ? matchbyorigin(deporigin, NULL) :
NULL;
if (depnames == NULL) {
depnames = alloca(sizeof(*depnames) * 2);
depnames[0] = p->name;
depnames[1] = NULL;
}
for (i = 0; depnames[i] != NULL; i++)
undepend(depnames[i], pkg);
}
}
```

What exactly "removing dependency on package" means? (i.e., what is `undepend()` for?) I didn't figure it out yet (anyone?), but from what I read from the code, it calls `matchbyorigin(deporigin, NULL)` -- which is the super-slow part -- while processing each `@pkgdep` one-by-one. `matchbyorigin()`, in turn, scans `+CONTENTS` of each entry in `db/pkg/*` to get `'@comment ORIGIN:'` value. As a result, each operation (like "Trying to remove dependency on package 'xineramaproto-1.1.2' with 'x11/xineramaproto' origin.") takes 3-4 seconds (first one ~30s), and there are ~120(!) dependencies for `urxvt` (I imagine them evil penguins are all too glad to make everything complete chaos, but hopefully making things `_so_` modular has at least some benefits).

Replacing `'if (!Fake)'` with `'if (FALSE)'` makes `pkg_delete` go `_really_` fast -- almost instantaneous. I didn't notice any side effects of that hack so far, but there sure are some. Anyway, I understand the desire to move functions like `matchbyorigin()` to `libinstall`, but can't that one at least be made to accept an array of strings (package names) and process them in a single pass? The whole of `src/usr.sbin/pkg_install` needs rewriting, IMO, as this is just one of many examples of slow code.

Here's another example. Checking if a package is already installed was also quite slow. Much of the slowness, turns out, was caused by the following in `ports/Mk/bsd.openssl.mk`:

```
.if !defined(OPENSSSL_PORT) && \
exists(${LOCALBASE}/lib/libcrypto.so)
# find installed port and use it for dependency
PKG_DBDIR?= ${DESTDIR}/var/db/pkg
OPENSSSL_INSTALLED!= grep -l -r "^lib/libssl.so." "${PKG_DBDIR}" | \
while read contents; do \
sslprefix=`grep "^@cwd " "${contents}" | ${HEAD} -n 1`; \
if test "${sslprefix}" = "@cwd ${LOCALBASE}"; then \
echo "${contents}"; break; fi; done
OPENSSSL_PORT!= grep "^@comment ORIGIN:" "${OPENSSSL_INSTALLED}" | ${CUT} -d : -f 2
OPENSSSL_SHLIBFILE!= grep "^lib/libssl.so." "${OPENSSSL_INSTALLED}"
```

## ports system woes

```
OPENSSSL_SHLIBVER?= ${OPENSSSL_SHLIBFILE:E}  
.endif  
OPENSSSL_PORT?= security/openssl
```

Simply defining 'OPENSSSL\_PORT=security/openssl' in make.conf helped a lot (although really, we need to figure a better method of finding installed openssl port -- perhaps a script that updates OPENSSSL\_PORT in make.conf after port installs?). It is still fairly slow, though.

Another example:

```
[root@freen0de /usr/ports/x11/nvidia-driver-96xx]# make fetch  
====> Vulnerability check disabled, database not found  
====> Found saved configuration for nvidia-driver-96.43.01  
=> NVIDIA-FreeBSD-x86-96.43.05.tar.gz doesn't seem to exist in /usr/ports/distfiles/.  
=> Attempting to fetch from http://jp.download.nvidia.com/freebsd/96.43.05/.  
NVIDIA-FreeBSD-x86-96.43.05.tar.gz 100% of 9444 kB 71 kBps  
[root@freen0de /usr/ports/x11/nvidia-driver-96xx]# time make fetch  
====> Vulnerability check disabled, database not found  
====> Found saved configuration for nvidia-driver-96.43.01
```

```
real 0m53.340s  
user 0m0.299s  
sys 0m0.886s  
[root@freen0de /usr/ports/x11/nvidia-driver-96xx]# time make maintainer  
<deleted>
```

```
real 0m22.817s  
user 0m0.265s  
sys 0m0.685s  
[root@freen0de /usr/ports/x11/nvidia-driver-96xx]#
```

So all these examples are only tip of the iceberg that is currently making pkg\_\* & ports sink, but of course no other bugs are as serious as the one with pkg\_delete (or so I hope).

And I'm not even talking about portupgrade. Dragging it's feet at a snail pace, running sluggish 'pkgdb -Q' each tiny step... it's a rather useful tool. But, I believe portupgrade is not really needed -- if FreeBSD's base ports/packaging tools get fixed and extended, they'd do the job (yah, there's portmaster, but with broken pkg\_\*, it's just as good as portupgrade).

BTW, to make `portupgrade -a` at all useful, a while ago I whipped up a bash command that lists all dependencies of installed packages:

```
export PORTSBASE=/usr/ports; cd /var/db/pkg  
for dir in `ls`; do tmp=`grep '@comment ORIGIN' "${dir}/+CONTENTS"|awk -F ":" '{ print $2 }'; echo  
${tmp}; cd ${PORTSBASE}/${tmp} && (make package-depends-list|awk -F " " '{ print $3 }'); cd  
/var/db/pkg/; done|sort|uniq > /tmp/deps.lst
```

Allow it 3-5 hours per 1000 installed ports 8-) The good thing is that

## ports system woes

theoretically this should get all the dependencies in fresh ports tree right, when fed from old (not updated) /var/db/pkg, so then you could do:

```
for dir in `cat /tmp/deps.lst`; do cd ${PORTSBASE}/${dir} && make config-conditional; done
```

which in ~10 minutes should take care care of all the new unconfigured (`make config`) ports in single attempt. Practically, there are always some problems with the gazillion versions of `tcl*` and `tk*` (go to ports tree and manually run `make config` for each and every version of `tcl*` and `tk*` that's there), and any cases where 'dialog' coredumps (long lists of options). `/tmp/deps.lst` can be usually reused. Now, just to convert these commands to normal `/bin/sh` script...

[SorAlx] ridin' VN1500-B2

---

freebsd-hackers@xxxxxxxxxxx mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-hackers>

To unsubscribe, send any mail to "freebsd-hackers-unsubscribe@xxxxxxxxxxx"