Building -stable Packages on OpenBSD

A Primer
(for the stubborn)

SEMIBUG

December 15, 2005

Josh Grosse
josh@jggimi.homeip.net
Explaining **-stable**

- 3 flavors of OpenBSD
  - -release
  - -current
  - -stable
Explaining Packages

- Third party applications
- Pre-compiled, ready to be installed
- Includes dependency requirements
  - Libraries
  - Called applications
- Administered with pkg_* tools suite
Flavors and Packages

- Three flavors of third party applications
  - -release is frozen!
    - CVE remediations (may) go to -stable
    - Other application updates go to -current
  - The Project does not (currently) build -stable packages
M:Tier provides -stable packages

- This is a public -stable service:
  https://stable.mtier.org

- *M:Tier* employs OpenBSD Project developers
  - Offers -stable systems and -stable packages.
  - Cryptographically signed binaries
  - Reputable

*But stubborn sysadmins can also build these packages.*
Why build your own?

- Using an architecture M:Tier doesn't provide?
- Trust issues? Curiosity?
- Governance requirements?
- Independent streak?
- Knowledge acquisition?

“I've always built my own -stable packages.”
Packages are built from Ports

- Applications are ported to OpenBSD
- The Ports Tree contains thousands of ports
- Numbers available vary by architecture
What's in an OpenBSD Port?

- Scaffolding
  - Fetch & Build instructions
  - OS-specific patches (if needed)
  - Checksums
  - Packing list
  - Description

The purpose of a port is to produce consistent binary packages from the upstream source.
Caution

- Port building is easy … except when problems occur.
- Port building is also resource intensive.
  - In particular, of CPUs, storage, and time.
- Using M:Tier's service may be an appropriate solution.
Isn't this already in the FAQ?

- The OpenBSD FAQ has lots of guidance. For example, how to build -stable systems.
- The OpenBSD FAQ doesn't cover this particular process – bulk building of -stable packages.
Building -stable systems

- Follow FAQ 5
  - The release(8) man page is the definitive doc.
  - Obtain -stable source, build -stable system.
  - For multiple deployments, build -stable release.
  - Upgrade systems with your -stable release.
  - Multiple architectures? Build the next -stable system.
  - Lather, rinse, repeat.

Then celebrate!
Onward to package building!

But first….a little level setting…
Considerations

- Mixing and matching OpenBSD branches
  - -release / -stable vs. -current

- Multiple architectures?

- Depending on dependencies
  - Run dependencies
  - Build dependencies
    - *You may be building many packages*

- Manual vs. Automatic packages \(^1\)
  - pkg_info -m
  - pkg_delete -a
  - pkg_add -a and -aa

\(^1\) See the *pkg_info(1)*, *pkg_delete(1)*, and *pkg_add(1)* man pages. D'oh!
Updating the ports tree

- OpenBSD uses CVS.
- Yeah, CVS. Deal with it.

$ cd /usr/ports
$ cvs -q up -Pd

- The -q will show you only what has changed.
CVS reports can be a little cryptic

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P file</td>
<td>Patched</td>
</tr>
<tr>
<td>U file</td>
<td>Updated (replaced)</td>
</tr>
<tr>
<td>M file</td>
<td>Locally modified. CVS will leave as-is. You can clear local mods with -C: $ cvs -q up -PdC.</td>
</tr>
<tr>
<td>C file</td>
<td>Conflict. CVS will leave as-is. You must resolve manually.</td>
</tr>
</tbody>
</table>
Preparation steps on each production system

- Delete any unneeded automatic dependencies
  
  # pkg_delete -a

- Run out-of-date(8)
  
  $ export PATH=$PATH:/usr/ports/infrastructure/bin
  
  $ out-of-date | tee my.report

- Concatenate reports from multiple systems (of the same architecture)
dpb(1) – Distributed Ports Builder

• Builds locally or across a server farm
• Start as root. dpb(1) will drop privilege for:
  • Fetch
  • Build

  Read the dpb(1) man page section on the Security Model

• Use -R option to build all needed dependencies
  $ export PATH=$PATH:/usr/ports/infrastructure/bin
  
  # dpb -R my.report

• The dpb(1) man page is required reading.
  • Read it again!
  • Have the man page handy when you run dpb(1).
  • Consider # pkg_delete -X for clean builds!
Typical pkg.conf(5) on a -stable build machine, selecting -stable packages if available, -release packages otherwise:

    installpath = /usr/ports/packages/%a/all
    installpath += http://<project.mirror>/pub/OpenBSD/%c/packages/%a/

• Delete unneeded build dependencies
  
    # pkg_delete -a

• Update local packages
  
    # pkg_add -u
Deployment to -stable systems

- Place packages on a local web server.
- Typical pkg.conf(5) on a -stable system, selecting -stable packages if available, -release packages otherwise:

  installpath = http://<.my.web>/path/to/local/packages/
  installpath += http://<project.mirror>/pub/OpenBSD/%c/packages/%a/

- On each system, # pkg_add -u
Questions?
No questions? Great!