Faster and more secure packages in OpenBSD

Marc Espie <espie@openbsd.org>

October 4, 2015

Foreword: OpenBSD packages

Foreword: OpenBSD packages

specific to OpenBSD

We're *not* talking `pkg_add` from NetBSD
Nor `pkgng` from FreeBSD

Foreword: OpenBSD packages

specific to OpenBSD

We're *not* talking `pkg_add` from NetBSD
Nor `pkgng` from FreeBSD

- Everything in perl
- We've had functional updates since 2006
- Really fast considering it's an interpreted language

OpenBSD packages (2)

Standardized tools

```
#! /usr/bin/perl
use OpenBSD::PackingList;

package OpenBSD::PackingElement;
sub walk
{
# do nothing
}

package OpenBSD::PackingElement::Sample;
sub walk
{
my $self = shift;
print $self->fullname, "\n";
}

package main;

my $p = OpenBSD::PackingList->from_installation("pkgname");

$p->walk;
```

OpenBSD packages (3)

Look ma! no cache

- On-the-fly install
- Package names are standardized
- We download only what's needed for updates (e.g., the packing-list)

OpenBSD packages (3)

Look ma! no cache

- On-the-fly install
- Package names are standardized
- We download only what's needed for updates (e.g., the packing-list)

Check first, install later

- First we check the packing-list and the file-system
- ...If everything is A-ok, we install/update
- NO rollback system

Traces of a signing protocol from last century

Traces of a signing protocol from last century

Gzip

Gzip's header can be packed with shit.

- for instance, an X509 signature
- ... which leads to gzsig

Traces of a signing protocol from last century

Gzip

Gzip's header can be packed with shit.

- for instance, an X509 signature
- ... which leads to gzsig

Drawbacks

- You have to download the full package first
- ... if you install on the fly, that's unacceptable

Bringing signatures up to 2015

In the beginning

- It came from a 3rd party, 4 years ago (actually several people came to me at almost the same time)
- leads to a non-specialized mechanism $\rightarrow$ simpler
- X509-certificates

Bringing signatures up to 2015

In the beginning

- It came from a 3rd party, 4 years ago (actually several people came to me at almost the same time)
- leads to a non-specialized mechanism $\rightarrow$ simpler
- X509-certificates

Just-In-Time

- Let's just sign the packing-list
- It contains sha256 hashes for basically everything
- The signature gets checked during the update validation process
- ...then each individual file gets checked right after extraction

Bringing signatures up to 2015 (2)

What about meta-data

- Let's bring from the tar header
- ... everything that's really important into the PLIST
- ... and be paranoid about it
- (linking to building without root)

X509 pain

- OpenSSL oddities (mime-encoding)
- Complicated
- Rocket science (PKI)

X509 pain

- OpenSSL oddities (mime-encoding)
- Complicated
- Rocket science (PKI)

Enter Signify

Post-Modern cryptography

- Written by Ted Unangst
- elliptical curves
- signs sha512 hashes
- no PKI support at all

Enter Signify

Example

```
untrusted comment: signature from openbsd 5.6 packages private key
RWR0EANmo9nqhtL3waUA0Buq/b2QHW06S0rufjAwgztC0U5P6+7kh+YnyetC6jiaV57WURH9n0EvoiMbzPmbR+qxIeUf6jWfBgg=
SHA256 (INSTALL.amd64) = 84b7e7cb7e5bc44a85dd60c1f6c1730900cc833f66a209e32b3d21132f637308
SHA256 (base56.tgz) = 0db2b0336007cac50f289a5d4f71cb4cbed085cf8656e27deef390b758138a0d
SHA256 (bsd) = 03b95b2e4f00421aab0c74ae3d6a2ef90992c765022a14af989e4d74b6d360ac
SHA256 (bsd.mp) = fae0d3b4a7ef6dc0d840996a3c0820682f63d516af70dfea3d28b0d7788eff0b
SHA256 (bsd.rd) = fc440856dcb0c0f09f363ddf7db6b93e7555eb0630b1136d732d2100108e38bb
SHA256 (cd56.iso) = d203cd7774d6f09f4555f7f23b74ce969e96162c23baeed2fafc5fcd35575a75
SHA256 (cdboot) = fd65d49a4765bb9c83ccbb77e9a99385aa7c4cd0cc7636db9327ca102d8106b3
SHA256 (cdbr) = c5244fd55f85263035feb411f6e7fc17614b160116c878e850d84b17f67f2951
SHA256 (comp56.tgz) = e30dacce8a067ead6cedb0dff63f0a605e9f2065d404cc78608a584e99190009
SHA256 (etc56.tgz) = 22b9cc137df79a7ec0c910d80f1b98b561adf33ff589ab4625f48090b76fef74
SHA256 (floppy56.fs) = 8c0a5cc9836d09c5d0a357294b61f58f076ea632afebd319aa143de56244732d
SHA256 (game56.tgz) = 4694c173836a095cc7bbea751995934cbff301aff64ea2a075d1ff7f2b8f1abf
SHA256 (install56.fs) = 9407ba385a5fc19587f77e1e4debbd0a54538c6ea86d3861ac6d8c78ccd67917
SHA256 (install56.iso) = 4fc8acfbb27315a96ad030270a0500342dc8a3f54866ea97c088ae8ac987ce9f
SHA256 (man56.tgz) = 46492a6b6751e1e400f064a8fdadfa68fa5090f54962acade2e4537d7494f8e7
SHA256 (miniroot56.fs) = 0bf0b54018cb1f96fe21f20e5f761c4051512627c77cc650af17d3600e0c945d
```

All is in the process

Transparency

- One key, one usage pattern
- After the build signing, disconnected from about everywhere
- No revocation process

What about packages

Specificities

- No global sha256
- Again, signing is done after the bulk building process, on a seperate box

What about packages

Specificities

- No global sha256
- Again, signing is done after the bulk building process, on a seperate box

Issues

- Signing = gunzip/sign/repack
- waaaays too slow
- chunked gzip

Properties

Perl

- `$plist->write_no_sig($fh);`
- Everything gets signed (the timestamp as well)
- this includes the key particulars (so we can give meaningful error messages)
- No need to recompute sha256

Properties

Perl

- `$plist->write_no_sig($fh);`
- Everything gets signed (the timestamp as well)
- this includes the key particulars (so we can give meaningful error messages)
- No need to recompute sha256

Consistency

- One package always gets updates (quirks)
- Display signing date
- Contains a list of "risky packages"

Quirks?

- Okay I lied that's the database
- Bunch of perl methods to
  - figure out package name changes
  - remove packages that migrate to base
  - deprecate very old shit
  - fix weird stuff on the spot

Wibbly wobbly timey wimey... stuff

Look I know what I'm talking about, I'm a doctor

Basically, that's my "can't predict the future" "get out of jail for free" mechanism.

Insecure packages

Known issue

- there is this paper about package security
- that's always a problem with "inconsistent collection"

Insecure packages

Known issue

- there is this paper about package security
- that's always a problem with "inconsistent collection"

Two part solution

- "deprecate" packages in the quirks package
- timestamps are signed and verified, but we don't know what this means: let the user decide
- this has *nothing to do with certificate revocation*.

Sidetrack

Tar balls

- You could actually do the same thing with tar
- just build a sha256 list of every chunk (file) in the tarball
- create a MANIFEST file that you sign and prepend
- use the same chunk gzip technique

Straightforward minimal changes

- Have tar create the MANIFEST/check the MANIFEST while it goes
- add an option to build an "unfinished" tarball (no chunk of 0)
- You can put pieces together as a shell script to sign
- Just fork out signify while checking
- ...not done because no practical interest and ETOOLARGETODOLIST anyway

Bonus!

Reordered packages

- Files that change migrate at the start of packages
- LRU
- Requires very little extra info! (history for sha256)

Bonus!

Reordered packages

- Files that change migrate at the start of packages
- LRU
- Requires very little extra info! (history for sha256)

Fan-out

- 40–50 Go
- Rsyncable gzip $\rightarrow$ fragile
- low-tech solution
- reset timestamps in the tar header, move them to plist
- chunked gzip "from end"
- importance of reproducible builds, avoid random gratuitous variations

Questions !!!

Marc Espie <espie@openbsd.org>