Sponsored by Red Hat

Secure your Delivery Chain

Reproducible Software Builds with Pulp

Matthias Dellweg

Software Engineer

5. Feb. 2024

pulpproject.org

pulp

Security

There are two threats against security in software delivery:

- Malicious code injection.
- Uncertainty.

The solution is simply:

- You need control what is built.
- You need the ability to rebuild your software reliably.

Agenda

Own Your Content

Package Ecosystems vs. Curated Distributions

Approaches to Solutions

How can Pulp help?

Sponsored by Red Hat

Package Ecosystems vs. Curated Distributions

pulp

Package Ecosystems

You do not write a new software project from first principles.
Modern programming languages are accompanied by a wealth of libraries.
These libraries are usually exchanged via an unrestricted package shop.
They are installed via a specific package manager.

Package Ecosystems

- pip : PyPi → Python
- bundler : RubyGems → Ruby
- mvn : MavenCentral → Maven
- yarn : npm → ECMAScript (JS)
- podman : quay → OCI images
- curl : http → files

Unrestricted

Unrestricted access imposes a threat.
Whoever first pushes a library owns the namespace.
No guarantees two packages can be installed simultaneously.
Every change to the upstream library is a risk.

- Release a new version
- Remove an old version
- Replace a version

Curated Distributions

Curated software distributions (Debian, RHEL, CentOS, Fedora, ...) on the other hand:

- Are published by a single entity.
- Are self contained.
- Verify that packages can be installed.
- Verify that packages do not break each other.

Common threats

Upstream could be

- unavailable
- out of service
- introduce rate limits
- compromised
- turn evil

Approaches to Solutions

Semantic versioning

Yes, you can specify `Django~=4.2.0`.
Only guards partially against the new version.
Not all libraries use semantic versioning.
No good option if your business depends on reliable, continuous delivery.

Pin to exact versions

Specifying the exact versions for each dependency help reproducibility.
Usually you can create a lockfile.
Still tomorrow something could be missing / changed.

Control the Source

Hosting the content, you can:

- Control what is installed on build nodes.
- Ensure that the libraries are available tomorrow.
- Recreate the same build environment over and over.

How can Pulp help?

How can Pulp help?

Pulp is a toolbox:

- Provides a lot of different workflows you can / need to define.
- Handles multiple package formats in a natural way.
- Acts as a custom server for client package managers.

pulp

Mix and Match

Pulp allows, depending on the type of content, to:

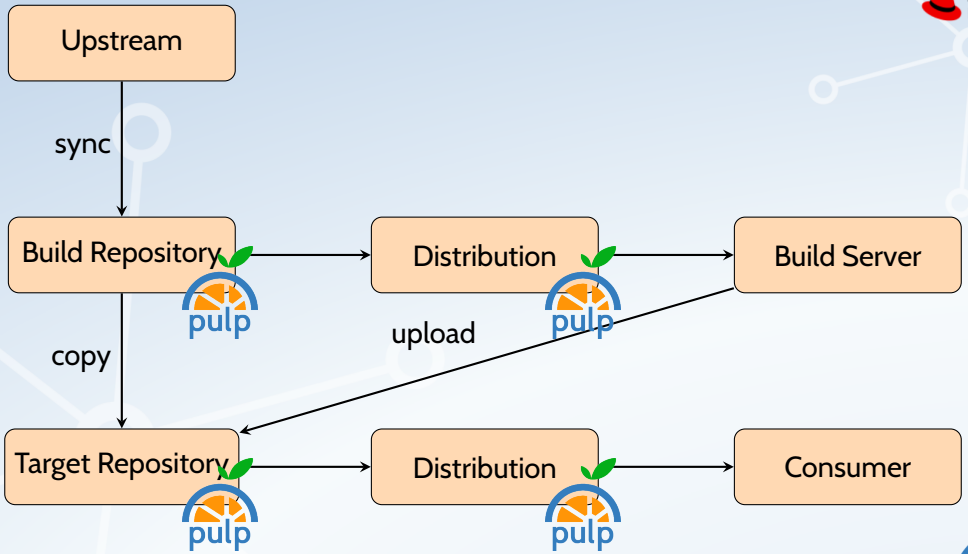
- Sync content from upstream.
  - immediate
  - on-demand
- Upload individual packages.
- Copy packages from one repository to another.
- Setup a pull-through cache.

Repository Versions

In Pulp a repository is always versioned.
Each version is an immutable snapshot.
You can keep a small history.
Today's build failed, how did we build it yesterday?

dev-test-prod Lifecycle

Distributions are relative paths on the content server.
Configured in your clients.
Hooked up to repository versions.
Lifecycles can span package types.
(e.g. matching python packages and container base images.)



Conclusion

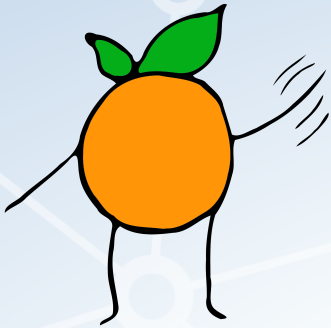
You need to control what you do not trust.

You need to trust what you do not control.

You need to assess the risk and find a balance.

Pulp can assist you to control the build environment.

Sponsored by Red Hat



THANK YOU!

pulpproject.org

pulp

Contact us

Find information about Pulp:

- `https://pulpproject.org`

Find the Pulp team at:

- `https://discourse.pulpproject.org`
- `#pulp:matrix.org`
- `#pulp-dev:matrix.org`
- PulpEvent @ CfgMgmtCamp