Des Paquets pour Why-3

Loïc Correnson
CEA-LIST, Laboratoire de Sûreté Logicielle

Proof In Use
21 novembre 2022

Le Contexte

Gros Projets en Why-3
Paquets OCaml extraits depuis Why-3

Des Paquets pour Why-3 : quelques challenges…

- Installer & Retrouver des librairies Why-3 ?
- Co-développement de librairies Why-3 ?
- Accéder à la documentation ?
- Quelles preuves ?
- Quelles hypothèses ?
- Utiliser le code extrait depuis OCaml ?

Quelques expérimentations…

Expérimentations

- ✗ MakeWhy3 : limité, peu pratique
- ✗ ocamlfind : uniquement installation, *hackish*
- ✓ why3find + dune : une bonne solution

Why3find : un wrapper autour de Why3

- ★ why3find init
- ★ why3find config
- ★ why3find install, uninstall
- ★ why3find query, where
- ★ why3find compile, ide, replay
- ★ why3find prove, server, worker
- ★ why3find doc
- ★ why3find extract
- ★ why3find.ppx

why3find init

Initialise un répertoire pour le paquet

- ✓ configuration du paquet Why3 : why3find.json
- ✓ configuration du projet dune : dune-project
- ✓ suivi de version : .gitignore

why3find config

Configure le paquet Why3

- ✓ Calibrage des prouveurs (Cf. why3find prove)
- ✓ Transformations (Cf. why3find prove)
- ✓ Dépendances Why3
- ✓ Dépendances OCaml, drivers Why3 d'extraction

- ✓ également nbr. max. de prouveurs (via ~/.why3conf)

why3find query, list, where

Recherche de paquets Why3

- ✓ Site(s) d'installation(s) : fourni par dune-site
- ✓ Meta-data des paquets
- ✓ Recherche récursive par dépendance

why3find install, uninstall, query, list, where

Installation et recherche de paquets Why3

- ✓ Site(s) d'installation(s) : fourni par dune-site
- ✓ Meta-data des paquets
- ✓ Recherche récursive suivant les dépendances
- ✓ Installation des sources, des témoins de preuves, de la documentation, du code OCaml extrait, etc.
- ✓ Génère un fichier dune pour l'installation
- ✓ Au final : un paquet Why3 s'installe comme un paquet OPAM

why3find compile, ide, replay

Wrappers simples autour de Why3

- ✓ sur des fichiers ou des répertoires
- ✓ utilise les informations du paquet (-L <depends>)
- ✓ why3find compile utilise why3 prove —type-only
- ✓ why3find ide lance why3 ide avec une stratégie de preuve définie qui ressemble à why3find prove

why3find prove

Une stratégie de preuve automatique pour Why3

- ✓ sur des fichiers ou des répertoires
- ✓ utilise un cache local pour les résultats des prouveurs
- ✓ stabilise les temps et les timeout entre machines différentes par calibrage
- ✓ combine prouveurs et transformations (sans paramètres) pour chercher efficacement et automatiquement des preuves
  (Cf. travaux de Benjamin Jorge)
- ✓ stocke un *témoin de preuve* pour documentation et pour faciliter le rejeu et la mise au point incrémentale des preuves
- ✓ génère une session Why3 pour exploration dans why3 ide / why3find ide
- ✓ peut ouvrir automatiquement why3 ide en cas d'échec

why3find prove —hyps

Calcul des hypothèses (axiomes)

- ✓ axiomes
- ✓ types abstraits, paramètres logiques (fonctions, prédicats)
- ✓ valeurs abstraites (contrats)
- ✓ suit les dépendances (use foo.bar)
- ✓ filtre les instances (clone foo.bar with …)
- ✓ utilise les drivers d'extraction OCaml pour raffiner le calcul (paramètre vs. symbole externe)
- ★ extension possible : utiliser les drivers de réalisation Coq

why3find config -m

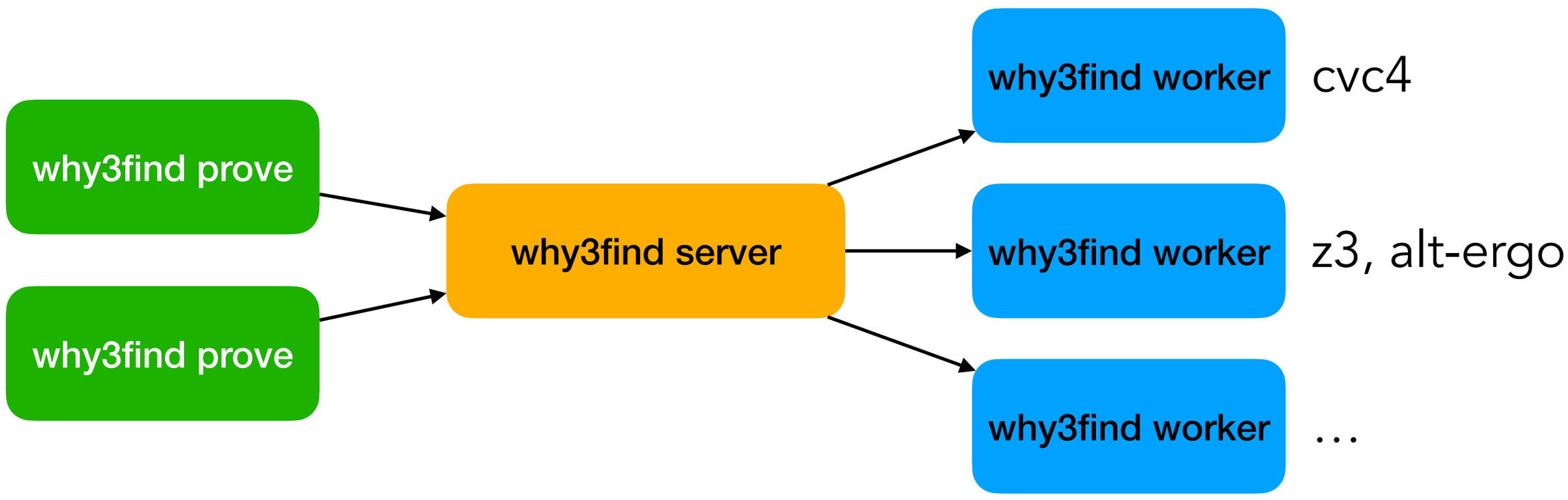
Calibration des prouveurs

- ✓ utile pour rejouer des preuves avec des timeouts adaptés pour différentes machines
- ✓ nécessite une ou des familles de problèmes de références pour les prouveurs afin de comparer la vélocité des machines
- ✓ la limite de steps pourrait suffire pour le re-jeu, mais n'est pas franchement utilisable pour l'exploration des preuves, on a besoin d'une limite de temps

why3find server, worker

Serveur de preuves

- ✓ utilise aussi la notion de calibrage pour normaliser les résultats
- ✓ nécessite un petit hack de Why3 (prove_prepared_buffer)
- ✓ supporte la connection dynamique à chaud de clients / de workers
- ✓ fait le dispatch, gère un cache global



why3find doc

Générateur de Documentation

- ✓ refonte complète de why3 doc
- ✓ supporte (un peu) plus de markdown
- ✓ pages de documentation utilisateur (eg. readme.md)
- ✓ marquage de fragments de code avec fold/unfold
  eg. (*proof*) … (*qed*)
- ✓ liens aux symboles why3 inter-paquets
- ✓ incorpore les *témoins de preuves*
- ✓ incorpore les *hypothèses et axiomes*
- ✓ pour les clones, documente les symboles générés

why3find extract / why3find.ppx

Extraction de code OCaml

- ✓ wrapper autour de why3 extract —modular
- ✓ génère un répertoire avec un fichier dune pour compiler et installer le code extrait

PPX pour Why3

- ✓ hack de why3 extract pour générer des tables de symboles
- ✓ permet à du code client OCaml d'invoquer du code extrait d'un paquet Why3

```
✓ [%%why3use "foo.bar.Jazz as M" ] (* import *)
✓ [%why3: t M.ty]                  (* Why3 type *)
✓ [%why3 M.fn]                     (* Why3 value *)
✓ [%why3 e.M.fd]                   (* Why3 field *)
✓ [%why3 M.C(e,…)]                 (* Why3 constructor *)
✓ [%why3? M.C(p,…)]                (* Why3 pattern *)
```

A tester, à discuter !