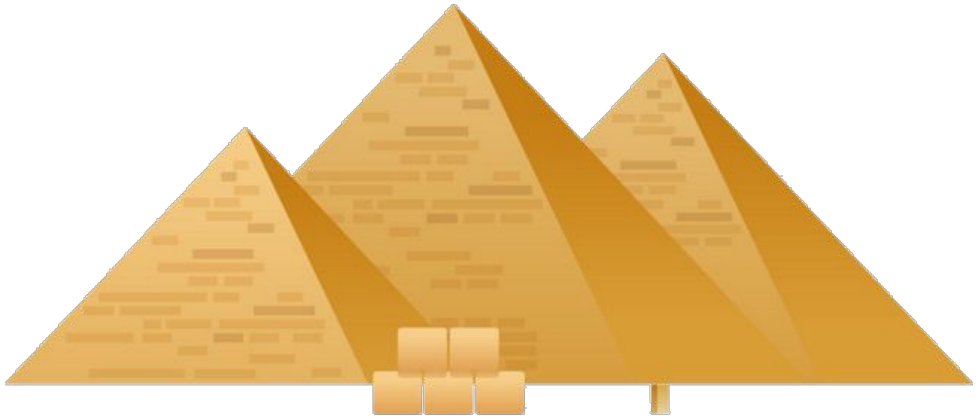


Thoth: healing Python applications

September 3rd, DevConf.US 2021

Fridolin Pokorny <fridolin@redhat.com>

Red Hat

$ whoami && whoweare

- Fridolin "*fridex*" Pokorny
  - Twitter: @fridex
  - Thoth team member since 2019
  - I like Python & road cycling

- Thoth
  - Started as a research project AICoE team, Office of the CTO
  - One of the main offerings: cloud based Python resolver
  - https://thoth-station.ninja

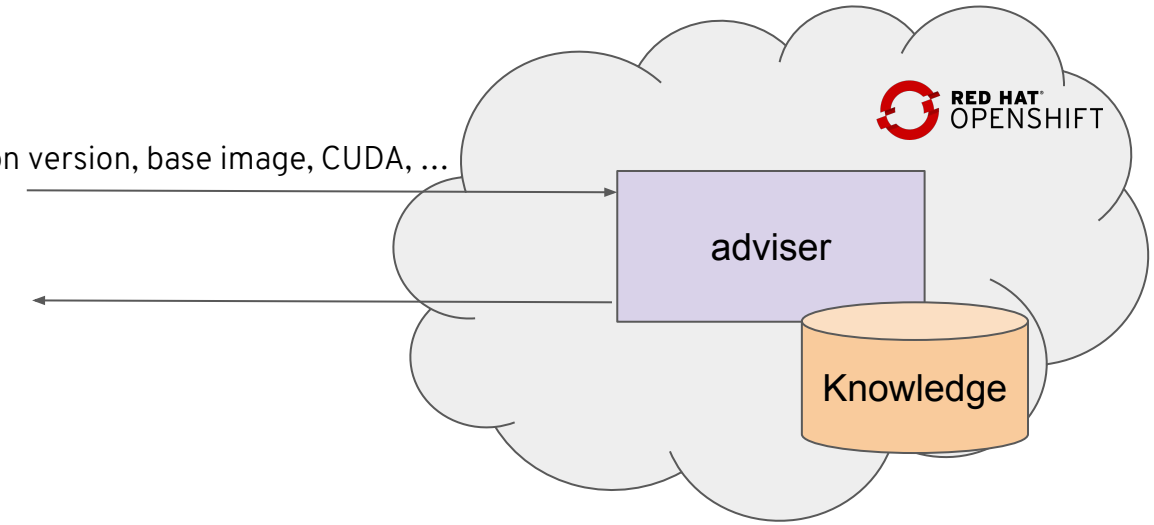
The Python resolver run in cloud

- Recommendation engine for Python applications
- Publicly available to the community
- Stochastic resolver implementing gradient-free reinforcement learning methods
  - Temporal difference learning
- See documentation for more information:
  - https://thoth-station.ninja/docs/developers/adviser

Python cloud resolver

Requirements
Constraints
Runtime environment information - OS, Python version, base image, CUDA, ...
Static source-code analysis
Recommendation type

Lockfile + justification




```
$ pip install thamos
$ thamos config
$ thamos advise
```


Red Hat

Why gradient-free reinforcement learning?

- Wide range (*infinite?*) of possible resolutions depending on requirements used in the application and other inputs to the resolver
- A model is trained on each request to the resolver
- Exploration phase and subsequent exploitation phase comes with the resolved software stack

Resolution pipeline

- The resolution process is using a pipeline made out of units of different type
  - Base pipeline types: boots, pseudonyms, sieves, steps, strides, wraps
- Pipeline units can be implemented directly in Python or declarativelly in YAML files
- The resolution pipeline is constructed dynamically based on inputs to the resolution engine

Declarative interface for the resolver to resolve Python packages following prescribed rules

Prescriptions - declarative interface to the cloud based resolver

- Provide a way to declaratively state how the resolution process should look like
- A set of YAML files that are automatically consumed by resolver in a deployment
- See documentation for more information:
  - https://thoth-station.ninja/docs/developers/adviser/prescription.html

Prescriptions - Example 1

- Pillow in version 8.3.0 does not work with NumPy

  https://github.com/python-pillow/Pillow/issues/5571

```
with PIL.Image.open(filepath) as img:
      numpy.array(img, dtype=numpy.float32)
```

```
>           frame_paletted = np.array(im, np.uint8)
E           TypeError: __array__() takes 1 positional argument but 2 were given

/lib/python3.9/site-packages/imageio/plugins/pillow.py:745: TypeError
```

```yaml
units:
 steps:
 - name: Pillow830TypeErrorStep
   type: step
   should_include:
     adviser_pipeline: true
   match:
     package_version:
       name: pillow
       version: ==8.3.0
       index_url: https://pypi.org/simple
     state:
       resolved_dependencies:
       - name: numpy
   run:
     not_acceptable: Pillow in version 8.3.0 does not work with NumPy
     stack_info:
     - type: WARNING
       message: Pillow in version 8.3.0 does not work with NumPy
       link: https://github.com/python-pillow/Pillow/issues/5571
```

```
units:
steps:
- name: Pillow830TypeErrorStep
  type: step
  should_include:
    adviser_pipeline: true
  match:
    package_version:
      name: pillow
      version: ==8.3.0
      index_url: https://pypi.org/simple
    state:
      resolved_dependencies:
      - name: numpy
  run:
    not_acceptable: Pillow in version 8.3.0 does not work with NumPy
    stack_info:
    - type: WARNING
      message: Pillow in version 8.3.0 does not work with NumPy
      link: https://github.com/python-pillow/Pillow/issues/5571
```

```
units:
 steps:
 - name: Pillow830TypeErrorStep
   type: step
   should_include:
     adviser_pipeline: true
   match:
     package_version:
       name: pillow
       version: ==8.3.0
       index_url: https://pypi.org/simple
     state:
       resolved_dependencies:
       - name: numpy
   run:
     not_acceptable: Pillow in version 8.3.0 does not work with NumPy
     stack_info:
     - type: WARNING
       message: Pillow in version 8.3.0 does not work with NumPy
       link: https://github.com/python-pillow/Pillow/issues/5571
```

```
units:
 steps:
 - name: Pillow830TypeErrorStep
   type: step
   should_include:
     adviser_pipeline: true
   match:
     package_version:
       name: pillow
       version: ==8.3.0
       index_url: https://pypi.org/simple
     state:
       resolved_dependencies:
       - name: numpy
   run:
     not_acceptable: Pillow in version 8.3.0 does not work with NumPy
     stack_info:
     - type: WARNING
       message: Pillow in version 8.3.0 does not work with NumPy
       link: https://github.com/python-pillow/Pillow/issues/5571
```

```
units:
 steps:
 - name: Pillow830TypeErrorStep
   type: step
   should_include:
     adviser_pipeline: true
   match:
    package_version:
      name: pillow
      version: ==8.3.0
      index_url: https://pypi.org/simple
    state:
      resolved_dependencies:
      - name: numpy
   run:
     not_acceptable: Pillow in version 8.3.0 does not work with NumPy
     stack_info:
     - type: WARNING
       message: Pillow in version 8.3.0 does not work with NumPy
       link: https://github.com/python-pillow/Pillow/issues/5571
```

```
units:
 steps:
 - name: Pillow830TypeErrorStep
   type: step
   should_include:
     adviser_pipeline: true
   match:
     package_version:
       name: pillow
       version: ==8.3.0
       index_url: https://pypi.org/simple
     state:
       resolved_dependencies:
       - name: numpy
 run:
   not_acceptable: Pillow in version 8.3.0 does not work with NumPy
   stack_info:
   - type: WARNING
     message: Pillow in version 8.3.0 does not work with NumPy
     link: https://github.com/python-pillow/Pillow/issues/5571
```

Prescriptions - Example 2

Adjust requirements in GPU enabled environments.

- Use tensorflow-gpu as a "*pseudonym*" to tensorflow if GPU enabled environment is available
  - tf_gpu.yaml
- Use the right tensorflow-gpu for the environment following support matrix
  - tf_cuda.yaml
  - tf_cudnn.yaml

Prescriptions - Example 3

Fixing library overpinning issues.

- TensorFlow in version 2.1 can cause runtime errors when running with h5py>=3 caused by overpinning
  - tf_21_h5py.yaml

Prescriptions - Example 4

Use a specific Python package index providing optimized wheel builds.

- Prioritize resolving AICoE builds of TensorFlow for AVX2 enabled environments
  - tf_avx2.yaml
- Use *only* CUDA 11.1 builds of torch available on a PyTorch index:
  - gpu_index.yaml

Prescriptions - Example 5

Block using certain library functions due to security reasons.

- mktemp is deprecated due to vulnerability to race conditions
  - tempfile.yaml

Prescriptions - Example 6

Check configuration of the runtime environment. Pipeline units can also act on base image used.

- A GPU is available but no CUDA is available
  - gpu_no_cuda.yaml

Prescriptions - Example 7

Resolve considering RPM packages available in the runtime environment (similarly ABI or Python packages available).

- GitPython requires Git present in the runtime environment
  - rpm.yaml

References

- Prescriptions for Python open-source projects
  - Feel free to contribute to build better Python ecosystem:
    - https://github.com/thoth-station/prescriptions
- Declarative interface for the resolver to state requirements on Python packages in runtime environment
  - https://thoth-station.ninja/docs/developers/adviser/prescription.html
- Thoth homepage: https://thoth-station.ninja

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

youtube.com/user/RedHatVideos

facebook.com/redhatinc

twitter.com/RedHat

Red Hat