



Thoth

How to recommend the best possible packages for
your application

Thoth Station Inhabitants
2019-Jan-25

Thoth

Who are we?

Christoph “goern” Görn - <https://github.com/goern>

Fridolín “fridex” Pokorný - <https://github.com/fridex>

- Members of Red Hat’s AI CoE
 - <https://github.com/AICoE>
- Thoth Station on GitHub:
 - <https://github.com/thoth-station>




AI Stack challenges

Understanding the problem is the key to successfully solve it.

Software Stacks are complex and only not-changing when running in production:

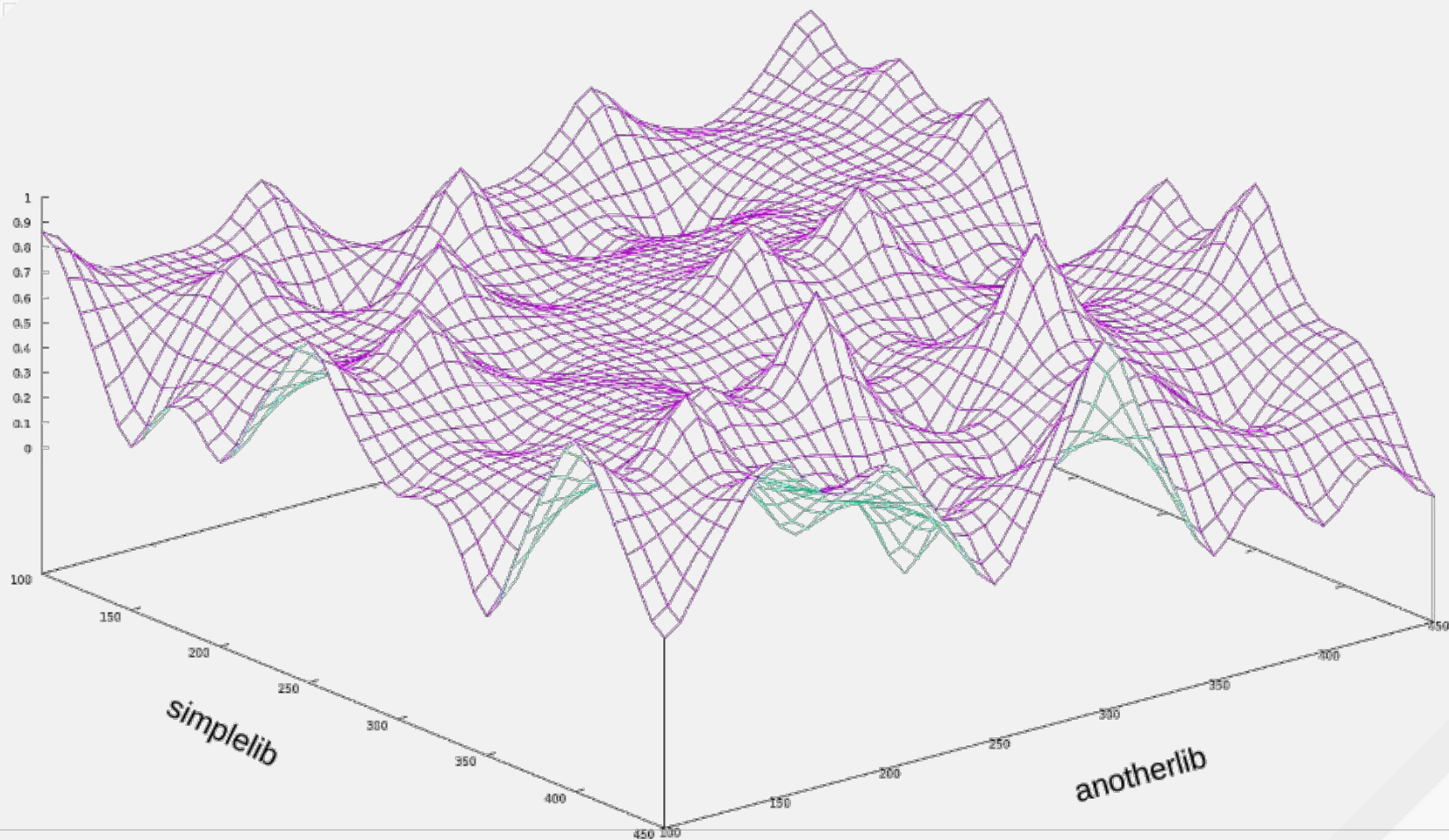
- AI Stacks depend on a lot of components, they keep changing all the time
- Vast amount of platforms available as runtime alternatives
- Off the shelf builds and upstream libraries do not fit corporate needs

How good is my software stack?



```
simplelib  
anotherlib
```

score



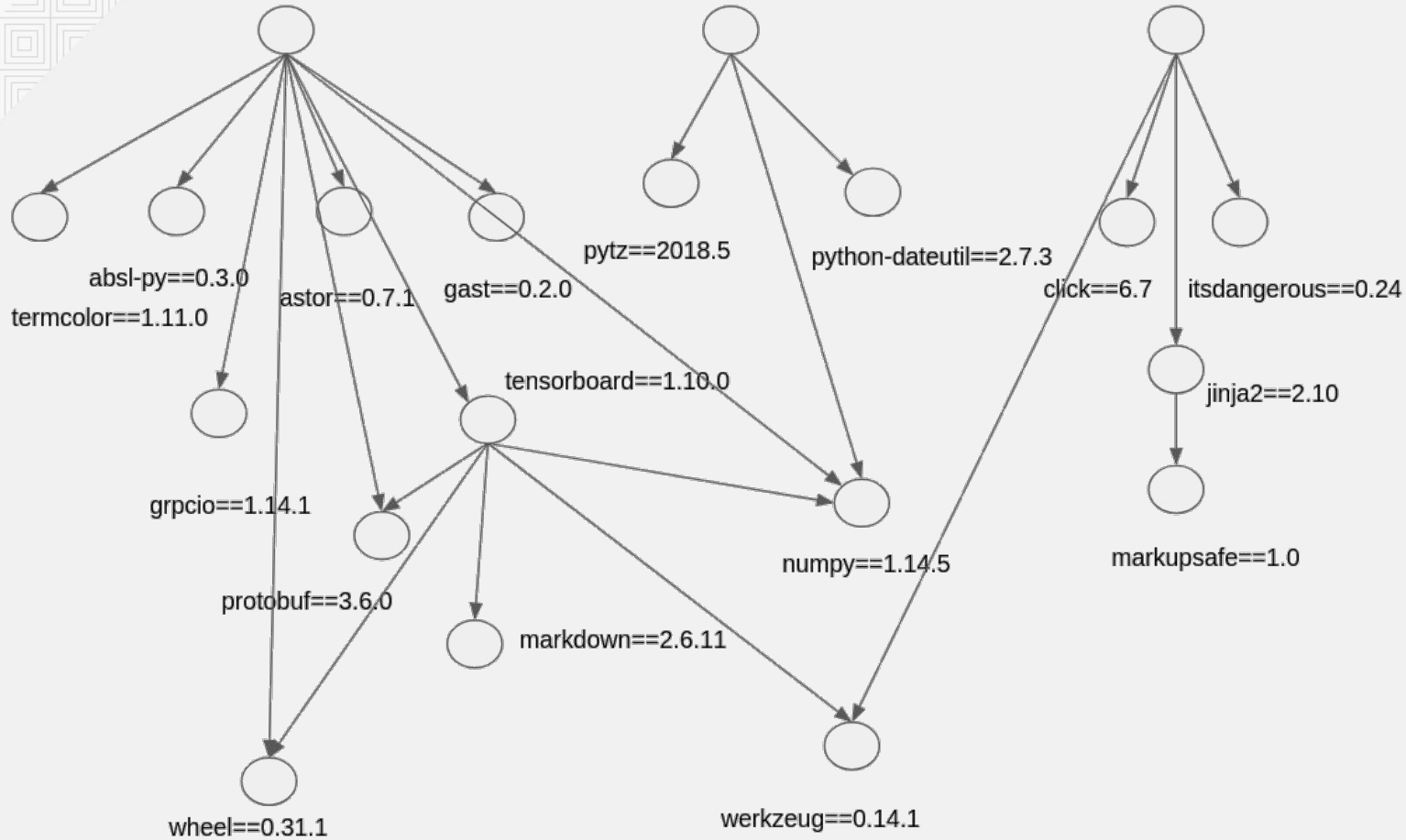
```
$ pip3 install flask tensorflow pandas gunicorn
```

tensorflow==1.10.0

pandas==0.23.4

flask==1.0.2

gunicorn==19.9.0



*package six was omitted

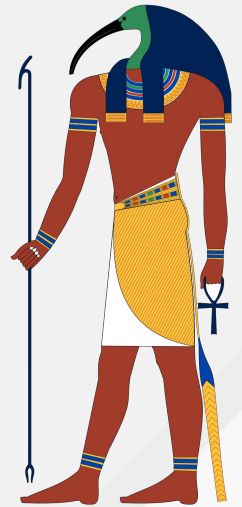
AI Stacks build pipeline

Create AI Stack variations and validate them. Automatically.

- Amun and Dependency Monkey create and validate Software Stacks
 - creation could be parameterized
 - validation is “does it build” and any given test application
- Osiris gathers and analyses build and inspection logs
 - to derive observations
- Right now, all team up to drive the core knowledge generation of Thoth
 - update our knowledge base on package updates
 - derive observations per software stack

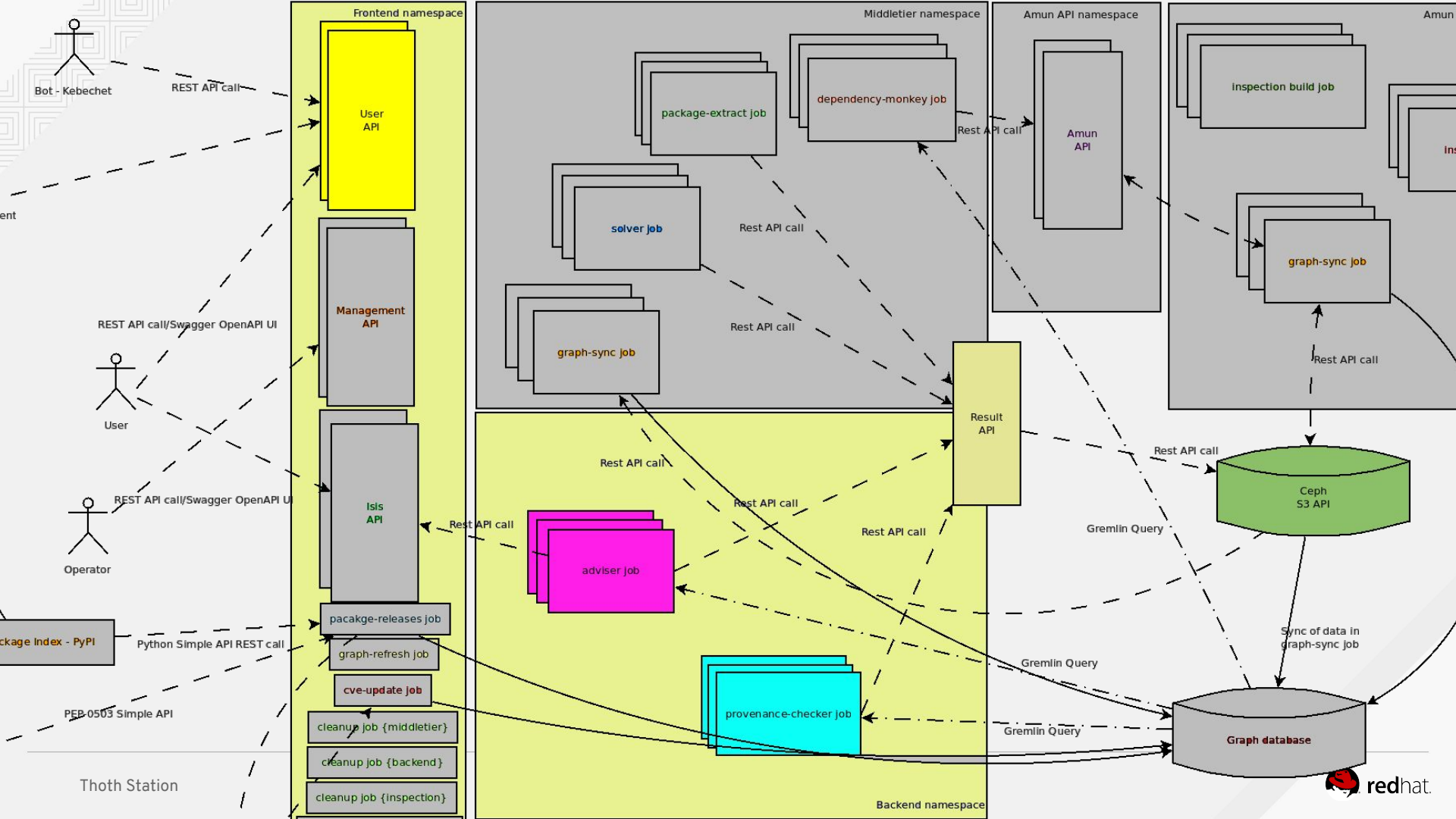
Thoth: Advise

“Thoth became heavily associated with the arbitration of godly disputes, the arts of magic, the system of writing, the development of science, learning, writing knowledge the judgment of the dead.”



Demo

```
$ thamos advise  
2019-01-24 15:18:31,397 [19305] INFO      thamos.lib: Sucessfully submitted advise analysis 'adviser-bb64d081d4880857'  
🕒 Waiting for response from Thoth (analysis: adviser-bb64d081d4880857)...
```

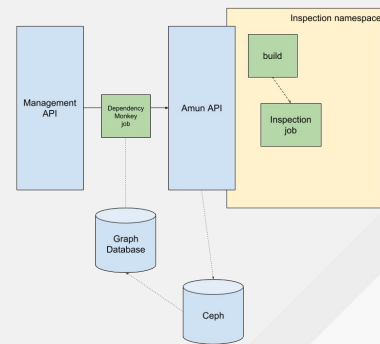
Thoth: Dependency Monkey

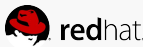


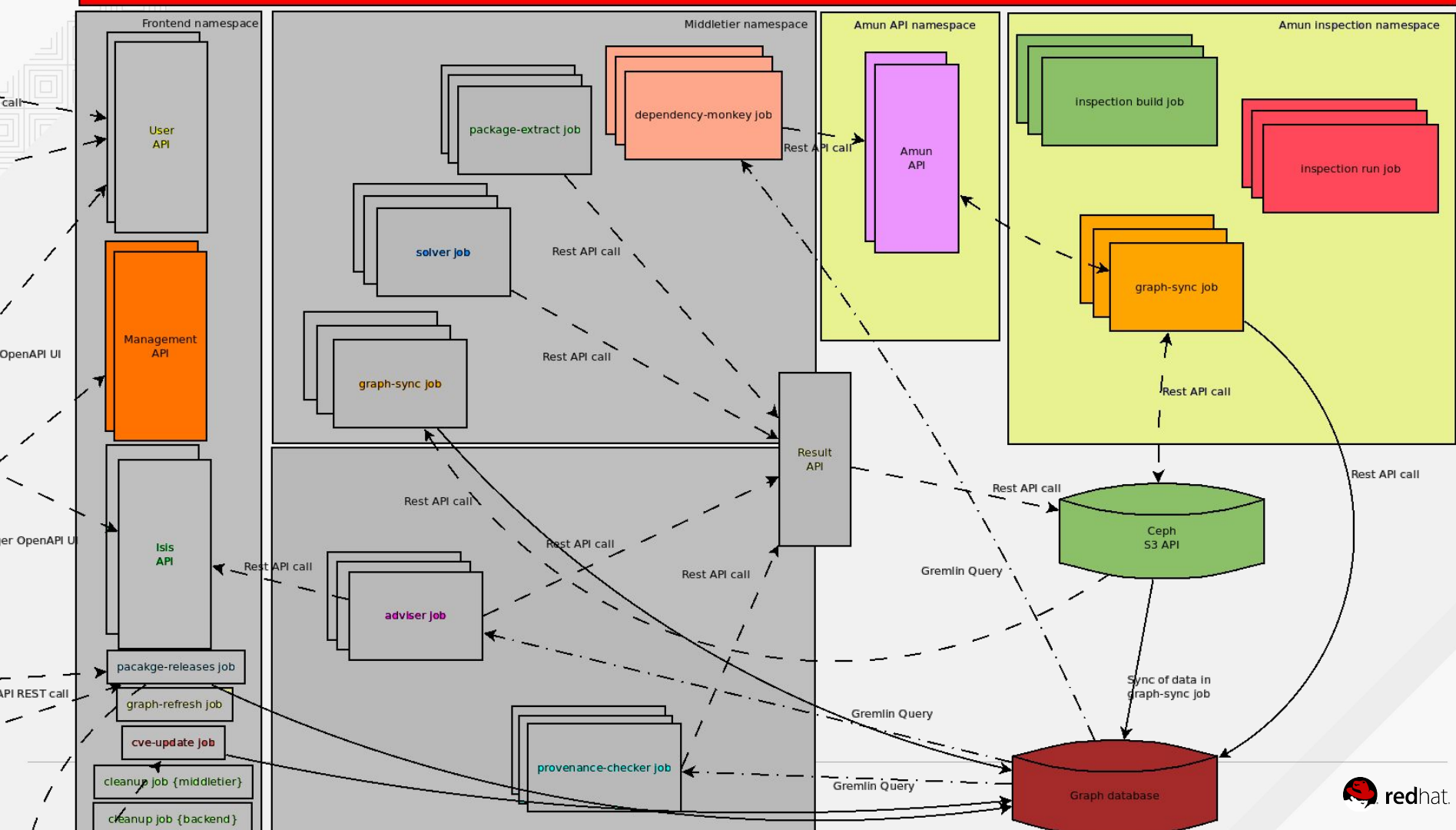
AI Stacks build pipeline: Dependency Monkey

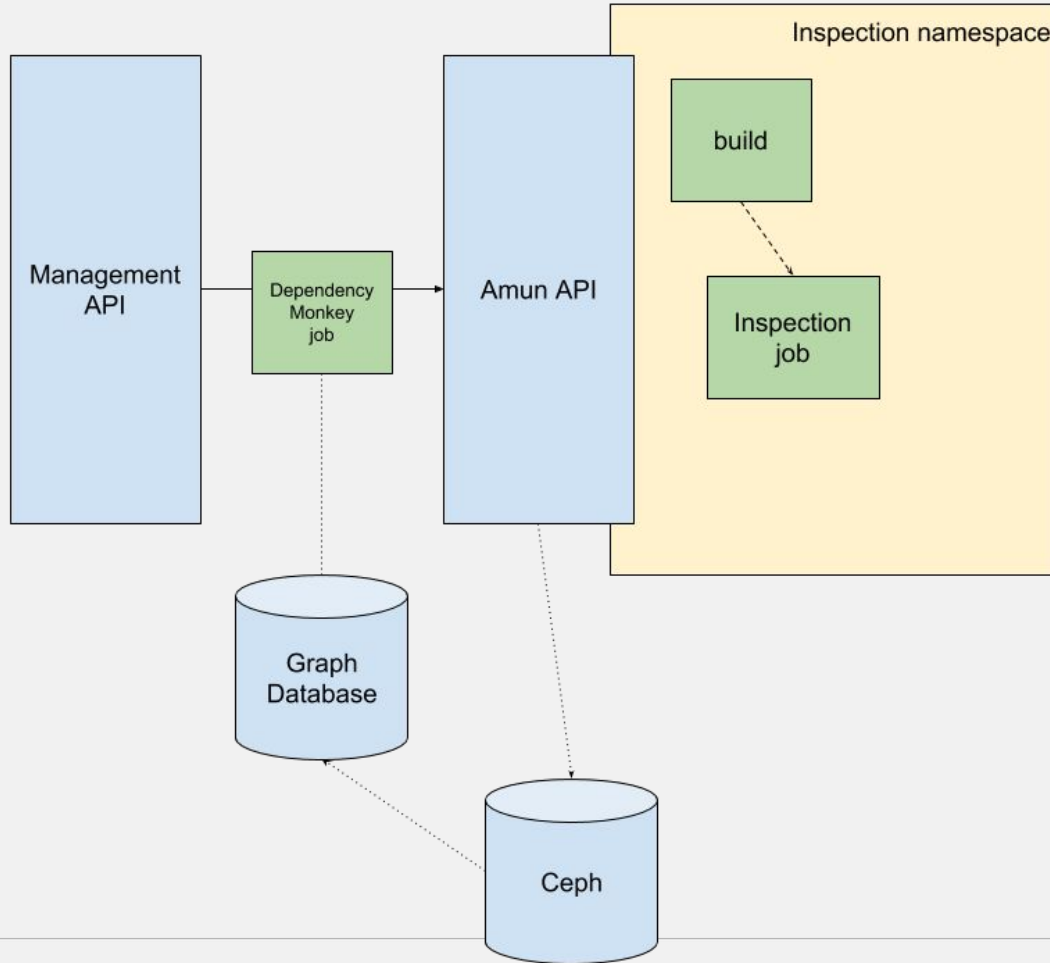
Generate all possible software stack permutations and request their validation, this cycle keeps knowledge base up to date.

- resolution of software stacks specifications, based on knowledge base rather than online PyPI data
- Solving software stacks works cross index, so optimized TensorFlow wheels are taken into account
- uses a base container image and can install additional RPM
- software stack permutations are generated in-memory, and a validation for each/latest/random is requested via the Amun API









AI Stacks build pipeline: Amun

Amun, a constraint workload scheduler, validates if a software stack could be built and executes provided test applications; observations are stored in a knowledge base.

- request validation of one **pinned down software stack**
 - Python Package Names and Version
 - Python Package Index to be used
- to be executed in a given **Runtime Environment**
 - CPU and microarchitecture specification
 - Base Operating System
- execute a test application
 - to verify function
 - to obtain performance indicators
- described declaratively using an API request
- all build log data is analysed to derive **Thoth Observations** and store them within our knowledge base

POST /dependency-monkey/python

[Run Dependency Monkey on the given application stack.](#)

Response Class (Status 202)

The Dependency Monkey analysis is scheduled.

Model Example Value

```
{
  "analysis_id": "string",
  "cached": true,
  "parameters": {}
}
```

Demo

Response Content Type [application/json](#)

Parameters

Parameter	Value	Description	Parameter Type	Data Type
input	<pre>{ "context": { "base": "fedora:29", "build": { "requests": { "cpu": "500m", "hardware": { "cpu_family": 6, "cpu_model": 94, "physical_cpus": 32, "processor": "Intel Core Processor (SkyLake, IBRS)" }, "memory": "512Mi" } }, "packages": ["pipenv", "which", "python36"], "run": { "requests": { "cpu": "500m", "hardware": { "cpu_family": 6, "cpu_model": 94, "physical_cpus": 32, "processor": "Intel Core Processor (SkyLake, IBRS)" }, "memory": "512Mi" } }, "script": "https://raw.githubusercontent.com/thoth-station/performance/master/tensorflow/matmul.py" }, "analysis_id": "string", "cached": true, "parameters": {} }</pre>	Specification of Python application stack with Context to Amun.	body	Model Example Value

Data Aggregation

Data Aggregation - solver

We aggregate information to give additional AI stack guidance.

- Solver - checks dependency graphs
 - Periodically monitoring package indexes and running solver for new releases
 - Create a report stating package dependencies
- An artificial TensorFlow==X stack:

```
absl-py>0.2.0
```

```
grpcio
```

```
numpy==1.13.3
```

```
protobuf>2.0.0<3.7.0
```

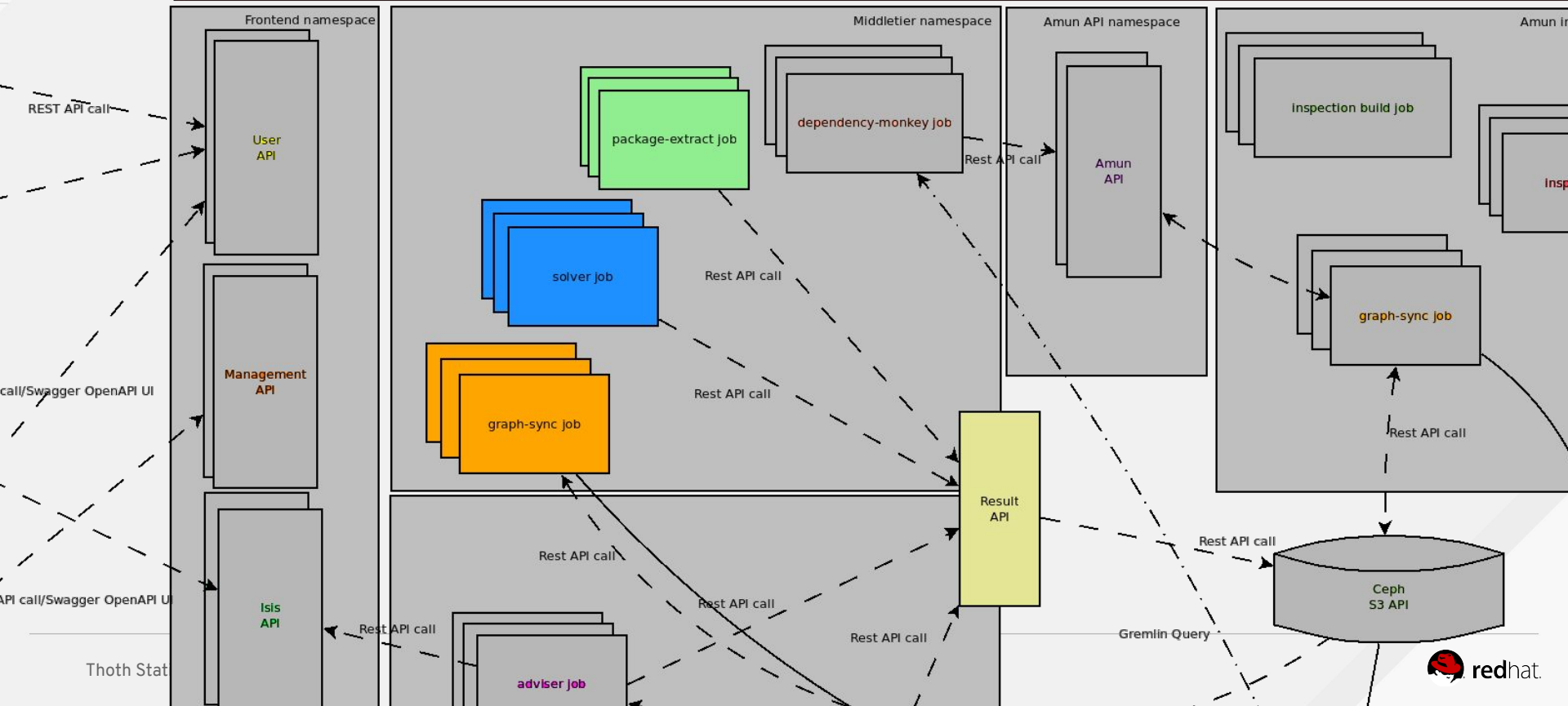
```
tensorboard>=1.0.0
```

Data Aggregation - package-extract

Check which packages are used by analyzing container images.

- Extracting container images and gathering information about packages used:
 - Native RPM packages
 - Python packages
 - Debian packages (experimental)
- Monitoring used packages for updates and automatically triggering analyses

OpenShift Container Platform - Master



GET

/installed-solvers

Retrieve a list of solvers installed and available.

POST

/solver/python

Register the given Python package.

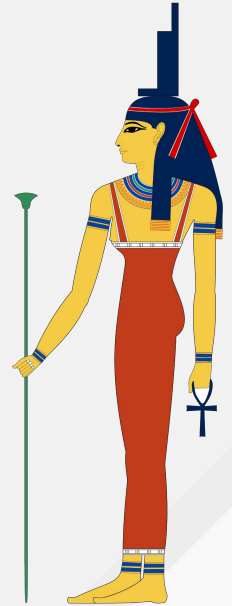
Demo

Parameters

Parameter	Value	Description	Parameter Type	Data Type				
secret	<input type="password" value="*****"/>	A secret to register the given Python package.	query	string				
no_subgraph_checks	<input type="checkbox" value="false (default)"/>	Do not perform subgraphs checks for packages already solved.	query	boolean				
debug	<input type="checkbox" value="false (default)"/>	Run the given analyzer in a verbose mode so developers can debug analyzer.	query	boolean				
python_package	<pre>{ "package_name": "tensorflow", "version_specifier": "==1.9.0" }</pre>	Python package that should be registered.	body	<table><tr><th>Model</th><th>Example Value</th></tr><tr><td></td><td><pre>{ "package_name": "selinon", "version_specifier": "==1.0.0" }</pre></td></tr></table>	Model	Example Value		<pre>{ "package_name": "selinon", "version_specifier": "==1.0.0" }</pre>
Model	Example Value							
	<pre>{ "package_name": "selinon", "version_specifier": "==1.0.0" }</pre>							

Parameter content type:

Isis: project2vec



"Isis was goddess of life and magic. She was one of the greatest goddesses."

Isis: project2vec

- A vector space model
- Each vector in vector space corresponds to a project
- Each item in vector represents a feature
- Allows feature based queries and similar projects search

$F = \{\text{python, machine-learning, web, django-framework, webassembly, sql, spark, gpu-support, Java}\}$

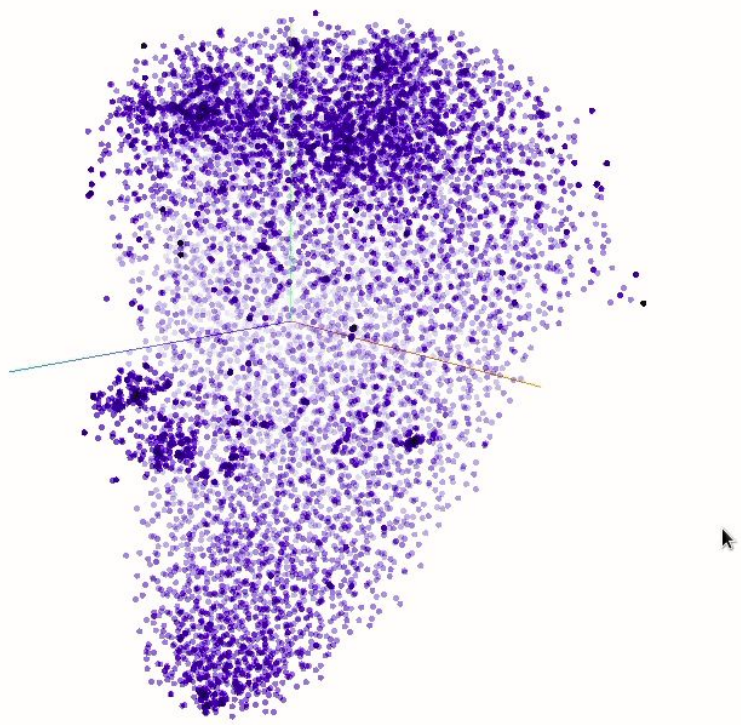
$F_{tf} = \{1, 1, 0, 0, 0, 0, 0, 1, 0\}$

F - feature vector

F_{tf} - feature vector for project TensorFlow

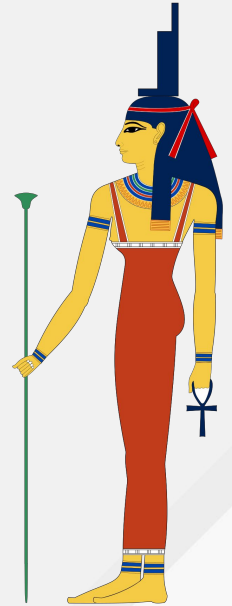


Demo



Kebechet: bot of freshness

"The goddess of freshness. She was thought to assist her father in his role as the god of embalming."



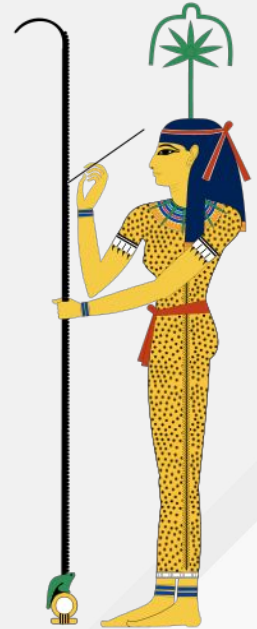
Kebechet: bot of freshness

- Bot consuming data from Thoth
- Monitoring repositories for library updates
- Automatic releases of Thoth libraries
- Checking application stacks of monitored repositories
- Communicating with Sesheta and Zuul-CI

<https://github.com/thoth-station/kebechet>

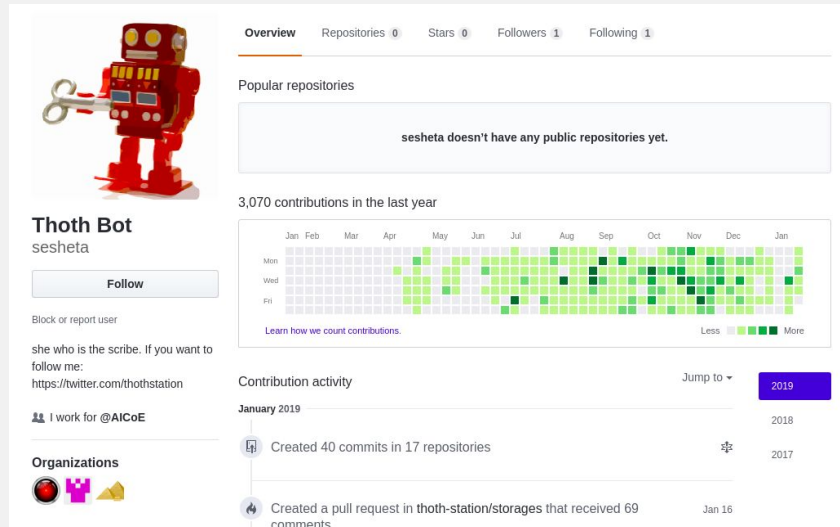
Sesheta: bot of writing

"She who is the scribe, and is credited with inventing writing. She also became identified as the goddess of accounting, architecture, astronomy, astrology, building, mathematics, and surveying."



Sesheta:

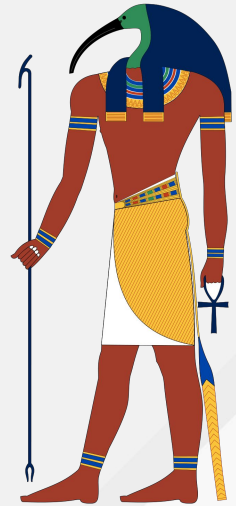
- Notifications on our chat channels
- Notifications on our GitHub repositories
 - Information from Zuul-CI
 - Package updates
- Opening issues and PRs to automated give stack guidance



<https://github.com/sesheta>

Thoth: mythological parts we didn't speak about

“Thoth's roles in Egyptian mythology were many. He served as scribe of the gods, credited with the invention of writing and Egyptian hieroglyphs. The ancient Egyptians regarded Thoth as One, self-begotten, and self-produced. He was the master of both physical and divine law. He was credited as author of all works of science, religion, philosophy, and magic.”



Thoth: parts we didn't speak about

- monitoring Thoth - Prometheus metrics
 - Grafana Dashboard - <https://url.corp.redhat.com/thoth-test-dashboard>
- Workflow management
 - OpenShift operators
- JanusGraph
 - [Database Schema](#)
- Zuul-CI
- Jupyter Notebooks - (not only) data science experiments
- Background data gathering pipeline - <https://github.com/thoth-station/selinon-api>
- Optimized TensorFlow releases on <https://tensorflow.pypi.thoth-station.ninja/>
 - TensorFlow build pipeline - @Subin Modeel
- Deployment using Ansible
 - DevOps container

Python packages
pip/Pipenv

Operator

Python Package Index - PyPI

Python Simple API REST call

PEP-0503 Simple API

AICoE Python package index

REST API call

pyup.io CVE database on GitHub

REST API call

Wheel binaries upload

TensorFlow
Build
Pipeline
API

package-releases job

graph-refresh job

cve-update job

cleanup job {middletier}

cleanup job {backend}

cleanup job {inspection}

workload-operator {middletier}

workload-operator {backend}

workload-operator {inspection}

graph-sync-operator {middletier}

graph-sync-operator {inspection}

adviser job

provenance-checker job

TensorFlow wheel files build job

Backend namespace

TensorFlow build pipeline namespace

It's all on you

- Community sitting at <https://github.com/thoth-station/>
- A lot of graph related data
- A lot of JSON files

```
$ pip3 install thamos  
$ cd ~/repositories/my-repo/  
$ thamos config
```

References

We are open community. Open to ideas, contributions or feedback in any form.

- Core concepts and general overview:
 - <https://github.com/thoth-station/thoth>
- Infrastructure and deployment
 - <https://github.com/thoth-station/core>
- Documentation for adviser, provenance checks and Dependency Monkey
 - <https://github.com/thoth-station/adviser>
- Amun
 - <https://github.com/thoth-station/amun-api>
- project2vec
 - <https://github.com/thoth-station/isis-api>



[thothstation](https://twitter.com/thothstation)



THANK YOU



plus.google.com/+RedHat



facebook.com/redhatinc



linkedin.com/company/red-hat



twitter.com/RedHat



youtube.com/user/RedHatVideos