Enable language package managers in TERN, increase test coverage.

About me

Name: Abhay Katheria

Academic Details: 2nd-year Computer Science undergraduate at MNNIT Allahabad (Expected Graduation August 2022)

Github: http://github.com/abhaykatheria

Time zone: GMT+5:30

Code contributions

Project-related:

Tern :

- **Merged PR's**
  - Added a check to confirm the mount points #544
  - Added --rm in docker_run.sh #553
  - Fix TypeError for Dockerfile scratch base images
  - Add -wd argument to change the working directory #564
- **Issues Working on**
  - Increase Test Coverage #539
  - Add test for expand_from_images in test_analyze_docker_dockerfile.py #567
  - Extracting metadata of packages and modules installed by language package managers

Others:

Mozilla/bugbug:

- **Merged PR's**
  - made sure duplicate features are not used #1215
  - Set label as 0 in the QANeeded model when one of the qa flags are removed #1251
  - Stop using sklearn.metrics.classification as it is deprecated #1185
  - Made utils.download_check_etag automatically detect the file name from the URL #1112

Project Information

Sub-Org Name: TERN

Project Abstract:

1. Add functionality in Tern to get information about the packages installed through different package managers.
2. Add more Unit tests to increase the coverage of tern.

Detailed Description:

1. Adding language packages metadata extraction

Tern needs to be able to understand how to get license information for packages installed with language package managers. This includes packages and libraries installed via **pip**, **npm, gem, and golang.**

Tern already has support for this. We can add scripts for extracting the metadata of each package manager in snippets.yml tern will execute those scripts in a chroot jail and get the metadata required.

I've already figured the required scripts and tested them on different relevant docker images. Here are they

Pip

```
pip3:
 install: 'install'
 remove: 'uninstall'
 ignore:
   - 'freeze'
   - 'list'
   - 'download'
   - 'show'
 packages:
   - name: default
     version:
       invoke:
         1:
           container:
             - "pip3 show {package} 2> /dev/null | sed -n '2 p' | awk '{print $2}'"
     proj_url:
       invoke:
         1:
           container:
             - "pip3 show {package} 2> /dev/null | sed -n '4 p' | awk '{print $2}'"
     license:
       invoke:
         1:
           container:
             - "pip3 show {package} 2> /dev/null | sed -n '7 p' | awk '{print $2}'"
     deps:
       invoke:
         1:
           container:
             - "pip3 show {package} 2> /dev/null | sed -n '9 p' |  awk -F':' '{ print $2}' | tr ',' '\n' |tr -d '^ ' "
       delimiter: "\n"
```

Npm

```
npm:
 install: 'install'
 remove: 'uninstall'
 ignore:
  - 'ls'
  - 'list'
  - 'ping'
  - 'restart'
 packages:
  - name: default
   version:
    invoke:
     1:
      container:
       - "npm view {package} | sed -n '2 p'  | awk '{print $1}' | awk -F'@' '{print $2}'"
   proj_url:
    invoke:
     1:
      container:
       - "npm view {package} | head -n 4 | tail -n 1 "
   license:
    invoke:
     1:
      container:
       - "npm view {package} | sed -n '2 p'  | awk '{print $3}'"
   deps:
    invoke:
     1:
      container:
       - npm view {package} | awk '/dependencies:/{flag=1;next}/maintainers:/{flag=0}flag'
    delimiter: "\n"
```

Gem

```
gem:
  install: 'install'
  remove: 'uninstall'
  ignore:
    - 'ls'
    - 'list'
    - 'ping'
    - 'restart'
  packages:
    - name: default
      version:
        invoke:
          1:
            container:
              - "gem info {package} 2>/dev/null |sed -n '1 p' | awk -F '[()]' '{print $2}'"
      proj_url:
        invoke:
          1:
            container:
              - "gem info rake 2>/dev/null |sed -n '3 p'| awk '{print $2}'"
      license:
        invoke:
```

```
    1:
     container:
      - "gem info rake 2>/dev/null |sed -n '4 p'| awk '{print $2}'"
  deps:
   invoke:
    1:
     container:
      - #working on this
   delimiter: "\n"
```

Yarn

I’ve recently saw some images which uses yarn as a package manager in their node js project instead of npm. So I can also write similar scripts for it.

GoLang

I am currently writing scripts to get the package level data I will figure it out soon. And like others, it will be added to snippets.yml in the same manner. Adding this is tough as golang manages its packages in a different way than other package managers.

The above written scripts are not finalized; they need thorough testing on different images.

2. Writing Unit Tests.

- I’ve already written tests for
  i. tern/analyze/docker/analyze.py
  ii. tern/analyze/docker/helper.py
  iii. Unit test for expand_from_image in tern/analyze/docker/dockerfile.py

  But I’ve not opened a PR, because some of the tests need to be run on a DockerImage Once we have an official test_image on docker hub I will complete these and open the pull requests.

- I intend to write more unit tests for
  - tern/command_lib/command_lib.py
  - tern/report/content.py
  - tern/report/report.py

3. Stretch Goals

- If everything mentioned above is completed within the time limit then, in that case, I can help with:
  i. Perform analysis on multistage Dockerfiles #612
  ii. Also in the past, I’ve done a few projects on flask, and web.py so I have some idea about how micro web frameworks and can be used to display json on the browser.
  UI for Tern json data #614

Timeline

Time Period	**Milestone**
Community Bonding Period: **May 5 - May 31**	● Writing some tests to increase coverage. ● Interact with the mentor and finalize project details. ● Learning about Golang package Management
Week 1: **June 1 - June 7**	● Testing Scripts for pip
Week 2: **June 8 - June 14**	● Writing Scripts for Yarn
Week 3**: June 15 - June 21**	● .Testing Scripts for Yarn
Week 4**: June 22 - June 28**	● Writing scripts for Golang
Week 5**: June 29 - July 5**	● Testing Scripts for Golang
Week 6**: July 6 - July 12**	● Writing scripts for npm.
Week 7: **July 13 - July 19**	● Testing scripts for npm.
Week 8: **July 20 - July 26**	● Finalizing the proposal 1.
Week 9: **July 27 - August 2**	● Start working on tests.
Week 10: **August 3 - August 9**	● Increasing test coverage
Week 11: **August 10 - August 16**	● Finishing the testing part. ● Documenting everything.
Week 12: **August 17 - August 23**	● Extra week for covering backlogs.
Week 13: **August 24 - August 31**	● Final Week: Finalization of the project for evaluation.

Other Commitments

Due to the Covid-19 outbreak, my summer break has started early and it is possible that I will have my end-semester examination in b/w the GSoC period. So I might have to take a week off for the preparation.