Testing the entire npm registry

nodechecker.com

Who am i

- Pedro Dias
  - @pedromdias
  - http://abru.pt

- Fullstack engineer at ptisp.pt
- Lecturer at ipt.pt
- Petrolhead
- Hiker
- Soon to be father (if I leave this room running, now you know why)

Why

- Autonomously test all modules in npm registry
  - Will results have statistical weight?
  - Is it practicable?
    - Resource and time wise
  - Could it be done autonomously?

I had no idea... that made me curious...

Disclaimer

- Do not make judgements on these results.
- This isn’t a scientific metric.
- All data is available.
- So far only was debugged and fixed what was needed for this to run.

1st iteration

Like all nightly proof-of-concepts ...

So...

1. Read npm’s couchdb dump
2. Iterate all modules
    a. git clone
    b. npm install
    c. npm test
3. Save exit codes in Redis

Issues...

- npm
  - npm init
  - npm test
- Outdated repositories

Weird things started happening...

- Redis started gaining life.
- But it got worse...

People really do crazy shit in their tests without any warning.

Next time you do a ‘npm test’ think about this...

2nd iteration

- Test environment needed to be sandboxed.
- Switching from repos to tarballs

Docker

- Why docker
  - Docker isn’t trying to be what it isn’t.
  - Good toolset (try using plain lxc or openvz)
  - API
  - Execution is easy
- Test standardization/repeatability
- Process isolation

Docker image

- https://index.docker.io/u/apocas/nodechecker
- Installed
  - Node.js v0.10.15
  - Git
  - wget
  - Redis (recent)

Container lifecycle

1. Create a container
2. Attach to container
3. Start the container
4. Wait for it to end
5. Destroy it

Container execution

- https://github.com/dotcloud/docker/wiki/Docker-run-improvements
  - Giving ‘docker run’ a process’ish like behavior.

- http://docs.docker.io/en/latest/api/docker_remote_api_v1.3/#inside-docker-run
  - How ‘docker run’ works.

3rd iteration

- Improving container lifecycle
  - Attach to it
  - Control containers IDs
  - Remote API
    - http://docs.docker.io/en/latest/api/docker_remote_api_v1.3
- Multiple module sources

Module sources

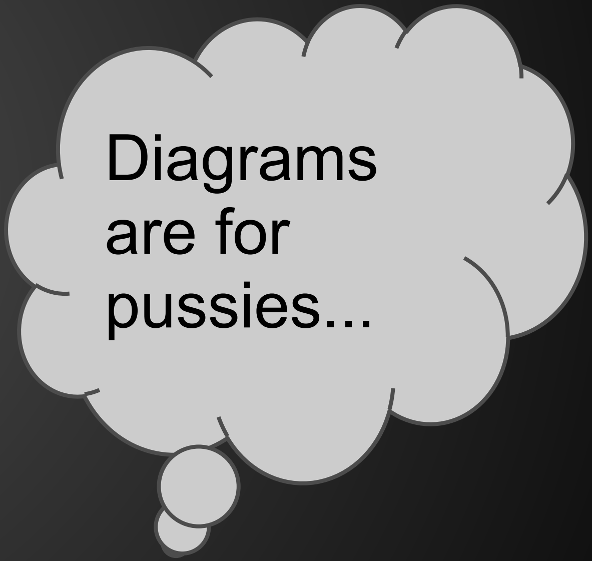
- Module tarball
  - From npm registry
  - Tests in .npmignore
- Repository
  - Specified in package.json
  - May be outdated/invalid
  - May be unstable

.npmignore

- Scenario
    a. Module’s package.json specifies a test script.
    b. Dev added his test code to .npmignore

Cycle

- Module has test script in package.json?
  - No
    - NOTTESTED
  - Yes
    - It’s a dummy script?
      - Yes
        - NOTTESTED
      - No
        - Have been tested before?
          - No
            - Test using tarball
              - Exit code 0
                - OK
              - Exit code != 0
                - NOK
          - Yes
            - Test using repository
              - ...



Container limits

- CPU is not a problem.
- Memory is!
  - Modules using 1gb+
  - Container memory limit
    - Stuck containers

Funny

- https://npmjs.org/package/ifyouwanttogetthesumoftwonumberswherethosetwonumbersarechosenbyfindingthelargestoftwooutofthreenumbersandsquaringthemwhichismultiplyingthembyitselfthenyoushouldinputthreenumbersintothisfunctionanditwilldothatforyou
- https://npmjs.org/package/dos-fork-bomb

Weird

- 404’s in some packages - Fixed. Isaac did his magic.
- Empty npm info - Yet to debug/question.

Source

- nodechecker-test
    - Container abstraction
    - https://github.com/apocas/nodechecker-tester
- nodechecker-engine
    - Uses nodechecker-test
    - Iterates the npm registry
    - https://github.com/apocas/nodechecker-engine
- nodechecker.com
    - Website
    - API
    - https://github.com/apocas/nodechecker.com