Building The Data Used To Build The Internet2 Route Reports

Ryan Harden
Senior Cyberinfrastructure Security Architect
Internet2 – Network Services - Security
What are the Internet2 Route Reports?

- Talk to Steve Wallace! (ssw@internet2.edu)
- If you manage IP resources, you should familiarize yourself with them.
- "Report Card" for Routing Integrity
- https://github.internet2.edu/ssw/IRR-report/tree/master/Connectors
How we “used” to build them?

- Steve’s laptop
- Monolithic python script (2500 lines)
- Lots of manual gathering of data
- External API/Whois Lookups
- Hours to complete
  - API rate limits
- PoC code grew to what it is today
- ”Quick and Dirty”
Why the change?

• Hugely Popular Reports

• Ryan likes to automate…
  – Obligatory XKCD
• “Has to be a better way…”
• Steve is allowed PTO
• Time to rethink and refactor.

• Scheduled Runs
• Repeatable
• Quick corrections

• Lots of other uses for source data
CI/CD with Gitlab-CI

• “CI/CD is a continuous method of software development, where you continuously build, test, deploy, and monitor iterative code changes.”
• Focused on continuous and/or automated software releases.

• But you don’t have to use it like that...

• I use it like a really advanced cron.
  – Barely scraping the surface.
  – Scheduled Runs
  – Job Dependencies
  – Automated Deployment

• Why not GitHub Actions?
Definitions

- **Pipeline**
  - End-to-End configuration of the whole process
  - Executed on a schedule or when triggered

- **Stage**
  - A collection of jobs that do similar tasks
  - All jobs in a stage must finish/succeed before progressing to next stage
    - Configurable

- **Job**
  - A set of commands needed to complete the work
  - Containers
  - Runners
    - Cloud vs Local
**CI Stages**

- **CI Pipeline Stages**
  - Internal mapping data (Connector lists, etc) static files.

- **.pre**
  - Download previous run data.
    - To check for changes
  - Gather Internet2 Router Output Data
    - Executed in a different repository and CI Pipeline
  - Download Various Datasets
    - RPKI, PeeringDB Info
    - ARIN Bulk Whois, IRR Databases
    - MANRS data, Global Internet Routing Table
CI Stages

- build
  - Compare IRR CURRENTSERIAL data to check for changes
    - If necessary, download updated IRR databases
  - Parse and Pickle IRR Data
    - Read flat text file, parse it into radix trees, pickle it.
  - Pickle RPKI Data
  - Parse ARIN Bulk Whois Data
  - Get AS-SETs and AS Cones
  - Parse Global Routing Table
CI Stages

- test
  - Does output JSON conform to a schema, etc
    - jsonschema
  - Might do some unit testing, probably not much
CI Stages

- deploy
  - Kind of a misnomer for this project
  - Bundle all gathered and processed data
    - Easy consumption by downstream projects
parse_global_table:
  stage: build
  tags:
    - security-runner
  script:
    - python parse-bgp-table.py
  needs:
    - job: download_global_table
      artifacts: true
  artifacts:
    name: "Global Table"
    untracked: false
    expire_in: "1 days"
    paths:
      - "Global_Table/"
    when: on_success

get_manrs_asns:
  stage: .pre
  needs:
    - job: get_latest_artifacts
      artifacts: true
  script:
    - mkdir MANRS
    - pip install beautifulsoup4 requests
    - python get-manrs-asns.py
  artifacts:
    expose_as: "MANRS_ASNs"
    name: "MANRS_ASNs"
    untracked: false
    expire_in: "1 days"
    paths:
      - "MANRS/"
    when: on_success
    allow_failure: true
<table>
<thead>
<tr>
<th>.pre</th>
<th>build</th>
<th>test</th>
<th>deploy</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️ download_arin_bulk_whois</td>
<td>✔️ download_irr_data</td>
<td>✔️ check_artifact</td>
<td>✔️ deploy_artifact</td>
</tr>
<tr>
<td>✔️ download_global_table</td>
<td>✔️ get-as-sets-and-as-cones</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>✔️ download_rpki_data</td>
<td>✔️ parse_arin_bulk_data</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>✔️ get_latest_artifacts</td>
<td>✔️ parse_global_table</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>✔️ get_mans_asns</td>
<td>✔️ parse_irr_data</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>✔️ get_nso_report_json</td>
<td>✔️ pickle_irr_data</td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>✔️ get_peeringdb_info</td>
<td>✔️ pickle_rpki_data</td>
<td>✔️</td>
<td></td>
</tr>
</tbody>
</table>
## Stages and Jobs

<table>
<thead>
<tr>
<th>pre</th>
<th>build</th>
<th>test</th>
<th>deploy</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="download_arin_bulk_whois" alt="check" /></td>
<td><img src="download_irr_data" alt="check" /></td>
<td><img src="check_artifact" alt="check" /></td>
<td><img src="deploy_artifact" alt="check" /></td>
</tr>
<tr>
<td><img src="download_rpki_data" alt="x" /></td>
<td><img src="get-as-sets-and-as-cones" alt="check" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td><img src="get_latest_artifacts" alt="check" /></td>
<td><img src="parse_arin_bulk_data" alt="check" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td><img src="get_manrs_asns" alt="check" /></td>
<td><img src="parse_irr_data" alt="check" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td><img src="get_nso_report_json" alt="check" /></td>
<td><img src="pickle_irr_data" alt="check" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td><img src="get_peeringdb_info" alt="check" /></td>
<td><img src="pickle_rpki_data" alt="check" /></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Job Dependencies
Interesting Notes and Tidbits

- 1100 lines of python (so far)
- 22 minute runtime (Daily)

- Python Pickle
- Radix Trees
- BeautifulSoup (What the heck is that?)
- TTP (Text Template Parser) Hurray!
- FTP? In 2023?

- Foundations done, still working on some details
  - Normalized output
    - AS1234 vs 1234, what about 4-byte ASNs?
THANKS

rharden@internet2.edu