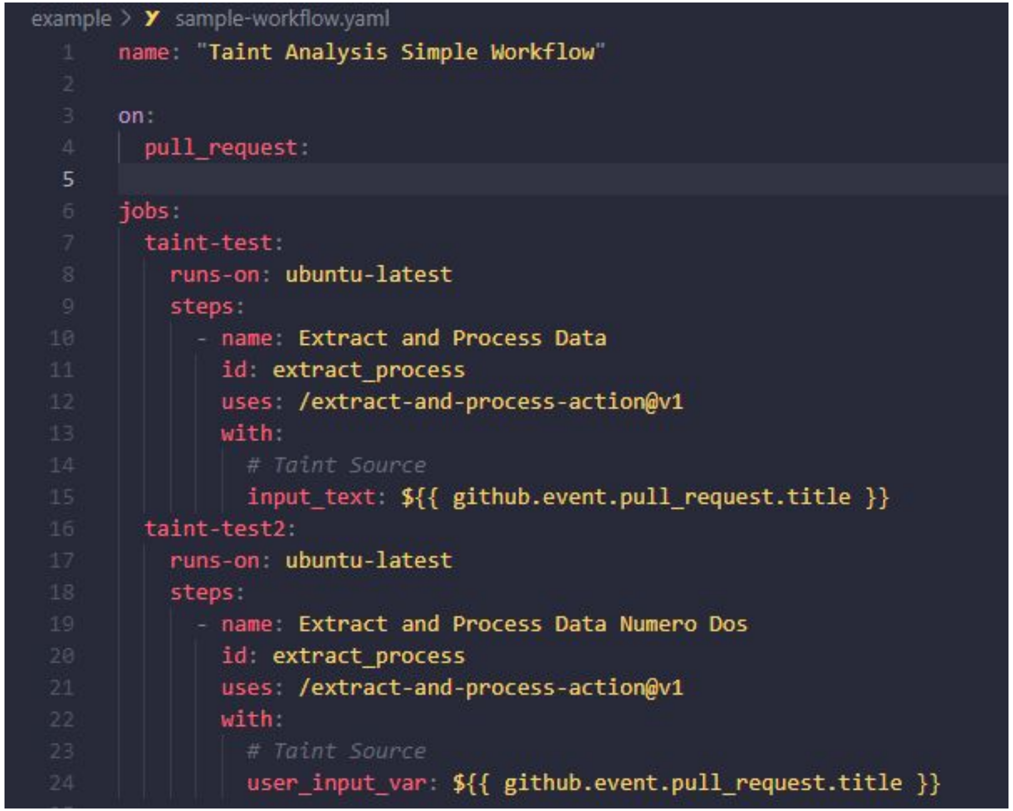
Staged Static Taint Analysis Tool for Docker Actions

Travis Hill

Background/Recall - GitHub Workflow

- GitHub Workflow?
  - YAML file in special directory
  - Executes jobs on a GitHub Event
- Jobs?
  - Compromised of steps.
  - Steps execute a specific task



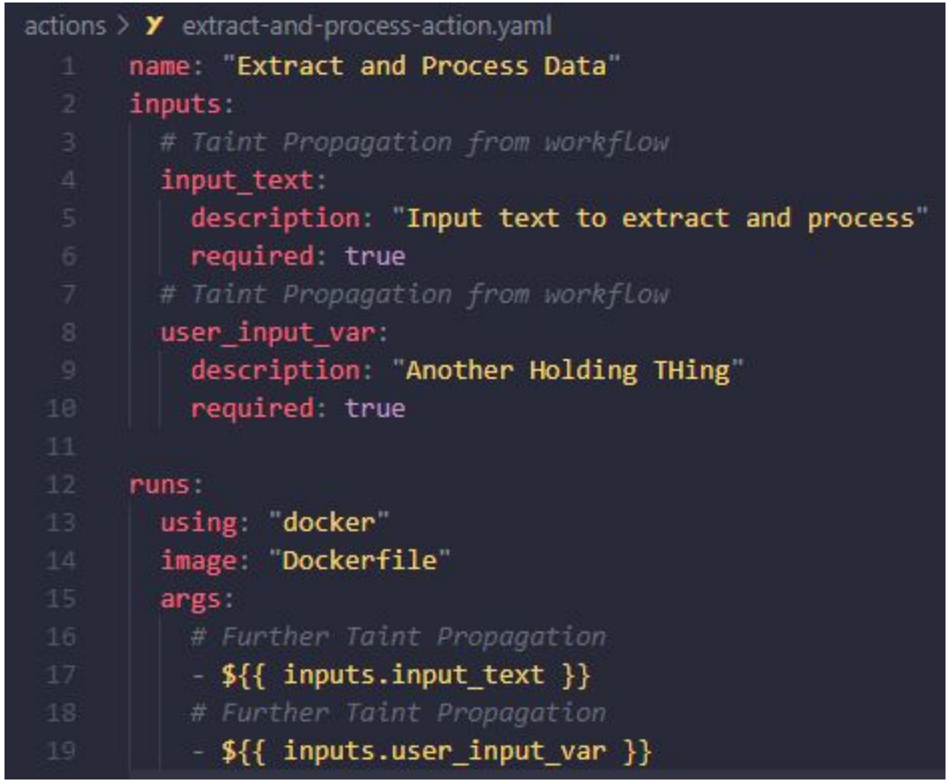
```yaml
name: "Taint Analysis Simple Workflow"

on:
  pull_request:

jobs:
  taint-test:
    runs-on: ubuntu-latest
    steps:
      - name: Extract and Process Data
        id: extract_process
        uses: /extract-and-process-action@v1
        with:
          # Taint Source
          input_text: ${{ github.event.pull_request.title }}
  taint-test2:
    runs-on: ubuntu-latest
    steps:
      - name: Extract and Process Data Numero Dos
        id: extract_process
        uses: /extract-and-process-action@v1
        with:
          # Taint Source
          user_input_var: ${{ github.event.pull_request.title }}
```

Background/Recall - GitHub Actions

- GitHub Action?
  - Reusable Code Snippets
  - Input from Workflows
- Docker Action?
  - Executes using a Docker Container
  - Local or Remote Image
- Other Actions?
  - Executes using JavaScript/Shell



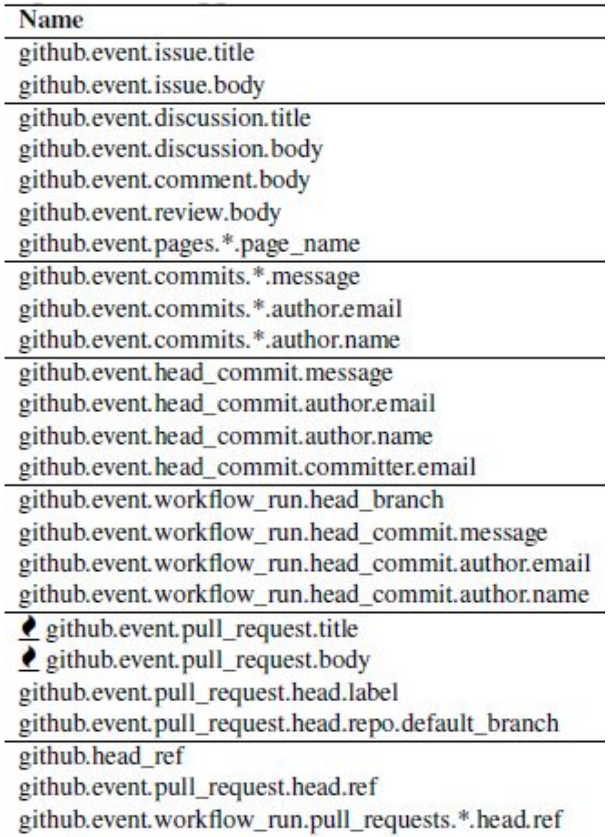
actions > extract-and-process-action.yaml

```yaml
name: "Extract and Process Data"
inputs:
  # Taint Propagation from workflow
  input_text:
    description: "Input text to extract and process"
    required: true
  # Taint Propagation from workflow
  user_input_var:
    description: "Another Holding THing"
    required: true

runs:
  using: "docker"
  image: "Dockerfile"
  args:
    # Further Taint Propagation
    - ${{ inputs.input_text }}
    # Further Taint Propagation
    - ${{ inputs.user_input_var }}
```

Background/Recall - What's Considered Tainted?

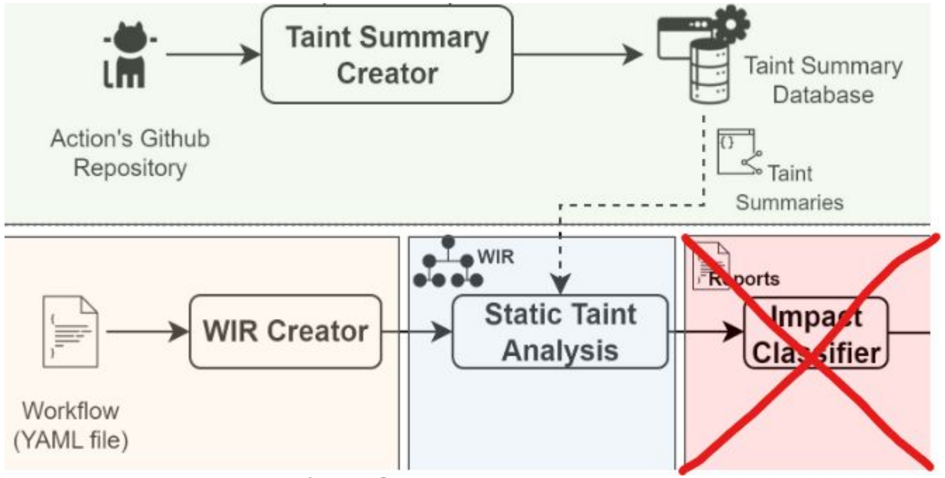
- GitHub Events With User Input
  - Issues, discussions, commits, pull & merge requests



Name
github.event.issue.title
github.event.issue.body
github.event.discussion.title
github.event.discussion.body
github.event.comment.body
github.event.review.body
github.event.pages.*.page_name
github.event.commits.*.message
github.event.commits.*.author.email
github.event.commits.*.author.name
github.event.head_commit.message
github.event.head_commit.author.email
github.event.head_commit.author.name
github.event.head_commit.committer.email
github.event.workflow_run.head_branch
github.event.workflow_run.head_commit.message
github.event.workflow_run.head_commit.author.email
github.event.workflow_run.head_commit.author.name
🔥 github.event.pull_request.title
🔥 github.event.pull_request.body
github.event.pull_request.head.label
github.event.pull_request.head.repo.default_branch
github.head_ref
github.event.pull_request.head.ref
github.event.workflow_run.pull_requests.*.head.ref

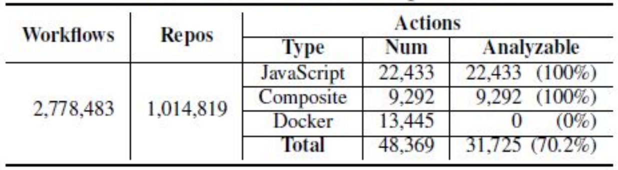
MY APPROACH/CONTRIBUTION

- New Standalone Tool
- Focus specifically on Docker Action Support
- Adopt the ARGUS workflow
- Substitute CodeQL with Python



Taken from ARGUS

Table 2: Dataset 1: Public Repositories



Workflows	Repos	Actions		
		Type	Num	Analyzable
2,778,483	1,014,819	JavaScript	22,433	22,433 (100%)
		Composite	9,292	9,292 (100%)
		Docker	13,445	0 (0%)
		Total	48,369	31,725 (70.2%)

SHORT DEMO/CODE WALKTHROUGH

Shortcomings

- Python Regex != CodeQL
  - Issues tracking the entire flow of execution
  - Issues tracking which taint sources affect which sinks
- Standalone tool
  - No support for JavaScript or Composite Actions
- No effective way to thoroughly test the tool
  - Dataset used in ARGUS lacks Docker Actions
  - Crawling 2.8 million repositories like ARGUS is not very practical for the time frame
  - Using Docker Actions is relatively rare

Future Plans

- Add Functionality To Parse/Analyze Dockerfiles
    - “Merge” Workflows Into One WIR During Processing to see how the data interacts directly
    - Add a new data structure in taint analysis class that keeps track of taint sources and the variables they touch. Similar to a WIR structure
- Rework the taint identification to prevent less false positives
    - Didn’t incorporate the table shown in the background yet
- Reorganize the file structuring of the parsing to work more universally
    - Does not read from the expected workflow file destination
- Try to find better testing material in public repositories
    - Specifically Docker Action usage
    - Not sure how to go about this efficiently