ARGUS: A Framework for Staged Static Taint Analysis of GitHub Workflows and Actions

Siddharth Muralee, Igibek Koishybayev, Aleksandr Nahapetyan, Greg Tystahl, Brad Reaves, Antonio Bianchi, William Enck, Alexandros Kapravelos, Aravind Machiry

Purdue University                            North Carolina State University
Github Actions

- CI/CD (Continuous Integration/Continuous Deployment) platform developed by GitHub in 2018
- Features
  - Developers define **Workflows** which automate various steps of the software development lifecycle, such as building, testing, and deploying code.
  - Workflows can use **Actions** which are applications that perform commonly repeated tasks. These are developed by the community and can be found on the market place.
  - Workflows are **Event Driven** and can be triggered by specific GitHub events such as a pushing a commit, creating a pull request, or opening an issue.

**Our Goal**: Identify Command Injection vulnerabilities in Github Actions
Motivating Example

```yaml
name: Sample Workflow
on:
  issues:
    types: [opened]
jobs:
  notify:
    - name: Log title
      run: echo "${{ github.event.issue.title }}"

  - name: Log title
    run: echo "${{ github.event.issue.title }}"
```

```
# echo "My first issue"
ls;
# echo "Hello"; ls; #
```

- **49 Open** 0 Closed
- My first issue
  - #74 opened now by R3x

- **50 Open** 0 Closed
- Hello"; ls; #
  - #75 opened now by R3x

```bash
```
Triggers
Determines the events on which the workflow is run

Jobs
Independent tasks that usually run in different Virtual Environments

Steps
Jobs have a number of steps that get executed sequentially

```
name: My custom workflow
on:
  pull_request:
    types: [opened]
jobs:
  build:
    outputs:
      build: '${{ steps.build.outputs.build }}'
    steps:
      - uses: actions/checkout@v3
        with:
          ref: main
      - id: build
        run: |
          sudo ./build.sh
          if [ $? -eq 0 ]; then
            echo "build=true" >> $GITHUB_OUTPUT
          fi
    test:
      needs: build
      steps:
      - if: needs.build.outputs.build == 'true'
        run: |
          echo "Successfully built - ${{ github.event.pull_request.title }}"
```
### Background

**Triggers**
Determines the events on which the workflow is run.

**Jobs**
Independent tasks that usually run in different Virtual Environments.

**Steps**
Jobs have a number of steps that get executed sequentially.

**Outputs/Env Variables**
Jobs and Steps can share data using Outputs and Env Variables.

**Dependencies**
A Job executes only after all its dependent Jobs have completed.

**Event Data**
Workflows can access event data using special variables.

---

```yaml
name: My custom workflow
on:
  pull_request:
    types: [opened]
jobs:
  build:
    outputs:
      build: ${steps.build.outputs.build}
    steps:
      - uses: actions/checkout@v3
        with:
          ref: main
      - id: build
        run: |
          sudo ./build.sh
          if [ $? -eq 0 ]; then
            echo "build=true" >> $GITHUB_OUTPUT
          fi
test:
  needs: build
  steps:
    - if: needs.build.outputs.build == 'true'
      run: |
        echo "Successfully built - ${{ github.event.pull_request.title }}"
```
Motivation

Permutations

Controls the privileges of the GITHUB_TOKEN used by the workflow to interact with Github API

```
static.yml
# Simple workflow for deploying static content to GitHub Pages
name: Deploy static content to Pages
on:
  # Runs on pushes targeting the default branch
  push:
    branches: ['master']
# Allows you to run this workflow manually from the Actions tab
workflow_dispatch:
# Sets permissions of the GITHUB_TOKEN to allow
# deployment to GitHub Pages
permissions:
  contents: read
  pages: write
  id-token: write
```

```
build.yml
# Simple workflow for building the environment
name: Deploy static content to Pages
on:
  # Runs on pushes targeting the default branch
  push:
    branches: ['master']
jobs:
  build:
    steps:
      - name: Checkout code
        uses: actions/checkout@v2
        with:
          repository: my-org/my-repo
          ssh-key: '${{ secrets.SSH_PRIVATE_KEY }}'
```
Threat Model

- Execute **Arbitrary Commands** without visible code changes
- Gain **Unauthorized Read/Write** access to repository
- Exfiltrate **Confidential Secrets** present in the pipeline
Challenges

- Capture Workflow’s **semantics and execution flow**
- Track **dataflow** across workflows and the actions
- Support multiple **programming languages**
- Predict the potential **impact** of identified vulnerabilities
Our Solution!

ARGUS

Staged Static Taint Tracking tool for Github Workflows and Actions
Argus

Workflow IR Generation

Static Taint Analysis

Impact Classifier

Action Taint Summaries

Design
Generating Taint Summaries (Actions)
IR Generation

```
"build_step1" {
    exec {type: "gh_action", name: "action/checkout@v2"}
    execution_id: 0
    args {name: "token", value: "${{ secrets.GITHUB_TOKEN }}"}
    environment {}
    CIvars {name: "GITHUB_TOKEN", type: "secrets", ref:arg1 }
}
"build-proj" {
    exec {type: "shell_cmd", command: "/build.sh"}
    execution_id: 1
    args {}
    environment {name: "CFLAGS", value: "-Wall"}
    CIvars {}
}
```
Workflow Dependency Graph

Design

3

Workflow Dependency Graph

Reusable workflow.yml

Job: build $j_1$

Job: test $j_1$

Job: deploy $j_3$

Job: notify $j_4$

Workflow.yml

Job: build $j_1$

Job: scan $j_2$

Success() Always()
**Impact Classifier**

**High Impact**
- Access to R/W GITHUB_TOKEN or Secrets
- No Maintainer Interaction Required

**Medium Impact**
- Access to R/W GITHUB_TOKEN or Secrets
- Requires Maintainer Interaction

**Low Impact**
- No Access to R/W GITHUB_TOKEN or Secrets

- Opens an issue
- Profit

- Creates a Pull Request
- Maintainer merges the PR
- Profit

- Creates a Pull Request
- Profit
Case Study

name: Issue Workflow
on:
  issues:
    types: [opened, edited]
jobs:
  # This job will check the issue to determine if it should be moved to a different repository
  redirectIssue:
    name: Check for issue transfer
    env:
      content_analysis_response: undefined

Dynamo Public
Open Source Graphical Programming for Design

C# 1,434 586 379 (3 issues need help) 36 Updated 22 minutes ago

- name: Check Information
  id: check-info
  run:
    - echo "content_analysis_response=$(pwsh .\github\scripts\title_analyzer.ps1 "{{steps.remove_quotations.outputs.replaced}}")" >> $GITHUB_ENV
    - name: Label issue
      if: env.content_analysis_response != 'Valid'
      run:
        - curl -v -u admin:${{ secrets.DYNAMOBOTOKEN }} -d '"labels": ["{{env.content_analysis_response}}"]' ${{ github.event.issue.url }}/labels
name: Issue Workflow

on:
  issues:
    types: [opened, edited]

jobs:
  #This job will check the issue to determine if it should be moved to a different repository
  redirectIssue:
    name: Check for issue transfer
    env:
      content_analysis_response: undefined
Case Study

Taint Summary for replace-string-action @ v1.2

<table>
<thead>
<tr>
<th>Input Name</th>
<th>Sinks</th>
<th>Output Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>N/A</td>
<td>replaced</td>
</tr>
</tbody>
</table>

- uses: actions/checkout@v2
- name: Remove conflicting chars
  env:
    ISSUE_TITLE: ${github.event.issue.title}
  uses: frabert/replace-string-action@v1.2
  id: remove_quotations
  with:
    pattern: "\\"""\n    string: "${env.ISSUE_TITLE}"
    replace-with: '-'
- name: Check Information
  id: check-info
  run: |
  echo "content_analysis_response=$(pwshe .\.github\scripts\title_analyzer.ps1 "\${steps.remove_quotations.outputs.replaced }")" >> $GITHUB_ENV
- name: Label issue
  if: env.content_analysis_response != 'Valid'
  run: |
  curl -v -u admin:${{ secrets.DYNAMOBOTTOKEN }} -d '{"labels": ["${env.content_analysis_response}"],}' ${{ github.event.issue.url }}/labels
Evaluation

Accuracy and Precision of taint tracking on Actions

Effectiveness in identifying security vulnerabilities

Accuracy and Precision of taint tracking on Workflows

Comparative evaluation with existing state-of-the-art tools
Evaluation Dataset

Real World Dataset
2.8 million Workflows
1 million Repos

Vuln Bench
Collected a set of 24 previously reported vulnerable workflows

Breakdown of Workflows and Actions in the Dataset

<table>
<thead>
<tr>
<th>No. of Workflows</th>
<th>No. of Repos</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Type</td>
</tr>
<tr>
<td>2,778,483</td>
<td>1,014,819</td>
<td>Javascript</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Composite</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Docker</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
</tr>
</tbody>
</table>
1. **Taint Analysis on Actions**

### Input Flow

```javascript
const property = core.getInput('property');
console.log('property: ${property}');

const value = core.getInput('value');
console.log('value: ${value}');

exec('grep -r "^[#]*$\{\property\}={.*}$\{\path\}"',
    (grepError) => {
        if (grepError !== null) {
            ...
        } else {
            exec(`sed -Ir "s/^[#]*$\{\property\}={.*}$\{\path\}"`
                , (error, stderr) => {
                    ...
                });
    });
```

*Reedyuk/write-properties*

### Direct Flow

```javascript
function renderMark() {
    return **${context.payload.action === 'reopened' ?
        context.payload.sender.login + 'issue': 'issue'/**
    > **${context.payload.issue.title}
    ...
}

const markdownString = renderMark();
exec(`
    `curl ${wxhook} \n    -H 'Content-Type: application/json' \n    -d '{
      "... "markdown": {
        "content": "${markdownString.replaceAll('"', '\"')}"
      }
    }';
```

*94dreamer/create-report*
## Taint Analysis on Actions

### Precision of Taint Analysis by ARGUS on Actions

<table>
<thead>
<tr>
<th>Type</th>
<th>Javascript</th>
<th>Composite</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True Positives</td>
<td>False Positives</td>
<td>Precision</td>
</tr>
<tr>
<td>Input Flow</td>
<td>138</td>
<td>10</td>
<td>93.2%</td>
</tr>
<tr>
<td>Direct Flow</td>
<td>27</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>Cumulative</td>
<td>175</td>
<td>10</td>
<td>94.2%</td>
</tr>
</tbody>
</table>

**Precision:**
- **Javascript Actions:** 94%
- **Composite Actions:** 96%
- **Unique Direct Flow Actions:** 80%
Evaluation

2 Taint Analysis on Workflows

27,489 Vulnerable Workflows Identified
3,643 High Impact Vulnerabilities
ALL VulnBench Workflows Identified
### Vulnerability Identification

#### Severity Assignment of Vulnerabilities using the Impact Classifier

<table>
<thead>
<tr>
<th>Flow Type</th>
<th>No. of Workflows</th>
<th>Num. Repos</th>
<th>Direct Flow Actions</th>
<th>Input Flow Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>High (Total: 3,643)</td>
<td>Medium (Sampled: 1,000)</td>
<td>Low (Sampled: 1,000)</td>
<td>Total (Expected: 5,643)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intra-WF</td>
<td>2,875</td>
<td>467</td>
<td>769</td>
<td>4,111</td>
</tr>
<tr>
<td>Inta-WF-Ac</td>
<td>787</td>
<td>597</td>
<td>287</td>
<td>1,671</td>
</tr>
<tr>
<td>Total</td>
<td>3,322 (91.18%)</td>
<td>985 (98.5%)</td>
<td>991 (99.1%)</td>
<td>5,298 (93.88%)</td>
</tr>
</tbody>
</table>

**Precision in finding vulnerabilities**: 93%

**Zero day vulnerabilities found**: 5298
### Comparative Evaluation

Comparative Evaluation of ARGUS with other state-of-the-art works in finding Code Injection Vulnerabilities

<table>
<thead>
<tr>
<th>Tool</th>
<th>High/Medium</th>
<th></th>
<th></th>
<th></th>
<th>Low</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP</td>
<td>FP</td>
<td>FN</td>
<td>P</td>
<td></td>
<td>TP</td>
<td>FP</td>
<td>FN</td>
</tr>
<tr>
<td>GHAST</td>
<td>744</td>
<td>157</td>
<td>3,563</td>
<td>82.6%</td>
<td></td>
<td>331</td>
<td>363</td>
<td>660</td>
</tr>
<tr>
<td>GITSEC</td>
<td>1,527</td>
<td>53</td>
<td>2,870</td>
<td>96.6%</td>
<td></td>
<td>204</td>
<td>3</td>
<td>787</td>
</tr>
<tr>
<td>ARGUS</td>
<td>4,307</td>
<td>336</td>
<td>0</td>
<td>92.8%</td>
<td></td>
<td>991</td>
<td>9</td>
<td>0</td>
</tr>
</tbody>
</table>

7x More vulnerabilities discovered Compared to State-of-the-art
Conclusion

Responsible Disclosure

Four tips to keep your GitHub Actions workflows secure

Researchers from Purdue and NCSU have found a large number of command injection vulnerabilities in the workflows of projects on GitHub. Follow these four tips to keep your GitHub Actions workflows secure.

150+ Vulnerable Workflows Fixed

20 Security Advisories
Conclusion

- Introduced ARGUS, the first static taint analysis system for GitHub Actions
- Our system can track taint across Workflows and Actions
- Conducted a Large scale evaluation of over 2.8million workflows identifying critical vulnerabilities

Website: secureci.org[argus]
Code: github.com/purs3lab/argus