Learnings from scanning 2 million hosts daily for Bug Bounty

VIDOC
Security lab

1 whoami

Dawid Moczadło

- Co-founder of Vidoc Security Lab
- Bug bounty hunter (ex-Top 1 in Poland)
- CTF player for P4
- Climber and coffee lover

2 Story

We made

$120 000

in bug bounty using only automation

facebook PayPal

SAMSUNG GitLab

Microsoft SONY

amazon verizon media

Spotify shopify

3 Scale

- **1300** bug bounty programs (public and private)
- **3000** root domains
- **2** million unique web servers daily
- **100+** million HTTP requests daily
- **200+** security issues detected weekly

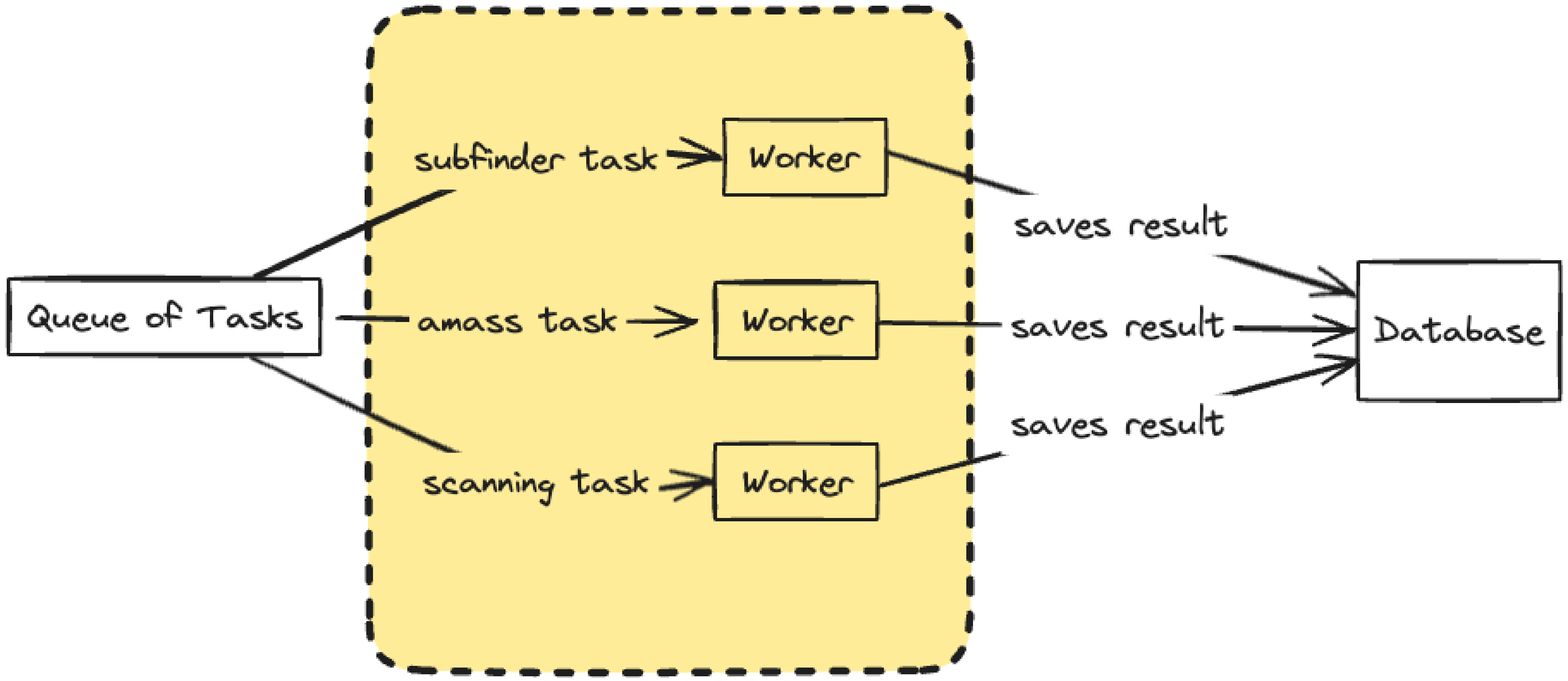
4 Initial goal

We want to automate our bug bounty and do it **cheap ($$)**

4 First try, let's use open source

- Amass + Subfinder
- Nmap
- Nuclei
- Kubernetes
- Google Cloud

Pool of 20-30 workers



4 ...and it failed

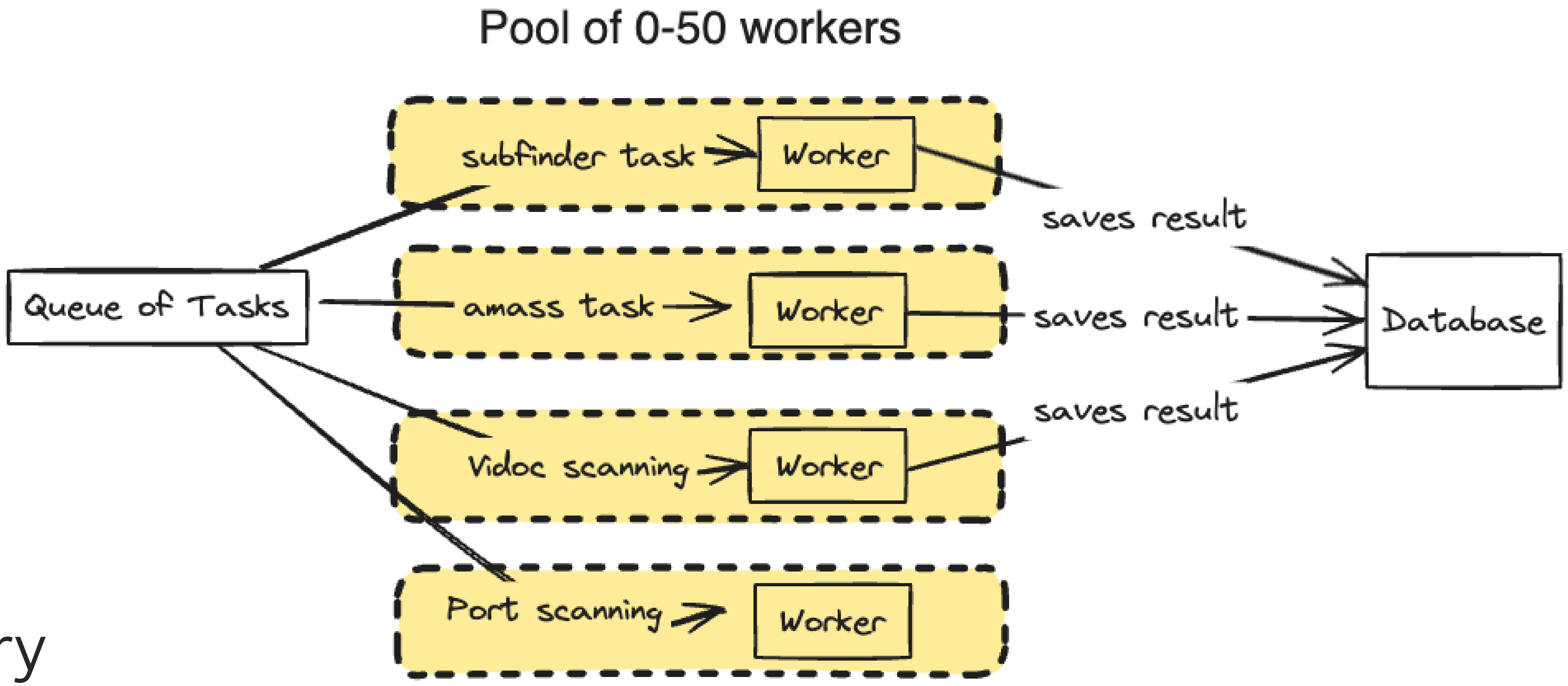
- Nuclei, Amass, and Nmap **were turbo slow**
- **Workers were too expensive** - they worked all the time
- The database was overloaded and too slow
- Networking costs of Google Cloud were too high

5 Next iteration - goals

- Speed and accuracy
- Scanning only 3000 of the most popular ports
- Decreasing costs

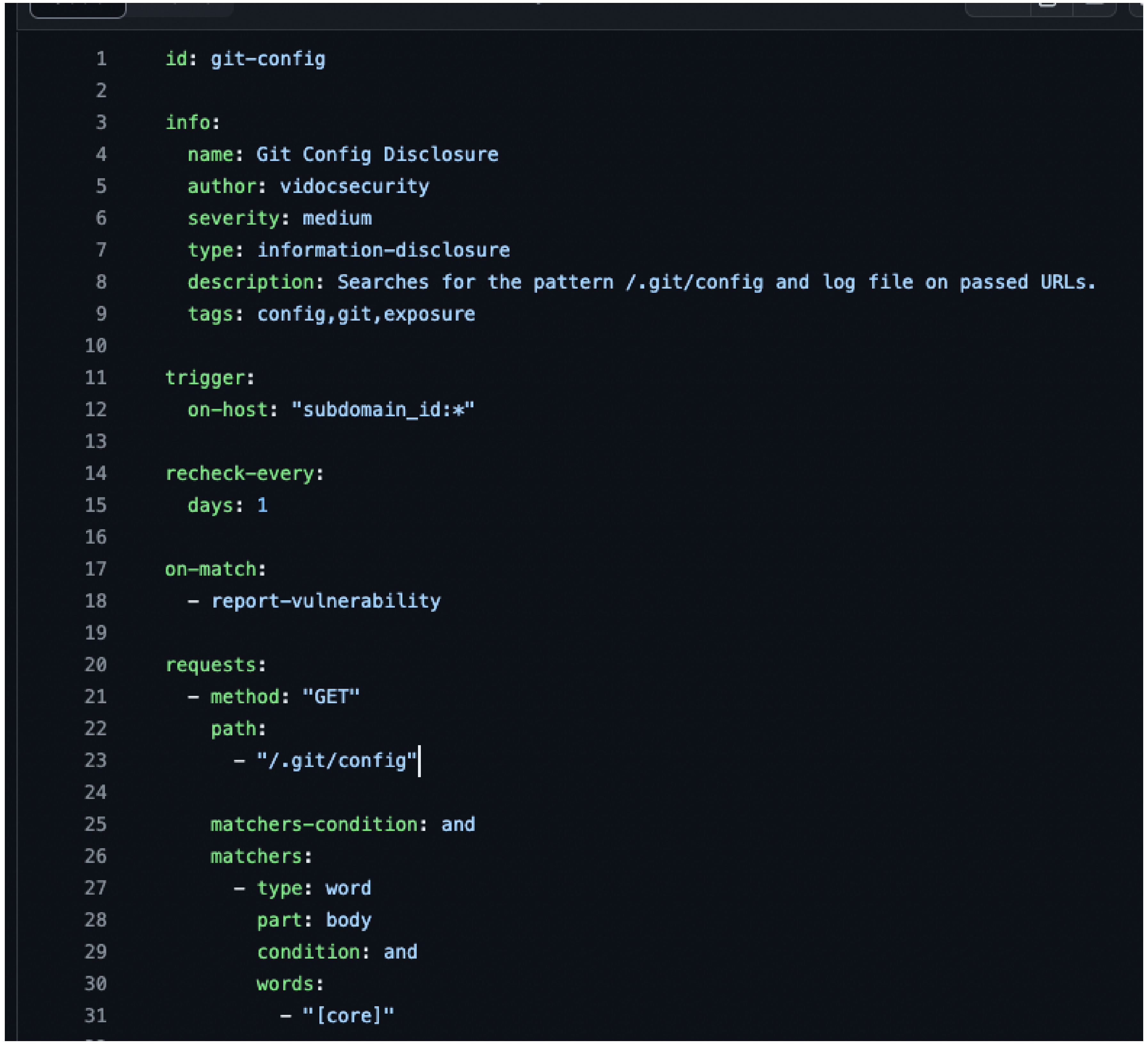
6 Next iteration

- Amass + Subfinder
- ~~Nmap~~ Masscan
- ~~Nuclei~~ Custom Scanning Engine
- Kubernetes
- ~~Google Cloud~~ Digitalocean
- Automatic scaling 0 machines
- IP Rotation (for every scan)
- NoSQL DB



7 Custom Scanning Engine

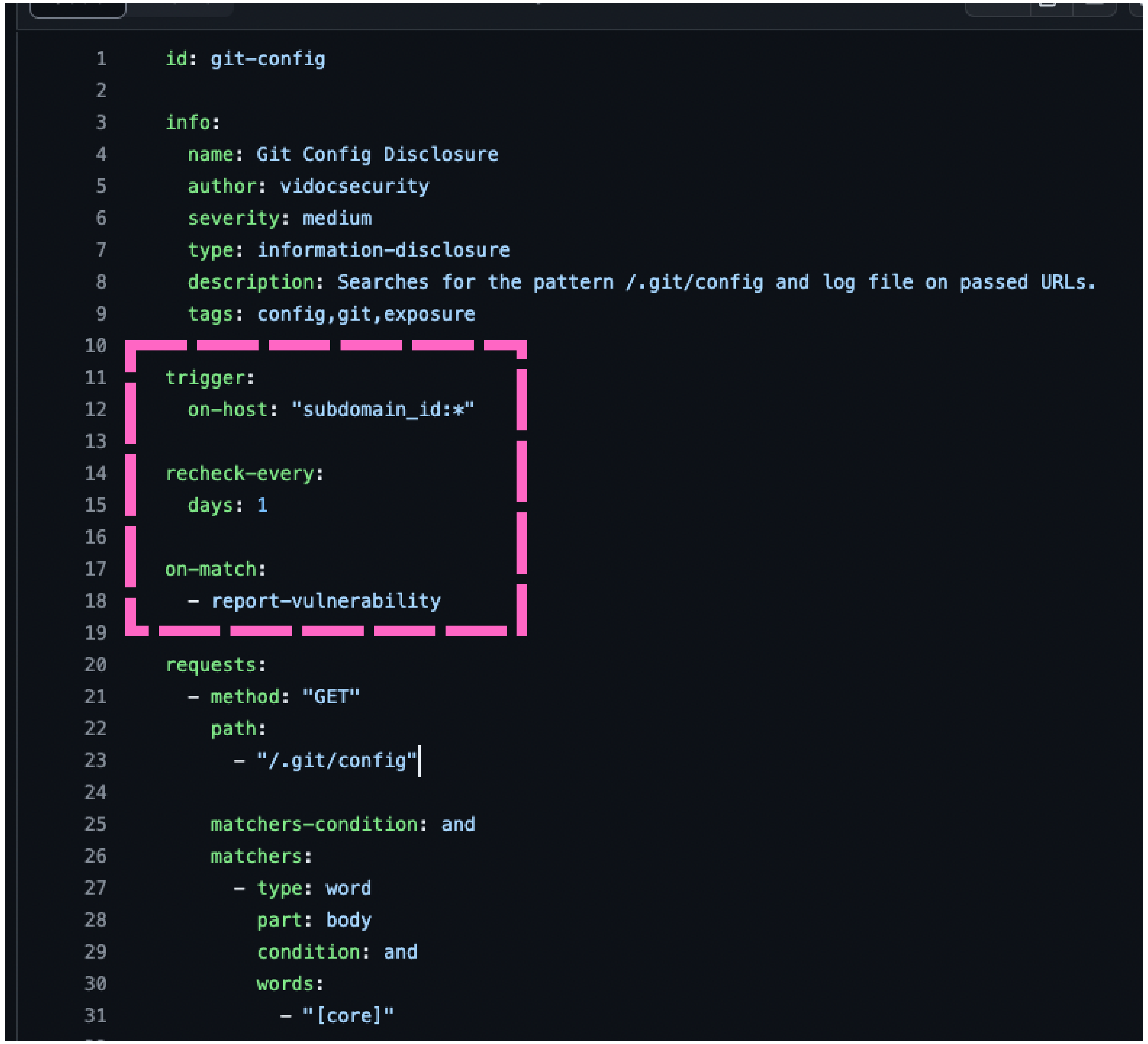
- Optimized for big-scale
- We took everything good from Nuclei and **we made it better**
- Speed >>>



```yaml
id: git-config

info:
  name: Git Config Disclosure
  author: vidocsecurity
  severity: medium
  type: information-disclosure
  description: Searches for the pattern /.git/config and log file on passed URLs.
  tags: config,git,exposure

trigger:
  on-host: "subdomain_id:*"

recheck-every:
  days: 1

on-match:
  - report-vulnerability

requests:
  - method: "GET"
    path:
      - "/.git/config"

    matchers-condition: and
    matchers:
      - type: word
        part: body
        condition: and
        words:
          - "[core]"
```

8 Custom Scanning Engine

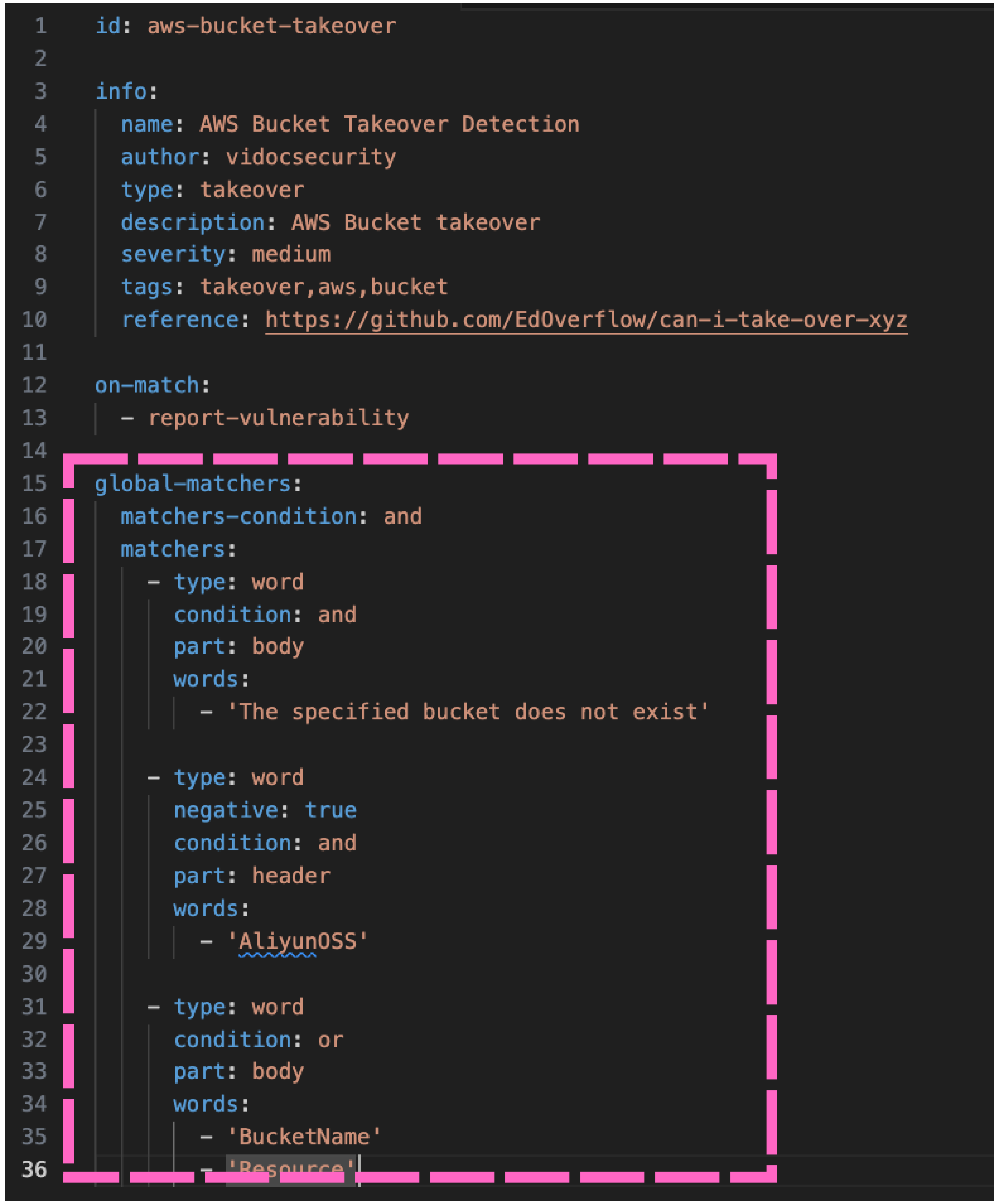
- Scheduled scanning
- Defining query to match targets (only specific technology)



```yaml
id: git-config

info:
  name: Git Config Disclosure
  author: vidocsecurity
  severity: medium
  type: information-disclosure
  description: Searches for the pattern /.git/config and log file on passed URLs.
  tags: config,git,exposure

trigger:
  on-host: "subdomain_id:*"

recheck-every:
  days: 1

on-match:
  - report-vulnerability

requests:
  - method: "GET"
    path:
      - "/.git/config"

    matchers-condition: and
    matchers:
      - type: word
        part: body
        condition: and
        words:
          - "[core]"
```

9 Custom Scanning Engine

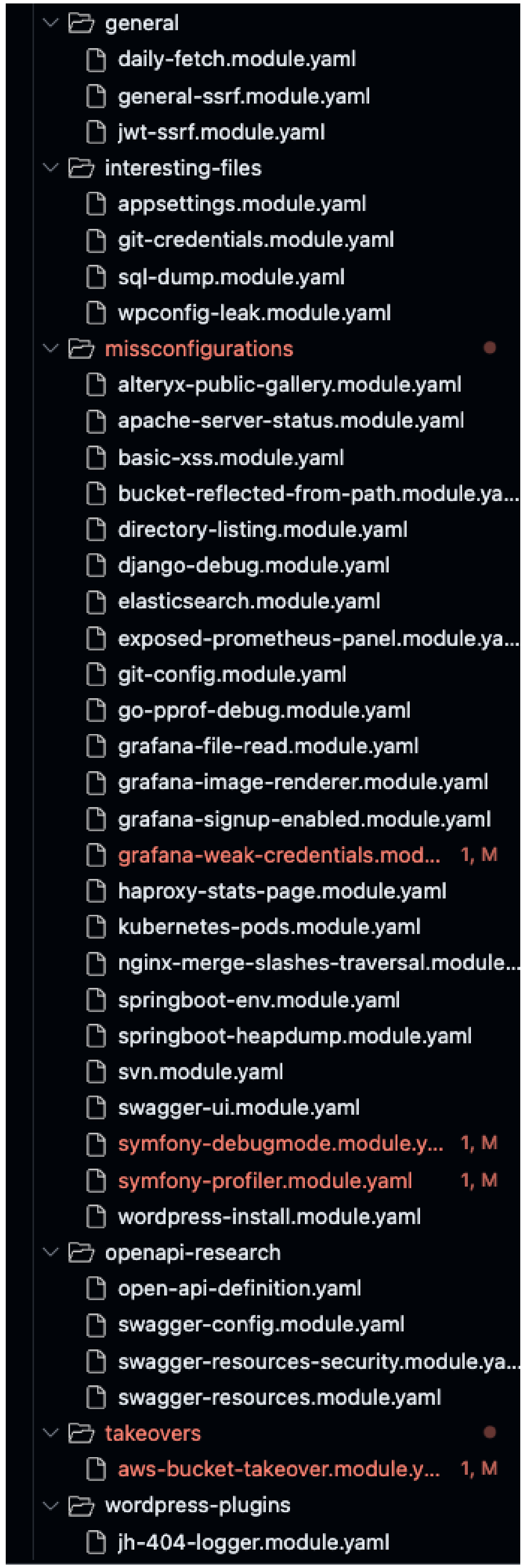
- Global Matchers
  - Match on responses from different modules
  - Passive Modules
  - Like a grep for all responses
- Great for finding weird bugs
  - Bucket takeovers
  - DB errors



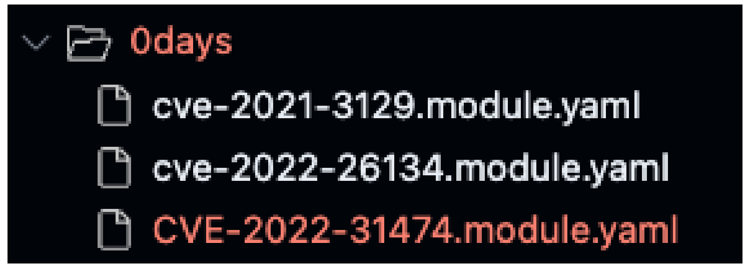
```yaml
id: aws-bucket-takeover

info:
  name: AWS Bucket Takeover Detection
  author: vidocsecurity
  type: takeover
  description: AWS Bucket takeover
  severity: medium
  tags: takeover,aws,bucket
  reference: https://github.com/EdOverflow/can-i-take-over-xyz

on-match:
  - report-vulnerability

global-matchers:
  matchers-condition: and
  matchers:
    - type: word
      condition: and
      part: body
      words:
        - 'The specified bucket does not exist'

    - type: word
      negative: true
      condition: and
      part: header
      words:
        - 'AliyunOSS'

    - type: word
      condition: or
      part: body
      words:
        - 'BucketName'
        - 'Resource'
```

10 Our scanning approach

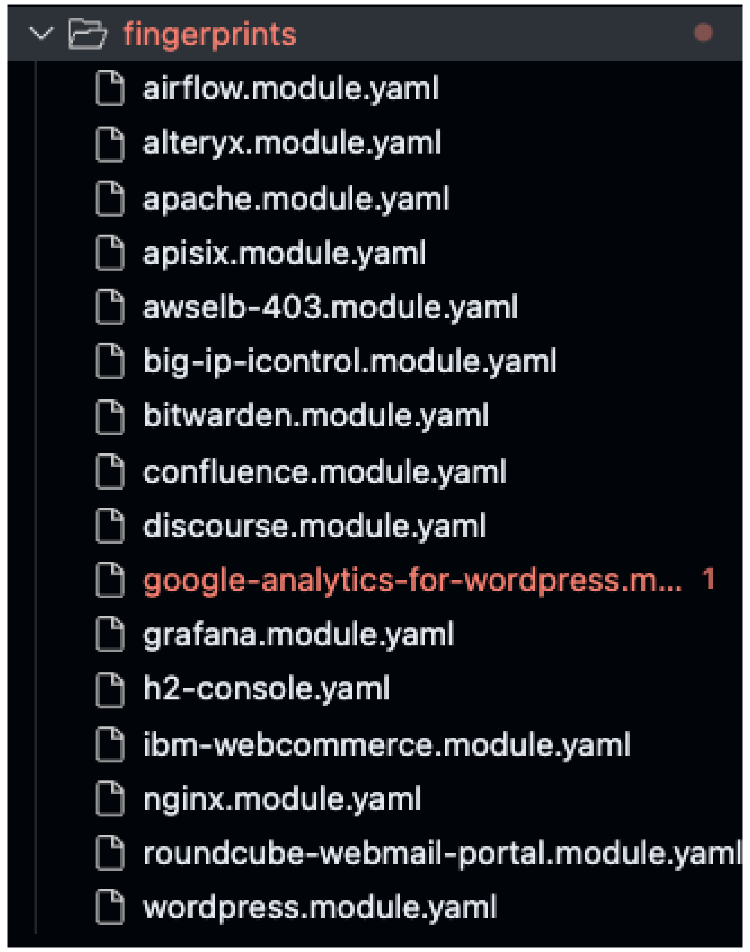
- Write **custom modules or edit existing ones**
- Created/edited 71 modules in one year
- Used max 30 modules
- Collaborate with others!
- **Scan for the same bugs, over and over...**



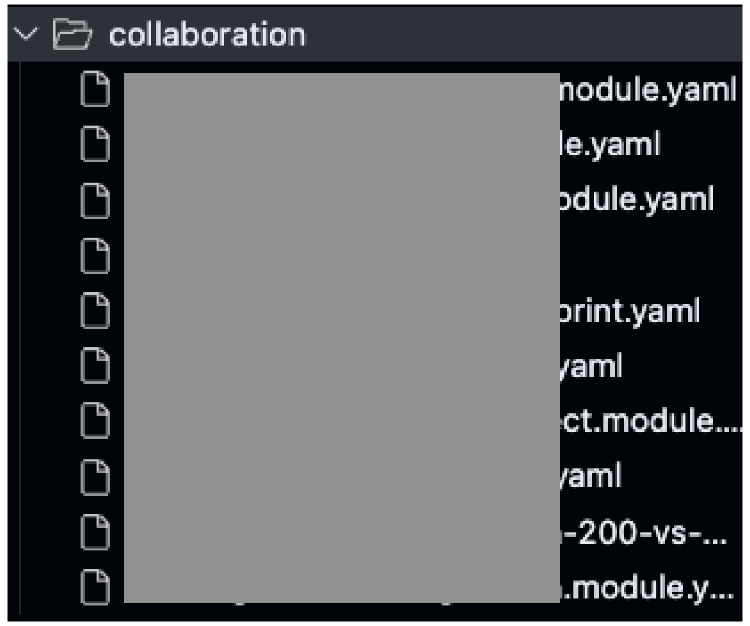
```
general
  daily-fetch.module.yaml
  general-ssrf.module.yaml
  jwt-ssrf.module.yaml
interesting-files
  appsettings.module.yaml
  git-credentials.module.yaml
  sql-dump.module.yaml
  wpconfig-leak.module.yaml
missconfigurations
  alteryx-public-gallery.module.yaml
  apache-server-status.module.yaml
  basic-xss.module.yaml
  bucket-reflected-from-path.module.ya...
  directory-listing.module.yaml
  django-debug.module.yaml
  elasticsearch.module.yaml
  exposed-prometheus-panel.module.ya...
  git-config.module.yaml
  go-pprof-debug.module.yaml
  grafana-file-read.module.yaml
  grafana-image-renderer.module.yaml
  grafana-signup-enabled.module.yaml
  grafana-weak-credentials.mod...     1, M
  haproxy-stats-page.module.yaml
  kubernetes-pods.module.yaml
  nginx-merge-slashes-traversal.module...
  springboot-env.module.yaml
  springboot-heapdump.module.yaml
  svn.module.yaml
  swagger-ui.module.yaml
  symfony-debugmode.module.y...       1, M
  symfony-profiler.module.yaml        1, M
  wordpress-install.module.yaml
openapi-research
  open-api-definition.yaml
  swagger-config.module.yaml
  swagger-resources-security.module.ya...
  swagger-resources.module.yaml
takeovers
  aws-bucket-takeover.module.y...     1, M
wordpress-plugins
  jh-404-logger.module.yaml
```



```
0days
  cve-2021-3129.module.yaml
  cve-2022-26134.module.yaml
  CVE-2022-31474.module.yaml
```



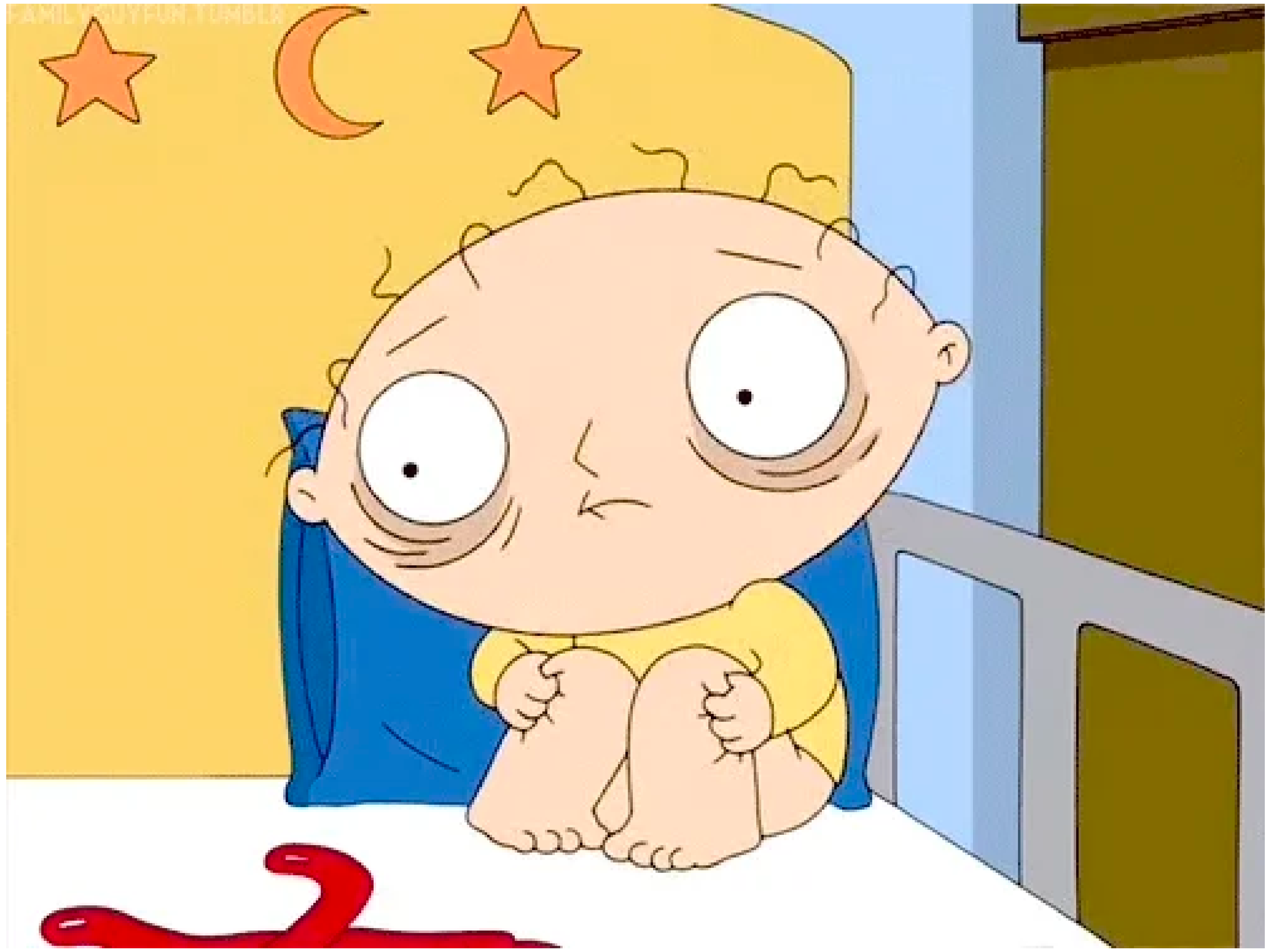
```
fingerprints
  airflow.module.yaml
  alteryx.module.yaml
  apache.module.yaml
  apisix.module.yaml
  awselb-403.module.yaml
  big-ip-icontrol.module.yaml
  bitwarden.module.yaml
  confluence.module.yaml
  discourse.module.yaml
  google-analytics-for-wordpress.m...   1
  grafana.module.yaml
  h2-console.yaml
  ibm-webcommerce.module.yaml
  nginx.module.yaml
  roundcube-webmail-portal.module.yaml
  wordpress.module.yaml
```



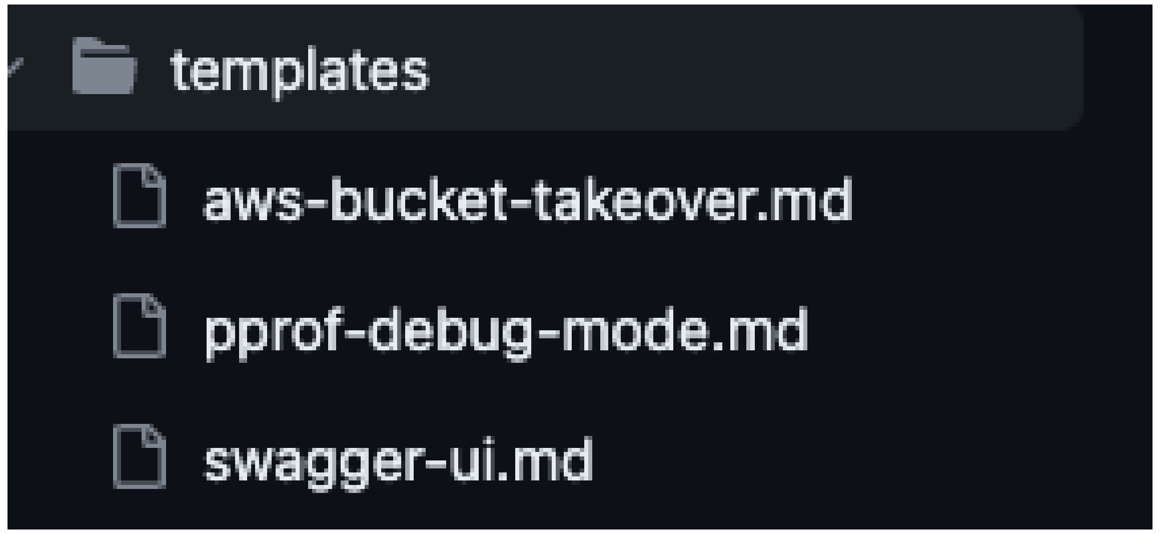
```
collaboration
  [redacted]
```

11 We are finding bugs! Too many bugs

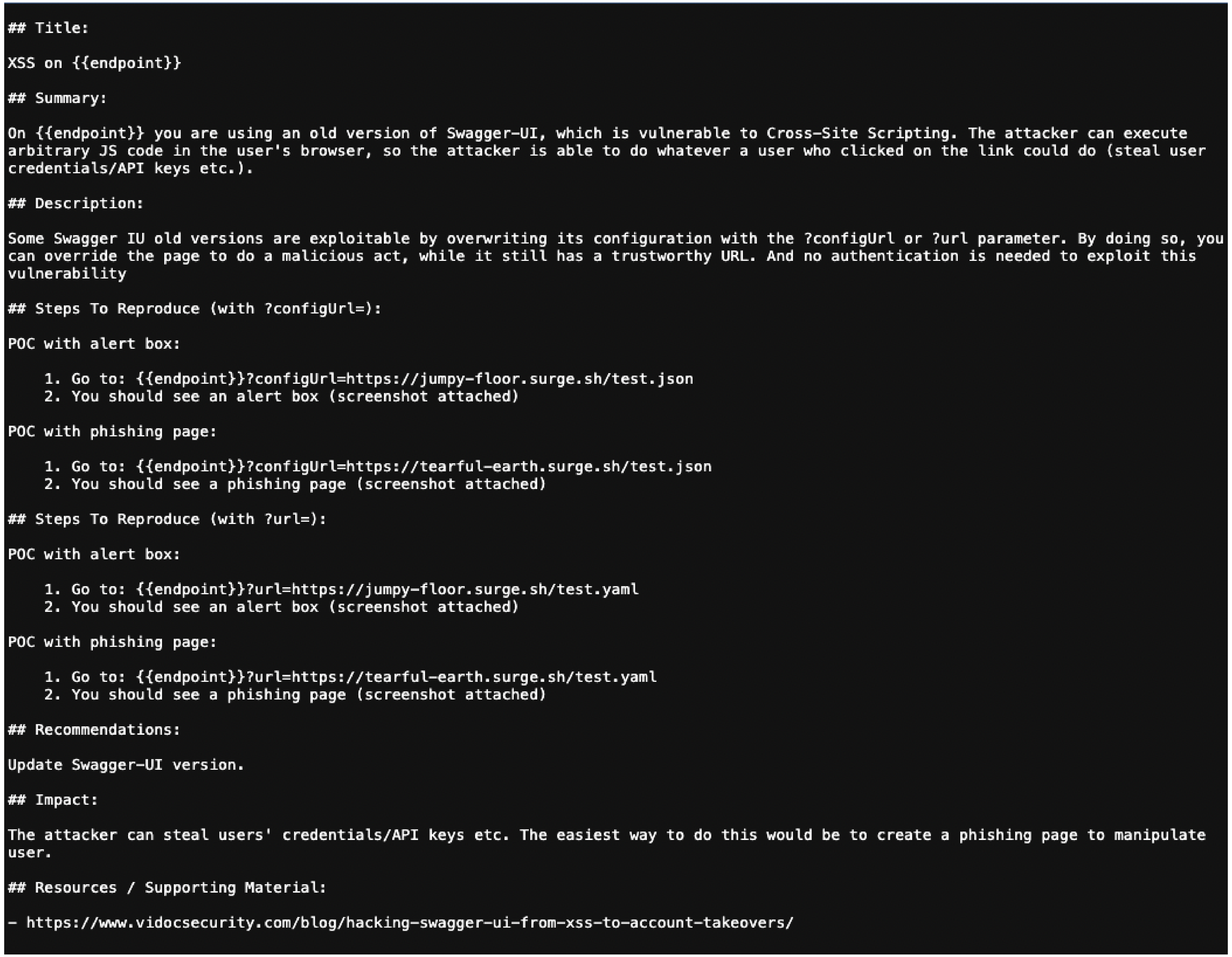
- **200+** security issues detected weekly
- Being too slow == **duplicate**
- Manual escalation for better payouts **$$**
- Unpredictable revenue



12 Semi-automatic reporting



templates
aws-bucket-takeover.md
pprof-debug-mode.md
swagger-ui.md



```
## Title:

XSS on {{endpoint}}

## Summary:

On {{endpoint}} you are using an old version of Swagger-UI, which is vulnerable to Cross-Site Scripting. The attacker can execute
arbitrary JS code in the user's browser, so the attacker is able to do whatever a user who clicked on the link could do (steal user
credentials/API keys etc.).

## Description:

Some Swagger IU old versions are exploitable by overwriting its configuration with the ?configUrl or ?url parameter. By doing so, you
can override the page to do a malicious act, while it still has a trustworthy URL. And no authentication is needed to exploit this
vulnerability

## Steps To Reproduce (with ?configUrl=):

POC with alert box:

    1. Go to: {{endpoint}}?configUrl=https://jumpy-floor.surge.sh/test.json
    2. You should see an alert box (screenshot attached)

POC with phishing page:

    1. Go to: {{endpoint}}?configUrl=https://tearful-earth.surge.sh/test.json
    2. You should see a phishing page (screenshot attached)

## Steps To Reproduce (with ?url=):

POC with alert box:

    1. Go to: {{endpoint}}?url=https://jumpy-floor.surge.sh/test.yaml
    2. You should see an alert box (screenshot attached)

POC with phishing page:

    1. Go to: {{endpoint}}?url=https://tearful-earth.surge.sh/test.yaml
    2. You should see a phishing page (screenshot attached)

## Recommendations:

Update Swagger-UI version.

## Impact:

The attacker can steal users' credentials/API keys etc. The easiest way to do this would be to create a phishing page to manipulate
user.

## Resources / Supporting Material:

- https://www.vidocsecurity.com/blog/hacking-swagger-ui-from-xss-to-account-takeovers/
```

13 Key learnings

- Building **good infrastructure** for bug bounty is **turbo-hard** and you will have to maintain it

- **You can’t report everything** - there is not enough time

- Scan for the same bugs every day
  - **Developers constantly change code** and they will make mistakes

- Prioritize only good bug bounty programs - do not waste time reporting to the bad ones

14

How to win with automation in 2023?

- Rescan often for the same bugs - every X hours **(be consistent)**
  - git-config
  - directory-listing
  - exposed elastic search, prom
  - debug mode in Django
  - spring-boot heap dump
  - subdomain takeovers (aws, github...)
  - ...

15 How to win with automation in 2023?

- Find new and unknown misconfigurations or variations **(be smarter)**
  - Edit existing modules
  - Add more paths or params for fuzzing

How to win with automation in 2023?

- Look for new CVEs and develop POCs first **(be faster)**
  - Log4Shell (we made
  - Confluence 0day CVE-2022-26134
  - ...

How to win with automation in 2023?

- Automate scanning on private programs - people usually scan only public programs
    - We had access to ~400 private programs
    - Hackerone, Intigriti, Bugcrowd, Yeswehack, Hackenproof, ...

Thank you!

Email

dawid@vidocsecurity.com

Twitter

twitter.com/kannthu1

Web

https://www.vidocsecurity.com

Read full story how we earned $120 000 using our own product