OFFZONE 2022

Vulnerability Management for Dummies or How to train your automation

Roza Abdullaeva

Leading Information Security Specialist, PS Development

Dmitriy Sherstoboyev

Application Security Engineer, PS Development

Moscow, August 26, 2022

InfraSec part

Just scan	Good scan
Collect target range of IP addresses	Collect target range of IP addresses
Start scan	Start scan
* Download summary report	Explore results and findings
	Create tickets to fix / Fix
	Check fixing
	* Download summary report

* For some audits

Useful tools for good scan process

- Vulnerability scanner
- Inventory system
- Task manager
- Messenger / Mail

Our case:

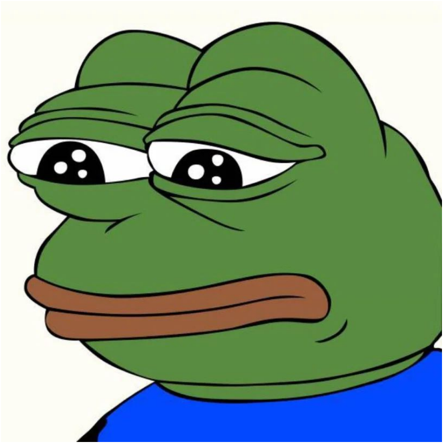
- Vulnerability scanner => Nessus Professional and Nessus Agents linked to Tenable.io
- Inventory system => GLPI*
- Task manager => Cloud Jira
- Messenger / Mail => Slack

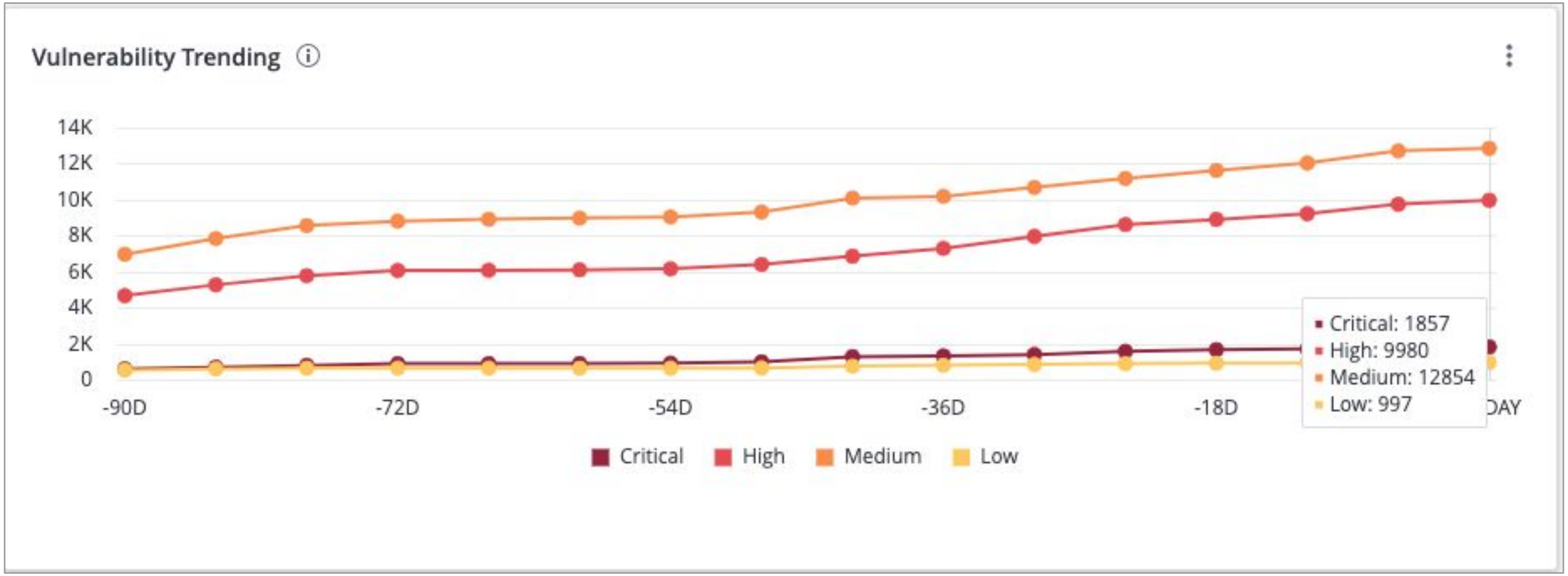
* https://glpi-project.org/

First scan results

500 servers

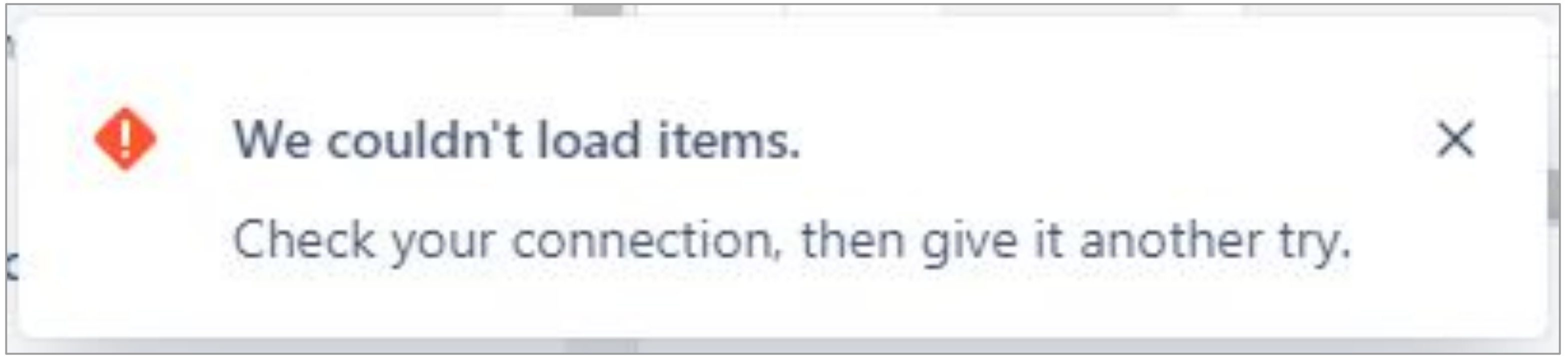
10 000+ high and critical vulnerabilities





First attempts of automation

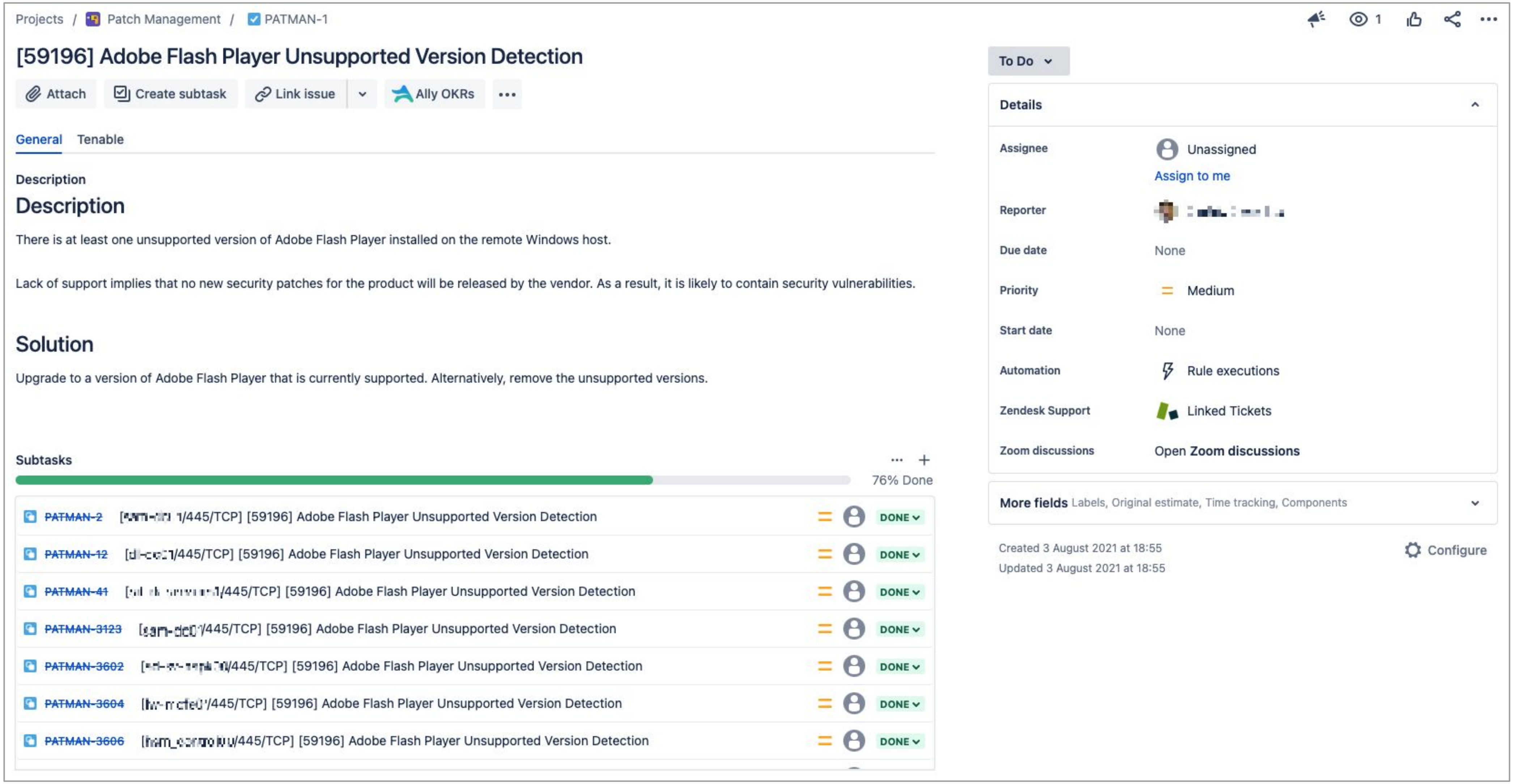
https://github.com/tenable/integration-jira-cloud



We couldn't load items.

Check your connection, then give it another try.

Tenable integration for Jira Cloud



Projects / Patch Management / PATMAN-1

[59196] Adobe Flash Player Unsupported Version Detection

Description

Description

There is at least one unsupported version of Adobe Flash Player installed on the remote Windows host.

Lack of support implies that no new security patches for the product will be released by the vendor. As a result, it is likely to contain security vulnerabilities.

Solution

Upgrade to a version of Adobe Flash Player that is currently supported. Alternatively, remove the unsupported versions.

Subtasks

76% Done

- ~~PATMAN-2~~ [[redacted]1/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE
- ~~PATMAN-12~~ [[redacted]1/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE
- ~~PATMAN-41~~ [[redacted]1/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE
- ~~PATMAN-3123~~ [[redacted]/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE
- ~~PATMAN-3602~~ [[redacted]/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE
- ~~PATMAN-3604~~ [[redacted]/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE
- ~~PATMAN-3606~~ [[redacted]/445/TCP] [59196] Adobe Flash Player Unsupported Version Detection — DONE

Details

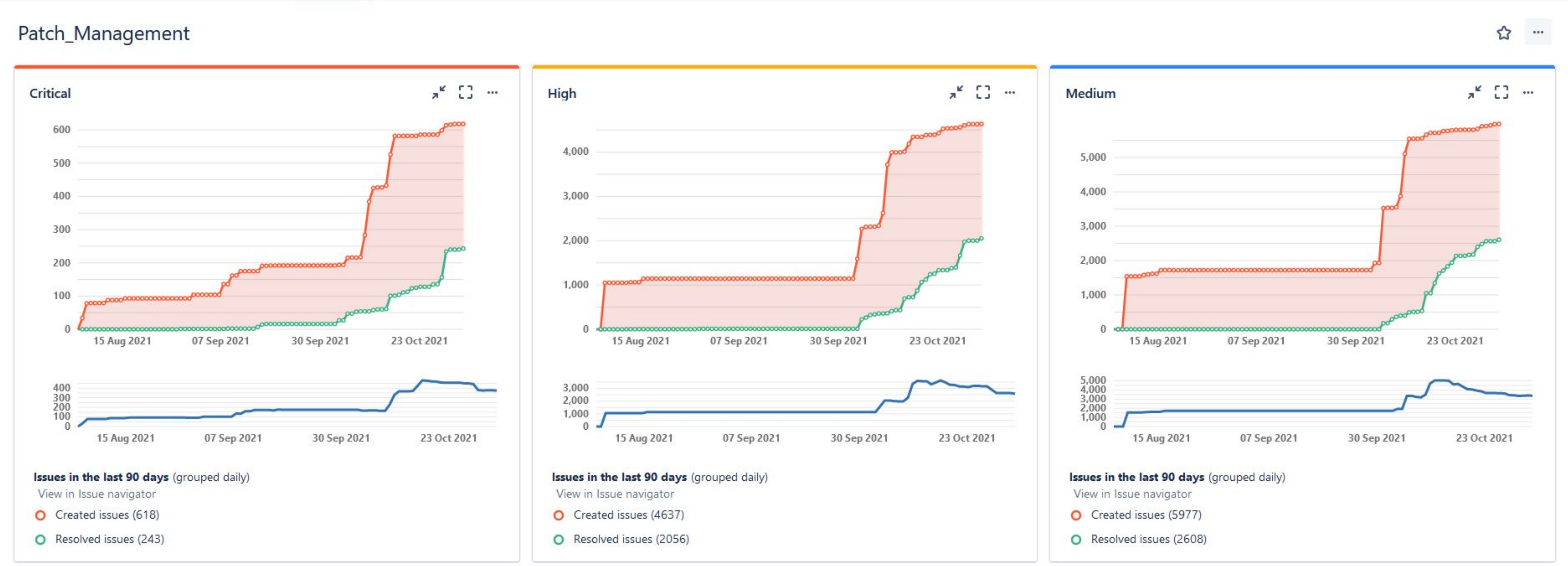
Assignee	Unassigned Assign to me
Reporter	[redacted]
Due date	None
Priority	Medium
Start date	None
Automation	Rule executions
Zendesk Support	Linked Tickets
Zoom discussions	Open Zoom discussions

More fields Labels, Original estimate, Time tracking, Components

Created 3 August 2021 at 18:55
Updated 3 August 2021 at 18:55

Configure

Tenable integration for Jira Cloud



Patch_Management

Critical

Issues in the last 90 days (grouped daily)

View in Issue navigator

○ Created issues (618)

○ Resolved issues (243)

High

Issues in the last 90 days (grouped daily)

View in Issue navigator

○ Created issues (4637)

○ Resolved issues (2056)

Medium

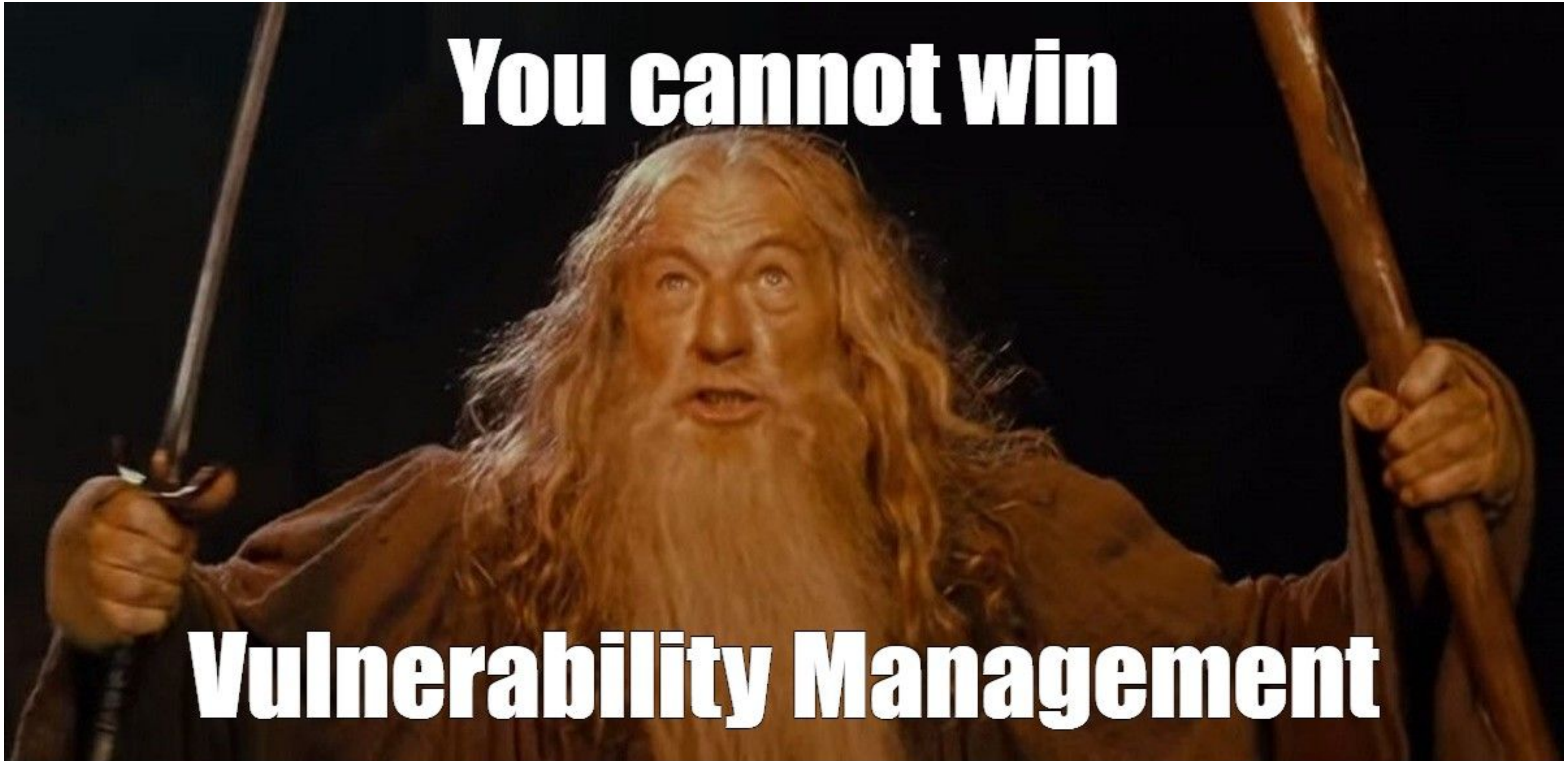
Issues in the last 90 days (grouped daily)

View in Issue navigator

○ Created issues (5977)

○ Resolved issues (2608)

Problems and conclusions



How we can use automation?

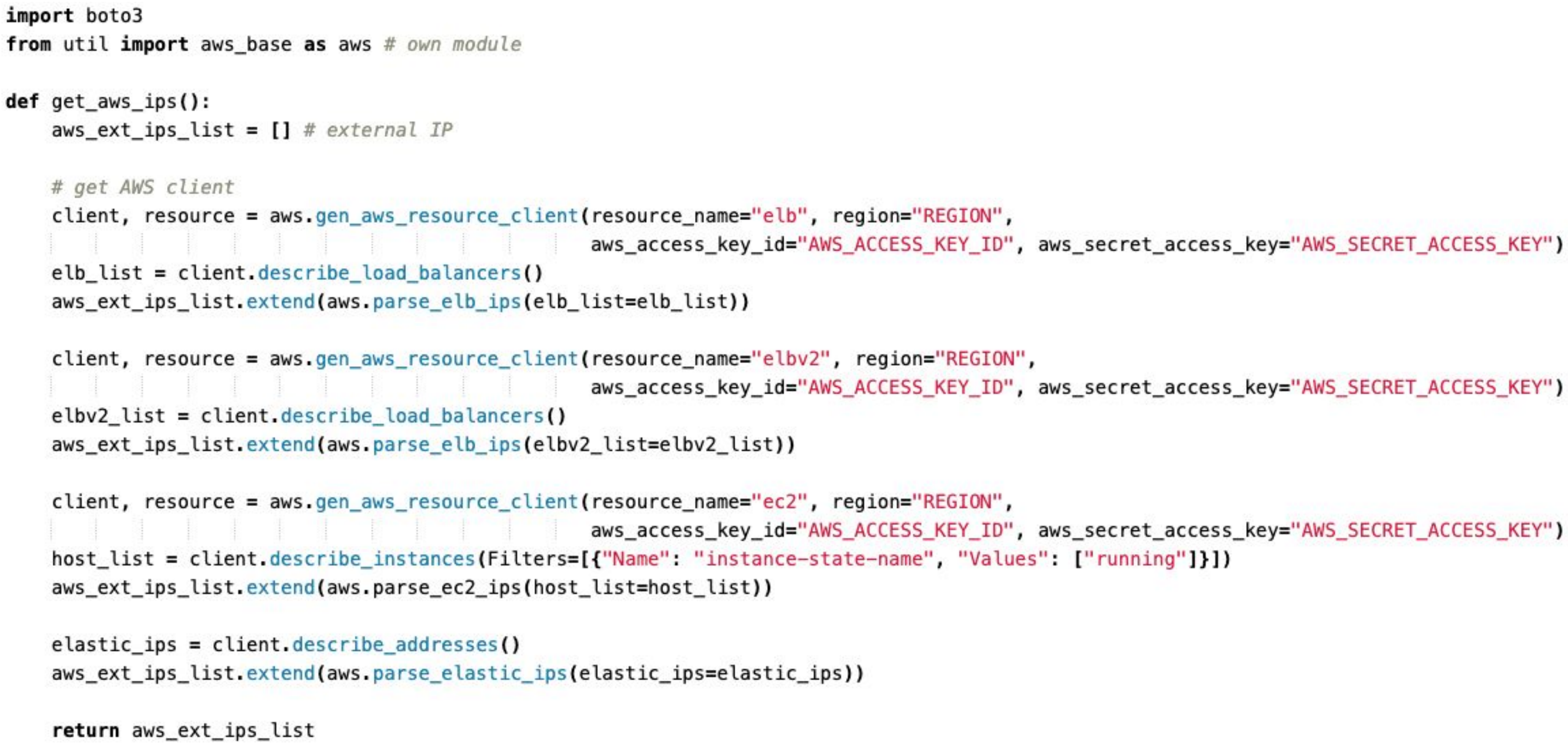
- Python scripts

Good scan	**Automation can be applied**
Collect target range of IP addresses	Collect needed IP range from inventory system or other sources
Start scan	Create scan and/or start it
Explore results and findings	Extract to some analysis platform
Create tickets to fix / Fix	Create remediation tickets to target teams
Check fixing	Check fixing and close fixed findings

How we can use automation?

Automation can be applied	Need to integration with...
Collect needed IP range from inventory system or other sources	**System inventory / cloud / other (VMware, etc.)**
Create scan and/or start it	Scanner
Extract to some analysis platform	Scanner, DefectDojo
Create remediation tickets to target teams	Scanner, Task manager, System inventory
Check fixing and close fixed findings	Scanner, DefectDojo

Integration with cloud

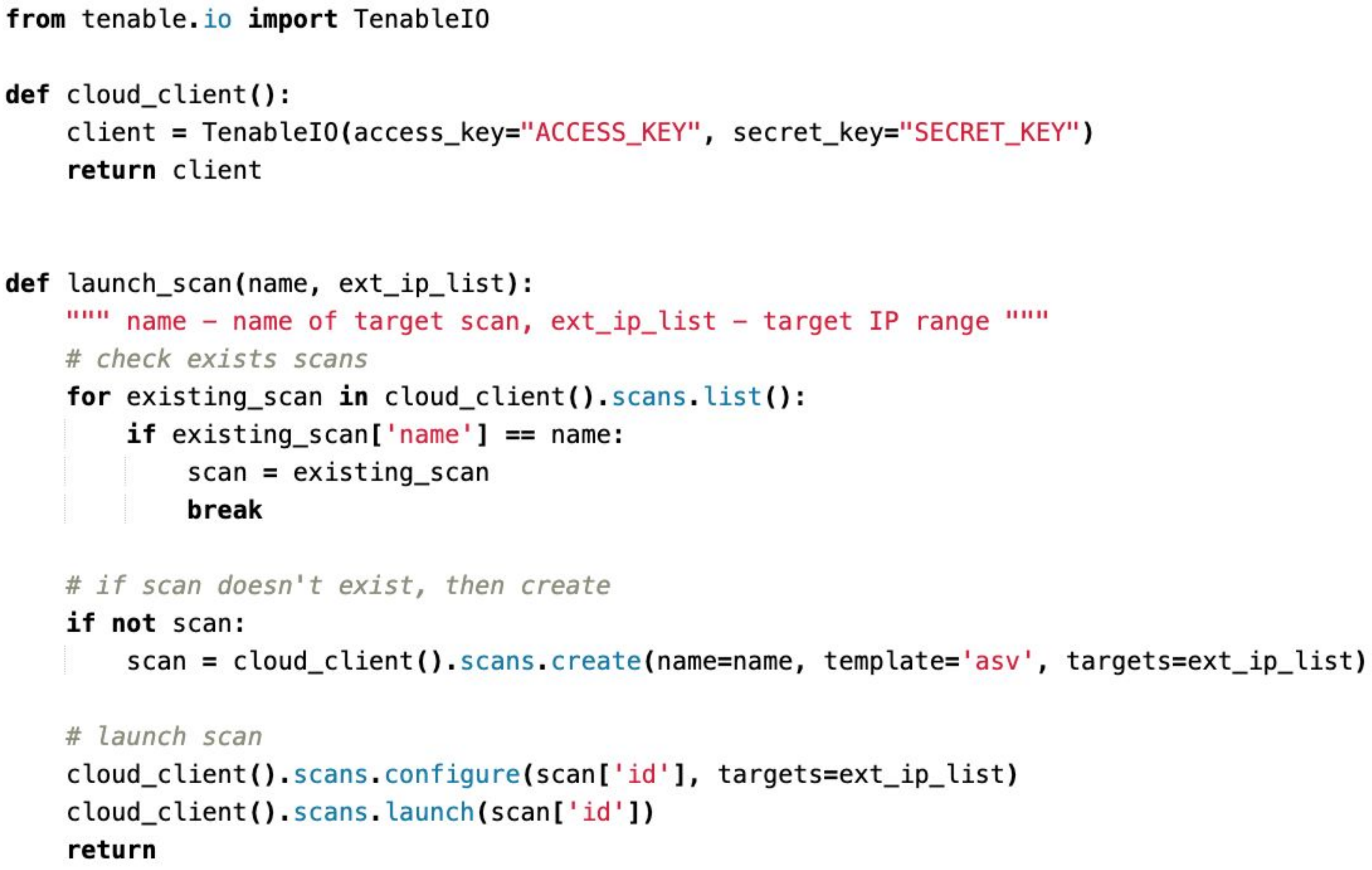


```python
import boto3
from util import aws_base as aws # own module

def get_aws_ips():
    aws_ext_ips_list = [] # external IP

    # get AWS client
    client, resource = aws.gen_aws_resource_client(resource_name="elb", region="REGION",
                                                   aws_access_key_id="AWS_ACCESS_KEY_ID", aws_secret_access_key="AWS_SECRET_ACCESS_KEY")
    elb_list = client.describe_load_balancers()
    aws_ext_ips_list.extend(aws.parse_elb_ips(elb_list=elb_list))

    client, resource = aws.gen_aws_resource_client(resource_name="elbv2", region="REGION",
                                                   aws_access_key_id="AWS_ACCESS_KEY_ID", aws_secret_access_key="AWS_SECRET_ACCESS_KEY")
    elbv2_list = client.describe_load_balancers()
    aws_ext_ips_list.extend(aws.parse_elb_ips(elbv2_list=elbv2_list))

    client, resource = aws.gen_aws_resource_client(resource_name="ec2", region="REGION",
                                                   aws_access_key_id="AWS_ACCESS_KEY_ID", aws_secret_access_key="AWS_SECRET_ACCESS_KEY")
    host_list = client.describe_instances(Filters=[{"Name": "instance-state-name", "Values": ["running"]}])
    aws_ext_ips_list.extend(aws.parse_ec2_ips(host_list=host_list))

    elastic_ips = client.describe_addresses()
    aws_ext_ips_list.extend(aws.parse_elastic_ips(elastic_ips=elastic_ips))

    return aws_ext_ips_list
```

How we can use automation?

Automation can be applied	Need to integration with...
Collect needed IP range from inventory system or other sources	System inventory / cloud / other (VMware, etc.)
Create scan and/or start it	**Scanner**
Extract to some analysis platform	Scanner, DefectDojo
Create remediation tickets to target teams	Scanner, Task manager, System inventory
Check fixing and close fixed findings	Scanner, DefectDojo

Integration with Tenable



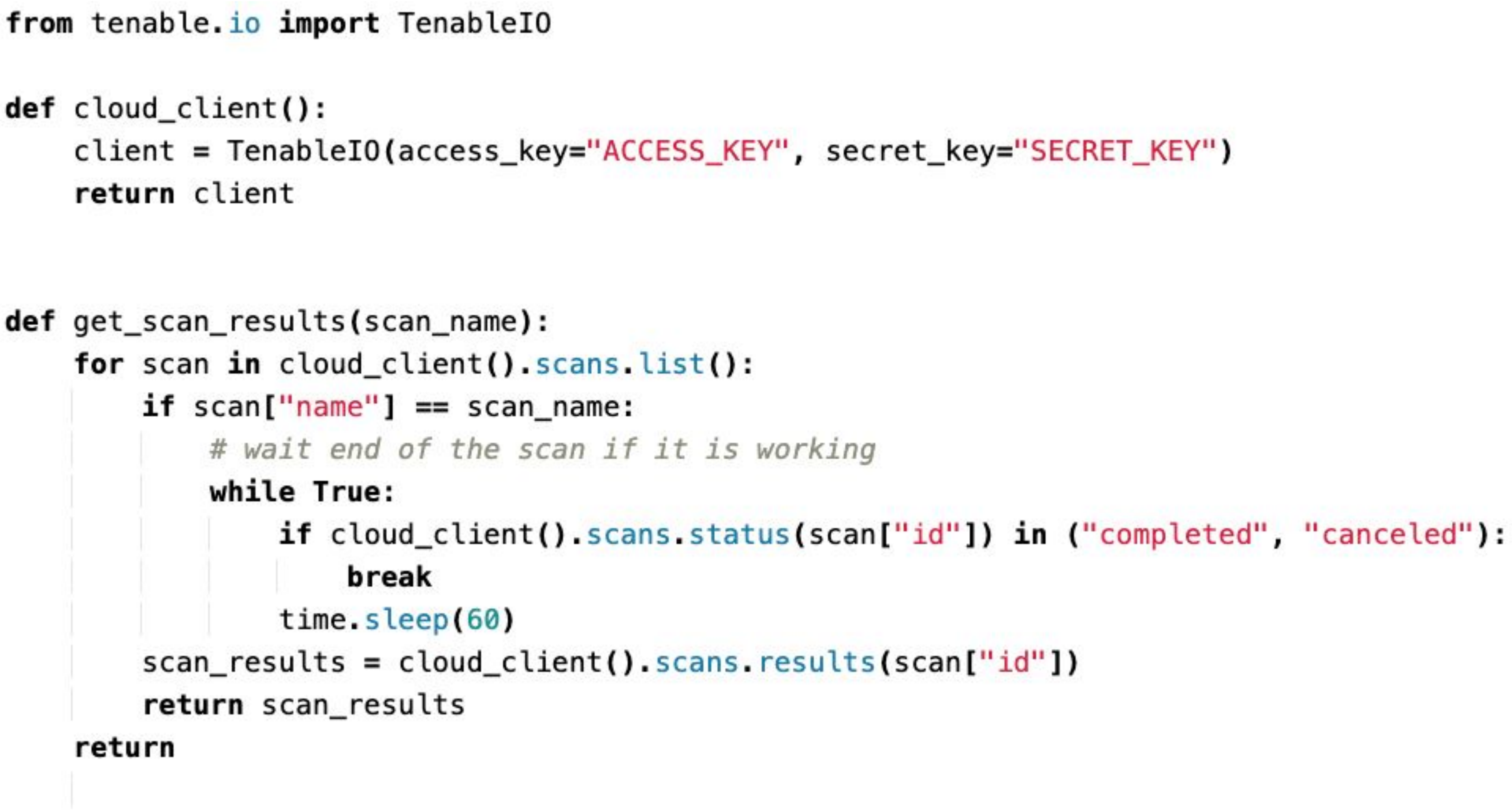
```
from tenable.io import TenableIO

def cloud_client():
    client = TenableIO(access_key="ACCESS_KEY", secret_key="SECRET_KEY")
    return client


def launch_scan(name, ext_ip_list):
    """ name – name of target scan, ext_ip_list – target IP range """
    # check exists scans
    for existing_scan in cloud_client().scans.list():
        if existing_scan['name'] == name:
            scan = existing_scan
            break

    # if scan doesn't exist, then create
    if not scan:
        scan = cloud_client().scans.create(name=name, template='asv', targets=ext_ip_list)

    # launch scan
    cloud_client().scans.configure(scan['id'], targets=ext_ip_list)
    cloud_client().scans.launch(scan['id'])
    return
```

How we can use automation?

Automation can be applied	**Need to integration with...**
Collect needed IP range from inventory system or other sources	System inventory / cloud / other (VMware, etc.)
Create scan and/or start it	Scanner
Extract to some analysis platform	**Scanner, DefectDojo***
Create remediation tickets to target teams	Scanner, Task manager, System inventory
Check fixing and close fixed findings	**Scanner, DefectDojo**

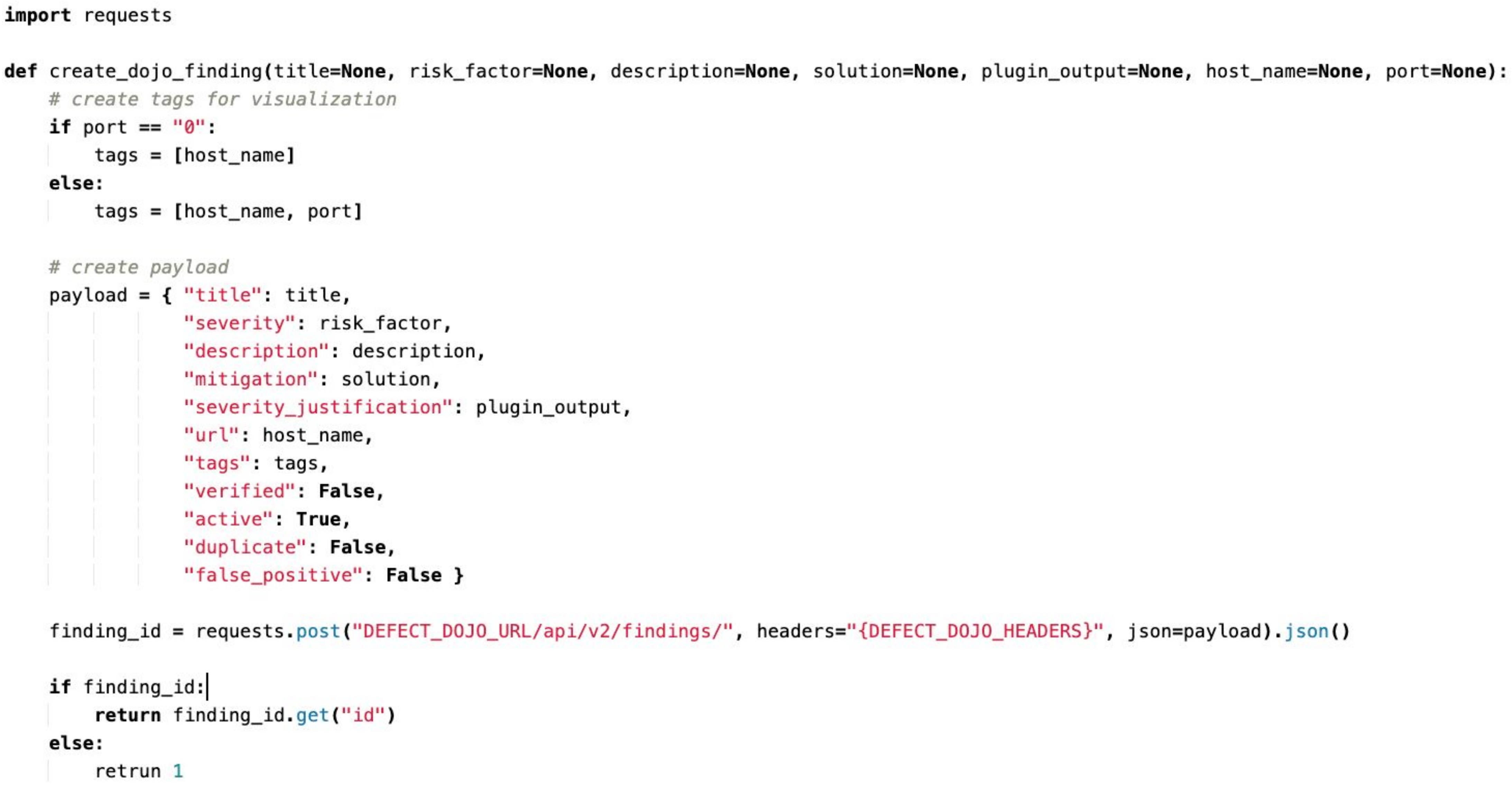
* https://www.defectdojo.org/

Integration with Tenable



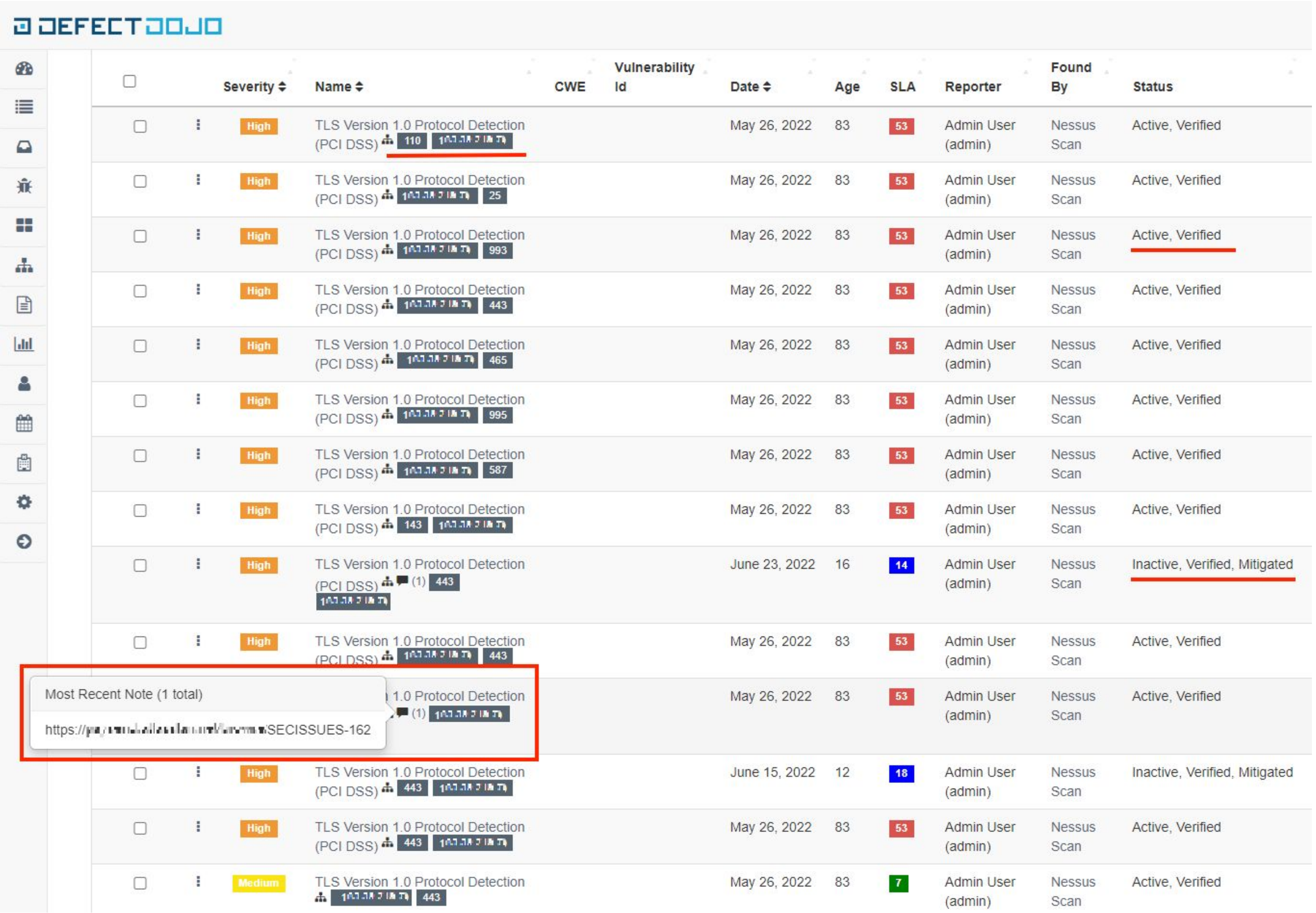
```
from tenable.io import TenableIO

def cloud_client():
    client = TenableIO(access_key="ACCESS_KEY", secret_key="SECRET_KEY")
    return client


def get_scan_results(scan_name):
    for scan in cloud_client().scans.list():
        if scan["name"] == scan_name:
            # wait end of the scan if it is working
            while True:
                if cloud_client().scans.status(scan["id"]) in ("completed", "canceled"):
                    break
                time.sleep(60)
        scan_results = cloud_client().scans.results(scan["id"])
        return scan_results
    return
```

Integration with DefectDojo



```
import requests

def create_dojo_finding(title=None, risk_factor=None, description=None, solution=None, plugin_output=None, host_name=None, port=None):
    # create tags for visualization
    if port == "0":
        tags = [host_name]
    else:
        tags = [host_name, port]

    # create payload
    payload = { "title": title,
                "severity": risk_factor,
                "description": description,
                "mitigation": solution,
                "severity_justification": plugin_output,
                "url": host_name,
                "tags": tags,
                "verified": False,
                "active": True,
                "duplicate": False,
                "false_positive": False }

    finding_id = requests.post("DEFECT_DOJO_URL/api/v2/findings/", headers="{DEFECT_DOJO_HEADERS}", json=payload).json()

    if finding_id:
        return finding_id.get("id")
    else:
        retrun 1
```

DefectDojo



□		Severity	Name	CWE	Vulnerability Id	Date	Age	SLA	Reporter	Found By	Status
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) 110 [redacted]			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 25			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 993			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 443			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 465			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 995			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 587			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) 143 [redacted]			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) (1) 443 [redacted]			June 23, 2022	16	14	Admin User (admin)	Nessus Scan	Inactive, Verified, Mitigated
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) [redacted] 443			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
			1.0 Protocol Detection (1) [redacted]			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) 443 [redacted]			June 15, 2022	12	18	Admin User (admin)	Nessus Scan	Inactive, Verified, Mitigated
□	⋮	High	TLS Version 1.0 Protocol Detection (PCI DSS) 443 [redacted]			May 26, 2022	83	53	Admin User (admin)	Nessus Scan	Active, Verified
□	⋮	Medium	TLS Version 1.0 Protocol Detection [redacted] 443			May 26, 2022	83	7	Admin User (admin)	Nessus Scan	Active, Verified

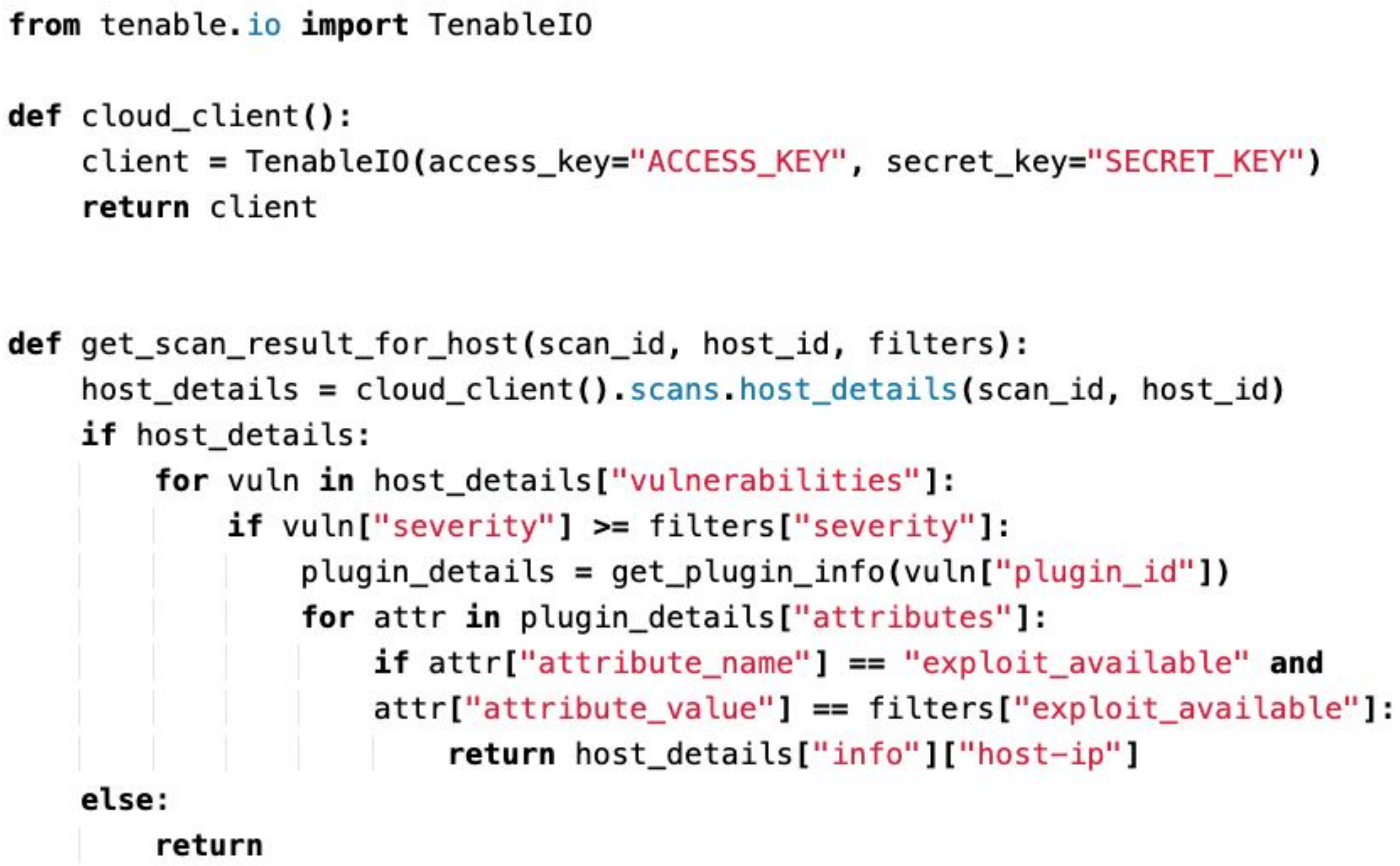
Most Recent Note (1 total)

https://[redacted]SECISSUES-162

How we can use automation?

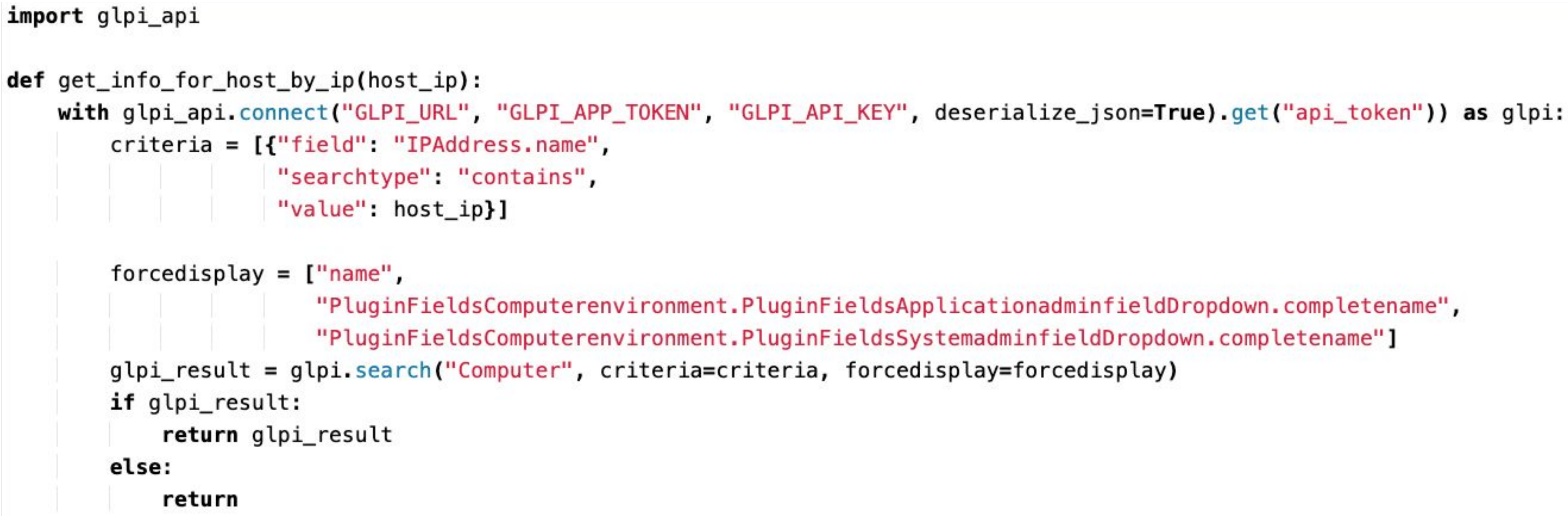
Automation can be applied	**Need to integration with…**
Collect needed IP range from inventory system or other sources	System inventory / cloud / other (VMware, etc.)
Create scan and/or start it	Scanner
Extract to some analysis platform	Scanner, DefectDojo
Create remediation tickets to target teams	**Scanner, Task manager, System inventory**
Check fixing and close fixed findings	Scanner, DefectDojo

Integration with Tenable



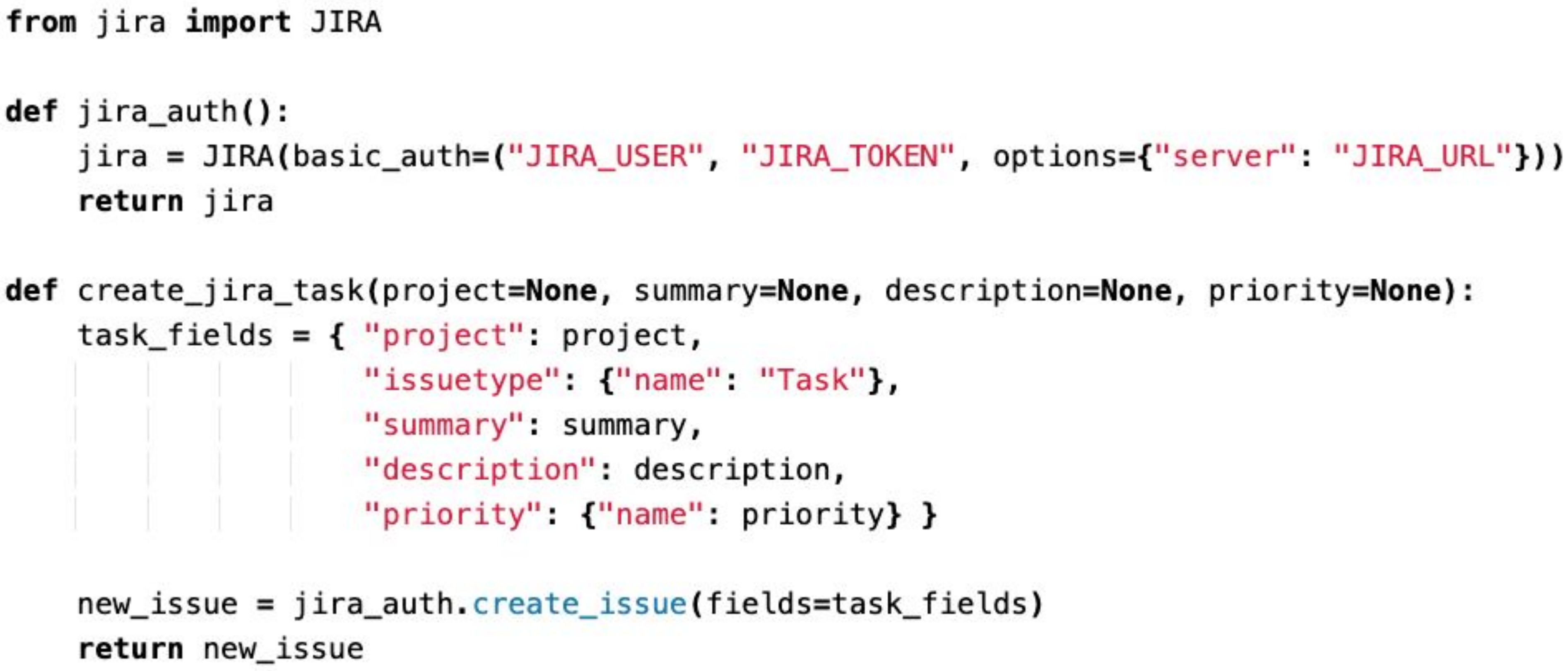
```python
from tenable.io import TenableIO

def cloud_client():
    client = TenableIO(access_key="ACCESS_KEY", secret_key="SECRET_KEY")
    return client


def get_scan_result_for_host(scan_id, host_id, filters):
    host_details = cloud_client().scans.host_details(scan_id, host_id)
    if host_details:
        for vuln in host_details["vulnerabilities"]:
            if vuln["severity"] >= filters["severity"]:
                plugin_details = get_plugin_info(vuln["plugin_id"])
                for attr in plugin_details["attributes"]:
                    if attr["attribute_name"] == "exploit_available" and
                    attr["attribute_value"] == filters["exploit_available"]:
                        return host_details["info"]["host-ip"]
    else:
        return
```

Integration with GLPI



```
import glpi_api

def get_info_for_host_by_ip(host_ip):
    with glpi_api.connect("GLPI_URL", "GLPI_APP_TOKEN", "GLPI_API_KEY", deserialize_json=True).get("api_token")) as glpi:
        criteria = [{"field": "IPAddress.name",
                     "searchtype": "contains",
                     "value": host_ip}]

        forcedisplay = ["name",
                        "PluginFieldsComputerenvironment.PluginFieldsApplicationadminfieldDropdown.completename",
                        "PluginFieldsComputerenvironment.PluginFieldsSystemadminfieldDropdown.completename"]
        glpi_result = glpi.search("Computer", criteria=criteria, forcedisplay=forcedisplay)
        if glpi_result:
            return glpi_result
        else:
            return
```

Integration with Jira



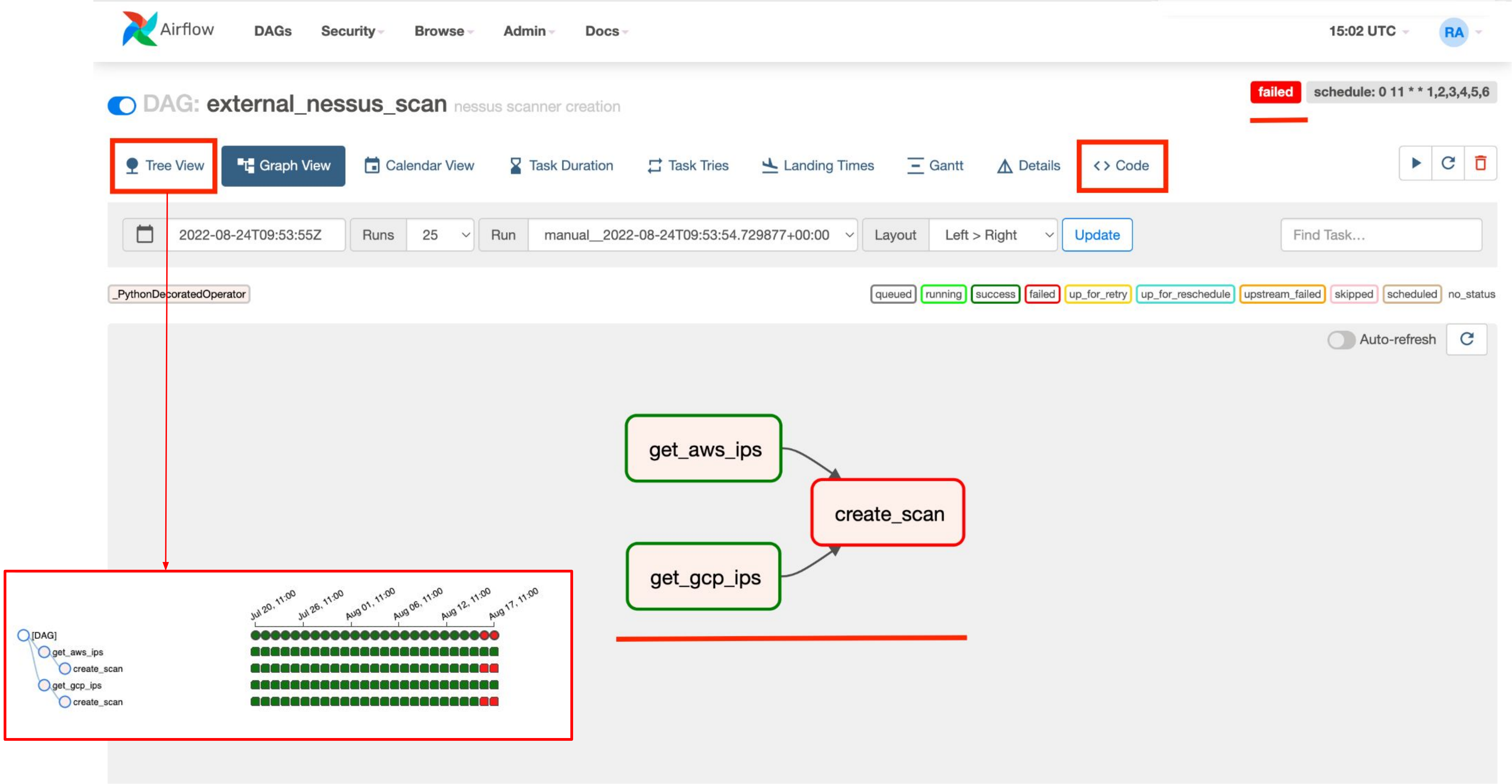
```python
from jira import JIRA

def jira_auth():
    jira = JIRA(basic_auth=("JIRA_USER", "JIRA_TOKEN", options={"server": "JIRA_URL"}))
    return jira

def create_jira_task(project=None, summary=None, description=None, priority=None):
    task_fields = { "project": project,
                    "issuetype": {"name": "Task"},
                    "summary": summary,
                    "description": description,
                    "priority": {"name": priority} }

    new_issue = jira_auth.create_issue(fields=task_fields)
    return new_issue
```

Airflow

- Python code
- Scheduler
- Built-in integrations*
  - AWS
  - GCP
  - Slack
  - Jira
  - …
- Built-in secret store

* https://airflow.apache.org/docs/apache-airflow-providers/packages-ref.html

Airflow



DAG: external_nessus_scan nessus scanner creation

failed

schedule: 0 11 * * 1,2,3,4,5,6

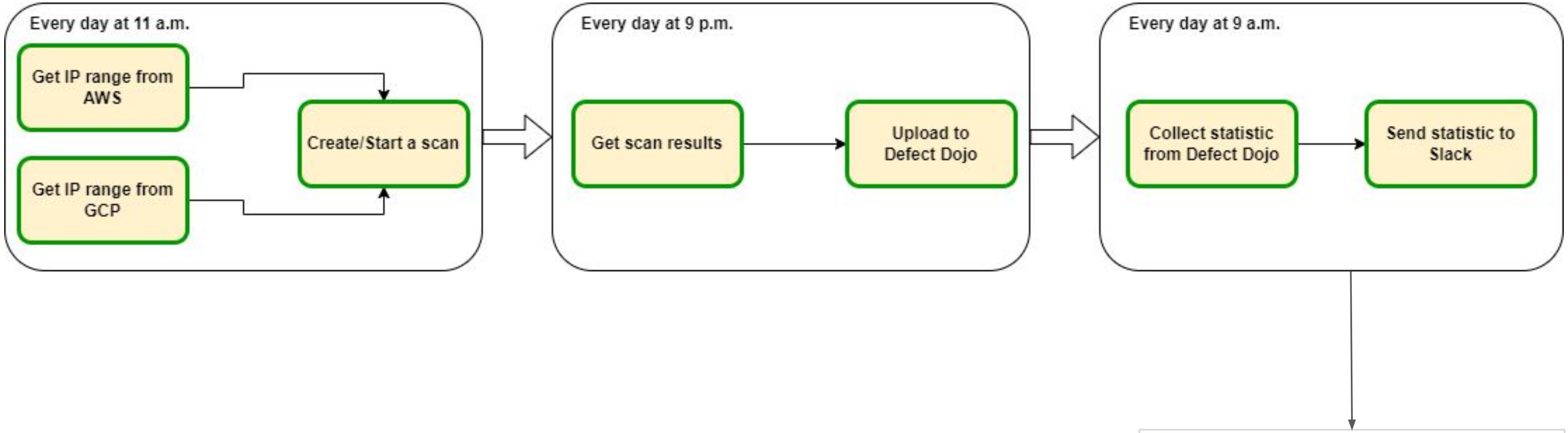
2022-08-24T09:53:55Z

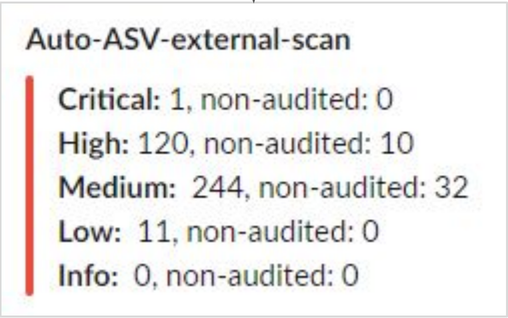
Runs 25

Run manual__2022-08-24T09:53:54.729877+00:00

Layout Left > Right

3 DAG in Airflow for external scanning

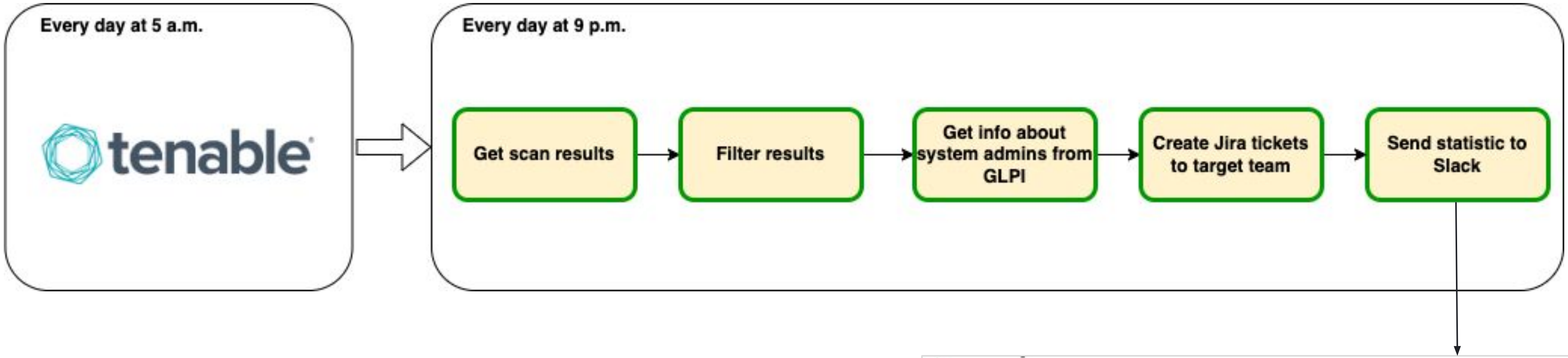


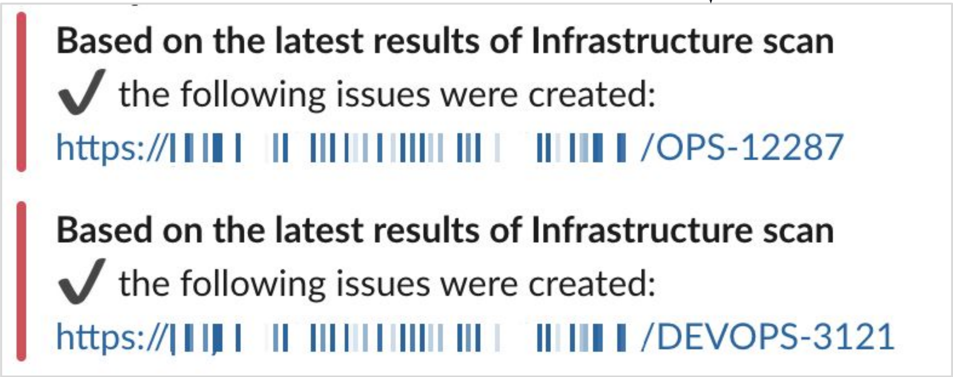


Auto-ASV-external-scan

Critical: 1, non-audited: 0
High: 120, non-audited: 10
Medium: 244, non-audited: 32
Low: 11, non-audited: 0
Info: 0, non-audited: 0

1 DAG in Airflow for internal scanning





Based on the latest results of Infrastructure scan

✓ the following issues were created:

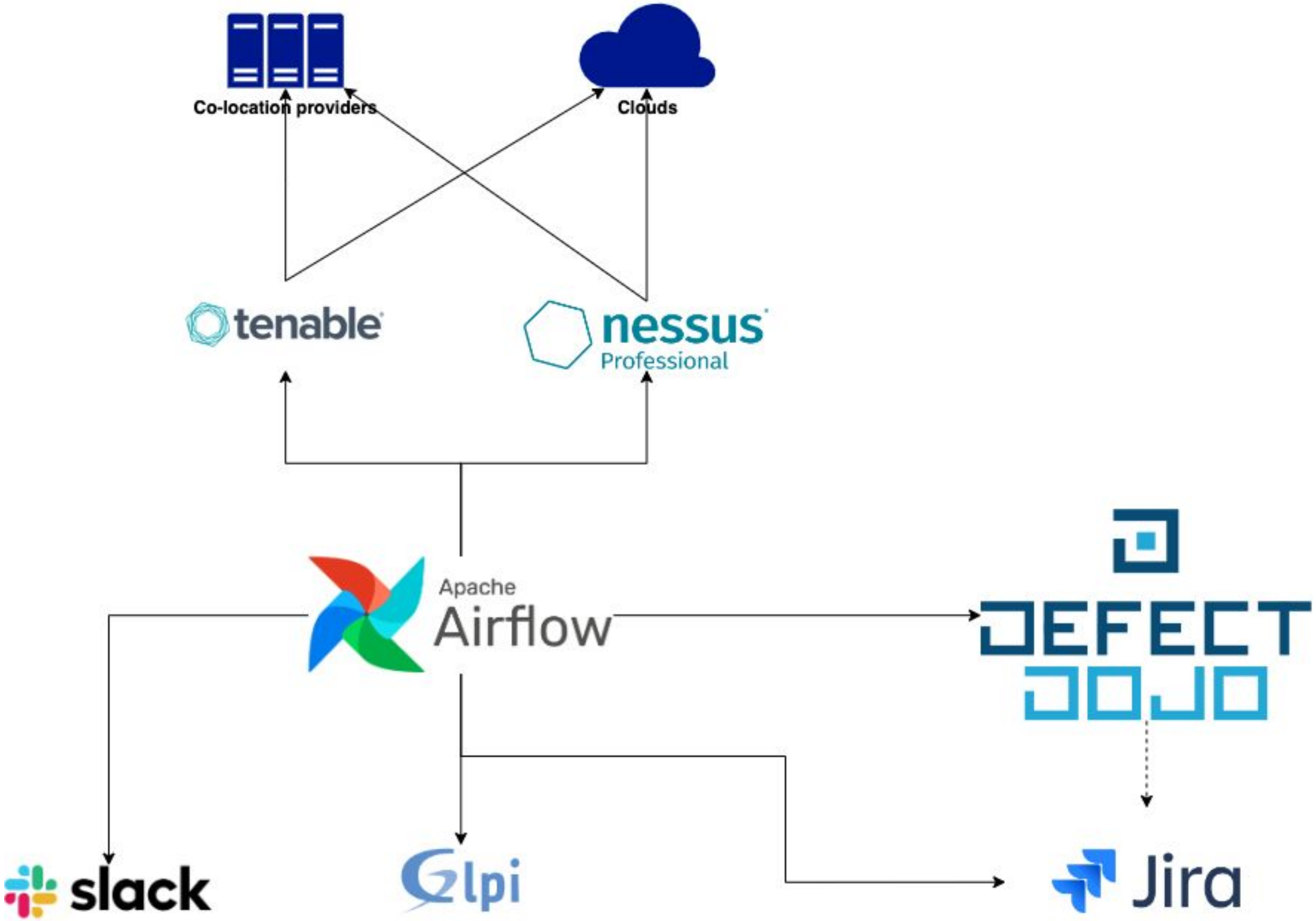
https://[redacted]/OPS-12287

Based on the latest results of Infrastructure scan

✓ the following issues were created:

https://[redacted]/DEVOPS-3121

Current state and further improvements



AppSec part

Goals:

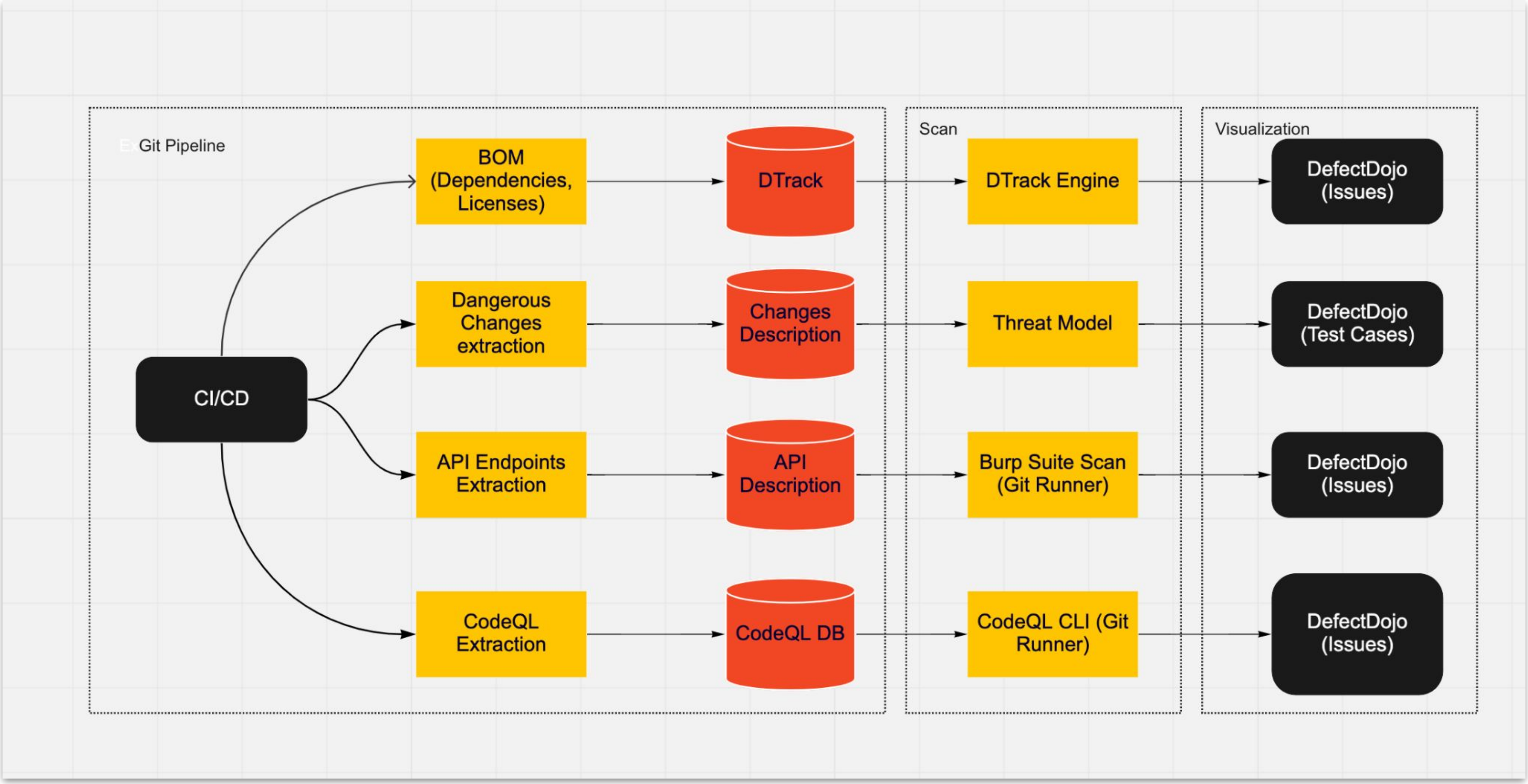
- Support different scan types for our codebase
- Clear mechanism for connecting new products to scanners
- Convenient distribution of scan results

Scan types

Our scans:

- SAST (CodeQL)
- DAST (Burp Suite)
- Dependencies (dependency track)
- Secrets (gitleaks)
- Licenses (dependency track)

Scan types



Git Pipeline

Scan

Visualization

CI/CD

BOM (Dependencies, Licenses)

Dangerous Changes extraction

API Endpoints Extraction

CodeQL Extraction

DTrack

Changes Description

API Description

CodeQL DB

DTrack Engine

Threat Model

Burp Suite Scan (Git Runner)

CodeQL CLI (Git Runner)

DefectDojo (Issues)

DefectDojo (Test Cases)

DefectDojo (Issues)

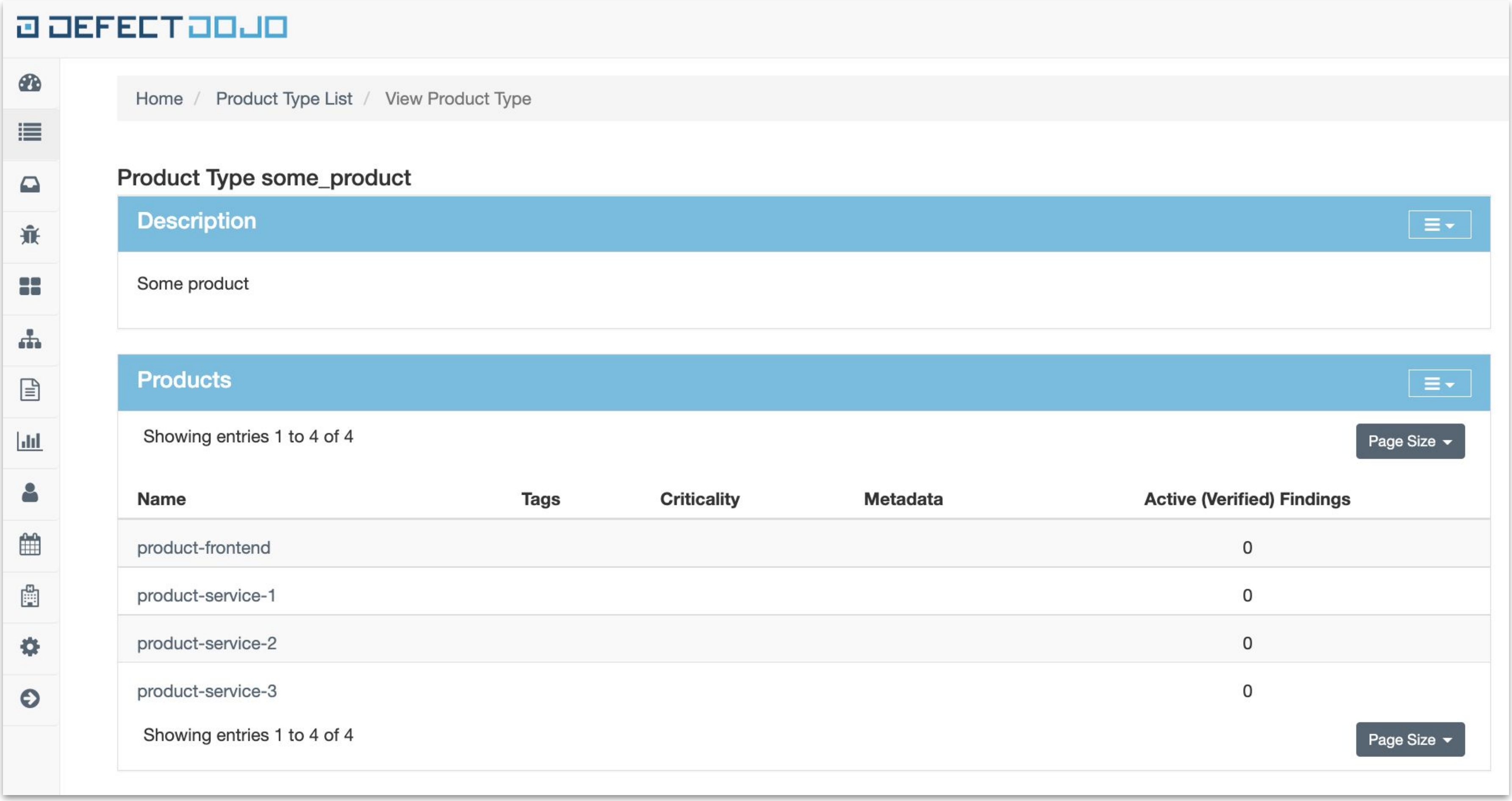
DefectDojo (Issues)

Scan results organization

Mapping:

- Our product -> DD product type
- Service of product -> DD product linked to product type
- Service scan results -> DD test linked to product

Scan results organization



Home / Product Type List / View Product Type

Product Type some_product

Description

Some product

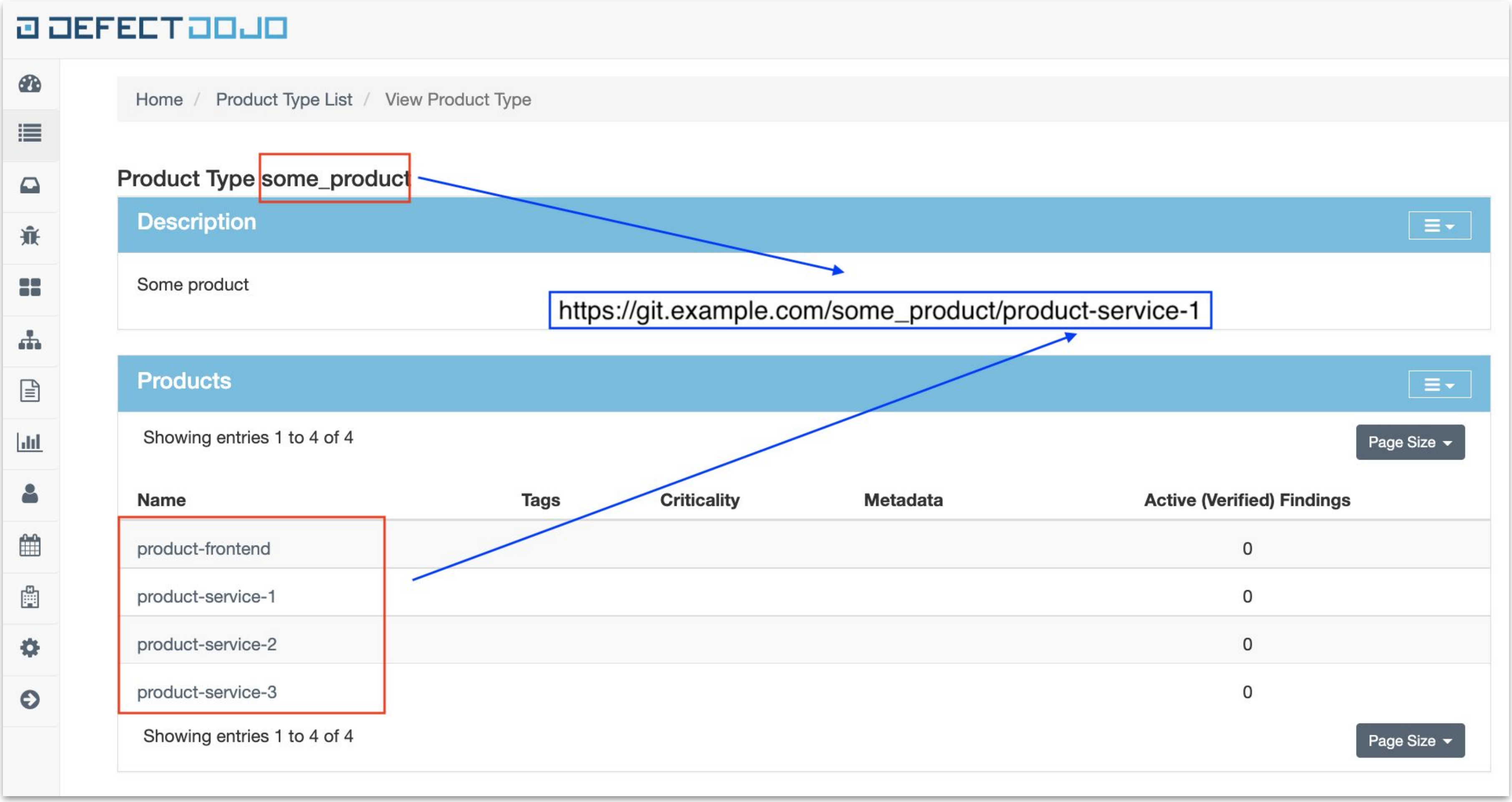
Products

Showing entries 1 to 4 of 4

Name	Tags	Criticality	Metadata	Active (Verified) Findings
product-frontend				0
product-service-1				0
product-service-2				0
product-service-3				0

Showing entries 1 to 4 of 4

Scan results organization



Home / Product Type List / View Product Type

Product Type some_product

Description

Some product

https://git.example.com/some_product/product-service-1

Products

Showing entries 1 to 4 of 4

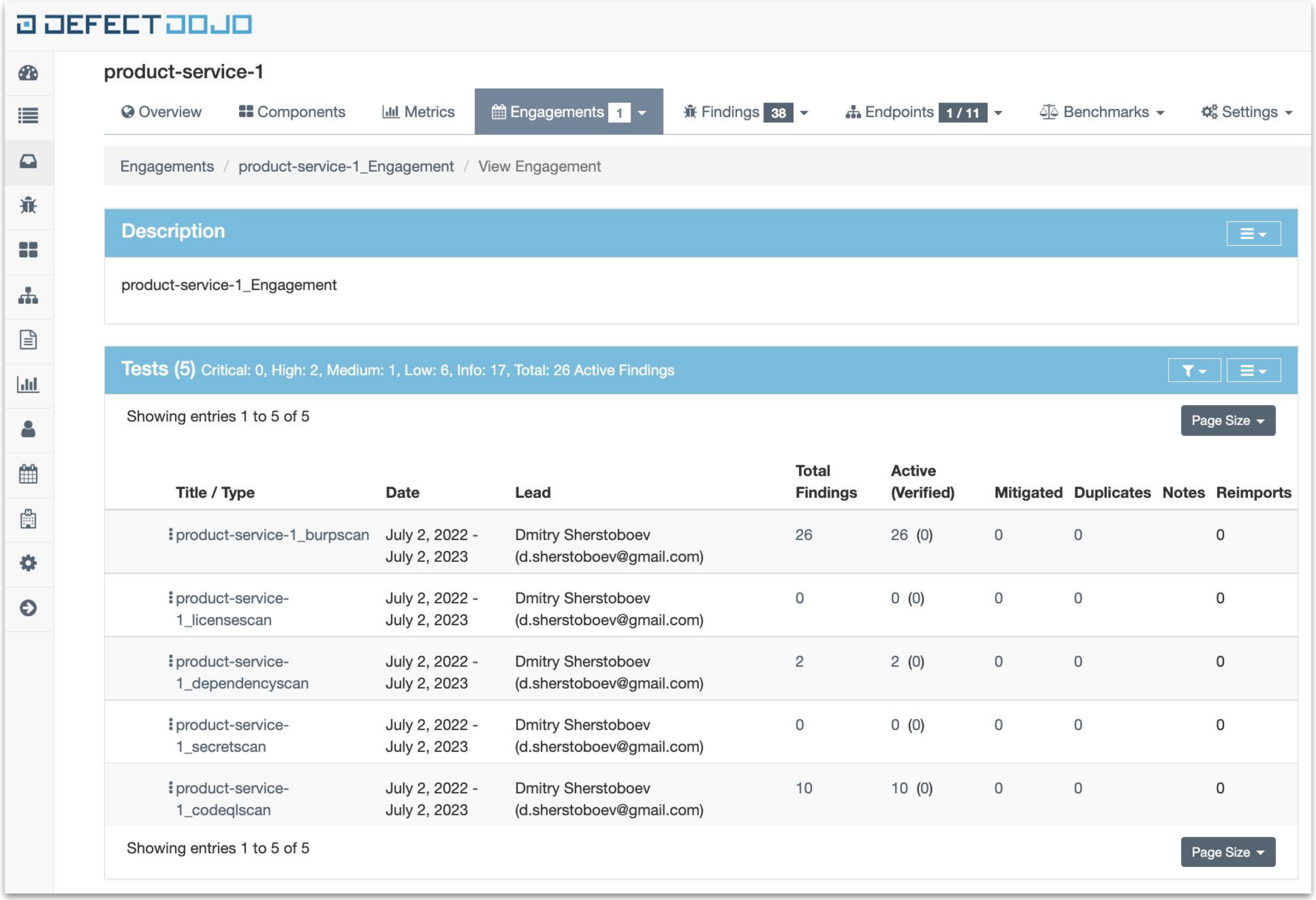
Page Size

Name	Tags	Criticality	Metadata	Active (Verified) Findings
product-frontend				0
product-service-1				0
product-service-2				0
product-service-3				0

Showing entries 1 to 4 of 4

Page Size

Scan results organization



product-service-1

Engagements / product-service-1_Engagement / View Engagement

Description

product-service-1_Engagement

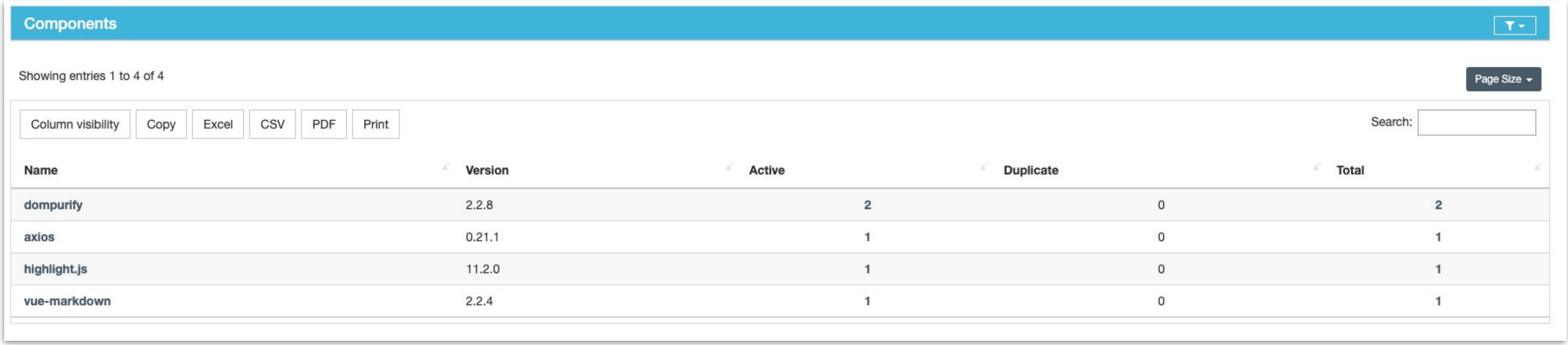
Tests (5) Critical: 0, High: 2, Medium: 1, Low: 6, Info: 17, Total: 26 Active Findings

Showing entries 1 to 5 of 5

Title / Type	Date	Lead	Total Findings	Active (Verified)	Mitigated	Duplicates	Notes	Reimports
product-service-1_burpscan	July 2, 2022 - July 2, 2023	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	26	26 (0)	0	0		0
product-service-1_licensescan	July 2, 2022 - July 2, 2023	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	0	0 (0)	0	0		0
product-service-1_dependencyscan	July 2, 2022 - July 2, 2023	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	2	2 (0)	0	0		0
product-service-1_secretscan	July 2, 2022 - July 2, 2023	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	0	0 (0)	0	0		0
product-service-1_codeqlscan	July 2, 2022 - July 2, 2023	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	10	10 (0)	0	0		0

Showing entries 1 to 5 of 5

Scan results organization

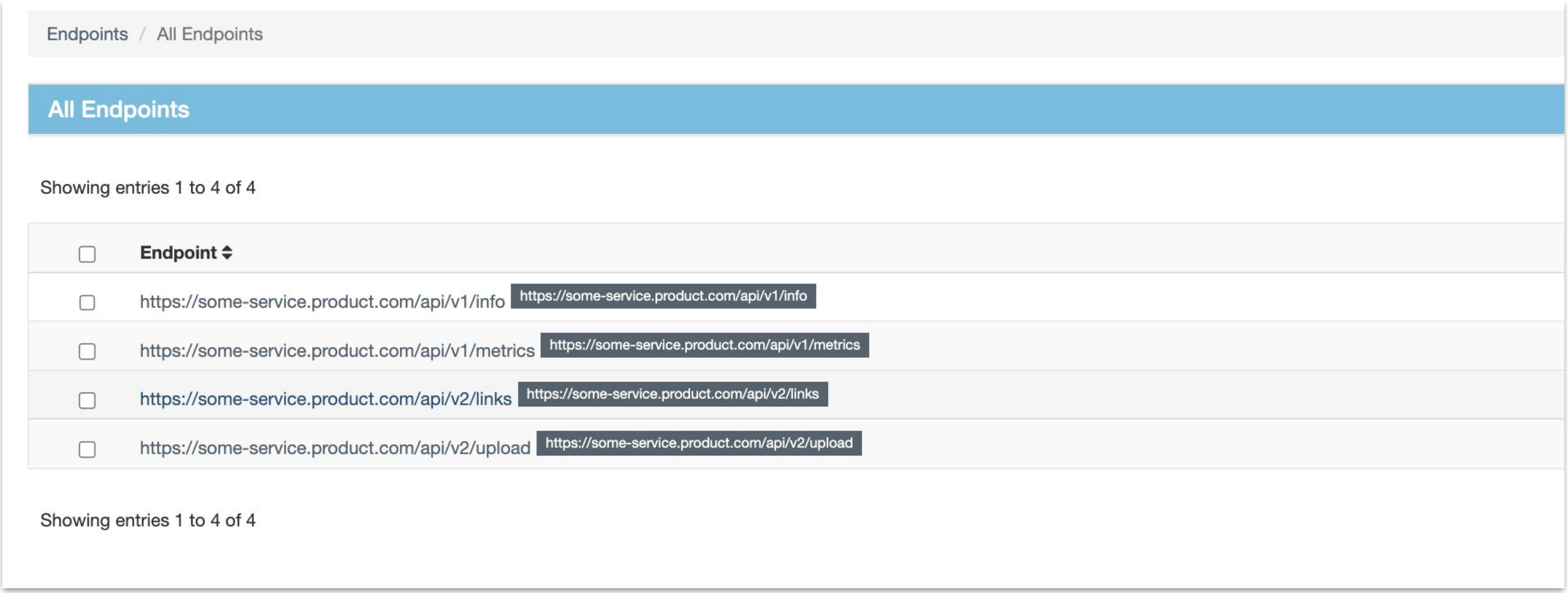


Components

Showing entries 1 to 4 of 4

Name	Version	Active	Duplicate	Total
dompurify	2.2.8	2	0	2
axios	0.21.1	1	0	1
highlight.js	11.2.0	1	0	1
vue-markdown	2.2.4	1	0	1

Scan results organization



Endpoints / All Endpoints

All Endpoints

Showing entries 1 to 4 of 4

□	Endpoint ⬍
□	https://some-service.product.com/api/v1/info https://some-service.product.com/api/v1/info
□	https://some-service.product.com/api/v1/metrics https://some-service.product.com/api/v1/metrics
□	https://some-service.product.com/api/v2/links https://some-service.product.com/api/v2/links
□	https://some-service.product.com/api/v2/upload https://some-service.product.com/api/v2/upload

Showing entries 1 to 4 of 4

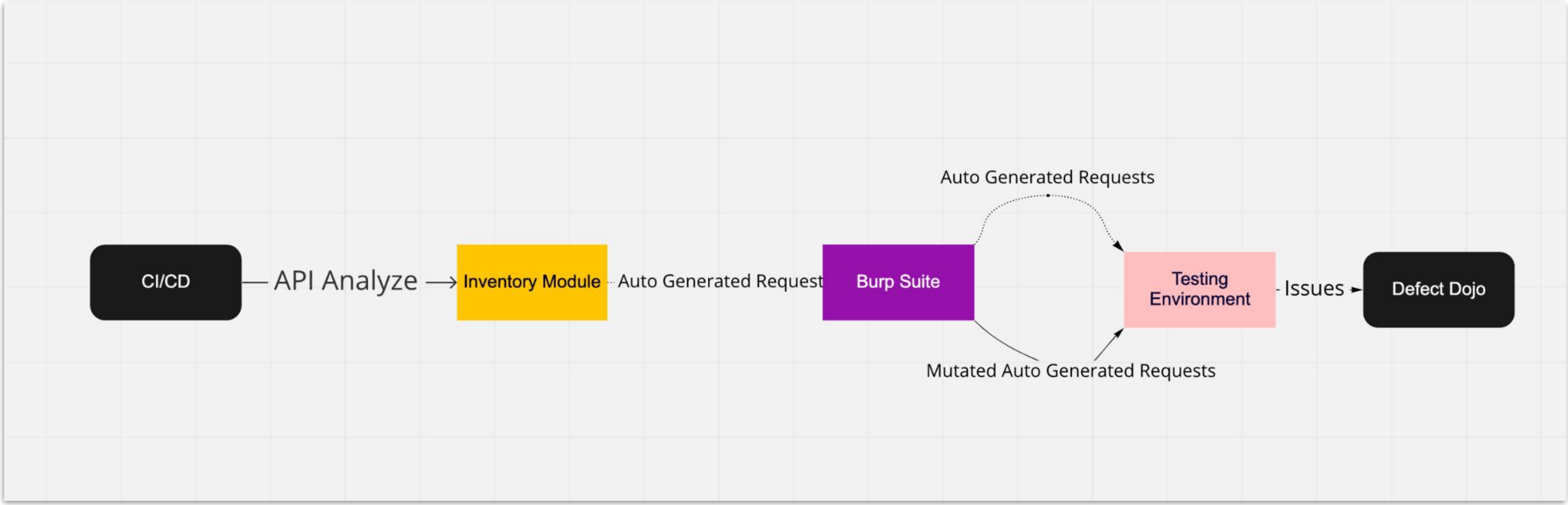
Upload results

- Scan uploaders have a shared logic for results distribution in DefectDojo
- This part checks DD project structure and finds or creates needed entity (project, engagement, test)

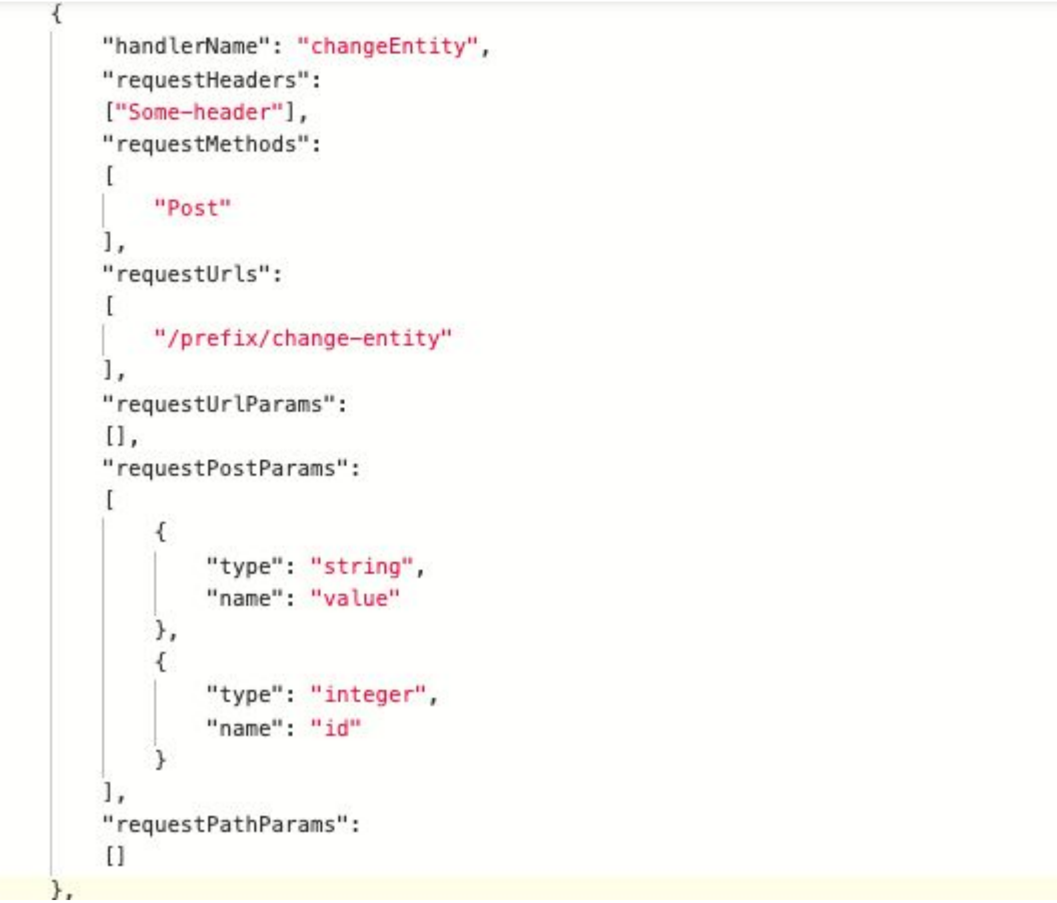
About DAST

- API discover
  - API analyze
    - Swaggers
    - Custom solution (Parse code to get handler’s parameters)
  - Path params/Headers/URL params/Request bodies
  - Real data - autotests needed
- Authentication
- Session (configurable auth tokens/cookies TTL)

DAST schema

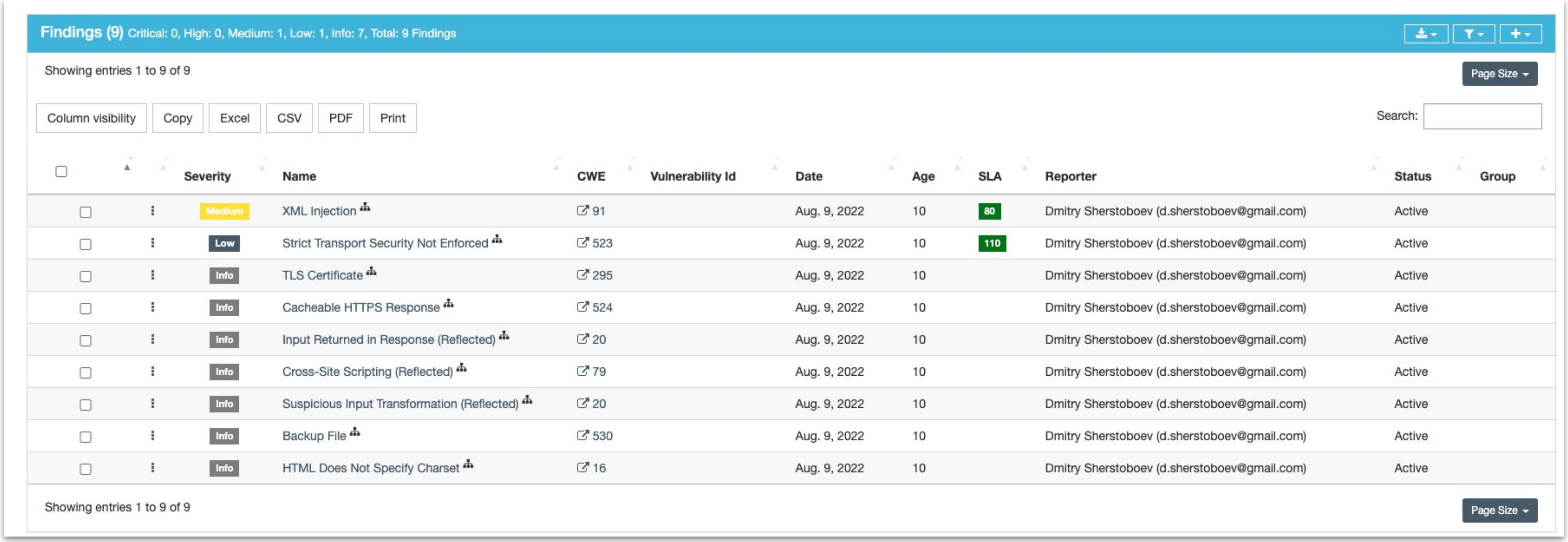

CI/CD
API Analyze
Inventory Module
Auto Generated Request
Burp Suite
Auto Generated Requests
Mutated Auto Generated Requests
Testing Environment
Issues
Defect Dojo

Collect handler’s info



```
{
    "handlerName": "changeEntity",
    "requestHeaders":
    ["Some-header"],
    "requestMethods":
    [
        "Post"
    ],
    "requestUrls":
    [
        "/prefix/change-entity"
    ],
    "requestUrlParams":
    [],
    "requestPostParams":
    [
        {
            "type": "string",
            "name": "value"
        },
        {
            "type": "integer",
            "name": "id"
        }
    ],
    "requestPathParams":
    []
},
```

DefectDojo default burp scan import



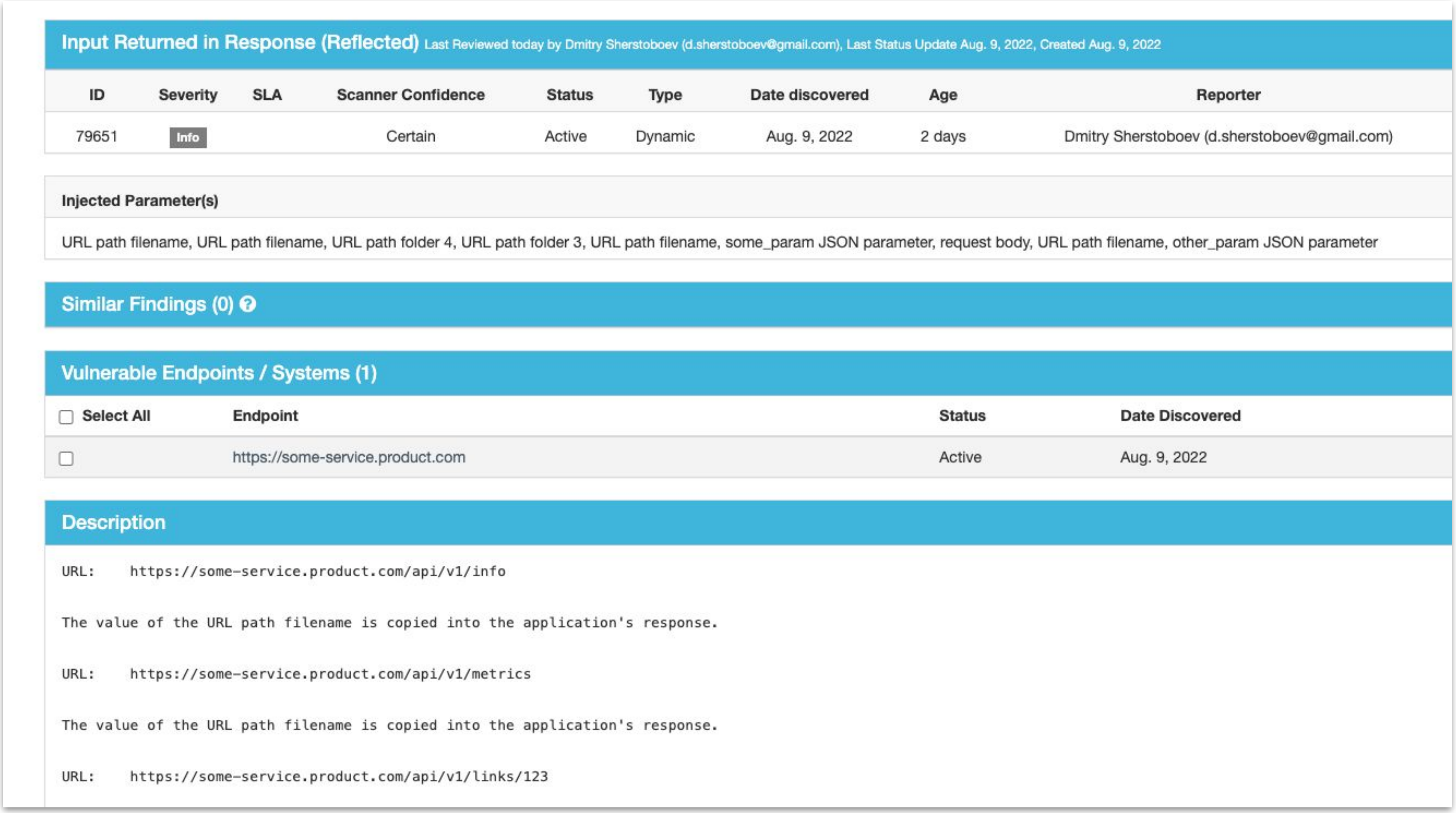
Findings (9) Critical: 0, High: 0, Medium: 1, Low: 1, Info: 7, Total: 9 Findings

Showing entries 1 to 9 of 9

□		Severity	Name	CWE	Vulnerability Id	Date	Age	SLA	Reporter	Status	Group
□	⋮	Medium	XML Injection	91		Aug. 9, 2022	10	80	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Low	Strict Transport Security Not Enforced	523		Aug. 9, 2022	10	110	Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	TLS Certificate	295		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	Cacheable HTTPS Response	524		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	Input Returned in Response (Reflected)	20		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	Cross-Site Scripting (Reflected)	79		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	Suspicious Input Transformation (Reflected)	20		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	Backup File	530		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	
□	⋮	Info	HTML Does Not Specify Charset	16		Aug. 9, 2022	10		Dmitry Sherstoboev (d.sherstoboev@gmail.com)	Active	

Showing entries 1 to 9 of 9

DefectDojo default burp scan import



Input Returned in Response (Reflected)

Last Reviewed today by Dmitry Sherstoboev (d.sherstoboev@gmail.com), Last Status Update Aug. 9, 2022, Created Aug. 9, 2022

ID	Severity	SLA	Scanner Confidence	Status	Type	Date discovered	Age	Reporter
79651	Info		Certain	Active	Dynamic	Aug. 9, 2022	2 days	Dmitry Sherstoboev (d.sherstoboev@gmail.com)

Injected Parameter(s)

URL path filename, URL path filename, URL path folder 4, URL path folder 3, URL path filename, some_param JSON parameter, request body, URL path filename, other_param JSON parameter

Similar Findings (0) ?

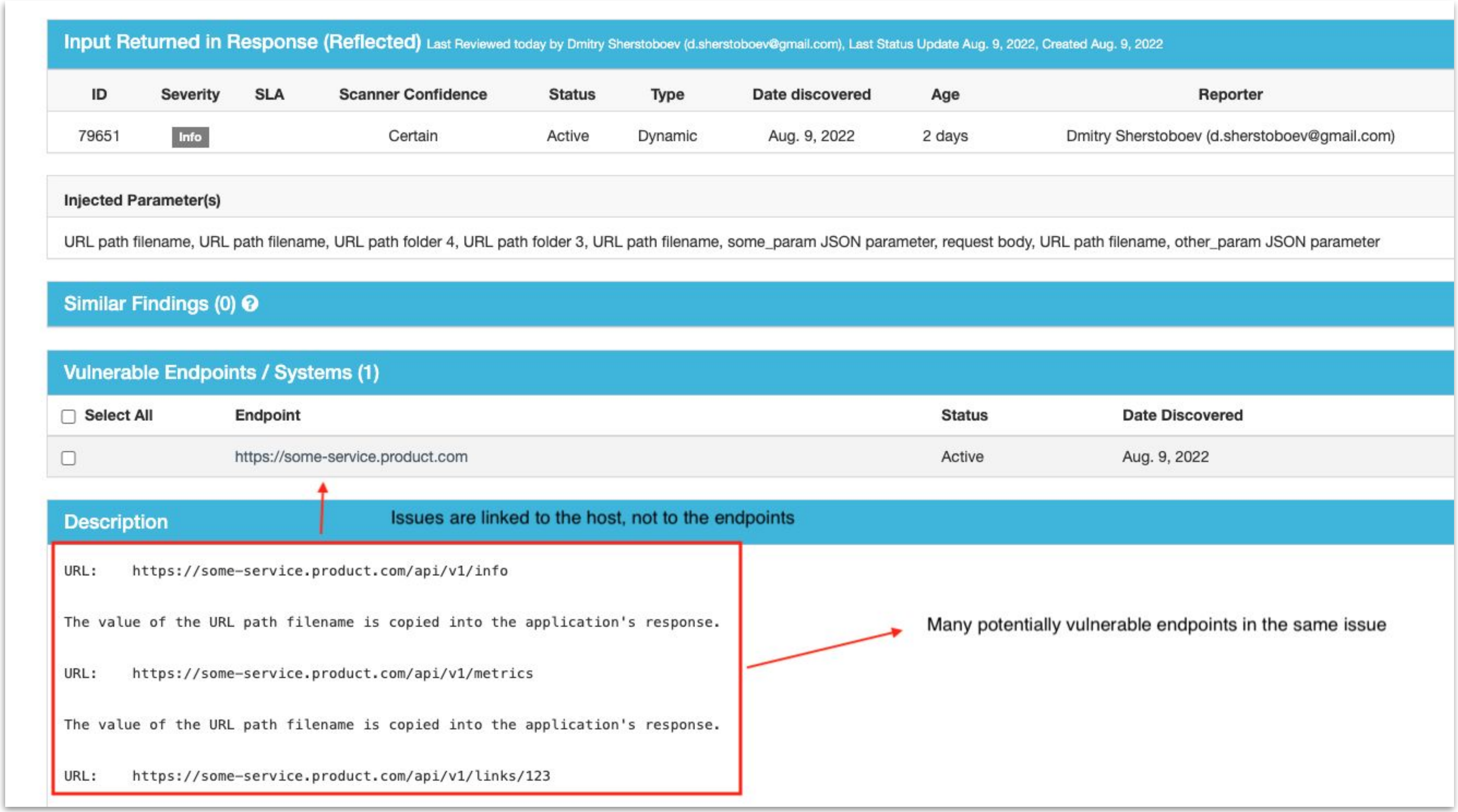
Vulnerable Endpoints / Systems (1)

□ Select All	Endpoint	Status	Date Discovered
□	https://some-service.product.com	Active	Aug. 9, 2022

Description

```
URL:    https://some-service.product.com/api/v1/info

The value of the URL path filename is copied into the application's response.

URL:    https://some-service.product.com/api/v1/metrics

The value of the URL path filename is copied into the application's response.

URL:    https://some-service.product.com/api/v1/links/123
```

DefectDojo default burp scan import



Input Returned in Response (Reflected) Last Reviewed today by Dmitry Sherstoboev (d.sherstoboev@gmail.com), Last Status Update Aug. 9, 2022, Created Aug. 9, 2022

ID	Severity	SLA	Scanner Confidence	Status	Type	Date discovered	Age	Reporter
79651	Info		Certain	Active	Dynamic	Aug. 9, 2022	2 days	Dmitry Sherstoboev (d.sherstoboev@gmail.com)

Injected Parameter(s)

URL path filename, URL path filename, URL path folder 4, URL path folder 3, URL path filename, some_param JSON parameter, request body, URL path filename, other_param JSON parameter

Similar Findings (0)

Vulnerable Endpoints / Systems (1)

□ Select All	Endpoint	Status	Date Discovered
□	https://some-service.product.com	Active	Aug. 9, 2022

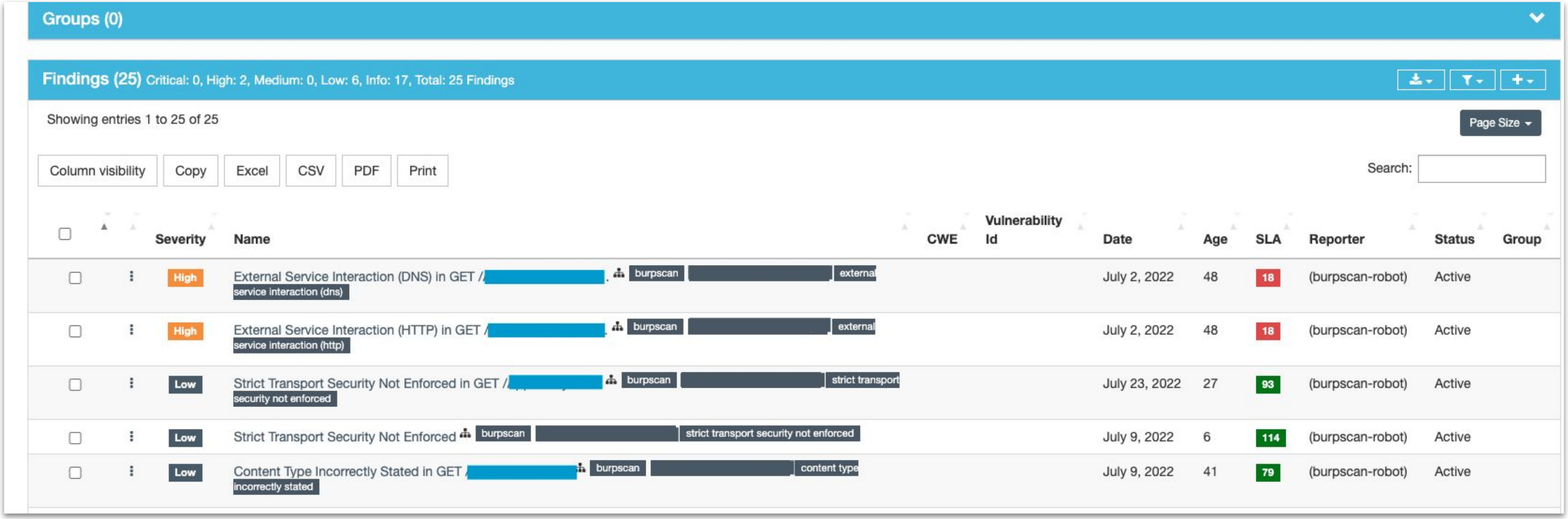
Issues are linked to the host, not to the endpoints

Description

```
URL:    https://some-service.product.com/api/v1/info

The value of the URL path filename is copied into the application's response.

URL:    https://some-service.product.com/api/v1/metrics

The value of the URL path filename is copied into the application's response.

URL:    https://some-service.product.com/api/v1/links/123
```

Many potentially vulnerable endpoints in the same issue

How we upload DAST results



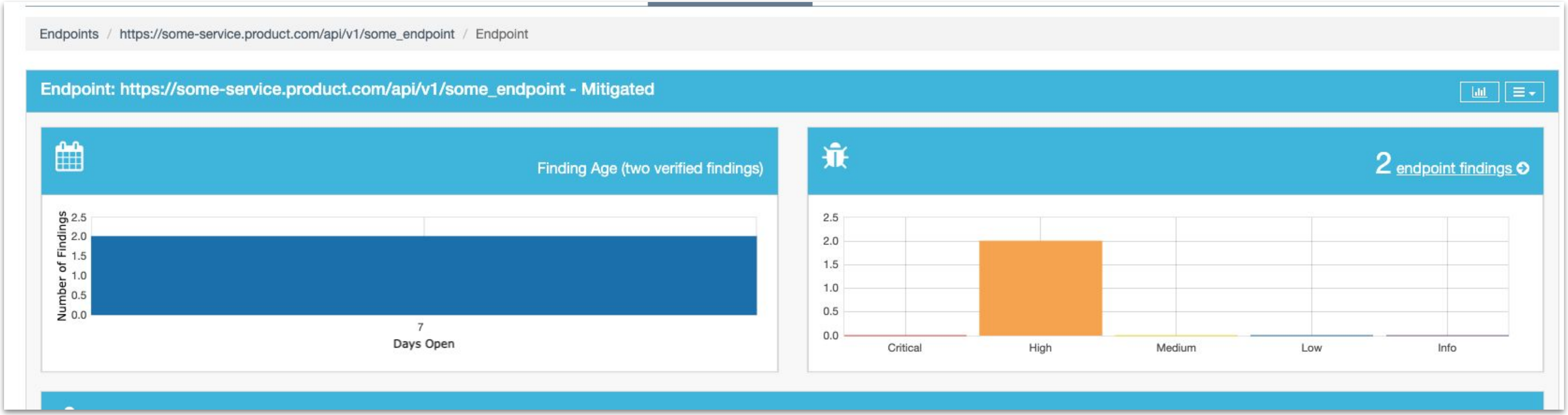
Groups (0)

Findings (25) Critical: 0, High: 2, Medium: 0, Low: 6, Info: 17, Total: 25 Findings

Showing entries 1 to 25 of 25

☐		Severity	Name	CWE	Vulnerability Id	Date	Age	SLA	Reporter	Status	Group
☐	⋮	High	External Service Interaction (DNS) in GET /[redacted]. burpscan [redacted] external service interaction (dns)			July 2, 2022	48	18	(burpscan-robot)	Active	
☐	⋮	High	External Service Interaction (HTTP) in GET /[redacted]. burpscan [redacted] external service interaction (http)			July 2, 2022	48	18	(burpscan-robot)	Active	
☐	⋮	Low	Strict Transport Security Not Enforced in GET /[redacted] burpscan [redacted] strict transport security not enforced			July 23, 2022	27	93	(burpscan-robot)	Active	
☐	⋮	Low	Strict Transport Security Not Enforced burpscan [redacted] strict transport security not enforced			July 9, 2022	6	114	(burpscan-robot)	Active	
☐	⋮	Low	Content Type Incorrectly Stated in GET /[redacted] burpscan [redacted] content type incorrectly stated			July 9, 2022	41	79	(burpscan-robot)	Active	

How we upload DAST results



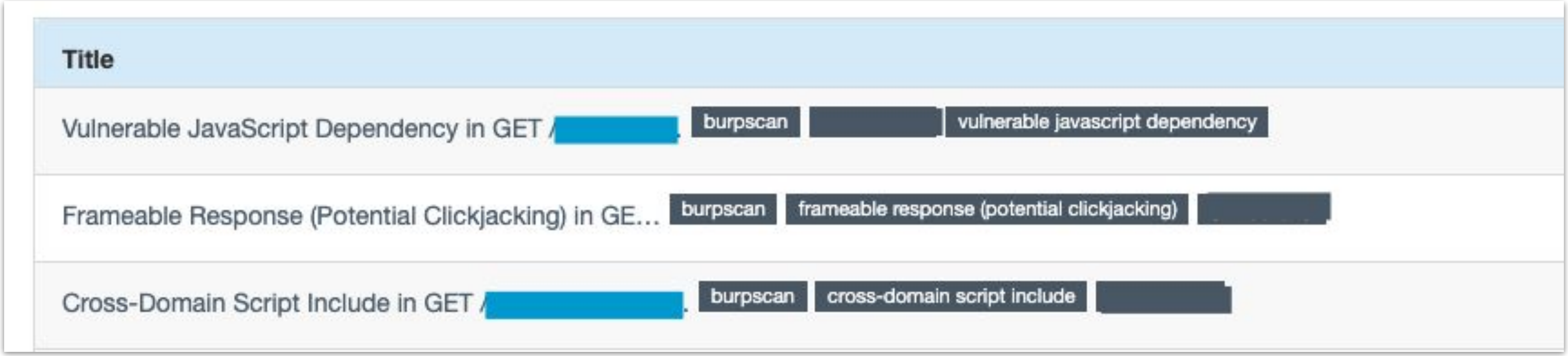
Endpoints / https://some-service.product.com/api/v1/some_endpoint / Endpoint

Endpoint: https://some-service.product.com/api/v1/some_endpoint - Mitigated

Finding Age (two verified findings)

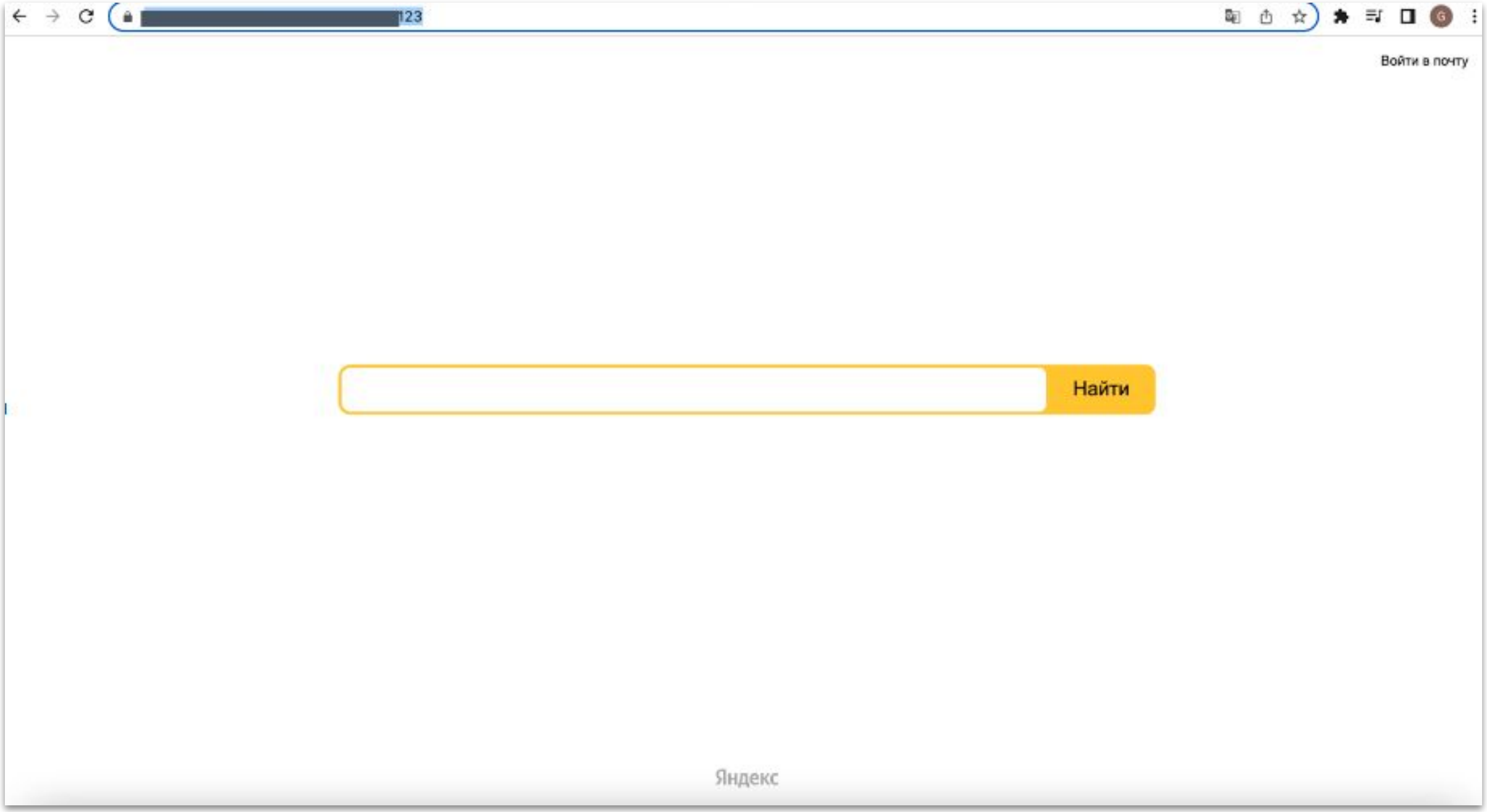
2 endpoint findings

Some findings 1/3



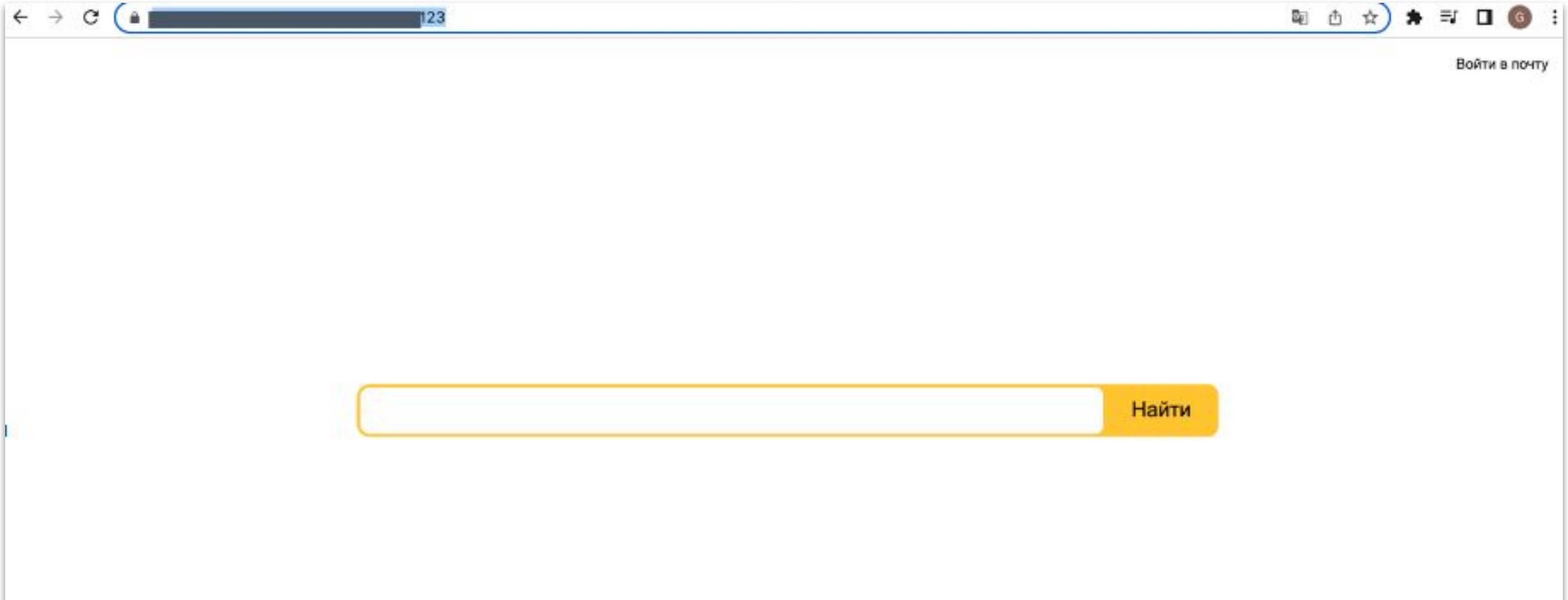
Title
Vulnerable JavaScript Dependency in GET /[redacted]. burpscan [redacted] vulnerable javascript dependency
Frameable Response (Potential Clickjacking) in GE… burpscan frameable response (potential clickjacking) [redacted]
Cross-Domain Script Include in GET /[redacted]. burpscan cross-domain script include [redacted]

Some findings 1/3



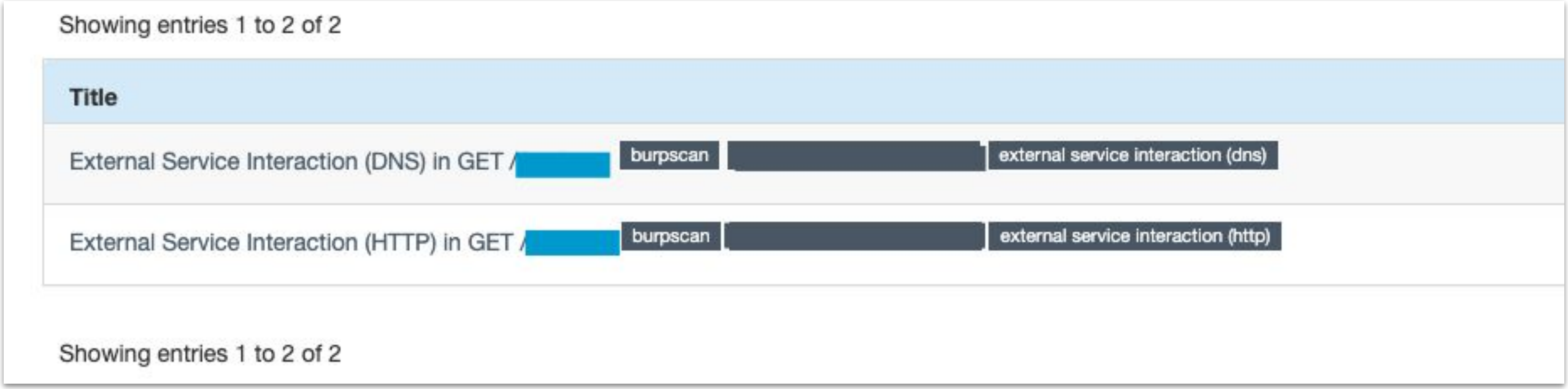
[redacted]

Some findings 1/3



[redacted]123

Some findings 2/3

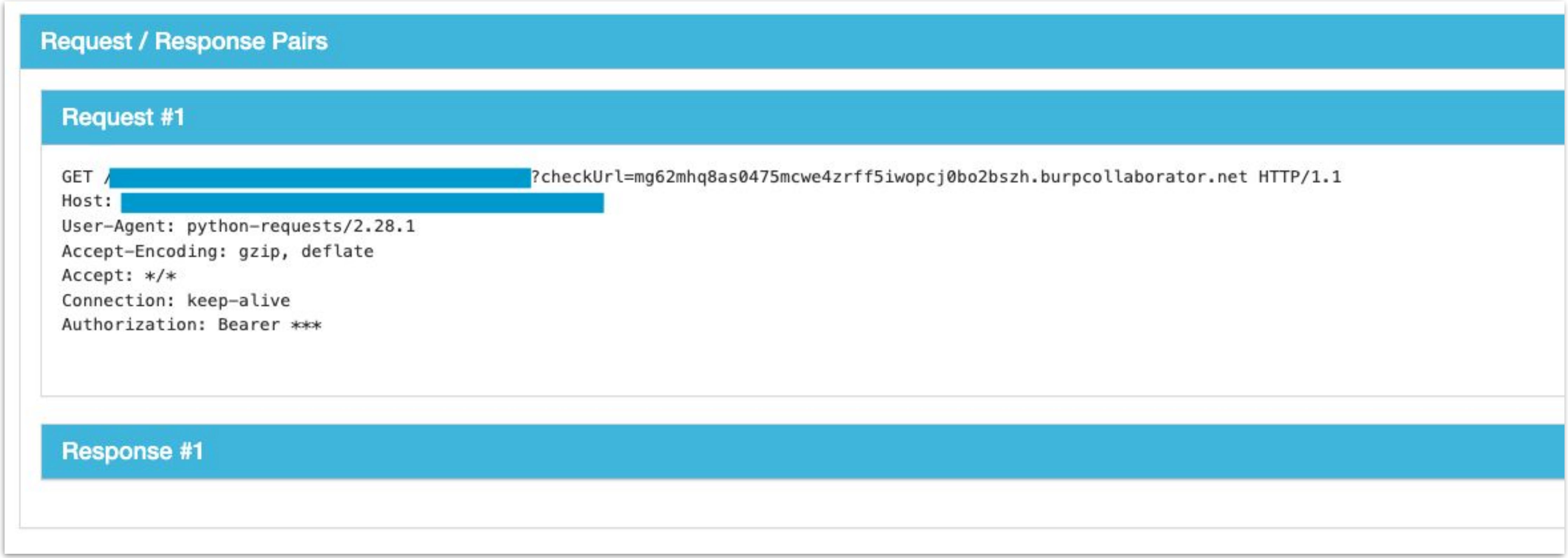


Showing entries 1 to 2 of 2

Title
External Service Interaction (DNS) in GET /[redacted] burpscan [redacted] external service interaction (dns)
External Service Interaction (HTTP) in GET /[redacted] burpscan [redacted] external service interaction (http)

Showing entries 1 to 2 of 2

Some findings 2/3



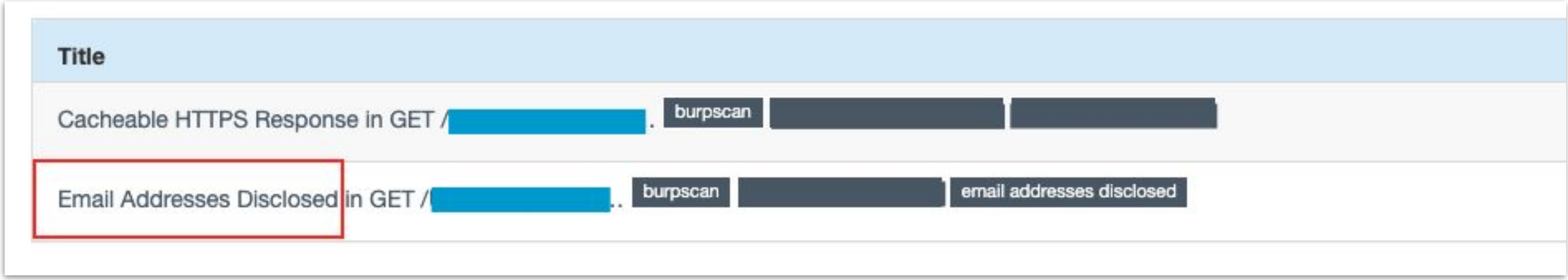
Request / Response Pairs

Request #1

```
GET /[redacted]?checkUrl=mg62mhq8as0475mcwe4zrff5iwopcj0bo2bszh.burpcollaborator.net HTTP/1.1
Host: [redacted]
User-Agent: python-requests/2.28.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: keep-alive
Authorization: Bearer ***
```

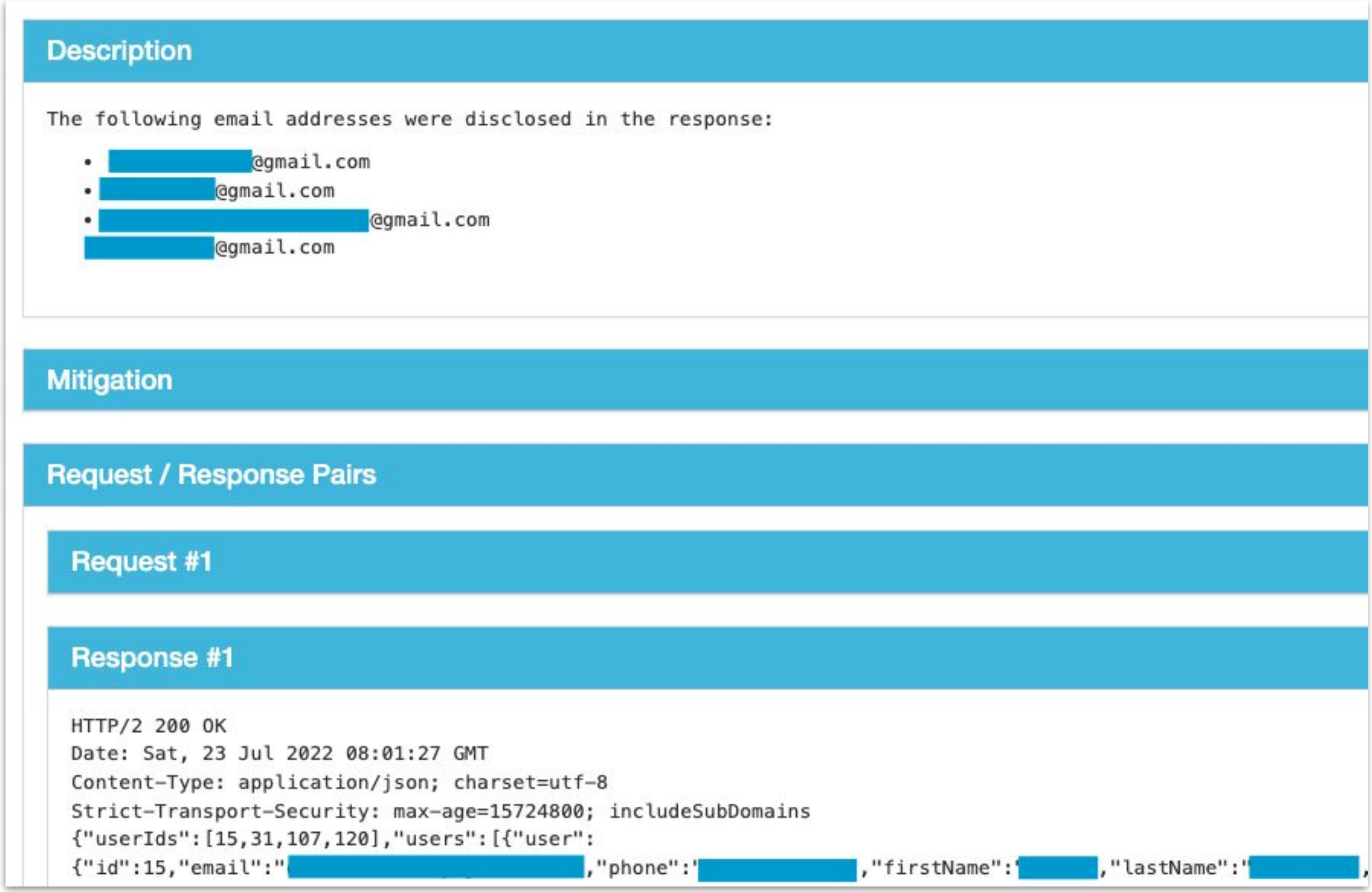
Response #1

Some findings 3/3



Title
Cacheable HTTPS Response in GET /[redacted]. burpscan [redacted] [redacted]
Email Addresses Disclosed in GET /[redacted].. burpscan [redacted] email addresses disclosed

Some findings 3/3



Description

The following email addresses were disclosed in the response:

- [redacted]@gmail.com
- [redacted]@gmail.com
- [redacted]@gmail.com

[redacted]@gmail.com

Mitigation

Request / Response Pairs

Request #1

Response #1

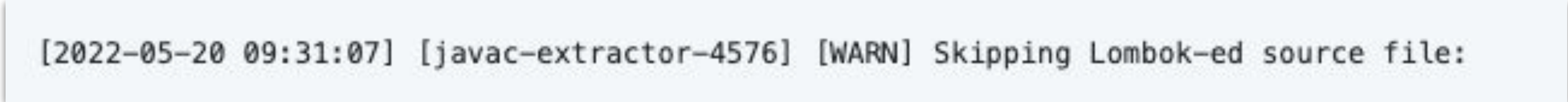
```
HTTP/2 200 OK
Date: Sat, 23 Jul 2022 08:01:27 GMT
Content-Type: application/json; charset=utf-8
Strict-Transport-Security: max-age=15724800; includeSubDomains
{"userIds":[15,31,107,120],"users":[{"user":
{"id":15,"email":"[redacted],"phone":"[redacted],"firstName":"[redacted],"lastName":"[redacted],
```

About SAST

- Write custom rules to highlight potential vulnerable code
- Support products with autogenerated boilerplate code
    - e.g. Lombok for Java
- Support connecting for different languages (because codeql needs to build sources)

Scan with autogenerated code

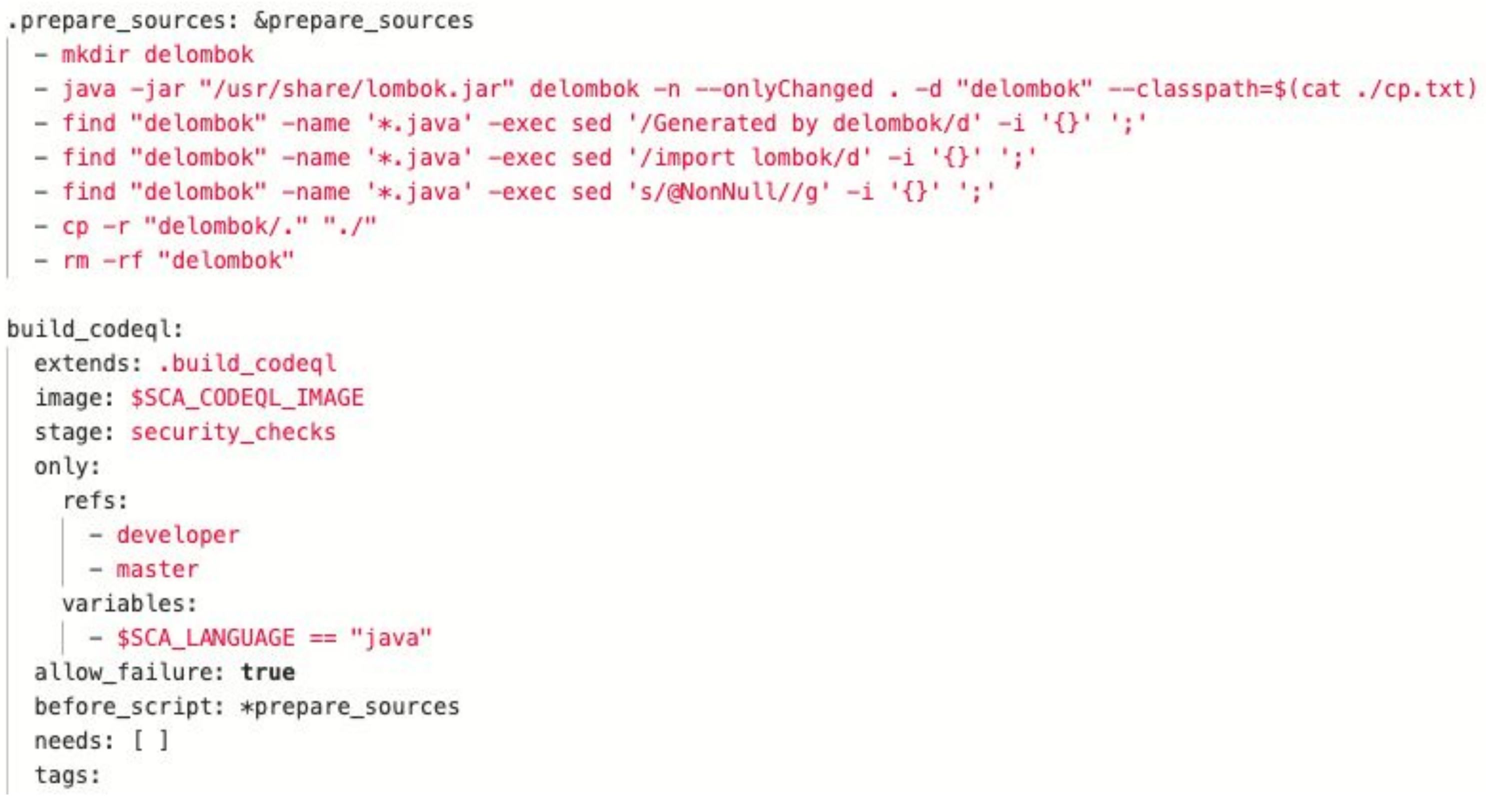
- Teams use Lombok and other libs to not write boilerplate code
- But codeql skipped these files:



```
[2022-05-20 09:31:07] [javac-extractor-4576] [WARN] Skipping Lombok-ed source file:
```

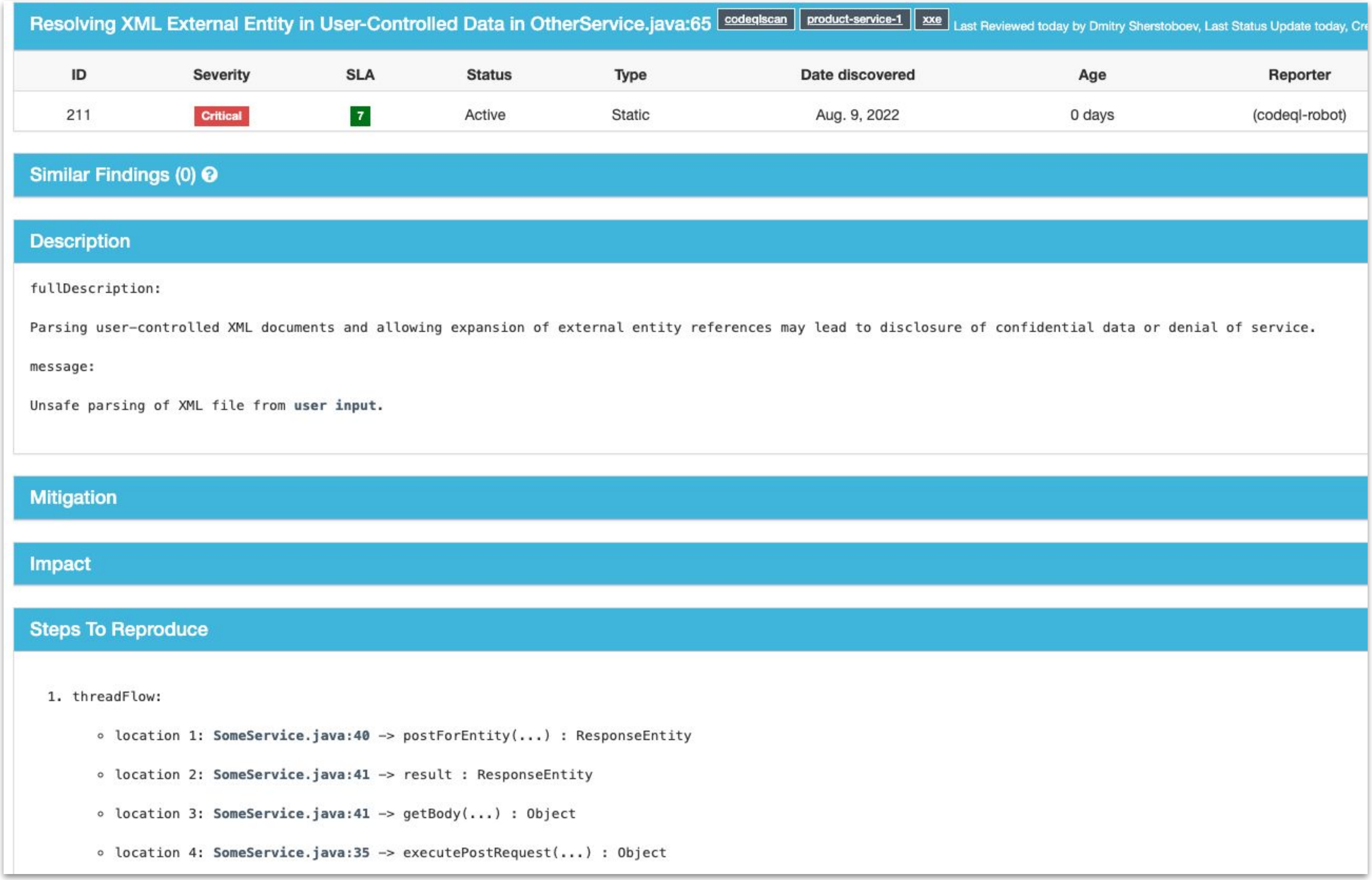
- So we need to support these projects

Scan with autogenerated code



```
.prepare_sources: &prepare_sources
  - mkdir delombok
  - java -jar "/usr/share/lombok.jar" delombok -n --onlyChanged . -d "delombok" --classpath=$(cat ./cp.txt)
  - find "delombok" -name '*.java' -exec sed '/Generated by delombok/d' -i '{}' ';'
  - find "delombok" -name '*.java' -exec sed '/import lombok/d' -i '{}' ';'
  - find "delombok" -name '*.java' -exec sed 's/@NonNull//g' -i '{}' ';'
  - cp -r "delombok/." "./"
  - rm -rf "delombok"

build_codeql:
  extends: .build_codeql
  image: $SCA_CODEQL_IMAGE
  stage: security_checks
  only:
    refs:
      - developer
      - master
    variables:
      - $SCA_LANGUAGE == "java"
  allow_failure: true
  before_script: *prepare_sources
  needs: [ ]
  tags:
```

How we upload SAST results



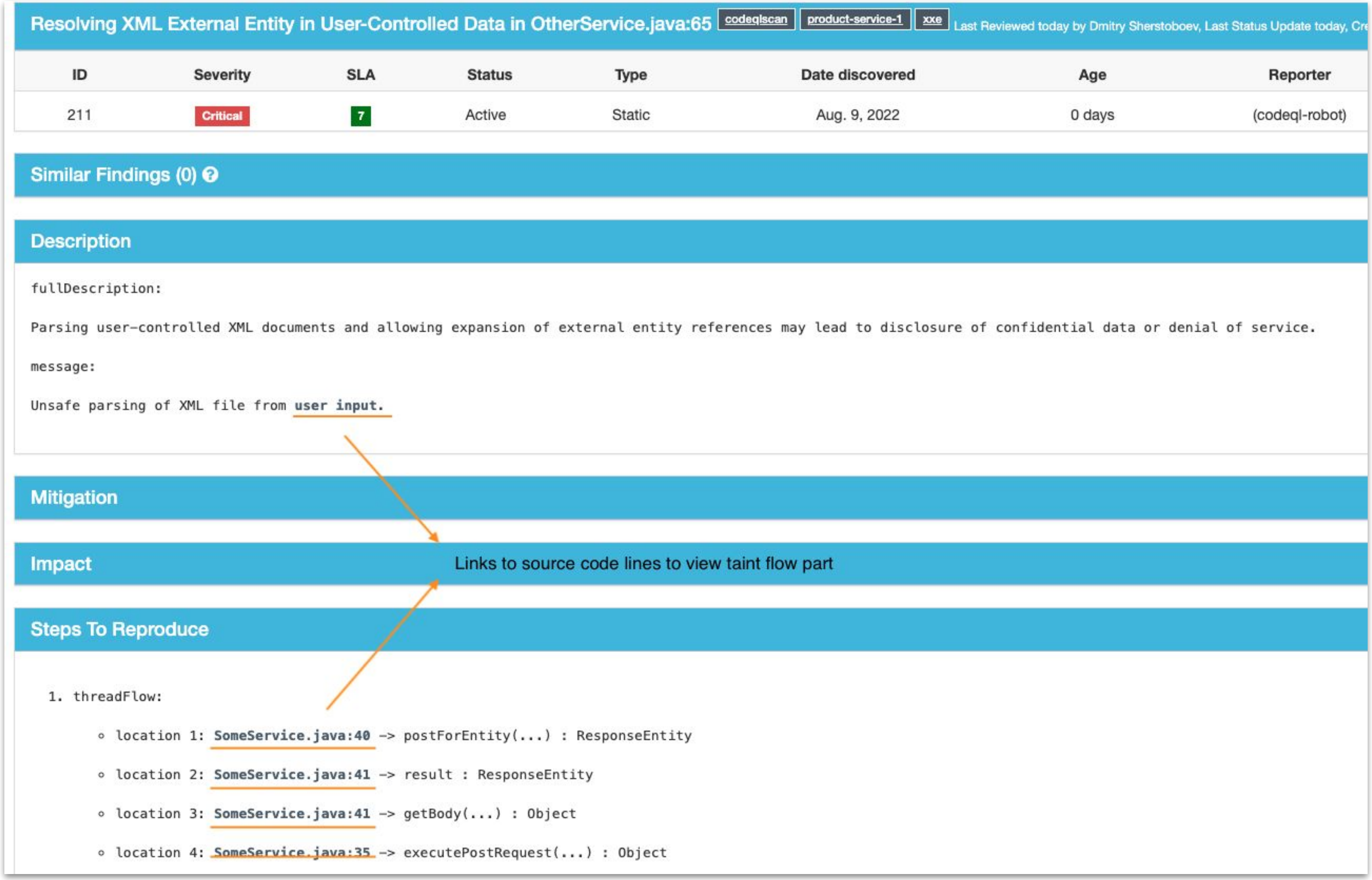
Resolving XML External Entity in User-Controlled Data in OtherService.java:65 Last Reviewed today by Dmitry Sherstoboev, Last Status Update today, Cre

ID	Severity	SLA	Status	Type	Date discovered	Age	Reporter
211	Critical	7	Active	Static	Aug. 9, 2022	0 days	(codeql-robot)

```
fullDescription:

Parsing user-controlled XML documents and allowing expansion of external entity references may lead to disclosure of confidential data or denial of service.

message:

Unsafe parsing of XML file from user input.
```

```
1. threadFlow:

   ◦ location 1: SomeService.java:40 -> postForEntity(...) : ResponseEntity

   ◦ location 2: SomeService.java:41 -> result : ResponseEntity

   ◦ location 3: SomeService.java:41 -> getBody(...) : Object

   ◦ location 4: SomeService.java:35 -> executePostRequest(...) : Object
```

How we upload SAST results



Resolving XML External Entity in User-Controlled Data in OtherService.java:65

codeqlscan | product-service-1 | xxe

Last Reviewed today by Dmitry Sherstoboev, Last Status Update today, Cre

ID	Severity	SLA	Status	Type	Date discovered	Age	Reporter
211	Critical	7	Active	Static	Aug. 9, 2022	0 days	(codeql-robot)

Similar Findings (0)

Description

```
fullDescription:

Parsing user-controlled XML documents and allowing expansion of external entity references may lead to disclosure of confidential data or denial of service.

message:

Unsafe parsing of XML file from user input.
```

Mitigation

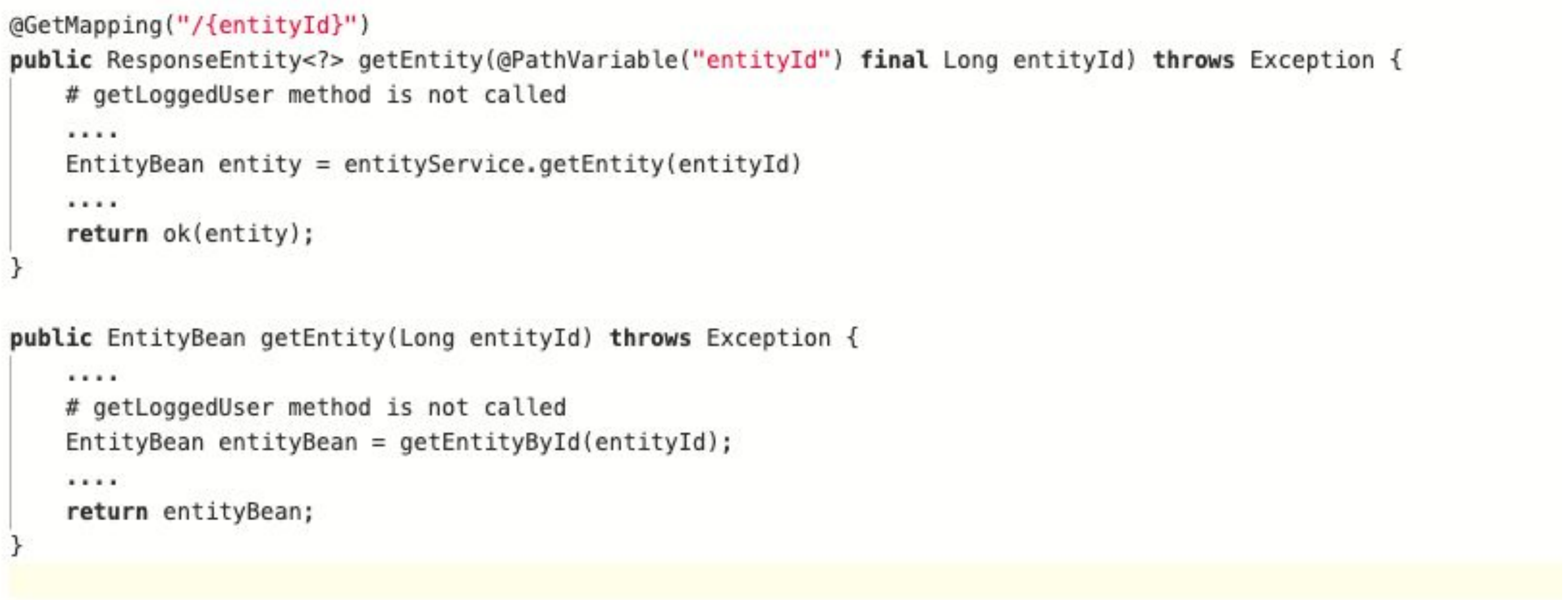
Impact

Links to source code lines to view taint flow part

Steps To Reproduce

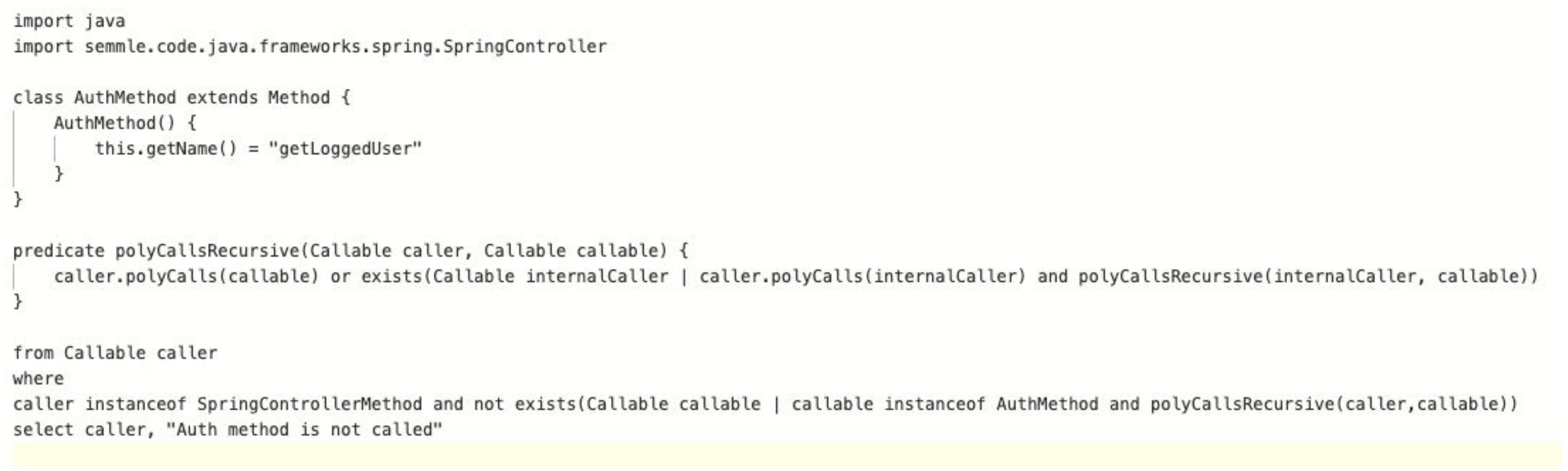
```
1. threadFlow:

   ◦ location 1: SomeService.java:40 -> postForEntity(...) : ResponseEntity

   ◦ location 2: SomeService.java:41 -> result : ResponseEntity

   ◦ location 3: SomeService.java:41 -> getBody(...) : Object

   ◦ location 4: SomeService.java:35 -> executePostRequest(...) : Object
```

Some rules (controllers w/o auth)



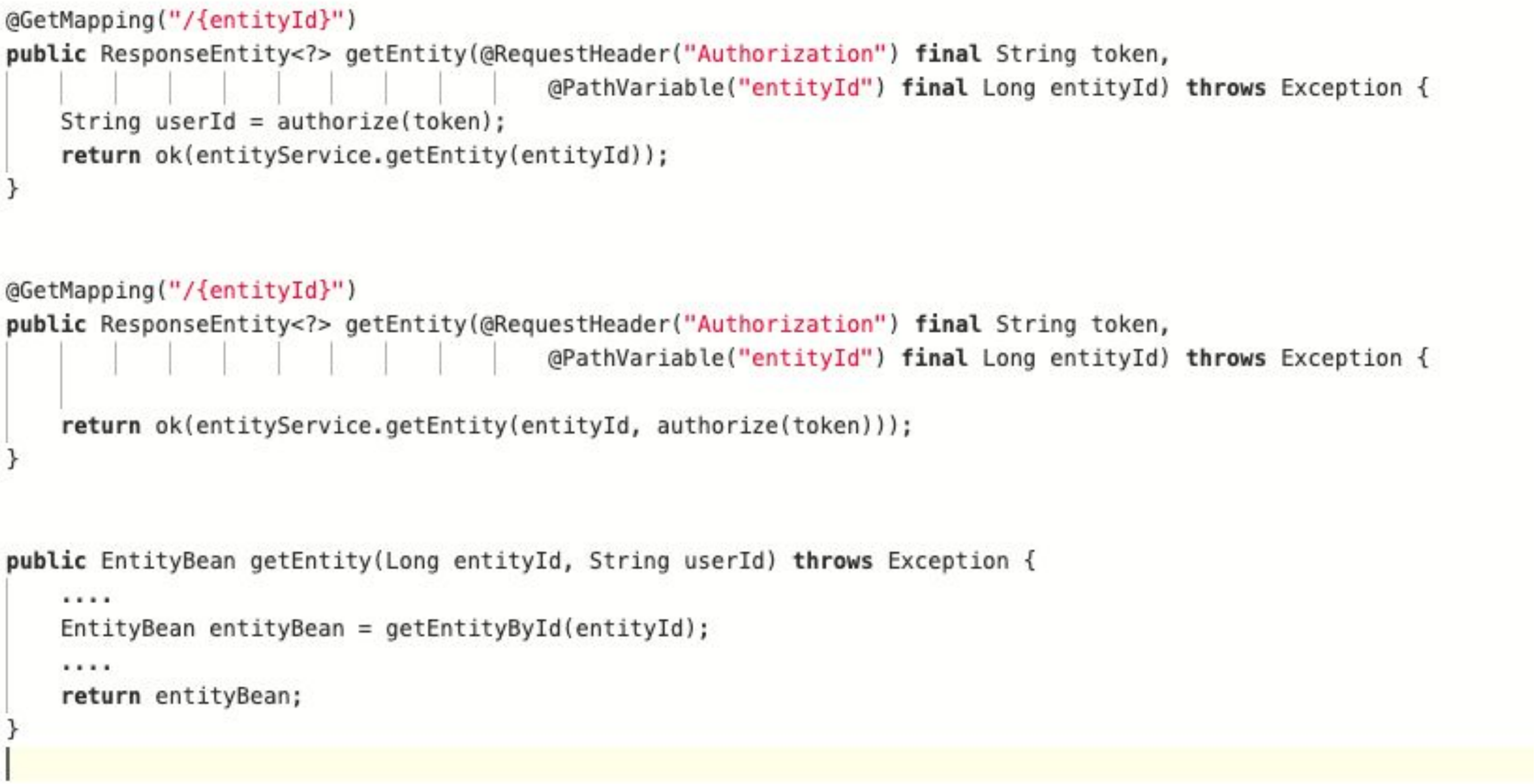
```
@GetMapping("/{entityId}")
public ResponseEntity<?> getEntity(@PathVariable("entityId") final Long entityId) throws Exception {
    # getLoggedUser method is not called
    ....
    EntityBean entity = entityService.getEntity(entityId)
    ....
    return ok(entity);
}

public EntityBean getEntity(Long entityId) throws Exception {
    ....
    # getLoggedUser method is not called
    EntityBean entityBean = getEntityById(entityId);
    ....
    return entityBean;
}
```

Some rules (controllers w/o auth)



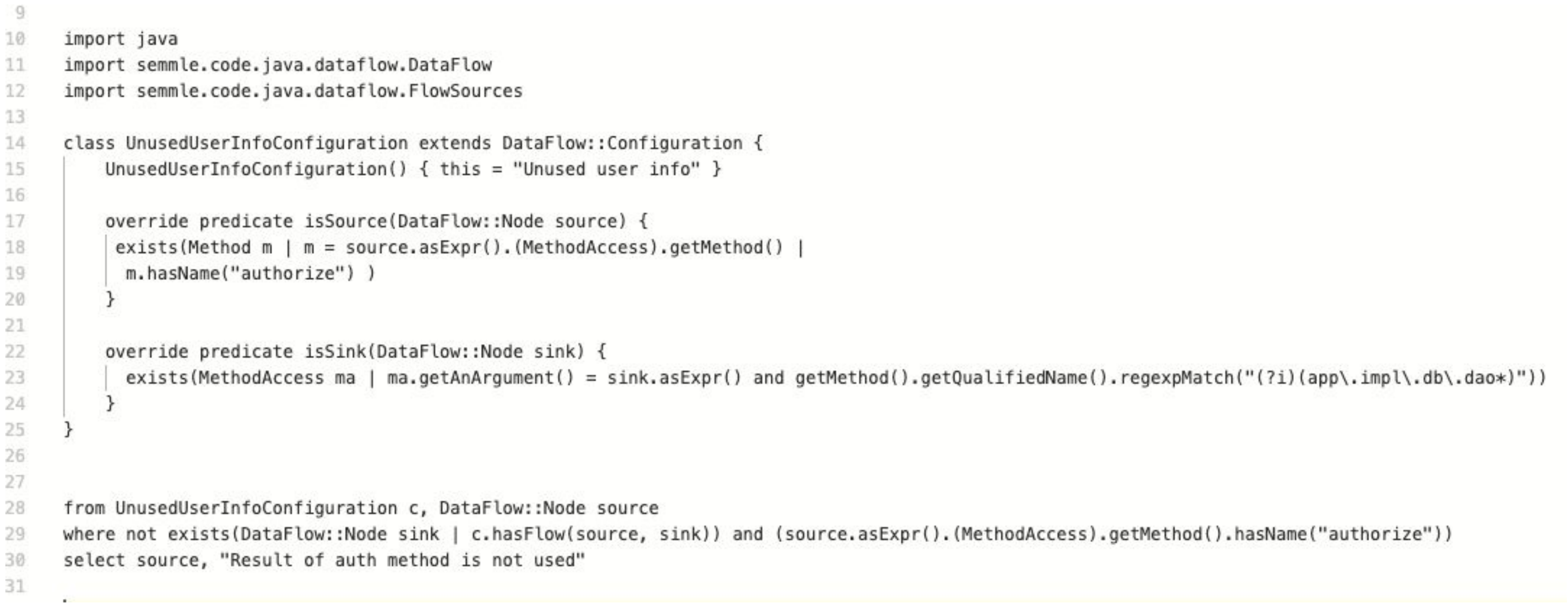
```
import java
import semmle.code.java.frameworks.spring.SpringController

class AuthMethod extends Method {
    AuthMethod() {
        this.getName() = "getLoggedUser"
    }
}

predicate polyCallsRecursive(Callable caller, Callable callable) {
    caller.polyCalls(callable) or exists(Callable internalCaller | caller.polyCalls(internalCaller) and polyCallsRecursive(internalCaller, callable))
}

from Callable caller
where
caller instanceof SpringControllerMethod and not exists(Callable callable | callable instanceof AuthMethod and polyCallsRecursive(caller,callable))
select caller, "Auth method is not called"
```

Some rules (controllers with unused auth)



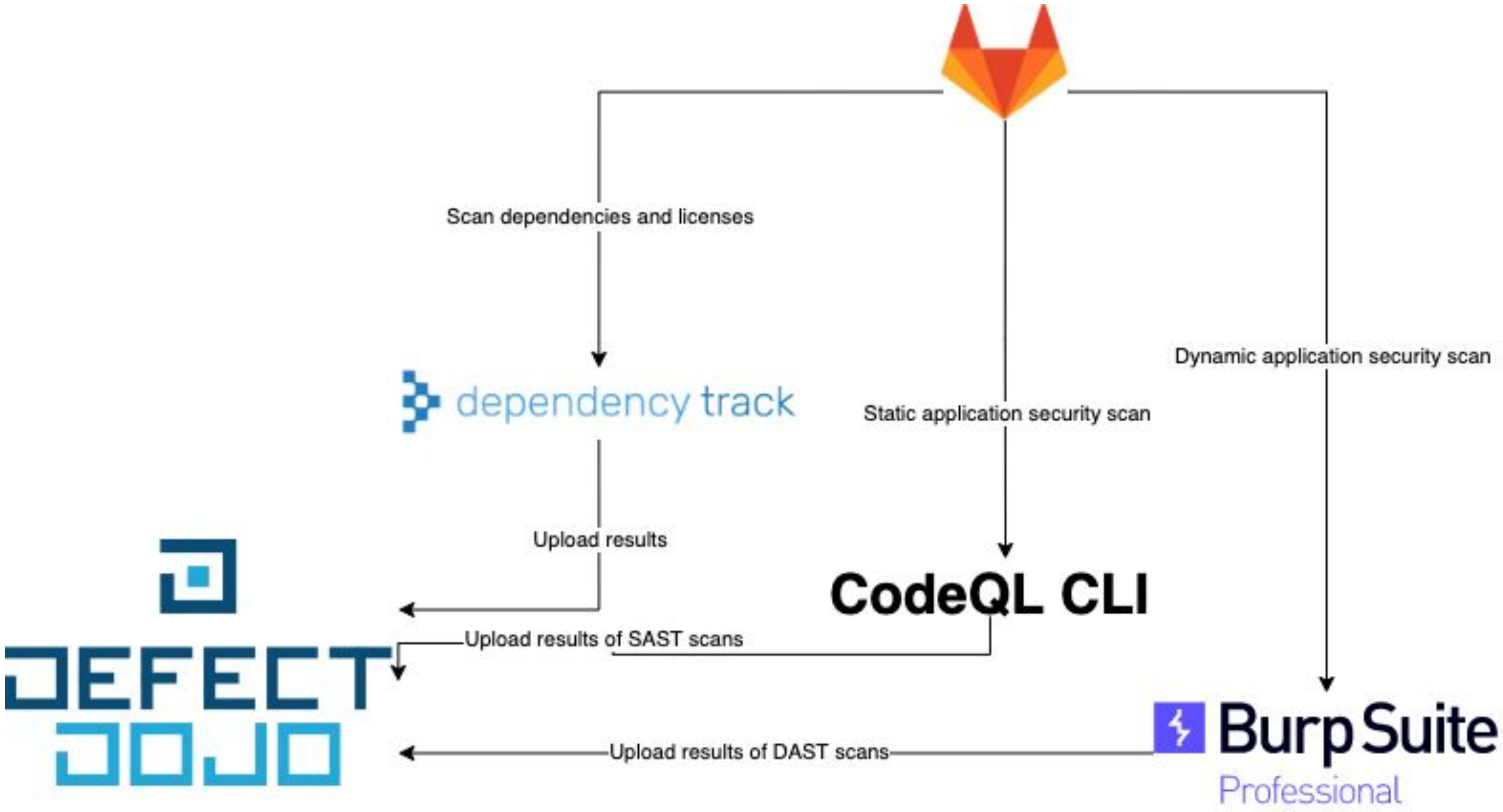
```
@GetMapping("/{entityId}")
public ResponseEntity<?> getEntity(@RequestHeader("Authorization") final String token,
                                        @PathVariable("entityId") final Long entityId) throws Exception {
    String userId = authorize(token);
    return ok(entityService.getEntity(entityId));
}


@GetMapping("/{entityId}")
public ResponseEntity<?> getEntity(@RequestHeader("Authorization") final String token,
                                        @PathVariable("entityId") final Long entityId) throws Exception {

    return ok(entityService.getEntity(entityId, authorize(token)));
}


public EntityBean getEntity(Long entityId, String userId) throws Exception {
    ....
    EntityBean entityBean = getEntityById(entityId);
    ....
    return entityBean;
}
```

Some rules (controllers with unused auth)



```
import java
import semmle.code.java.dataflow.DataFlow
import semmle.code.java.dataflow.FlowSources

class UnusedUserInfoConfiguration extends DataFlow::Configuration {
    UnusedUserInfoConfiguration() { this = "Unused user info" }

    override predicate isSource(DataFlow::Node source) {
     exists(Method m | m = source.asExpr().(MethodAccess).getMethod() |
      m.hasName("authorize") )
    }

    override predicate isSink(DataFlow::Node sink) {
      exists(MethodAccess ma | ma.getAnArgument() = sink.asExpr() and getMethod().getQualifiedName().regexpMatch("(?i)(app\.impl\.db\.dao*)"))
    }
}


from UnusedUserInfoConfiguration c, DataFlow::Node source
where not exists(DataFlow::Node sink | c.hasFlow(source, sink)) and (source.asExpr().(MethodAccess).getMethod().hasName("authorize"))
select source, "Result of auth method is not used"
```

Current state


Scan dependencies and licenses
dependency track
Static application security scan
Dynamic application security scan
Upload results
CodeQL CLI
Upload results of SAST scans
DEFECT DOJO
Upload results of DAST scans
Burp Suite Professional

OFF ONE 2022

Thank you for your attention!

ON

FF

ONE

2022