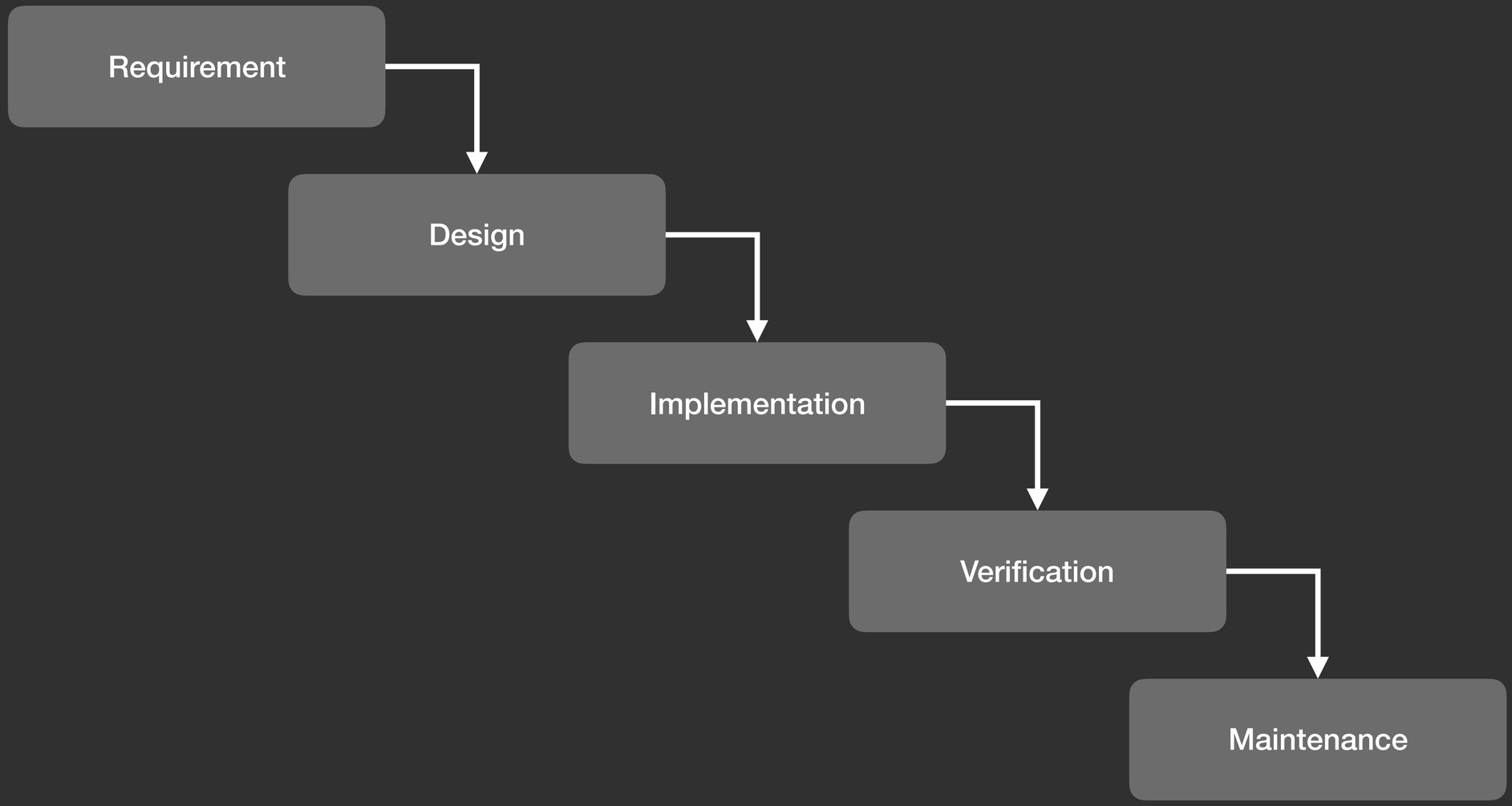
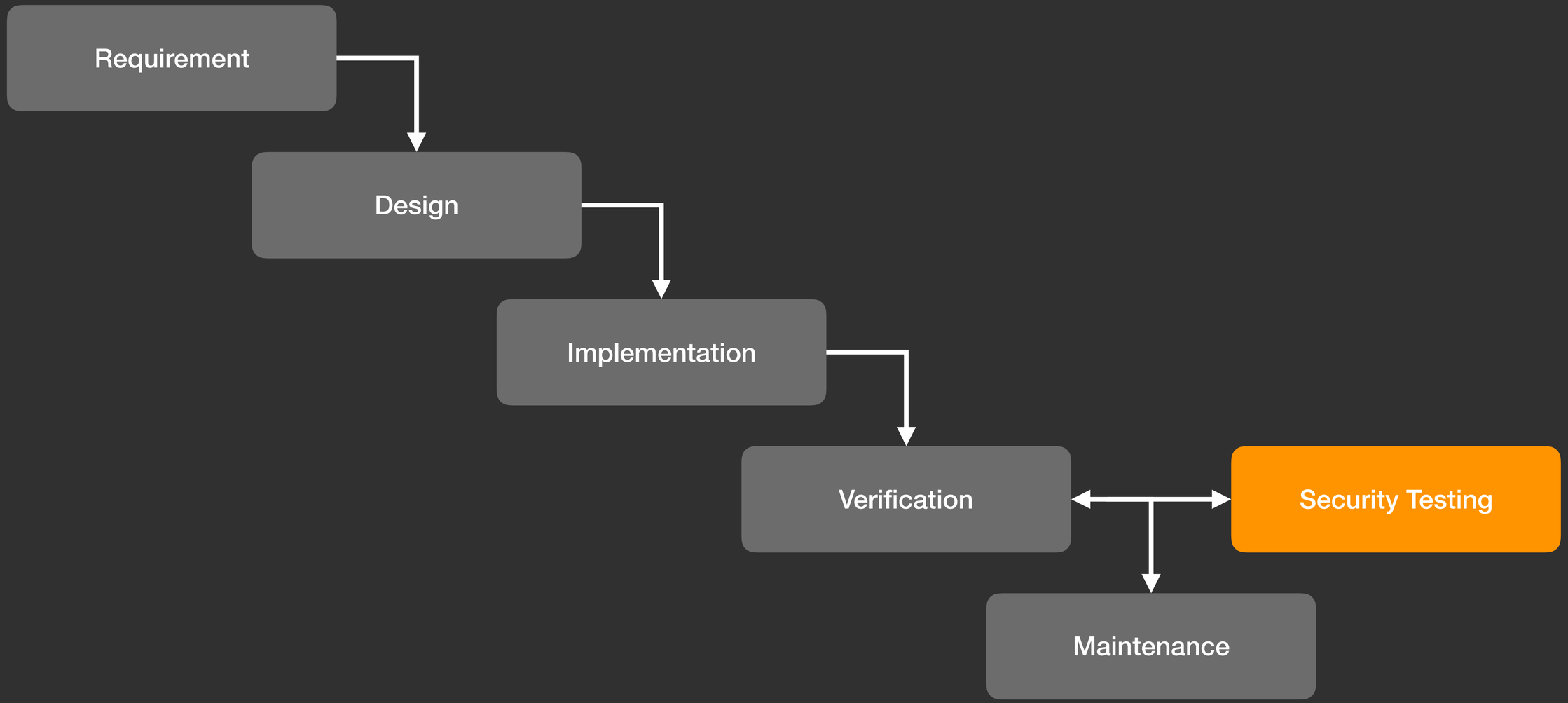
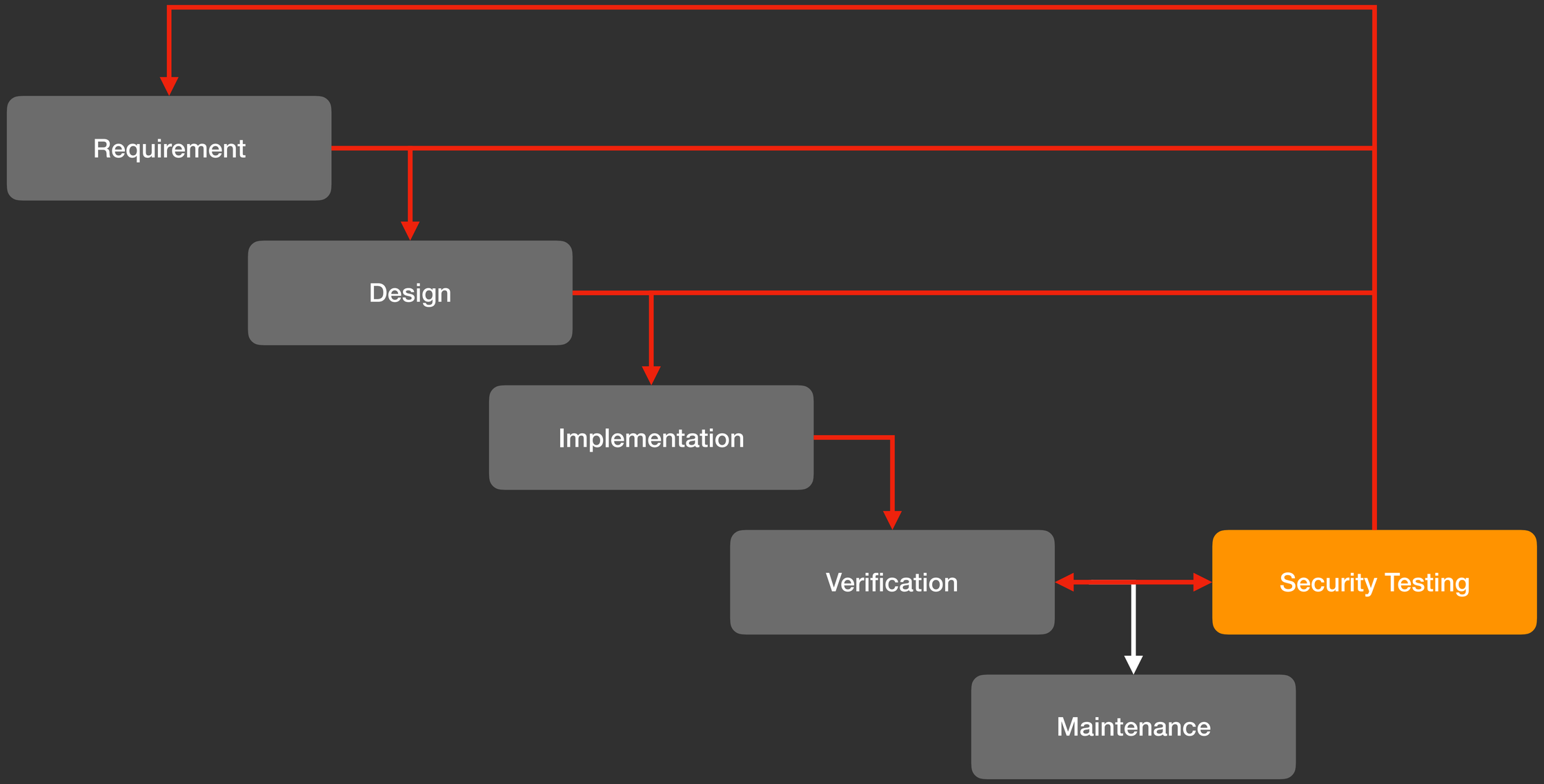
DevSecOps - 從 SAST 談持續式資安測試

YSc

HITCON DEFENSE Summit 2019


Requirement
Design
Implementation
Verification
Maintenance


Requirement
Design
Implementation
Verification
Security Testing
Maintenance


Requirement
Design
Implementation
Verification
Security Testing
Maintenance

Shift Left

DevOps

DevSecOps

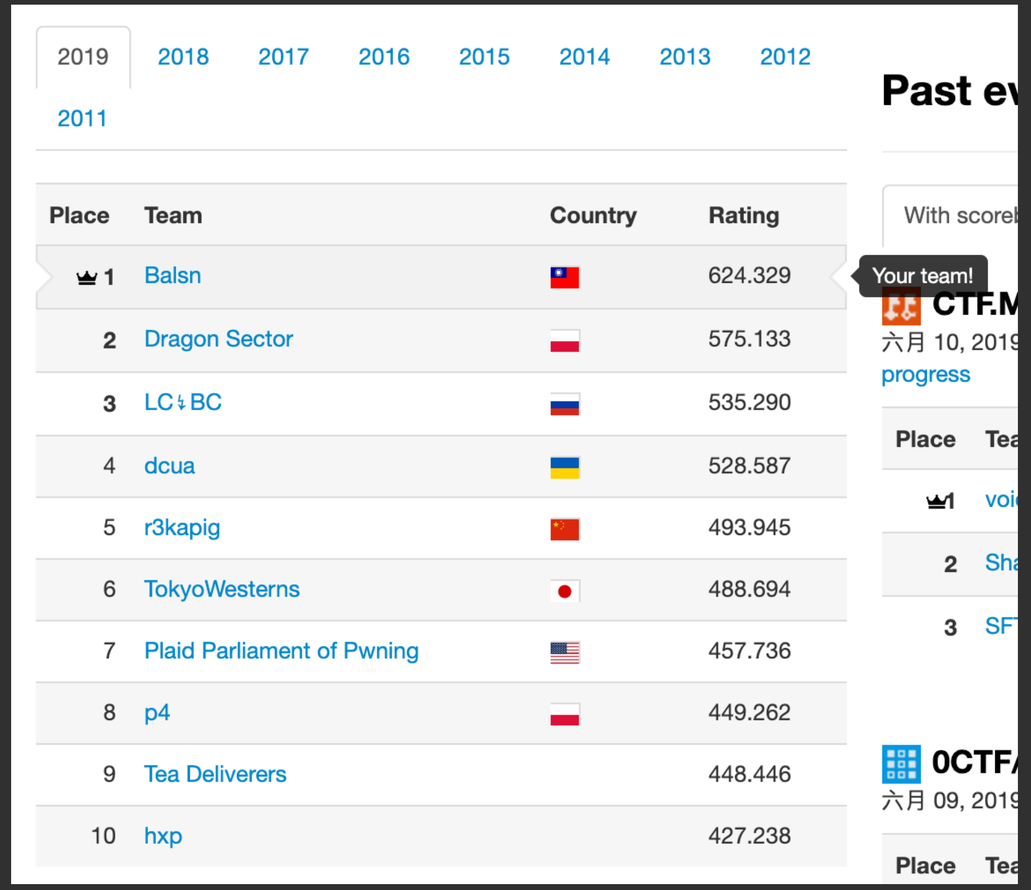
YSc

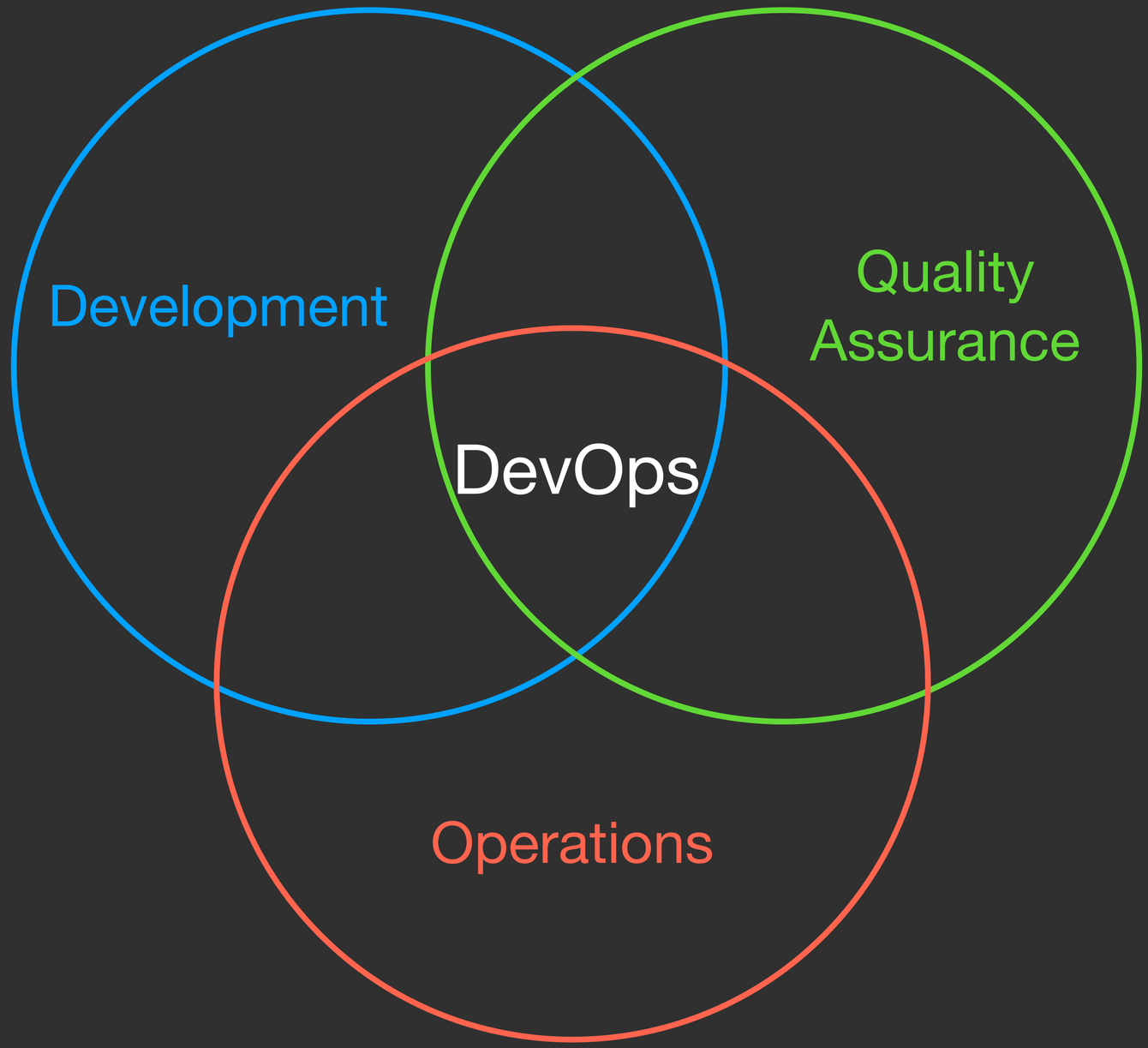
- HITCON Speaker & Trainer
- Modern Web & DevOps Days Speaker
- Balsn CTF Team Co-Founder
- 白帽觀點 https://secview.io/

Balsn

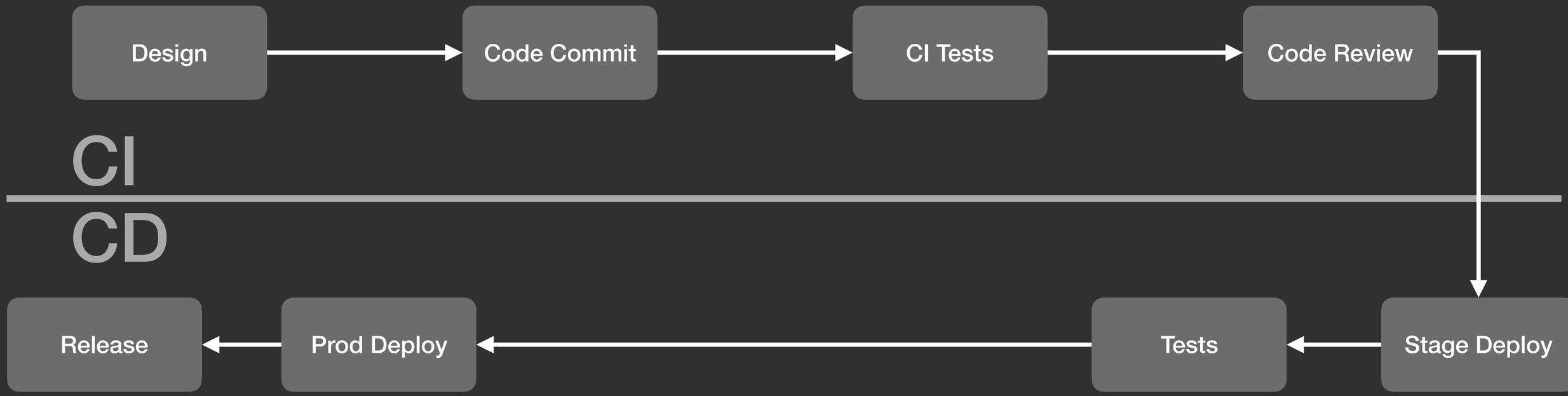
https://balsn.tw

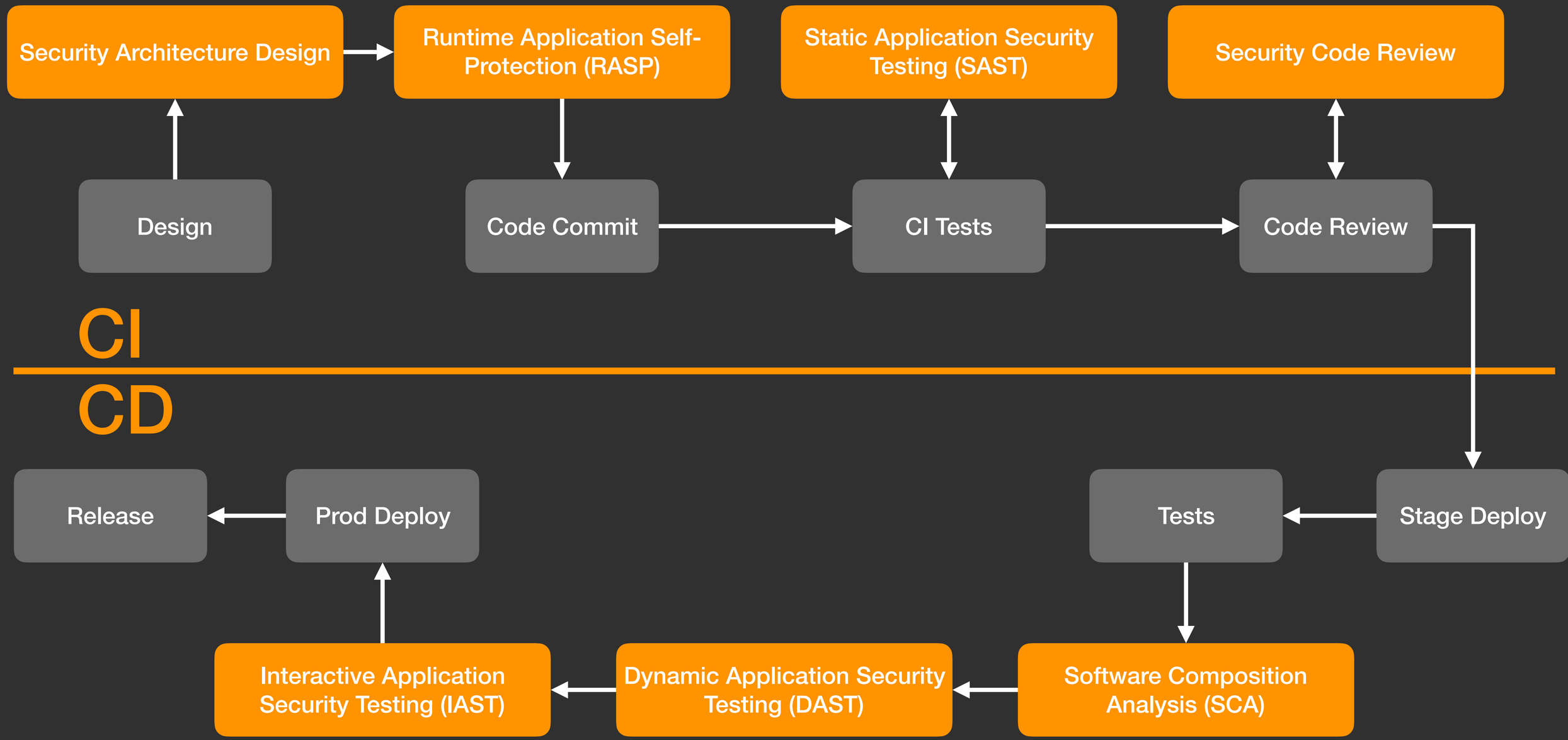
Balsn CTF 10/5 ~ 10/7



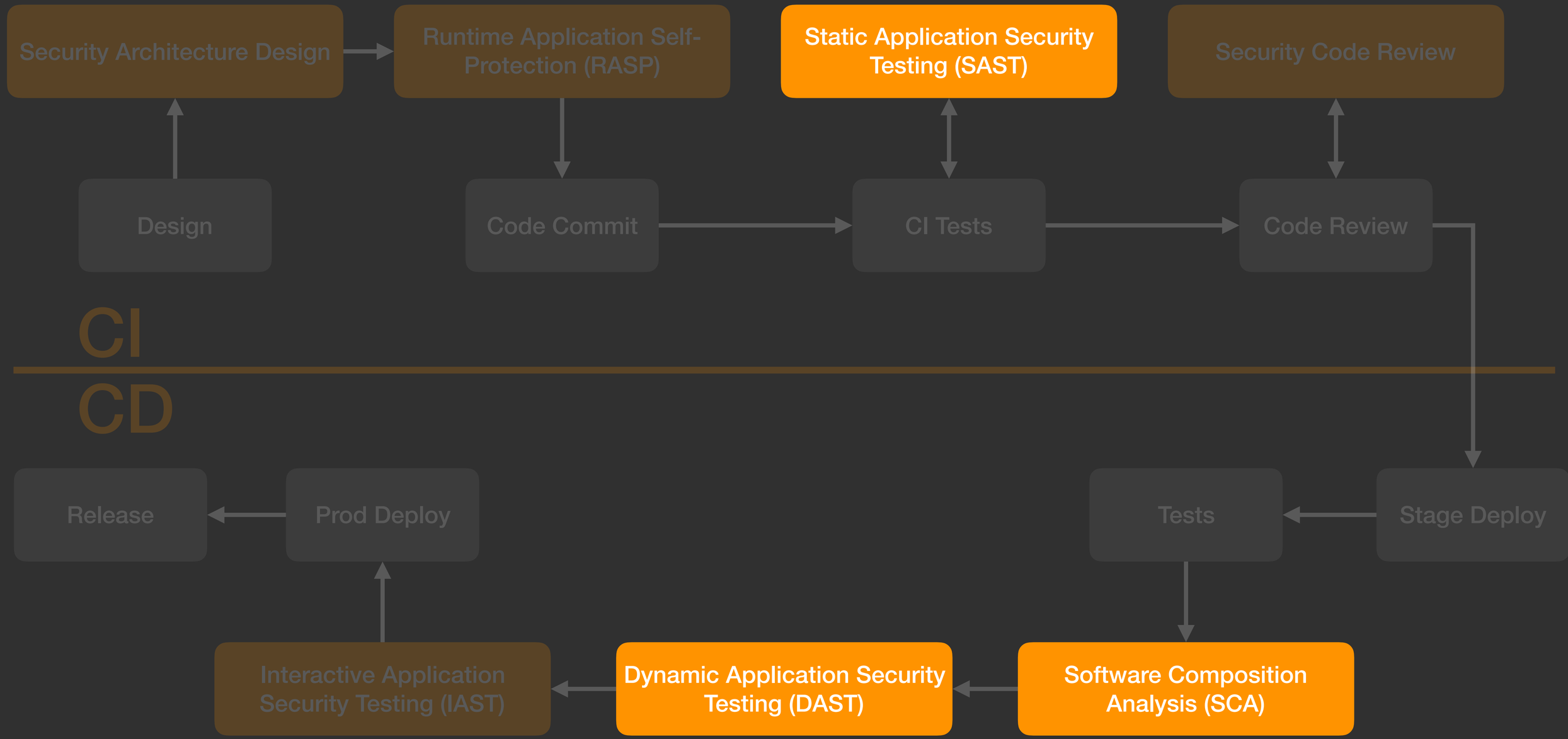

Development
Quality
Assurance
DevOps
Operations


Secure
Development
Quality & Security
Assurance
DevSecOps
Security Practice
Operations


Design
Code Commit
CI Tests
Code Review
CI
CD
Release
Prod Deploy
Tests
Stage Deploy



https://secview.io/posts/9-ci-cd/



https://secview.io/posts/9-ci-cd/

OWASP Testing Guide V4

https://www.owasp.org/index.php/OWASP_Testing_Guide_v4_Table_of_Contents

wing is the list of controls to test during the assessment:

Category	Test Name
	Information Gathering
OTG–INFO–001	Conduct Search Engine Discovery and Reconnaissance for Information L
OTG–INFO–002	Fingerprint Web Server
OTG–INFO–003	Review Webserver Metafiles for Information Leakage
OTG–INFO–004	Enumerate Applications on Webserver
OTG–INFO–005	Review Webpage Comments and Metadata for Information Leakage
OTG–INFO–006	Identify application entry points
OTG–INFO–007	Map execution paths through application
OTG–INFO–008	Fingerprint Web Application Framework
OTG–INFO–009	Fingerprint Web Application
OTG–INFO–010	Map Application Architecture
	Configuration and Deploy Management Testing
OTG–CONFIG–001	Test Network/Infrastructure Configuration
OTG–CONFIG–002	Test Application Platform Configuration
OTG–CONFIG–003	Test File Extensions Handling for Sensitive Information
OTG–CONFIG–004	Backup and Unreferenced Files for Sensitive Information
OTG–CONFIG–005	Enumerate Infrastructure and Application Admin Interfaces
OTG–CONFIG–006	Test HTTP Methods
OTG–CONFIG–007	Test HTTP Strict Transport Security
OTG–CONFIG–008	Test RIA cross domain policy
	Identity Management Testing
OTG–IDENT–001	Test Role Definitions
OTG–IDENT–002	Test User Registration Process
OTG–IDENT–003	Test Account Provisioning Process
OTG–IDENT–004	Testing for Account Enumeration and Guessable User Account
OTG–IDENT–005	Testing for Weak or unenforced username policy

OWASP Testing Guide V4

- Information Gathering
- Configuration and Deploy Management Testing
- Identity Management Testing
- Authentication Testing
- Authorization Testing
- Session Management Testing
- Input Validation Testing
- Error Handling
- Cryptography
- Business Logic Testing
- Client Side Testing

高 CP 值測試優先（低成本、高嚴重性）

將測試放在適合的步驟上


預防
自動化測試
滲透測試
漏洞獎勵計劃
未找到
利用

- 風險評估（Risk Assessment）
    - 判斷哪裡存在危險性、找出真正高風險
- 資安測試（Security Testing）
    - OWASP Testing Guide、自動化測試、持續檢測可能被利用的弱點
- 漏洞管理（Vulnerability Management）
    - 分析漏洞的嚴重性、評估是否可被利用、提供適合的修復時程和修補方法

整合要點

- 調校工具參數、範圍、測試速度
- 將測試容器化、自動化
- 持續疊代

Tools

- Sensitive information: https://github.com/dxa4481/truffleHog
- Static Application Security Testing (SAST): https://github.com/PyCQA/bandit
- Software Composition Analysis (SCA): https://snyk.io/

Sensitive Information

- 寫死在程式裡的密碼、金鑰、存取權杖、密鑰檔案等等
- 也可以在 Pre-commit Hook 或 IDE Plugin 加入檢測

- AWS Access Key ID & Secret
- Slack API Token & Webhook
- Github Client Secret
- Heroku API key
- ...

- Facebook Access Token
- Google Oauth Service Account
- Twitter API Secret
- Paypal client id and secret key
- ...

Over 100,000 code repositories on source code management site GitHub contain secret access keys that can give attackers privileged access to those repositories (repos) or to online service providers' services.

Researchers at North Carolina State University (NCSU) scanned almost 13% of GitHub's public repositories over nearly six months. In a paper revealing the findings, they said:

> *We find that not only is secret leakage pervasive – affecting over 100,000 repositories – but that thousands of new, unique secrets are leaked every day.*

The credentials that developers routinely publish on their GitHub repos fall into several categories. These include SSH keys, which are digital certificates that automatically unlock online resources. Another is application programming interface (API) keys (also known as tokens). These are digital keys that enable developers to access online services ranging from Twitter to Google Search directly

Thousands of keys leaking on GitHub every day

https://nakedsecurity.sophos.com/2019/03/25/thousands-of-coders-are-leaving-their-crown-jewels-exposed-on-github/

```
$ trufflehog \
--max_depth 3 \
--entropy=False \
--regex \
--include_paths ./inc.txt \
--exclude_paths ./exc.txt \
https://github.com/DefectDojo/django-DefectDojo.git
```

```
Reason: Password in URL
Date: 2019-11-09 20:26:12
Hash: 08946726bf6f4fcde07a18044644a62f7ec94e36
Filepath: docker-compose.yml
Branch: origin/master
Commit: Merge pull request #1642 from Maffooch/mast

[Master] Modify installation README's
mysql://defectdojo:defectdojo@mysql:3306/defectdojo

~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Reason: Generic API Key
Date: 2019-11-09 20:26:12
Hash: 08946726bf6f4fcde07a18044644a62f7ec94e36
Filepath: docker/sample_data/initial_dojo_data.json
Branch: origin/master
Commit: Merge pull request #1642 from Maffooch/mast

[Master] Modify installation README's
apikey", "app_label": "tastypie"}, "model": "conter
0}, {"fields": {"model": "contact", "app_label": "c
es.contenttype", "pk": 12}, {"fields": {"model": "p
dojo"}, "model": "contenttypes.contenttype", "pk":
duct", "app_label": "dojo"}, "model": "contenttypes
"fields": {"model": "scan", "app_label": "dojo"}, "
", "pk": 19}, {"fields": {"model": "engagement_type
el": "contenttypes.contenttype", "pk": 21}, {"fielc
": "dojo"}, "model": "contenttypes.contenttype", "p
elopment_environment", "app_label": "dojo"}, "model
```

Sensitive Information

- 調校工具很重要
  - 增加速度、降低 False Positive
- 直接使用：7m44.222s
- 調校過後：0m2.356s

Static Application Security Testing (SAST)

- Security linter、自動化白箱測試
- 檢測危險函數、注入、弱加密演算法等

$ bandit -r .
$ bandit -lll -ii --exclude ./tests -r .

```
3       import logging

--------------------------------------------------
>> Issue: [B320:blacklist] Using lxml.etree.parse
   Severity: Medium   Confidence: High
   Location: ./tests/validate_acunetix_scan_xml.p
   More Info: https://bandit.readthedocs.io/en/la
14          try:
15              tree = etree.parse(filename)
16              return tree.getroot()

--------------------------------------------------

Code scanned:
        Total lines of code: 34258
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0.0
                Low: 38.0
                Medium: 138.0
                High: 4.0
        Total issues (by confidence):
                Undefined: 0.0
                Low: 0.0
                Medium: 5.0
                High: 175.0
Files skipped (0):
```

B3xx Reference

B301 pickle
B302 marshal
B303 md5
B304 ciphers
B305 cipher_modes
B306 mktemp_q
B307 eval
B308 mark_safe
B309 httpsconnection
B310 urllib_urlopen
B311 random
B312 telnetlib
B313 xml_bad_cElementTree
B314 xml_bad_ElementTree
B315 xml_bad_expatreader
B316 xml_bad_expatbuilder
B317 xml_bad_sax
B318 xml_bad_minidom
B319 xml_bad_pulldom
B320 xml_bad_etree
B321 ftplib
B322 input
B323 unverified_context
B324 hashlib_new_insecure_functions
B325 tempnam

B6xx, B7xx Reference

B601 paramiko_calls
B602 subprocess_popen_with_shell_equals_true
B603 subprocess_without_shell_equals_true
B604 any_other_function_with_shell_equals_true
B605 start_process_with_a_shell
B606 start_process_with_no_shell
B607 start_process_with_partial_path
B608 hardcoded_sql_expressions
B609 linux_commands_wildcard_injection
B610 django_extra_used
B611 django_rawsql_used
B701 jinja2_autoescape_false
B702 use_of_mako_templates
B703 django_mark_safe

Other Issues

B201 flask_debug_true
B506 yaml_load
B507 ssh_no_host_key_verification
...

Static Application Security Testing (SAST)

- Baseline
- Critical、High 優先處理
- 搭配漏洞管理工具（e.g. DefectDojo）進行統整和評估
  - 已確定有問題的漏洞：評估嚴重程度、確保有漏洞程式碼不進入 repo
  - 增加風險的弱點：評估修復難易度、降低風險
- 偽陽性（False Positive）高，需要調校工具、使用漏洞管理工具優化
- 快速檢測、測試時間盡可能短

Software Composition Analysis (SCA)

- 現在開發幾乎都是基於函式庫或框架來開發
- 相依性套件分析：第三方函式庫、開發框架、Docker 映象檔、系統軟體套件
- 檢測已知漏洞（CVE）、惡意軟件

The npm Blog

Blog about npm things.

Reported malicious module: getcookies

Early May 2nd, the npm security team received and responded to reports of a package that masqueraded as a cookie parsing library but contained a malicious backdoor. The result of the investigation concluded with three packages and three versions of a fourth package being unpublished from the npm Registry.

No packages published to the npm Registry used the malicious modules in a way that would have allowed the backdoor to be triggered. Applications not published to the registry that directly required the malicious modules might have been vulnerable, but are out of the scope of our analysis.

Initial report

Initial information from the community reported that the package `getcookies` contained a potential backdoor, that `express-cookies` and `http-fetch-cookies` depended upon getcookies, and that a popular package, `mailparser`, depended upon http-fetch-cookies.

Triage

Upon receiving the report, npm's security team started triage. The goal of triage was determining

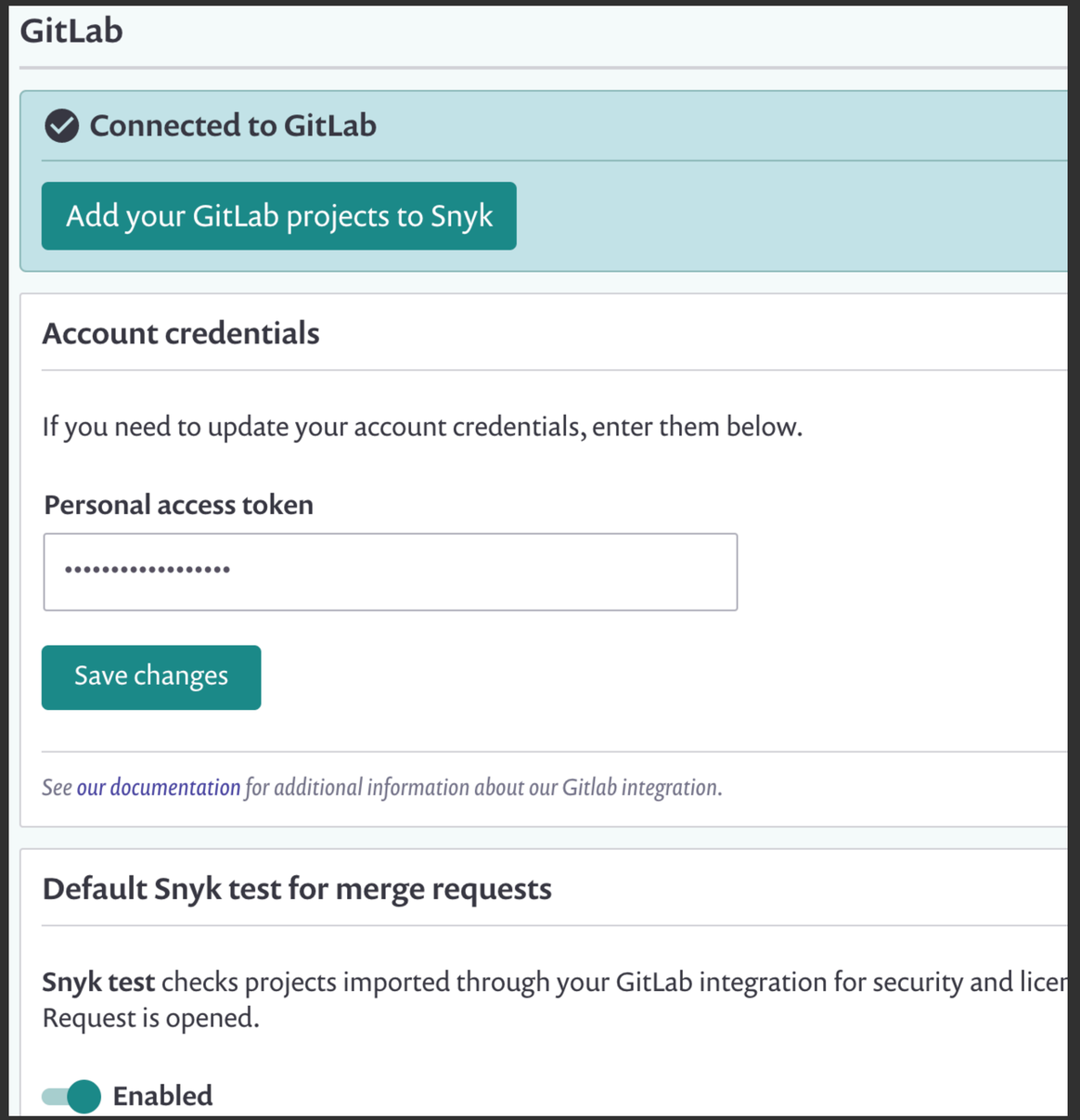
尚無安全補丁，（下表）包含了Angular或React樣板的漏洞，也就是說，開發者的應用程式因為引用了這些元件，因此在未做任何事之前就存在漏洞，而且都有未修補的漏洞。Angular的相依項目有952個，總共有2個漏洞，React則有1,257個相依項目，存在3個漏洞，還有一個為潛在的授權相容問題。

Boilerplate	Vulnerable module	Indirect vulnerability	Indirect vulnerability severity	Yearly module downloads	Fixable?
Angular	jasmine-core	ReDoS	low	94,559,055	✓
Angular	useragent	ReDoS	high	70,181,373	✗
React	lodash	Prototype Pollution	high	1,005,518,049	✓
React	mdn-data	MPL-2.0 License issue	high	89,291,454	✗
React	mixin-deep	Prototype Pollution	high	328,052,052	✓
React	set-value	Prototype Pollution	high	629,781,760	✓

Snyk 釋出最新 JavaScript 框架安全性報告，不少熱門框架模組存在 XSS 漏洞

https://www.ithome.com.tw/news/134029

Only fail for high severity issues
Only fail when the issues found have a fix available



Software Composition Analysis (SCA)

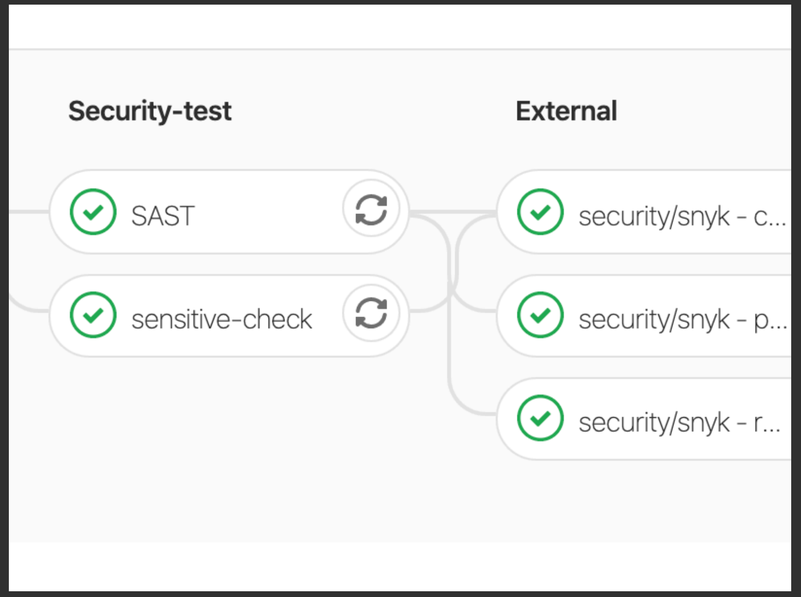
- 持續檢查相依性套件漏洞
- 限制、驗證可信的第三方套件
- 定期更新套件

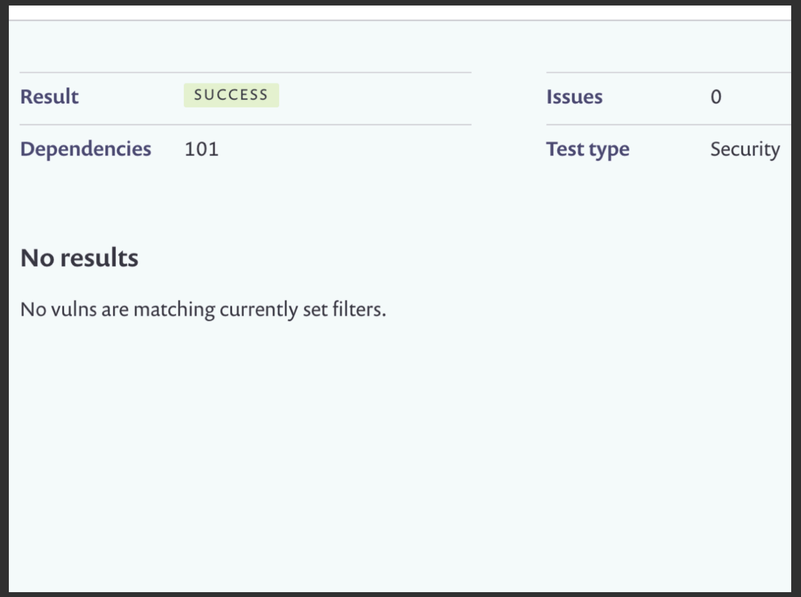
Configuration

- Infrastructure as Code
- 快速部署、固定部署環境
- 錯誤配置檢測
  - Nginx configuration: https://github.com/yandex/gixy
  - 雲端環境、Kubernetes 設定檔檢查
- 自動化合規（Automated Compliance）

Contents of .gitlab-ci.yml

```yaml
stages:
  - linter
  - security-test
  - deploy

image: python:3.7.5-alpine3.9

linter:
  stage: linter
  script:
    - "/bin/sh ci/linter.sh"

sensitive-check:
  stage: security-test
  script:
    - "/bin/sh ci/sensitive-check.sh"

SAST:
  stage: security-test
  script:
    - "/bin/sh ci/sast.sh"

deploy:
  stage: deploy
  script:
```





工具總結

- 持續且自動化整合
- 檢查 baseline
- 漏洞管理
  - 初期先只專注在 Critical、High 等級的漏洞
  - 降低 False Positive、降低潛在風險
- 風險評估、威脅模型的定義，更能發揮工具的價值
- 文化導向、資安是每個人的權責

Shift Left

及早發現

持續檢測

降低成本

Security is EVERYONE's job

– AWS CTO Werner Vogels

白帽觀點

https://secview.io

Thanks