Information Security

Double Lee/ Brad Wang/ Steve Y Lin/
Hao YH Lin/ Bill Yang

2022.Sep

Agenda

- 資安事件演示
- 密碼學基礎概念
- 開發程式相關
- CI相關
- CD相關

資安事件演示

謀攻：
「知已知彼，方能百戰不殆」

開發人員：

耗盡心思，讓系統照著需求動

駭客：

耗盡心思，讓系統照駭客的方式動

威脅建模 Threat Modeling - STRIDE

	代表涵義	描述	軟體安全特性
S	偽冒 (Spoofing)	假冒為某人或某物	身分驗證 (Authentication)
T	竄改 (Tampering)	惡意的修改資料	完整性 (Integrity)
R	否認行為 (Repudiation)	使用者可以否認曾經進行某行為	不可否認性 (Non-Repudiation)
I	資訊洩漏 (Information Disclosure)	資訊被洩漏給不被預期可存取的個體	機密性 (Confidentiality)
D	拒絕存取服務 (Denial of service)	對使用者拒絕服務或降低服務水準	可用性 (Availability)
E	權限提升 (Elevation of privilege)	未正常授權取得權限能力	授權 (Authorization)

常見攻擊方式

- 資訊洩露
  - 文件曝露
  - 爬蟲攻擊
- 命令執行
  - SQL Injection
  - 文檔上傳
  - 前端竄改
- 偽裝攻擊
  - XSS
  - CROS
  - CSRF
- 資訊竊取
  - 封包欄截
  - 用戶端監聽

資訊曝露

進攻難度：★★☆☆☆

防禦難度：★★☆☆☆

Server Error in '/' Application.

未指定的错误

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the

Exception Details: System.Data.OleDb.OleDbException: 未指定的错误

Source Error:

```
Line 27:     public string Show_storid(int id)
Line 28:     {
Line 29:         if (con.State == ConnectionState.Closed) { con.Open(); }
Line 30:         string sort_name_id = string.Empty;
Line 31:         string sql = "select top 1 * from system_text_info where id =" + id + " ";
```

Source File: d:\virtualhost\net3843533\www\Details.aspx.cs **Line:** 29

Stack Trace:

```
[OleDbException (0x80004005): 未指定的错误]
```

URL嘗試 DEMO

進攻難度：★☆☆☆☆

防禦難度：★★★☆☆

科技公司工程師張啟元：「我必需跟創辦人馬克祖克柏說聲抱歉，因為不是惡意的。」

向臉書創辦人道歉，他是科技公司工程師張啟元，今

克祖克柏的頁面，輕鬆刪除他1年前的發文，引發網

會有貼文的ID，只要連上這個ID，改那個參數，就可

張啟元說，日前因為貼不雅照被臉書通知刪除，但是

號，竟然可以任意刪除別人的發文，向官方反應不受

還是有跟他們回報，但是他們不重視，我就先去祖克

張啟元說，臉書明明鼓勵網友通報，確認有漏洞就會修補，並提供500美金以上的獎金，

技系學生，也能連續抓出20幾項臉書安全漏洞，反應後都不被臉書重視，才會刪除祖克柏

SQL Injection DEMO

進攻難度：★★☆☆☆

防禦難度：★☆☆☆☆

正常流程

帳號：double

密碼：135246

```
select * from userInfo where userid = '' and password = ''
```

```
sql = "select * from userInfo where userid = ' " +  textUser
+ " ' and password = ' " + textPass + " ' ";
```

攻擊流程

帳號：' or 1=1 #

密碼：000

```
select * from userInfo where userid = '' or 1=1 #' and password = '000'
```

Logs

Range: Custom 2021/10/8 to 2021/10/8

Type: Virus/Malware All results (2)

1-2

Date/Time	Infected File/Object	Security Threat	Result	Scan Type	F
2021/10/8 (Fri) 13:33	shell.php	PHP_Generic	Cleaned	Real-time Scan	C
2021/10/8 (Fri) 13:33	shell.php	PHP_Generic	Cleaned	Real-time Scan	C

刚才台北有发有个邮件

下午 05:57

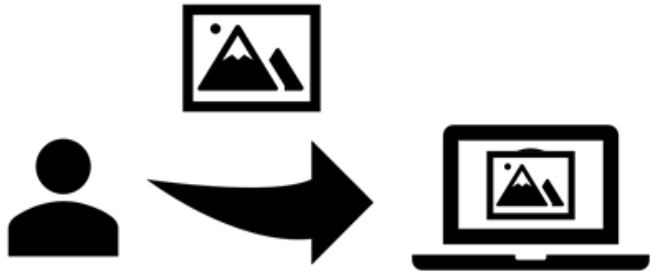
MxDR有偵測&通知 WKS同仁Double Lee電腦上，今日下午13:33時，C:\Users\10605123\Downloads\ 資料夾中有一個網頁後門程式shell.php

檔案上傳 DEMO

進攻難度：★★★★☆

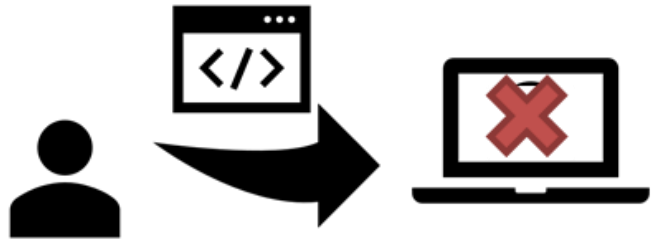
防禦難度：★★★☆☆

正常流程



http://.../upload.jpg

攻擊流程



http://.../upload.aspx

http://.../upload.sh

所有票券通通1元

iThome 新聞 產品&技術 專題 AI Cloud DevOps 資安 研討會 社群 IT EXPLAINED 搜尋

ibon售票系統遭爆有漏洞，專家：曝露網站設計不嚴謹的老問題

統一超商的ibon售票系統旅遊專區遭發現有漏洞，利用瀏覽器檢視程式碼就能竄改票價、結算金額，就能以1元買到8張票。資安專家認為這曝露出國內常見的不夠嚴謹問題，系統設計上忽視對回傳數值的驗證，才會導致這樣離譜的問題發生。

文/ 蘇文彬 | 2015-09-17 發表

張啟元是利用開發者常用的檢查網頁元素功能（在瀏覽器按下F12）檢視ibon旅遊售票網頁的程式碼，發現竟然可以修改票價金額，再以修改的金額完成線上訂票，最後他成功以1元買到8張票，引起國內媒體關注，紛紛報導。張啟元先前曾因發現到一個可刪除臉書創辦人Mark Zuckerberg文章的漏洞而聲名大噪。

https://www.ithome.com.tw/news/98753

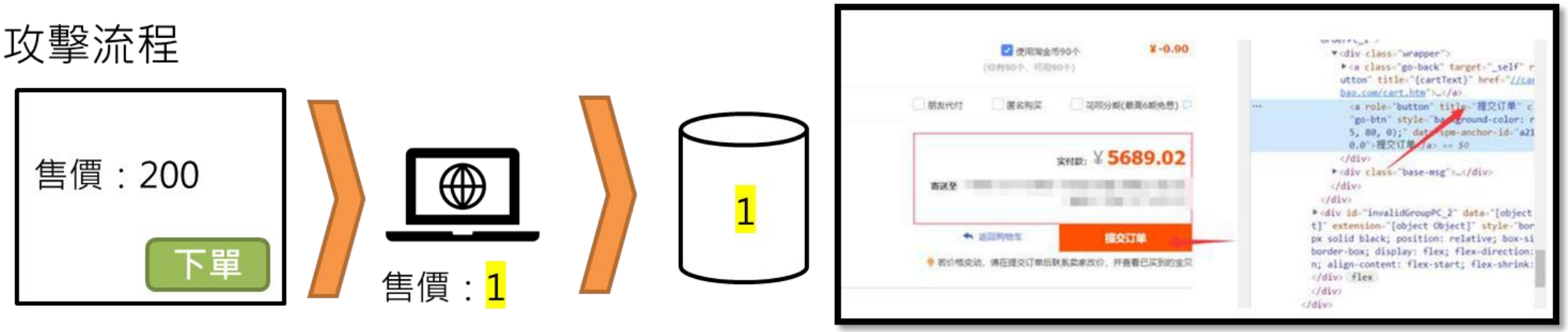
前端竄改 DEMO

進攻難度：★★☆☆☆

防禦難度：★★★☆☆

正常流程

攻擊流程



ibon爆漏洞？所有票券統統1元

https://www.ithome.com.tw/news/98753

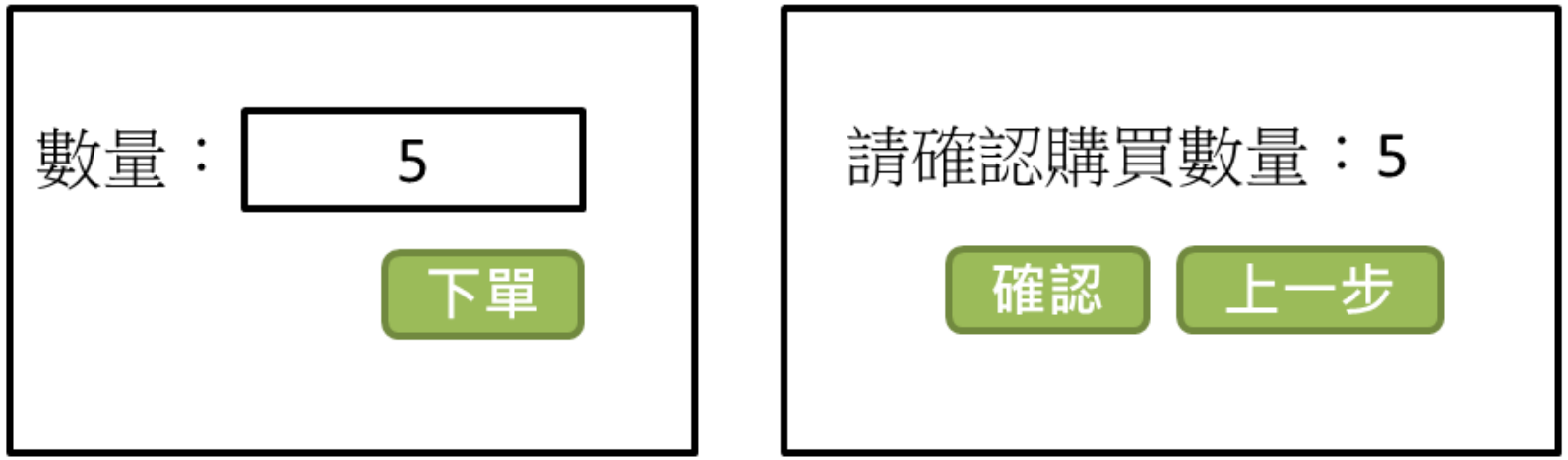
XSS

DEMO

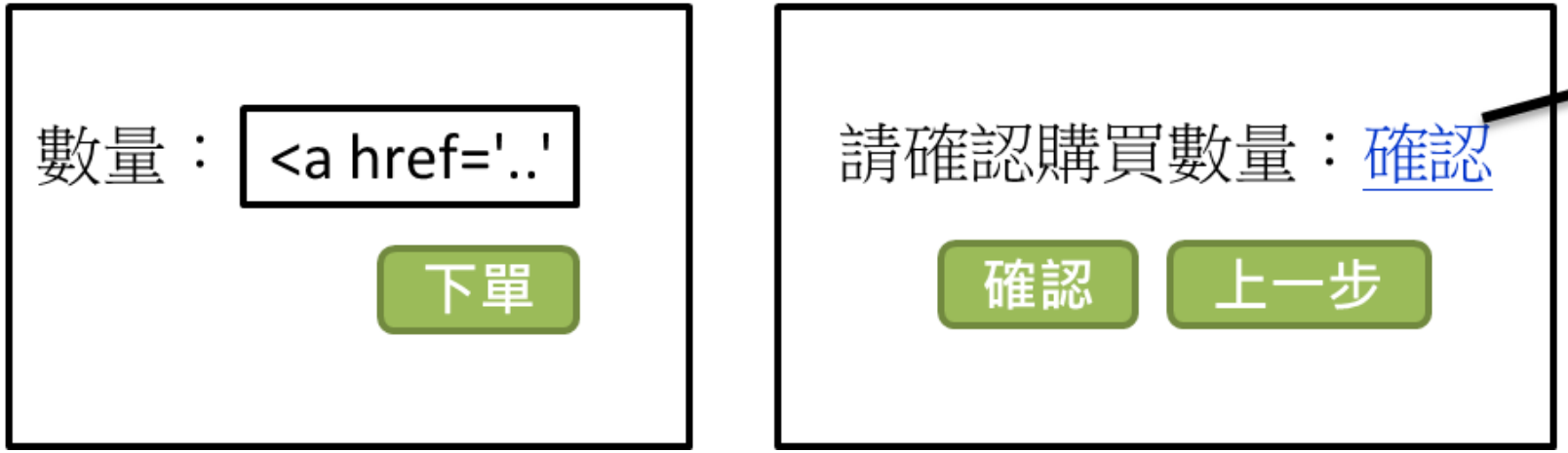
進攻難度：★★★☆☆

防禦難度：★★☆☆☆

正常流程



攻擊流程
(Dom XSS)



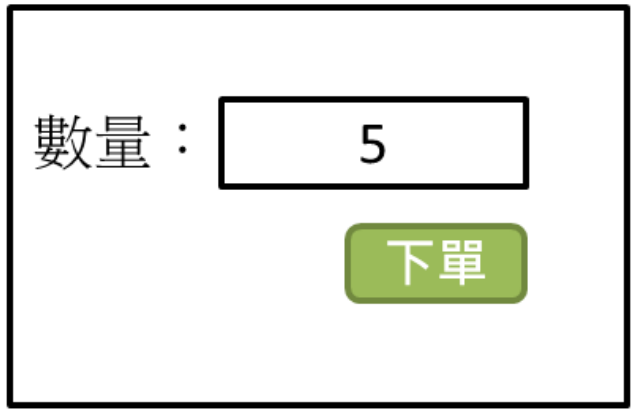
CROS

DEMO

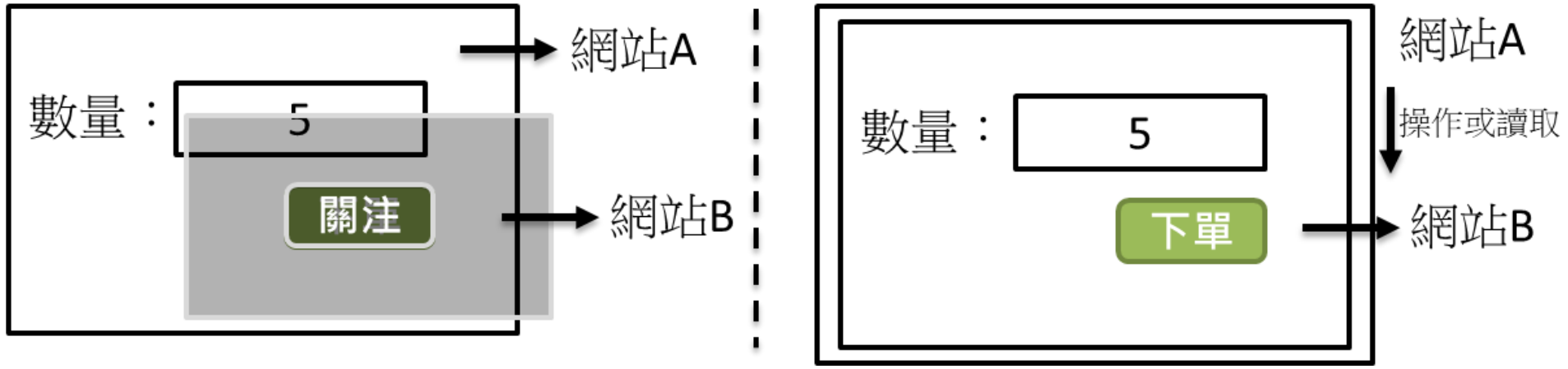
進攻難度：★★★☆☆

防禦難度：★☆☆☆☆

正常流程



攻擊流程



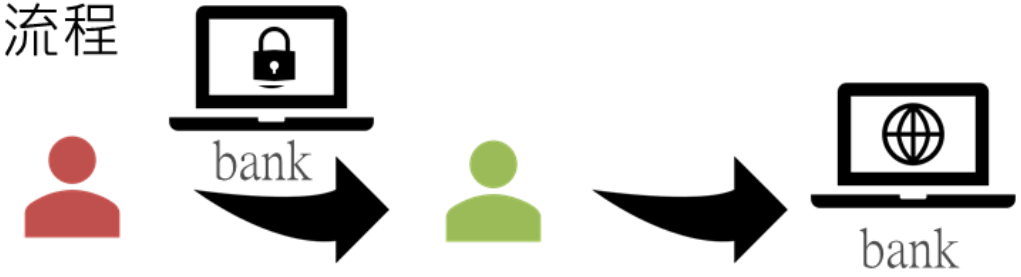
CSRF

DEMO

進攻難度：★★★☆☆

防禦難度：★★★☆☆

正常流程



http://bank/withdraw?account=bob&amount=1000000&for=bob2

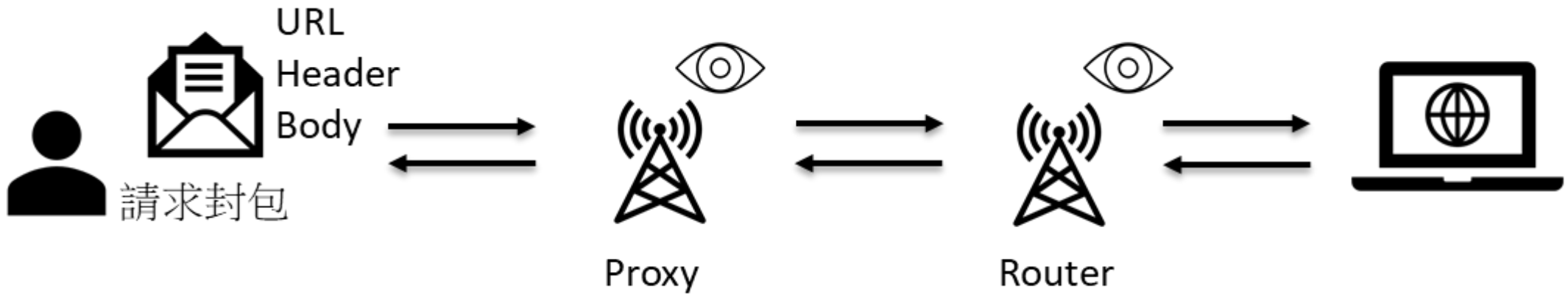
攻擊流程

http://bank/withdraw?account=bob&amount=1000000&for=double

封包攔截 / 修改

進攻難度：★★★☆☆

防禦難度：★★☆☆☆



http://sample/login?userid=double&password=135246

階段結論

「

你願意花多少成本，
守護你的資訊安全？

」

至少，不要被隨手可得. . .

Thank You !

加密基礎概念

Agenda-密碼學

- 加密原理
- 對稱/不對稱加密
- 區塊/串流加密
- 公鑰與私鑰
- 數位簽章與憑證

Caesar Cipher (凱撒密碼)

- 所有字母都在字母表上向後3位
- 根據偏移量的不同，還存在若干特定的凱撒密碼變形

例如:

```
meet me after the toga party
PHHW PH DIWHU WKH WRJD SDUWB
```

Monoalphabetic Cipher(單字母替換加密)

- 每個明文字母替換至隨機的密文表

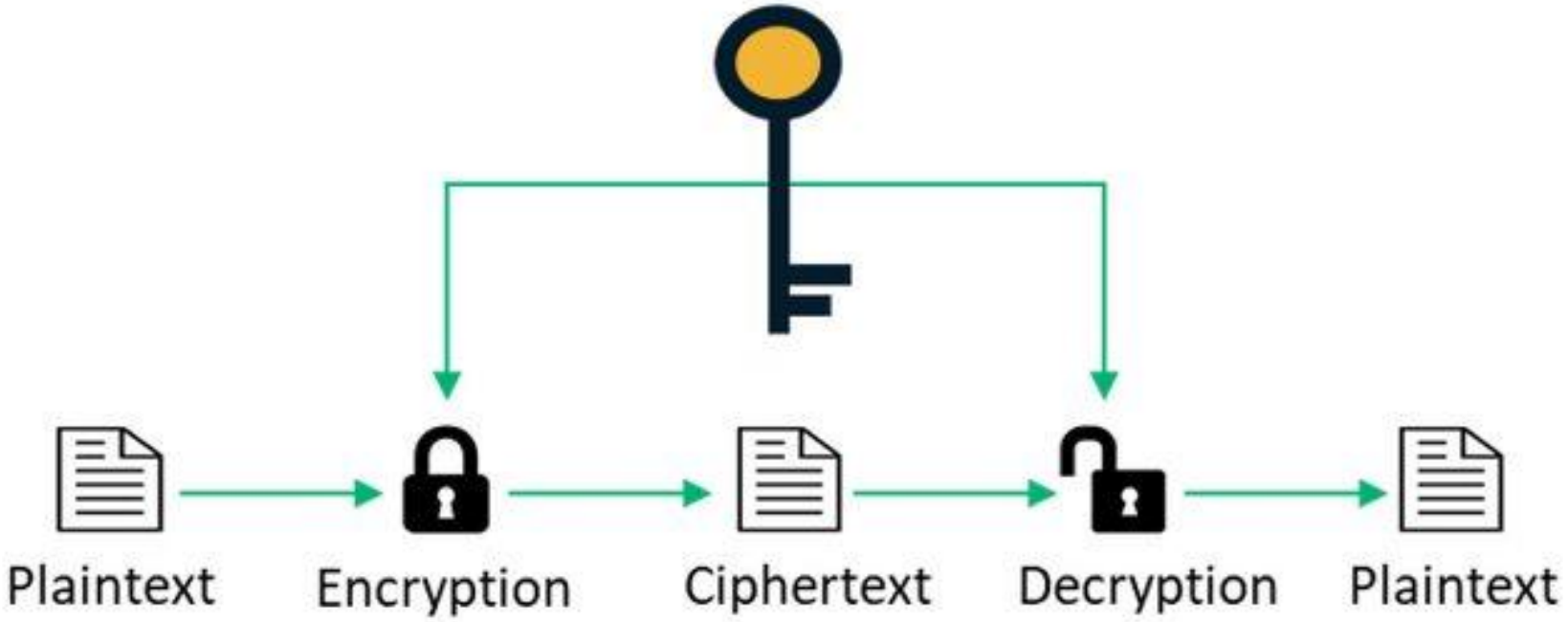
例如:

```
Plain:  abcdefghijklmnopqrstuvwxyz
Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN
Plaintext:  ifwewishtoreplaceletters
Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA
```

區塊/串流加密

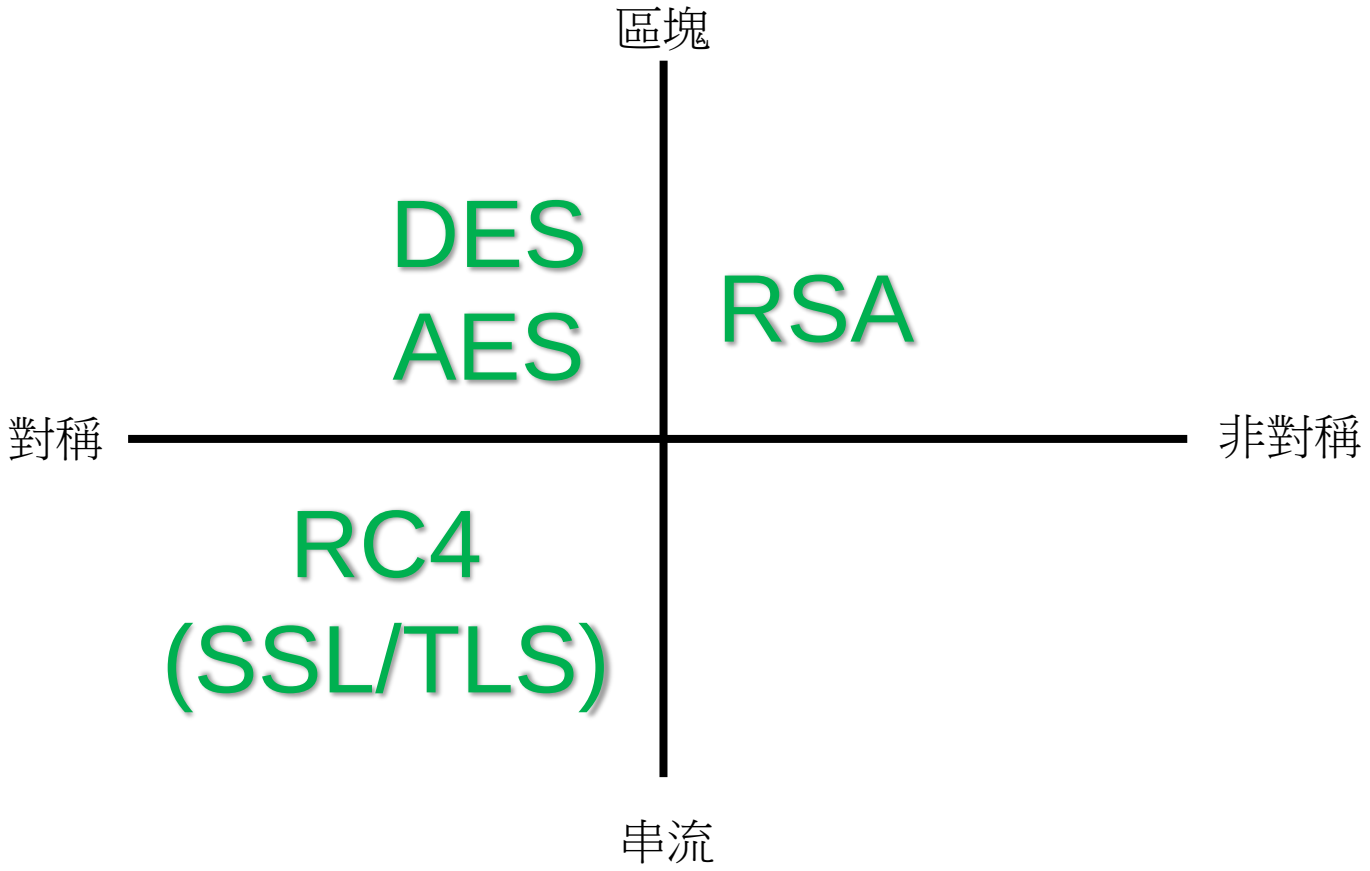
- 區塊是每次對一個區塊進行加解密
  - 每個區塊至少64-bits
- 串流加密是每次對一個 bit 或一個 byte進行加解密
- 當前大多主流是block ciphers

Symmetric Cipher Model(對稱加密)



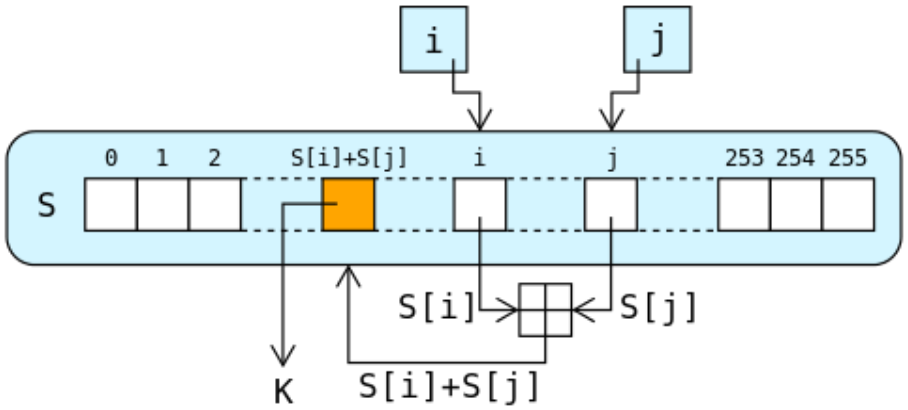
Asymmetric Cipher Model(不對稱加密)

加密方式分類

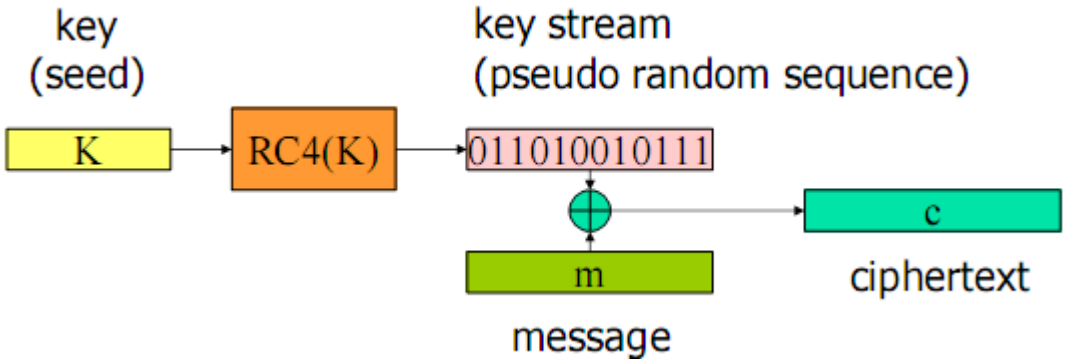


Rivest Cipher 4（RC4）

- 它使用相同的密鑰，屬於對稱加密算法
- 密鑰長度可變，範圍是[1,255]，以byte單位做串流加密
- 常用於 (web SSL/TLS, wireless WEP)協議和標準



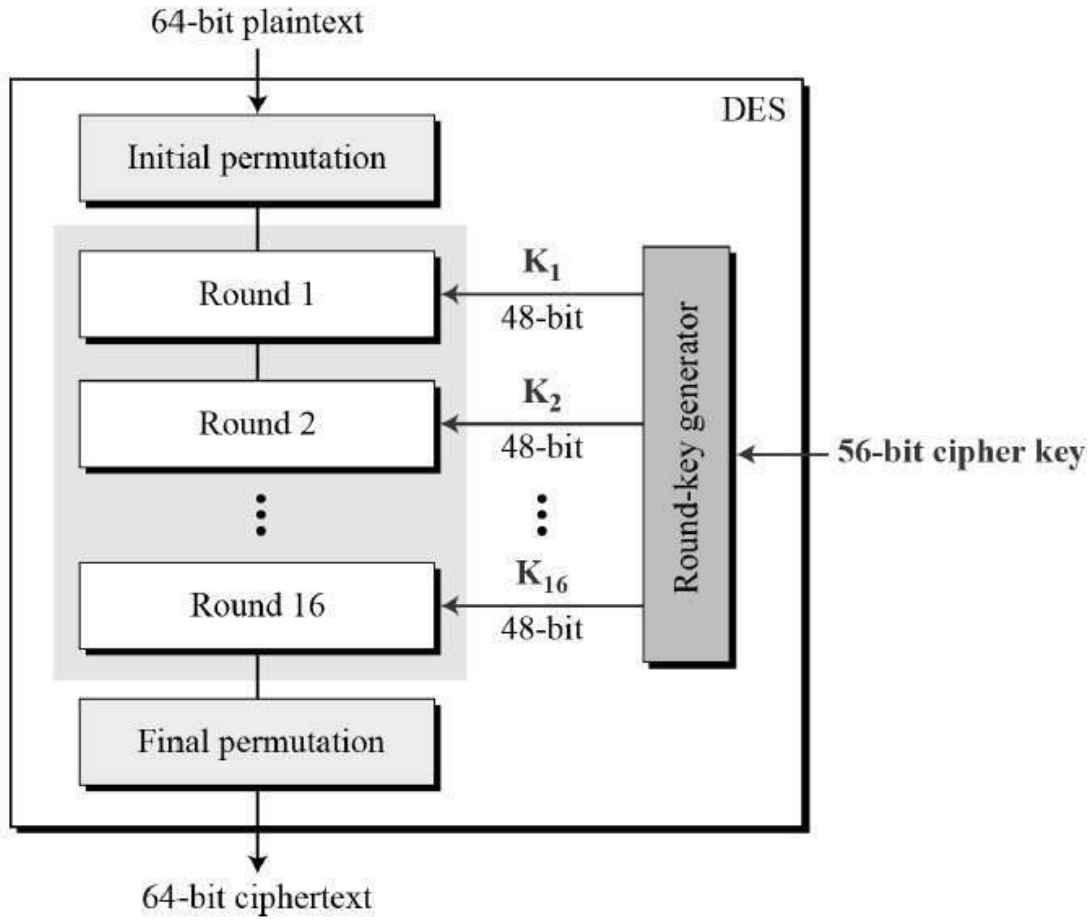
不斷生成擴充金鑰



透過變動金鑰進行加密

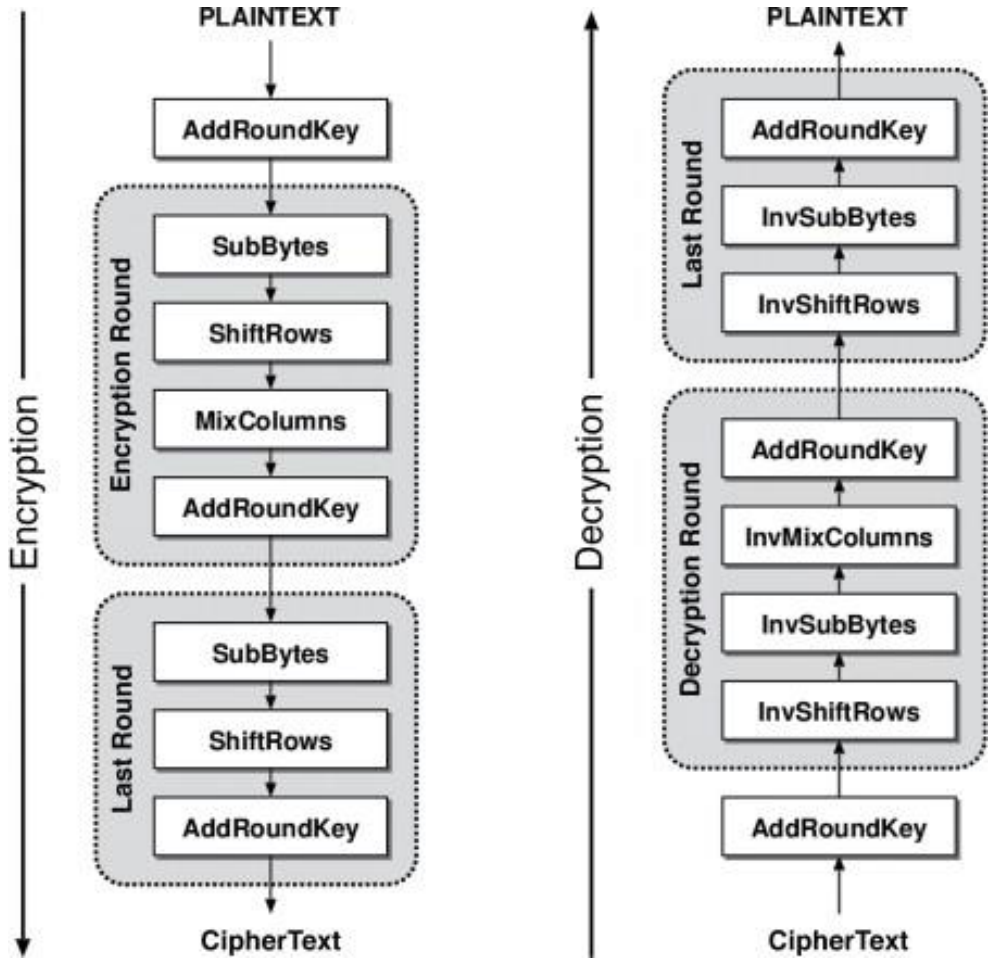
Data Encryption Standard (DES)

- 是一種”對稱”鑰加密塊密碼演算法，1976年被美國聯邦政府的國家標準局(NIST)確定為聯邦資料處理標準（FIPS），隨後在國際上廣泛流傳開來。它基於使用56位金鑰的對稱演算法。



Advanced Encryption Standard(AES)

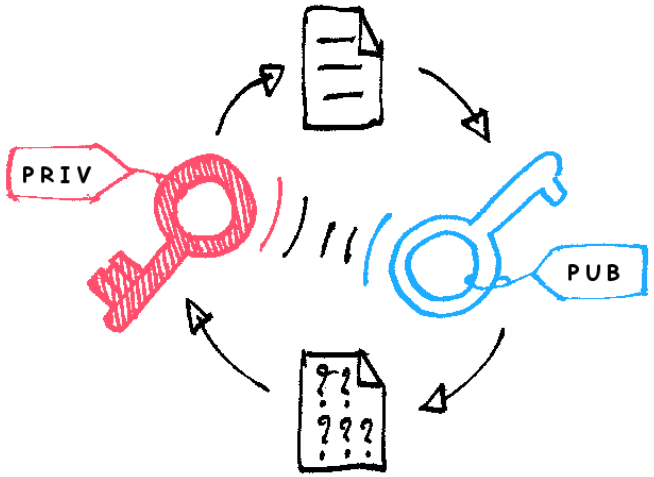
- 又稱Rijndael加密法（音似英文的「Rhine doll」），是美國聯邦政府採用的一種區塊加密標準。這個標準用來替代原先的DES，已經被多方分析且廣為全世界所使用。經過五年的甄選流程，進階加密標準由美國國家標準與技術研究院（NIST）於2001年11月26日發佈於FIPS PUB 197，並在2002年5月26日成為有效的標準。現在，進階加密標準已然成為對稱金鑰加密中最流行的演算法之一。



RSA

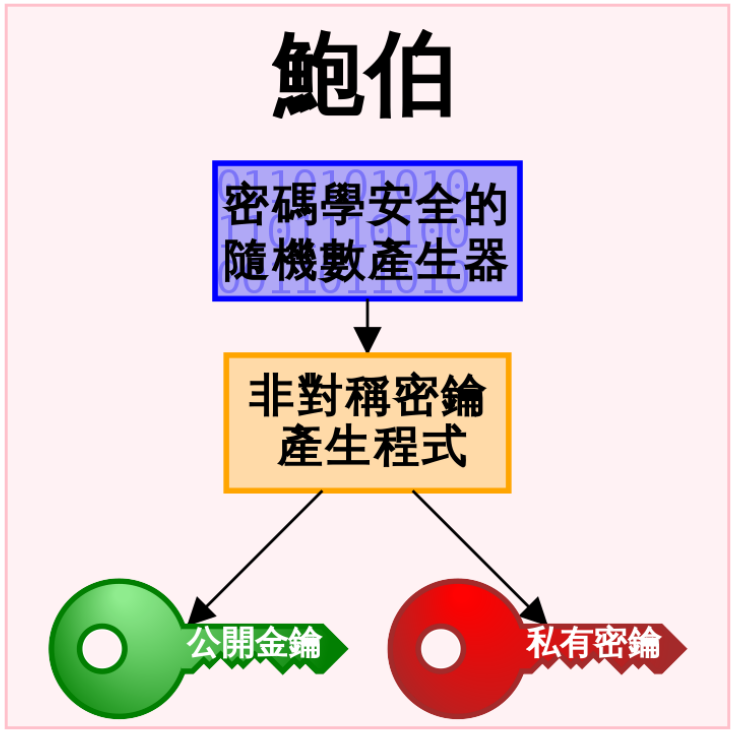
- RSA加密演算法是一種非對稱加密演算法，在公開金鑰加密和電子商業中被廣泛使用。
- RSA是由羅納德·李維斯特（Ron Rivest）、阿迪·薩莫爾（Adi Shamir）和倫納德·阿德曼（Leonard Adleman）在1977年一起提出的。當時他們三人都在麻省理工學院工作。RSA 就是他們三人姓氏開頭字母拼在一起組成的。
- RSA加密演算法是一種特殊的非對稱密碼法，利用兩個質數作為加密與解密的兩個鑰匙(key)。這兩個鑰匙分別稱為公開鑰匙 (public key) 和私人鑰匙 (private key 或是 secret key)，鑰匙的長度約在 40 個位元到 1024 位元。

公鑰與私鑰

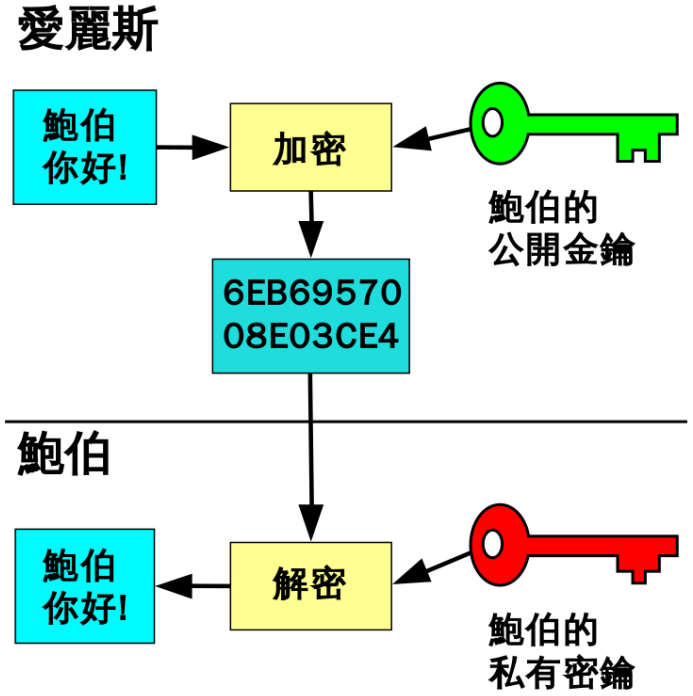


不可逆的過程

發送端



接收端



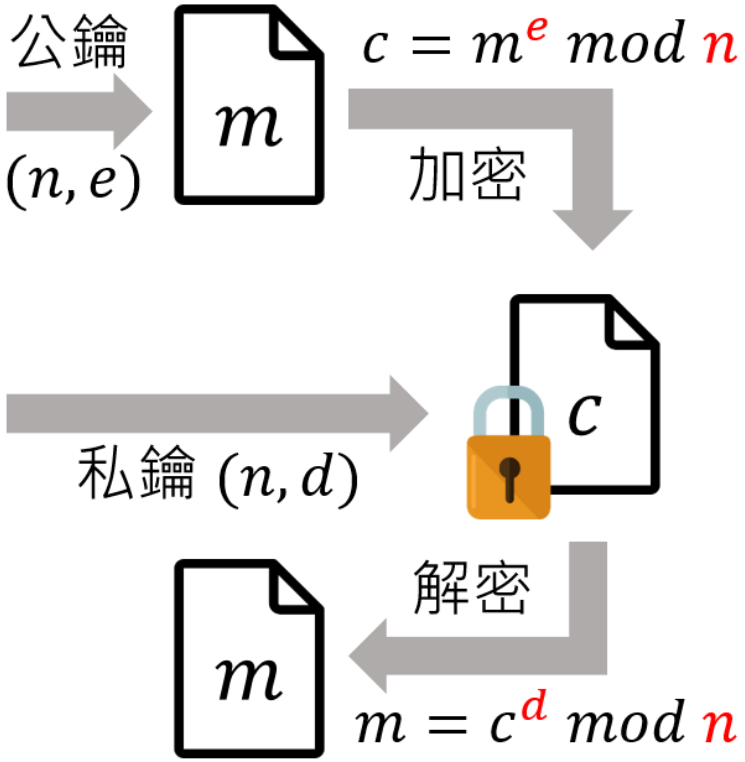
私鑰可以產出公鑰、公鑰無法產出私鑰，因此產生的兩把鑰匙中，私鑰會放在身上、公鑰才會在外流通或傳遞

RSA-金鑰產生流程

$n = p \times q$
$t = (p-1)(q-1)$
p, q為質數

公鑰 (n, e)
條件：$e \neq 0 \bmod t$

私鑰 (n, d)
條件：$ed = 1 \bmod t$



傳遞的資料
- n (計算出正整數)
- e (公鑰)
- c (密文)

解密需要資料
- n (計算出正整數)
- d (私鑰)
- c (密文)

MD5(Message-Digest Algorithm)

- MD5是輸入不定長度資訊，輸出固定長度128-bits的演算法。經過程式流程，生成四個32位元資料，最後聯合起來成為一個128-bits雜湊。基本方式為，求餘、取餘、調整長度、與連結變數進行迴圈運算。得出結果。

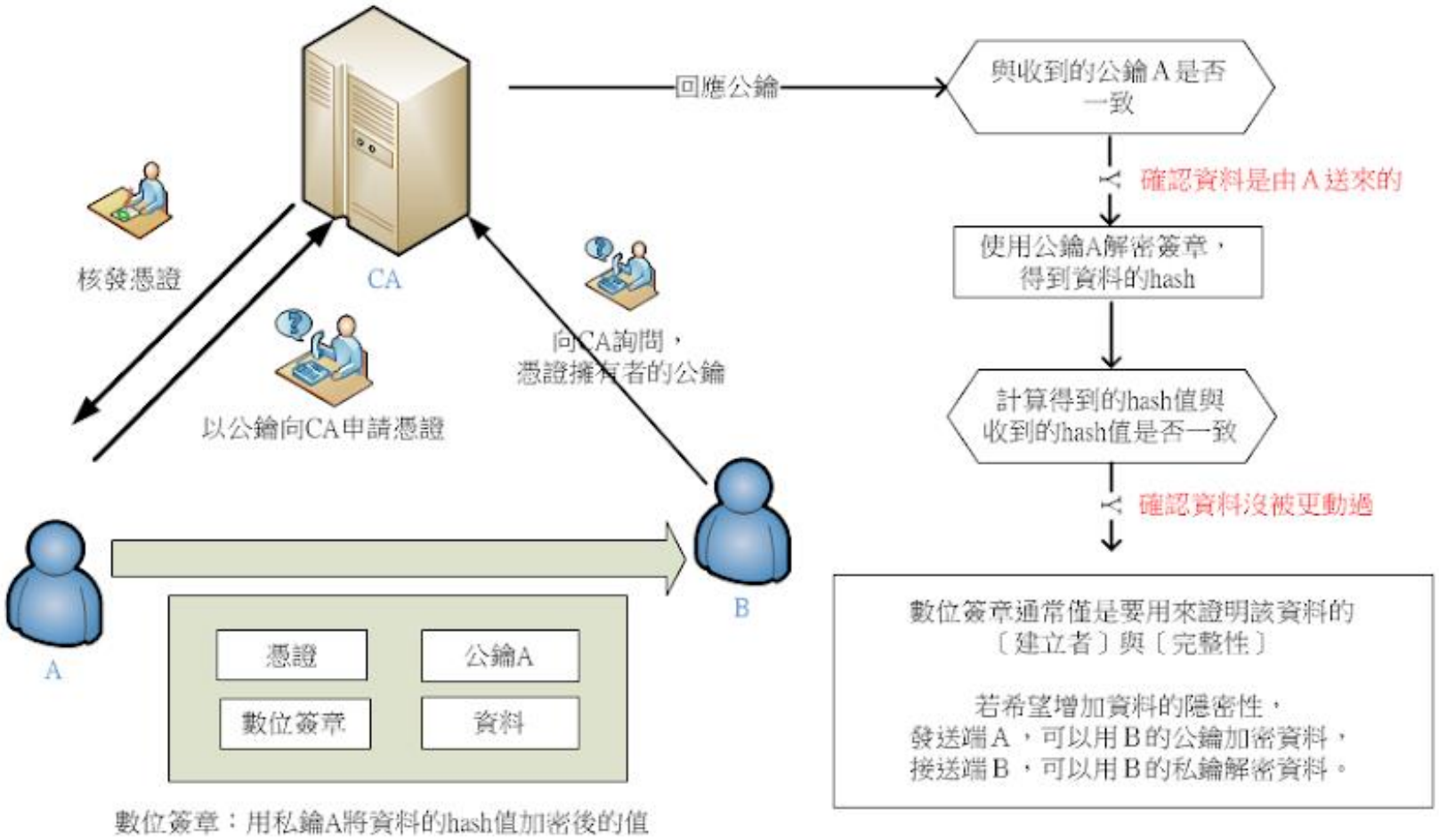
```
MD5("The quick brown fox jumps over the lazy dog")
= 9e107d9d372bb6826bd81d3542a419d6
```

128-bits

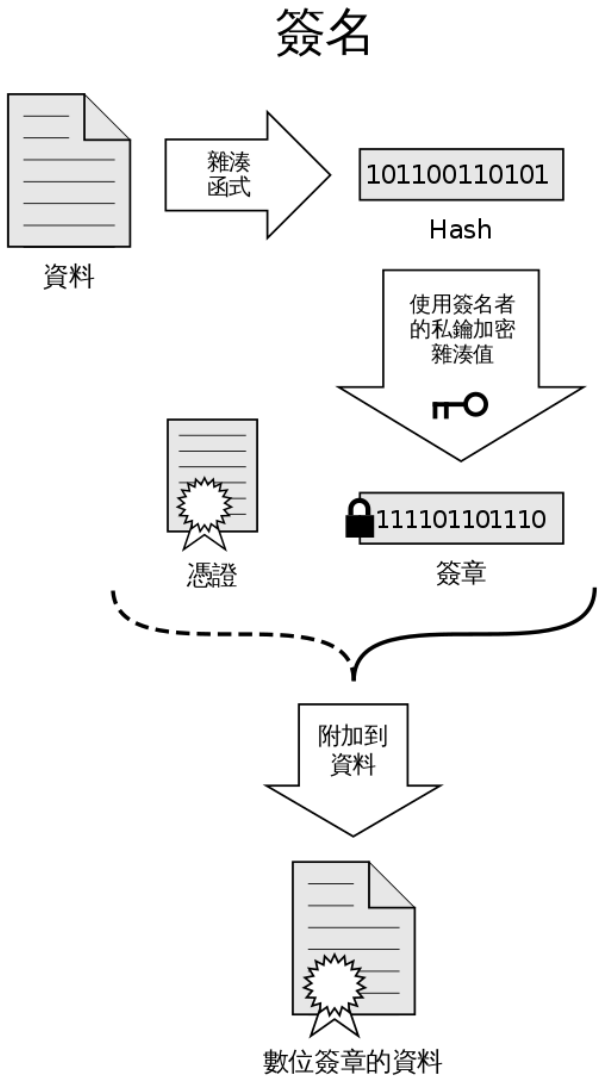
SHA(Secure Hash Algorithm)

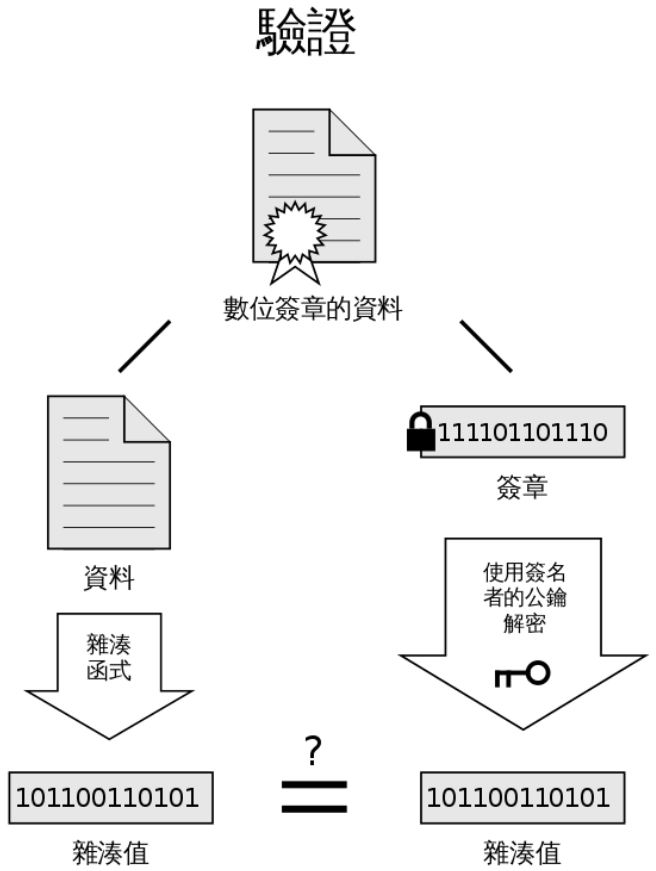
演算法和變體		輸出雜湊值長度（bits）	中繼雜湊值長度（bits）	資料區塊長度（bits）	最大輸入訊息長度（bits）	迴圈次數	使用到的運算子
MD5（作為參考）		128	128 (4 × 32)	512	無限[4]	64	And, Xor, Rot, Add (mod 2^{32}), Or
SHA-0		160	160 (5 × 32)	512	$2^{64} - 1$	80	And, Xor, Rot, Add (mod 2^{32}), Or
SHA-1		160	160 (5 × 32)	512	$2^{64} - 1$	80	
SHA-2	*SHA-224* *SHA-256*	224 256	256 (8 × 32)	512	$2^{64} - 1$	64	And, Xor, Rot, Add (mod 2^{32}), Or, Shr
	SHA-384 *SHA-512* *SHA-512/224* *SHA-512/256*	384 512 224 256	512 (8 × 64)	1024	$2^{128} - 1$	80	And, Xor, Rot, Add (mod 2^{64}), Or, Shr
SHA-3	*SHA3-224* *SHA3-256* *SHA3-384* *SHA3-512*	224 256 384 512	1600 (5 × 5 × 64)	1152 1088 832 576	無限[7]	24[8]	And, Xor, Rot, Not
	SHAKE128 *SHAKE256*	*d* (arbitrary) *d* (arbitrary)		1344 1088			

數位簽章與憑證


回應公鑰
與收到的公鑰A是否一致
Y
確認資料是由A送來的
使用公鑰A解密簽章，得到資料的hash
計算得到的hash值與收到的hash值是否一致
Y
確認資料沒被更動過
數位簽章通常僅是要用來證明該資料的〔建立者〕與〔完整性〕
若希望增加資料的隱密性，
發送端A，可以用B的公鑰加密資料，
接送端B，可以用B的私鑰解密資料。
核發憑證
CA
向CA詢問，憑證擁有者的公鑰
以公鑰向CA申請憑證
A
B
憑證
公鑰A
數位簽章
資料
數位簽章：用私鑰A將資料的hash值加密後的值

數位簽章與憑證





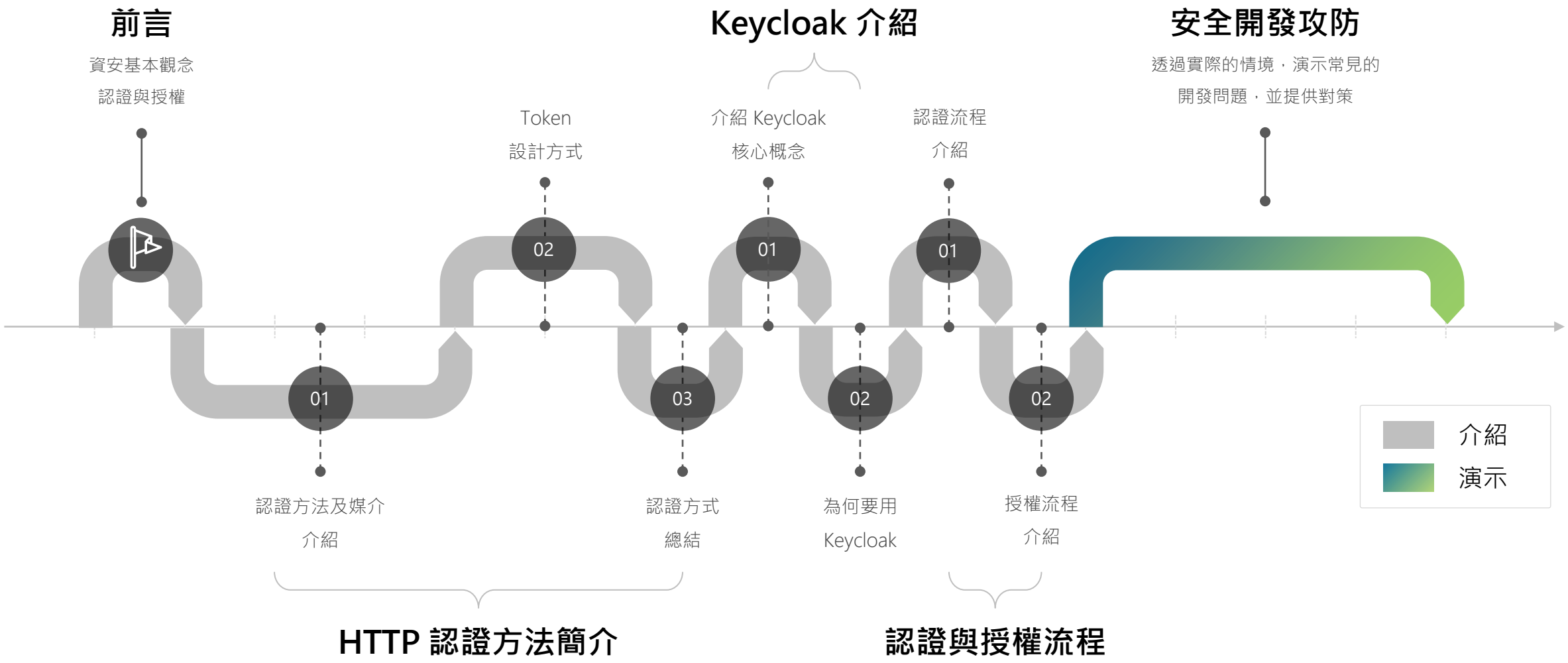
若雜湊值相同，則數位簽章有效。

Thank You !

Steve-30 min

開發程式相關

Course Roadmap



前言

資安基本觀念 - 認證與授權

大多數的資安事件都是因為 DevOps 流程及工具的機密遭到外洩而引起，因此適當的**認證**和**授權**機制非常重要。

分不清楚 Authentication 及 Authorization ?

我是誰？

Authentication

如同通行證的概念，用來證明你的身分。就以公司來說，進到公司的時候需要配戴員工識別證，才可以刷卡開門進公司。

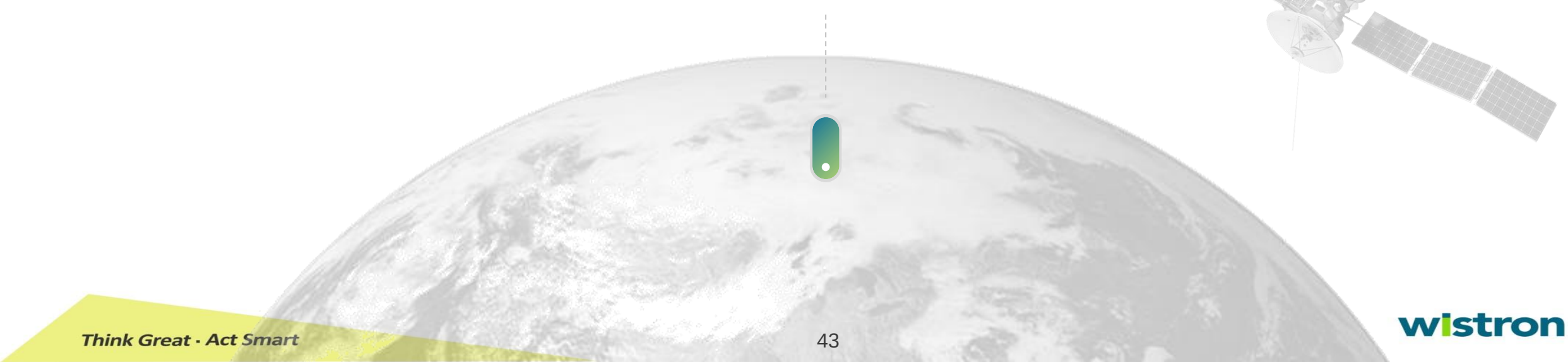
我能做什麼？

Authorization

經過認證後，被賦予了哪些權力。舉例來說，當我們進公司工作時，會根據應徵職位被賦予對應的權力，例如財務長被授予管理公司財務。

HTTP 認證方法簡介

常見認證方式及 Token 設計方式介紹



Basic Authentication

由帳號和密碼透過 Base 64 來組成認證碼

難度 ★☆☆☆☆

安全 ★☆☆☆☆

Session-Based Authentication

在 Cookie 中提供 Session ID 給 Server 驗證

難度 ★★★☆☆

安全 ★★★☆☆

OAuth2.0 Authentication

透過第三方認證，並將認證用的 credential 及執行用的 token 分離

難度 ★★☆☆☆

安全 ★★★☆☆

Token-Based Authentication

Client 在 HTTP Header 提供 Token，供 Server 以密鑰驗證

難度 ★★★☆☆

安全 ★★★☆☆

認證方法

Basic Authentication 與 OAuth2 比較

Basic Authentication

實現方式

取得用戶的帳號及密碼，並將其以 **{帳號}:{密碼}** 的格式合併成 Credential，並進行 Base64 encoded
最後添加至 HTTP Header，提交至對應的服務進行身分驗證。

```
Authorization: Basic YWRtaW46YWRtaW4K
```

優點

- 使用簡單
- 開發容易實現
- 沒有複雜的頁面挑轉或交互過程

缺點

- 安全性低
- 每次調用都需要提供帳號密碼
- 客戶端須保存帳號密碼

OAuth2 Authentication

實現方式

同樣以用戶的帳號密碼來組成 Credential，不同的是交給第三方平台認證身份且將認證身份的 Credential 和實際執行權限的Access Token 分離，並使用 Bearer Token 進行驗證。

```
Authorization: Bearer cn389ncoiwuencr
```

優點

- 安全性高
- 僅需提供一次帳號密碼
- 不會有密碼被竊取並竄改的問題

缺點

- 屬於協議，因此沒有具體的實現方式
- 因實現方式不同，兼容性較差
- Secret Key 遭竊取影響更為嚴重

認證媒介簡介

認證媒介 - Session-Based Authentication

整體流程如右所示。Client 端先將帳號密碼丟給伺服器端， Server 端透過帳密驗證，若確定是正確的用戶，就開始建立一個 Server 端與 Client 端間連線的 Session 資料，裡面會放置用來辨識這個 Session 由哪個用戶建立的資料，接著會回傳該 Session ID 回來，並要求 Client 端之後要驗證時需要帶這個 Session 的 ID 在 HTTP Cookie 上。

缺點

用 Cookie 傳遞資料，Cookie 只要是在同網域下的請求皆會帶上，因此可以透過別的網站連該 API，就可以對該 API 的資源進行存取，這個攻擊稱作 CSRF (Cross Site Request Forgery)

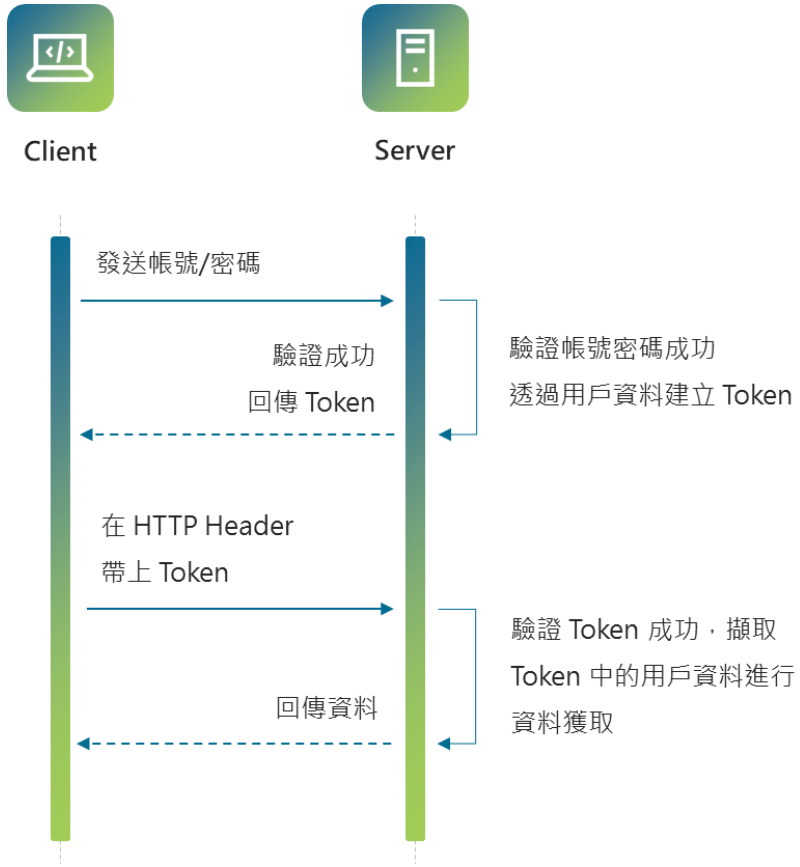
對策

防止方式通常是將 Same Site 設定加註在 Cookie 上，讓瀏覽器僅會在發送的請求是與該網頁同網域的 URI 的時候，才會帶上 Cookie 在請求中

認證媒介簡介

認證媒介 - Token-Based Authentication

整體流程如右所示。和 Session-Based Authentication 類似，但是 Server 端不需要另外儲存任何資料。相關的資料透過密鑰簽署組合成 Token，交由 Client 端保管。Client 端拿到 Token 後，只要在每次 HTTP request 發送的時候，在 header 中的 Authorization 區塊帶上 Token 給 Server，即可讓 Server 透過密鑰驗證 Token 後，進行資料的存取。



缺點

Token 通常必須要由 Client 端自行決定如何儲存。常見的 Client 端通常為瀏覽器，所以通常是使用 JavaScript 去進行儲存。若網頁被別人加上 JavaScript 程式碼，就可以盜取 Token

對策

常見的防止方式是將 Token 一樣使用 Cookie 去帶，並且對該 Cookie 使用 Http Only 和 Secure Flag 讓 Cookie 內容不能被 JavaScript 讀取

Token 設計方式

比較各種 Token 的實現方式

自行設計 Token

適用情境

自家 Server 服務對接，即呼叫的 Client 端，都為自家服務或 Server

優點 可自定義編碼及加密方式

缺點 更新 Token 的話 Client 和 Server 都要手動更新

JWT Token

適用情境

微服務 Server 或 Client 端串接，或單次認證行為使用 (e.g. Email 驗證)

優點 已有現成的套件可以直接使用

缺點 更新 Token 的話 Client 和 Server 都要手動更新

JWT 組成介紹

JSON Web Token

JWT 由三個 JSON Object 組成，分別為 **Header**、**Payload** 及 **Signature** 並且用「.」來做區隔，而這三個部分會各自進行編碼，組成一個JWT 字串。

Header

由 alg 及 typ 兩個欄位組合

Payload

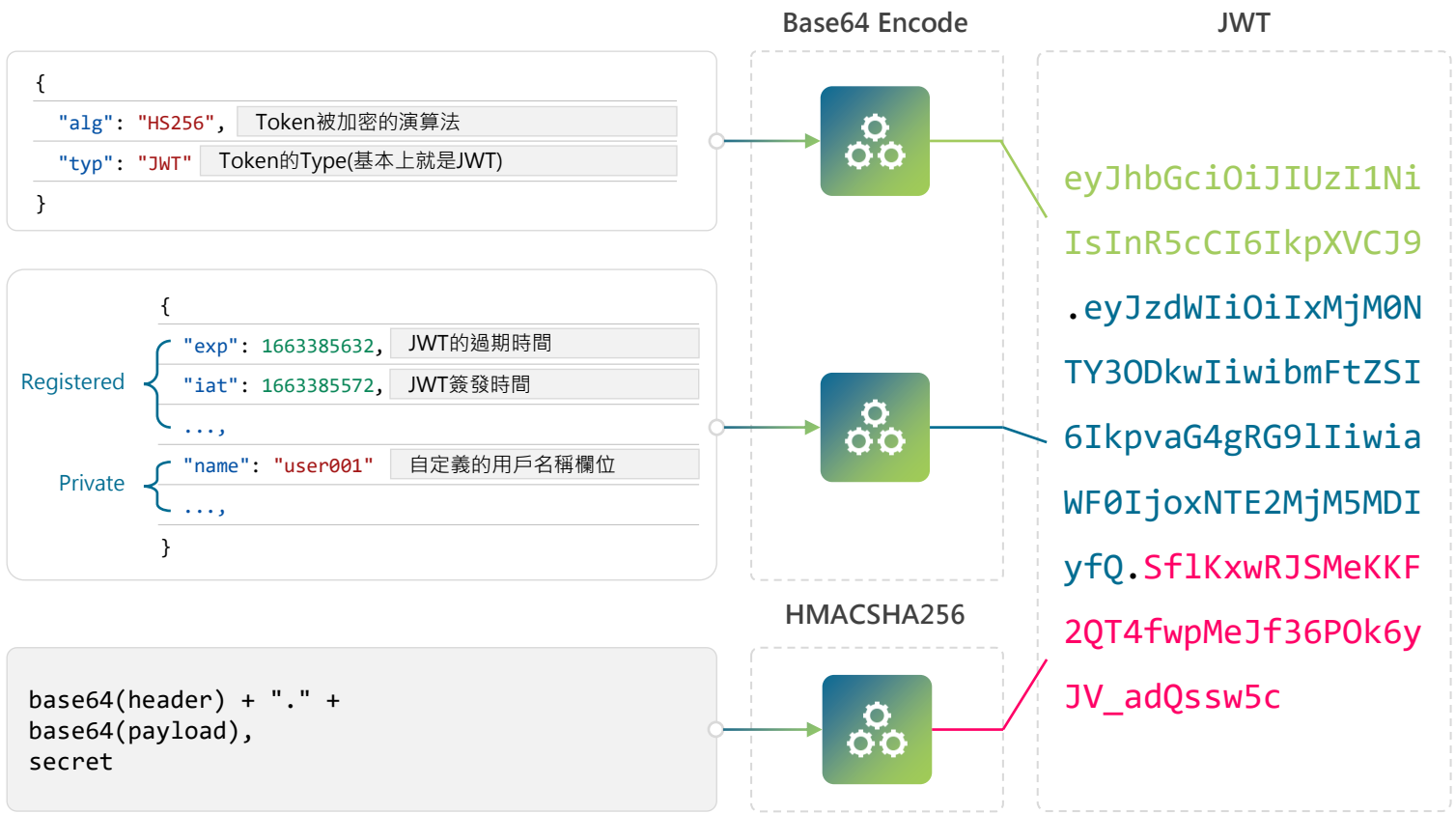
聲明(Claim)內容，也就是用來放傳遞訊息的地方，在定義上有三種聲明：

1. Registered claims:標準公認的建議訊息(不強迫添加)
2. Public claims:在 IANA JWT Registry 上註冊的名稱且不與 Registered claims 衝突
3. Private claims:可自定義的欄位，實務上會放用戶數據等**不敏感**的數據

Signature

由三部分組成:

base64(header) + base64(payload) + secret



認證方式總結

「完美」的安全難以辦到，但可以作到「夠好」的程度

Issue 1

Token 該存放在 Client 的哪裡？ LocalStorage V.S. Cookie

LocalStorage 可以被 Javascript 訪問，所以容易受到 XSS 攻擊，若能有效避免 XSS 也不是不行使用 LocalStorage。

Issue 1.1

我想使用 LocalStorage 保存 Token，但 XSS 該如何防範？

XSS 可分為 Reflected Attack 以及 Persistent Attack：

- Reflected Attack 主要針對 URL 植入惡意代碼，因此主要須提防 <a href>中的字串。
- Persistent Attack 主要將惡意代碼存入資料庫，再透過 UI 暄染 HTML 時產生。

上述問題只要不要直接操作 DOM，在 Angular 已經可以有效解決

詳細可參考 Angular Security

Reflected Attack 在 Angular 下的結果

```
<a _ngcontent-c0=""
  href="unsafe:javascript:alert('xss attack')">
  www.google.com
</a>
```

Persistent Attack 在 Angular 下的結果

Angular 在使用 [innerHTML] 時會避免產生 script 標籤

Issue 1.2

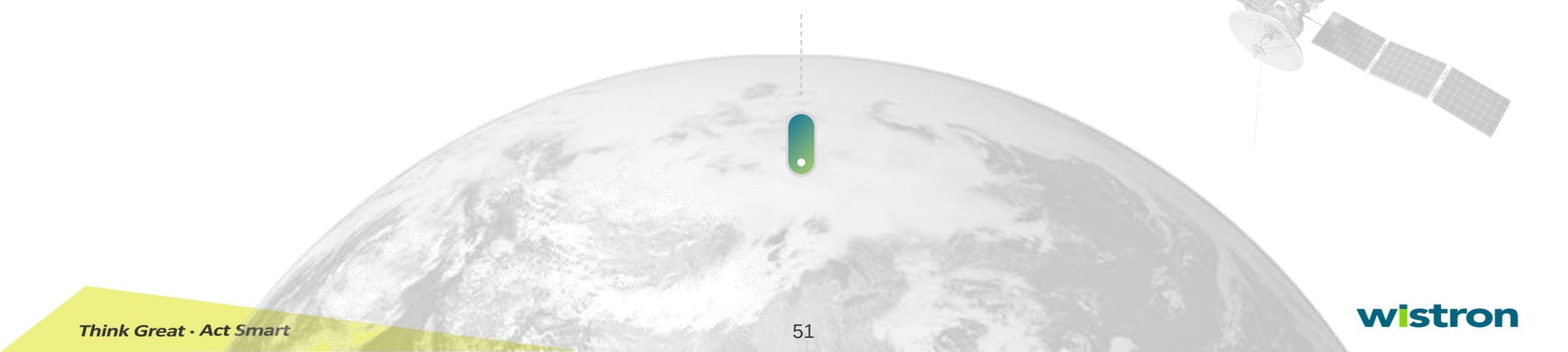
我還是不放心使用 LocalStorage，使用 Cookie 我應該避免什麼問題？

一樣需要避免 XSS 攻擊，同時也須提防 CSRF 攻擊

- XSS 攻擊: Cookie 可透過設定 httponly，來防止 Javascript 讀取，並搭配 Secure 設定，讓 Cookie 只能在 https 下才能傳輸。
- CSRF 攻擊: 通過 Same Site 設定加註在 Cookie 上，僅在發送的請求與該網頁同網域時，才上 Cookie，或搭配 CSRF Token 識別；更嚴謹的做法可以使用圖型驗證、隨機驗證碼等機制避免。

Keycloak 介紹

Keycloak 核心概念及導入前後分析



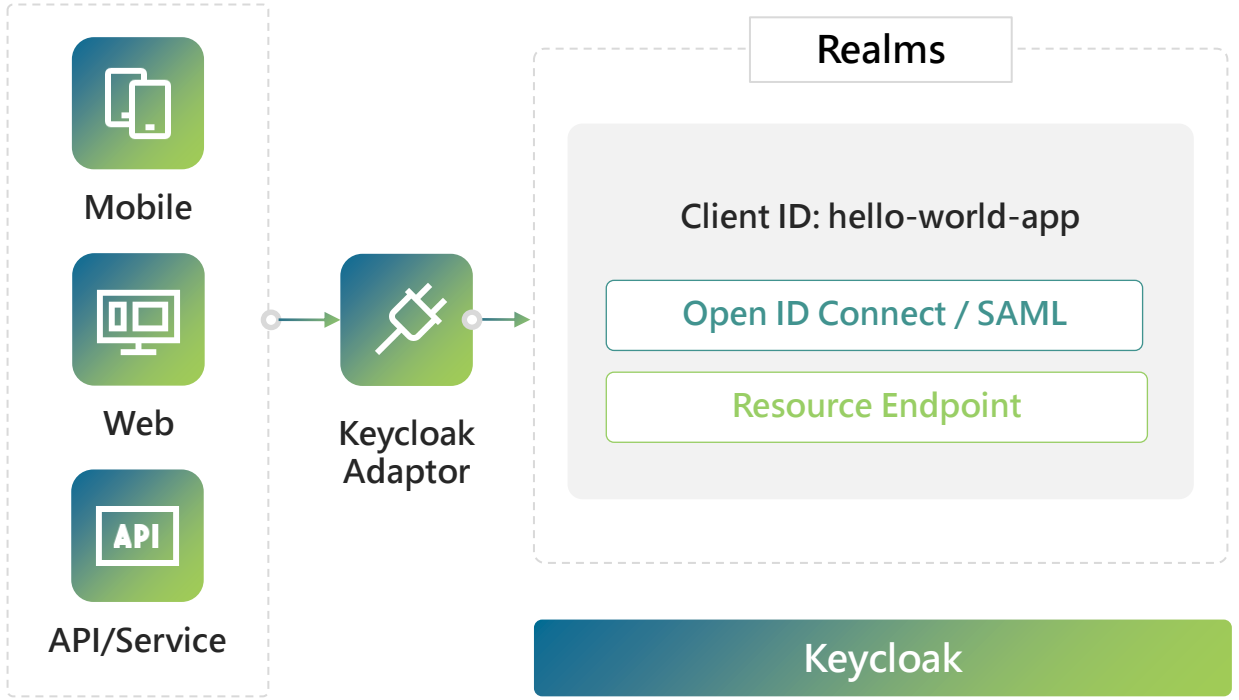
Keycloak 介紹

Keycloak 是一種開源 IAM（身份識別與訪問管理）的解決方案

Keycloak提供了單點登錄（SSO）功能，支援OpenID Connect、OAuth 2.0、SAML 2.0標準協議，擁有簡單易用的管理控制台，並提供對LDAP、Active Directory以及支援Github、Google等社交帳號登入的方式。

4個最常用的核心概念：

User	用戶，使用並需要登錄系統的對象
Roles	角色，用來對用戶的許可權進行管理
Clients	客戶端，需要接入並被 Keycloak 保護的應用和服務
Realms	領域，管理一批用戶、證書、角色、組等，域之間是互相獨立隔離的， 一個域只能管理它下面所屬的用戶



為何使用 Keycloak ?

使用 Keycloak 能在實務上解決那些痛點

省去開發登入驗證

用戶認證交給 Keycloak 實現，且 Keycloak 本身自帶 SSO 功能，開發團隊不用再重造輪子。

減少管理風險

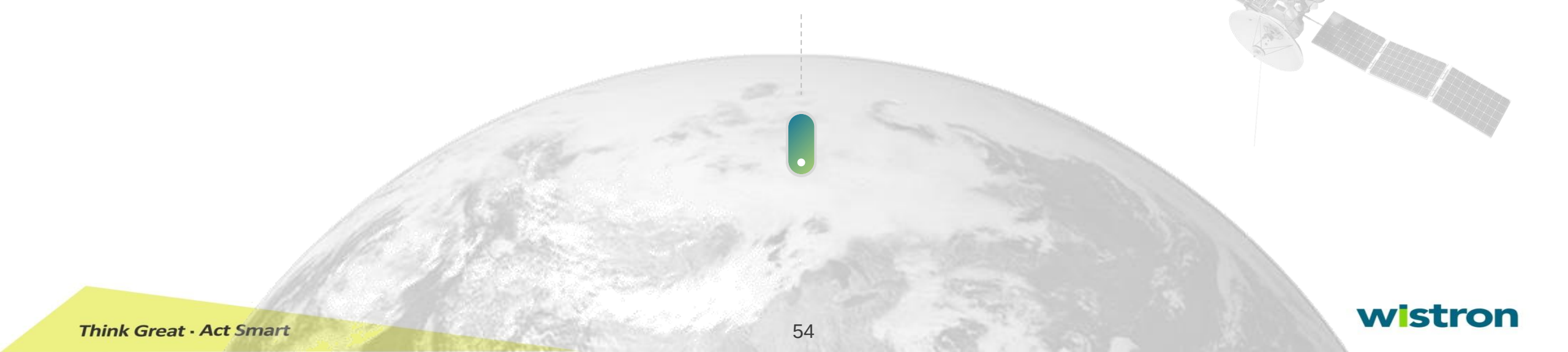
將用戶登入資訊管理這個燙手山芋交給 Keycloak，減少自身系統所需承擔的風險。

提升信任度

因系統不經手用戶登入資訊，且在認證後是透過 Token 進行認證，減少登入資訊被側錄的風險

認證與授權流程

Loopback 與 Keycloak 搭配應用

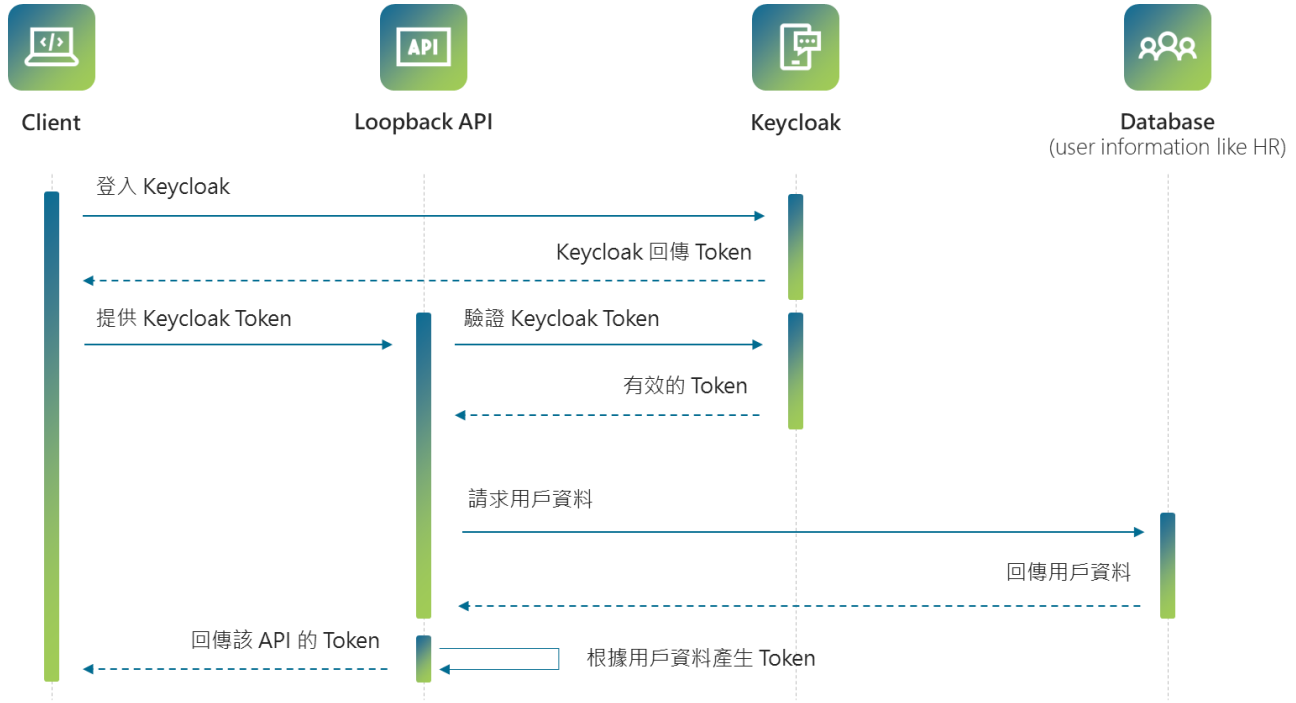


登入流程

Loopback & Keycloak

Keycloak 作為Oauth 認證伺服器，主要提供用戶的帳號認證，並於認證後給予Token。

Loopback 作為系統主要API ，為了不讓API 經手用戶登入資訊(帳號/密碼)，需取得Keycloak登入Token後，再次向Keycloak驗證Token的有效性。驗證確認有效後，回到各自系統或HR DB 取得用戶資訊，並以此為基礎**重新產生Token【該步驟主要為了彌補Keycloak 用戶資訊不完整以及角色授權不容易實現的問題】**

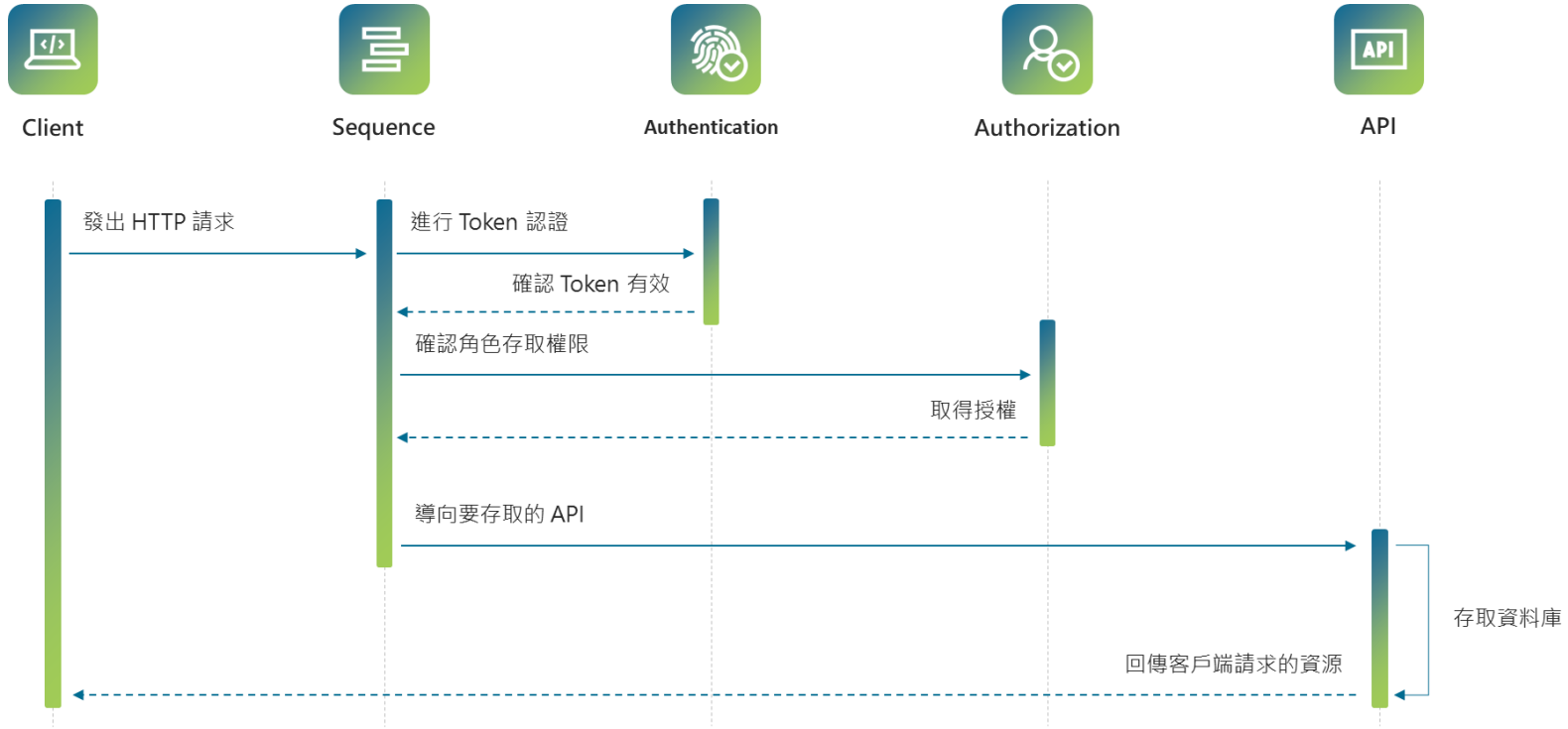


授權流程

客戶端再請求資源時，需先經過 Sequence 驗證 Token 確保登入，並在擷取 Token 中的資訊進行授權資格的確認。

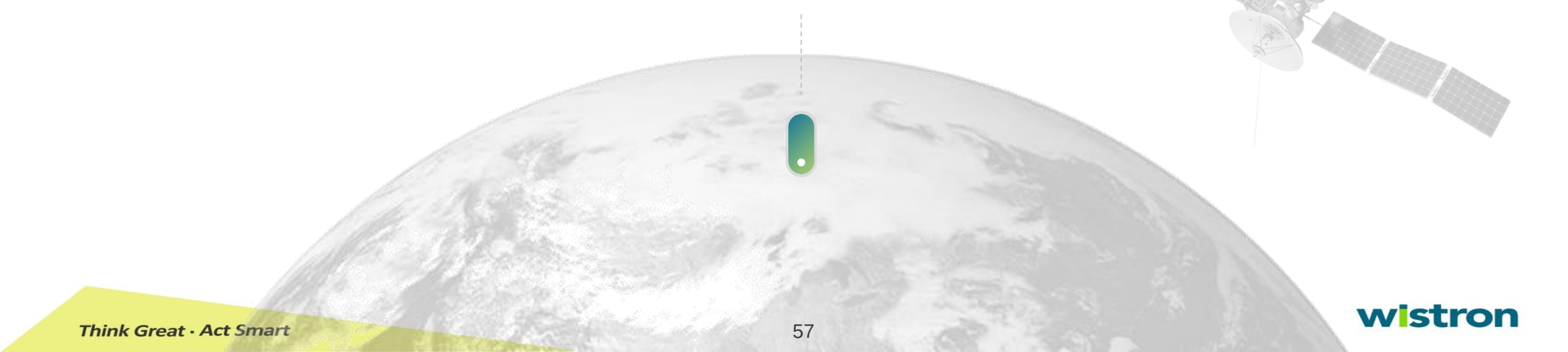
4個常用的授權方式：

Admin	管理員，具備最高權限
Owner	資料擁有者，可存取自身的資料
Team	資料群組，群組內成員僅能檢視
Leader	群組管理員，可存取群組資料



安全開發攻防

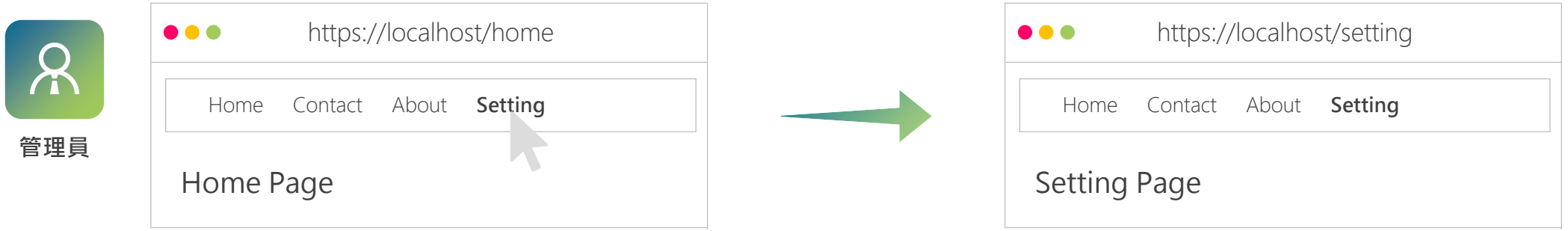
演示常見的開發問題，並提供對策



路由驗證

未實現路由驗證，進而訪問特殊權限頁面

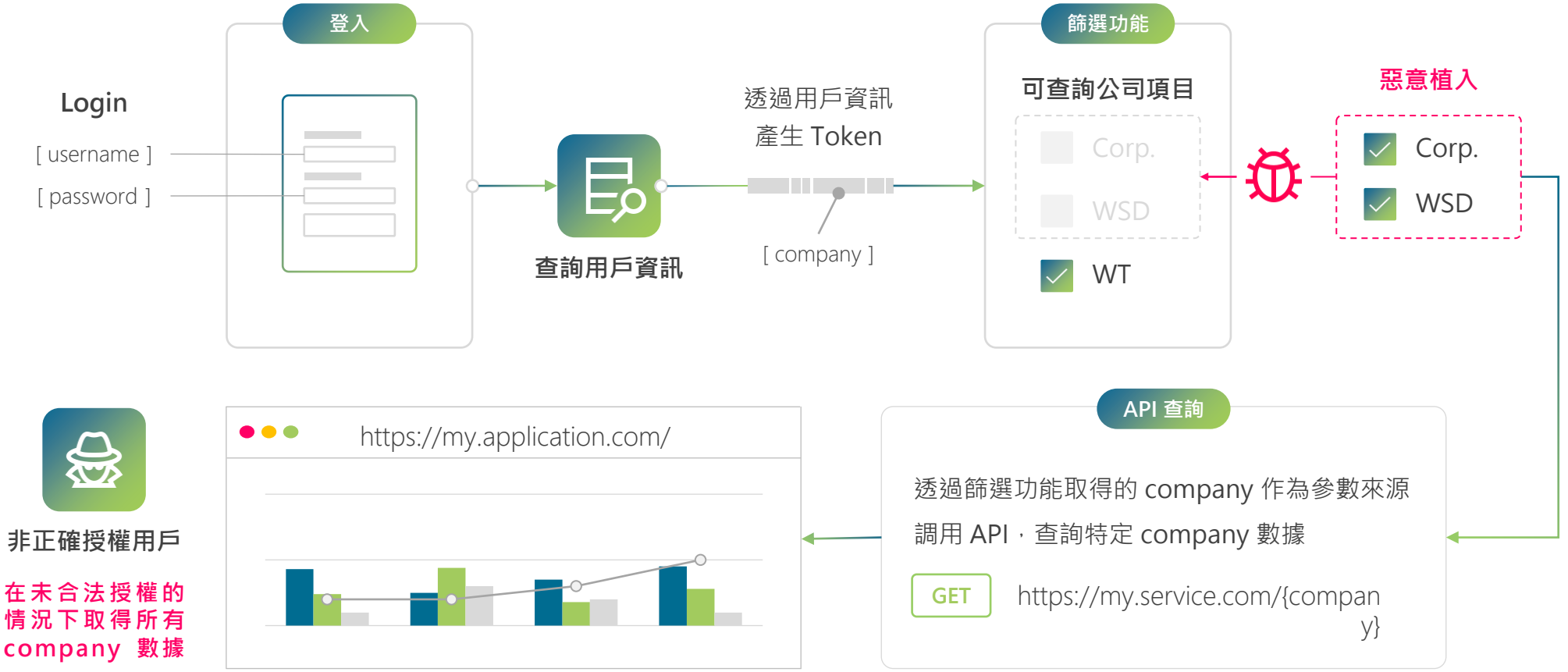
管理員帳號登入，可看到導覽列有 Setting 的項目，點擊後，可進入管理員設定頁面



一般用戶帳號登入，無法看到導覽列的 Setting 項目，但因為沒有設定路由守衛驗證，因此可以透過輸入 /setting 進到管理員設定頁面

權限確認

API 未設定存取權限，導致任何有登入的人員皆可取得所有資源



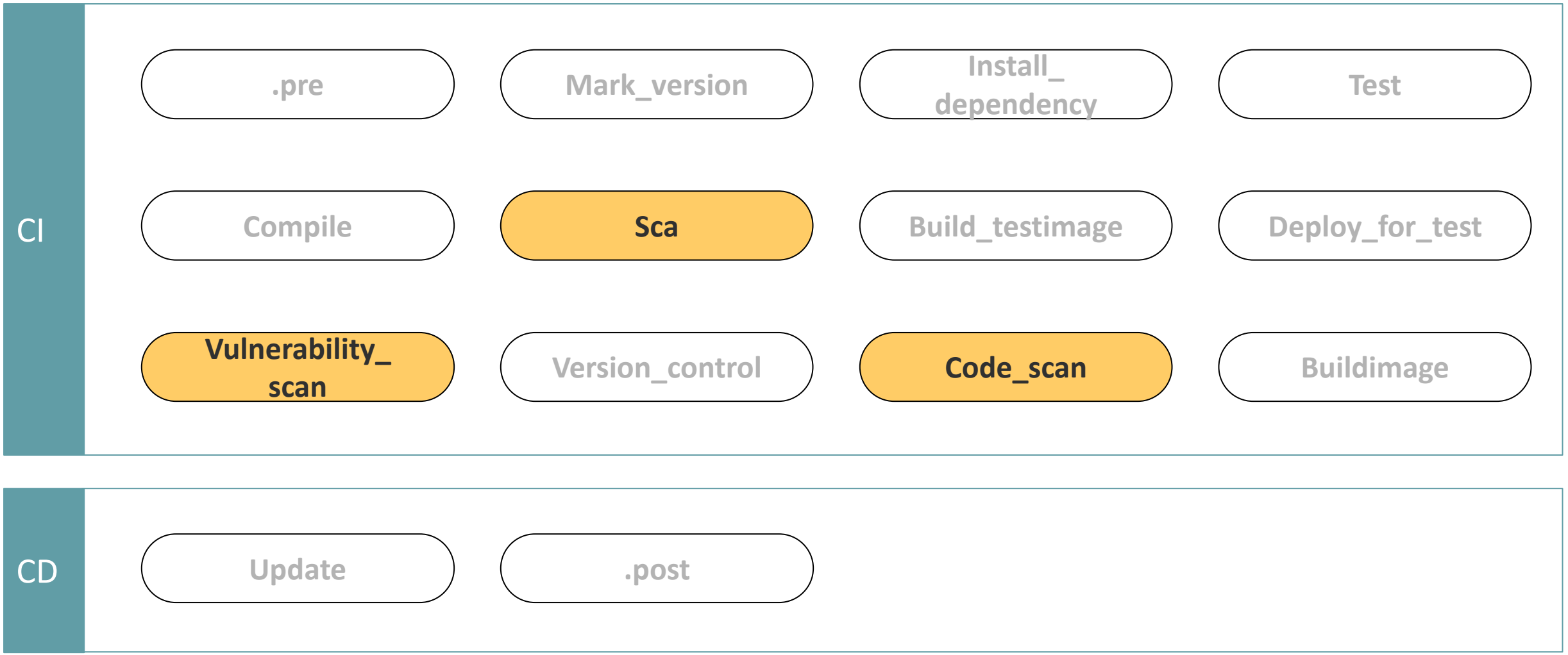
Thank You !

Hao-30 min

CI相關

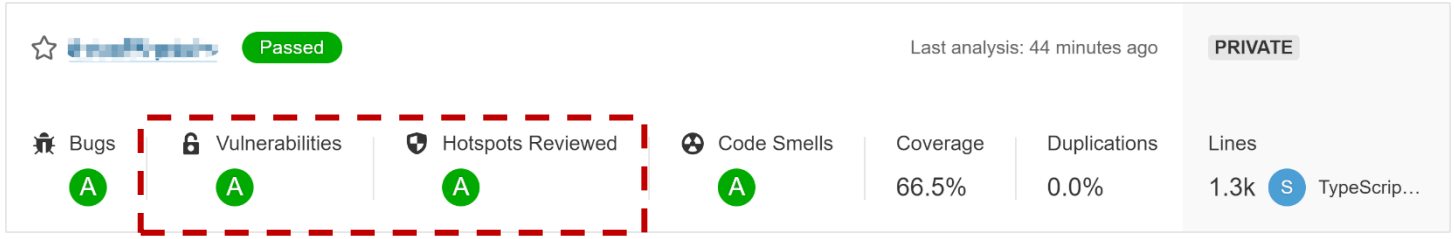
Contents

CICD pipeline flow



SonarQube – 程式品質分析工具

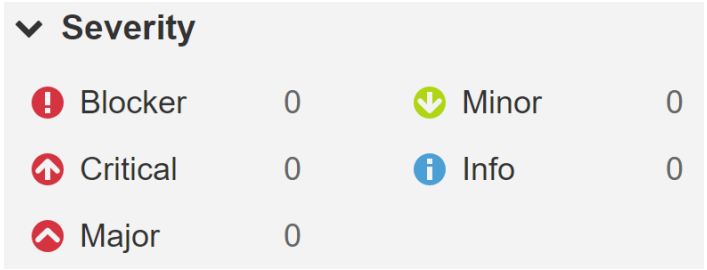
安全性相關指標 :



漏洞(Vulnerabilities) :
使程式碼中出現一個容易受到攻擊的點。

指標等級

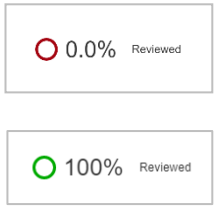
- E ≥ 1 Blocker
- D ≥ 1 Critical
- C ≥ 1 Major
- B ≥ 1 Minor
- A No vulnerabilities



安全疑慮熱點 (Security hotspots) :
需要手動檢查的安全敏感程式碼片段。通過查看，您會發現沒有威脅，或者存在需要修復的易受攻擊的程式碼。

指標等級

- E < 30%
- D ≥ 30% and < 50%
- C ≥ 50% and < 70%
- B ≥ 70% and < 80%
- A ≥ 80%



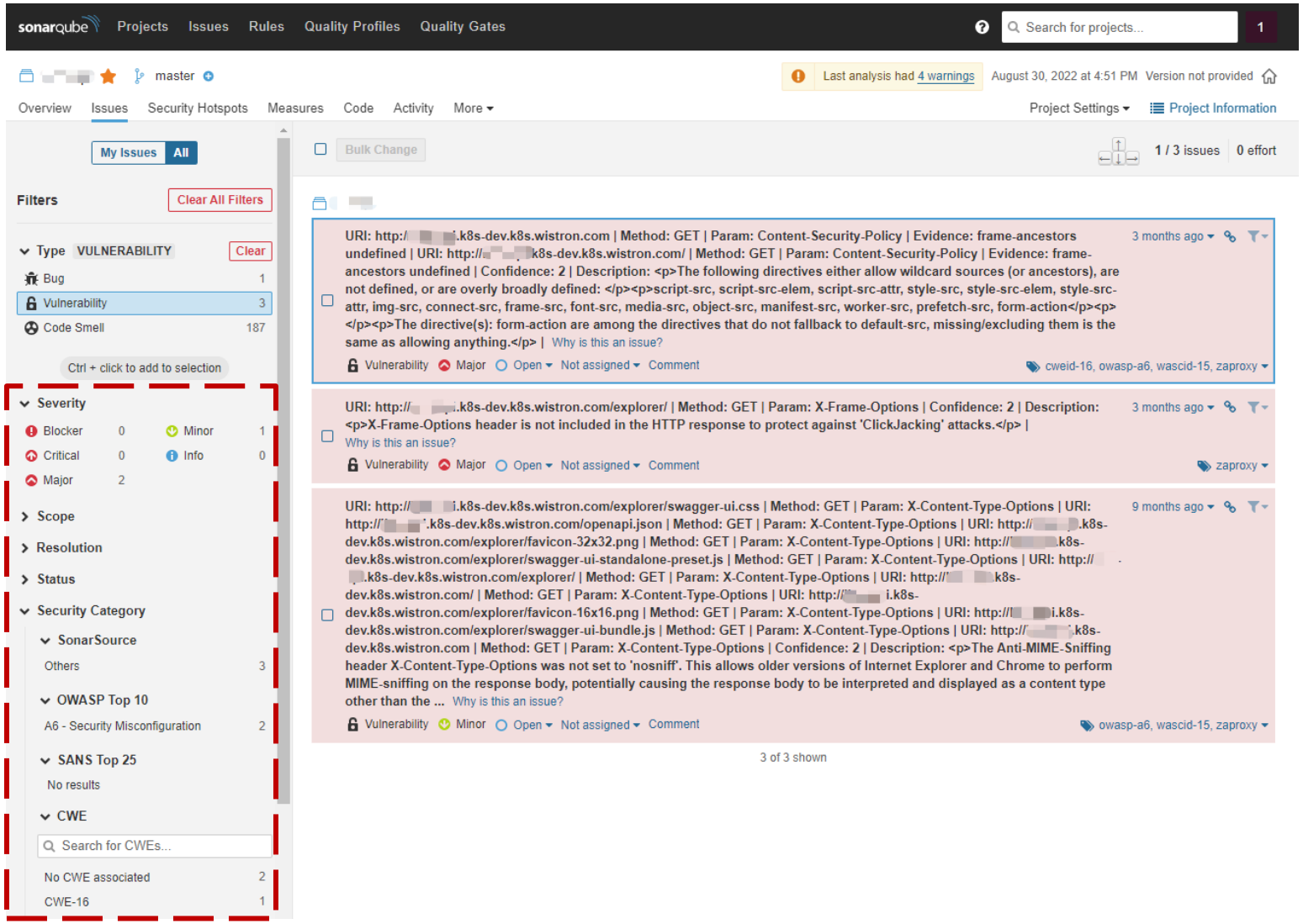
SonarQube – 問題 (Issues)

問題嚴重性等級

- **Blocker** : 錯誤高機率會影響運作中的應用程序行為，*必須立即修復*
- **Critical** : 錯誤低機率會影響生產的應用程序行為，也可能是表示安全漏洞的問題，*必須立即檢查代碼*
- **Major** : 質量缺陷嚴重影響開發人員的程式撰寫
- **Minor** : 質量缺陷會稍微影響開發人員的程式撰寫
- **Info** : 發現錯誤或質量缺陷並提醒

安全相關規則

- CWE (Common Weakness Enumerations)
- OWASP Top 10
- SANS Top 25
- SonarSource : WASC, zaproxy

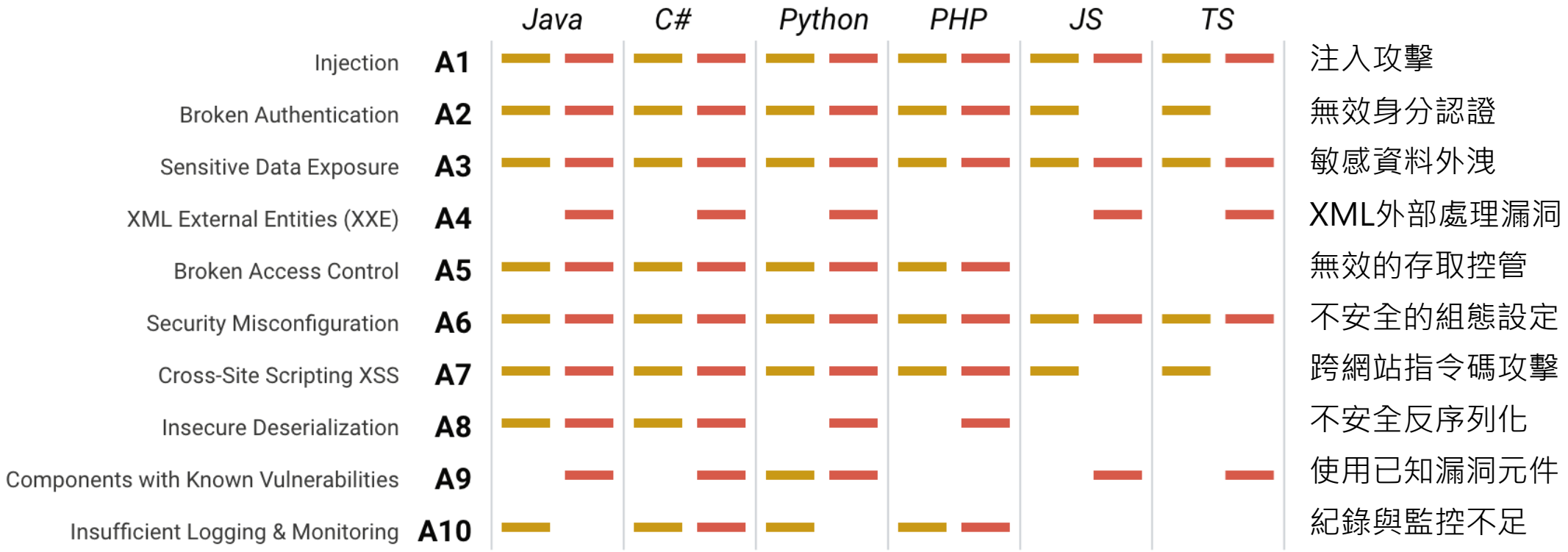


SonarQube – OWASP Top 10

HotSpot
Vulnerability

OWASP Top10 (2017)

OWASP 代表開放式網頁應用程式安全專案，OWASP Top 10 是一個列出廣泛種類弱點的列表，每一個種類可以映射到許多獨立的規則。



SonarQube – SANS Top 25

SANS Top25

元件之間不安全的互動(Insecure Interaction Between Components):

使用不安全的方式在不同元件、模組或程序間傳遞資料

等級	CWE ID	名稱
[1]	CWE-89	不當摧毀 SQL 指令中使用的特殊元素（SQL 注入）
[2]	CWE-78	不當摧毀作業系統指令中使用的特殊元素（作業系統指令注入）
[4]	CWE-79	不當摧毀產生網頁期間的輸入（跨網站指令碼）
[9]	CWE-434	未限定上傳危險類型的檔案
[12]	CWE-352	偽造跨網站要求 (CSRF)
[22]	CWE-601	URL 重新導向到未授信網站（開放重新導向）

有風險的資源管理(Risky Resource Management)

軟體沒有正確的管理重要系統資源, 如: CPU、記憶體的配置、使用、傳遞與釋放而導致的弱點

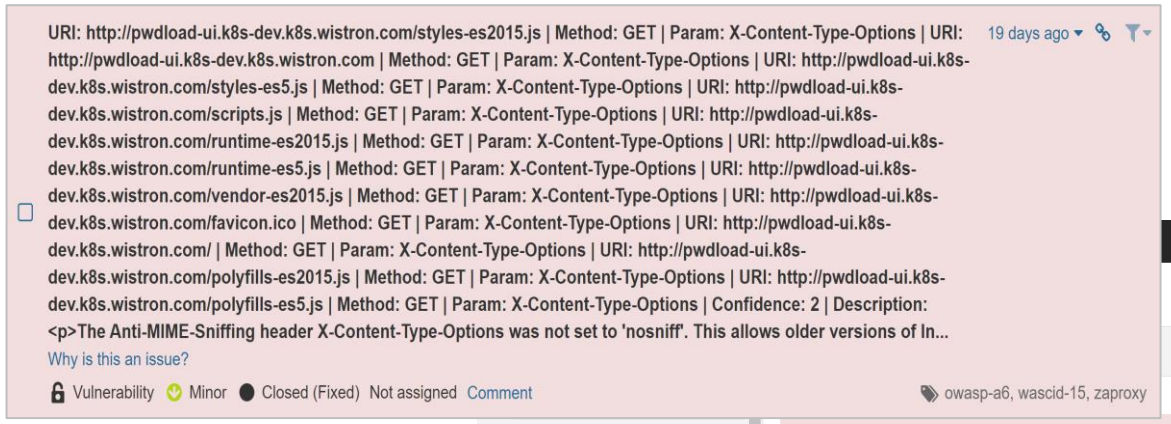
等級	CWE ID	名稱
[3]	CWE-120	未檢查輸入大小即進行緩衝區複製（典型緩衝區溢位）
[13]	CWE-22	不當限制受限目錄的路徑名稱（路徑遍訪）
[14]	CWE-494	未經完整性檢查即下載程式碼
[16]	CWE-829	包含來自未授信控制領域的功能
[18]	CWE-676	使用潛藏危險的功能
[20]	CWE-131	緩衝區大小的計算不正確
[23]	CWE-134	不受控制的格式字串
[24]	CWE-190	整數溢位或折返

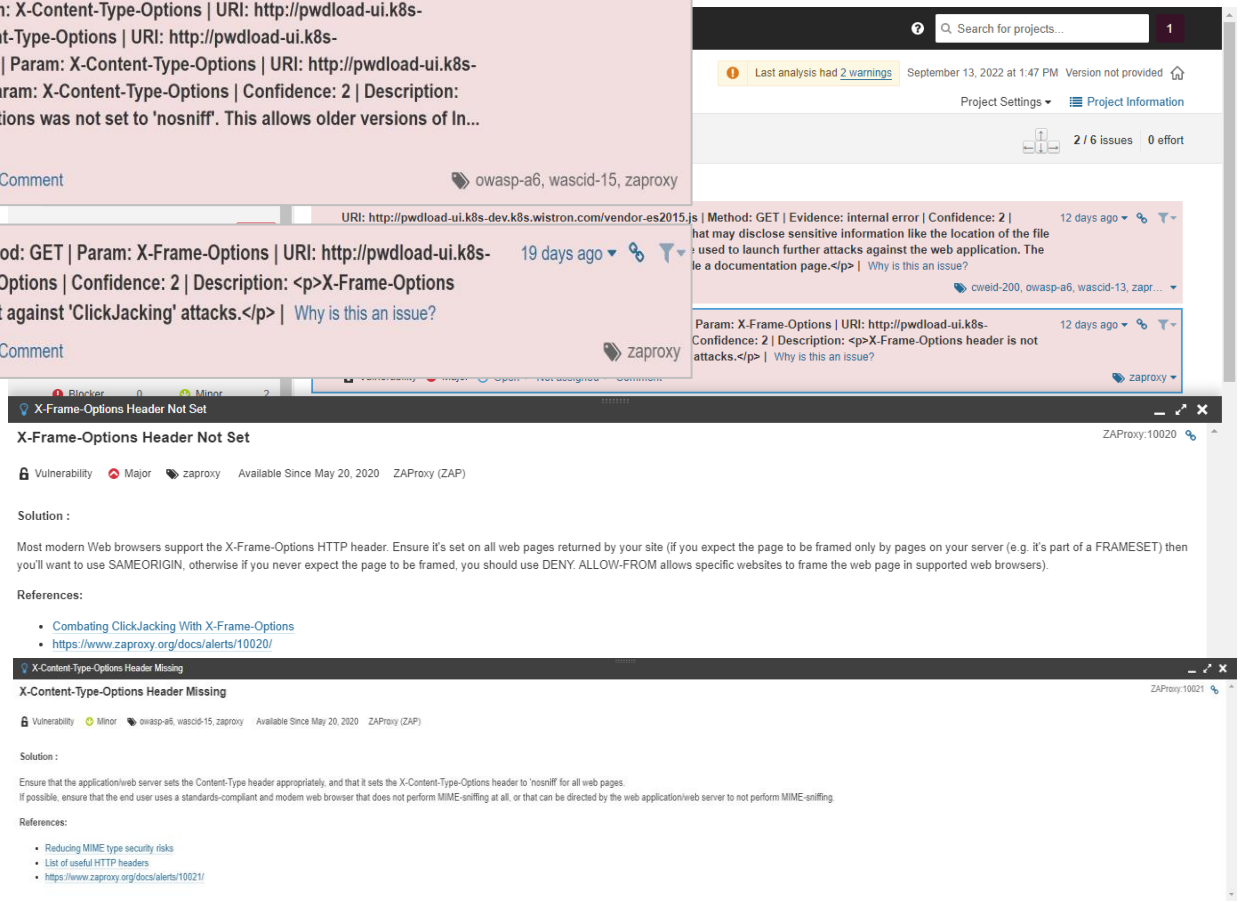
可滲透的防禦(Porous Defenses):

誤用、濫用與輕忽防禦技術所導致的弱點

等級	CWE ID	名稱
[5]	CWE-306	遺漏鑑別重要功能
[6]	CWE-862	遺漏授權
[7]	CWE-798	使用寫在程式中的認證
[8]	CWE-311	遺漏加密機密資料
[10]	CWE-807	倚賴安全決策中的非授信輸入
[11]	CWE-250	使用不必要的專用權來執行
[15]	CWE-863	不正確的授權
[17]	CWE-732	重要資源的許可權指派不正確
[19]	CWE-327	使用毀損或危險的加密演算法
[21]	CWE-307	不當的過多鑑別嘗試次數限制
[25]	CWE-759	使用沒有 Salt 的單向雜湊

SonarQube – 漏洞案例

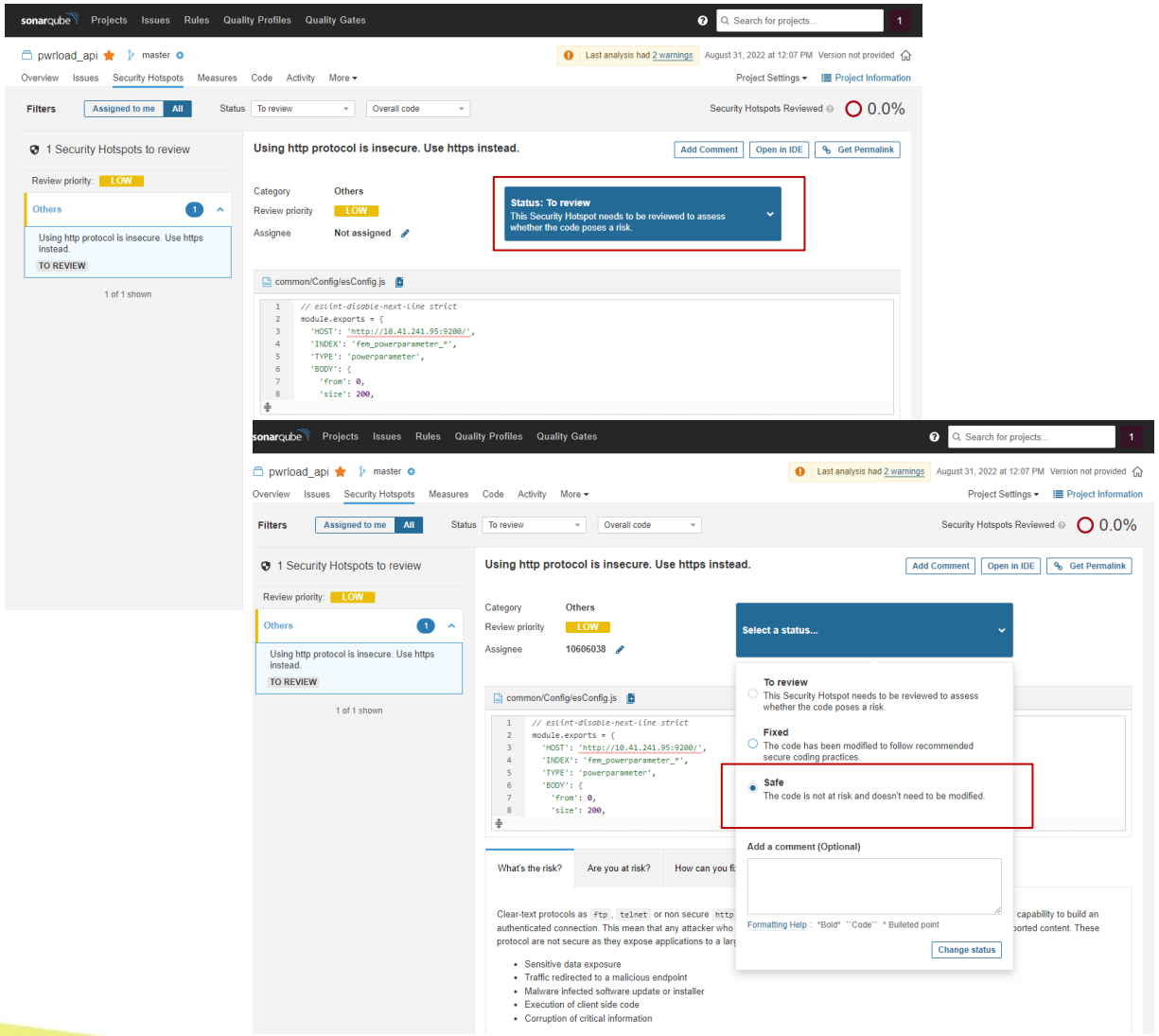





```
server {
  listen *:4200 ; # 指定port to serve

  port_in_redirect off;
  autoindex on;

  location / {
    root /usr/share/nginx/html; # 指定web根目錄
    index index.html; # 指定index為index.html
  }

  add_header X-Frame-Options SAMEORIGIN;
  add_header X-Content-Type-Options nosniff;

  error_page 404  /index.html;

  error_page 500 502 503 504 /50x.html;
      location = /50x.html {
  }
}
```

nginx.conf

SonarQube – 安全疑慮點案例



As is

1 Security Hotspots | 0.0% Reviewed | Security Review E

To be

0 Security Hotspots | 100% Reviewed | Security Review A

ZAP - Web應用程序安全掃描程序

sonarqube Projects Issues Rules Quality Profiles Quality Gates Search for projects... 1

pwrload_ui master Last analysis had 2 warnings September 13, 2022 at 1:47 PM Version not provided

Overview Issues Security Hotspots Measures Code Activity More Project Settings Project Information

Dependency-Check
License Check
ZAP

ZAP Scanning Report

Generated on Tue, 13 Sep 2022 05:46:01

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	2
Low	2
Informational	2

Alerts

Name	Risk Level	Number of Instances
Application Error Disclosure	Medium	1
X-Frame-Options Header Not Set	Medium	2
Private IP Disclosure	Low	1
X-Content-Type-Options Header Missing	Low	11
Information Disclosure - Suspicious Comments	Informational	8
Timestamp Disclosure - Unix	Informational	36

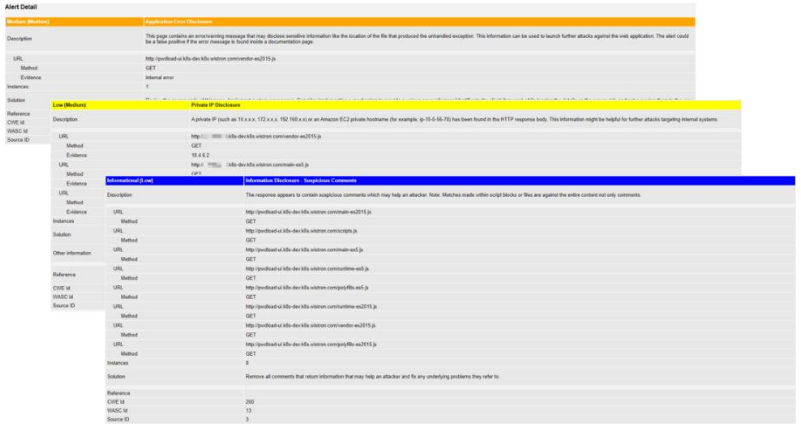
Alert Detail

Medium (Medium)	X-Frame-Options Header Not Set
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	http://pwdload-ui.k8s-dev.k8s.wistron.com/
Method	GET
Parameter	X-Frame-Options
URL	http://pwdload-ui.k8s-dev.k8s.wistron.com
Method	GET
Parameter	X-Frame-Options
Instances	2
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	16
WASC Id	15
Source ID	3

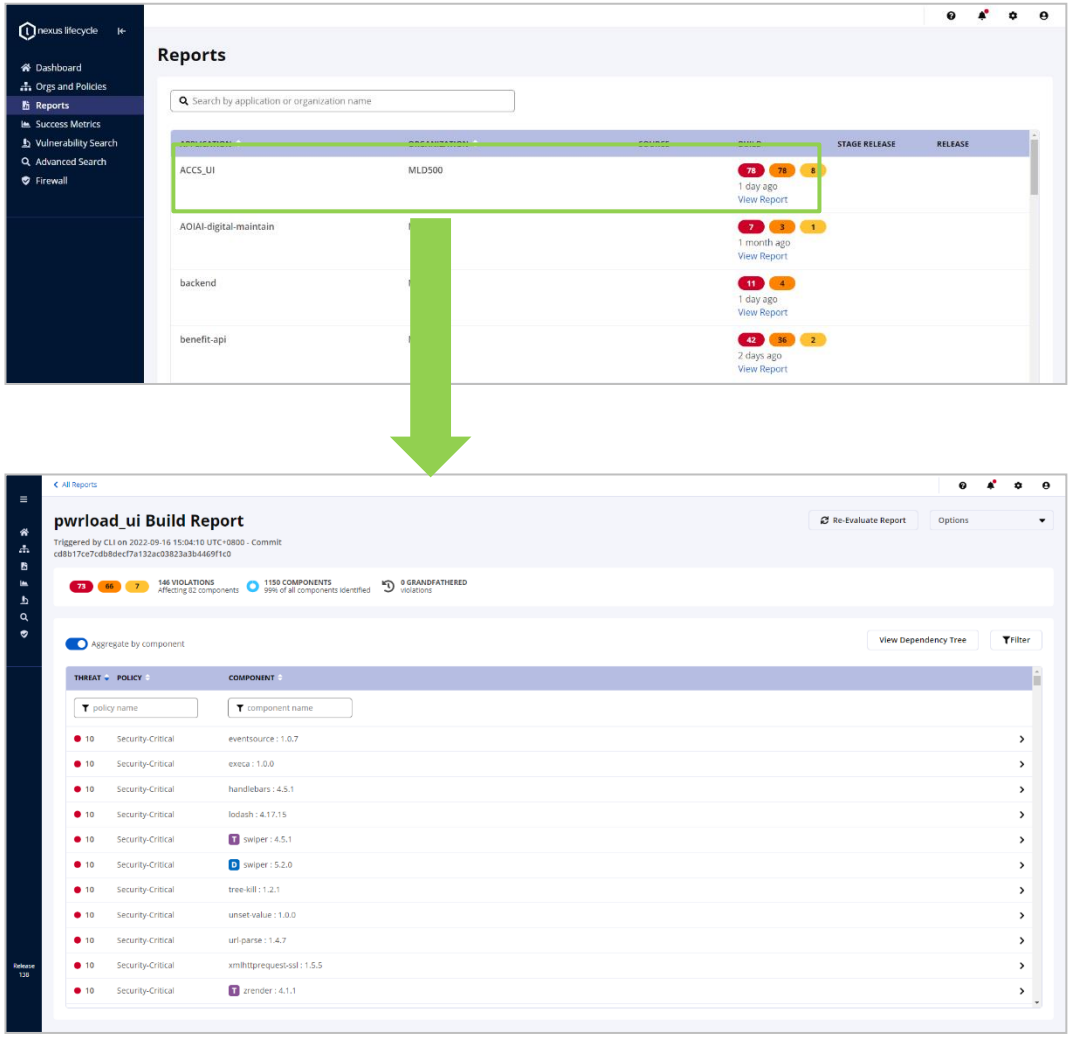
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 8.9.7 (build 52159) - LGPL v3 - Community - Documentation - Plugins - Web API - About

X-Content-Type-Options Header Missing

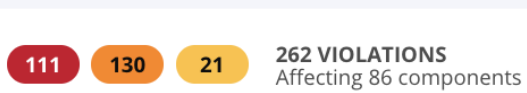
High
Low
Medium
Information



SCA –管理開源元件應用安全

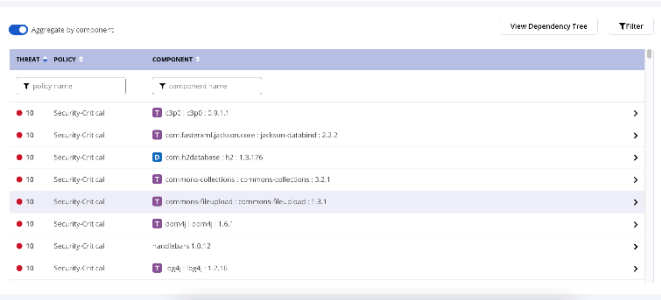


Violation Counts



Identified Component Count

Policy Violations Table



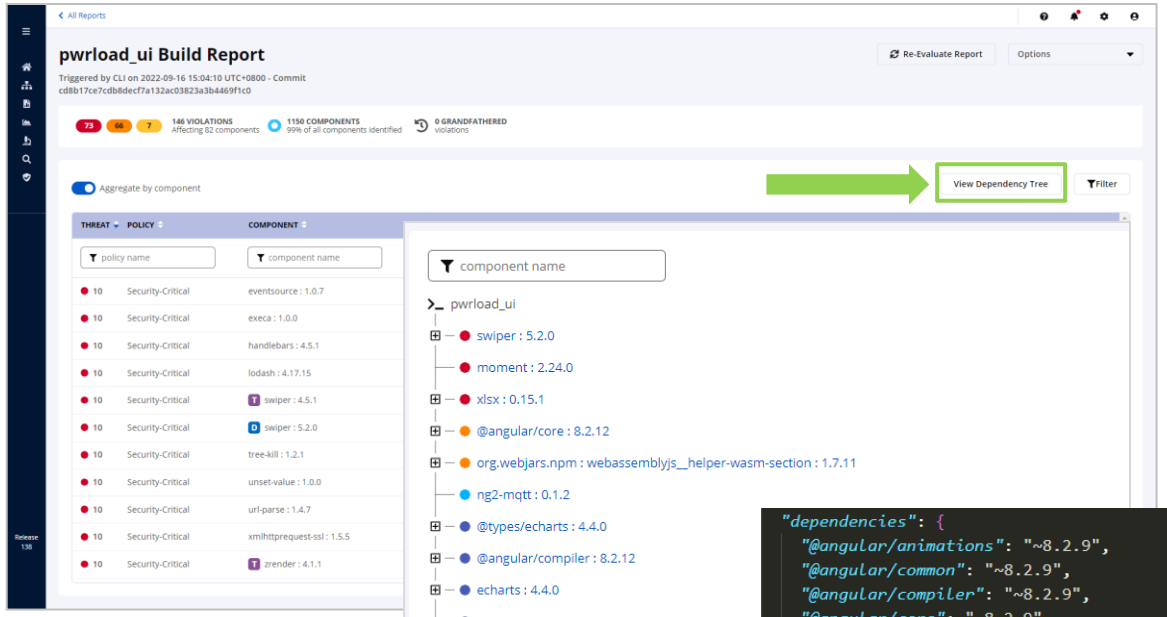
Threat

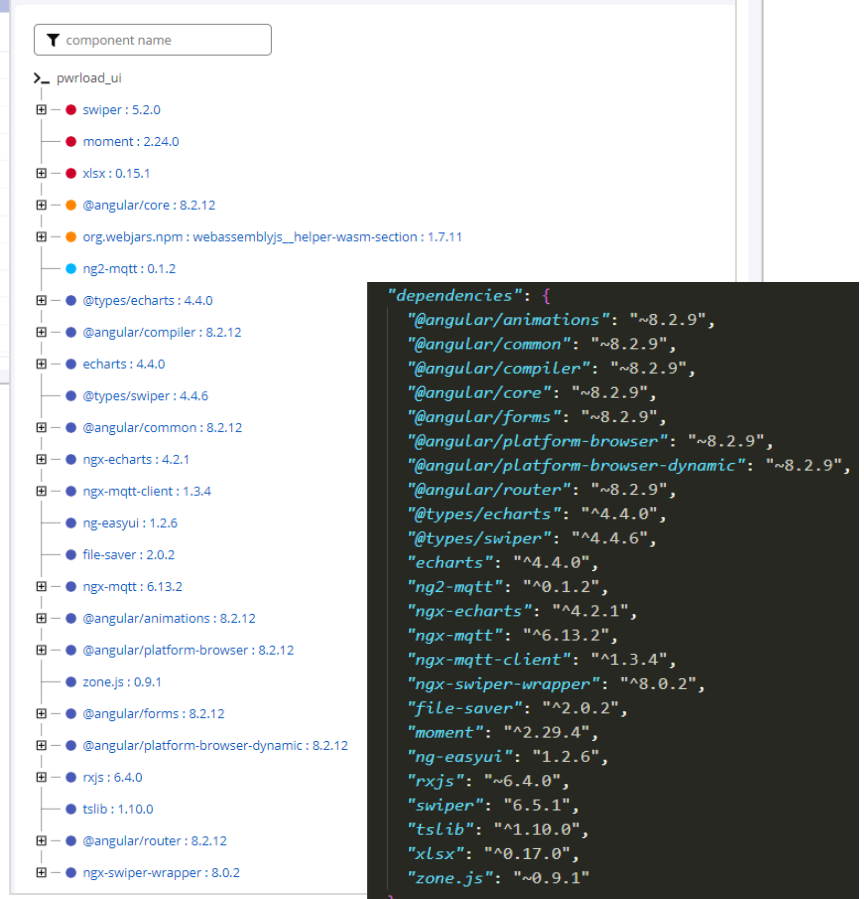
- 8-10 Critical
- 4-7 Severe
- 2-3 Moderate
- 1 Low
- 0 None

Dependency Type

- D — Direct Dependency
- T — Transitive Dependency
- D IS — InnerSource Direct Dependency
- T IS — InnerSource Transitive Dependency
- **No Indicator** — Unknown (dependency info is not available)

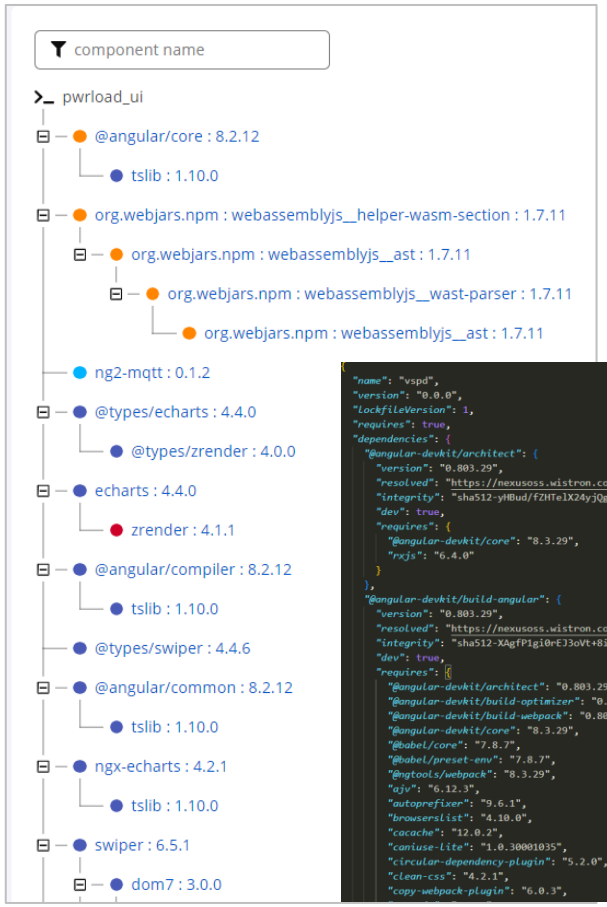
Dependency Tree






```
"dependencies": {
  "@angular/animations": "~8.2.9",
  "@angular/common": "~8.2.9",
  "@angular/compiler": "~8.2.9",
  "@angular/core": "~8.2.9",
  "@angular/forms": "~8.2.9",
  "@angular/platform-browser": "~8.2.9",
  "@angular/platform-browser-dynamic": "~8.2.9",
  "@angular/router": "~8.2.9",
  "@types/echarts": "^4.4.0",
  "@types/swiper": "^4.4.6",
  "echarts": "^4.4.0",
  "ng2-mqtt": "^0.1.2",
  "ngx-echarts": "^4.2.1",
  "ngx-mqtt": "^6.13.2",
  "ngx-mqtt-client": "^1.3.4",
  "ngx-swiper-wrapper": "^8.0.2",
  "file-saver": "^2.0.2",
  "moment": "^2.29.4",
  "ng-easyui": "1.2.6",
  "rxjs": "~6.4.0",
  "swiper": "6.5.1",
  "tslib": "^1.10.0",
  "xlsx": "^0.17.0",
  "zone.js": "~0.9.1"
},
```

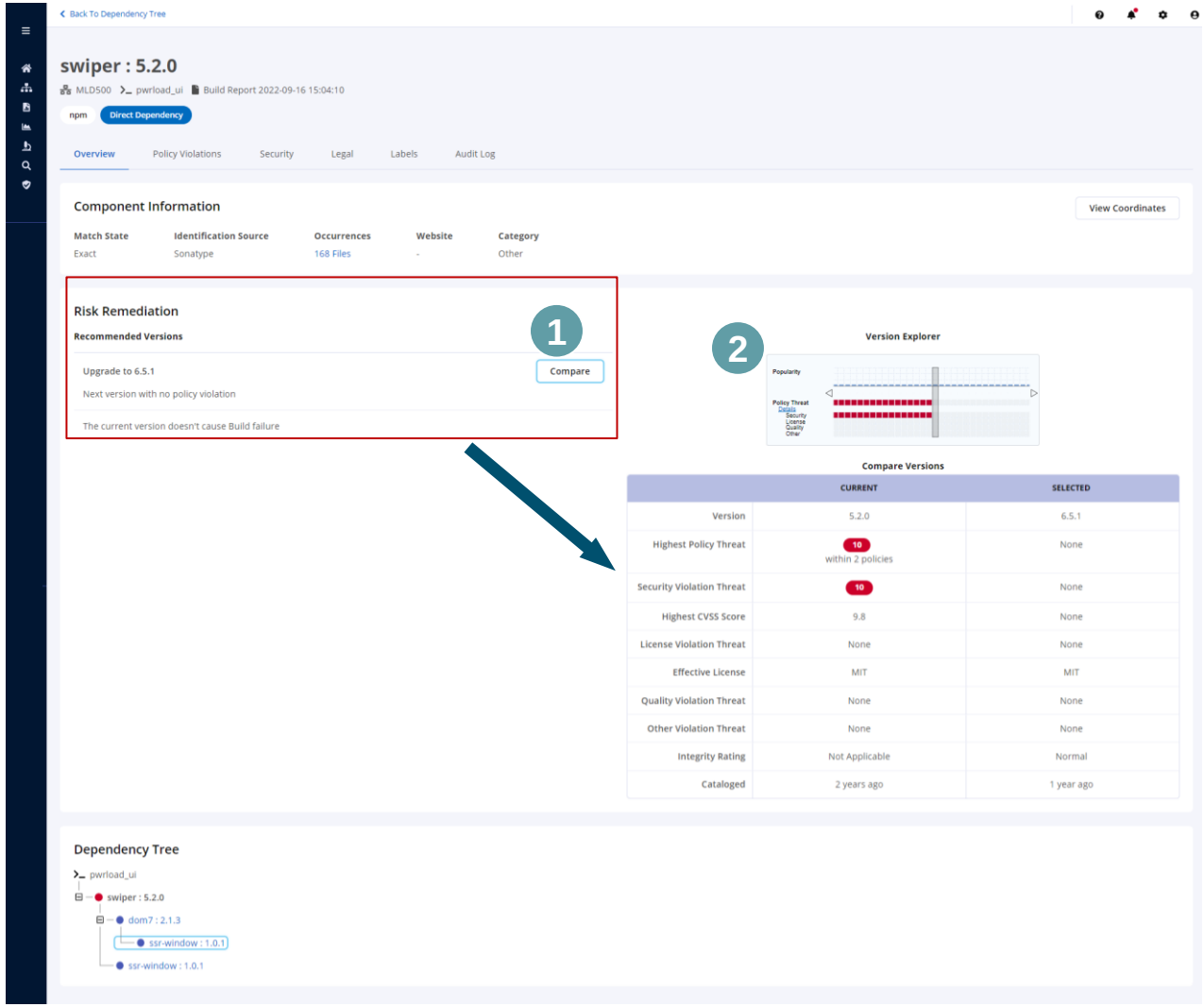
package.json




```
{
  "name": "vspd",
  "version": "0.0.0",
  "lockfileVersion": 1,
  "requires": true,
  "dependencies": {
    "@angular-devkit/architect": {
      "version": "0.803.29",
      "resolved": "https://nexusoss.wistron.com/repository/npm-group/@angular-devkit/architect/-/architect-0.803.29.tgz",
      "integrity": "sha512-yHBud/fZHTelX24yjQg5lefZrfIebruoFTGeOwF0JdX8+KiHcTIxS4LOnUTYriasfHarcHRFXBAV/bRm+wv5ow==",
      "dev": true,
      "requires": {
        "@angular-devkit/core": "8.3.29",
        "rxjs": "6.4.0"
      }
    },
    "@angular-devkit/build-angular": {
      "version": "0.803.29",
      "resolved": "https://nexusoss.wistron.com/repository/npm-group/@angular-devkit/build-angular/-/build-angular-0.803.29.tgz",
      "integrity": "sha512-XAgfP1gi0rEJ3oVt+8ipvS5RfPNbeK5r2n8Ll2H3xkKjU0p1PN8+S6/0XVBtmMfeQ06SJWEAKFcAYqrxXhVTzw==",
      "dev": true,
      "requires": {
        "@angular-devkit/architect": "0.803.29",
        "@angular-devkit/build-optimizer": "0.803.29",
        "@angular-devkit/build-webpack": "0.803.29",
        "@angular-devkit/core": "8.3.29",
        "@babel/core": "7.8.7",
        "@babel/preset-env": "7.8.7",
        "@ngtools/webpack": "8.3.29",
        "ajv": "6.12.3",
        "autoprefixer": "9.6.1",
        "browserslist": "4.10.0",
        "cacache": "12.0.2",
        "caniuse-lite": "1.0.30001035",
        "circular-dependency-plugin": "5.2.0",
        "clean-css": "4.2.1",
        "copy-webpack-plugin": "6.0.3",
```

package-lock.json

Component Detail Page



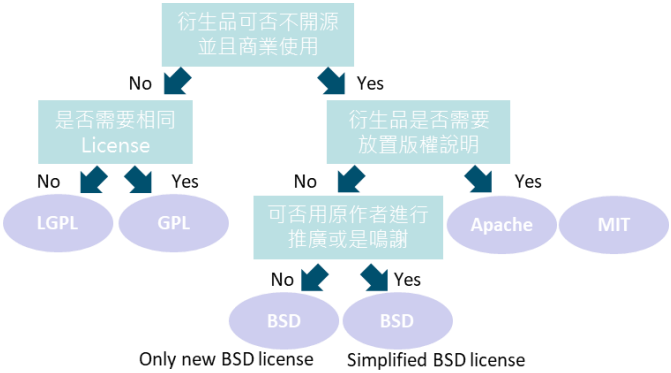
如何解決 Security issue?

1. 點擊 **Compare** 比較元件版本
2. 查看元件版本熱力圖尋找無威脅版本
3. 升/降級至對應無威脅版本

Software License

什麼是軟體授權條款(Software License)?

是一個開源概念的衍生物，用來管理軟體的使用與重新分配的法律文件



授權方式

Copyleft (著作傳)

概念著重在保護作品被眾人自由使用，**強制衍生作品要使用相同授權條款**來確保作品被使用的自由以促進軟體生態整體的進展

Copyright (著作權)

沒有讓作品被自由使用的特性

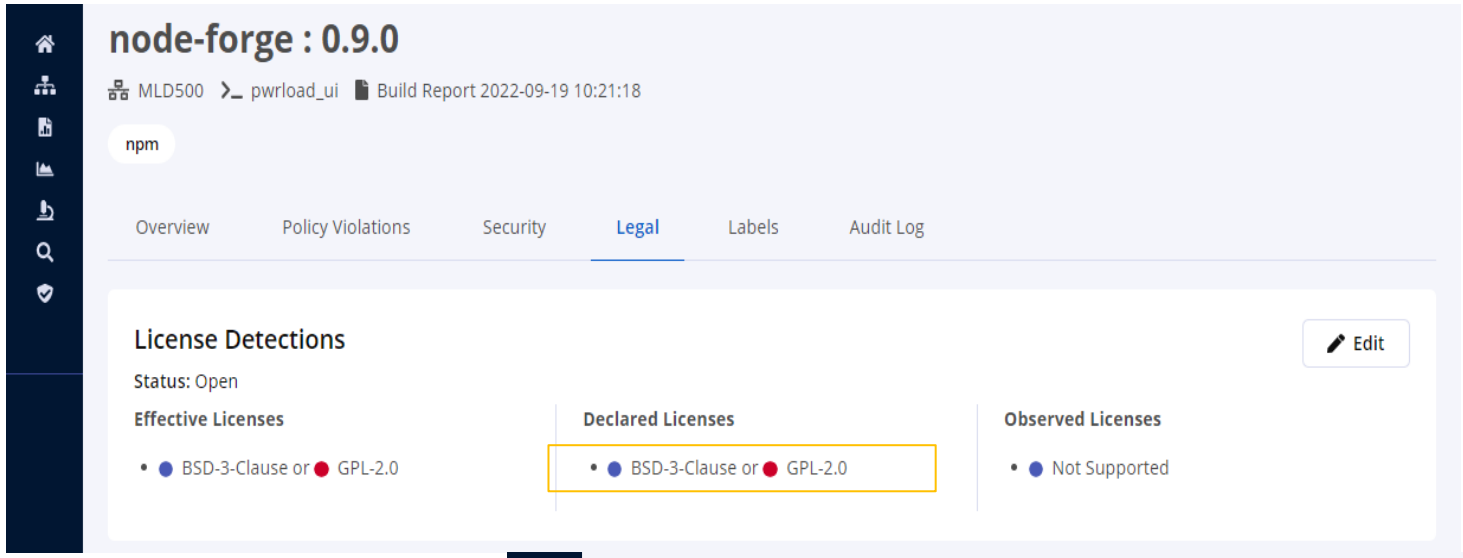
Copycenter

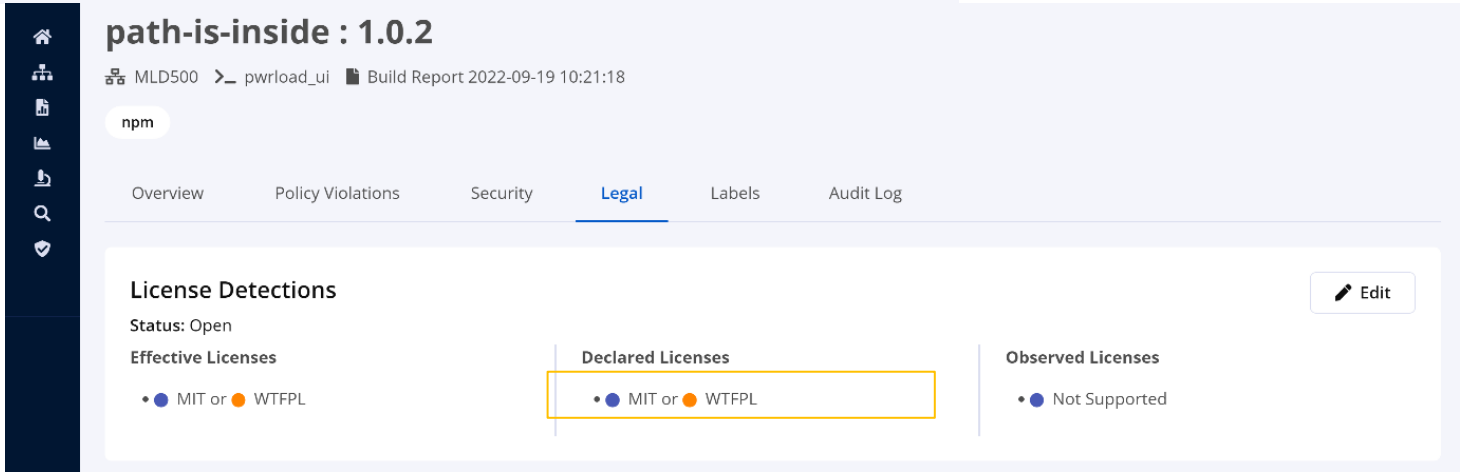
在保護作品被眾人自由使用的權利，但它不像copyleft一樣，而是**賦予後續使用者選擇其他授權條款的自由**。

常見授權條款

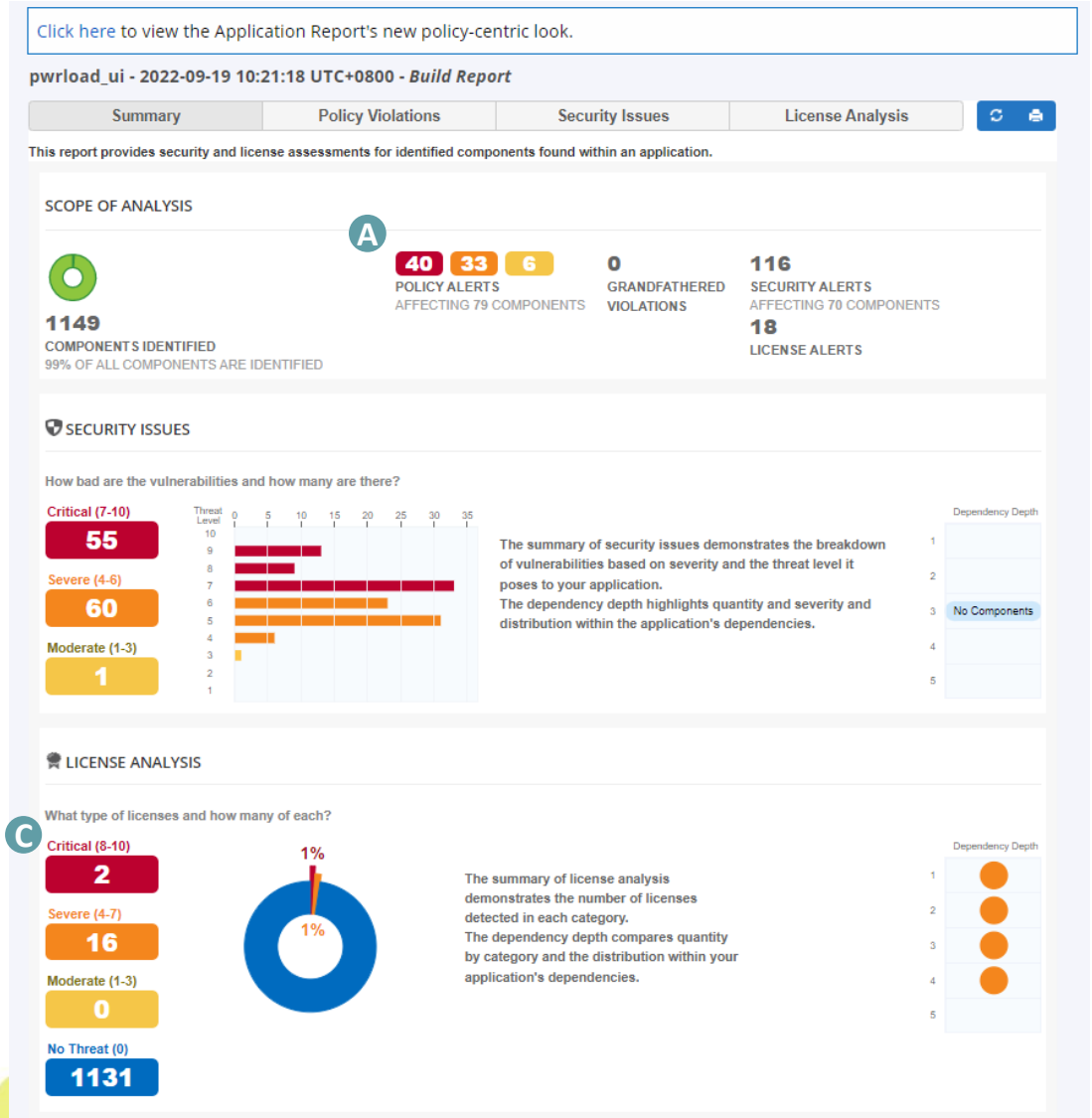
License	GPL	LGPL	BSD	APACHE	MIT
授權方式	Copyleft	Copyleft	Copycenter	Copycenter	Copycenter
公開原始碼	✓	✓			
以同樣方式授權	✓	✓			
標註修改的部分	✓	✓		✓	
必須包含Copyright	✓	✓	✓	✓	✓
必須包含License	✓	✓		✓	✓
高感染性	✓	✓			
著名開源軟體	Notepadd++ MySQL	Qt	Django React	Kubernetes Nifi Kafka	Angular JQuery .Net Core

Legal Tab: view the License Detections box



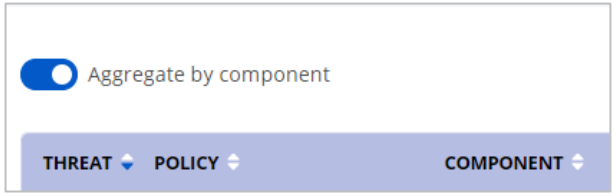


Legacy report



A SCOPE OF ANALYSIS

Policy violation table 聚合元件後的結果



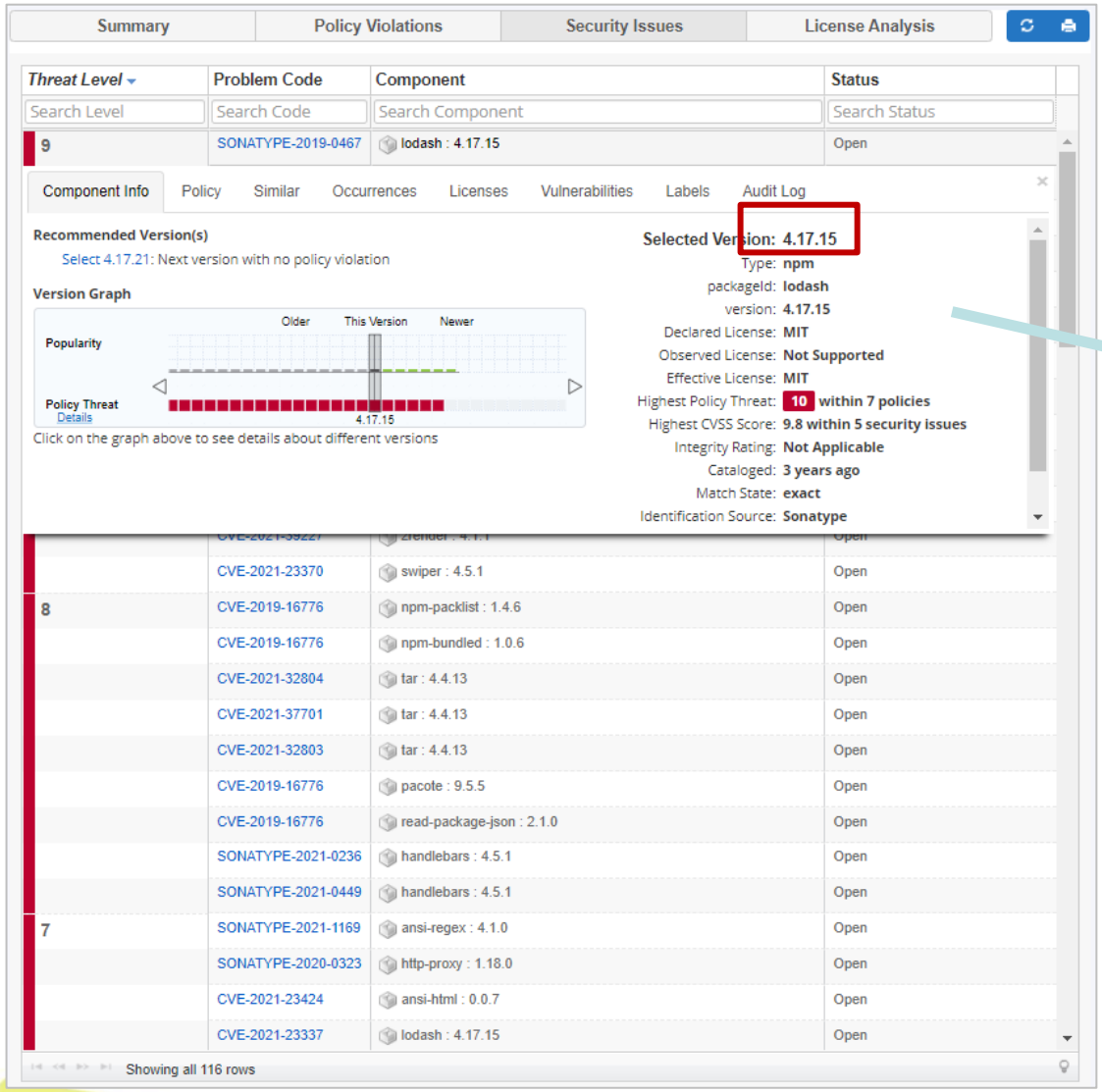
B SECURITY ISSUES

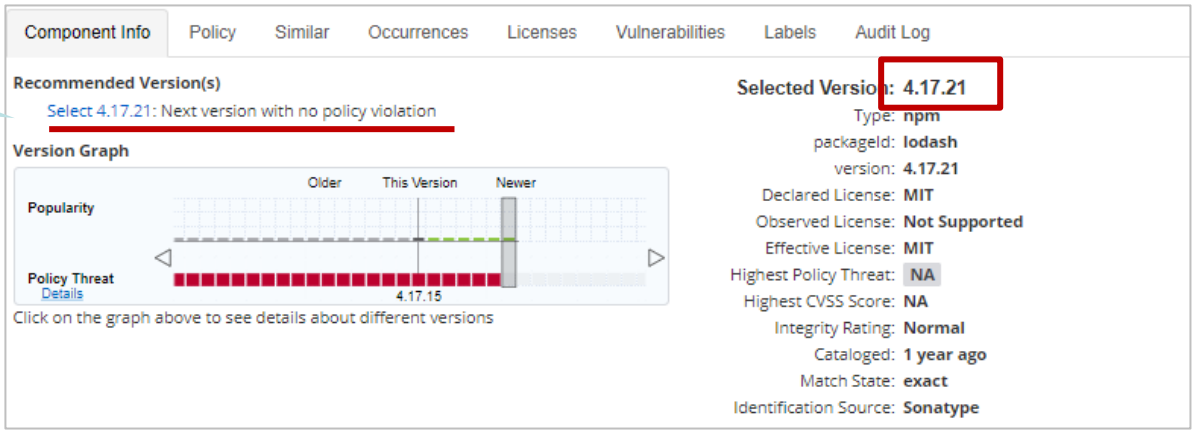
Policy violation table中，Policy為Security 的數量

C LICENSE ANALYSIS

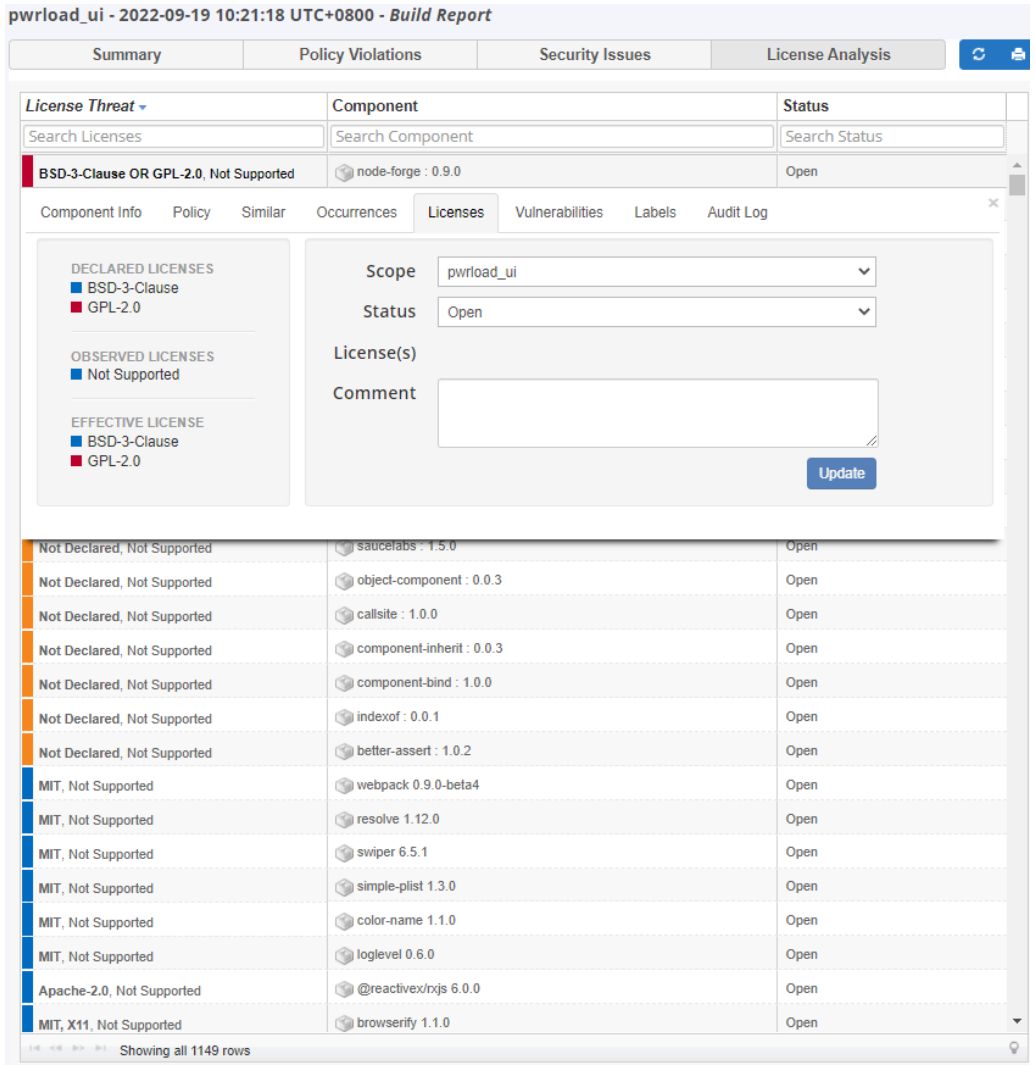
顯示被偵測到的license數量

Security Issues





License Analysis



Thank You !

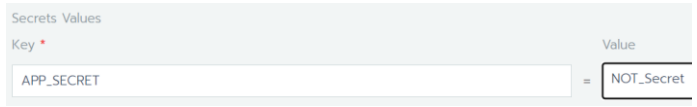
Bill-30 min

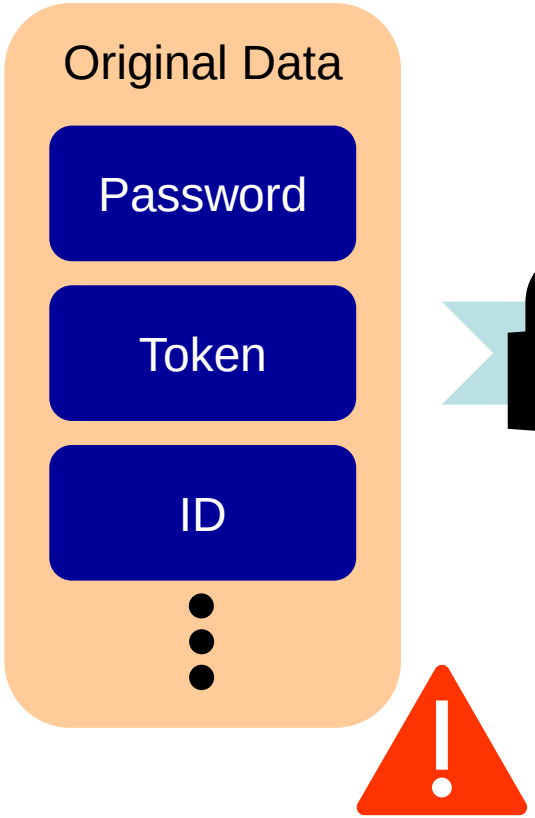
CD相關

Protect your sensitive data

authorization: Bearer ey[redacted]

```
DB_CONNECT: {
  name: 'postgresdb',
  connector: 'postgresql',
  url: '',
  host: '10.xx.xx.xx',
  port: 5432,
  user: 'demo',
  password: 'dmoe123$zxcV',
  database: 'postgresdb',
},
```





Protect your sensitive data

jasypt.
JAVA·SIMPLIFIED·ENCRYPTION

Sealed Secrets

Basic Encryption/Decryption

K8S secrets Protection

Http protocol Protection

Jasypt

- Introduction
  - Java Simplified Encryption
    - Support both sync encryption and digest.

```
C:\Users\10901143\Desktop\secc\jasypt-1.9.3\bin>listAlgorithms

DIGEST ALGORITHMS:   [MD2, MD5, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256,
 SHA3-384, SHA3-512]

PBE ALGORITHMS:      [PBEWITHHMACSHA1ANDAES_128, PBEWITHHMACSHA1ANDAES_256, PBEWITHHMACSHA224ANDAES_128, PBEWITHHMACSHA2
24ANDAES_256, PBEWITHHMACSHA256ANDAES_128, PBEWITHHMACSHA256ANDAES_256, PBEWITHHMACSHA384ANDAES_128, PBEWITHHMACSHA384AN
DAES_256, PBEWITHHMACSHA512ANDAES_128, PBEWITHHMACSHA512ANDAES_256, PBEWITHMD5ANDDES, PBEWITHMD5ANDTRIPLEDES, PBEWITHSHA
1ANDDESEDE, PBEWITHSHA1ANDRC2_128, PBEWITHSHA1ANDRC2_40, PBEWITHSHA1ANDRC4_128, PBEWITHSHA1ANDRC4_40]
```

    - Also support Node Js.

jasypt. JAVA SIMPLIFIED ENCRYPTION

Jasypt

- Encryption/Decryption
  - encrypt input="要加密的資料" password="加解密金鑰"
  - decrypt input="要解密的資料" password="加解密金鑰"

```
C:\Users\10901143\Desktop\secc\jasypt-1.9.3\bin>encrypt input="demo123$zxcV" password="demo_test_key"

----ENVIRONMENT-----------------

Runtime: Azul Systems, Inc. OpenJDK 64-Bit Server VM 17.0.1+12-LTS


----ARGUMENTS-------------------

input: demo123$zxcV
password: demo_test_key


----OUTPUT----------------------

HGT1W9OB2+MNjQYODaU4ED7OSa8Na3n2
```

```
C:\Users\10901143\Desktop\secc\jasypt-1.9.3\bin>decrypt input="HGT1W9OB2+MNjQYODaU4ED7OSa8Na3n2" password="demo_test_key"

----ENVIRONMENT-----------------

Runtime: Azul Systems, Inc. OpenJDK 64-Bit Server VM 17.0.1+12-LTS


----ARGUMENTS-------------------

input: HGT1W9OB2+MNjQYODaU4ED7OSa8Na3n2
password: demo_test_key


----OUTPUT----------------------

demo123$zxcV
```

jasypt. JAVA SIMPLIFIED ENCRYPTION

Jasypt

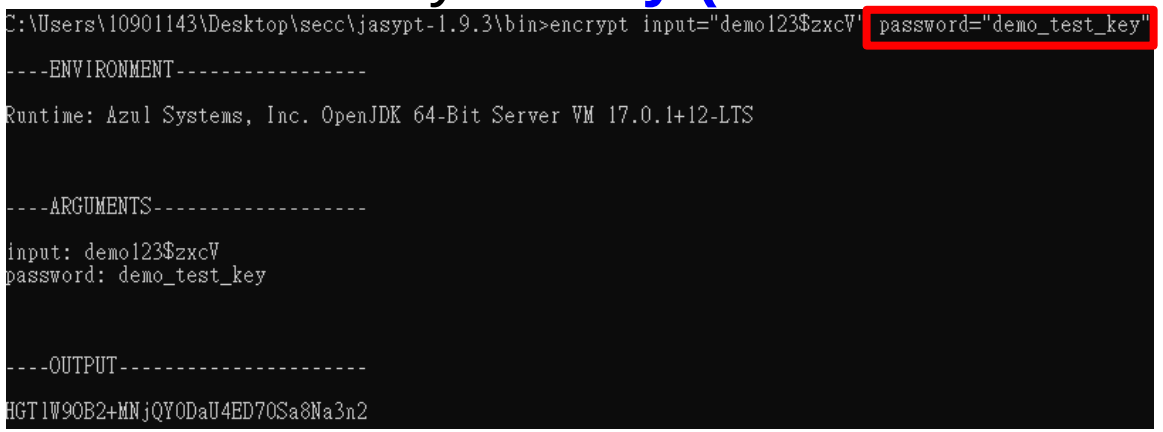
- Digest
  - digest input="要加密的資料"

```
C:\Users\10901143\Desktop\secc\jasypt-1.9.3\bin>digest input="demo123$zxcV"
----ENVIRONMENT-----------------
Runtime: Azul Systems, Inc. OpenJDK 64-Bit Server VM 17.0.1+12-LTS


----ARGUMENTS-------------------
input: demo123$zxcV


----OUTPUT----------------------
zoKcn86n2GJYE7wf6IAjbOBXnwy6TSk3
```

jasypt. JAVA SIMPLIFIED ENCRYPTION

Sealed secret

- Question
  - Where to store your **key (or other secret info.)** when deployment?



  - The data in secret component of k8s is **NOT** encrypted.

```
apiVersion: v1
data:
  POSTGRES_DB: dGVzdA==
  POSTGRES_PASSWORD: dGVzdA==
  POSTGRES_USER: dGVzdA==
```

Base 64 encode only.

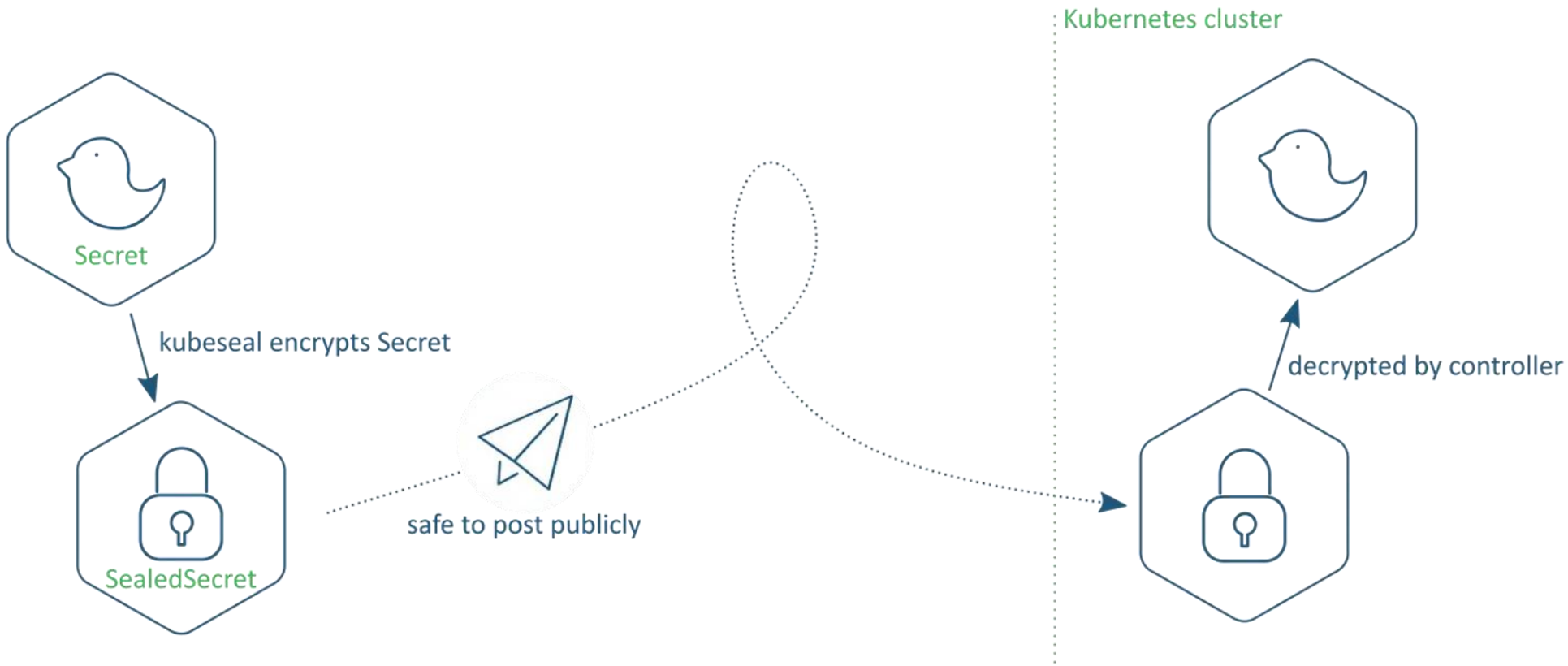
Sealed Secrets

wistron

Sealed secret

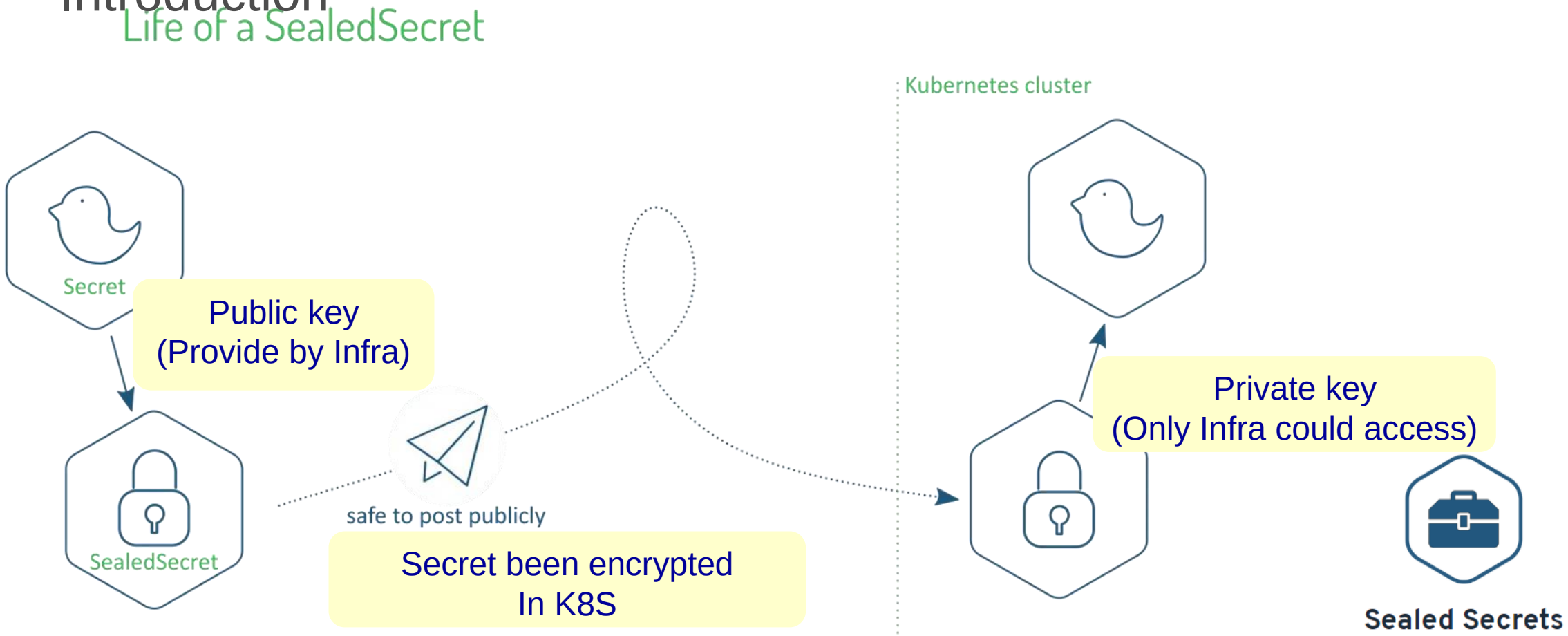
- Introduction

Life of a SealedSecret



Sealed secret

- Introduction

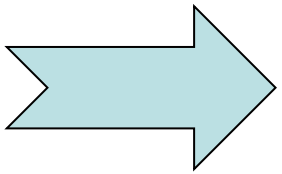


Sealed secret

- Sealed secret
  - Encrypt your secret.yaml with kubeseal
    - kubeseal -- cert CERT檔 -o yaml < 原始yaml > 新yaml檔名

```
C:\Users\10901143\Desktop\secc\kubeseal-0.17.3-windows-amd64>kubeseal -- cert gdc-dev.cert -o yaml < bill-test-ori.yaml
> bill-test-sec.yaml

C:\Users\10901143\Desktop\secc\kubeseal-0.17.3-windows-amd64>
```

```yaml
apiVersion: v1
data:
  POSTGRES_DB: dGVzdA==
  POSTGRES_PASSWORD: dGVzdA==
  POSTGRES_USER: dGVzdA==
kind: Secret
metadata:
  name: billtest-sec
  namespace: corphrdx
type: Opaque
```



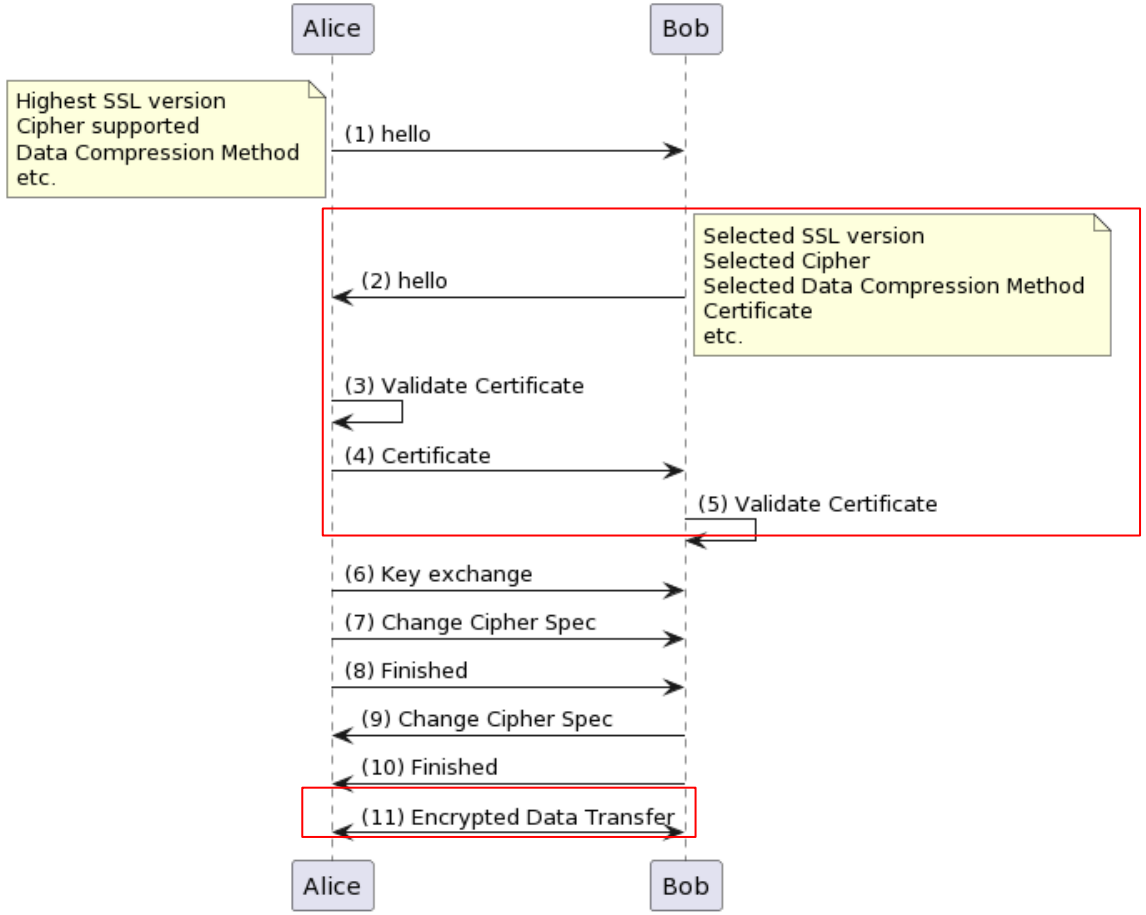
```yaml
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
  creationTimestamp: null
  name: billtest-sec
  namespace: corphrdx
spec:
  encryptedData:
    POSTGRES_DB: AgAJQyv2l8vovDMWcfGYslCRTme
    POSTGRES_PASSWORD: AgCpGi2lH9o4Rfmf83+pa
    POSTGRES_USER: AgAorzKPEKaIDblwNKfvjTlnJ
  template:
    data: null
    metadata:
      creationTimestamp: null
      name: billtest-sec
      namespace: corphrdx
    type: Opaque
```

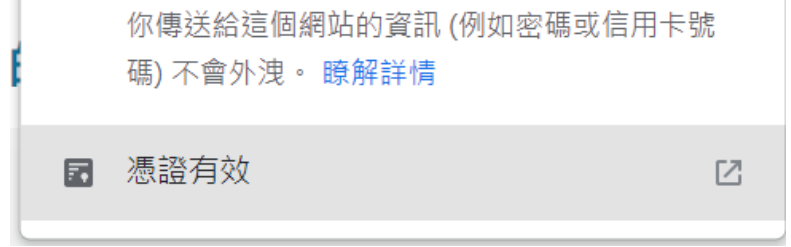
Sealed Secrets

wistron

SSL

- Protect your connection.





SSL

- Set your SSL by your need.
  - Use K8S default certificate.
    - Prepared by infra. No need to apply.
  - Create your own certificate.
    - Apply DNS > create CERT > Register to Wistron CERT server.
  - Apply *.wistron.com certificate.
    - Apply DNS & CERT.

Internal CERT

Issued To

Common Name (CN)	*.k8sprd-whq.k8s.wistron.com
Organization (O)	WISTRON
Organizational Unit (OU)	ML6400

Issued To

Common Name (CN)	esgportal.wistron.com
Organization (O)	Wistron Corporation
Organizational Unit (OU)	MIS

Public CERT

Issued To

Common Name (CN)	*.wistron.com
Organization (O)	WISTRON CORPORATION
Organizational Unit (OU)	<Not Part Of Certificate>

Conclusion

jasypt.
JAVA SIMPLIFIED ENCRYPTION

Sealed Secrets

Protect your **runtime data**

Protect your **deployment**

Protect your **transmission**

Thank You !