

+ Galaxy Software Services

應用系統安全把關 從流程考量做起

主講人 資訊安全事業處
資安經理 郭俐佳

GSS 叢揚資訊
Galaxy Software Services



國家產業創新獎
卓越中堅企業

2018

大綱

AGENDA

- 應用系統的威脅
- 開發人員資安意識提升
- 第三方元件管控
- 傳統應用系統 VS 行動化應用系統
- 持續整合與部署平台
- 結論



國家產業創新獎
卓越中堅企業

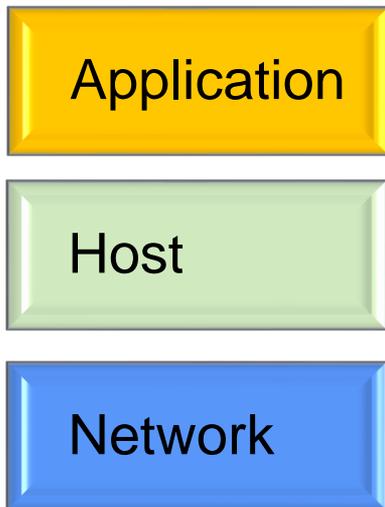
Chapter

1

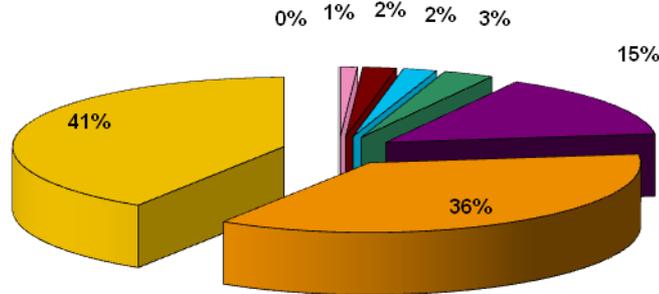
應用系統的威脅

應用系統的安全問題層出不窮

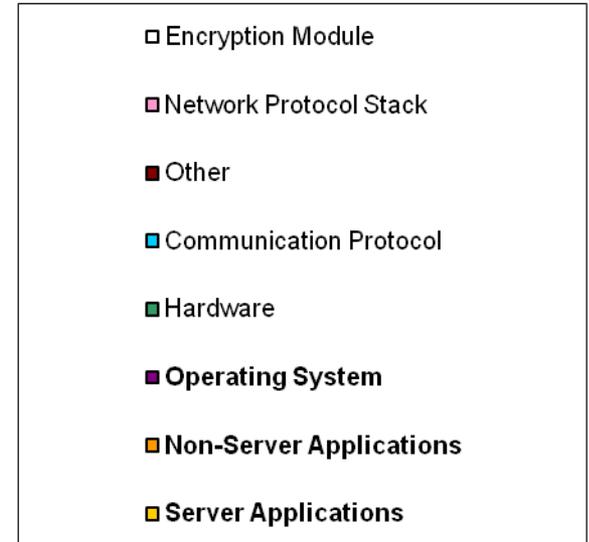
- Gartner
 - 75% of hacks occur at the application level
- NIST
 - 92% of reported vulnerabilities are in apps, not networks



75% of hacks occur at the application level



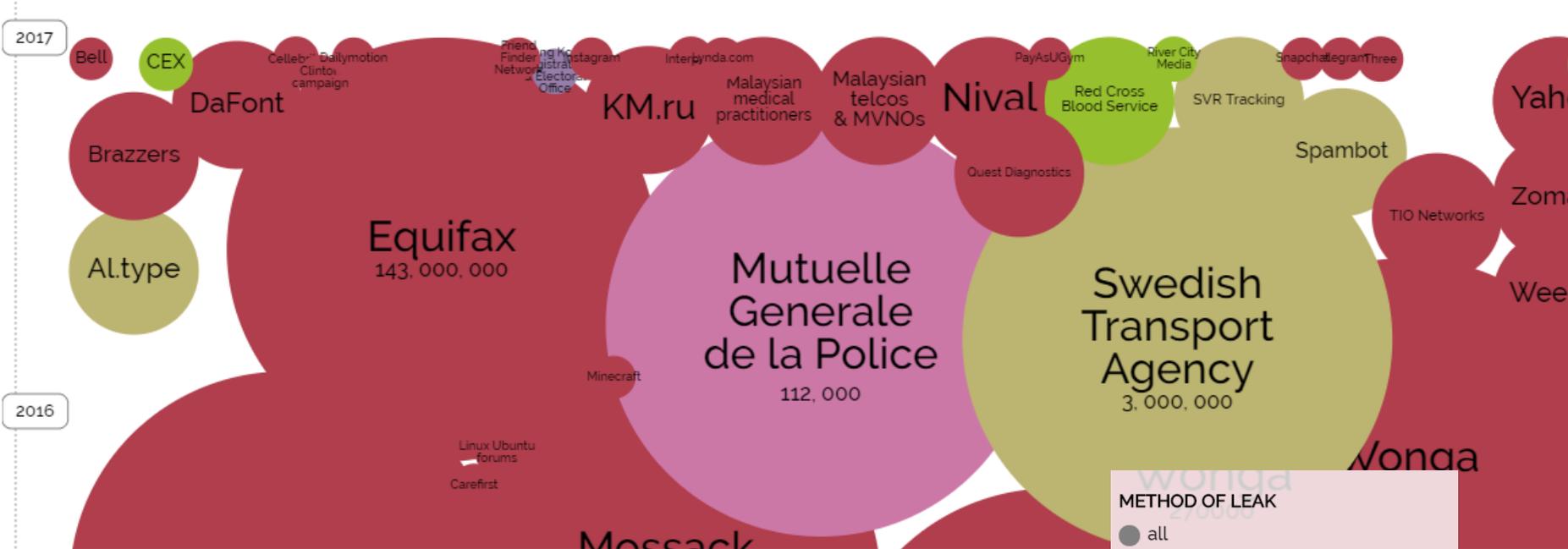
Source: NIST



資安大數據

Selected losses greater than 30,000 records
(updated 05th Jan 2018)

YEAR BUBBLE COLOUR YEAR METHOD OF LEAK BUBBLE SIZE NO OF RECORDS STOLEN DATA SENSITIVITY

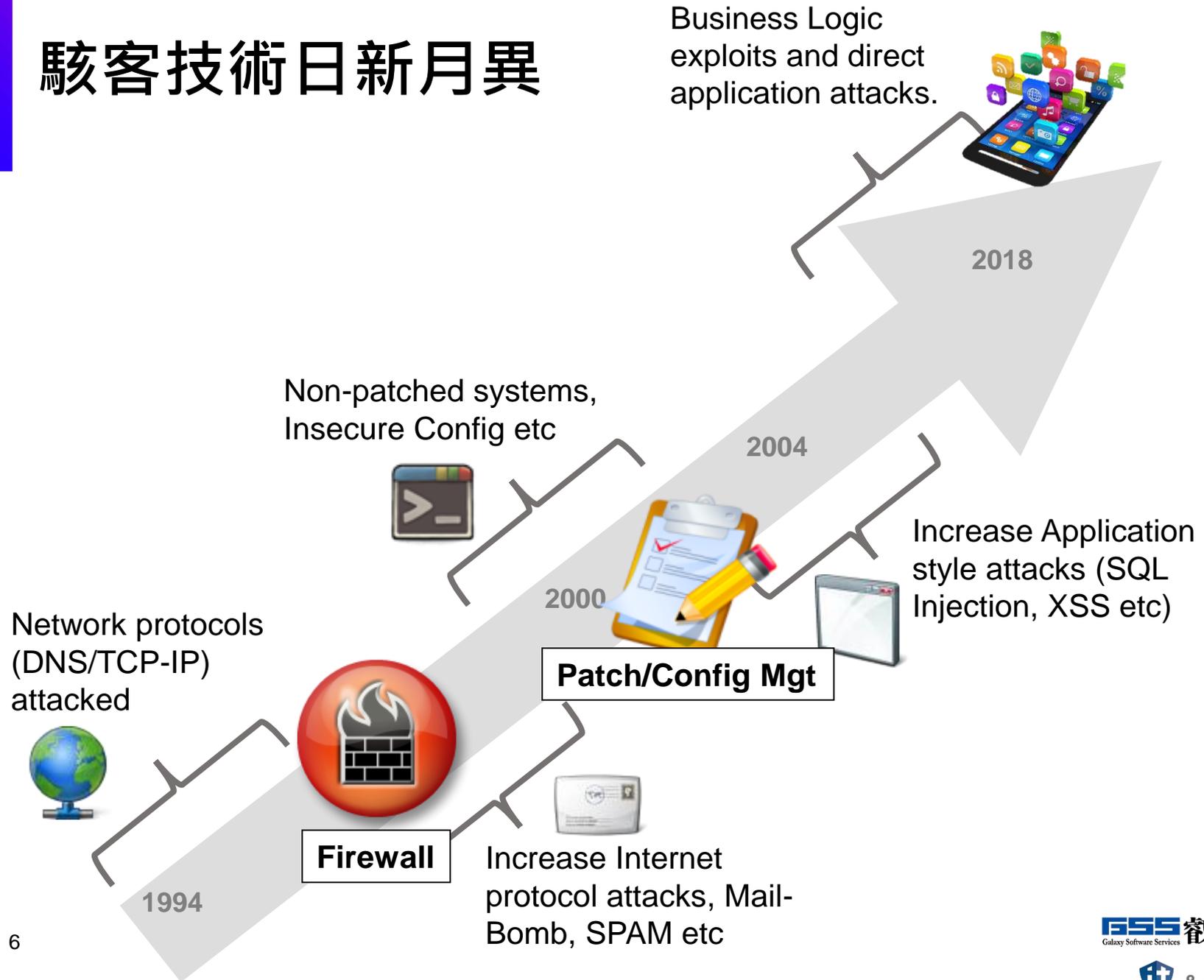


METHOD OF LEAK

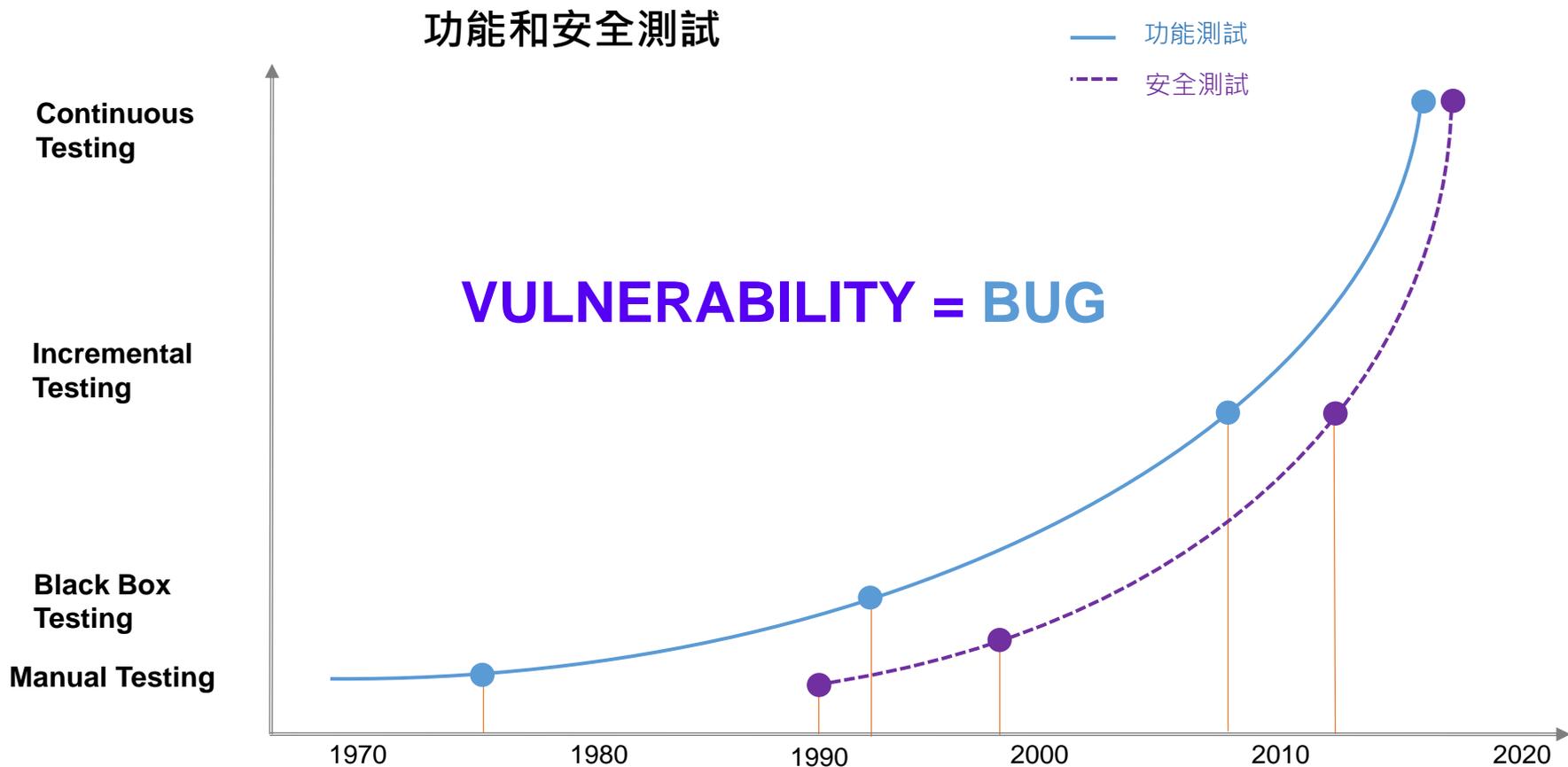
- all
- accidentally published
- hacked
- inside job
- lost / stolen device or media
- poor security

駭客技術日新月異

Business Logic exploits and direct application attacks.



測試方式的演進



OWASP Top 10 2017

-  **A1: Injection**
-  A2: Broken Authentication
-  A3: Sensitive Data Exposure
-  A4: XML External Entities (XXE)
-  A5: Broken Access Control (As it was in 2004)
-  A6: Security Mis-configuration
-  **A7: Cross-Site Scripting (XSS)**
-  A8: Insecure Deserialization
-  **A9: Using Components with Known Vulnerabilities**
-  A10: Insufficient Logging & Monitoring

Mobile OWASP Top 10 Risk 2016

 M1: Improper Platform Usage

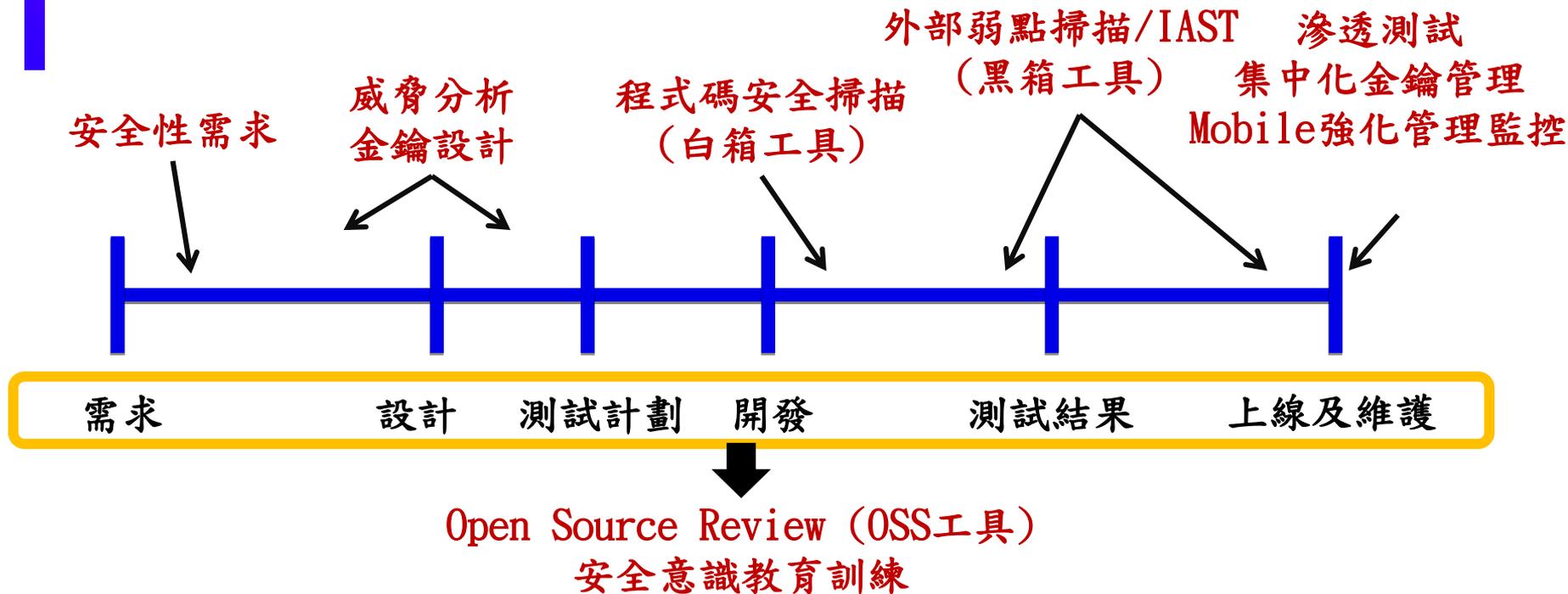
 M2: Insecure Data Storage

How/When
to fix?

 M9: Reverse Engineering

 M10: Extraneous Functionality

遵循軟體開發生命週期之作業流程

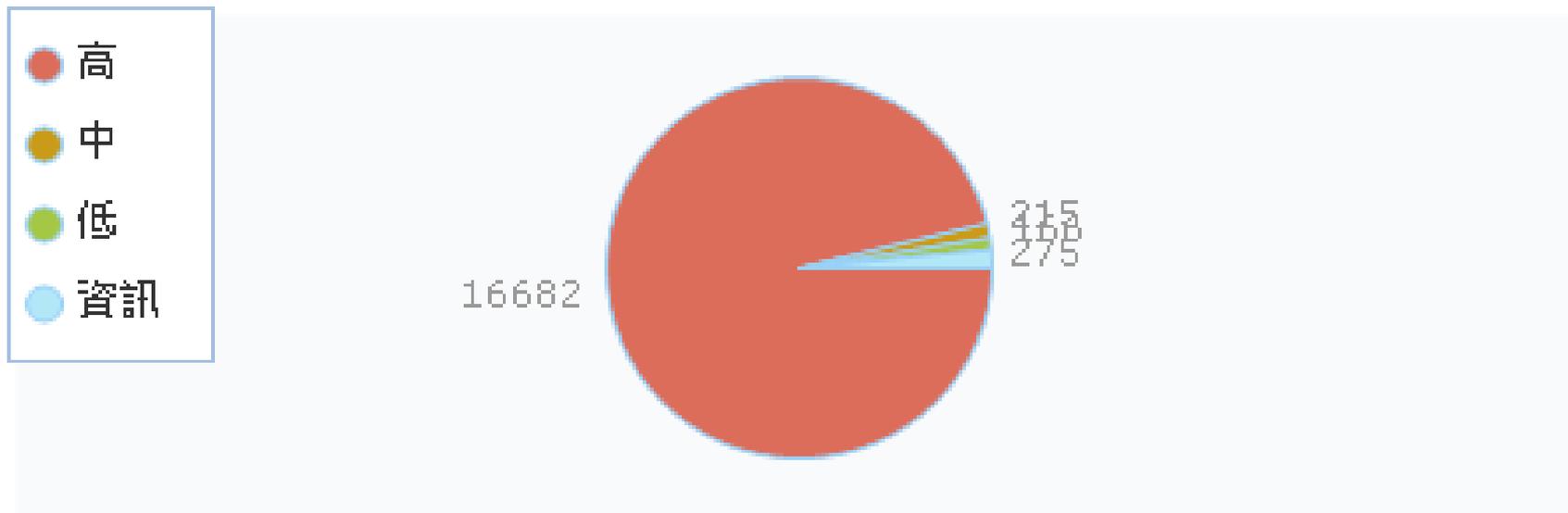


- 應針對應用系統進行整合性測試，包含原碼審查/弱點掃描，如發現弱點，並安排修復確保應用系統安全
- 針對應用系統所檢測之弱點應定期產生管理性報表，以進行分析追蹤

實際狀況

驗收前因為客戶要求所以進行安全掃描

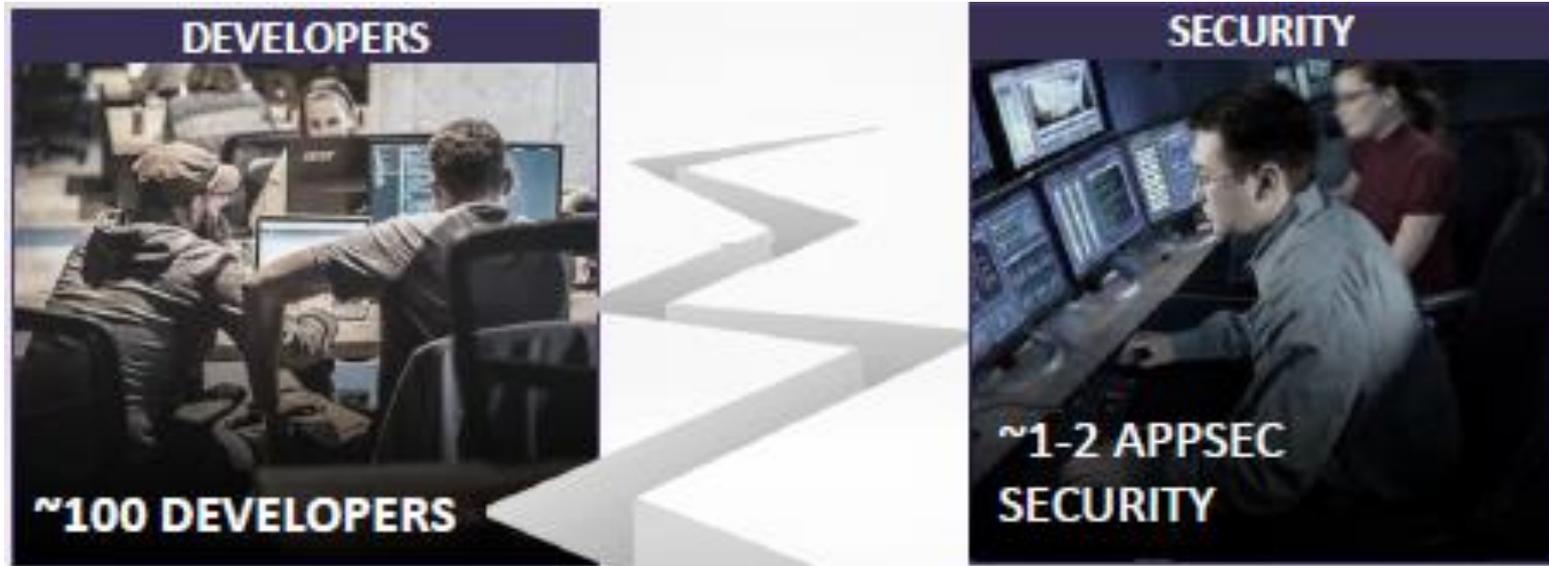
	問題總和	高	中	低	資訊
結果	17338	16682	215	166	275



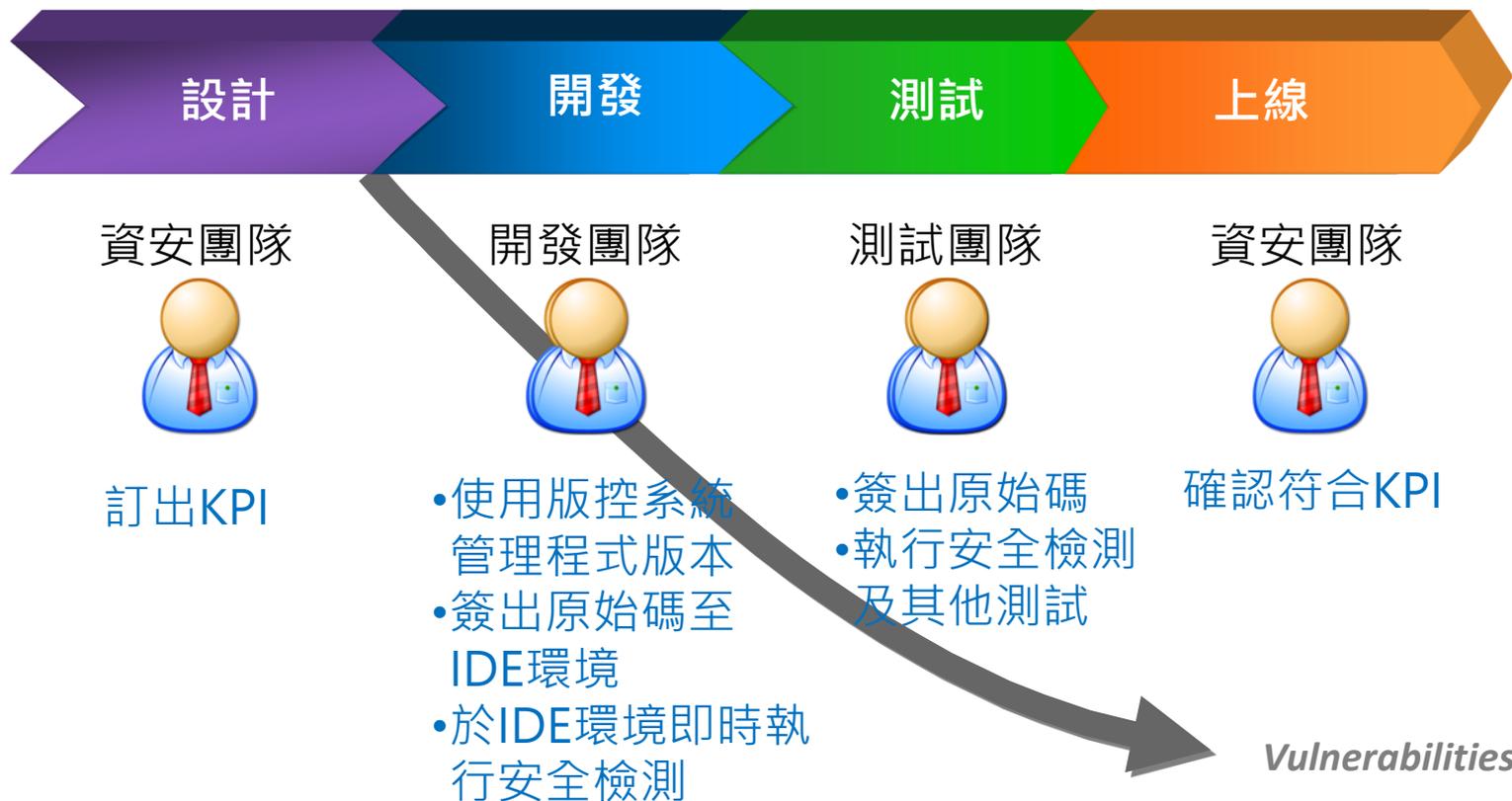
*第一次進行測試結果通常很可觀

資安官的挑戰

- 開發團弱缺乏安全知識和意識。
- 資安團隊缺乏開發經驗
- 傳統的開發課程注重功能開發
- AppSec團隊缺少有效的把關機制



遊戲規則制定



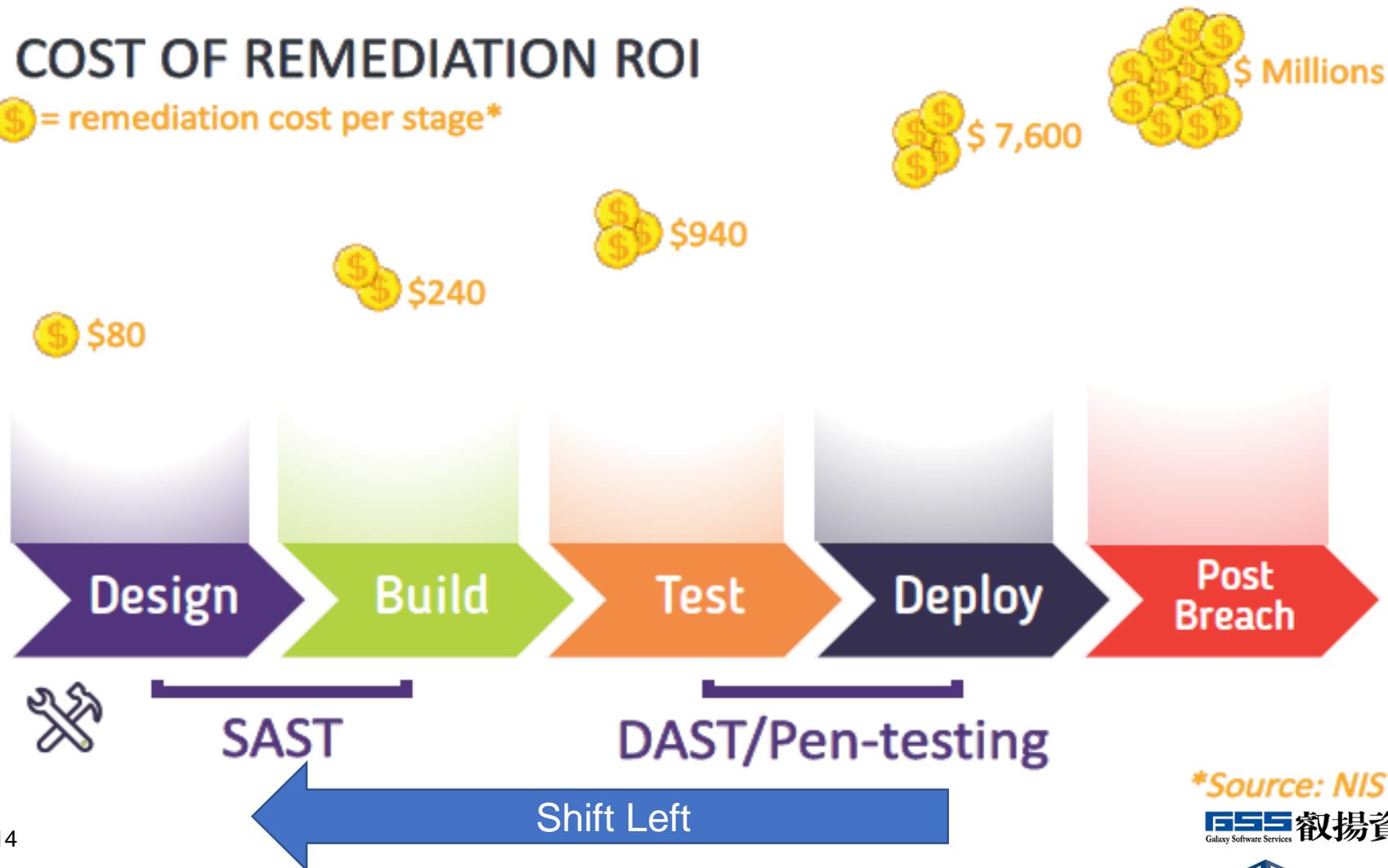
範例:源始碼檢測-Checkmarx 檢測要求

1. 不能有高風險OWASP TOP 10
2. 不能有 Injection HIGH issues

問題修復成本

COST OF REMEDIATION ROI

💰 = remediation cost per stage*



*Source: NIST

GSS 叢揚資訊
Galaxy Software Services

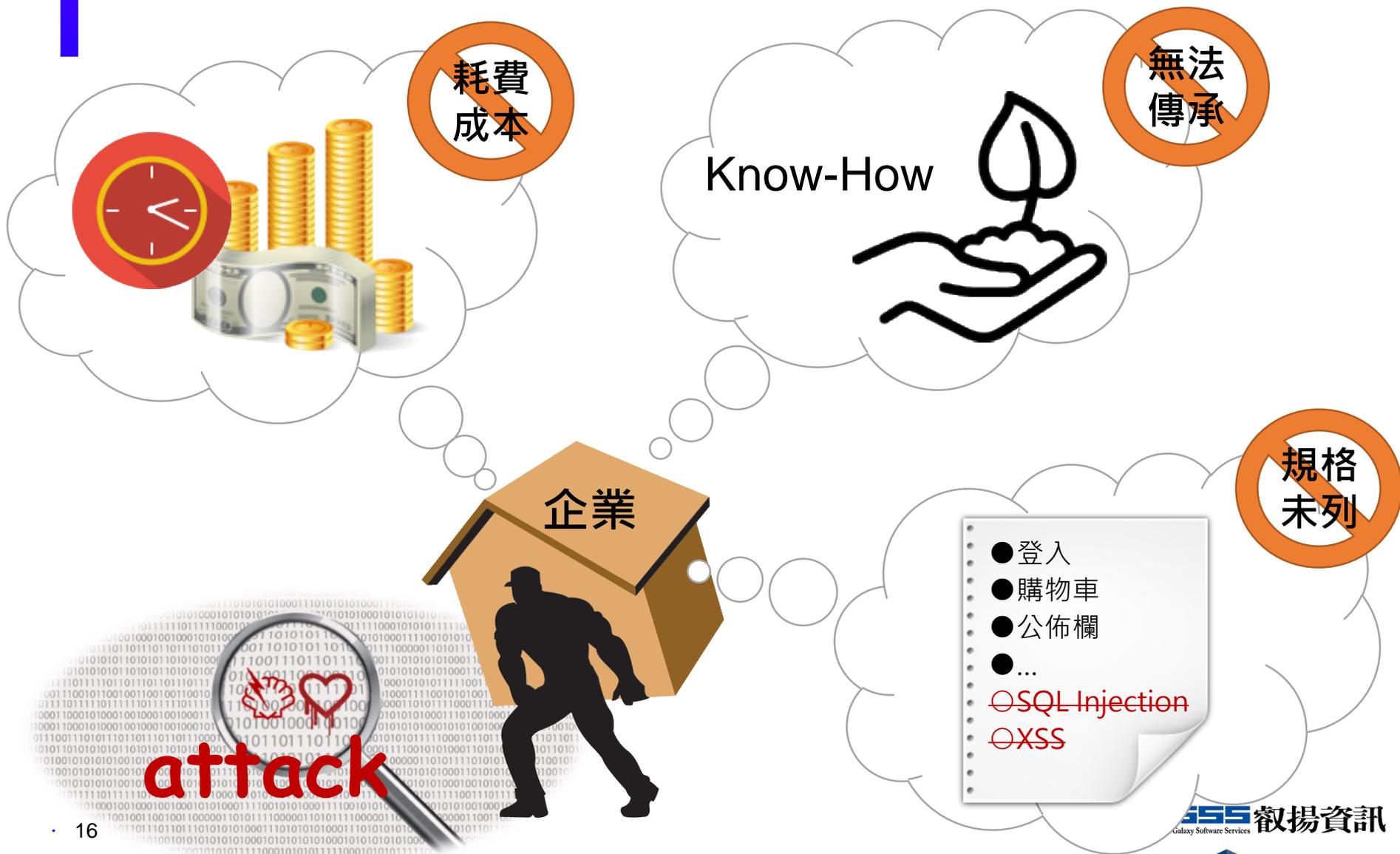
IT & Security

Chapter

2

開發人員資安意識提升

資安修復隱藏成本



了解漏洞、風險

WHAT

• 弱點風險

WHY

• 發生原因

HOW

• 如何解決

一般弱點解說

Reflected XSS All Clients

風險

可能發生什麼問題

攻擊者可能利用社交工程攻擊來導致使用者發送網站設計的輸入，重寫網頁並插入惡意腳本。然後，攻擊者可以偽裝成原來的網站，這將使攻擊者可以竊取使用者的密碼，要求使用者的信用卡資訊，提供偽造訊息，或執行惡意軟體。但從受害者的角度來看，這是原來的網站，受害人會責怪網站所產生的損害。

靜態文字，無法讓開發人員有深刻印象

原因

如何發生

從使用者輸入的資料。如果資料包含HTML片段，該漏洞主因為未先對

一般建議

如何避免

1. 驗證所有輸入，無論其來源為何。驗證應基於白名單：僅接受資料擬合一個指定的結構，而不是拒絕不良 patterns。應確認：
 - 資料類型
 - 大小
 - 範圍
 - 格式
 - 期望值
2. 在輸出嵌入之前完全編碼所有動態資料。
3. 編碼應該是上下文相關的。例如：
 - HTML內容使用HTML的編碼方式
 - HTML編碼特性是將資料輸出到特性的值
 - JavaScript的編碼方式為伺服器產生的Javascript
4. 考慮使用ESAPI的編碼庫，或它的內置功能。對於舊版的ASP.NET，請考慮使用AntiXSS。
5. 在HTTP類型對應的表頭，明確定義整個頁面的字元編碼。
6. 設置 httpOnly 標誌於會期資訊，以防止利用XSS來竊取資訊。

程式碼範例

CSharp

應用程式是使用「Referer」欄位字串來建立 HttpResponseMessage

```
public class ReflectedXssAllClients
{
    public static void foo(HttpRequest Request, HttpResponse Response)
    {
        string Referer = Request.QueryString["Referer"];
        Response.BinaryWrite(Referer);
    }
}
```

線上弱點學習平台

Learning from CodeBashing

Welcome to Codebashing!

Want to sharpen your secure coding skills and fix vulnerabilities quickly?

Select your programming language for interactive tutorials. Have Fun!



Java

Learn how to secure Java web applications.

Start



.NET

Learn how to secure .NET web applications.

Start



PHP

Learn how to secure PHP web applications.

Start

線上弱點學習平台

Learning from CodeBashing



SQL Injection

情境引導方式，直接感受受害者及攻擊者所會發生的事情

In this interactive tutorial you will understand how SQL injection attacks are used to compromise the security of a web application, and how to write code more securely to protect against this type of attack.

Let's Play



ALICE
Our hero



BOB
The bad guy

線上弱點學習平台

The screenshot displays a web application interface for a security learning platform. On the left, a sidebar lists course modules: 6. Authentication Logic, 7. Building the SQL Query, 8. Understanding Injection, and 9. Bypassing Authentication (highlighted). Below the sidebar, text explains SQL Injection and provides a challenge: "Try entering the following credentials: Username: alice@bank.com, Password: ' or 1=1)#". A note explains that the # character is used for code comments in MySQL. The main content area shows a browser window with the URL https://trade-portal.codebashing.com/sessions and a "TradePORTAL Login" form. The form has fields for "Username" (alice@bank.com) and "Password" (' or 1=1)#', and a "LOG ME IN" button. Below the form is a "CODE" editor showing a Java code snippet with a SQL injection payload highlighted in red. The code is as follows:

```
1 //String email = request.getParameter("email");
2 //String password = request.getParameter("password");
3
4 String sql = "select * from users where (email = " + alice@bank.com + " and password = "
5 + ' or 1=1)# + "'";
6 //Connection connection = pool.getConnection();
7 //Statement statement = connection.createStatement();
8 //ResultSet result = statement.executeQuery(sql);
9
10 //if (result.next()) {
11 //loggedIn = true;
12 // # Successfully logged in and redirect to user profile page
13 //} else {
14 // # Auth failure - Redirect to Login Page
15 //}
```

依步驟了解弱點發生原因及解決方式

互動式教學，加深學習印象

後台管理

4



2



2



Engagement Summary

From To Last Month

0

Total Likes

0

Total Badges

0

Total Content Contributions

Engagement summary view

Click and drag in the plot area to zoom in



— Daily lesson completions — Daily course completions
— Daily active users — Daily new users

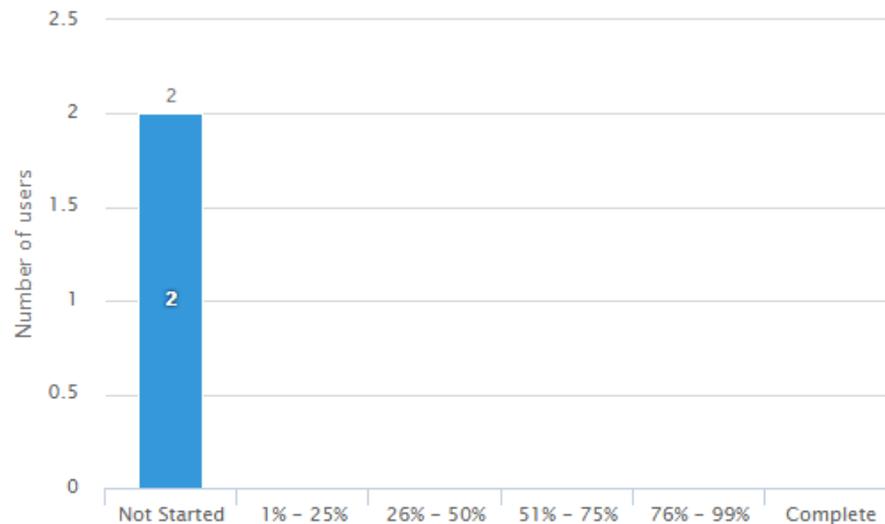
員工學習的狀況
(依時間、語言)

Course Progress

Courses by percentage (columns)

Percentages by course (bars)

Overall Users Completion Funnel



Java ASP.NET PHP Node.JS Ruby on Rails
Python Django Scala C/C++ Android iOS Go

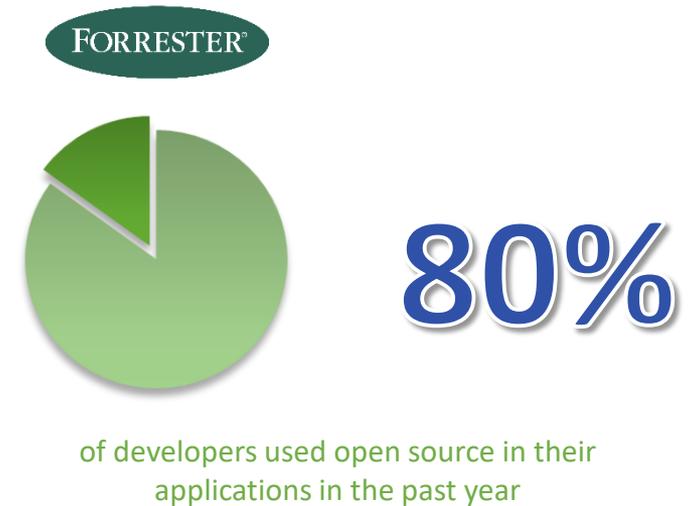
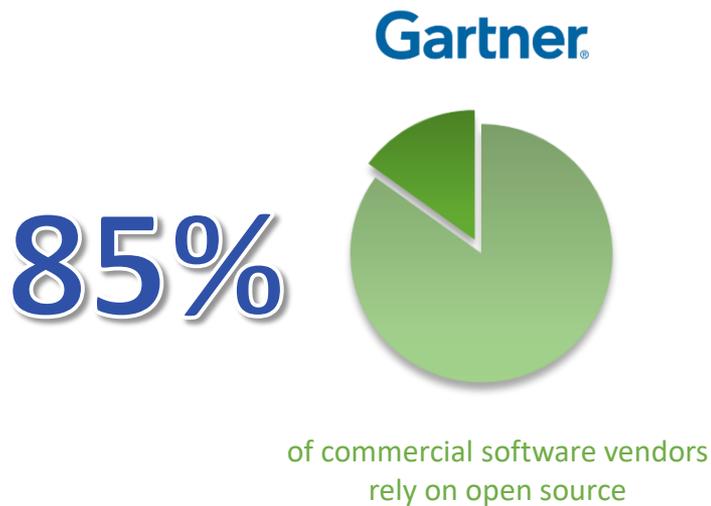
Chapter

3

第三方元件管控

OSS

OSS使用率增長



*Gartner User Survey Analysis: Open-Source Software, 2015

*Forrester developer survey 2015

GSS 叢揚資訊
Galaxy Software Services

IT & Security

潛在的資安問題

我們到底有使用多少OSS?

我所使用的OSS合規嗎？

我所使用的OSS安全嗎？



風險 - OWASP Top 10 2017

-  A1: Injection
-  A2: Broken Authentication
-  A3: Sensitive Data Exposure
-  A4: XML External Entites (XXE)
-  A5: Broken Access Control (As it was in 2004)
-  A6: Security Mis-configuration
-  A7: Cross-Site Scripting (XSS)
-  A8: Insecure Deserialization
-  **A9: Using Components with Known Vulnerabilities**
-  A10: Insufficient Logging & Monitoring

OSS風險

資安趨勢部落格 > 漏洞攻擊 > CVE-2017-5638 : Apache Struts 2 漏洞可能讓駭客從遠端執行程式

CVE-2017-5638 : Apache Struts 2 漏洞可能讓駭客從遠端執行程式

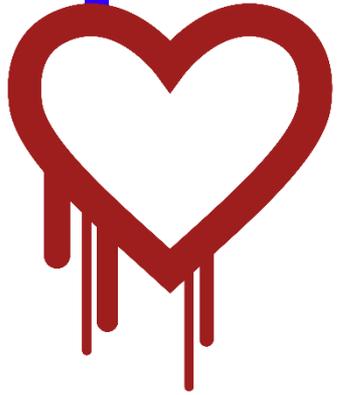
POSTED ON 2017 年 03 月 15 日 BY TREND LABS 趨勢科技全球技術支援與研發中心

Like 8 Share G+

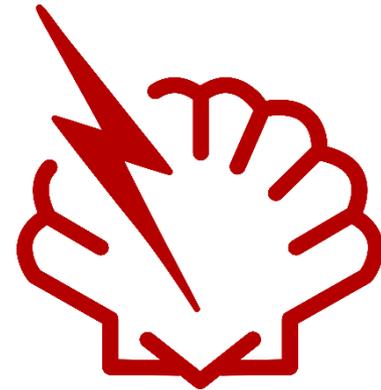
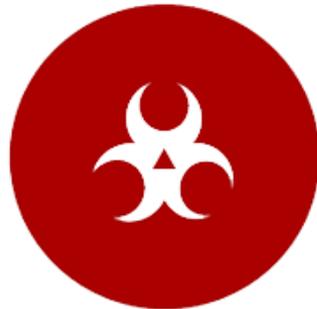
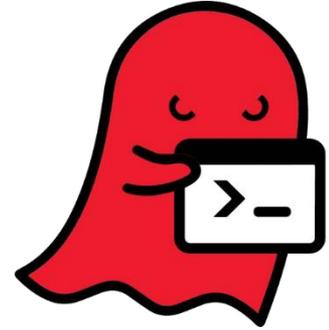


Apache Struts 是一個免費的開放原始碼程式開發架構，用來開發 Java 網站應用程式。我們仔細研究了過去 Apache Struts 被發現的幾個遠端程式碼執行 (Remote Code Execution，簡稱 RCE) 漏洞之後發現，歹徒大多使用 Object Graph Navigation Language (OGNL) 這個程式語言。OGNL 之所以很容易讓駭客從遠端執行任意的程式碼，是因為 Apache Struts 在大多數的流程當中都用到這個語言。

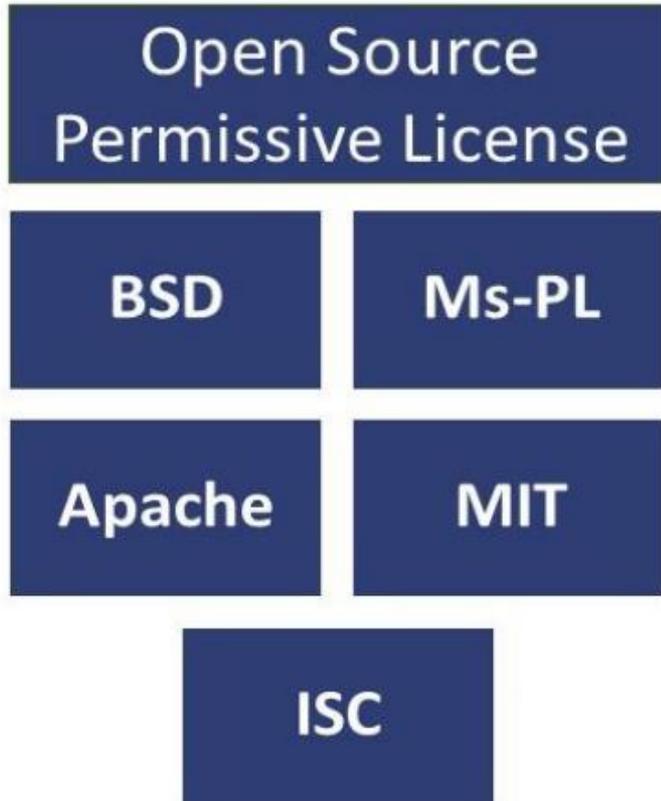
重大OSS議題



EQUIFAX



Permissive vs Copy Left



OSS應加強的管理

Use WhiteSource Get Better

雙重使用方式



發現：結合CI工具，早期發現風險

Know
Where You
Stand



報告：多面向資訊，全面了解現況



選擇：列出多項版本，選擇最適合您的



警示：依公司政策，建立預警機制

Continuous
Management

選擇適合的OSS

Use WhiteSource Get Better

The screenshot shows the Maven Repository page for the artifact `commons-beanutils/commons-beanutils/1.8.0`. The page includes a search bar, a graph of indexed artifacts (4.25M), and a list of popular categories. The main content area displays the artifact details for **Apache Commons BeanUtils » 1.8.0**, including its license (Apache 2.0), categories (Reflection Libraries), homepage, date (Aug 28, 2008), and files (Download (JAR) (288 KB)). A note indicates a new version (1.9.3) is available. The artifact is used by 1,710 other artifacts. A code block shows the Maven dependency declaration for the artifact. Below the code block, there are advertisements for Microsoft Office products, including Office 365 家用版 (TW\$3,190), Office 365 個人版 (TW\$219), and Mac 版 Word 2016 (TW\$3,990). The bottom of the page shows a table for compile dependencies.

Overlaid on the right side of the screenshot is a WhiteSource analysis panel for **COMMONS-BEANUTILS**. The panel includes a **Settings** button and the following information:

- IDENTIFIERS**: Group: commons-beanutils | Artifact: commons-beanutils | Version: 1.8.0
- ALREADY IN USE**: (0) None
- LICENSES**: Apache 2.0
- SECURITY VULNERABILITIES**: (1) CVE-2014-0114 Apache Commons BeanUtils,... **HIGH**
- QUALITY**: ★★★★★ FAIR
- POLICIES**: No Policy

Category/License	Group / Artifact	Version	Updates

OSS SDLC 觀點

Use WhiteSource Get Better



Chapter

4

傳統應用系統 VS 行動化應用系統

破解後影響

駭客想要做的事情

數位資產
Treasure Trove
&
Monetary Gains

- **Piracy and unauthorized distribution**
- **IP theft** (如.重要演算法) 透過反組譯逆向工程
- **Sensitive information** (如. 帳號,密碼,keys,憑證)
- **Bypass security controls** (如. 授權, 加密破解, licensing, DRM, root/jailbreak detection, ads)
- **Insertion of malware** or exploits in the application and repackaging

保護方式

Guards, or software protection routines, appear to be normal code and:

- Enable the program to **DEFEND** itself,
- To **DETECT** if it is attacked,
- To **ALERT** and **REACT** if it is modified
- Are policy and threat driven

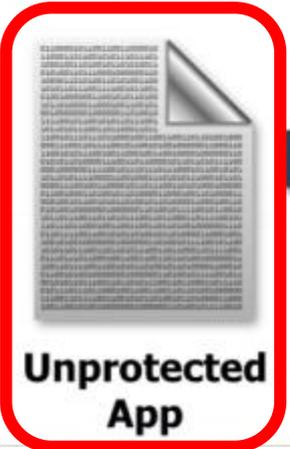
② 加上撰寫的GuardSpec

- Code Obfuscation
- Data Obfuscation
- Symbol Stripping
- Symbol Renaming
- String Encryption

- Damage
- Anti-Debug
- Self-Repair
- Checksum
- Resource Verification
- Root Detection
- Swizzling Detection
- Jailbreak Detection



① 保護前的檔案

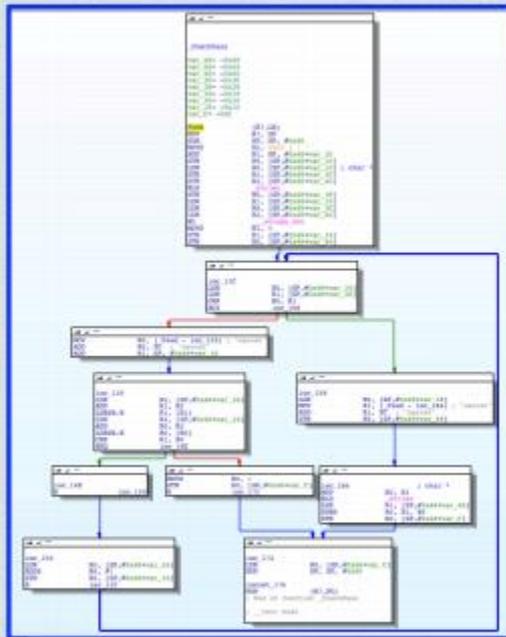


③ 加上Arxan Lib

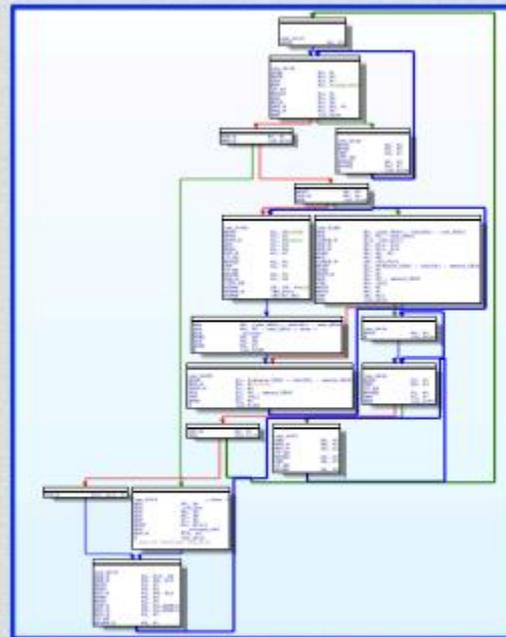


Arxan 可彈性設計混淆程度

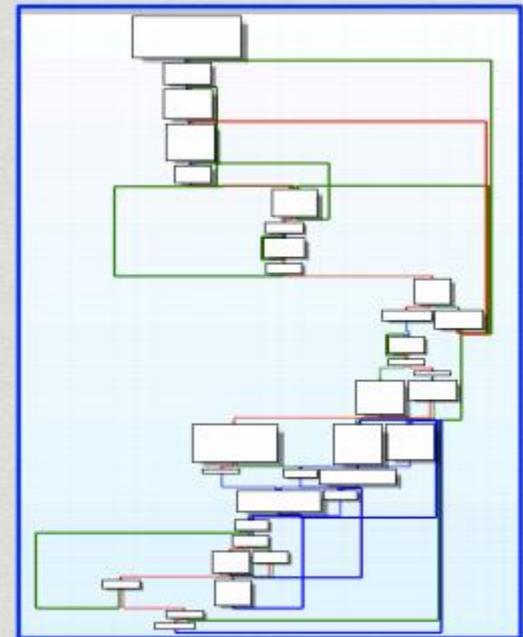
Usage: `gs.obfuscate(protected_range, percentage_growth_amount)`



1X



5X



10X

主要Guard介紹-Obfuscation

- Increases code complexity
- Protects against static analysis
- Obfuscating transformations

Before

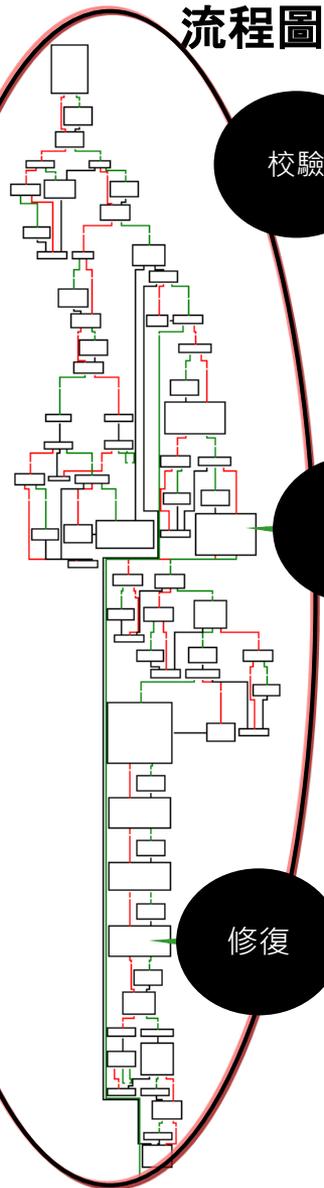
```
.text:0045625F      public start
.text:0045625F start  proc near
* .text:0045625F    push    60h
* .text:00456261    push    offset stru_4750D0 ; lpModuleName
* .text:00456266    call   __SEH_prolog
* .text:0045626B    mov     edi, 94h
* .text:00456270    mov     eax, edi
* .text:00456272    call   __alloca_probe
* .text:00456277    mov     [ebp-18h], esp
* .text:0045627A    mov     esi, esp
* .text:0045627C    mov     [esi], edi
* .text:0045627E    push   esi ; lpVersionInformation
* .text:0045627F    call   ds:GetVersionExA ; Get extended information about the
```

After

```
.text:0045625F      public start
.text:0045625F start  proc near
* .text:0045625F    jmp     near ptr 4ACBF0h
* .text:0045625F    start
* .text:0045625F    endp
* .text:0045625F ; -----
* .text:00456264    dd     134D6CE9h, 0E982EB00h, 2CB5DCh, 5E981EBh, 0EB001FADh
* .text:00456264    dd     8FEBFA8Dh, 0E95697EBh, 2E923Eh, 3A93EB89h, 207092E9h
* .text:00456264    dd     8FEBE900h, 8926E9E9h, 8CEB0014h, 0AD68E900h, 0EB8D000Fh
* .text:00456264    dd     9BE97988h, 440008C8h, 2E1E4BE9h, 93EB6200h, 0E9A3EB00h
* .text:00456264    dd     26F310h, 325DE9CCh, 0EB4C0007h, 89EBE9A3h, 0E0C76E9h
* .text:00456264    dd     5684EB00h, 2D43CCE9h, 95EBB700h, 20AEC3E9h, 8B1F7500h
* .text:00456264    dd     0C8033C48h, 4F791E9h, 12755200h, 1841B70Fh, 109A69E9h
* .text:00456264    dd     0E91F7400h, 15E85Dh, 75890574h, 0E927EBE4h, 1C4581h
```

獨家Guards網路式防護

流程圖



Guard網路保護機制

利用**交叉防護**來全面確保各種功能的

Guards

可監測APP受到攻擊時,即時相關聯的Guards網路**同步啟動保護**

攻擊者需同時破解所有**監控位置**
由於Guards相互保護之下,攻擊者**不易找到有關Guards的痕跡**

Guard可調整設定保護**特定範圍程式碼,或整個應用程式**

和Guard網路能有效**防止**用各種武碼
保護前層Guard
在執行階段**動態分析**後通過**反組譯**
被解

第三層自我修復的Guard來保護前兩層Guard

通過校驗來保護整個應用程式是否被修改

混淆

通過校驗和保護關鍵程式碼

校驗

修復

含有自我修復功能保護重要部份的程式碼

未受保護Key外漏案例

```
137 public static void init(String paramString)
138     throws Exception
139 {
140     try
141     {
142         cipher = Cipher.getInstance("AES/CBC/PKCS7Padding", "BC");
143         keySpec = new SecretKeySpec(paramString.getBytes("UTF8"), "AES");
144         return;
145     }
146     catch (Exception localException)
147     {
148         Debug.WriteLine("encrypt failed", localException);
149         throw localException;
150     }
151 }
152
153 public static void main(String[] paramArrayOfString)
154 {
155     try
156     {
157         encrypt(String.valueOf(new Date().getTime()), "██████████@86136982");
158         return;
159     }
160     catch (Exception localException) {}
161 }
162 }
```

Whitebox Cryptography?

Use TransformIT Ensure Security

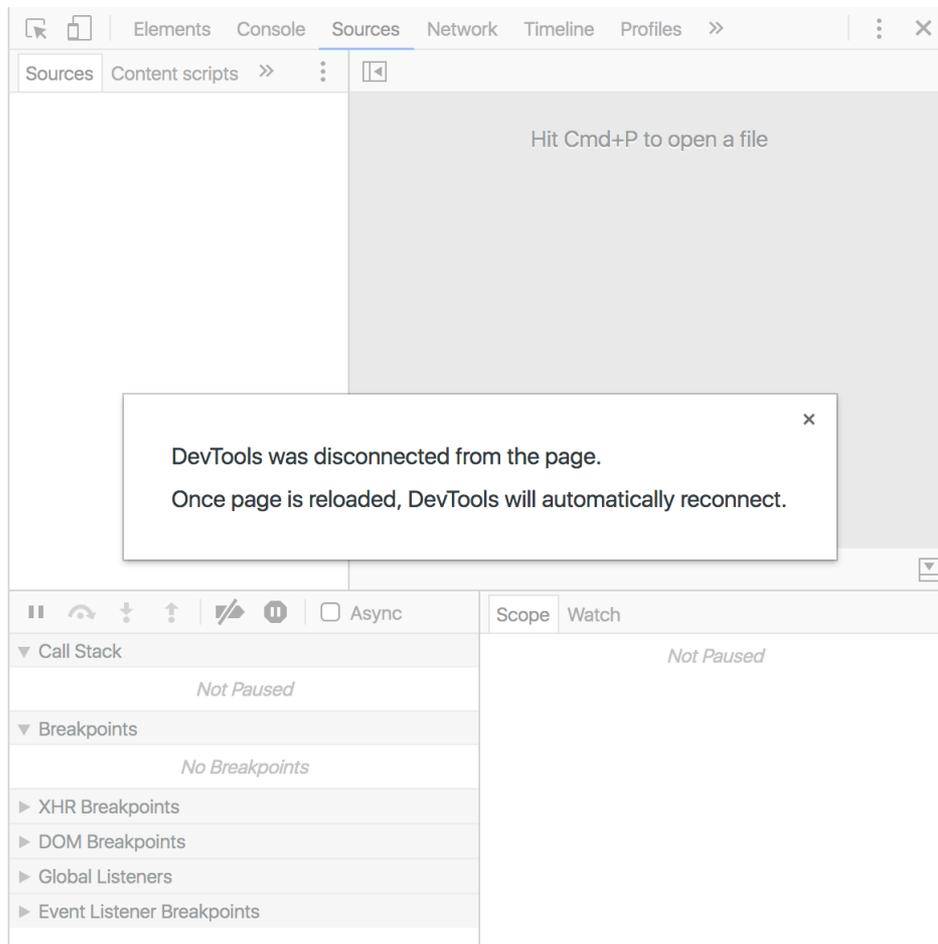
- 白箱加密(White-box Cryptography)是一種用於混淆加密演算法的續密方法，使得金鑰可充分隱藏、防止窺探，其目的在於防止加密運算中的關鍵資訊(比如私鑰)洩露，以免潛在攻擊者完全存取系統。
- 白箱定義在於因App是在行動裝置端執行，可透過反組譯察看到內容，攻擊者可進行記憶體偵查分析出私鑰內容，資訊都可透明看到，因此稱之為白箱。

使用Hybrid開發背景

- 減少開發時間以及學習雙平台開發成本
- 使用Javascript開發模式+HTML5方式
- Web/Hybrid應用
 - 只要使用一種方式就可以雙平台(iOS/Android)開發
 - 不需要多花時間學習Object-C/Swift,Android
 - 學習門檻低
- Web應用(RWD Responsive Web Design響應式網頁)
 - 直接透過行動裝置瀏覽器執行
 - 依照設備螢幕大小自動調整畫面
 - 不需要上架App store/Google Play

保護後的結果確認-動態攻擊

Debug偵測啟動,防止動態分析



← 無法進行Debug

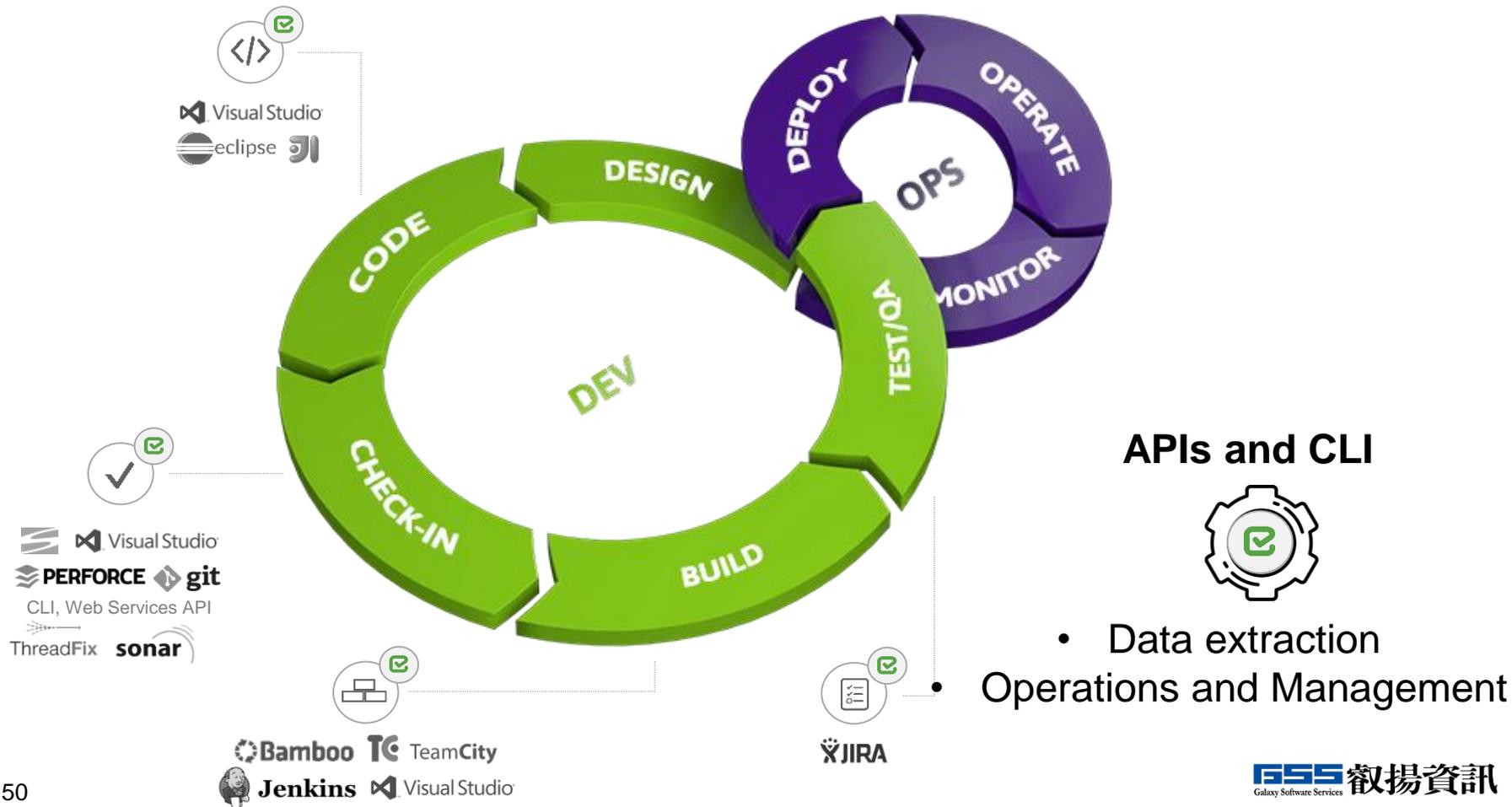
Chapter

5

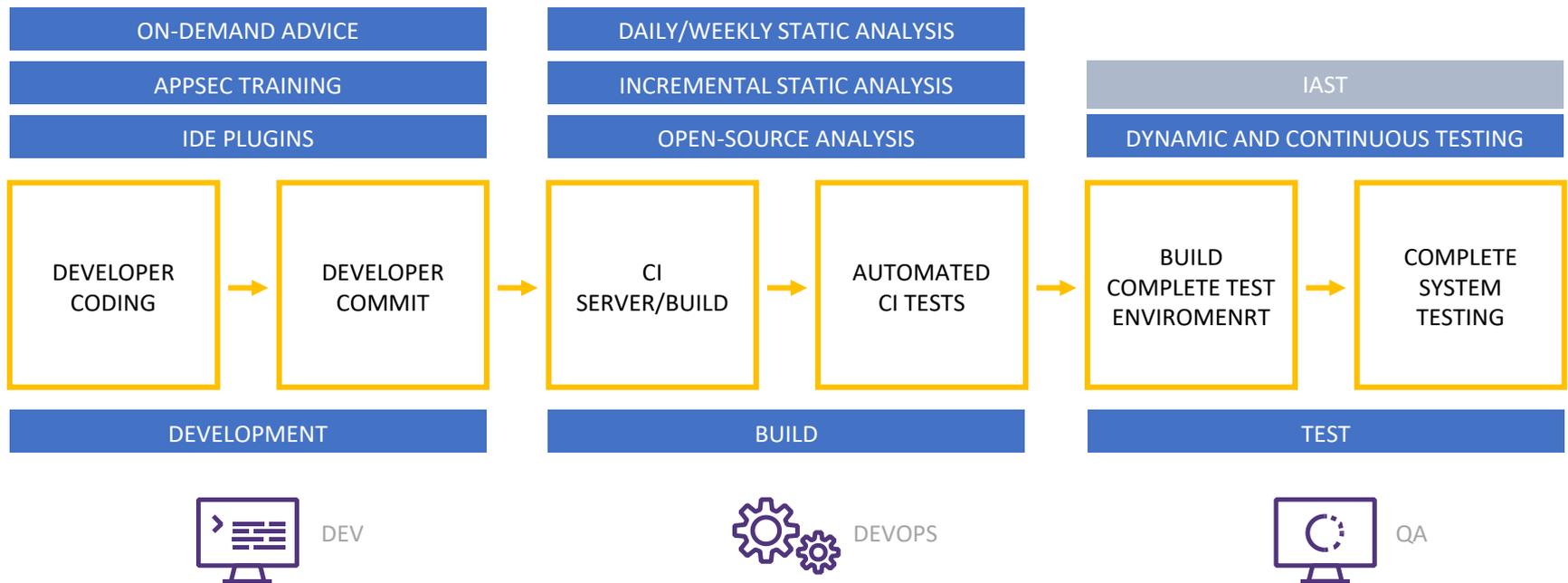
持續整合與部署平台

Secure SDLC Integration and Automation

(CI/CD, DevSecOps)



SEAMLESSLY EMBED APPSEC TESTING INTO DEVOPS PROCESSES



GSSDLC for Application & APP



自動上版平台



需求變更
系統



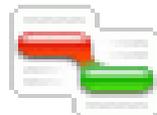
Jenkins自動整合平台



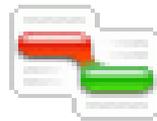
MS Build



Maven™



GSS-CIA
Compare



衝擊分析

黑/白箱檢測

開源碼檢測

發布前後
自動程序



AppScan

IBM Security



WhiteSource



ARXAN



APPERIAN
MOBILE APP MANAGEMENT



需求變更
介接

差異分析

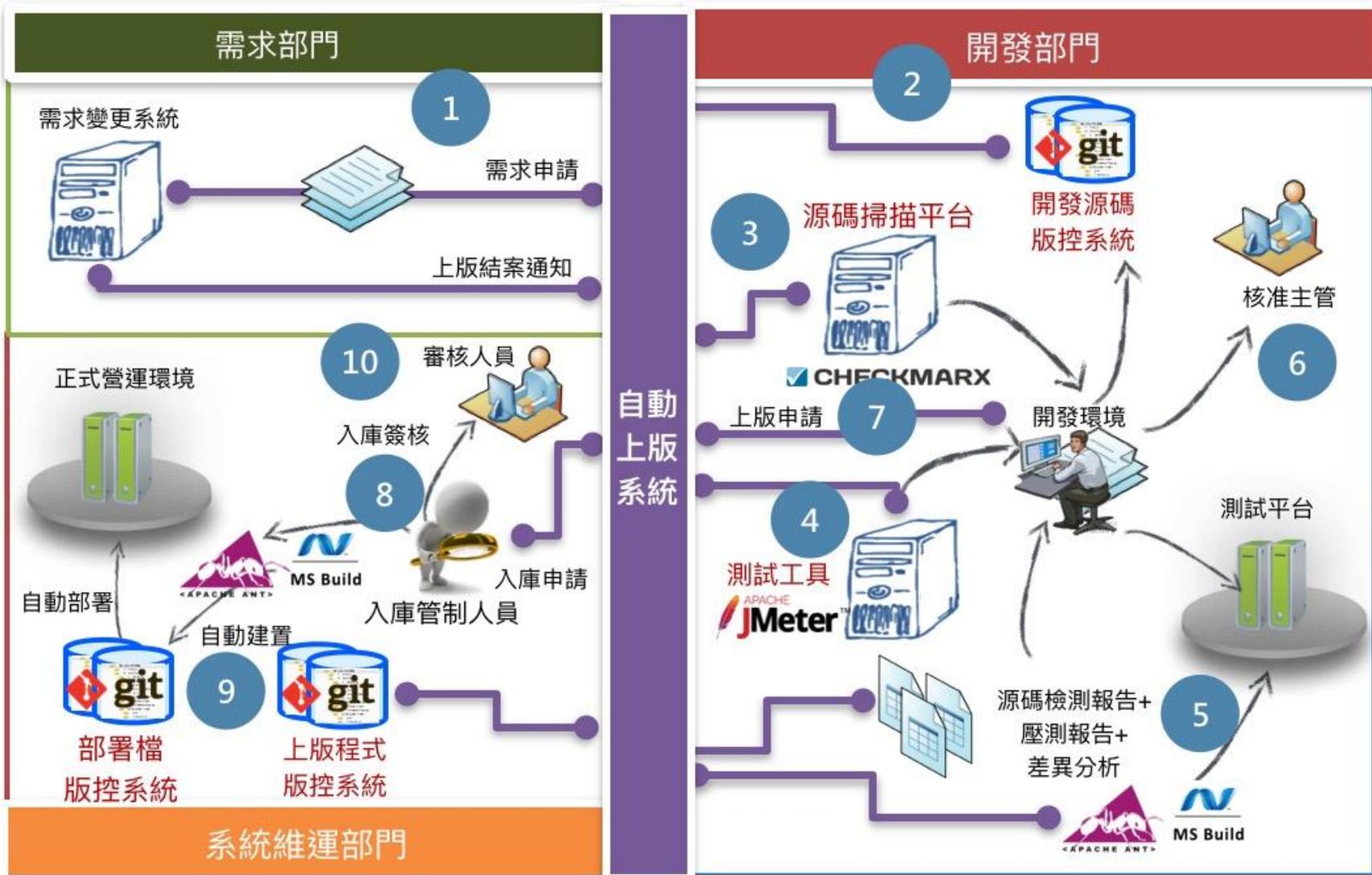
源碼檢測

自動測試

自動建置

自動部署

GSSDLC 案例



Chapter

6

結論

結論

- 對於企業
 - 可整合**企業資安規範**，依需求客製化檢測規則
 - 提升應用系統之**安全強度**
 - 整合現有 **SDLC** 早期發現應用系統弱點，**降低修復成本**
 - 結合自動化程式碼安全檢測機制，能大量**節省人力**成本
 - 提供**定期的更新**機制，符合安全現況趨勢
 - 設計獨一無二的APP防護，**強化APP安全**
- 對於使用者(開發者)
 - 互動式的學習平台，**增加開發人員資安意識**
 - 安全程式開發指引，**減少重工情境**
 - 提供圖形化檢測結果，方便進行修復程式碼弱點
 - 明確指出第三方元件適用與否便利選擇

請填寫問卷 會後抽出問卷禮

GSS 叢揚資訊
Galaxy Software Services

Thank You

感謝您的聆聽，敬請指教



國家產業創新獎
卓越中堅企業



GSS 叢揚資訊



Vital 雲端服務家族



GSS 技術部落格