Communications of the CCISA
Vol.30 No. 2 May. 2024

JAVA 網頁系統之反序列化弱點偵測技術

江竑緯[1]、周兆龍[2*]
[1]國防大學理工學院資訊工程學系、[2]逢甲大學資訊工程學系
[1]jw97981136@gmail.com、[2]chaolung.chou@gmail.com

摘要

隨著科技不斷的進步，網頁系統已越來越普及，而網頁系統的功能也越來越複雜，進而衍生出更多的漏洞造成了安全威脅。由於開發網頁系統會使用大量的函式庫進行撰寫，而部分有缺陷的函數或是可被利用的函數堆疊後將導致系統產生漏洞，於 OWASP TOP 10 : 2021 中，不安全的反序列化(Insecure Deserialization)被列為 A8，在現行的黑箱測試中不見得可以發現反序列化的漏洞，是需要透過人工的方式進行審查，才可避免漏洞的產生。

白箱測試是網頁滲透測試的主要方法之一，透過滲透測試員對目標網頁進行原始碼檢測發現弱點，開發攻擊程式碼對其網頁進行滲透測試攻擊進而修補網頁弱點。本研究設計 Java 網頁系統建置反序列化漏洞，並運用知名弱點掃描工具進行黑箱測試，經測試後並無發現可利用弱點，再透過白箱測試檢測原始碼發現網頁弱點，開發攻擊程式碼展示反序列化攻擊方法，進而獲得網頁系統管理者權限，藉此方法顯示白箱測試的重要性。

關鍵詞：白箱測試、網頁滲透測試、Java 網頁系統、反序列化

* 通訊作者 (Corresponding author.)

Communications of the CCISA
Vol.30 No. 2 May. 2024

JAVA Web System Deserialization Vulnerability Detection Technology

Hung-Wei Chiang[1], Chao-Lung Chou [2*]
[1,] Department of Computer Science and Information Engineering, Chung Cheng Institute of Technology, National Defense University, [2] Department of Information Engineering and Computer Science, Feng Chia University
[1] jw97981136@gmail.com 、[2] chaolung.chou@gmail.com

Abstract

With the continuous advancement of technology, web systems have become increasingly prevalent, and the functionalities of web systems have also become more complex, leading to the emergence of more vulnerabilities and security threats. Since the development of web systems involves the use of numerous libraries for coding, the existence of defective or exploitable functions in these libraries can result in system vulnerabilities. In OWASP TOP 10: 2021, insecure deserialization is classified as A8. Current black-box testing, may not necessarily uncover deserialization vulnerabilities, requiring a manual review to prevent the occurrence of such vulnerabilities. White-box testing is one of the primary methods in web penetration testing. Through source code analysis, penetration testers identify weaknesses in the target web application. They then develop attack code to perform penetration testing attacks on the web application, subsequently patching any identified vulnerabilities.

Based on this issue, this study designs a Java web system to build a deserialization vulnerability and utilizes well-known vulnerability scanning tools for black-box testing. After testing, no exploitable vulnerabilities were discovered. Subsequently, white-box testing is conducted to examine the source code, identifying web vulnerabilities. Attack code is developed to demonstrate deserialization attack methods, thereby gaining administrator privileges in the web system. Through this approach, the significance of white-box testing is highlighted.

Keywords: White-Box Testing, Web Penetration Testing, Java Web System, Deserialization

壹、前言

1.1 研究背景與動機

科技隨著時間不斷的進步，而網路安全這個議題越來越被各國企業及政府組織重視，但駭客組織的攻擊仍然無時無刻的在全世界發生，甚至間接影響了國家、國防、關鍵基礎設施。但大多數人對安全知識的認知並不是很了解，需要普及安全知識這是一件非常困難的事。

2022 年 8 月 2 日至 3 日美國聯邦眾議院議長裴洛西(Nancy Pelosi)率領美國眾議院國會訪問團來台進行訪問[2]，並共同召開國際記者會，而中共高層對此次事件極度不滿，不僅對台進行了軍事威脅，並試射飛彈飛越台灣本島上空試圖引發人民恐慌，更針對台灣各政府部門網站進行了網路攻擊，而這場攻擊整整持續了 9 天，對台灣各政府機關的網站進行了分散式阻斷服務攻擊(Distributed Denial-of-Service, DDoS)。網路攻擊事件不僅僅只在政府機關，我們的日常生活也受到了波及，以臺鐵電子看板如圖一及 7-11 數位看板如圖二為例，這兩者都遭受了內容置換(Deface)攻擊，散布謠言和錯誤的假訊息，進而引發人民恐慌。此事件凸顯了資訊安全防護的重要性，資訊安全不僅關係到國家機密和政府機關運作，也直接影響了人民的生活，企業和政府組織應加強資訊安全防護，投入更多資源來應對不斷升級的網路威脅。同時，國際社會也應共同努力，制定更加嚴格的資訊安全標準，共同維護全球資訊安全。

圖一：臺鐵電子看板遭攻擊示意圖[1]

圖二：7-11 數位看板遭攻擊示意圖[2]

根據 Fortinet 所屬的 FortiGuard Labs 威脅情報中心在他們最新發佈的《2023 上半年全球資安威脅報告》中指出[3]，新型勒索軟體組織的出現帶來了明顯的影響。勒索軟體威脅在 2023 年上半年呈現驚人的增長，比起前一年同期增加了超過八成。報告還強調了攻擊的頻率，每秒近 1.5 萬次攻擊，其中臺灣在亞太地區遭受攻擊的頻率處於領先地位，且該報告更清楚的顯示了攻擊事件已經時時刻刻都在發生的事實。

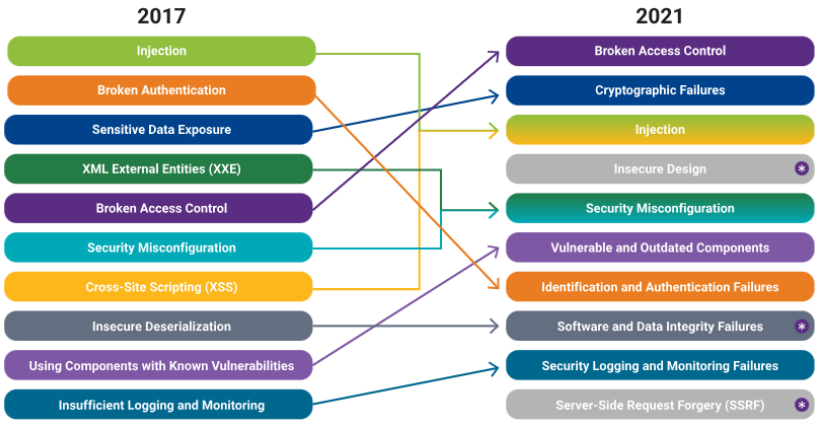
面對日益增加的網路攻擊，政府已在資通安全管理法裡規定各單位每年至少需進行兩次核心系統的弱點掃描與一次滲透測試，目的是降低資安事件的頻率。儘管各機關裝設了如入侵檢測系統(Intrusion Detection System, IDS)、入侵防禦系統(Intrusion Prevention System, IPS)、防火牆等多種安全設備，這些設備的存在可能讓 IT 人員認為系統已經相當安全。然而，這種思維是非常危險的，因為實際進行滲透測試時，仍可能發現一些重大的漏洞。

滲透測試是評估系統安全的一種方法，包括黑箱測試與白箱測試兩種方式。通常在黑箱測試中使用的弱點掃描工具在當前的防護環境下容易產生高誤報率。部署防護設備不僅提高了這種誤報率，也阻止了許多探測行為，導致測試結果可能不夠全面。為了解決這些問題，我們應該更加注重白箱測試。白箱測試不僅依靠滲透測試員的技術，還包括對系統內部結構的深入瞭解，模擬攻擊者的行為，透過白箱測試可以更全面的發現系統的潛在漏洞，提供更精準的安全評估，可以更徹底的了解系統的安全狀況，從而建立更堅固的資安防護，以應對持續變化的網路威脅。

1.2 研究目的

Java 是一個廣泛應用的程式語言，主要應用於網頁系統開發及程式開發，提供了各種技術和框架以支持構建可擴展的網頁應用元件。隨著資訊快速的發展，相應的也帶來了風險，其中不安全的反序列化問題在近年來不斷增加，並且已被 OWASP(The Open Web Application Security Project)列為十大安全風險之一如圖三[4]，這是一個值得我們特別關注的。滲透測試中常用的黑箱測試方法，雖然使用弱點掃描工具可快速地發現問題所在，但有時候仍舊無法識別出漏洞。本研究將通過白箱測試，即滲透測試員可直接針對原始碼進行審查，從而彌補黑箱測試的不足並識別出反序列化漏洞。通過這種方法，我們能更有效地找到並解決問題，減少被入侵的風險，並透過實驗設計來進行驗證。

而本研究僅針對 OWASP TOP 10:2017 中的不安全的反序列化(Insecure Deserialization)，該項次被併列於 OWASP TOP 10:2021 中 A08：軟體及資料完整性失效(Software and Data Integrity Failures)。本研究專注於探討反序列化的漏洞攻擊，會使用黑箱測試方法，運用弱點掃描工具尋找漏洞，但無法發現不安全的反序列化漏洞。進而通過白箱測試檢查原始碼並確認漏洞的存在，最後開發攻擊腳本來進行攻擊，以取得系統權限。



圖三：OWASP TOP 10 2017 與 2021 差異比較[4]

1.3 論文架構

本論文共分五節：第一節前言，說明研究背景、動機及目的。第二節文獻探討；第三節介紹本研究所提出的方法；第四節實驗結果，驗證本文所提出方法之有效性；第五章為結論。

貳、相關背景與文獻回顧

2.1 Java 網頁系統安全

Java 是一種被廣泛應用的程式語言，特別是在網頁系統開發領域扮演著關鍵角色。由於其能在不同平台上運行、擁有成熟的生態系、豐富的元件及框架支持，以及穩定的運行效能，Java 網頁系統贏得了許多開發者的喜愛。然而，像其他技術平台一樣，它在安全性方面有其優勢也有其不足。

對資安工作人員而言，網站的安全性是一個極其重要的考量點。網站的風險包括了資料保護、認證安全、身份保障、存取控制和輸入輸出安全等多方面。鑑於網站可能潛藏未知的安全漏洞，且駭客的攻擊手法和頻率在不斷進化，我們必須深入了解這些攻擊手法和模式，以便在建立網站時，就能針對已知的漏洞進行預防。

Java 網頁系統因其強大的安全措施而受到多數人的選擇，這包括了使用先進的加密技術、安全通訊協定、身份驗證和授權機制以及資料儲存的安全性。此外，Java 提供了多種安全框架，這些框架能夠有效應對身份驗證、授權和攻擊防護等安全需求，從而減少了安全漏洞的風險。儘管有了這些驗證及防護，漏洞仍會出現，但 Java 會定期更新以迅速修補這些漏洞。

然而，Java 網頁系統的安全也有缺點。由於其應用的複雜性，有時可能會因配置錯誤而出現安全漏洞。這些應用程式複雜的依賴關係和龐大的後端架構也使得管理變得更加困難。此外，Java 常見的反序列化操作，若處理不當，可能會讓攻擊者利用序列化資料執行遠端程式碼攻擊，導致嚴重的後果。

2.2 OWASP TOP 10 : 2021

OWASP 是一家總部設在美國的全球性非營利組織。OWASP 基金會旨在提升網路應用程式的安全性，支援全球的相關活動，並已在全球建立了超過 250 個地方分會，吸引了數萬名對資訊安全充滿熱情的會員。OWASP Top 10 是由該組織提出並定期更新的資安領域的重要資源，該文件彙整了網路應用程式的十大安全風險，並被廣泛用作資安專家和開發人員的參考指南。這份文件的目的不只是列出風險，還強調了這些風險的嚴重性，以便提醒相關人員應優先處理哪些問題。

雖然 OWASP Top 10 不是正式的標準文件，但它代表了資安界的共識，用於評估網路應用程式的安全性。該清單對企業和資安專業人員特別有價值，因為它幫助識別應用程式中的風險並提高對安全問題的警覺。此外，這份清單也常用於評估滲透測試報告，對漏洞獎勵平臺和企業安全團隊亦有參考價值。

2.3 滲透測試

滲透測試(Penetration Testing)，是一種模擬駭客行為的安全評估方式。這種測試針對企業的軟硬體系統進行攻擊模擬，以檢測是否存在各類型的漏洞。其目的在於識別並修補這些漏洞，強化系統對抗真實駭客的能力。企業與測試人員會簽訂合約明確測試範圍，以合法化進行測試過程。對於企業而言，進行滲透測試的益處眾多，不僅可以評估其防禦系統是否足以抵抗攻擊，還能及早發現並修補安全漏洞，從而減少被駭客入侵的風險。此外，滲透測試也區分為白箱測試及黑箱測試。

白箱測試與黑箱測試都有其獨特的優勢和局限性於表一中分析了兩者的優缺點。白箱測試能夠直接檢視程式碼，使測試人員能夠避免黑箱測試的誤判。這種方法不僅能夠精確指出需要修復的地方，還有助於增強應用程式的安全性。黑箱測試的優點是成本低，可以透過自動化工具模擬外部攻擊來識別安全風險，即使無法直接訪問到程式碼，這種測試方式能夠快速發現多種潛在問題，但運用自動化工具容易造成誤判。

白箱測試和黑箱測試各適用於不同的測試環境，可同時進行，關鍵是根據具體需求和情況選擇適當的方法，以確保網路應用程式的安全。滲透測試員會結合使用這些方法來提高應用程式的防護能力，確保它們能在各種威脅中保持安全。

表一：黑箱測試與白箱測試優缺點比較

項次	優點	缺點
黑箱測試	1.運用工具進行測試，效率比白箱測試高。 2.可模擬駭客發現網頁問題。 3.運用網頁掃描工具，可對各類網頁漏洞(OWASP Top 10)進行快速檢測。 4.不須深入了解網頁原始碼的各種撰寫型態。	1.原始碼無法檢測，因為黑箱測試不會檢測網頁原始碼的內容。 2.無法窮舉 Web 中所有可能的邏輯路徑。 3.自動化測試的結果有可能破壞 Web 程式中的運作。
白箱測試	1.檢查網頁原始碼運作情形，且可透過人工發現黑箱測試從外部滲透時，檢測不到的問題，例如:掃描網頁工具未發現漏洞。 2.檢查網頁原始碼中的函數，透過白箱測試檢查較全面。 3.可以避免黑箱測試時所造成的漏洞誤判。	1.需投入大量時間成本以及人力。 2.滲透測試員專業技術不一，可能導致無法進行全面檢測。 3.滲透測試過程中，可能發現的是內部的弱點，而不是外部的弱點。

2.4 序列化與反序列化漏洞成因分析

序列化是一種將結構化的物件轉換成位元組的過程，使其能夠被傳輸或儲存為文件格式。當我們在電腦上輸入文字時，系統會自動將其轉換為對應的二進制位元組，這即是序列化的過程。這些序列化後的位元組可以被傳送到其他地方，或者存儲在文件中以供將來使用。

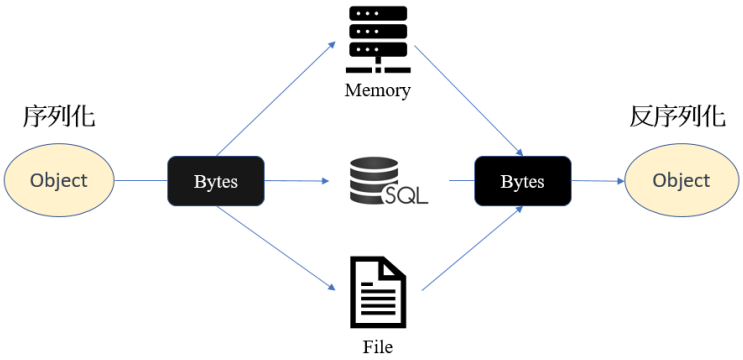
反序列化則是序列化的逆過程，以位元組表示的資料轉換回原始的結構化物件。其過程如圖四所示。在資訊安全領域中，反序列化本身並沒有風險。但攻擊者可以利用不安全的反序列化漏洞執行惡意命令，從而入侵系統或損害應用程式的安全性。因此，確保實施安全的反序列化是維護系統安全的一部分。

現今的網路生態中，網站和應用程式往往依賴大量資料庫和元件，這些資料庫和元件之間存在複雜的依賴關係。序列化是將物件轉換為可儲存或傳輸的格式。反序列化則是將這些資料還原為應用程式可用的物件，雖然這一過程在資料傳輸和儲存中十分常見，但同時也帶來潛在的安全風險。反序列化攻擊可能發生的原因如下：

1.在序列化和反序列化的過程中，開發人員經常會忽略對輸入進行完整驗證，這可能會導致漏洞的產生。

2. Java 類別中的某些物件，在進行序列化和反序列化的過程中可能存在著漏洞。

3.當進行反序列化操作時，若未驗證輸入資料的來源，可能導致未經授權訪問和操作。



圖四：序列化及反序列化過程

2.5 相關文獻回顧

近年來已有相關文獻針對 Java 系統的序列化和反序列化安全進行相關研究，如表二所示，分別概述如後。

Fingann 提供了 Java 序列化 API 研究的敘述，以及攻擊者如何使用不用技術利用其漏洞，以及可以採用那些緩解策略來最大化降低風險，包括使用白名單及黑名單防止被序列化[5]。

Dutka 設計了一種自動化安全測試方法包括靜態測試及動態測試，運用於測試 Java 的網頁安全性，且分析了在 Java 中可能出現的漏洞類型，以及如何修復這些漏洞，於不安全的反序列化中也說明了該漏洞的危害是非常高的，在預設的情況下是非常的脆弱，非常容易就會被攻擊者執行惡意的指令[6]。

Rasheed 評估了兩個靜態分析工具，用於檢測反序列化漏洞，且提出了一個用於檢測反序列化的過濾器，針對多種程式設計語言的 YAML 格式解析中的 DOS 漏洞進行研究，發現了前人未發現的安全性問題[7]。

Lai 等人提出了一種複合發現方法，此方法運用於 Java 反序列化漏洞中查找小工具鏈，且發現了 52 個小工具鏈，該方法結合了序列化協議及反射機制，此評估方法是在 Apache 中進行，結果顯示他們的方法克服了常見的假陽性與假陰性，且設計的正確性及高效性與其他相比佔據很大的優勢[8]。

Koutroumpouchos 等人提出了一種叫做 ObjectMap 的工具，該工具可用於檢測 PHP、Java 類型的反序列化漏洞，可以發現有漏洞的網站中可能的注入點，強調了需要運用自動化的方式來檢測能提高效率，但仍然缺乏完整的自動檢測工具和相關研究[9]。

Sayar 等人分析了 CVE(Common Vulnerabilities and Exposures)中 104 個 Java 漏洞，且針對了 19 種遠端程式碼執行(Remote Code Execution, RCE)反序列化攻擊進行了 256,515 次實驗。且發現類別中看似沒有威脅的細節，假設將其公開，就可能引入一個小工具，發現 37.5%的類別沒有做修補，使這些類別可運用於未來的攻擊中[10]。

表二：Java 序列化和反序列化安全相關文獻

項次	時間	作者	方法
1	2020	Fingann [5]	提供了 Java 序列化 API 研究的敘述，及可以採用那些緩解策略來最大化降低風險
2	2021	Dutka [6]	提出了一個自動化測試框架，用於檢測 Java Web
3	2021	Rasheed [7]	提出了一個用於檢測反序列化漏洞的工具，並且評估了兩個靜態檢測工具
4	2022	Lai [8]	提出了一種用於 Java 反序列化漏洞中查找小工具鏈的方法
5	2019	Koutroumpouchos [9]	提出了一種叫做 ObjectMap 的工具，用於檢測 PHP、Java 類型的反序列化漏洞
6	2023	Sayar [10]	分析了 104 個反序列化 CVE 漏洞，並研究出 Java 類別中有 37.5%並無法更新

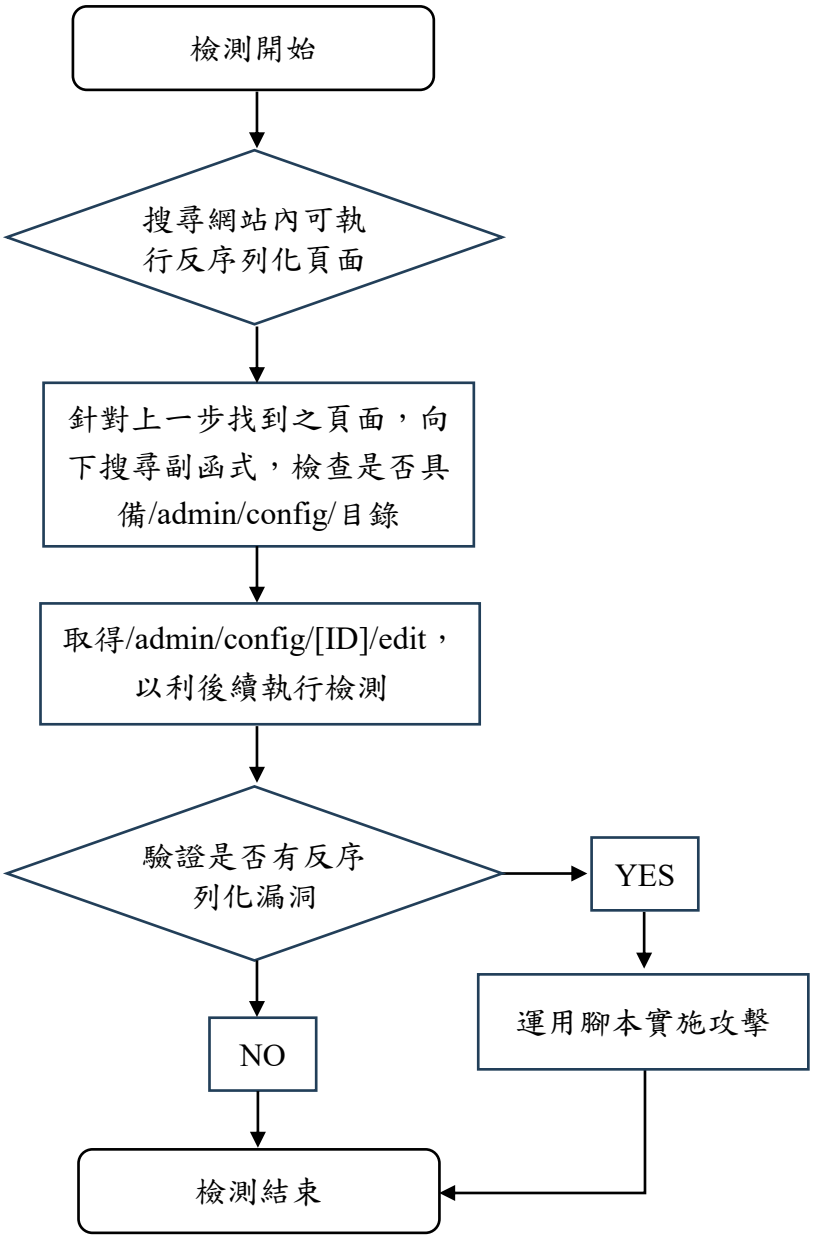
參、本研究提出之方法

Java 序列化是指物件轉換成位元組的過程，便於保存在文件或資料庫中，而 ObjectInputStream 類別的 writeObject()方法可以用於序列化。反序列化是指將位元組恢復成物件的過程，ObjectInputStream 類別的 readObject()方法可以用於反序列化。

而序列化及反序列化本身是安全的，並不存在威脅，會發生漏洞則是因為輸入的反序列化資料可被使用者控制，那麼攻擊者即可通過開發腳本進行輸入，讓反序列化產生非預期的過程，在這個過程中執行撰寫的惡意攻擊程式碼進行攻擊。

本研究聚焦於對 Java 中的 ObjectInputStream.class 的 readObject()方法進行反序列化的實驗環境設計及攻擊腳本開發。環境中包含各種服務版本皆無 CVE。研究分為黑箱測試及本文所提出之方法來針對本環境以檢測不安全的反序列化漏洞。重點在於測試人員的技術能力不一，所以需要對不同語言有一定的了解，才能在原始碼檢測時發現不安全的反序列化漏洞。

本研究透過解析 Java 原始碼的方式，檢測 readObject()方法的使用情況。根據大多數 Java 網頁系統的設計，先進行搜索網站上可執行序列化和反序列化操作的頁面。為了避免網頁遭受駭客攻擊並獲得管理者權限，本研究設計了反序列化頁面需要同時具備 Web 系統帳戶權限才能執行。搜索到可執行反序列化的頁面後，進行第二步驟，檢測該頁面是否有 Input 參數欄位的.class。接著進行第三步驟，取得管理者權限頁面的 ID，回到第二步驟將 ID 帶入，再撰寫攻擊腳本，對目標進行檢測。其詳細流程步驟如圖五所示。



圖五：本研究提出之反序列化漏洞偵測流程

肆、實驗與分析

4.1 實驗環境

為了防止駭客入侵並取得網頁管理者權限，進而對系統進行進一步滲透造成更嚴重的損害，我們設計了這個實驗環境來降低這種風險。在進行滲透測試時，我們無法保證每次都能檢測到漏洞的存在。為了提高測試的準確性，我們將使用白箱測試和黑箱測試

來測試這個實驗環境。我們首先使用黑箱測試進行弱點掃描。然後我們使用白箱測試進行檢測，只有擁有 Web System Account 權限的用戶才能夠訪問和執行反序列化頁面，透過 Admin 權限發現存在反序列化漏洞，然後開發攻擊腳本進行攻擊，以展示反序列化攻擊的方法。

本研究實驗環境為一般電腦處理器為 12 代 i7-12700H、圖形運算核心為 NVIDIA GeForce RTX 3050 Ti、主機記憶體為 DDR4 64GB、GPU 記憶體為 32GB。而本研究為自行架設虛擬環境於 VMware 中，為使用 Linux 5.10.0-kali3-amd64、虛擬機記憶體為 2GB、網頁伺服器為 Tomcat 8.5.63、資料庫伺服器為 MySQL 5.7.29-0ubuntu0.18.04.1、JVM 版本為 11.0.10+9-post-Debian-1、Java 版本為 OpenJDK version 17.0.9-ea。

4.2 黑箱測試

針對本研究環境運用黑箱測試工具進行弱點掃描，運用了於表三中 3 種滲透測試人員常用的工具進行測試，並進行了效能及操作難易度比較。

表三：黑箱測試工具比較

軟體名稱 / 功能	Zap	Invicti	Acunetix
可檢測反序列化漏洞	是	是	是
系統持續更新	是	是	是
圖形化介面	是	是	是
安裝難易度	易	易	易

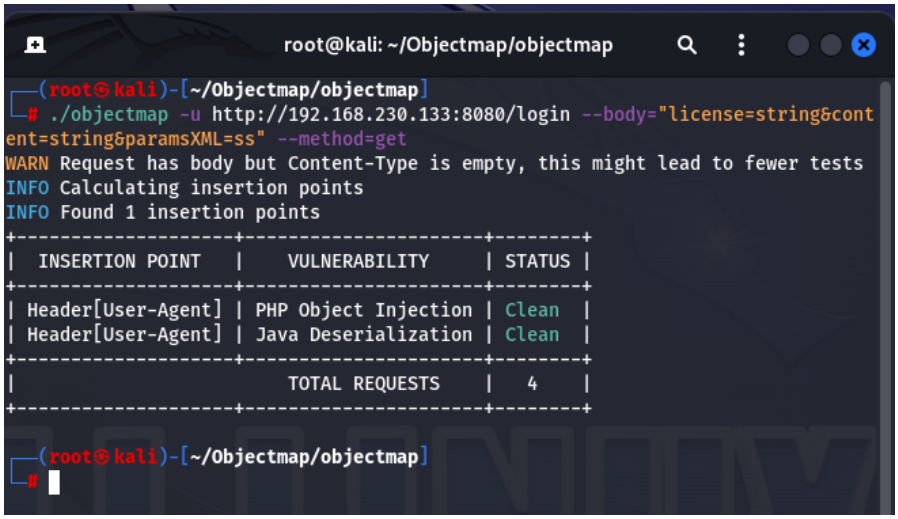
本研究使用滲透測試員經常採用的黑箱測試工具來進行測，這些工具都將 OWASP TOP 10 中的內容列入弱點清單，並針對其特徵進行了加強的掃描，於表四中列出本研究所使用的黑箱測試工具版本，由於 ZAP 為開源軟體，無須付費因此使用最新版本，而 Invicti 及 Acunetix 則是需要付費的軟體，所以本研究運用了舊版本進行測試。

表四：黑箱測試工具版本

軟體名稱 / 版本	Zap	Invicti	Acunetix
測試版本	2.14.0	22.12.0.38896	14.7.220401065
最新版本	2.14.0	24.4.0	24.3.240411164

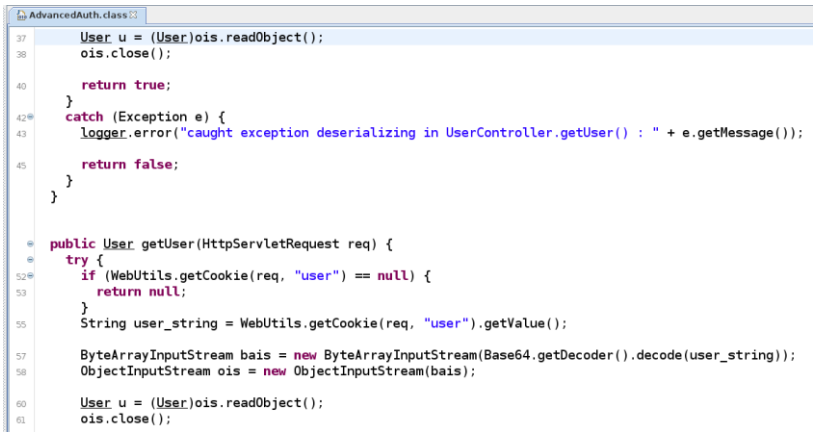
4.3 本文所提出之方法驗證

在本研究中，我們首先使用上述三種黑箱測試工具進行測試，沒有檢測到不安全的反序列化漏洞。另外，再採用文獻探討中提到的 ObjectMap 反序列化漏洞工具[9]進行了全面的檢測，卻未能成功偵測到不安全的反序列化漏洞。該工具僅確認了本研究所使用的程式中存在 Java 反序列化元件，但無法準確檢測出其漏洞，如圖六所示。這顯示了漏洞檢測工具的限制，強調了對於漏洞的有效檢測需要綜合考慮多種因素，並可能需要進一步的手動分析。



圖六：運用 ObjectMap 掃描本研究環境，未偵測到不安全的反序列化漏洞

執行本研究實驗必須具備 Web Seystem Account，為避免攻擊者利用管理者權限進一步滲透，所以設計本實驗環境。步驟一先於網頁中搜索可執行反序列化頁面，搜尋到後於圖七中"AdvancedAuth.class"中搜索"readObject()"，於原始碼第 37 及 60 行找到運用該函數的地方，透過這樣的步驟，我們可以確認確實存在著反序列化的物件，並且為後續實驗提供了必要的基礎。



圖七：檢查 AdvancedAuth.class 中原始碼

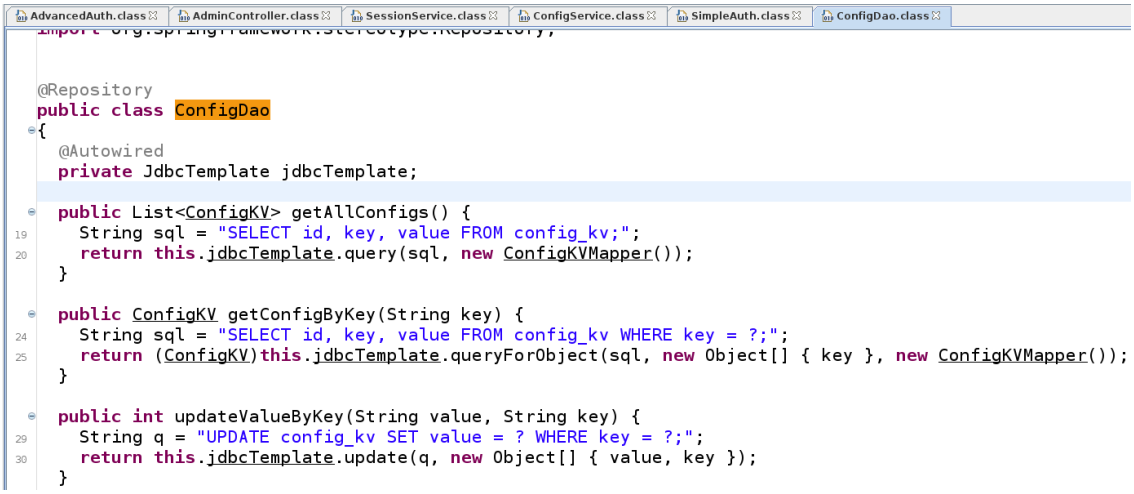
確認存在反序列化物件後，步驟二針對使用者會輸入的地方進行原始碼檢視，使用者會輸入的區域於本研究位於"AdminController.class"，並於其中發現具備/admin/config/目錄，於圖八中原始碼第 105 行發現，且"isAuthenticated"將被運用，而執行運用的是"SessionService.class"發現具備目錄後實施步驟三。

圖八：檢查 AdminController.class 中原始碼

步驟三進行搜尋取得 ID 於圖九中轉到"SessionService.class"，發現這個"isAuthenticated"的外部類別是"isAdvancedAuth"，而在"sesionService"的"init"函數中，可以看到根據"advanced_auth"來決定是運用"simpleAuth"還是 "advancedAuth"。

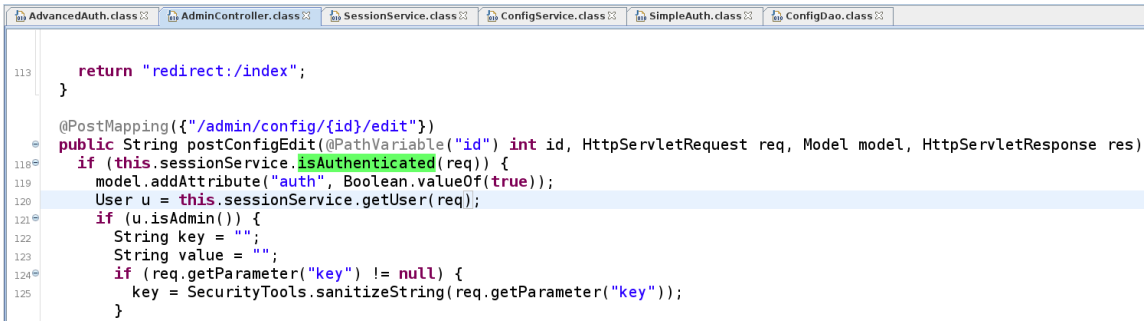
圖九：檢查 SessionService.class 中原始碼

使用"getConfigValue"進行追蹤至"ConfigDao.class"，於圖十在"ConfigDao.class"中發現"getConfigBykey"於表中，因為本研究為運用 admin 權限進行測試使得可以瀏覽該頁面，所以取得"id"的方式可運用 SQL Injection、運用 BurpSuite 中 Spider 模組爬取網頁等方式取得。



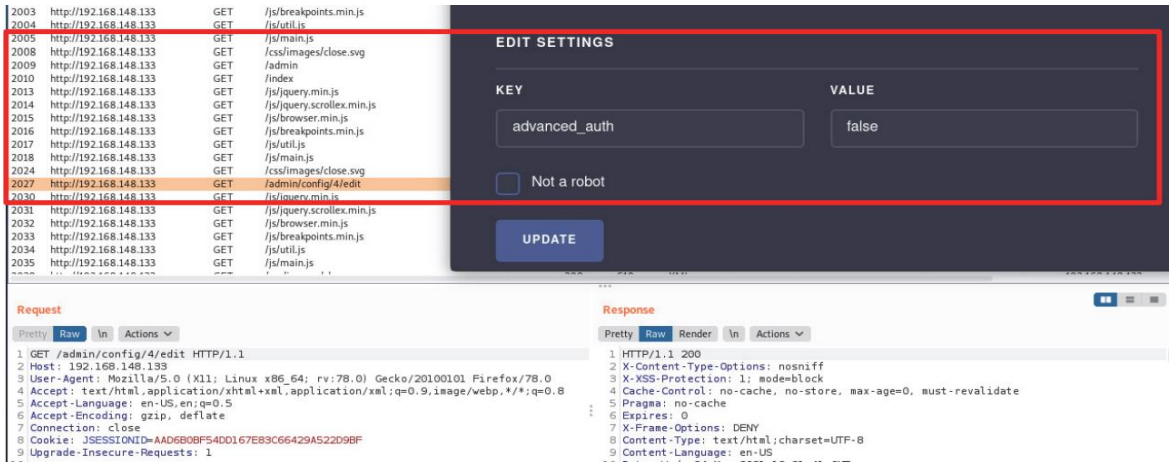
圖十：檢查 ConfigDao.class 中原始碼

圖十一中檢查"AdminController.class"，於 116 行中發現了"/admin/config/{id}/edit"，但它"id"實際上是固定的值，所以取得"id"的方式可運用 SQL Injection、運用 BurpSuite 中 Spider 模組爬取網頁等方式取得。



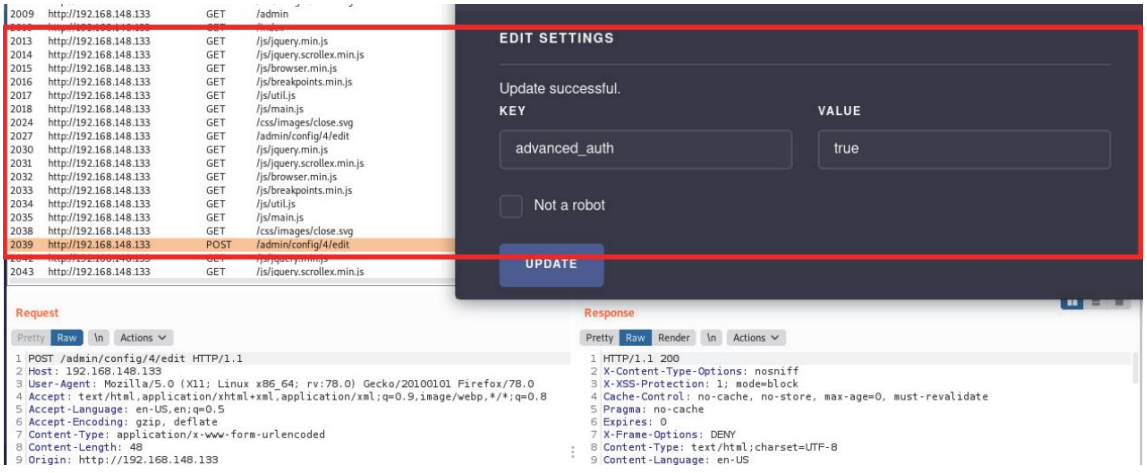
圖十一：再次檢查 AdminController.class 中原始碼

透過本研究所提出之方法，運用 BurpSuite 中 Spider 模組爬取網頁"id"後，於圖十二可發現 Get url 後顯示"/admin/config/4/edit"，這個"id"值抓取封包後發現是"4"，重整頁面跳轉到"/admin/config/4/edit"，連至該頁面可於 key、Value 兩個欄位進行輸入，該頁面 Value 欄位預設設定值為"false"，並無執行反序列化。



圖十二：發送第一次封包查看序列化狀態

進行步驟四第二次傳送封包將"false"改為"true"後於圖十三，可發現頁面內容已被成功竄改完成，且伺服器進行解析後回傳並非原始"false"狀態，可成功寫入更改"edit"頁面內容進行竄改，所以可得知一旦攻擊者取得 Admin 權限利用這個方式進行輸入，且發現進行了竄改，這樣就可以透過這個方式開發攻擊腳本針對目標進行反序列化攻擊。



圖十三：發送第二次封包查看序列化狀態

於圖十四再次檢查"AdvancedAuth.class"，發現"ois"來自"ObjectInputStram (bais)"和"cookie"中的用戶參數，並使用了 base64 編碼，將反序列化代碼中的 base64 編碼轉換成"cookie"中的使用者變數，檢查完全部流程後就可以撰寫反序列化攻擊腳本進行遠端攻擊。

圖十四：再次檢查 AdvancedAuth.class 中原始碼

4.4 YsoSerial 建置 Payload

在發現漏洞後，為確保攻擊程式碼的有效性，本研究使用 YsoSerial 工具來建立 Payload [11]。YsoSerial 是一個開源 Java 反序列化利用工具。這個工具專為測試 Java 應用程式中的反序列化漏洞而設計。當 Java 應用程式從網路或文件中讀取序列化元件時，若未正確驗證這些元件，攻擊者可利用此漏洞來構建惡意元件，從而實現遠端程式碼執行或 DDoS 等攻擊。因此，在開始編寫攻擊程式碼之前，必須嚴格使用 YsoSerial 工具來建立 Payload，這樣可以確保攻擊的有效性及成功性。YsoSerial 具有生成多種類型的 Java 反序列化 Payload 的能力，這些 Payload 對於滲透測試員來說是寶貴的資源，可用於評估和驗證應用程式的安全性。然而，YsoSerial 僅應用於合法的安全測試和研究目的，本實驗攻擊 Payload 運用 CommonCollections4，如圖十五所示，如不當使用這種工具可能導致嚴重後果因此在使用之前，必須深入了解相關的法律法規和安全標準，並確保遵守所有適用的法律和道德準則。

```
$  java -jar ysoserial.jar
Y SO SERIAL?
Usage: java -jar ysoserial.jar [payload] '[command]'
  Available payload types:
     Payload             Authors                    Dependencies
     -------             -------                    ------------
     BeanShell1          @pwntester, @cschneider4711 bsh:2.0b5
     C3P0                @mbechler                  c3p0:0.9.5.2, mchange-commons-java:0.2.11
     Clojure             @JackOfMostTrades          clojure:1.8.0
     CommonsBeanutils1   @frohoff                   commons-beanutils:1.9.2, commons-collections:3.1,
     CommonsCollections1 @frohoff                   commons-collections:3.1
     CommonsCollections2 @frohoff                   commons-collections4:4.0
     CommonsCollections3 @frohoff                   commons-collections:3.1
     CommonsCollections4 @frohoff                   commons-collections4:4.0
     CommonsCollections5 @matthias_kaiser, @jasinner commons-collections:3.1
     CommonsCollections6 @matthias_kaiser           commons-collections:3.1
     FileUpload1         @mbechler                  commons-fileupload:1.3.1, commons-io:2.4
     Groovy1             @frohoff                   groovy:2.3.9
     Hibernate1          @mbechler
     Hibernate2          @mbechler
     JBossInterceptors1  @matthias_kaiser           javassist:3.12.1.GA, jboss-interceptor-core:2.0.0.
     JRMPClient          @mbechler
     JRMPListener        @mbechler
```

圖十五：YsoSerial 工具 Payload 列表[11]

在完成攻擊腳本後，我們針對目標展開了攻擊，成功建立了 reverse shell 於圖十六所示。這種情況若發生在現實環境中，將對公司及相關單位造成巨大損害。駭客可能利用這種方式橫向擴散，進而取得公司內部的機敏資料。這不僅對業務運營造成直接衝擊，還可能損害公司聲譽並引發法律問題。因此，我們必須採取積極的安全措施，包括加強網絡防禦、定期漏洞掃描和強化內部安全培訓，以減少這種風險的發生。

```
Login Success!
Config Success!
--2024-04-29 23:15:41--  https://jitpack.io/com/github/frohoff/ysoserial/master-SNAPSHOT/ysoserial-master-SNAPSHOT.jar
Resolving jitpack.io (jitpack.io)... 172.67.72.129, 104.26.8.99, 104.26.9.99, ...
Connecting to jitpack.io (jitpack.io)|172.67.72.129|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: /com/github/frohoff/ysoserial/master-2874a69f61-1/ysoserial-master-2874a69f61-1.jar [following]
--2024-04-29 23:15:58--  https://jitpack.io/com/github/frohoff/ysoserial/master-2874a69f61-1/ysoserial-master-2874a69f61-1.jar
Reusing existing connection to jitpack.io:443.
HTTP request sent, awaiting response... 200 OK
Length: 59525398 (57M) [application/java-archive]
Saving to: 'ysoserial.jar'

ysoserial.jar       100%[==================>]  56.77M  3.44MB/s    in 16s

Last-modified header invalid -- time-stamp ignored.
2024-04-29 23:17:50 (3.54 MB/s) - 'ysoserial.jar' saved [59525398/59525398]

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Your Shell
listening on [any] 8888 ...
Serving HTTP on 0.0.0.0 port 6666 (http://0.0.0.0:6666/) ...
connect to [192.168.230.133] from (UNKNOWN) [192.168.230.130] 53492
whoami
tomcat8
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:89:a5:21 brd ff:ff:ff:ff:ff:ff
    inet 192.168.230.133/24 brd 192.168.230.255 scope global dynamic noprefixroute eth0
       valid_lft 1731sec preferred_lft 1731sec
    inet6 fe80::20c:29ff:fe89:a521/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

圖十六：運用腳本完成攻擊示意圖

伍、結論

近年來，不安全的反序列化漏洞持續浮現並被列入 OWASP TOP 10，突顯了其在資安領域中的重要性。隨著現代網路的快速發展，人們對網路的依賴程度不斷提升，許多公司已轉向使用網頁辦公系統，逐漸淘汰了傳統的手寫方式，使得駭客的攻擊隨時都有可能會發生，因此建立健全的資安意識變得至關重要。從資安專家的角度來看，強調白箱測試的重要性是確保系統安全的關鍵手段，組織和企業應將白箱測試納入滲透測試流程中，以確保對系統進行全方位的檢測和保護。

透過本研究，我們可以深入瞭解白箱測試在強化系統安全方面的獨特價值。白箱測試的關鍵在於其能夠檢測到黑箱測試無法直接觀察到的缺陷和漏洞，這對於維護系統的完整性至關重要。我們的研究清楚的顯示了白箱測試在測試過程中的不可或缺性，並強調了其在測試過程中的重要性。

在進行滲透測試時，單純依賴黑箱測試的報告是不足夠的。僅依靠黑箱測試所顯示的漏洞來評估系統的安全性，往往會忽略一些潛在的風險和漏洞。因此，我們強烈建議在測試流程中加入白箱測試，以全面評估系統的安全性。雖然同時進行白箱和黑箱測試會增加成本，但從長遠來看，這是值得的投資。系統安全性不容忽視，尤其對於需要保護的機敏資料和重要資產的機關而言，透過這種綜合的測試方法，可以更好的降低系統被駭客入侵的風險，從而確保系統的穩健性和可靠性，雖然成本可能較高，但這種綜合測試方法是確保系統安全性的有效手段，值得給予充分的重視和投入。

[誌謝]

本研究感謝國家科學及技術委員會計畫經費支持，計畫編號 NSTC 112-2221-E-606-009-MY2。

參考文獻

[1] 裴洛西訪台頻傳駭客攻擊 NCC：資安事件 1 小時內通報, https://www.setn.com/news.aspx?newsid=1156169 (2024/4/27).

[2] 7-11 電視遭駭！秀「裴洛西滾出台灣」字卡門市正常營運, https://today.line.me/tw/v2/article/j71KjQx (2024/4/27).

[3] 1H 2023 Global Threat Landscape Report, FortiGuard Labs, https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/threat-report-1h-2023.pdf

[4] OWASP Top 10 2021, https://www.synopsys.com/glossary/what-is-owasp-top-10.html (2024/4/27).

[5] S. Fingann, “Java Deserialization Vulnerabilities,” Master's thesis, University of Oslo, 2020.
[6] T. Dutka, “Automatisierte Security Tests von Java WebApplikationen,” PhD diss., University of Applied Sciences, 2021.
[7] S. Rasheed, “Security Analyses for Detecting Deserialization Vulnerabilities,” PhD diss., Massey University, New Zealand, 2021.
[8] Z. Lai, H. Qu, and L. Ying, “A Composite Discover Method for Gadget Chains in Java Deserialization Vulnerability,” In 10th International Workshop on Quantitative Approaches to Software Quality, pp. 10-17. 2022.
[9] N. Koutroumpouchos, G. Lavdanis, E. Veroni, C. Ntantogian, and C. Xenakis, “ObjectMap: Detecting Insecure Object Deserialization,” In Proceedings of the 23rd Pan-Hellenic Conference on Informatics, pp. 67-72. 2019.
[10] I. Sayar, A. Bartel, E. Bodden, and Y. Le Traon, “An In-depth Study of Java Deserialization Remote-Code Execution Exploits and Vulnerabilities,” *ACM Transactions on Software Engineering and Methodology*, no. 1, pp.1-45, 2023.
[11] YsoSerial, https://github.com/angelwhu/ysoserial (2024/4/27)

[作者簡介]

江竑緯，國防大學理工學院網路安全碩士在職專班，現任職於國防部資通電軍指揮部，研究方向為滲透測試及資訊安全。

周兆龍，國防大學國防科學研究所博士，現為逢甲大學資訊工程學系副教授，主要研究領域為資訊安全、影像處理、深度學習及生物辨識等。