優先考量應用程式安全

做好準備迎擊最新的 OWASP 10 大風險與威脅

簡介

最近網路應用程式防火牆又成爲眾所矚目的焦點，這個原因其來有自。還是老問題，就是我們一直聽到的安全漏洞與資料外洩。

維持網路應用程式的安全非常困難 — 甚至可以說是極其困難。

更別說企業所耗費的大量時間和成本。建立和維護完整的網路安全控制項可能會消耗公司有限預算當中很大的一部分，而這些預算原本應該用來開發使用者所需的實際應用程式功能。

實際上，網路應用程式安全性是如此深具挑戰，WhiteHat Security 在 2017 年應用程式安全性統計報告中指出，一般的網路應用程式有三個漏洞。[2] 這是因爲我們在滲透測試和補救措施上所投入的資金不夠嗎？我們難道不了解風險何在嗎？我們難道沒有部署適當的工具來降低這些漏洞所帶來的風險嗎？

這些問題一直都存在，由於難以在已發佈的每個新應用程式中建構和重建補救措施，使得這些問題始終難以解決。了解和防範網路應用程式漏洞通常需要專門的安全專業知識，只有少數的開發人員可以在完成實際開發的同時執行這項技術。

還好，我們還有其他選擇。擁有適當的工具和第三方控制項不僅可以大大降低風險，還可以加快應用程式的開發。

30%

報告指出，30% 的資料外洩與網路應用程式攻擊有關。[1]

[1] http://www.verizonenterprise.com/verizon-insights-lab/dbir/2017/

[2] https://info.whitehatsec.com/rs/675-YBI-674/images/WHS%202017%20Application%20Security%20Report%20FINAL.pdf

OWASP
10 大

漏洞與降低風險

開放式網路應用程式資安專案：
教育可以減少安全漏洞嗎？

無所不在的網路應用程式安全漏洞並非真的完全消失不見了。**2001** 年，許多的資安專家聚在一起，共同建立開放式網路應用程式資安專案（**OWASP**），以教育開發人員並希望藉此減少資安問題。**OWASP** 是一個非營利的國際組織，針對網路應用程式安全性提供許多公開的方法、文件、工具和培訓。

OWASP 10 大風險：風險分類法

OWASP 專案中最有名的是「OWASP 10 大風險」清單，該組織會不定期更新此清單，列出網路應用程式最常見的安全問題。公佈「OWASP 10 大風險」的目的是爲了提供有關網路應用程式漏洞的基本風險分類。

未來版本的「OWASP 10 大風險」將與廣爲人知的風險框架（如 ISO 31000：2015[3]）緊密結合，其目的是爲了提高專案的可信度和適用性，反過來說，這也將放大專案所秉持與堅持的理念。

保護*您的*應用程式：威脅概述

若您是負責網路應用程式的開發、安全與操作，熟知「OWASP 10 大風險」可幫助您有效保護應用程式。除此之外，許多產業與法規標準（如 PCI DSS）的基本核心要求是以「OWASP 10 大風險」來進行安全測試。OWASP 網站也列出其他參考「OWASP 10 大風險」的相關國際安全標準。[4] 透過研究最常見的網路應用程式安全問題並了解有效的緩解措施，來提高組織的安全性與保護關鍵應用程式，幫助確保資料的機密、完整與可用性。

[3] https://www.iso.org/iso-31000-risk-management.html

[4] https://www.owasp.org/index.php/Industry:Citations

如果某個網路應用程式無法正確清理外部資源的輸入，則會允許隱藏的注入式命令通過。

A1 注入式攻擊

注入式攻擊是常見的漏洞類別之一，其中外部來源所提供的輸入並未經過充分清理，這些外部來源通常含有攻擊者隱藏的應用程式命令。由於網路應用程式未妥善過濾輸入的內容，導致注入式命令可以進入本機系統或相關獨立（Dependent）系統。

常見的例子是 SQL 注入式攻擊。許多應用程式仰賴使用者的輸入來建立 SQL 陳述式，以擷取或記錄資訊，例如：

```
select * from USERTABLE where USERID = '[userid-from-web-form]' and PASSWORD = '[passwd-from-web-form]'
```

在正常的情況下，這個指令會比對 USERTABLE 資料表中的輸入項，若爲真，陳述式就會判斷其爲「真」，然後使用者可成功登入。

但是，如果使用者在網路表單上輸入“**password' OR 1=1**”作爲密碼，該怎麼辦：

```
select * from USERTABLE where USERID = '[userid-from-web-form]' and PASSWORD = 'password' OR 1=1;
```

如果沒有適當的清理和轉譯，SQL 伺服器會將 1=1 條件評估爲 TRUE，並使用提供的使用者名稱登入，而不管提供的密碼是什麼。這是一個非常簡單明瞭且目標明確的範例說明。SQL 注入式攻擊的手法可以非常複雜且充滿惡意，這種攻擊曾成功刪除過整個資料庫、修改過記錄，甚至使機密資料外洩。

> 降低注入式攻擊風險

關於資料輸入與安全性，您不能像過去一般無條件信任使用者所提供的資料。所有的輸入都必須經過檢驗、轉譯、清理與過濾。注入式攻擊可能會在一般使用者輸入表單時發生，也可能隱藏在網路領域中。不管使用者輸入內容爲何，利用參數化 SQL[5] 可以透過分離輸入資料並將其與程式碼區分開來，來緩解這類風險。同時，建議您監看傳回給使用者的外送回應，以偵測注入式攻擊所導致的資訊外洩情況。

[5] https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet#Defense_Option_1:_Prepared_Statements_.28with_Parameterized_Queries.29

A2 無效的身分驗證

安全的基本概念是，準確知道使用者是誰（驗證）以及允許這些使用者做什麼（授權），這兩者相輔相成，缺一不可。談到網路應用程式驗證，我們不得不從密碼開始說起。儘管密碼有其基本特性，並且伴隨著固有的風險與不便，但很抱歉，密碼仍然是驗證使用者身分最基本的方式。

然而，憑證被竊是網路應用程式洩密的主要問題。[7] 不安全的使用者密碼，使諸如帳號密碼填充這類的攻擊既輕鬆又容易成功。當攻擊者將他們的手伸向大型資料庫時，就會發生帳號密碼填充攻擊。然後，他們可使用自動化工具針對各種網站和服務進行密碼測試，以查看哪種方法有效。據估計，財富 100 強企業中，登入網路應用程式的所有嘗試中有 90% 是帳號密碼填充，而不是合法的登入。[8] 由於我們一直拒絕移除舊密碼，加上聯合身分驗證的廣泛採用，使得帳號密碼填充這類攻擊變得輕而易舉。[9]

> 降低無效的身分驗證風險

避免發生密碼問題的其中一個方法就是不使用密碼。用戶端憑證、Token 式的雙重因素及聯合身分驗證都是可以減輕對密碼依賴的好方法。要建立強大的身分驗證系統並不容易，更遑論要確保其安全或維護其正常運作，因此，您必須善用聯合驗證功能來加速開發與交付您的應用程式 — 同時提高使用者的滿意度。切記：不管您的應用程式有多麼吸引人，使用者都不會想再多管理一組密碼。

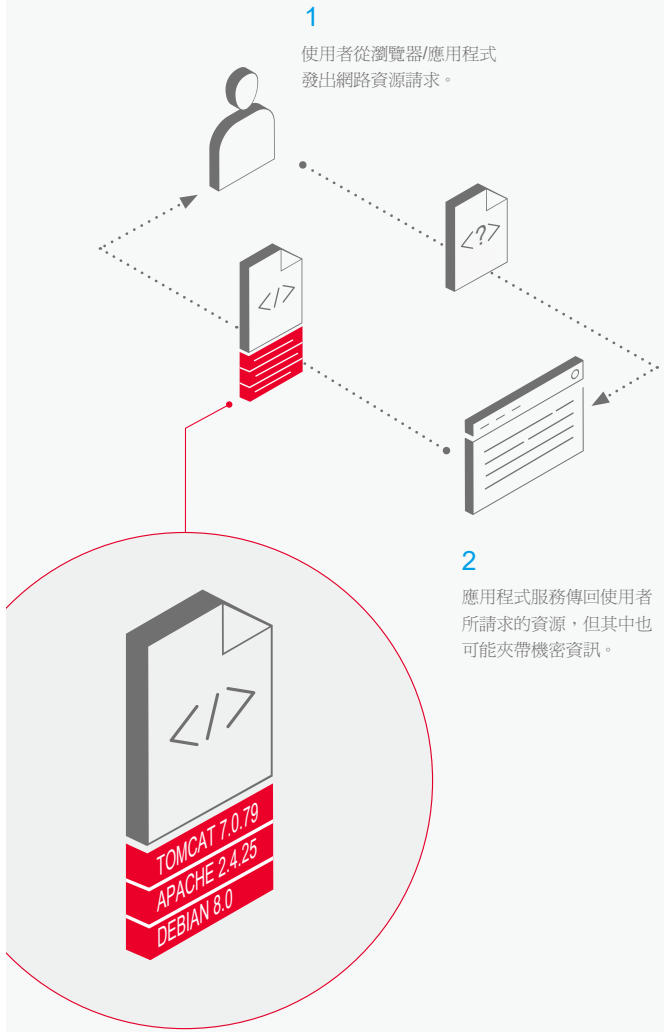
[6] https://www.teamsid.com/worst-passwords-2017-full-list/

[7] http://www.verizonenterprise.com/verizon-insights-lab/dbir/2017/

[8] http://www.darkreading.com/attacks-breaches/credential-stuffing-attacks-take-enterprise-systems-by-storm/d/d-id/1327908

[9] https://f5.com/about-us/blog/articles/credential-stuffing-what-is-it-and-why-you-should-worry-about-it-24784

網站通常會傳回比您所需還要多的資料，讓攻擊者有機可趁來利用這些有用資訊。



A3 機密資料外洩

機密資料外洩一直是一個嚴重的問題。什麼樣的機密資料會被洩露則視情況而定，但是，向攻擊者透露有關如何設計網路應用程式的任何資訊則讓情況大爲不妙。這類資訊對自動化掃描工具而言是垂手可得的結果，而且這些資訊可以讓攻擊者有機可趁。

網路應用程式所洩露的各種資訊當中，對攻擊者有用的通常包含以下：

- 詳細說明如何處理非預期輸入的錯誤訊息
- 伺服器上檔案的實際位置
- 元件或資料庫的特定版本
- 來自故障功能的堆棧追蹤（可以反向編譯和檢查）
- 揭露使用者識別碼有效性的「忘記密碼」功能錯誤訊息，可用來發現使用者帳號及密碼與發動暴力攻擊

> 降低機密資料外洩風險

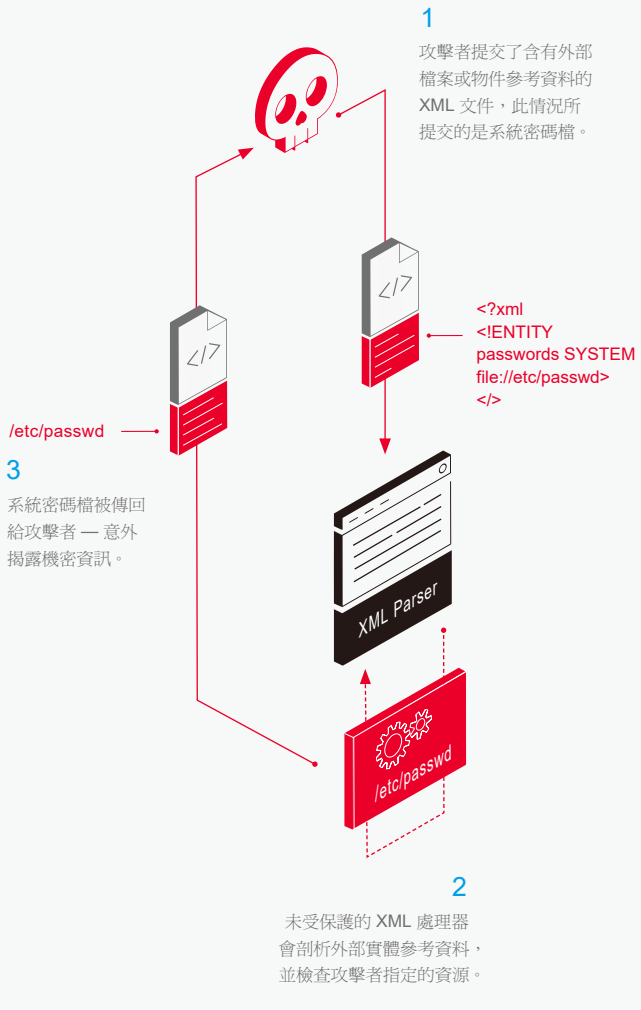
您可以採取以下幾個步驟來降低資料外洩的風險。除了其他方法外，較常見的是由網路伺服器提供供應商和版本資訊。[10] 確保使用者名稱不能透過伺服器回應代碼進行驗證：使用者名稱錯誤和密碼不正確應會產生相同的錯誤訊息。務必使用瀏覽器安全性功能來幫助保護傳輸中的機密資料。避免使用過時和已知與不夠強大的加密演算法和方法。易於使用的傳輸層安全性協定（TLS）已逐漸成爲網際網路的普遍規範。[11] 請使用加密或 Token 等技術獲得存取權限，讓所有儲存在網路應用程式中的機密資料無法直接被讀取，如果攻擊者能取得應用程式的存取權限的話。[12]

[10] https://f5.com/labs/articles/threat-intelligence/identity-threats/phishing-for-information-part-4-beware-of-data-leaking-out-of-your-equipment

[11] https://f5.com/Portals/1/PDF/labs/R065%20-%20REPORT%20-%20The%202016%20TLS%20Telemetry%20Report.pdf

[12] https://www.owasp.org/index.php/File:Reducing_Your_Data_Security_Risk_Through_Tokenization.pptx

接受 XML 輸入的應用程式可能會無意間允許外部命令，從而導致 XML 處理器洩露資料。



A4 XML 外部實體

強化或配置不正確的 XML 剖析器或處理器來評估含有 XML 文件的外部實體參考資料。接受 XML 輸入的應用程式（如 SOAP 服務）可能會無意間允許含有意外的外部參考資料和命令存取，這些外部參考資料和命令會導致 XML 處理器洩露內部檔案共享的資料、執行程式碼、啓動內部連接埠掃描以及執行拒絕服務（DoS）攻擊。

> 降低 XML 外部實體風險

仔細考量要先公開哪些 API，並確保易受攻擊的 XML 處理器已經過升級、修復或強化。除此之外，確保 HTTP 訊息大小、版本和方法都已強制執行。使用網路應用程式防火牆（WAF），您可以選擇在第一時間將適當的請求加入白名單或將發送到 XML 處理器的惡意請求封鎖。還有，WAF 可以幫助驗證 JSON、XML 和 SOAP 請求，並對可能導致 DoS 攻擊和未經授權資訊外洩的已知攻擊套用攻擊特徵加以防護。最後，網路防火牆和流量管理可以對傳送到其他端點、內部或外部的外送請求設定限制。

A5 無效的存取控制

無效的存取控制漏洞是指網路應用程式設計中的缺陷，此缺陷會不當執行對敏感物件（如目錄或記錄）的未經授權存取。例如，任何匿名使用者都可以僅透過了解發出什麼樣的 URL 請求來查看網站上的某些檔案；或者，應用程式可以執行已經進行須經某種程度驗證或授權的功能，但並未經過相關驗證或授權的程序。

> 減少無效的存取控制

所有的網路應用程式物件和頁面均應具有防護的機制，可以依預設拒絕存取。據此，系統應強制執行明確式授權，並僅授權給與特定使用者角色有關的使用者。

A6 安全配置錯誤

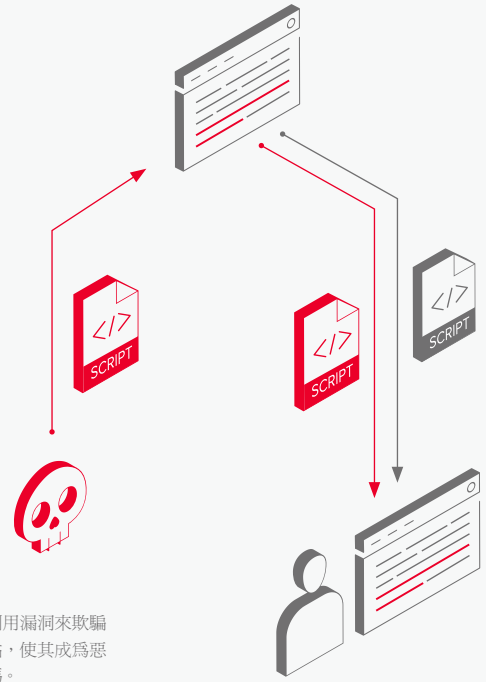
安全控制項可能基於各種原因被錯誤配置錯誤，但是常見的原因是由於使用者的文件中存在錯誤或遺漏，導致安全配置間存在空白或遺漏了某些步驟。然而，這可能只是系統管理員的疏忽或錯誤，畢竟，人非聖賢，孰能無過。我們也可能會忽略掉相關的軟體和基礎架構。大部分的網路應用程式都必須依賴於其他軟體（如 Apache、IIS 或 Nginx），而且可能也需連接其他應用程式、媒體庫和資料庫（如 PHP、ASP 或 SQL）。

> 降低安全配置錯誤風險

爲妥善保護網路應用程式安全，您必須適當管理配置軟體本身，還必須包含其他不可或缺的元件。還有，定期執行徹底審核，以確保控制措施確實執行與配置也同樣重要。

[13] http://www.verizonenterprise.com/verizon-insights-lab/dbir/2017/

XSS 可以在任何地方使外部使用者提供內容到網站，因此成爲最常見的漏洞類型之一。



1
攻擊者利用漏洞來欺騙合法網站，使其成爲惡意程式碼。

2
使用者從正常可信的網站發出請求，以取得正常的網路資源。

3
除了合法網站的程式碼外，使用者還不知不覺地接收並執行了攻擊者的惡意程式碼。

A7 跨網站指令碼（XSS）

跨網站指令碼（XSS）是一種輸入驗證漏洞，攻擊者可以利用該漏洞在受害者存取有漏洞的網站時，在瀏覽器中執行自己的惡意程式碼。XSS 可用於竊取工作階段 Token、發起隱藏的交易或顯示僞造或誤導的內容。更複雜的 XSS 指令碼甚至可以載入金鑰記錄器，以在受害者鍵入密碼時監看受害者的密碼，並將該資訊轉發給攻擊者操作的命令和控制伺服器。

XSS 可以在任何地方使外部使用者提供內容到網站，因此成爲最常見的漏洞類型之一。由於 XSS 使用的是與呈現網站頁面相同的 HTML 命令，因此要識別和刪除 XSS 並不是那麼容易。此外，攻擊者可以採用多種方式對 XSS 攻擊進行編碼。舉例來說，網頁上所彈出的“XSS”基本攻擊指令碼可能如下：

```
<script>alert('XSS')</script>
```

但經過編碼，它可能變成：

```
%253Cscript%253Ealert('XSS')%253C%252Fscript%253E
```

或：

```
<IMG SRC="jav&#x0D;ascript:alert('XSS');">
```

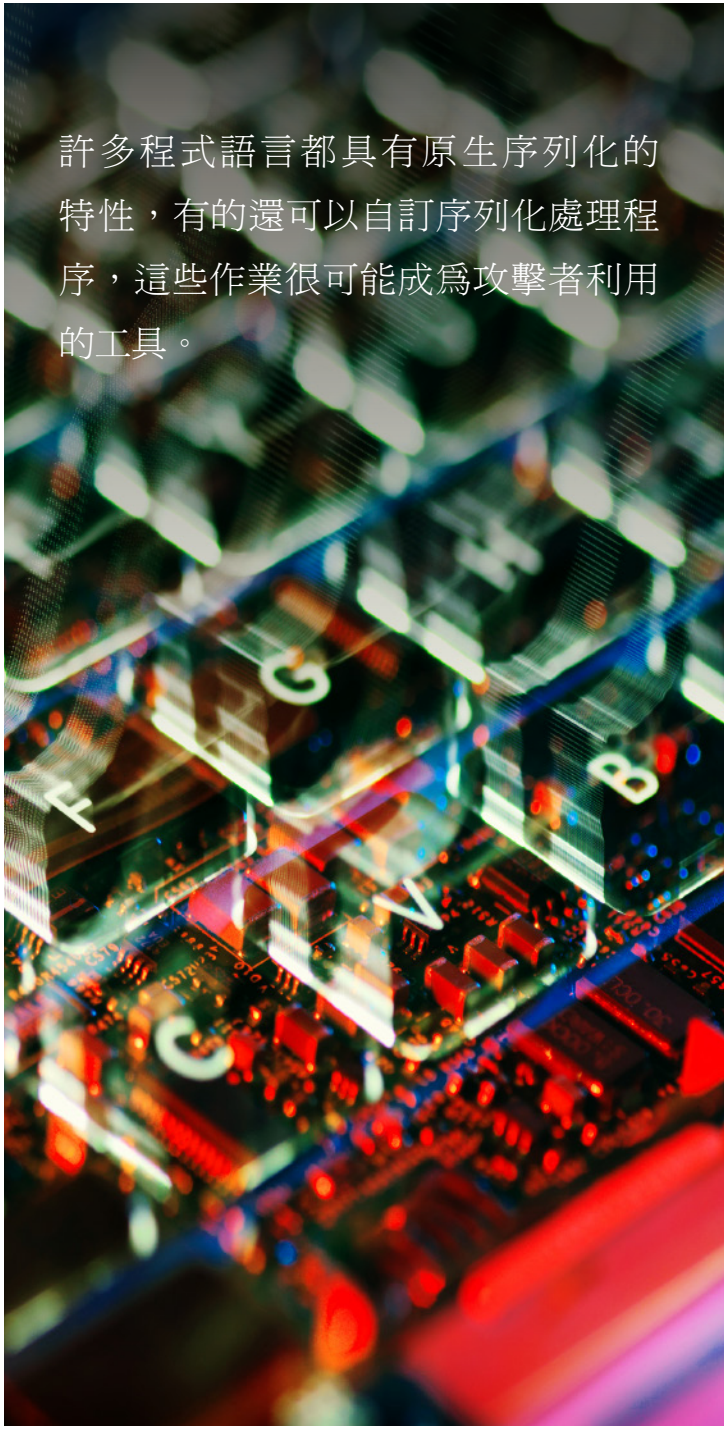
甚至是：

```
<IMG SRC=&#x6A&#x61&#x76&#x61&#x73&#x63&#x72&#x69&#x70&#x74&#x3A&
#x61&#x6C&#x65&#x72&#x74&#x28&#x27&#x58&#x53&#x53&#x27&#x29>
```

原生瀏覽器支援這麼多的編碼方式，我們很容易看出 XSS 漏洞是如何被利用的。更複雜的是，攻擊者可以採用多種不同的 XSS 交付方法，因此 XSS 變成非常好用的攻擊媒介。

> 降低跨網站指令碼風險

停止 XSS 攻擊可以往回追溯到使用者提供的任何外部資料都不可信任的想法。由於任何資料的輸入都可能嵌入惡意的負載，因此過濾所有傳入的內容變得至關重要，儘管這一系列動作全靠自己完成並不是那麼容易。還好，我們有 WAF 可以爲您提供幫助，利用可重複利用的架構來節省您的時間與資源。



A8 不安全的反序列化漏洞

此部分是 10 大風險中新增的項目，與物件的序列化有關，序列化是指將物件轉成可稍後儲存之資料格式的過程。想想看，檔案是如何儲存到硬碟中的，再想想看，資料是如何在網路上傳送的。資料會以 JSON 或 XML 等格式儲存在特定的結構中。去序列化則是完全相反的概念 — 從結構化的資料中進行讀取，然後從中建構物件。

許多程式語言都具有原生序列化的特性，有的還可以自訂序列化處理程序，這些作業很可能成爲攻擊者利用的工具。不安全的去序列化作業讓遠端程式碼的執行、拒絕服務、重播、注入式及權限升級等攻擊變得輕而易舉。

> 降低不安全的去序列化風險

若要避免去序列化的過程中遭受攻擊，您不應從不受信任的來源接受序列化物件。當您還是必須從不信任的來源接受序列化物件時，您可以強制執行 HTTP 訊息大小、版本、方法和所需的表頭。此外，進階的 WAF 技術可以驗證 JSON、XML 和 SOAP 請求，避免遭受已知的攻擊特徵攻擊，同時驗證資料的長度、數值與結構。您的 WAF 也應能讓您自訂機密資料的處理方式。

A9 使用有已知漏洞的元件

這是顯而易見的諸多風險領域當中的一個，但也值得我們探討，因爲許多的軟體元件被選用，純粹是爲了滿足某些基本操作要求。這類元件在實作時可能沒有已知的漏洞存在，而且使用者經常擔心會破壞功能或失去寶貴的傳統功能，因而避免升級。

這些擔心其來有自，但是成功利用已知的安全漏洞可能會導致服務遭受重大損失或使客戶失去信心，因此必須在此風險與可覺察的升級風險之間加以權衡。

> 降低使用有已知漏洞元件的風險

此處的關鍵在於，儘可能確保元件獲得更新。在無法確保元件是否更新的情況下，採用強大的 WAF 這類安全防護有助於避免被已知漏洞利用，同時確保軟體的完整與順暢運作。WAF 還可以在開發和推出修補程式時爲您節省時間。

A10 記錄和監控不足

儘管大多數應用程式的建構會記錄某一級別的存取和授權資訊，但許多應用程式沒有這麼做，或者在預設情況下並不會這麼做。這裡存在的風險是，未被收集或未經過分析的記錄資料和存取資訊將無法協助偵測資料的外洩，甚至在事件發生時無法幫助服務復原。

大多數的攻擊者都依賴安全部門盲目的自信與盲點，讓自己不被發現，並且依此入侵應用與網路。認真監控這些資料可以幫助您及早發現攻擊，使您能夠建立或進一步加強防禦姿態，並最終將企業所受到的破壞和影響降到最低。

> 降低記錄和監控不足風險

首先，確定哪些應用程式和端點可能會產生最有用的記錄資訊，並且可作爲異常行爲預警之用。您必須定義要收集哪些資料，以及如何分析和監控這些資料。優異的 WAF 解決方案可讓您將所有的網路應用程式記錄標準化 — 同時即時記錄該資訊以進一步進行分析和報告。

OWASP 10 大風險：
確保應用程式安全的其中一個難題

在評估無數的潛在攻擊媒介時，我們越來越清楚：您的網路應用程式面臨許多複雜的威脅，這些威脅越來越難以防禦且代價昂貴。大部分的開發團隊根本沒有足夠的資源來抵禦與每個媒介相關的攻擊，而且您需要有較高的專業知識，即使您有足夠的專案預算和時間，也很難獲得幫助。

好消息是，相較於過去，進階的 WAF 技術越來越容易取得，成本也越來越低。功能齊全的現代化 WAF 可幫助各種規模的企業確保其關鍵應用程式安全，無論這些應用程式是部署在資料中心還是混合雲端環境中。獨特而靈活的部署選項可以簡化實作程序，並輕鬆自訂應用程式的防護功能。

在滿足安全需求以防禦「OWASP 10 大風險」的同時，您必須考量應用程式的所有其他威脅：DDoS 攻擊、機器人攻擊、竊取智慧財產權和詐欺只是其中幾個例子。

沒有列入「OWASP 10 大風險」的威脅並不代表其重要性不高。

舉例來說，跨網站請求偽造（CSRF）就是使用瀏覽器誘騙受害者點擊實際上執行惡意操作（如在您的銀行應用程式內啓動轉帳作業）且看起來正常的連結。儘管 CSRF 沒有出現在今年的名單上，但可以肯定的是，犯罪分子仍在尋找機會利用它，並透過各種聰明的技巧來破壞您的應用程式。避免遭受「OWASP 10 大風險」威脅是一個複雜且必要的防禦措施，但這只是完整防禦策略的一部分，幫助保護您的應用程式、資料和業務。

如需更多有關影響應用程式和企業營運的更多威脅資訊，以及如何抵禦這些威脅，請造訪 f5.com/security。

[14] http://www.verizonenterprise.com/verizon-insights-lab/dbir/2017/

網路安全事件[14]

每個模式的百分比和發生次數

優先考量應用程式安全

始終保持在線狀態的應用程式可以強化並幫助企業轉型 — 但是駭客也可透過它們越過一般防火牆的保護，將資料外洩出去。大多數的攻擊是發生在應用程式層級，保護驅動業務的功能意味著要保護使其發生的應用程式。

您可透過 **AppProtectLibrary** 了解更多應用程式安全資訊。

如需進一步了解歡迎隨時聯繫我們的團隊，
我們將竭誠為您提供最優質的服務和支援。

詳情內容請洽代理商逸盈科技
台北 02 6636-8889　　新竹 03 621-5128
台中 04 3606-8999　　高雄 07 976-8909

美國總公司：401 Elliott Ave W, Seattle, WA 98119 | 888-882-4447 // 美洲：info@f5.com // 亞太地區 apacinfo@f5.com // 歐洲/中東/非洲：emeainfo@f5.com // 日本：f5j-info@f5.com