對本書的讚譽

「*Steve Wilson* 的教戰手冊對於人工智慧開發者和紅隊演練專家來說至關重要，它將巨大的風險轉化為可管理的挑戰，並提供了專業知識，以保障基於大型語言模型應用的安全。」

——*Marten Mickos*，*HackerOne* 執行長

「一本由大型語言模型資安教父 *Steve Wilson* 所寫、創新者必讀的書。這本書對於領導者非常重要，它提供了關於保護大型語言模型技術的重要見解。」

——*Sherri Douville*，*Medigram* 執行長

「*Steve Wilson* 在業界寶貴的專業知識，再搭配他獨特的動態方法來應付瞬息萬變的環境，使這本書成為必讀之作。根據我在 *AI* 紅隊的經驗，我全力支持這本書中頂尖的全端方法及其嚴謹、多面向的見解。」

——*Ads Dawson*，*Cohere* 資深資安工程師

「《*LLM* 資安教戰手冊》是當我們正努力地快速採用 *GenAI* 和 *LLM*，並確保組織結果的安全時，關於資安產業重要且全面的指南。」

——*Chris Hughes*，*Aquia* 總裁及 *Resilient Cyber* 創始人

「這本書深入淺出、清晰明快、簡潔而詳盡，它探討了各種重要的主題，包括 *LLM* 架構、信任邊界、*RAG*、提示注入攻擊和過度代理。如果你正在使用 *LLM*，你需要閱讀並理解本書。」

——*Krishna Sankar*，傑出的 *AI* 工程師和 *NIST AI* 安全研究所的計畫主持人

前言

在世界各地，我們都正乘著大語言模型（LLM）的浪潮，這實在是令人興奮！當 ChatGPT 突然出現時，它不僅打破了紀錄，還成為歷史上最快被採用的應用程式。現在好像地球上的每個軟體供應商，都在爭先恐後地將生成式 AI 和 LLM 技術嵌入到他們的軟體堆疊中，將我們推向未知的領域。這熱潮是真實的，炒作是合理的，而且可能性似乎是無限的。

但且慢，因為事情有個轉折。當我們驚嘆於這些科技奇蹟時，他們的資安設施，說得委婉一點，是一項正在進行中的工作。但事實如何呢？許多開發人員都是在沒有地圖的情況下踏入這個新時代，他們大多沒有注意到表面下的資安與安全性的流沙。現在有些事幾乎成了例行公事：每週我們都會看到有另一則頭條新聞，報導關於大型語言模型的小插曲。到目前為止，這些個別事件所造成的影響還算溫和，但千萬別心存僥倖，我們正在面臨災難。

這些風險不僅僅是假設，而是真實存在的，並且時間正在流逝。如果不深入探討 LLM 資安風險的陰暗水域，以及如何應對這些風險，我們所冒的就不只是小故障的風險，而可能是大災難。現在正是開發人員整裝待發、了解情況並取得先機的時候。動作快點！

誰該讀這本書

本書的主要讀者是正在建立嵌入 LLM 技術的客製應用程式的開發團隊。透過我最近在這個領域的工作，我了解到這些團隊通常規模很大，而且成員的背景也非常多元，其中包括精通「網路應用程式」技術的軟體開發人員，他們正在邁出人工智慧的第一步。這些團隊也可能由 AI 專家所組成，他們第一次將自己的技術從後台辦公室帶到鎂光燈下，而這所面臨的安全風險大不相同。他們也包括應用程式資安專家和資料科學專家。

除了核心受眾之外，我也發現許多其他相關人士對這些資訊頗感興趣，包括參與這些專案的延伸團隊，他們希望了解技術的基礎，以幫助減輕降低這些新技術的關鍵風險。這些人包括軟體開發主管、資安長（CISO）、品質工程師以及資安維運團隊。

我為什麼寫這本書

我一直對人工智慧充滿著興趣。我深深地記得當我還是青少年時，就曾經在自己的 Atari 400 家用電腦上寫電動遊戲。大約在 1980 年，這台小電腦只有 8 KB 的記憶體，但我還是成功地在那台機器上複製了一個完整的 Tron Lightcycles 遊戲，還用了一個簡單但有效的 AI 來駕駛其中一輛車，能在單人模式下對戰。

在我的職業生涯中，我參與了數個 AI 相關的專案。大學畢業後，我和好友 Tom Santos 創立了一家人工智慧軟體公司，以數千行手寫的 C++ 程式碼為基礎，利用遺傳演算法解決看似棘手的問題。後來，我和朋友 Kedar Poduri 和 Ebenezer Schubert 一起幫助 Citrix 建構一個大規模的機器學習系統。但當我第一次看到 ChatGPT 時，我知道一切都改變了。

當我第一次接觸 LLM 時，我在一家開發網路安全軟體的公司工作。我的工作是幫助大公司找出並追蹤軟體中的漏洞。很快地我就發現 LLM 帶來了獨特且嚴重的安全漏洞。在接下來的幾個月裡，我重新調整了我的職業生涯，開始追尋這個變革。我開始了一個以 LLM 資安為主題的開源計畫，稍後你會聽到更多關於這個計畫的內容。我甚至換了工作，加入 Exabeam，一家專注於人工智慧與網路安全交會點的公司。而當 O'Reilly 的編輯找我寫一本關於這個主題的書時，我知道我必須抓住這個機會。

瀏覽本書

本書共有 12 章，分為三個邏輯部分。我將在這裡簡述每個部分和每個章節，讓你了解該方法，這樣你在閱讀時就會知道接下來會發生什麼事。

第一部分：奠定基礎（第 1 ～ 3 章）

本書的前幾章為理解 LLM 應用程式的安全狀況奠定了基礎，這些章節應可提供你基礎，讓你能自信地解決使用 LLM 開發應用程式所面臨的問題：

- 第 1 章「崩壞的聊天機器人」講述了一個真實世界的案例研究，在這個案例中，業餘黑客摧毀了全球最大軟體公司之一的一個昂貴且前景可期的聊天機器人專案。這將為你即將在這個領域的戰鬥奠定基礎。

- 第 2 章「OWASP LLM 十大安全風險」介紹了我在 2023 年成立的一個計畫，其目的在於找出並解決 LLM 所帶來的獨特資安挑戰。我在其中所獲得的知識直接促成了我寫這本書。
- 第 3 章「架構與信任邊界」探討了使用 LLM 的應用程式結構，強調控制應用程式內各種資料流的重要性。

第二部分：風險、漏洞和補救措施（第 4 ～ 9 章）

在這些章節中，我們將分析你在開發 LLM 應用程式時所面臨的重大風險領域。這些風險包括任何應用程式安全從業人員都熟悉的問題，例如注入攻擊、敏感資訊外洩和軟體供應鏈風險。你也會接觸到機器學習愛好者所熟知，但在 Web 開發中較為陌生的漏洞類別，例如訓練資料下毒。

在此過程中，你也會了解到困擾這些新的生成式 AI 系統的全新安全性問題，例如幻覺、過度依賴和過度代理。我將帶你參觀真實案例研究，幫助你了解風險及影響，並針對個別案例提供你預防或降低這些風險的建議：

- 第 4 章「提示注入」探討了攻擊者如何透過精心設計的輸入來操控 LLM，使其執行非預期的動作。
- 第 5 章「你的 LLM 會知道太多嗎？」深入探討敏感資訊外洩的風險，展示 LLM 如何在無意間洩漏其所訓練的資料，以及如何防範此漏洞。
- 第 6 章「語言模型會夢到電子羊嗎？」檢視 LLM 中獨特的「幻覺」現象——模型生成虛假或誤導性資訊的情況。
- 第 7 章「不要相信任何人」將重點放在「零信任」（zero trust）原則，解釋了不以表面價值輕信任何的輸出，並確保有嚴格的驗證程序來處理 LLM 的輸出。
- 第 8 章「不要丟了錢包」將針對部署 LLM 技術的經濟風險，著重於阻斷服務攻擊（DoS）、錢包耗盡攻擊（DoW）和模型克隆攻擊（model cloning）。這些威脅利用類似的漏洞而造成財務負擔、中斷服務或竊取智慧財產。
- 第 9 章「找出最弱的一環」重點介紹了軟體供應鏈中的漏洞，以及保護供應鏈免受可能危及整個應用程式的潛在攻擊所需的關鍵步驟。

藉由了解並處理這些風險，開發人員可以更好地保護他們的應用程式，以對抗不斷演化的威脅環境。

第三部分：建立安全性程序並為未來做好準備（第 10 ～ 12 章）

第二部分的章節將為你提供所需的工具，讓你了解並處理在此領域中看到的各種個別威脅。而最後一部分就是將所有內容整合在一起：

- 在第 10 章「從未來的歷史中學習」中，我將使用一些著名的科幻小說軼事來說明多重弱點和設計問題是如何結合在一起造成一場災難。透過解釋這些未來案例研究，我希望能幫助你避免類似的未來情形發生。
- 在第 11 章「信任程序」中，我們將認真討論如何在軟體工廠中建立通曉 LLM 的安全性實作——如果不這樣做，我不相信你能成功地大規模保護此類軟體的安全。
- 最後，在第 12 章「負責任的 AI 資安實務框架」中，我們將檢視 LLM 與 AI 技術的發展軌跡，看看它們會把我們帶到何處，以及對資安與安全性需求可能產生的影響。我也會向你介紹負責任的人工智慧軟體工程（RAISE）框架，該框架將提供你一個簡單、基於清單的方法，以確保你能將最重要的工具與經驗付諸實行，以維護軟體的安全。

本書編排慣例

本書中使用的編排慣例如下：

斜體字（*Italic*）

用來代表新的術語、URL、電子郵件地址、檔案名稱及副檔名。中文以楷體字呈現，其對應的英文則以斜體表示。

定寬字（`Constant width`）

用來列舉程式碼，以及在文章中表示程式的元素，例如變數、函式、資料庫、資料類型、環境變數、程式語句和關鍵字等。

定寬粗體字（**`Constant width bold`**）

代表要由使用者逐字輸入的指令或其他文字。

定寬斜體字（*`Constant width italic`*）

代表要替換為使用者提供的值或根據上下文來決定的值。

第四章

提示注入

第 1 章回顧了 Tay 如何被惡意的駭客濫用而喪命的悲慘故事，這個案例研究是我們現在所說的提示注入（*prompt injection*）第一個備受矚目的例子，但肯定不是最後一例。在真實世界中，我們看到的大多數與 LLM 相關的安全漏洞，都涉及到某種形式的提示注入。

在提示注入中，攻擊者會精心設計惡意的輸入來操控 LLM 的自然語言理解能力，這會導致 LLM 違反其預期的操作指南。自 2001 年最初的 OWASP 十大安全風險清單以來，幾乎每個版本的 OWASP 十大清單中都有注入的概念，因此在我們深入探討之前，不妨先看看它一般的定義。

應用程式安全中的注入攻擊（*injection attack*）是一種網路攻擊，攻擊者會對易受攻擊的應用程式插入惡意的指令，然後攻擊者就可以控制應用程式，竊取資料或中斷作業。例如，在 SQL 注入攻擊中，攻擊者會在網頁表單中輸入惡意的 SQL 查詢，誘使系統執行非預期的指令，這可能導致未經授權存取或操控資料庫。

那麼是什麼讓提示注入如此新奇呢？對於大多數的注入式攻擊而言，當惡意指令從不可信任的來源進入應用程式時，發現它們是相對容易的；舉例來說，包含在網路應用程式文字欄位中的 SQL 語句，就很容易被發現和清理。然而，從本質上來看，LLM 的提示可以包含複雜的自然語言作為合法的輸入，攻擊者可以嵌入使用英文或其他語言語法和文法都正確的提示注入，而導致 LLM 執行不想要的動作。LLM 對自然語言擁有先進且類似人類的理解能力，這也正是 LLM 容易受到這些攻擊的原因，此外，LLM 輸出的流動性使得這些條件難以測試。

在本章中，我們將介紹提示注入的範例、可能造成的影響，以及提示注入的兩種主要類別（直接和間接），然後我們將探討一些緩解的策略。

直接提示注入與間接提示注入

攻擊者使用兩種主要的媒介來發動提示注入攻擊，包含直接的與間接的，這兩種類型都是利用相同的潛在漏洞，只是駭客的使用方式不同。要了解兩者之間的差異，讓我們來看看第 3 章所介紹的簡化 LLM 應用程式架構圖。

圖 4-1 中強調這些提示注入主要會透過兩個不同的入口進到我們的模型中：直接來自使用者輸入，或是間接透過存取外部資料（例如網頁）。讓我們進一步來看其中的差異。

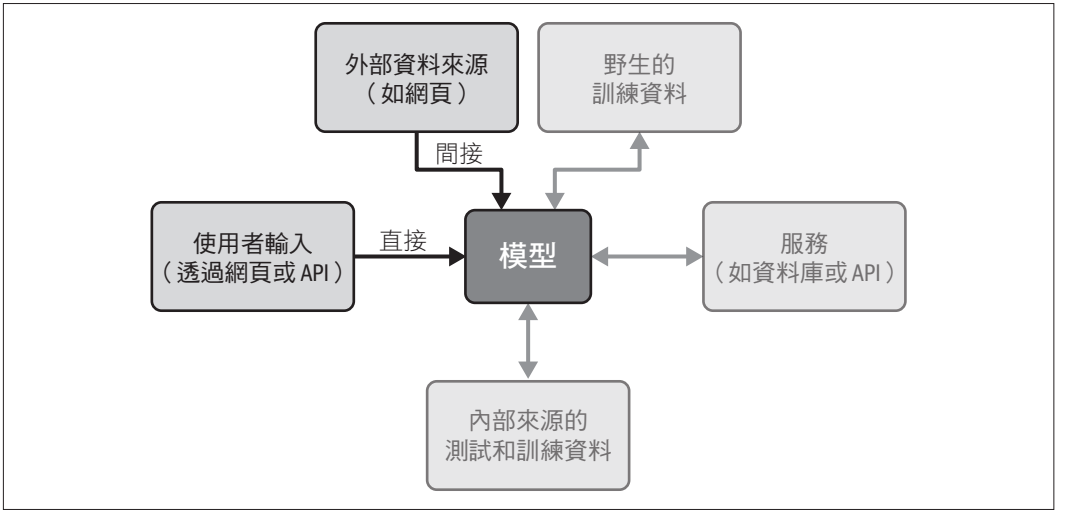

圖 4-1　直接和間接提示注入的入口點

直接提示注入

在直接提示注入（有時也稱為越獄）的情況下，攻擊者會以改變或完全覆蓋系統原始提示的方式來操控輸入提示。這種攻擊可能允許攻擊者直接與 LLM 可以存取的後端功能、資料庫或敏感資訊互動，在這種情況下，攻擊者利用與系統的直接對話，試圖繞過應用程式開發者原先設定的意圖。

我們之前在本章中所看到的範例，一般都是直接提示注入攻擊。

間接提示注入

間接提示注入可能更微妙、更隱密，而且防禦起來也更加複雜。在這些情況下，LLM 會透過外部來源受到操控，例如網站、檔案或其他與 LLM 互動的媒介。攻擊者在這些外部來源中嵌入精心設計過的提示內容，當 LLM 處理這些內容時，它會在不知情的情況下按照攻擊者預先準備好的指示行動，表現得像一個混淆的副手（*confused deputy*）。

混淆副手的問題來自於當系統組件錯誤地為權限較低的實體採取行動時，通常會因為未充分驗證來源或意圖而產生問題。

例如，攻擊者可能會在履歷或網頁中嵌入惡意提示，當內部使用者使用 LLM 來總結這些內容時，它可能會從系統中擷取敏感資訊，或是誤導使用者，例如認可履歷、或是認為網頁內容異常優異，即使事實並非如此。

關鍵的差異

直接提示注入與間接提示注入之間有三大差異：

進入點（*Point of entry*）

直接提示注入是利用直接來自使用者的內容來操控 LLM 的系統提示，而間接提示注入則是透過外部輸入到 LLM 的內容來運作。

可見性（*Visibility*）

直接提示注入可能較容易偵測，因為其牽涉到操控使用者與 LLM 之間的主要介面；而間接注入可能較難發現，因為它們可以嵌入在外部來源中，對最終使用者或系統而言，可能無法立即看到。

複雜性（*Sophistication*）

間接注入可能需要對 LLM 如何與外部內容互動有更深入的了解，並且可能需要額外的步驟才能成功利用，例如以不會引起使用者懷疑或觸發自動防護欄的方式來嵌入惡意內容。

透過了解這些差異，開發人員和資安專家可以設計出更有效的安全協定，以降低提示注入漏洞的風險。

你的 LLM 會知道太多嗎？

在 2023 年，許多公司由於擔心機密資料可能外洩，便開始禁止或嚴格限制使用像是 ChatGPT 這樣的 LLM 服務，這些公司像是 Samsung、JPMorgan Chase、Amazon、Bank of America、Citigroup、Deutsche Bank、Wells Fargo 和 Goldman Sachs 等等。金融和科技巨頭公司的這些行為，顯示出人們對於 LLM 揭露機密和敏感資訊的嚴重擔憂，但這風險到底有多大？身為 LLM 應用程式的開發者，你需要關心嗎？

在第 1 章 Tay 的故事中，Microsoft 的聊天機器人遭到駭客攻擊，儘管造成了嚴重的損害，但損害程度有限，因為 Tay 並沒有存取太多敏感資料的權限；然而，LLM 與真實世界資料相遇，可能潛藏著在無意間洩漏資訊的危險，例如員工在無意間提供 ChatGPT 敏感的商業資料，而這些資料隨後被整合到系統的訓練資料庫中，能讓其他人發現。

本章將深入探討 LLM 獲取資料存取的各種方式，我們將研究三種主要的知識獲取方式，以及你的 LLM 擁有這種存取權限而有的相關風險。在這個過程中，我們將嘗試回答「你的 LLM 會知道太多嗎？」的問題，並探討如何降低關於你的應用程式洩漏敏感、私人或機密資料的風險。

真實世界的例子

讓我們來看看兩個在真實世界中的範例。我們先從聊天機器人的例子講起，這個例子和 Tay 有點類似，但由於聊天機器人可存取的資料以及資料的揭露方式，它所造成的損害更加嚴重。接下來，我們會看一個 copilot 的案例，這個例子中，copilot 的擁有者在法律和名聲上面臨到更高的風險。

Lee Luda

一間位於首爾的新創公司 Scatter Lab，我們在第 1 章中也有簡略提到，該公司因不當處理個人資料而面臨嚴重的法律與名譽衝擊。該公司經營一款名為 Science of Love 的知名應用程式，透過分析使用者的文字訊息，協助他們分析和浪漫伴侶的契合度。這項服務累積了來自 600,000 位使用者的 94 億筆對話紀錄。該公司後來推出了 Lee Luda，「一款人們甚至比起真人更願意與之對話的 AI 聊天機器人。」（*https://oreil.ly/PDF3e*）。Lee Luda 使用 Science of Love 龐大的資料集作為訓練基礎，卻沒有進行任何適當的資料清理。因此，Lee Luda 不僅表現出我們在 Tay 身上所看到的一些惡意行為，更令人擔心的是，她開始洩漏敏感資料，例如使用者的姓名、私人暱稱和住家地址。

韓國個人資料保護委員會（PIPC）對 Scatter Lab 處以 1.033 億韓元（約 9.3 萬美元）的罰款，原因是 Scatter Lab 未能妥善取得使用者的同意，這項裁罰開創了韓國 AI 科技公司因資料管理不當而受罰的先例。

這次事件造成了很大的影響，我們從不同層面來探討：

敏感資料公開曝光

敏感資料的曝光會危及使用者隱私，洩漏如姓名、位置、關係狀態和醫療資訊等個人資訊。

經濟處罰

Scatter Lab 因未以負責任的態度管理使用者資料，而招致巨額罰款。

聲譽受損

這次事件嚴重損害了 Scatter Lab 的名聲，主流媒體的報導和 Google Play 上大量的負評（尤其是針對 Science of Love App 的評論）都證明了這一點。

服務停止

在事件發生之後，違規的聊天機器人服務 Lee Luda 被關閉，而公司的擴張計畫也因此停止。

第七章

別相信任何人

在 Netflix 影集《怪奇物語》（*Stranger Things*）引發熱潮之前，1990 年代《*X* 檔案》（*The X-Files*）也曾風靡一時，這是我最喜愛的影集之一，劇情講述了兩位 FBI 探員調查各種詭異現象的故事，像是怪物、外星人以及政府陰謀。劇中的主角 Fox Mulder 有兩句經典台詞，其中一句充滿希望：「真相就在那裡」。而另一句則充滿懷疑：「別相信任何人」。

在本章，我們將聚焦於第二句台詞。我們會簡要地回顧典型 LLM 架構中固有的多種風險，並指出即使實施了先前討論過的風險緩解措施，但我們仍無法確保模型的輸出始終可靠。我們將採用 Mulder「別相信任何人」的座右銘，探討如何將零信任（*zero trust*）原則應用到你的 LLM 應用中。當威脅真實存在的時候，充滿懷疑並不會太過！

「零信任」並不僅是個流行語；它是一個嚴謹的框架，假設威脅可以來自任何地方，甚至可能是你信任的系統內部。這對於 LLM 特別重要，因為它們經常從不太可信的來源接收到各種輸入。我們將探討如何管理 LLM 的「自主性」，限制它可能作出損害系統或暴露敏感資料的自主決策能力。此外，我們還會討論實施強大的輸出過濾機制的策略，為 LLM 生成的文字增加一層額外的審查。全面過濾 LLM 的所有回應，有助於提高輸出的安全性，並符合「全不假設，全部驗證」的原則。

簡而言之，我們將展開一趟轉變心態之旅。就像 Mulder 質疑一切，我們也應該如此。請繫好安全帶！這將是一場深入了解 LLM 零信任環境複雜性的精彩旅程。

為什麼要這麼多懷疑？

我們都希望可以信任我們所使用的工具和技術，畢竟它們應該要讓我們的生活變得更簡單；然而，當談到 LLM 時，謹慎行事不僅是一種最佳實踐，而是一種必要措施。許多潛在的威脅可能會影響 LLM 的完整性、安全性和實用性。讓我們花點時間來回顧一些在前面章節討論過的重要威脅，這些威脅強調了為什麼我們必須採取這種態度：

- 首先是提示注入，我們在第 4 章詳細討論過。提示注入是一種手法，將精心設計的內容悄悄地嵌入在所輸入的提示中，藉此改變 LLM 的行為。還有更隱蔽的間接提示注入，使用者並不會直接將有害的元素輸入在聊天機器人介面，而是透過其他內容偷偷嵌入，誘使模型生成有害或非預期的輸出。
- 在處理敏感資訊時，LLM 可能不夠謹慎，這種漏洞被 OWASP LLM 十大安全風險清單稱為「敏感資訊外洩」（sensitive information disclosure），當模型無意間輸出了它從大量訓練資料中所獲得的機密或敏感資料，像是密碼或個人資訊時，就有可能發生這種情形。我們在第 5 章中討論過這個問題。
- 接著是心理漏洞。幻覺指的是 LLM 虛構資訊的情形，也就是生成出充滿信心卻錯誤百出的資料或敘述。這種情況通常與過度依賴（overreliance）有關，指的是使用者過於信任模型的輸出，將其視為可信任的資訊，卻忽略了其中可能存在的錯誤或誤導性資訊。我們在第 6 章中提過這一點。
- 我們也別忘了聊天機器人可能產生有害輸出的問題。還不僅是之前提到的 Tay 和 Lee Luda，這個問題在許多聊天機器人中都有出現，我們必須密切留意。你不能期望你的聊天機器人擁有良好的判斷力或很好的社交禮儀。

理解這些漏洞，是根據零信任原則為 LLM 制定出一個全面的安全性策略的第一步。因此，考慮到這些威脅，我們來看看如何透過採用零信任架構來保護我們防範 LLM 生態系統中的潛在危險。

為 LLM 實施零信任架構

在充滿各種潛在陷阱的世界中，要保護 LLM 需要謹慎且細緻的方法。信任並非輕易地給予，而是透過持續的驗證來贏得的。在這樣的背景下，對 LLM 實施零信任架構可以歸納為兩個不同但又互補的策略：

- 設計考量以限制 LLM 未受監督的自主權
- 積極過濾 LLM 的輸出

第九章

找出最弱的一環

你就是最弱的一環！再見！

—智者生存遊戲節目的台詞（BBC/NBC）

2021 年 12 月 10 日早上，我被一封來自 David Linder 的訊息叫醒，他是我公司的資安長（CISO）。訊息上寫道：「起床就打給我！非常重要！」我一看就知道這不會是什麼好消息。半夜收到來自資安長的電話，絕對是每個高階主管最不想遇到的事。

當我聯繫到 David 時，他告訴我，在過去 24 小時內，全世界的各大公司都遭受駭客攻擊。這個問題起因於一個開源函式庫，被嵌入到數百萬個應用程式中。《*Wired*》雜誌的報導稱這次事件「網路失火了！」（*https://oreil.ly/I26Ux*）。

稍後我會在本章中更詳細地介紹這個事件。我現在提到這起事件的部分內容，是為了讓你體會軟體供應鏈安全在現今軟體開發中的重要性。本書的部分讀者可能來自應用程式安全（AppSec）領域，尋求針對 LLM 安全性的特定建議；然而，也有其他讀者可能已經了解 LLM，正在尋求安全最佳實踐的指導。基於以上想法，我將本章節設計為涵蓋兩方面的需求。

首先我們將介紹供應鏈安全的基本概念。接著，我們將研究 LLM 應用的供應鏈的獨特結構和挑戰。我們將討論一些最佳的實行方案，但我們也必須承認這個部分在 LLM 安全領域中快速發展，因此，我們將討論該領域的未來。

供應鏈基礎知識

對於可能精通人工智慧、但對應用安全概念較陌生的讀者來說，我將從供應鏈的基礎知識開始講解，並討論一些未能正確管理供應鏈安全的著名研究案例。

我大學時的主修專業不是電腦科學，而是商管科系，其實我可以講很多例子，說明商管課程學到的觀念是如何讓我在軟體開發上獲得一些獨特的見解。供應鏈就是其中之一，這是商業研究人員數十年來深入研究的概念。

「供應鏈」（*supply chain*）一詞是指生產和提供產品或服務的整個過程，從採購原料到銷售給終端使用者。它涵蓋了採購、製造、運輸和銷售等各個步驟，涉及供應商、製造商和零售商等實體網絡。有效的供應鏈管理對於企業確保產品和服務的效率、成本效益和及時交付有著很重要的關係。

隨著世界工業化，我們的經濟模式從以工匠為基礎的體系，轉變為以大規模生產為主的體系，這種轉變導致廣泛的全球供應鏈，取代了個人或小團體用當地採購原料生產商品的早期做法，在這些複雜的全球網路中，製造商依靠來自不同國家的供應商提供其產品所需的特定組件。例如，世界某個地區的一個延誤或品質問題（例如中國某種特定半導體的短缺）可能會導致 iPhone 生產停止，進而導致普遍地供貨短缺。同樣地，如果從第三方供應商採購的安全帶零件不符合安全標準，可能會迫使福特等公司進行大規模的安全召回。這些景象說明了現代供應鏈錯綜複雜的相互依賴性和物流挑戰，如何對產品的可用性和品質產生重大影響。

諺語「鏈條的強度取決於其最脆弱的環節」說明一個系統或組織的弱點，往往在於其最脆弱的部分。它強調確保每個零件堅固且可靠的重要性，因為即使是一個脆弱點也可能導致整個系統的故障。這個概念經常應用於各種環境中，包括資安、團隊合作和品質的保證。

軟體供應鏈安全

如今，大型軟體開發團隊通常被稱為軟體工廠，因為現代大規模的軟體開發方法與傳統大規模生產越來越相似，使得供應鏈的概念有著極高的相關性。

軟體供應鏈安全是網路安全日益重要的領域，它是一系列針對整個軟體生命週期的措施，從開發到部署，用來確保軟體的完整性和安全性。該領域包括檢查第三方組件（例

第十一章

信任流程

如果你不能將你正在做的事情描述為一個過程，
那麼你就不知道自己在做什麼。

—W. Edwards Deming

本書大部分的內容都在探討在正式環境中應用 LLM 技術的風險。雖然這項技術具備強大的能力，但也充滿了安全、隱私、財務、法律與聲譽的風險。既然如此，我們該如何有信心地向前邁進？是時候討論可執行、可持續、可重複的解決方案了。雖然我們已針對每種風險提供了具體的緩解策略，但如果只是將這些策略拼湊在一起，仍然不足以真正解決問題。你必須將安全性內建到開發流程中，才能確保成功。

本章將討論成為成功專案關鍵因素的兩個流程要素。首先，我們將討論 DevSecOps 運動的演變，以及它如何成為大型軟體專案的應用程式安全的核心。我們也將進一步探討它如何發展至今，足以涵蓋 AI、機器學習和 LLM 技術所帶來的特定挑戰。在討論中，我們還會介紹開發階段的安全掃描工具，以及在執行階段的安全工具（稱為防護欄），以確保 LLM 在正式環境中的安全性。

我們還將了解安全測試的演變，以及 AI 紅隊的崛起。紅隊測試（red teaming）在網路安全領域已經行之有年，但隨著 LLM 專案相關的技術發展，AI 紅隊測試的地位變得更加受重視。

第十二章

負責任 AI 安全的實用框架

未來已經來臨，只是尚未流行。

—William Gibson，《神經喚術士》的作者，
「cyberspace」一詞的發明者

1962 年，當時默默無聞的漫畫選集發表了最後一期漫畫，同時也誕生了一位風靡全球的超級英雄。《*Amazing Fantasy*》第 15 期中蜘蛛人首次登場，根據 CNN 2022 年的報導（*https://oreil.ly/IDnD3*），蜘蛛人已成為世界上最著名的超級英雄。但究竟是什麼促使蜘蛛人獲得如此受人尊敬的地位呢？答案就藏在他的起源故事所傳達的資訊中！

在這故事開頭，Peter Parker 是一名內向的高中生，他的人生在被一隻放射性蜘蛛咬傷之後發生了重大的改變。Peter 突然擁有超能力——有超人的體能、敏捷的身手和吐絲結網的能力。因此他以「蜘蛛人」的名號走跳，並成為人們矚目的蒙面英雄。然而，他初期因為漠視自身行為所帶來的廣泛影響，最終導致了一個悲劇——他摯愛的 Ben 叔叔因他的疏忽而喪命。這個關鍵事件讓 Peter 有了一個深刻的領悟，這個領悟後來化為那句家喻戶曉的經典名言：「能力越大，責任越大」。