淡江大學電機工程學系碩士班(積體電路與計算機系統組)

碩士論文

指導教授： 衛信文 博士

基於熵與機器學習之 SDN 多層 DDoS 緩解系統

Multilayer DDoS Mitigation System for SDN based on Entropy and Machine Learning

研究生：廖昱先撰

中華民國 111 年 6 月

致謝

我要感謝衛信文教授與李維聰教授，每當我有疑問時都非常熱心的指導，讓我培養了解決問題的能力。在面臨困難時，也是多虧了教授提出的建議，讓我度過了許多難關。我也要感謝教授在論文的題目上，提供了許多方向，讓我能從論文題目的選擇障礙中找到解答。

我在實驗室的生活影響了我許多，也讓我學會了待人處事的態度。多虧了實驗室同伴，在我需要幫忙時能挺身而出，讓我每次在遇到困難時能重新振作，讓我了解到我們是同在一艘船上的夥伴。當我在學習上陷入泥淖時，也是因為同學們的鼓勵，我才有能再次站起來面對困難的勇氣。我要感謝我的同學們，因為有你們才有現在的我。

我要感謝我的家人，在我陷入困難時能作為我最強的後盾，在我難過的時候幫我加油打氣，除此之外，每當我停滯不前時，因為有你們幫我開導方向，我才能不斷的向前走，度過了波瀾萬丈的碩士生活。

論文名稱： 頁數：60

基於熵與機器學習之 SDN 多層 DDoS 緩解系統

校系(所)組別： 淡江大學電機工程學系碩士班積體電路與計算機系統組

畢業時間及提要別： 110 學年度第 2 學期 碩士學位論文提要

研究生：廖昱先 指導教授： 衛信文 博士

論文提要內容：

分散式阻斷服務攻擊(DDoS)是近年來越來越嚴重的問題之一，隨著目前網路流量的需求越來越大，對於伺服器的負荷能力也是很大的考驗，原本就十分吃緊的網路資源如果再面臨 DDoS 攻擊的威脅，將會使伺服器資源的使用情況更為嚴峻。因此，應對 DDoS 攻擊已成為目前相當重要的課題。過去二十年來，針對分散式阻斷服務攻擊的防護機制已經付出許多努力，除了偵測 DDoS 攻擊的準確度外，更重要的是快速且準確的防禦攻擊，才能維持良好的使用者體驗。

目前在 SDN 上相關的 DDoS 緩解系統研究，大多數僅著重在於 DDoS 攻擊偵測的精確度，而非整體系統效能上的考量，因此，本論文提出了「基於熵與機器學習之 SDN 多層 DDoS 緩解系統」，主要目的是提出一種多層的 DDoS 緩解系統的設計架構，在架構中，透過兩階段的 DDoS 緩解設計，使緩解系統達到快速的反應外，亦不失準確度的 DDoS 偵測計算。

在本文提出的多層緩解系統中，為了對 DDoS 攻擊做出快速的反應，我們設計了第一階段的「基於熵交換器分流架構」，在交換器中加入簡易的運算以提高應對流量的效率，透過計算出來的熵值給予不同的分級，根據分級採取不同的行動我們，評定為正常流量就將其導向的路徑指定為負載率較低的路徑，評定為較差流量就將其導向較差路徑或是直接丟棄。以此為緩解策略確保正常用戶能保有較好的使用體驗，同時將攻擊流量在網路層時第一時間作出反應，以面對瞬息萬變的網路攻擊。

除此之外，我們能結合機器學習模型，做為第二階段的 DDoS 攻擊偵測系統，經過第一階段 DDoS 攻擊防禦模組，「基於熵交換器分流架構」能將封包根據熵計算後分級並採去不同策略行動。接著進入第二階段的 DDoS 攻擊偵測系統。交換器會將各式數據提供給控制器，而控制器則將資料轉交給「機器學習緩解 DDoS 攻擊服務器」。使得系統可以藉由訓練好的機器學習模型，以達到更好的攻擊判斷效果。

根據先前研究，機器學習的攻擊判斷精準度已可達 90%~99%左右，換句話說，只要使用先前的研究模型就可以達到相同的判斷精準度，因此本論文的研究與實驗僅著重在第一

階段的結果呈現。而實驗結果顯明，在合適的熵值判斷考量與對應策略下，本論文第一階段的架構的確可以有效且快速的緩解 DDoS 攻擊。

關鍵詞:

軟體定義網路、分散式阻斷服務攻擊、緩解系統

表單編號 ATRX-Q03-001-FM030-03

Title of Thesis： Multilayer DDoS Mitigation System for SDN based on Entropy and Machine Learning Total pages: 60

Key word: Software-Defined Networking, Distributed Denial of Service, Mitigation System

Name of Institute: Department of Electrical and Computer Engineering, Tamkang University

Graduate date: JUNE, 2022 Degree conferred: Master of Engineering

Name of student: YU-SIAN LIAO 廖昱先 Advisor: Hsin-Wen Wei 衛信文 博士

Abstract:

Nowadays, with the increasing demand for network traffic and internet services, service providers are facing with great challenges in sever deployment and network environment improvement. However, the Distributed Denial of Service (DDoS) attack can make the severe situation worse. Therefore, how to deal with DDoS attack has become an important issue.

In the past, many studies have proposed some solutions to conquer the DDoS attack problem. However, most of the related DDoS mitigation systems based on SDN only focus on the accuracy of DDoS attack detection, rather than the overall system performance. Hence, the main purpose of this thesis is to propose a multi-layer DDoS mitigation system that considers the response time of mitigation system while maintaining the accuracy of attack detection. In the system, through the two-stage DDoS mitigation design, the mitigation system can achieve rapid response and accurate DDoS attack detection.

There are two-stage in the multi-layer DDoS mitigation system, in order to respond quickly to DDoS attacks, we designed a first-stage detector and handler called "Entropy based Traffic Exchanger". The basic concept of the traffic exchanger is to add simple operations to switches to improve the efficiency of dealing with normal traffic and malicious traffic. According to entropy value of incoming traffic, the traffic can fall into different grades, i.e., normal, suspicious, and malicious. And different actions are taken by the handler according to the grades of incoming traffic. The system will assign a route with lower load rate to the traffic with normal grade; assign a route with lower quality to the traffic with suspicious grade; drop packets of the traffic with malicious grade. With this mitigation strategy, the normal users can maintain a better user experience, and at the same time, the system can respond to the attack traffic at the link layer as soon as possible to face the ever-changing network attacks.

In addition, we include the machine learning based DDoS detection model" as the second-stage of the mitigation system. After the first-stage DDoS attack defense module, the "Entropy based Traffic Exchanger" can divide packets according to entropy calculations into different grades with different strategic actions. Then enter the second stage of the

DDoS attack detection system, the switches will provide various data to the controller, and the controller will forward the data to the machine learning based DDoS detector. So that the system can achieve a better DDoS attack detection through the well-trained machine learning model. According to previous research, the accuracy of machine learning based attack detection has reached about 90%~99%. In other words, the same accuracy can be achieved as long as the previous research model is used. Therefore, the research and experiments in this thesis only focus the first stage, and only results of the first stage are presented. The experimental results show that, under the consideration of entropy value and different strategic actions, the first stage of the mitigation system in this thesis can indeed effectively and quickly mitigate DDoS attacks.

目錄

圖目錄

表目錄

第一章 緒論

1.1 前言

近年來隨著人們的生活型態改變，人們在網路上花的時間越來越多，自從 2020 年疫情爆發以來，已經有越來越多人仰賴網路生活，無論是因為疫情須居家辦公，又或是居家影音娛樂使用增加，良好的網路使用體驗越來越重要。隨著人們越來越依賴網路，網路的流量也與日俱增，原本有限的網路資源逐漸入不敷出。更糟的是，近年來分散式阻斷網路攻擊(DDoS)所造成的危害日益嚴重，許多公司企業都身受其害，導致產生鉅額損失。DDoS 攻擊以大量且無用的封包流量佔用伺服器的資源，使伺服器端運算能力消耗殆盡，網路頻寬受到阻塞，阻礙一般客戶的使用。

在這個網路架構逐漸不敷使用的情況下，建立一個全新的管控機制已是相當重要的課題。近年來，軟體定義網路(SDN)是一個全新的網路架構，相比傳統型網路交換器與路由器，SDN 網路具有網路的可視性與集中控制的功能，更能應對雜亂無序的 DDoS 攻擊。軟體定義網路將控制層與資料層分開，利用 Openflow 協議讓控制層管理資料層，就能自訂規則生成 DDoS 攻擊防禦策略。

儘管 SDN 具有相當好應對 DDoS 攻擊的條件，但是如何使用 SDN 的特性建立合適的 DDoS 防禦系統，卻是一個棘手的難題，最大的困難點主要就在 DDoS 千變萬化的攻擊模式，例如 TCP SYN ，UDP 或 ICMP 泛洪攻擊，如何在特定的時間點找到對應的攻擊模式也是一個困難點。

1.2 動機與目的

根據 A10Networks[1]調查：DDoS 攻擊的次數有上升的趨勢，光是 A10 研究團隊追蹤到的 DDoS 攻擊武器就有 1540 萬件，不只如此，網路攻擊的報告也相比去年增加了超過 161%。今年最有名的例子，就是烏俄衝突升溫時，烏克蘭政府的網路及商業網際網路受到大規模的攻擊[1]。

根據 2020 ，Yebo Feng 等人所提出來的論文[2]，基於深度學習的應用層 DDoS 攻擊偵測已能達到 98.73%，其減緩效果是目前應用層 DDoS 攻擊減輕中具有指標性地位的。儘管目前針對 DDoS 攻擊辨識已經有相當的成果，但如何快速又不失精準度的判斷攻擊流量，仍是待解的難題。應用層作為網路中的上層，收到資訊打算抵擋時會經過一點延遲，儘管應用層辨識準確率相當高，但是面對 DDoS 攻擊仍需以簡單且快速的處理才能應變瞬間湧入的大量封包。

由於各封包會先經由基礎架構層，在此層進行判斷就能保有快速的反應。因此我們能在 SDN 基礎架構層中的交換器放置簡單的 DDoS 判斷功能，透過簡單且快速的判斷作為抵擋 DDoS 攻擊的第一階段判斷。同時為了保有應用層辨識帶來的高準確率，會導入高準確率的機器學習緩解作為第二階段攻擊判斷，結合兩階段的攻擊防禦，對於攻擊的應對想必會更加完善。

1.3 論文章節架構

本論文第二章將介紹 SDN 的背景相關研究，同時還會介紹 DDoS 攻擊種類與熵相關的介紹與研究，還有機器學習緩解 DDoS 的相關文獻探討。

第三章會介紹本論文使用的主要架構，基於熵與機器學習之 SDN 多層 DDoS 緩解系統，說明如何建立起兩階段的 DDoS 攻擊緩解系統。此外，本論文的重點在於建立起能與第二階段機器學習辨識結合的底層判斷機制，因此我們的重點會著重在第一階段的攻擊快速緩解部分。

第四章會展示本論文的實驗結果與分析，我們會以正常客戶端與攻擊客戶端交叉訪問伺服器，並記錄本系統對攻擊流量的緩解數據。證明本論文系統具有緩解效果。

第五章則提出總結與未來發展方向。

第二章　　相關研究與背景資料

本章節會探討到幾個面向，分別是 DDoS 攻擊防禦、SDN 網路、熵與機器學習，並介紹幾個以上領域的相關文獻。

2.1　DDoS

DDoS(distributed denial-of-service attack)攻擊，又稱分散式阻斷服務攻擊，是一種極具破壞力的網路服務攻擊，攻擊者對被害者發送大量無意義的封包，占用服務器的資源或壅塞網路頻寬，導致其他使用者無法正常訪問被攻擊的伺服器。DDoS 攻擊通常使用許多受到入侵的魁儡電腦作為跳板，並使其對被害者發送大量服務請求，以實現對服務的阻斷與妨害。攻擊者的目的通常可分為幾種。

(1) 妨礙企業服務

攻擊者可能會透過網路攻擊使大型企業伺服器受影響，並要求企業支付贖金。多數情況下，儘管只是短暫的服務中斷都會造成企業的鉅額損失，因此也有許多企業會選擇支付贖金。

(2) 阻止某用戶連結伺服器

DDoS 攻擊的特性可以使網路頻寬壅塞，因此某些 DDoS 攻擊工具提供攻擊者阻斷被害者對外連線的能力，此類攻擊目的在於阻止被攻

擊者存取某項服務，在某些競技遊戲上，有些人可能會使用 DDoS 工具防止對手贏得勝利。

(3) 攻擊國家機關

近年來各國政府受到 DDoS 攻擊的事件也層出不窮，攻擊者可能出於政治目的而對政府機關服務發起攻擊，例如烏俄衝突時，烏克蘭政府就受到來自俄羅斯的大量 DDoS 攻擊[3]。

DDoS 攻擊乍看攻擊行徑明顯，但是卻相當難以防護，其重點就在攻擊者通常使用合法的服務請求，因此服務提供商無法有效分辨正常客戶與攻擊發起者。

DDoS 攻擊的棘手之處不只於此，現今 DDoS 攻擊的種類五花八門，因此針對 DDoS 攻擊可能需要好幾種應對模式。Arbor Networks 根據 DDoS 攻擊，提出了以下三大分類[4]。

2.1.1 體積型的攻擊

體積型的攻擊是三大分類中最常見的，攻擊者會對被害者發送大量流量，使網路頻寬受到壅塞，導致使用者無法訪問服務，其中最常見的幾種包括: DNS 放大攻擊、NTP 放大攻擊(NTP Amplification)、UDP 洪水攻擊(UDP flood)、 TCP 洪水攻擊(TCP flood)等等。

2.1.2 TCP 狀態窮盡攻擊(TCP State-Exhaustion Attacks)

TCP 狀態窮盡攻擊與體積型的攻擊最大的差異在 TCP 狀態窮盡攻擊並非消耗頻寬，而是在於消耗伺服器的資源。例如攻擊伺服器與網站之間的中間設備使其崩潰。SYN 洪水攻擊(SYN flood)是一種常見的 TCP 狀態窮盡攻擊，攻擊者會不斷發送 SYN 封包使所有伺服器資源受到占用，使其他服務請求無法使用。

2.1.3 應用層的攻擊(Application-Layer Attacks)

應用層是傳統網路七層架構中的第七層，應用層的攻擊會透過較少的資源攻擊應用程式中的漏洞，並大量發出合法服務請求，使伺服器服務受影響。常見的攻擊方式例如：HTTP 洪水攻擊(HTTP flood)、慢讀攻擊(slow read)、慢速發布攻擊(slow post)等等。

2.2 SDN

SDN 是近年來提出的一種新型網路架構，與分為七層的傳統網路架構不同，SDN 網路將網路架構分為三層，分別是應用程式層(Application Layer)、控制層(Control Layer)、基礎架構層(Infrastructure Layer)，整體架構如圖 2.1 所示。

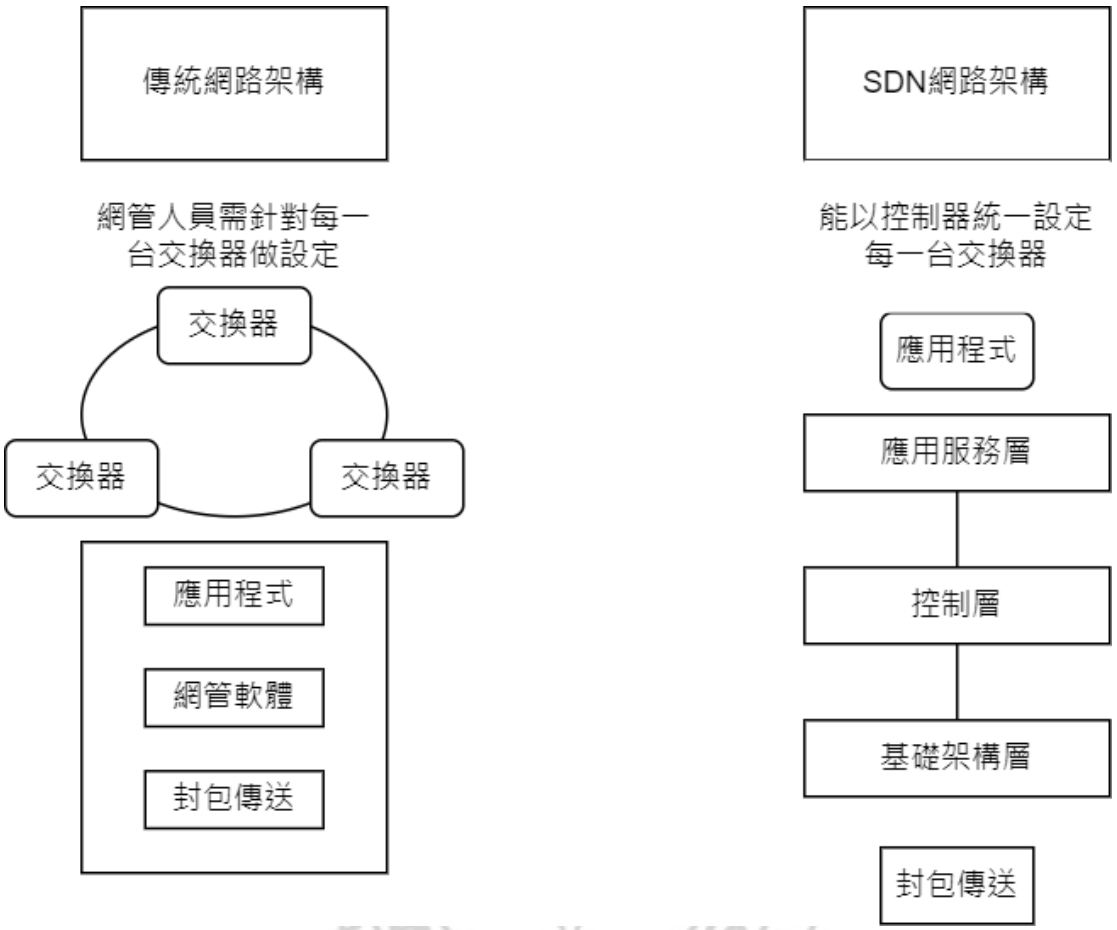

圖2.1　SDN 架構示意圖

2.2.1　應用程式層(Application Layer)

應用程式層是提供開發者開發應用服務的一層，開發者能自由定義對使用者提供何種服務。例如網路的可視化、網路狀態監控、網路拓樸結構、網路配置管理、網路安全策略等。

2.2.2 控制層(Control Layer)

控制層負責處理 SDN 網路中的基礎架構層，同時也是負責管理網路的大腦，網路管理人用可以透過連接不同 SDN 控制器來對網路進行不同的管理。SDN 控制器可以透過軟體自由定義控制基礎架構層的策略，因此我們能將 DDoS 攻擊的防禦策略寫在控制器裡，實現對基礎架構層的 DDoS 攻擊緩解。

由於控制層集中控管基礎架構層的特性 SDN 控制器有以下幾項優點：

(1) 容易部屬，直接由控制器對新交換器下達命令即可。

(2) SDN 控制器能直接控管封包流向，或是拒絕連接。

(3) 能使用軟體控制，能以較人性化的方式控管交換器。

控制層會透過北向 API 與應用程式溝通，並透過南向 API 與底層硬體溝通，因此可說是介於底層硬體與應用服務之間的溝通橋樑，目前常見的控制器功能包含：路由與存取控制、負載平衡、群撥及流量工作等等。北橋 API 目前尚無統一標準，南橋 API 則普遍使用由開放式網路基金會所制定的 OpenFlow 協定[5]。

2.2.3 基礎架構層(Infrastructure Layer)

基礎架構層負責處理封包的轉發，也能稱為資料層(Data plane)，此層會由交換器負責轉發網路的封包，交換器會接受控制器的控制，並執行控制器下達的指令。當交換器遇到不知如何處理的封包時，會詢問控制器並由控制器的策略執行。至於交換器在面臨封包時，會根據流程表(Flow table)來決定封包轉交的路線，如果封包的流向定義不在表中，則會由控制器根據策略判斷如何轉發。

SDN 將數據轉發與網路的控制分離，數據轉發就交給基礎架構層處理，並由控制層負責控制基礎架構層。由於基礎架構層單純負責數據的轉發，因此能使用通用的硬體。由於整個基礎架構層會交由控制層管理，因此對於網路的監視與控制，甚至是網路的部屬都會更加簡單。除此之外，SDN 的網路頻寬使用效率也較傳統網路高[6]。

根據剛才提到的 SDN 分層特性，我們能了解到，SDN 對於控制基礎架構層能使用軟體控制。因此我們能將基礎的 DDoS 攻擊辨識放在控制器裡，並配上應用層的機器學習 DDoS 攻擊辨識，完成雙層的 DDoS 攻擊辨識系統。

SDN 的應用範圍相當廣，例如影像串流的品質感知[7]，探討 QoS (Quality of Service)，影像串流服務品質保證。此外也有透過 NFV 技術虛擬化網路功能，探討串流服務的資源分配，以達到良好的用戶體驗 (QoE)[8]。

2.3 熵

本論文使用 Entropy(熵)作為判斷系統是否有誤的公式，在熱力學裡，熵用於度量一個系統不能作功的能量數，同時也能用於計算一個系統裡混亂的程度。在資訊領域哩，熵也能用來計算事件的訊息量，也能被理解為一個消息所佔用的重要程度。當熵值越小，那麼此事件的訊息含量也就越小，當越大時，就代表此事件能帶來許多訊息，同時也意味著此事件相較其他事件有著較不同的地方，帶來的資訊量很大[16]。

熵對於度量訊息大小的特性也能應用在檢測系統異常，如果今天我們將各來源客戶端的 IP 視作一個事件，那麼這些事件對整個網路系統的影響力，就能類比成這個事件對整個系統帶來的訊息量多寡，訊息量並不是指流入封包的大小，而是指這些大小的流量對比其他事件的驚訝程度。當 DDoS 攻擊流量遠遠大過正常流量時，那麼這個事件對比正常事件就是異

常的，帶來的訊息量很大。因此，本論文使用這個特性來找出網路環境是否遭受巨大流量的 DDoS 攻擊。

為了確保交換器的效能，我們會使用簡單的判斷確保應對 DDoS 攻擊的快速反應能力，此判斷能快速檢測系統是否遭受巨大流量的攻擊。至於其他類型的 DDoS 攻擊，我們未來會增加其他演算法予以對應，如果要增加其他類型會需要增加許多更加耗資源的機制，因此其他類型的攻擊可以利用第二階段的分類判別，來補足阻擋攻擊的效果。

2.4 機器學習

機器學習是人工智慧(AI)中的其中一個領域，機器學習會透過給予數據訓練得出辨識模型。之後有新的數據可以輸入進模型中，並得出模型的辨識結果。一般機器學習可分為三類，分別是監督式學習、非監督式學習與強化學習。

監督式學習最大的特徵，就是會把數據先標記後再去給機器學習，使機器發現錯誤並逐步的改善模型。假設今天需要訓練機器判斷這張圖片是狗還是貓，我們需要先給一些貓狗的圖片並告訴它哪張圖片是狗，哪張圖片是貓。機器就能訓練出一個模型，我們把圖片輸入進模型後就可以得出輸出，告訴我們這張輸入圖片裡面是狗或貓。

非監督式學習與監督式學習的差異，就在於非監督式學習不會把數據標記，他會直接把數據拿去給機器學習，並讓機器學習去找出特徵分類數據。因此非監督式學習無法告訴我們分類的數據是哪種，但是因為數據未標記，使非監督式學習的資料更具豐富性，很有可能透過機器找出我們沒找到的隱藏分類。

強化學習會透過分析不斷變動的環境，嘗試學習正確的決策，系統會在沒有人干預的情況下不斷訓練，並找出最佳的執行決策。強化學習會有一個 Agent (大腦)與 Environment (環境)，系統會不斷嘗試，並訓練出一個能適應 environment 的 agent。environment 會接收 agent 的 action(動作)並給出 reward(獎勵)與 observation(觀察)給 agent。並嘗試找出最高獎勵值的行動。

在文獻[9]中，提到了熵並搭配動態閥值設計實作出 DDoS 緩解系統，此研究以計算平均值與標準差動態獲得適合的流量與熵判斷閥值，除此之外也提供高流量下非攻擊流量 load balance 功能，降低環境傳輸延遲。該論文研究成果能有效阻擋高流量灌輸阻斷服務的類 DDoS 攻擊，並提供負載平衡環境負載。

文獻[10]提出了一個針對 SDN 控制器的 real time(即時)DDoS 攻擊檢測方法，該文計算窗口中的 50 個封包目的 IP 位址的 Renyi entropy，該

entropy 用於異常檢測模組，並於異常發生時獲取 Openflow flowtabel，分析 SDN 環境下的 DDoS 特徵。然後用 BiLSTM-RNN 對數據進行訓練，生成 BiLSTM 模型對實時流量進行分類。該論文結論說明能有效地對 DDoS 攻擊流量進行實時檢測。

文獻[2]提供了一個應用層的 DDoS 機器學習防禦判斷，該論文提出了一個基於強化學習的應用層 DDoS 攻擊防禦方法，該方法可以在應用層 DDoS 攻擊大量發生時緩解盡可能多的惡意請求，並在結論時說明該論文防禦方法能減緩 98.73%的惡意應用程序訊息。

文獻[11]與文獻[12]都使用軟體定義網路搭配 SVM 作為 DDoS 防禦的分類器，並且都有良好的準確率大約 90%~95%左右。文獻[13]使用決策樹來進行 DDoS 攻擊的分類，並使用 mininet 與 RYU 控制器作為拓樸工具與 SDN 控制器，該文結論提到與 SVM 相比，決策樹有更佳的性能。

文獻[14]使用多種機器學習算法進行了研究，文中比對二次判別分析(QDA)、高斯樸素貝葉斯(GNB)、K-近鄰演算法(k-NN)與決策樹(CART)，並說明儘管許多機器學習模型都有良好作用，但是決策樹的準確率與預測速度和訓練時間都有較佳表現，其中準確率可以達到 98%左右。

與先前的研究不同的是，本論文提出兩階段式的 DDoS 緩解系統，第一階段的目標是著重在利用交換器進行快速的計算與篩選，而非利用控

制器進行判斷與篩選，因此，相較於控制器能進行動態調整與較精確的檢測方式，交換器檢測的設計目的則在於快速與簡易為主，藉由這樣的設計方式，本論文第一階段可以防禦高流量的 DDoS 攻擊，而配上其他種類的機器學習模型，就能使防禦種類增加，偵測的精準度提升。換句話說，我們所提出的論文架構能結合快速檢測與優秀的機器學習模組，形成兩階段的快速防禦系統。

第三章 基於熵與機器學習之 SDN 多層 DDoS 緩解系統

目前關於 DDoS 的研究有許多了成果，有著重在偵測，也有著重在緩解系統上，著重在偵測 DDoS 攻擊的研究成果含有相當高的攻擊辨識率。著重在緩解系統上的研究也有許多有效的解決方案。但是，如何保有高辨識率，同時應對瞬息萬變的網路流量，並找出 DDoS 攻擊流量來自何處，仍是至今難解的問題。

而且具有高辨識率的 DDoS 偵測系統，必須通過網路架構中的應用層收集資料才得以運作，因此在收集資料並加以實施的過程中，勢必會產生處理上的延遲。相對的，找出攻擊並同時緩解系統，讓系統保有應對攻擊流量的同時，讓正常客戶的訪問不受影響，這種兼顧偵測與緩解的研究還有待著墨。

本論文的目的在於如何快速應對攻擊以及確保 DDoS 攻擊偵測的辨識率，在客戶擁有正常服務體驗的前提下，處理來自 DDoS 攻擊的威脅。我們使用 SDN 網路定義網路架構，此種類型的網路特徵就在可以使用軟體控制網路，我們能使用 SDN 網路來結合網路架構與緩解伺服器。 然而，一般而言，大多數在 SDN 上的 DDoS 研究針對控制器的判斷與解析，

相較之下，本論文則將簡易的判斷與解析放在交換器上。我們會在網路架構中的資料層就率先對流量進行初步的處理，此層由於位在處理封包流量的第一線，因此在此施展初步的攻擊處理便能有效且快速的應對攻擊流量。配合上機器學習攻擊辨識，能形成兩階段的攻擊防護。

此外，本論文使用 mininet 模擬整個 SDN 網路環境，mininet 是一個能模擬 SDN 網路環境的套件，使用 mininet 能迅速在模擬環境裡建立一個網路環境，方便實驗的進行與模擬。

3.1 系統架構

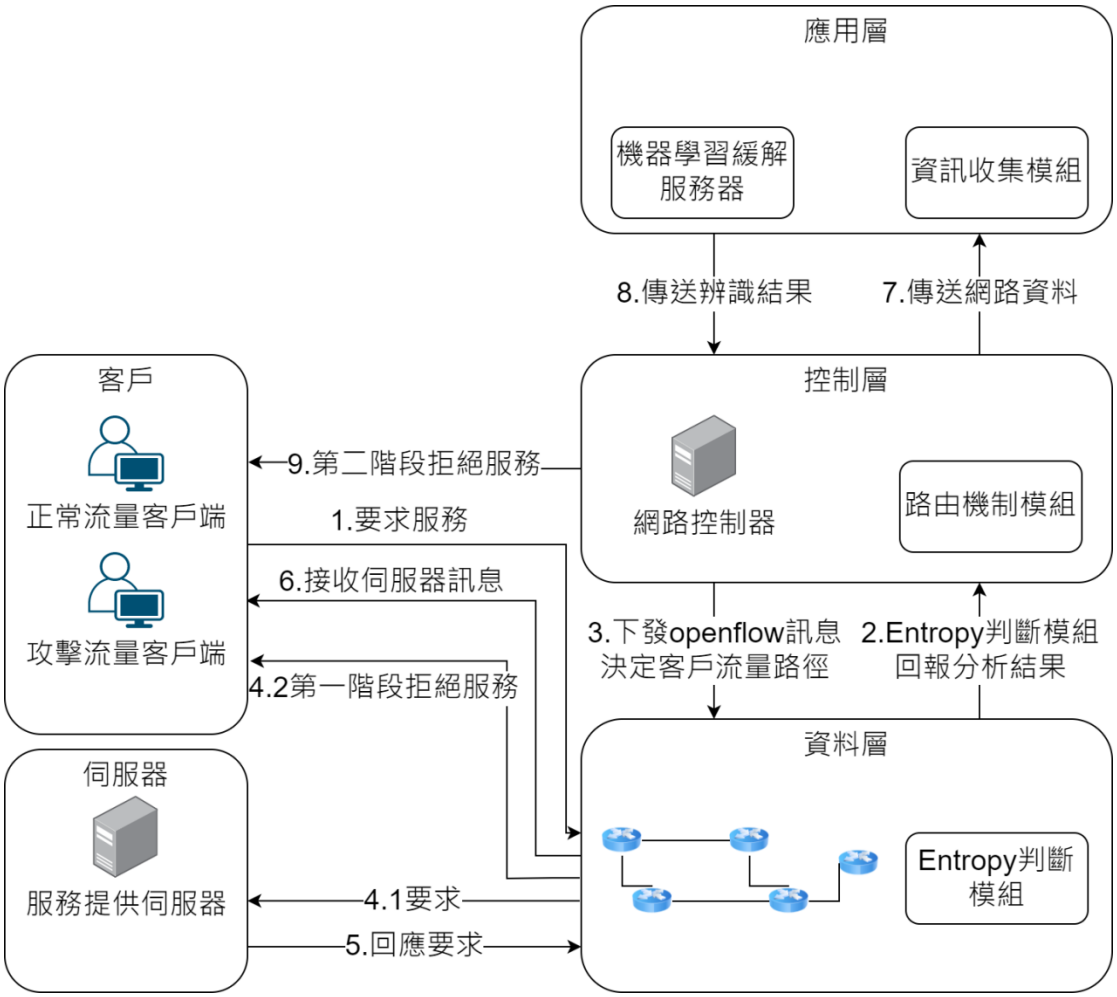

圖3.1 系統架構圖

上圖 3.1 為本論文提出的系統架構圖，根據 SDN 網路的特性可分為三層，分別為應用層、控制層與資料層。

(1) 應用層：負責處理應用程式服務，同時也是機器學習緩解服務器所在的一層。由於 DDoS 攻擊類型眾多，若想要達到眾多類型的偵測與防禦式，則必須要收集更多的資訊，若僅在網路層進行偵測除

了無法正確判斷之外，也加重了控制器的負擔，因此相較之下，機器學習緩解服務器更適合放置在應用層。

(2) 控制層：負責控制資料層，網路控制器所在的一層，此層有路由機制模組負責執行選路策略。

(3) 資料層：收到未知封包會詢問控制層，並根據控制層下達的訊息執行封包流向處理，客戶端發送服務請求會先經過資料層，待控制層回應後轉發客戶服務封包。在此層中，我們設計了熵判斷模組，負責計算流經資料層的封包熵，計算熵能快速偵測系統是否處在異常狀態，熵的公式計算會受到幾個因素影響：客戶端來源 IP 位址、接收端 IP 位址、客戶端請求封包數量等等。當我們偵測到網路正在被攻擊時，熵判斷模組會發現攻擊，並禁止攻擊來源的服務要求。禁止攻擊端服務要求能確保其他正常客戶不受攻擊影響。

除此之外還有正常流量客戶端、攻擊流量客戶端與服務提供伺服器，以及網路控制器，以下將會逐一解釋。

(1) 正常流量客戶端：負責向服務提供伺服器產生正常客戶流量的設備

(2) 攻擊流量客戶端：負責向服務提供伺服器產生攻擊流量的設備，其要求服務頻寬會遠遠超過正常客戶端的流量，如不處理將會占用

大量網路資源如傳輸頻寬或控制器運算能力，使其他正常設備運作受到影響。

(3) 服務提供伺服器：負責提供服務的伺服器，當客戶端訪問伺服器時，伺服器會回應客戶端要求。

(4) 網路控制器:

所有客戶端與服務提供伺服器都會使用到資料層的資源，如果資料層被攻擊後沒做任何處理，網路將產生嚴重的掉包情形，使用戶體驗受到影響。因此會需要有設備負責管理整體的網路情形，也就是控制層中的網路控制器。網路控制器是 SDN 網路中不可或缺的存在，能透過 Openflow 協定管理交換器封包流向或是禁止封包進入。

由於實體交換器的限制，我們將熵判斷模組與網路控制器結合。透過網路控制器模擬代替資料層的交換器進行熵判斷，以達到預期的設計成果。

當控制器收到熵判斷模組的分析結果後，會對客戶端進行一次分級，總共有三個種類，分別是：正常、疑似攻擊、攻擊。根據不同分級控制器會採取不同行動策略。

(1) 正常：當控制器判斷為正常時，給予分配最佳路徑以確保其使用體驗。

(2)疑似攻擊：當控制器判斷為疑似攻擊時，會給予分配最差路徑，如此一來便能不影響正常客戶端的服務，同時也不會導致誤判讓正常客戶體驗受到嚴重影響。

(3)攻擊：當控制器判斷為攻擊時，會直接丟棄來自攻擊客戶端的封包請求，確保資料層資源不受攻擊占用。

控制器會對交換器發送 Openflow 訊息，讓交換器決定封包走向，此時路由機制模組會根據不同分級給予不同的選路策略，將攻擊封包丟棄，並給予正常流量最佳路徑，給予疑似攻擊流量最差路徑。最後服務提供伺服器會收到客戶端要求並回應，至此便完成第一階段的攻擊辨識流程。

當第一階段攻擊辨識完成後，會進入第二階段，也就是機器學習緩解的部分。資料收集模組會定時向控制器要求資料層的資訊，並交由機器學習緩解服務器判斷是否有攻擊發生。我們可以使用 SVM 演算法或其他深度學習模型來替客戶端分類，當系統判斷有攻擊時，傳送辨識結果讓控制器把攻擊來源列入黑名單。之後來自攻擊來源的封包一律丟棄。

當系統執行完第一階段緩解後會開始第二階段的攻擊判斷，此時下一輪進來的封包會開始下一輪的第一階段並不斷輪迴，第二階段的系統會定時進行高準確率的攻擊辨識，以達成快速反應並同時具有高準確率的系統。

雖說架構上有兩階段，但是本論文研究將著重在第一階段，基於 Entropy 交換器分流架構的部分，因能接上簡易的 SVM、KNN 等分類器或深度學習分類器，讓兩階段結合且同時保有彼此的優點。

3.2 基於熵交換器分流架構

由於硬體限制，本論文提出的系統「基於熵交換器分流架構」，會以放置於控制器中的方式進行模擬，但是因為控制器會直接控管交換器，因此我們能預期其效果會與直接將系統放置於交換器相近。透過控制層中的控制器控管交換器形成第一層攻擊緩解系統，再接上應用層中的機器學習 DDoS 攻擊辨識服務器形成第二層攻擊判斷，就能達成兩階段的攻擊辨識，同時保有快速緩解與高準確率兩項優點。

接下來將先介紹「基於熵交換器分流架構」應對網路封包的運作流程，再逐一解釋分流架構中的不同模組功能。

3.2.1 分流架構流程

當封包進入系統後會先查看是 ARP 封包或是 IP 封包，因為兩種封包的功能與性質皆不相同，所以對於不同種類的封包將會給予不同的分流策略。

當進入系統的封包種類是 ARP 封包時，採用圖 3.2 的流程。

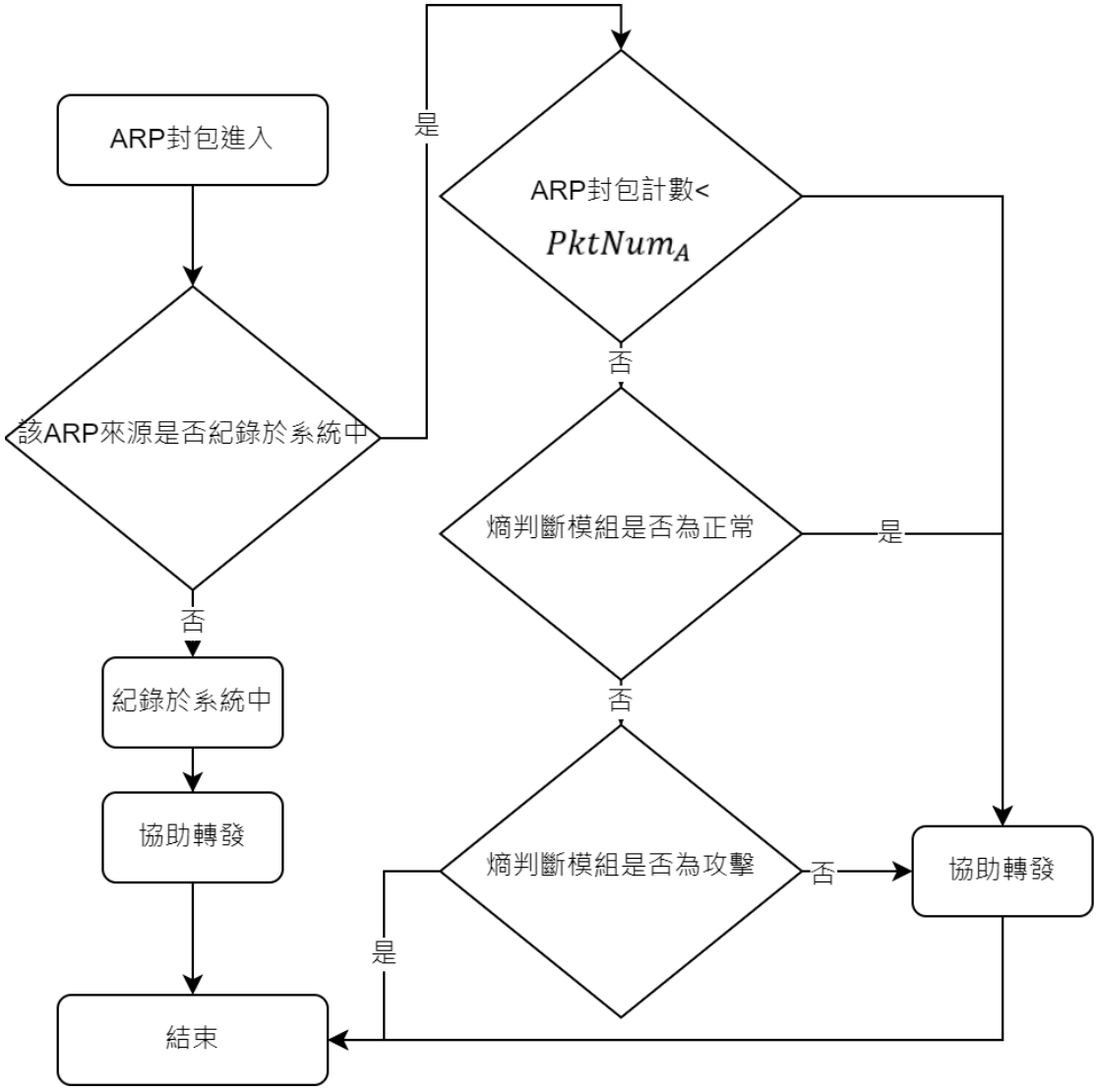

圖3.2　ARP 分流架構流程圖

當交換器收到 ARP 封包時，會先比對系統紀錄的 IP 來源表，根據有無會給予不同處理。收到 ARP 封包後，會確認該 ARP 封包欲訪問對象是否存在系統中，如不存在，熵判斷模組判斷並非攻擊或該封包進入次數小於$PktNum_A$後，將 ARP 封包送往全 host，並紀錄於系統中。如

果存在於系統中，並判斷並非攻擊且封包計數小於閾值$PktNum_A$，則會直接將其封包轉向目標 host。

之所以要加入封包計數，是為了讓系統不會將 ARP 封包誤判為攻擊，因為相對於其他種類的請求，ARP 封包發送的次數很少，熵判斷模組會認為該流量太少屬於異常狀態，因此需要多加一個條件來避免。雖然一般服務發送到的 ARP 封包很少，但是這種判斷方式能抵擋住大量偽造成 ARP 封包，企圖佔滿網路資源的攻擊，因此判斷 ARP 封包仍有其必要性。判斷封包數量的閾值$PktNum_A$用意在於過濾掉過量封包的請求，設定正常 ARP 封包不會到達的數量即可。

圖 3.3 為 IP 封包進行分流的流程，首先熵判斷模組會確認是否為攻擊，如果判斷為攻擊則直接丟棄封包。當判斷為疑似攻擊時，將封包導向最差路徑以確保不影響正常封包，且仍保有基本用戶體驗。判斷為正常時則給予最佳路徑，讓其保有最好體驗

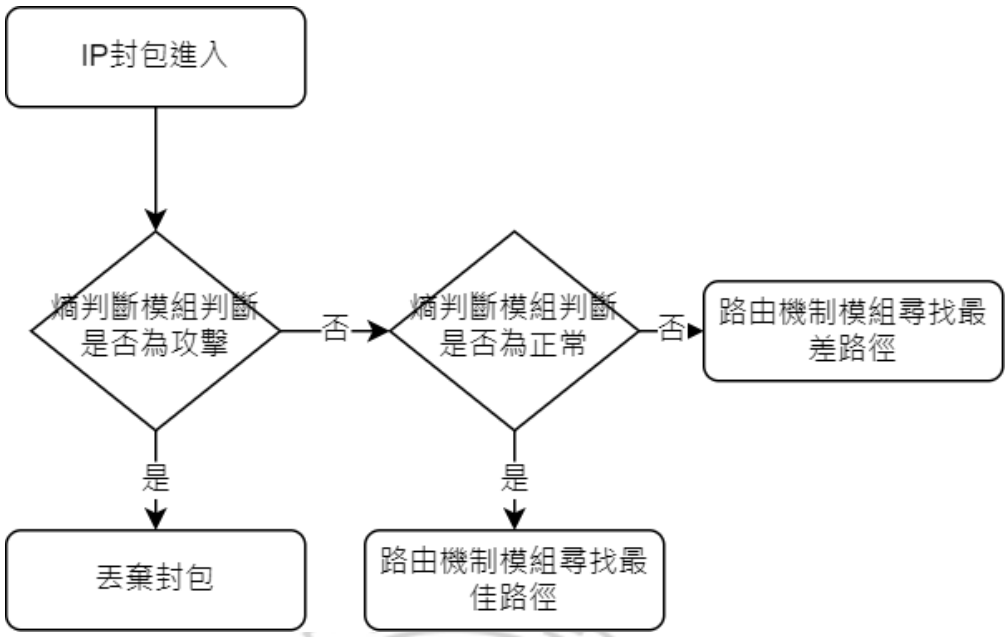

圖3.3　IP 分流架構流程圖

3.2.2 熵判斷模組

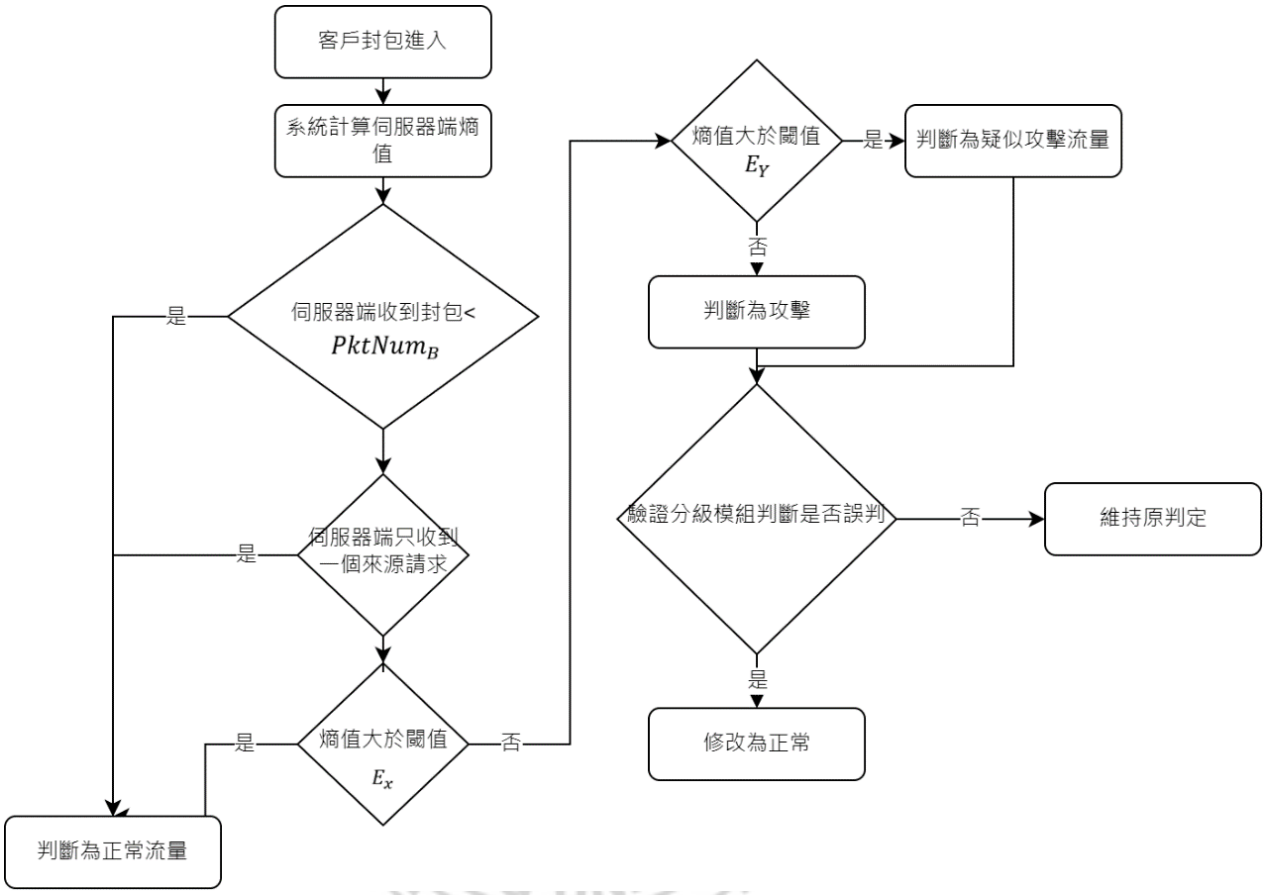

圖3.4　熵判斷模組流程圖

圖 3.4 為熵判斷模組流程圖，此模組的功能為判斷系統是否正遭受攻擊，判斷的熵公式如式 3.1 所示 [16]。

$$H = -\sum_{i=1}^{n} P_i \log_2 P_i \qquad (3.1)$$

其中H為欲計算的熵值，n為流量來源總數，P_i為來源i流向 server 的機率，其中，P_i的計算方法如式 3.2。f_i為來源端流向 server 的封包數量，f_t則為 server 端接收到的所有來源端的封包數量，如式 3.3 所示。

$$P_i = \frac{f_i}{f_t} \tag{3.2}$$

$$f_t = \sum_{i=1}^{n} f_i \tag{3.3}$$

計算完公式後，熵判斷模組會根據計算出來的結果給予不同的分級，當熵大於閾值E_x，或是 server 端收到封包數量少於，或是該 server 端指收到一個來源的封包，便會將分級定為正常。

當熵介於E_x與E_Y中間時，系統會將分級定為疑似攻擊，這個分級是為了區分出介於正常與攻擊之間的模糊地帶，當熵無法確定其為攻擊時，卻又達不到正常的標準，所訂的級別。

當熵小於E_Y時，系統會把分級定為攻擊，為了確保其他正常流量的運作，此分級的封包將會被丟棄。

區分分級的閾值E_x、E_Y會依據實驗來源數進行調整，閾值$PktNum_B$是為了防止熵公式誤判，所設定的條件，能確保 server 接收數量小於$PktNum_B$時必定判斷為正常。

判定完後會交由驗證分級模組確認該分級是否正確，有時會因為剛開始要求服務時，封包的數量太少導致熵認定現在有異常封包，但是因為數量少的封包並不是攻擊，此時必須將其改判定為正常。

3.2.3 驗證分級模組

本論文提出了三種方法的驗證分級模組，以應對不同封包訪問的情況，分別為圖 3.5 與圖 3.6 與圖 3.7。

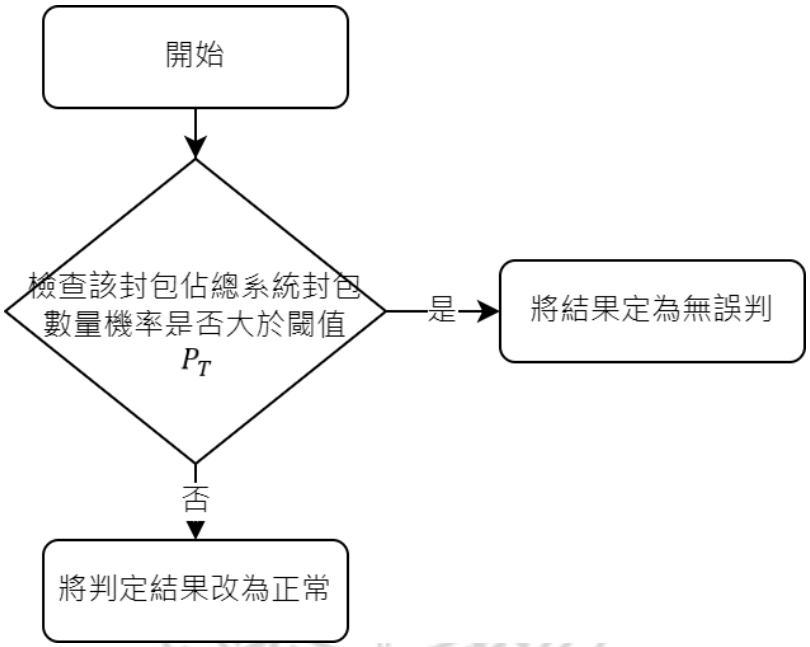

圖3.5　驗證分級模組流程圖(方法一)

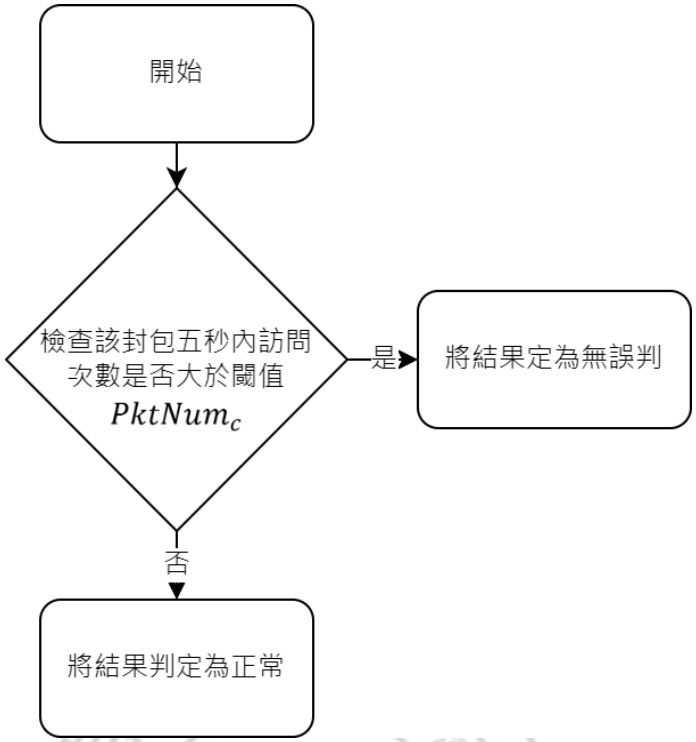

圖3.6　驗證分級模組流程圖(方法二)

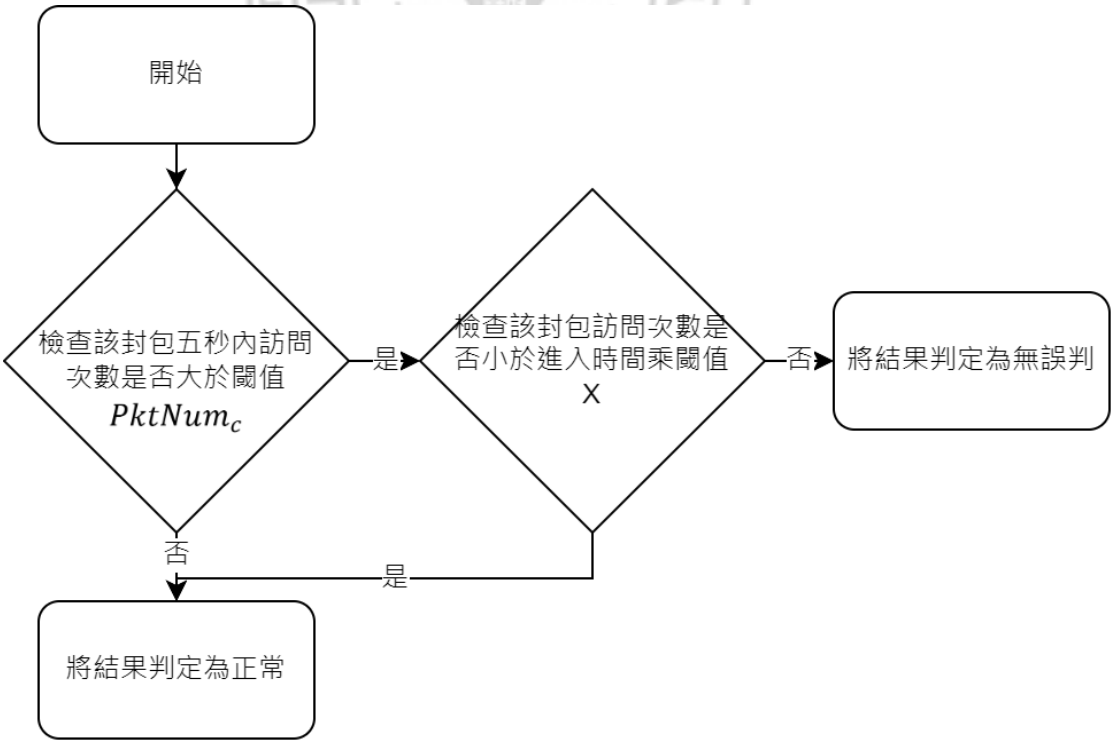

圖3.7　驗證分級模組流程圖(方法三)

根據方法的不同，驗證分級模組允許的流量也不同，當有封包被熵判斷模組認定為危險流量時，為了確保正常流量不被影響，驗證分級模組使用不同方式避免正常封包被誤判為危險封包。

首先是方法一。此方法會計算該封包占用整體流量的機率，檢查封包機率的用意在過濾因數量太少導致熵誤判為攻擊的流量。由於一開始進入的封包流量一定是最少的，熵公式檢測到此封包相對於其他封包與眾不同，因此熵計算出來的值就會偏低。此時驗證分級模組會檢查被判定為攻擊或疑似攻擊的封包機率是否高於閾值P_T，如果高於P_T則能確定為可疑流量，無誤判。如果小於P_T時將結果設定為誤判，交由熵判斷模組修正分級為正常流量。

方法二會每五秒計算一次封包訪問的次數，並將小於閾值$PktNum_c$的認定為正常流量。此種方式的好處在於可以考量到該流量請求的頻率，並將請求頻率過高的流量排除。

方法三則為方法二的延伸，系統會計算該封包來源的訪問時間，並根據時間秒數動態調整閾值 X，當該封包訪問次數小於閾值時則視為正常訪問。此方法的重點在於評估封包總共時間內的發送頻率，而不是只在五秒內，這個方法能排除因為其他攻擊流量被擋掉，導致正常流量在短時間內增多的情況。

3.2.4 路由機制模組

熵判斷模組結束後，將交由路由機制模組負責選擇路徑，網路的拓樸環境相當複雜，選擇路徑需要有專用的演算法處理，公式(3.4)為我們定義的路徑負載率公式 [7][8]。為一條路徑上所有節點與路徑的負載率最大值，負載率判斷方式為目前使用頻寬/總頻寬上限。

$$l(r) = \max\{l(v,e) | v \in r(s,t), e \in r(s,t)\} \quad (3.4)$$

(1) V：為所有網路環境上的節點集合

(2) E：為所有網路環境上的路徑集合

(3) $F(v)$：為所有網路服務經過節點v的集合，$v \in V$

(4) $F(e)$：為所有網路服務經過路徑e的集合，$e \in E$

(5) $C(v)$：為節點v容量(頻寬)，$v \in V$

(6) $B(e)$：為路徑e頻寬上限，$e \in E$

(7) $Bit(f)$：為每秒資料量，$f \in F(v), F(e)$

(8) $l(v)$：為v的負載率，$l(v) = \frac{\sum_{f \in F(v)} Bit(f)}{C(v)} * 100\%$，$v \in V$

(9) $l(e)$：為e的負載率，$l(e) = \frac{\sum_{f \in F(e)} Bit(e)}{B(e)} * 100\%$，$e \in E$

(10) $r(s,t)$：為來源節點 s 到目標節點 t 的路徑 r

$$r \in R(s,t)$$

(11) $l(r)$：為$r(s,t)$的最大負載率，負載率越低越佳。

選擇最佳或最差路徑時，須考慮到從客戶端到伺服器端的所有路徑。計算路徑負載率時，會選擇負載率最低的路徑為最佳路徑，選擇負載率最高的為最差路徑。

根據分級的不同，路由機制模組會採取不同行動，具體行動邏輯如下。

(1) 正常：選擇路徑負載率最低的路徑，為最佳路徑。

(2) 疑似攻擊：選擇路徑負載率最高的路徑，為最差路徑。

(3) 攻擊：不選擇路徑，丟棄封包

選擇完路徑後，依據選擇的路徑將負載率更新，讓之後的封包能以較新的負載率作為判斷依據，當系統判斷為攻擊時，移除攻擊所使用的負載率，並列入黑名單，列入黑名單後該來源發送的封包將不再接受，直接丟棄。整體流程如圖 3.8 所示。

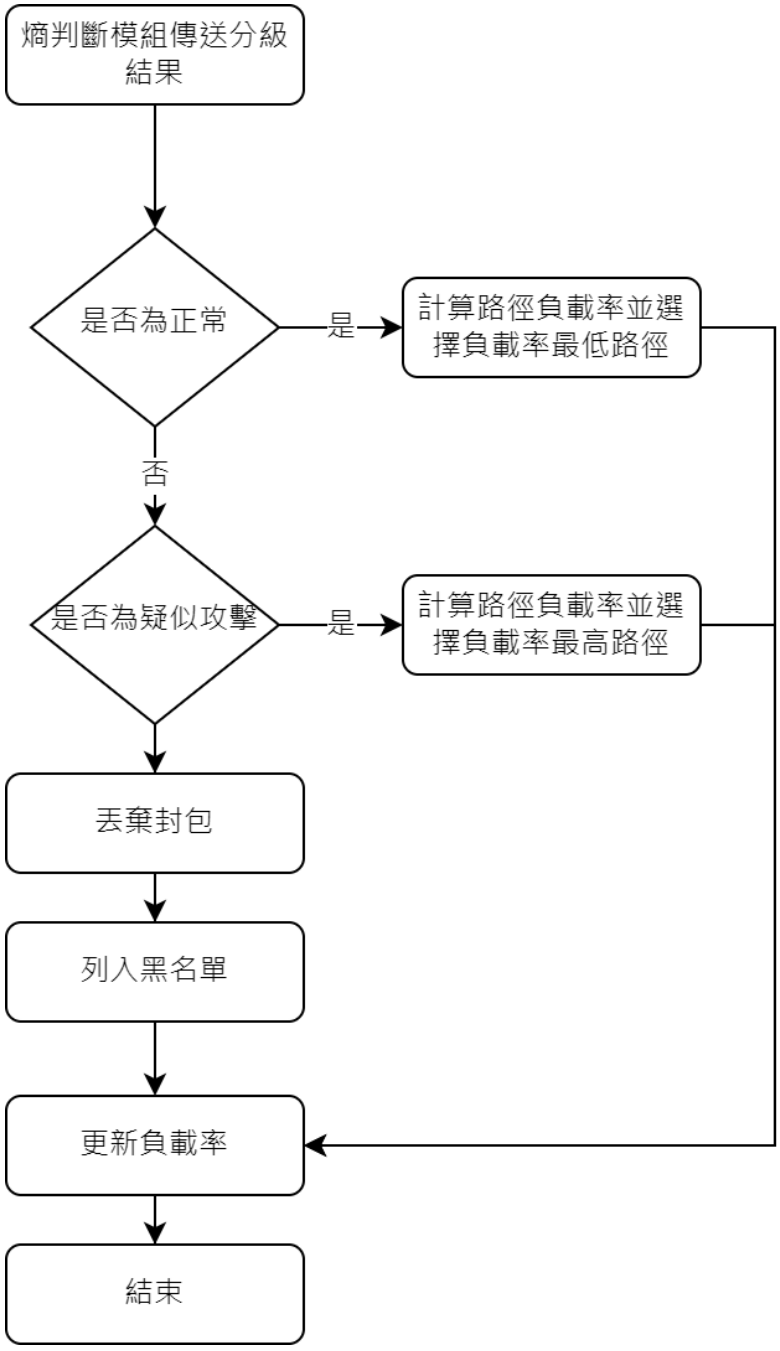

圖3.8　路由機制模組流程圖

3.3 系統流程圖

下圖 3.9 為第一階段熵判斷分流架構全部流程，此分流架構能在網路層率先進行第一階段的 DDoS 攻擊緩解，並接上第二階段機器學習緩解服務器，達成兩階段分層攻擊緩解。

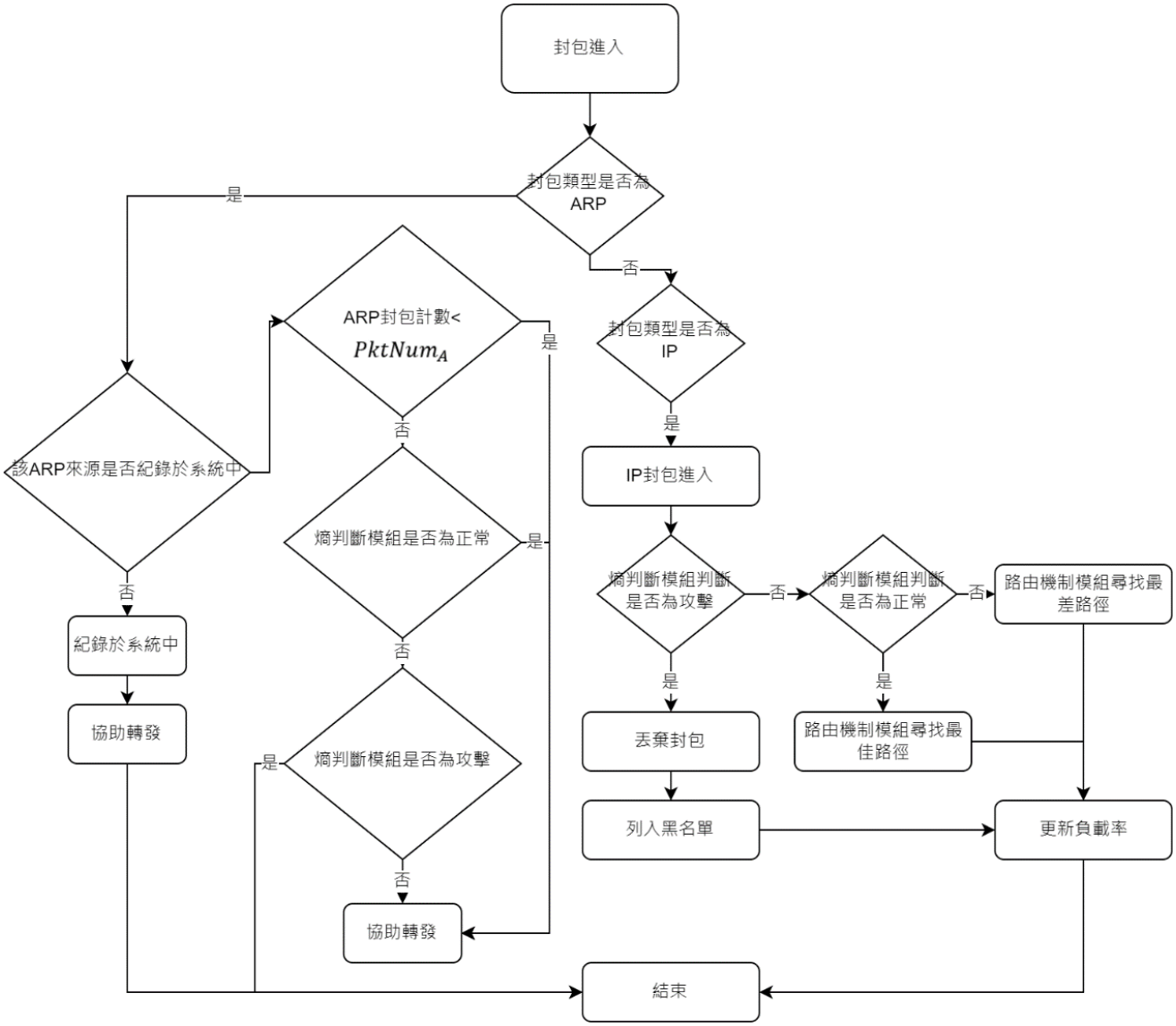

圖3.9 系統流程圖

第四章 實驗結果與效能分析

4.1 實驗環境與網路拓樸

本章節會展示基於熵交換器分流架構執行的模擬環境與測試結果。證明本論文提出的分流架構能有效找出 DDoS 攻擊並阻擋，確保第一階段攻擊辨識能確實運作。

本章節實驗將會以模擬進行，透過 mininet 套件模擬 SDN 網路拓樸環境，並連結 RYU 控制器[15]作為 SDN 網路控制器。控制器負責使用 Openflow 協定與 mininet 中的交換器溝通，並加以管理。因為硬體的限制，我們會將 entropy 交換器分流架構安裝進控制器裡操控交換器，期待能與在交換器上執行分流達成類似的效果。

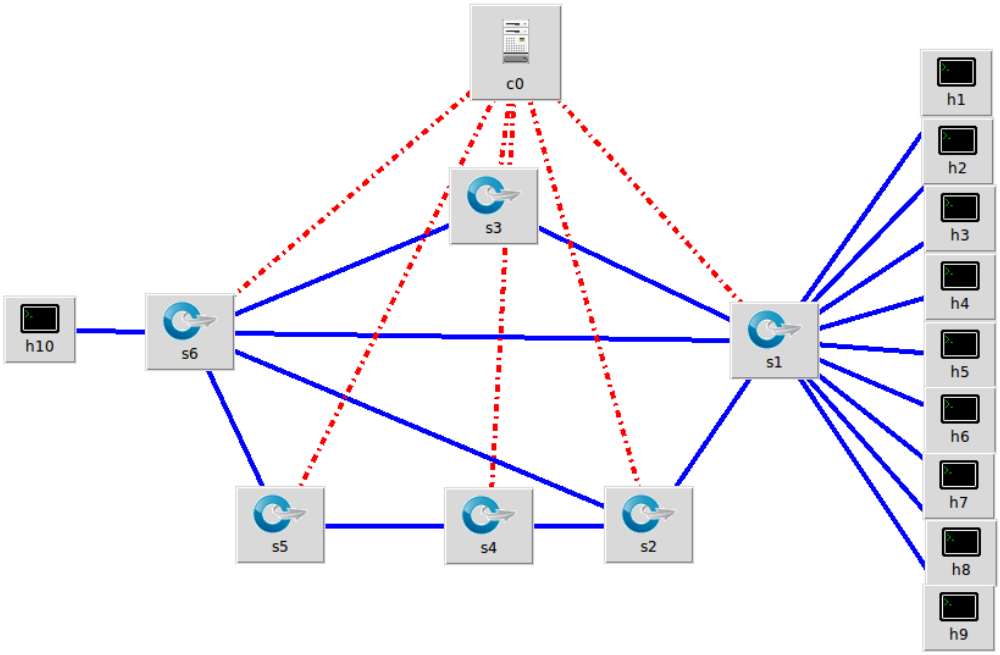

圖4.1　網路拓撲圖

圖 4.1 為模擬環境的網路拓樸圖，我們使用 1 個控制器 c0 控制 6 個交換器 s1~s6，並假設有十個 host 使用這個環境，並將 h10 設定為提供服務的伺服器，並與 s6 連結，h1~h9 作為客戶端與 s1 連結。路線的頻寬設定如下。

(1)h1~h9 與 s1 間的頻寬為 50M

(2)h10 與 s6 頻寬為 50M

(3)除此之外皆為 1.35M

之所以區分兩種是為了確保所有影響皆在 s1 到 s6 間，讓 host 的頻寬能完全運作於 s1，以實現攻擊情況的模擬。設定 1.35M 是為了縮小頻寬

以放大 DDoS 攻擊造成的影響。此外，本實驗會讓正常客戶與攻擊客戶依序進行 30 秒的 iperf 串流，並以不同頻寬訪問伺服器，攻擊客戶端為了模擬大流量傳輸，我們將其串流頻寬設定高於路線頻寬。客戶端的串流頻寬如下：

(1) 正常客戶端：頻寬設定為 0.25M

(2) 攻擊客戶端：頻寬設定為 5M

為了呈現系統應對大量攻擊的情境，我們會針對三種情況進行實驗，分別是客戶端分開出現、同時出現與循環出現。分開出現會讓所有客戶端錯開，同時出現則是同一時間點訪問伺服器，循環出現則會讓正常客戶端與攻擊客戶端進入的順序交叉，表 4.1 為客戶端的循環出現示意表。

表4.1　客戶循環出現示意表

	1	2	3	4	5	6	7	8	9
	Host1	Host2	Host3	Host4	Host5	Host6	Host7	Host8	Host9
類型	正常	攻擊	正常	攻擊	正常	攻擊	正常	攻擊	正常

4.2　實驗數據分析

在這裡我們將展示客戶進入時的遺失率與吞吐量，並記錄攻擊流量是否有被擋下，以及時間點。

首先我們紀錄了未使用熵分流架構的客戶端循環出現數據，以驗證本論文提出的架構確實具有緩解功能。如表 4.2、表 4.3 與表 4.4 所示。分別為分開進入、同時進入與循環進入三種情況。

表4.2　未使用熵分流架構客戶端分開出現示意表

	用戶種類	時間(s)	平均遺失率	平均吞吐量	
1	正常	0~30	0%	243.6	Kbits/sec
2	攻擊	0~30	18.31%	4670.9	Kbits/sec
3	正常	0~30	0%	243.6	Kbits/sec
4	攻擊	0~30	38%	4477.6	Kbits/sec
5	正常	0~30	100%	0.0	Kbits/sec
6	攻擊	0~30	100%	0.0	Kbits/sec
7	正常	0~30	100%	0.0	Kbits/sec
8	攻擊	0~30	100%	0.0	Kbits/sec
9	正常	0~30	100%	0.0	Kbits/sec

表4.3　未使用熵分流架構客戶端同時出現示意表

	用戶種類	時間(s)	平均遺失率	平均吞吐量	
1	正常	0~30	23%	91.1	Kbits/sec
2	攻擊	0~30	20.58%	700.7	Kbits/sec
3	正常	0~30	81%	49.0	Kbits/sec
4	攻擊	0~30	25%	1423.4	Kbits/sec
5	正常	0~30	56%	23.0	Kbits/sec
6	攻擊	0~30	84%	140.9	Kbits/sec
7	正常	0~30	100%	2.3	Kbits/sec
8	攻擊	0~30	100%	0.0	Kbits/sec
9	正常	0~30	100%	0.0	Kbits/sec

表4.4　未使用熵分流架構客戶端循環出現示意表

	用戶種類	時間(s)	平均遺失率	平均吞吐量	
1	正常	0~30	20%	362.1	Kbits/sec
2	攻擊	0~30	28.23%	3092.3	Kbits/sec
3	正常	0~30	86%	136.3	Kbits/sec
4	攻擊	0~30	57%	1884.4	Kbits/sec
5	正常	0~30	99%	21.1	Kbits/sec
6	攻擊	0~30	100%	0.0	Kbits/sec
7	正常	0~30	100%	0.0	Kbits/sec
8	攻擊	0~30	100%	0.0	Kbits/sec
9	正常	0~30	100%	0.0	Kbits/sec

從上表 4.2, 4.3, 4.4 的結果，我們可以很明顯發現，未使用熵分流架構的系統無法抵擋高流量攻擊，從 host5 開始，系統無法負荷來自攻擊客戶端的流量，並且開始出現遺失率高的現象，平均吞吐量也開始出現無法測量的情況，且無論是分開進入、同時進入與循環進入三種方式皆無法避免高遺失率。而圖 4.2、4.3、4.4 則為各種客戶端進入情形下，未使用熵分流架構下的客戶端吞吐量。

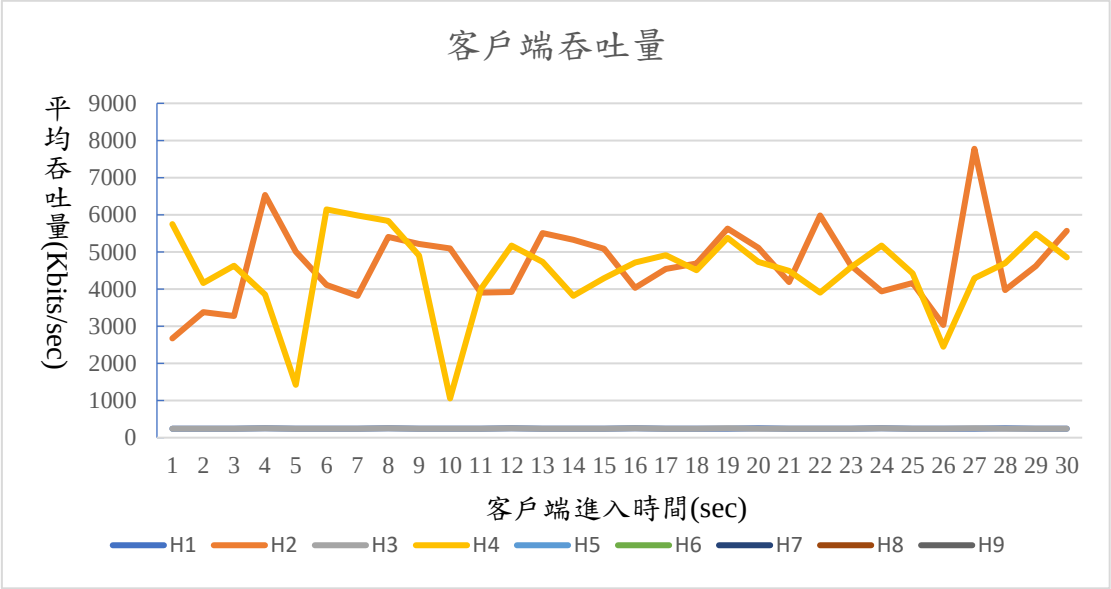

圖4.2　未使用熵分流架構客戶端分開出現客戶端吞吐量

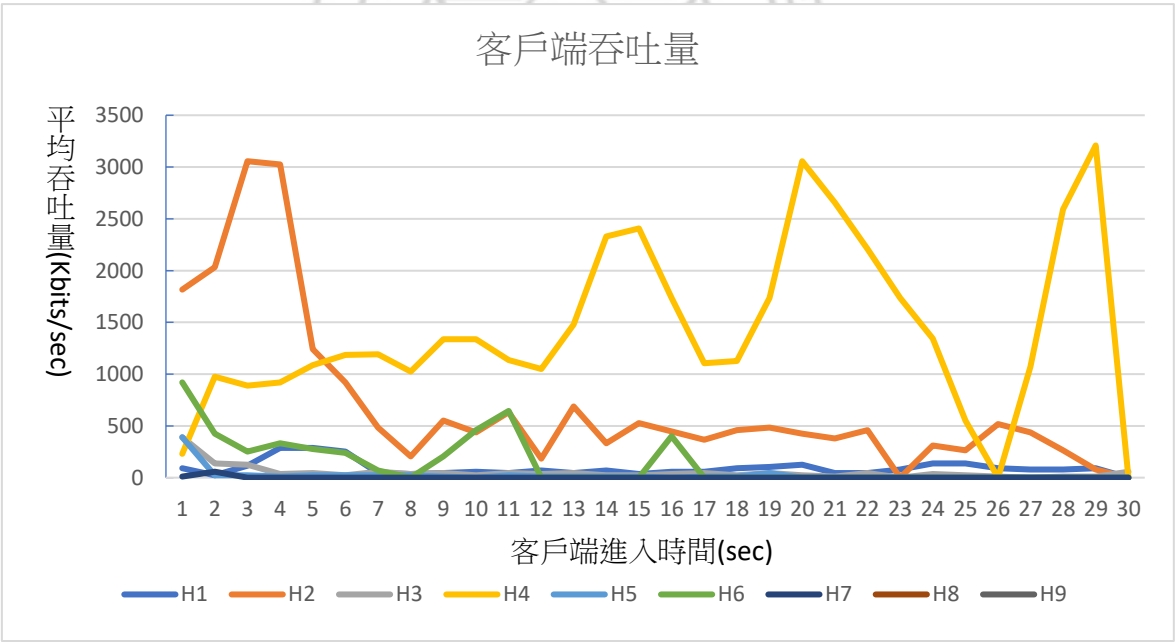

圖4.3　未使用熵分流架構客戶端同時出現客戶端吞吐量

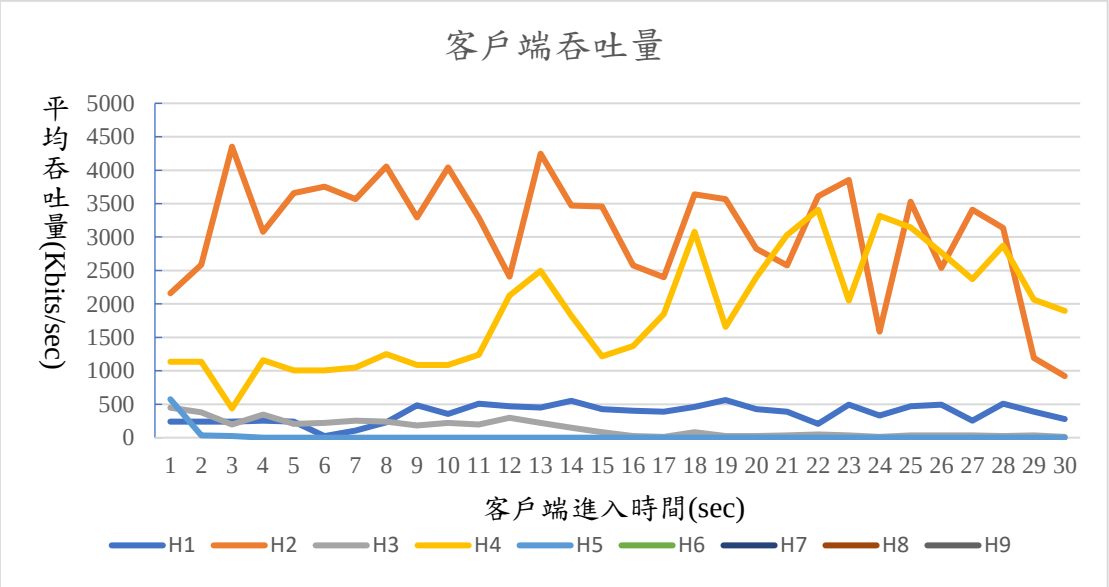

圖4.4　未使用熵分流架構客戶端循環出現客戶端吞吐量

從以上圖表可得知，如果不使用特殊的演算法加以控制，讓所有封包都給予分配最好的路徑，將會導致攻擊封包占用大量資源。從表中可以得知，H2 與 H4 兩個攻擊流量由於佔用了所有路徑頻寬資源，導致其他的客戶端吞吐量異常，有些客戶端甚至無法在 30 秒內送達封包至測試伺服器。

接著將展示，使用本論文提出的基於熵交換器分流架構產生的數據，我們根據驗證分級模組設計的不同分為三種方法，接下來我們會以數據逐一展示每種方法的優劣，並探討何種方法為最佳辦法。

方法一的數據如表 4.5 所示，此種方法會辨別封包占總封包次數的機率是否高於閾值P_T。

表4.5　方法一：客戶端分開出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	0%	243.6	Kbits/sec
2	攻擊	正常	0~5	6.88%	1990.9	Kbits/sec
		疑似攻擊	5~7	32%	2584.0	Kbits/sec
		攻擊	7~30	100%	0.0	Kbits/sec
3	正常	正常	0~30	0%	243.7	Kbits/sec
4	攻擊	正常	0~8	23%	261.8	Kbits/sec
		疑似攻擊	8~10	81%	4580.0	Kbits/sec
		攻擊	10~30	100%	0.0	Kbits/sec
5	正常	正常	0~30	0%	243.6	Kbits/sec
6	攻擊	正常	0~7	12%	3611.4	Kbits/sec
		疑似攻擊	7~10	27%	3954.7	Kbits/sec
		攻擊	10~30	100%	0.0	Kbits/sec
7	正常	正常	0~30	0%	243.6	Kbits/sec
8	攻擊	正常	0~11	38%	2208.4	Kbits/sec
		疑似攻擊	11~13	100%	3596.0	Kbits/sec
		攻擊	13~30	100%	0.0	Kbits/sec
9	正常	正常	0~30	0%	243.6	Kbits/sec

從表 4.5 可以看出使用方法一的情況下，攻擊客戶端如：H2 與 H4 與 H6 與 H8 皆有被辨別為攻擊流量。且其餘客戶端平均遺失率皆為 0%，與未使用本論文架構的數據相比較佳。接著是同時出現的情況如表 4.6 表示。我們能從表 4.6 得知除了 H9 所有客戶端皆被判斷為攻擊，導致正常客戶端無法獲得正常體驗，這是由於方法一使用了機率作為辨別是否為正常流量的判斷，導致在同時進入這種彼此流量相差不大的情況很容易

誤判，相比分開進入由於新客戶端剛進入時系統中的封包次數已經有一定數量，因此能保有正常功能。

表4.6　方法一：客戶端同時出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	攻擊	0~30	100%	0.0	Kbits/sec
2	攻擊	攻擊	0~30	100%	0.0	Kbits/sec
3	正常	攻擊	0~30	100%	0.0	Kbits/sec
4	攻擊	攻擊	0~30	100%	0.0	Kbits/sec
5	正常	攻擊	0~30	100%	0.0	Kbits/sec
6	攻擊	攻擊	0~30	100%	0.0	Kbits/sec
7	正常	攻擊	0~30	100%	0.0	Kbits/sec
8	攻擊	攻擊	0~30	100%	0.0	Kbits/sec
9	正常	正常	0~30	0%	273.5	Kbits/sec

表4.7　方法一：客戶端循環出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~5	0%	195.0	Kbits/sec
		攻擊	5~30	100%	0.0	Kbits/sec
2	攻擊	正常	0~8	40%	4339.0	Kbits/sec
		攻擊	8~30	95%	126.2	Kbits/sec
3	正常	正常	0~4	100%	0.0	Kbits/sec
		攻擊	4~30	100%	0.0	Kbits/sec
4	攻擊	正常	0~4	38%	1087.2	Kbits/sec
		攻擊	4~30	100%	0.0	Kbits/sec
5	正常	正常	0~6	33%	166.4	Kbits/sec
		攻擊	6~30	100%	0.0	Kbits/sec
6	攻擊	正常	0~11	47%	2877.8	Kbits/sec
		攻擊	11~30	100%	0.0	Kbits/sec
7	正常	正常	0~4	100%	0.0	Kbits/sec
		攻擊	4~30	100%	0.0	Kbits/sec

8	攻擊	正常	0~4	80%	2410.0	Kbits/sec
		攻擊	4~30	100%	0.0	Kbits/sec
9	正常	正常	0~30	3%	240.2	Kbits/sec

接著是方法一對循環出現的數據如表 4.7，可以看出與同時出現的數據皆把 H9 以外都辨識為攻擊，與同時出現的情形相似，機率對循環出現無法反映出是否為正常封包。接著是方法一的三種情況客戶端吞吐量。如圖 4.5、圖 4.6 與圖 4.7 所示。

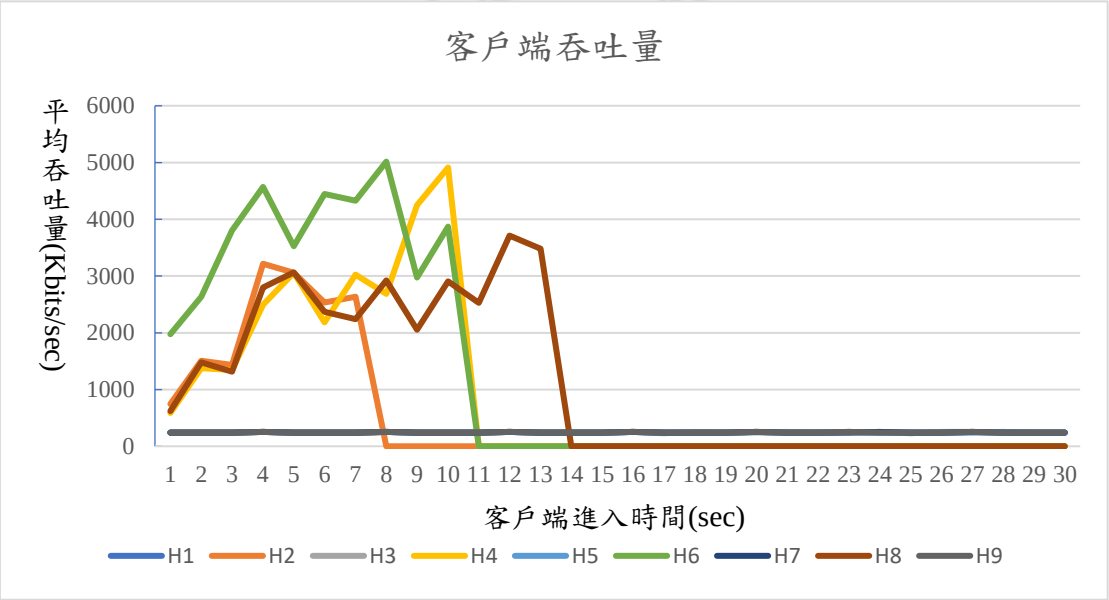

圖4.5　方法一分開出現客戶端吞吐量

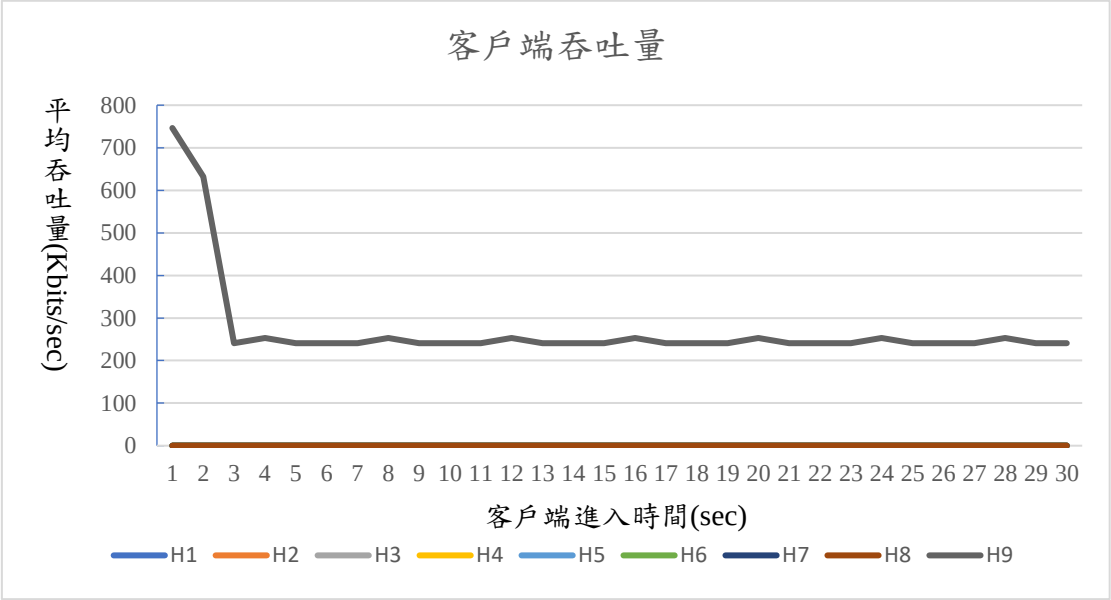

圖4.6　方法一同時出現客戶端吞吐量

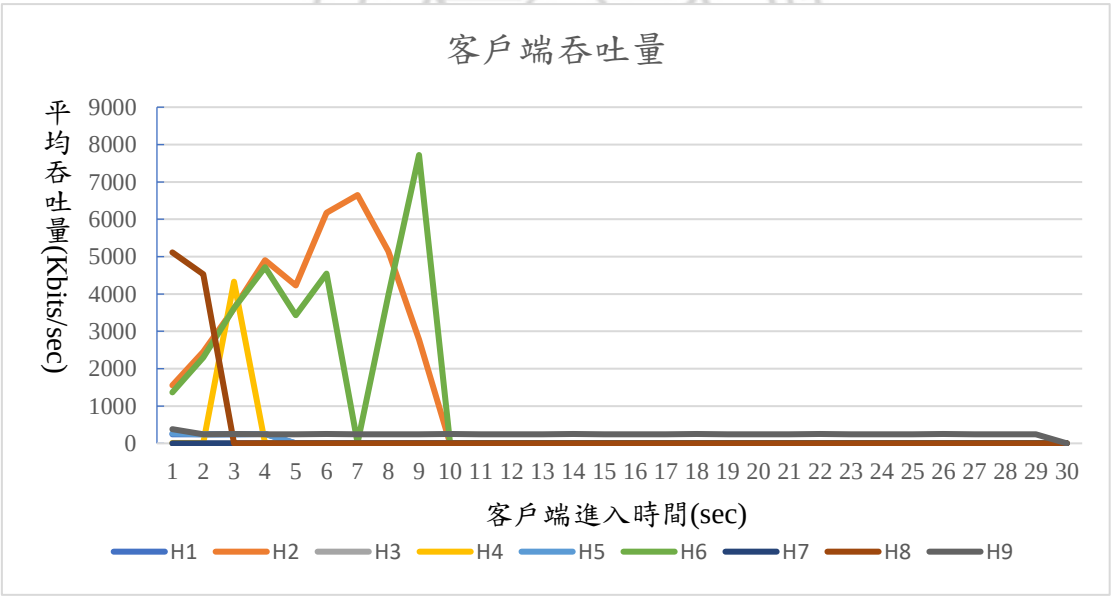

圖4.7　方法一循環出現客戶端吞吐量

從這三張圖可得知方法一在面對分開進入的情況能在十秒左右辨識出攻擊，同時進入的情況則會在新的封包進來時導致前面進來的封包機率太大而被誤判，循環進入也是類似原理，一旦有新客戶端服務請求就會導致舊的客戶端機率相較太大而誤判。

從數據來看方法一對分開進入以外的情況較為無力，但是仍有其優點所在，當系統執行一段時間後，各客戶端封包數量逐漸維持在一個比例，當此時有其中一個服務請求突然上升，就會使機率超過閾值而被判斷為攻擊，在此種情況下方法一也有使用的情況。

接著是方法二的數據，如表 4.8、表 4.9 與表 4.10 所示。

表4.8　方法二客戶端分開出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	0%	243.6	Kbits/sec
2	攻擊	正常	0~8	30.65%	2734.0	Kbits/sec
		攻擊	8~30	100%	0.0	Kbits/sec
3	正常	正常	0~30	0%	243.6	Kbits/sec
4	攻擊	正常	0~6	24%	2488.0	Kbits/sec
		疑似攻擊	6~8	77%	4308.0	Kbits/sec
		攻擊	8~30	100%	54.2	Kbits/sec
5	正常	正常	0~30	0%	243.6	Kbits/sec
6	攻擊	正常	0~6	19%	3658.7	Kbits/sec
		疑似攻擊	6~9	48%	3085.3	Kbits/sec
		攻擊	9~30	100%	0.0	Kbits/sec
7	正常	正常	0~30	0%	243.6	Kbits/sec
8	攻擊	正常	0~7	25%	2530.3	Kbits/sec
		疑似攻擊	7~9	51%	4248.0	Kbits/sec

		正常	9~10	26%	6152.0	Kbits/sec
		攻擊	10~30	100%	0.0	Kbits/sec
9	正常	正常	0~30	0%	243.6	Kbits/sec

表4.9　方法二客戶端同時出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	49%	173.8	Kbits/sec
2	攻擊	正常	0~3	31%	1226.7	Kbits/sec
		攻擊	3~30	96%	32.7	Kbits/sec
3	正常	正常	0~28	49%	801.4	Kbits/sec
		攻擊	28~30	100%	0.0	Kbits/sec
4	攻擊	正常	0~6	73%	1586.0	Kbits/sec
		攻擊	6~30	100%	8.6	Kbits/sec
5	正常	正常	0~30	58%	164.6	Kbits/sec
6	攻擊	正常	0~9	89%	1477.1	Kbits/sec
		攻擊	9~30	100%	4.4	Kbits/sec
7	正常	正常	0~30	74%	169.6	Kbits/sec
8	攻擊	正常	0~15	94%	741.5	Kbits/sec
		攻擊	15~30	100%	7.7	Kbits/sec
9	正常	正常	0~29	76%	410.5	Kbits/sec
		攻擊	29~30	100%	0.0	Kbits/sec

表4.10 方法二客戶端循環出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	17%	284.7	Kbits/sec
2	攻擊	正常	0~4	13%	1898.0	Kbits/sec
		攻擊	4~30	97%	110.9	Kbits/sec
3	正常	正常	0~14	44%	1166.1	Kbits/sec
		攻擊	14~30	100%	125.7	Kbits/sec
4	攻擊	正常	0~4	47%	2090.0	Kbits/sec
		攻擊	4~30	99%	39.2	Kbits/sec
5	正常	正常	0~19	46%	862.4	Kbits/sec
		攻擊	19~30	100%	0.0	Kbits/sec

6	攻擊	正常	0~5	73%	2102.4	Kbits/sec
		疑似攻擊	5~7	50%	3928.0	Kbits/sec
		正常	7~9	55%	2204.0	Kbits/sec
		攻擊	9~30	100%	0.0	Kbits/sec
7	正常	正常	0~30	75%	340.2	Kbits/sec
8	攻擊	正常	0~7	62%	1093.3	Kbits/sec
		攻擊	7~30	100%	11.0	Kbits/sec
9	正常	正常	0~23	81%	1708.6	Kbits/sec
			23~24	100%	0.0	
			24~30	100%	0.0	

從以上三個表中可看出，相較於方法一，在分開進入的情況下也能辨識出攻擊的流量，且同時出現與循環出現的數據則能較佳的找出攻擊的客戶端。圖 4.8、圖 4.9 與圖 4.10 為方法二的客戶端吞吐量表。

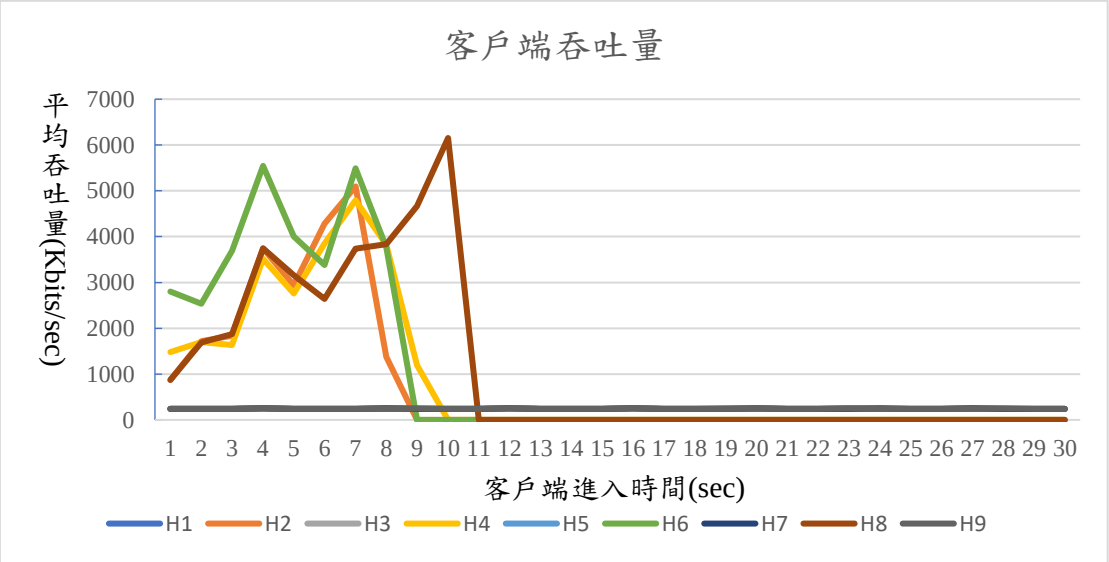

圖4.8　方法二分開出現客戶端吞吐量

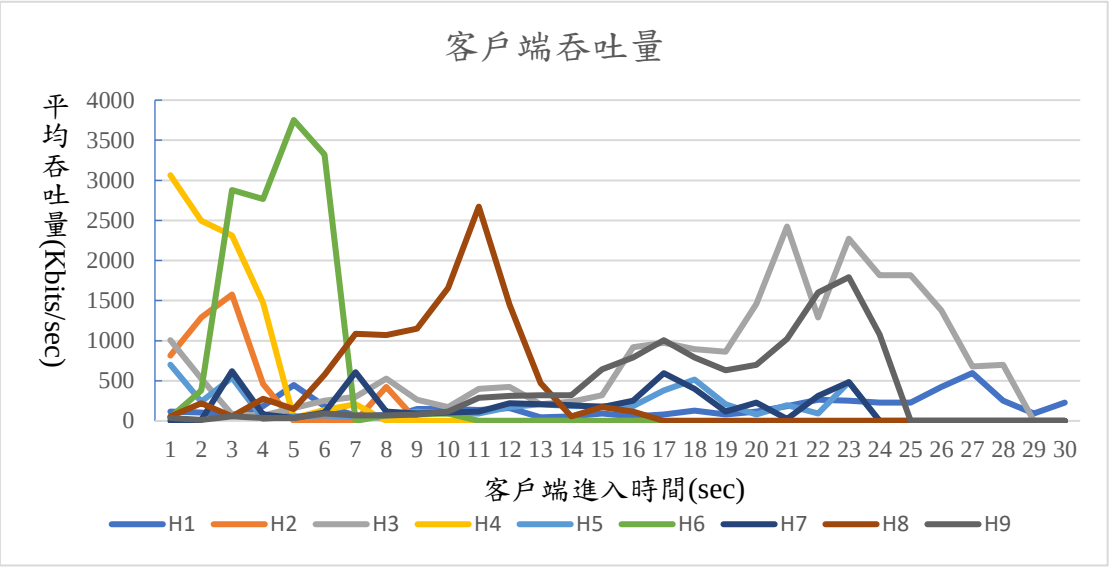

圖4.9　方法二同時出現客戶端吞吐量

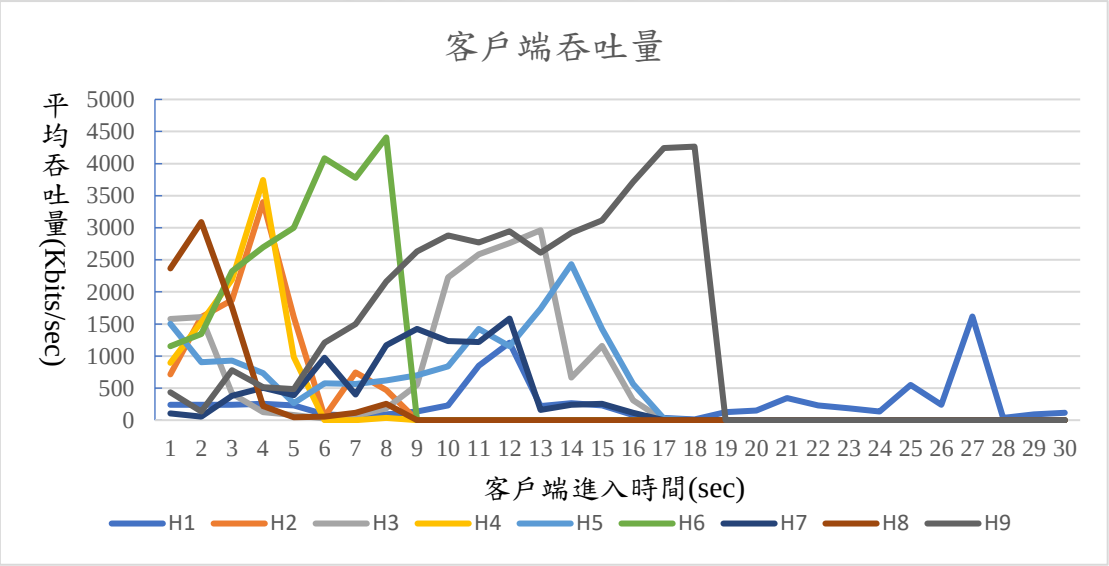

圖4.10　方法二循環出現客戶端吞吐量

儘管遺失率處在偏高的階段，但正常客戶端能持續訪問，除了少部分例外：如同時出現的 H3 與循環出現的 H3 與 H5 與 H9。

方法二採用的方式能計算每一個封包來源的訪問次數，並在五秒內重置，藉以計算每個五秒周期內的封包次數，這個計算能找出每一個客戶端發送的頻率並找到頻率較高者為攻擊，但是當面對找出攻擊後導致原本堵塞的正常流量一次大量進入時，卻會發生誤判的情形。因此，針對這個情況，我們設計了方法三作為驗證分級模組的第三個演算法。表 4.11、4.12 與 4.13 為方法三客戶端循環出現示意表。

表4.11 方法三客戶端分開出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	0%	243.6	Kbits/sec
2	攻擊	正常	0~5	15.72%	2228.8	Kbits/sec
		疑似攻擊	5~6	46%	3472.0	Kbits/sec
		攻擊	6~30	96%	6.7	Kbits/sec
3	正常	正常	0~30	0%	243.6	Kbits/sec
4	攻擊	正常	0~8	31%	2742.0	Kbits/sec
		疑似攻擊	8~9	100%	0.0	Kbits/sec
		攻擊	9~30	100%	0.0	Kbits/sec
5	正常	正常	0~30	0%	243.6	Kbits/sec
6	攻擊	正常	0~9	31%	3541.3	Kbits/sec
		攻擊	9~30	100%	0.0	Kbits/sec
7	正常	正常	0~30	0%	243.6	Kbits/sec
8	攻擊	正常	0~9	43%	2928.0	Kbits/sec
		疑似攻擊	9~11	65%	2232.0	Kbits/sec
		攻擊	11~30	100%	0.0	Kbits/sec
9	正常	正常	0~30	0%	243.6	Kbits/sec

表4.12 方法三客戶端同時出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	52%	84.6	Kbits/sec
2	攻擊	正常	0~5	17%	1779.8	Kbits/sec
		攻擊	5~30	100%	32.2	Kbits/sec
3	正常	正常	0~30	81%	776.3	Kbits/sec
4	攻擊	正常	0~8	88%	1365.3	Kbits/sec
		攻擊	8~30	100%	0.0	Kbits/sec
5	正常	正常	0~30	92%	61.2	Kbits/sec
6	攻擊	正常	0~23	91%	927.5	Kbits/sec
		攻擊	23~30	100%	0.0	Kbits/sec
7	正常	正常	0~30	76%	63.9	Kbits/sec
8	攻擊	正常	0~22	90%	970.1	Kbits/sec
		攻擊	22~30	100%	8.6	Kbits/sec
9	正常	正常	0~30	82%	67.0	Kbits/sec

表4.13 方法三客戶端循環出現示意表

	用戶種類	判斷結果	時間(s)	平均遺失率	平均吞吐量	
1	正常	正常	0~30	10%	281.2	Kbits/sec
2	攻擊	疑似攻擊	0~3	7%	1029.3	Kbits/sec
		攻擊	3~30	93%	13.6	Kbits/sec
3	正常	正常	0~30	29%	528.0	Kbits/sec
4	攻擊	正常	0~2	12%	1092.0	Kbits/sec
		疑似攻擊	2~3	26%	1872.0	Kbits/sec
		攻擊	3~30	100%	0.0	Kbits/sec
5	正常	正常	0~30	46%	660.1	Kbits/sec
6	攻擊	正常	0~6	25%	3486.7	Kbits/sec
		疑似攻擊	6~8	28%	3384.0	Kbits/sec
		攻擊	8~30	97%	44.9	Kbits/sec
7	正常	正常	0~30	50%	302.0	Kbits/sec
8	攻擊	正常	0~5	88%	1052.8	Kbits/sec
		攻擊	5~30	100%	4.1	Kbits/sec
9	正常	正常	0~30	85%	563.9	Kbits/sec

從方法三的數據表可以看出，並沒有誤判情形發生，並且攻擊流量皆有被擋下。與方法二相比，方法三考量了新的閾值，且這個閾值會根據訪問時間而動態增加，這樣就能避免因攻擊被擋下，突然增加的頻寬資源讓先前受攻擊影響的堵塞流量增加，瞬間的訪問次數增加以致誤判。

其中有一些客戶端遺失率偏高，主要是因為即使攻擊客戶端有被辨識出來，在辨識出來前攻擊流量確實影響著網路環境。在擋下一個攻擊後，另一個攻擊擋下的時間點前就會持續受到攻擊流量的影響。以下圖4.11、4.12 與 4.13 是方法三的吞吐量圖。

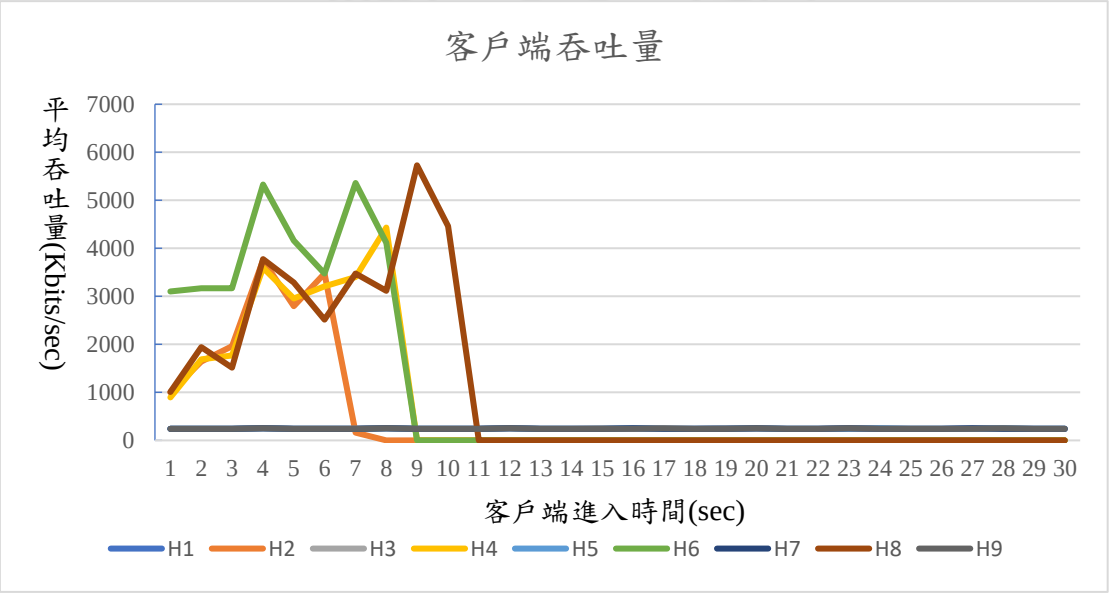

圖4.11 方法三分開出現客戶端吞吐量

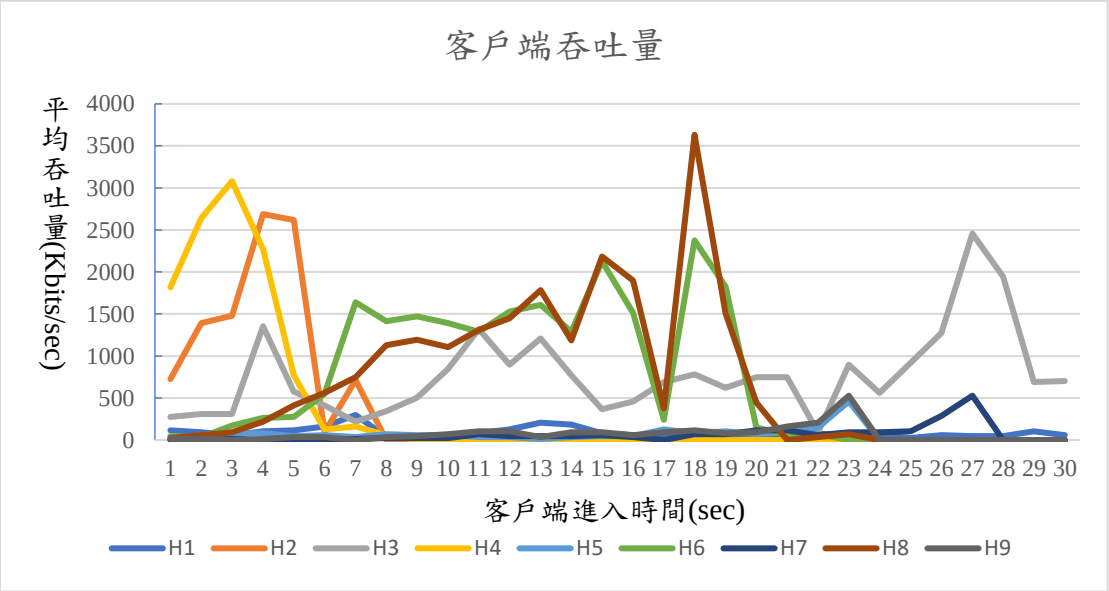

圖4.12 方法三同時出現客戶端吞吐量

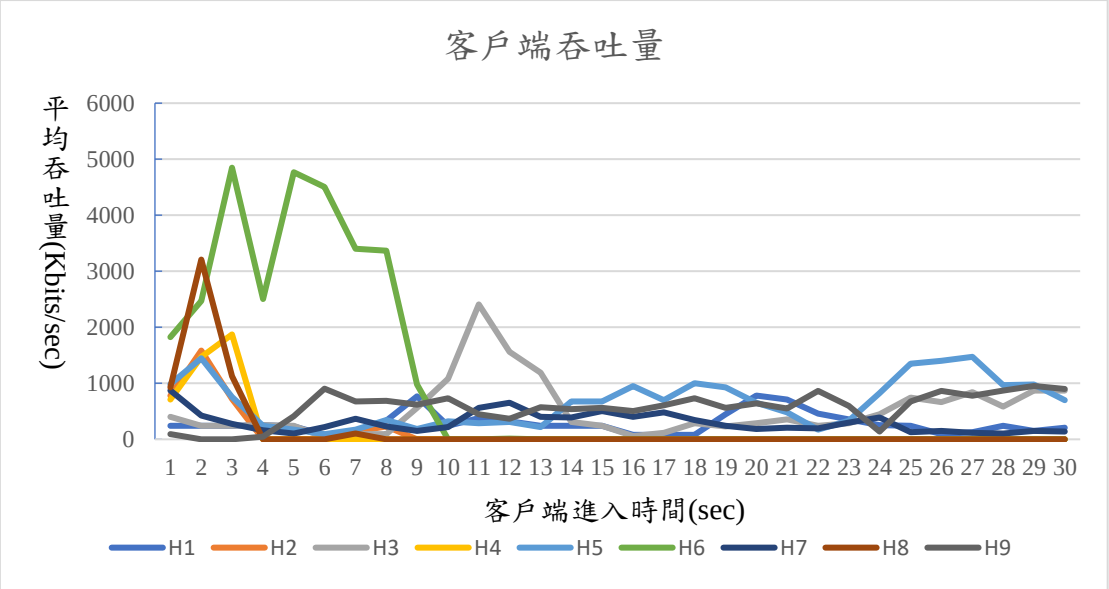

圖4.13 方法三循環出現客戶端吞吐量

從方法三的數據可得知，方法三的攻擊辨識是較佳的，至於如何使遺失率更低，可以藉由調整驗證分級模組的閾值來改進，我們也可得

知，錯誤的閾值設計會導致系統無法使用，但是如果閾值設計妥當且正確，則能有效阻擋並緩解攻擊。

4.3 主要貢獻

本論文的重點著重在使用 SDN 網路中的網路封包數量計算熵，並利用其擅長偵測系統異常的特性設計了一個可快速應對 DDoS 攻擊並加以緩解的系統。結合機器學習分類器後，便形成了兩階段判斷的 DDoS 攻擊偵測緩解系統。此系統具有快速應對攻擊流量，並同時具有一定水準攻擊辨識率，足以面對瞬間大量且快速變化的 DDoS 攻擊。且我們也藉由實驗結果證實在閾值設定正確的情況下，系統的確能化解大流量攻擊。

第五章　　結論與未來展望

透過實驗結果的數據分析，能得知此系統確實能應對瞬間高流量的攻擊，在系統遭受龐大流量的攻擊時偵測發生異常並加以緩解。基於熵交換器分流架構，透過在底層網路架構就進行阻擋，讓攻擊流量在第一時間就進行緩解處理。機器學習緩解模組負責在系統回報網路資料後進行更進一步的攻擊判斷，以提高攻擊的辨識率。

本論文目前著重在兩階段的架構與 SDN 網路中的初階攻擊判斷，我們建立起了能與其他優秀機器學習模組的成果結合的框架，期待未來能加入並訓練更多且更精準的機器學習模組。

根據先前的研究我們可以採用 SVM 作為機器學習分類器的演算法，未來可以考量更強大的機器學習判斷模組，就能同時擁有快速的偵測與強大的辨識能力。並將模擬的環境部屬到實際網路環境上，將熵判斷模組與控制器分開，達成實際意義上的兩階段分層攻擊判斷。我們也期待未來能實現動態的閾值判斷，以適應隨時變化的網路環境。

除此之外，我們期待未來能加入更多樣化的 DDoS 攻擊辨識種類，應對不斷進化的網路資安攻擊。

參考文獻

[1] 王智仁(2022)。A10 Networks 調查：DDoS 攻擊武器來源已超過 1500 萬件。

Retrieved from: https://www.netadmin.com.tw/netadmin/zh-tw/snapshot/271CE26111F44026A9A778A9156B43EB

[2] Yebo Feng ; Jun Li ; Thanh Nguyen (2020,Oct). Application-Layer DDoS Defense with Reinforcement Learning. 2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)

[3] 劉季清(2022)。俄烏火線衝突 網戰打先鋒 攻擊軟體供應鏈

Retrieved from https://ctee.com.tw/news/tech/600613.html

[4] 李宗翰(2014)。DNS DDoS 攻擊的類型

Retrieved from https://www.ithome.com.tw/tech/87818

[5] LYNN(2017)。通訊設備商的挑戰與電信商的機遇：藉由 SDN 與 NFV，5G 軟體主宰時代來臨。

Retrieved from https://kopu.chat/2017/08/18/sdn-nfv-5g/(2019)

[6] 信銳技術(2018)。SDN 將宏圖大展？引領數據中心未來的到底是啥。

Retrieved from https://kknews.cc/tech/bl2pnon.html

[7] WEI-CHUN CHENG. “Quality aware routing for video streaming over SDN”. Tamkang University, 2015, Master’s Thesis.

[8] 楊峻昇. (2020). 基於 SDN 與 vCDN 整合之多媒體串流最佳化研究. Tamkang University, 2020, Master's Thesis.

[9] Hong, Guo-Chih, Chung-Nan Lee, and Ming-Feng Lee. "Dynamic threshold for DDoS mitigation in SDN environment." 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC). IEEE, 2019.

[10] Wenwen Sun; Yi Li; Shaopeng Guan*(2020, February). An Improved Method of DDoS Attack Detection for Controller of SDN. 2019 IEEE 2nd International Conference on Computer and Communication

[11] Kokila, R. T., Selvi, S. T., & Govindarajan, K. (2014, December). DDoS detection and analysis in SDN-based environment using support vector machine classifier. In 2014 sixth international conference on advanced computing (ICoAC) (pp. 205-210). IEEE.

[12] Ye, J., Cheng, X., Zhu, J., Feng, L., & Song, L. (2018). A DDoS attack detection method based on SVM in software defined network. Security and Communication Networks, 2018.

[13] Kousar, H., Mulla, M. M., Shettar, P., & Narayan, D. G. (2021, June). Detection of DDoS Attacks in Software Defined Network using Decision Tree. In 2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT) (pp. 783-788). IEEE.

[14] Sangodoyin, A. O., Akinsolu, M. O., Pillai, P., & Grout, V. (2021). Detection and Classification of DDoS Flooding Attacks on Software-

Defined Networks: A Case Study for the Application of Machine Learning. IEEE Access, 9, 122495-122508.

[15]Ryu SDN Framework. Retrieved from: https://osrg.github.io/ryu/

[16]Entropy (Information Theory). Retrieved from: https://en.wikipedia.org/wiki/Entropy_(information_theory)