行政院國家科學委員會專題研究計畫　成果報告

子計畫一:研究與實做在 SoC 環境下考慮狀態之網路內容分類的語言、編譯器與執行引擎

計畫類別： 整合型計畫
計畫編號： NSC93-2213-E-002-113-
執行期間： 93 年 08 月 01 日至 94 年 07 月 31 日
執行單位： 國立臺灣大學資訊管理學系暨研究所

計畫主持人： 孫雅麗
共同主持人： 陳孟彰
計畫參與人員： 蕭舜文、黃仲君

報告類型： 精簡報告

處理方式： 本計畫可公開查詢

中　華　民　國 94 年 11 月 1 日

以內容為基礎之網路安全－子計畫一：研究與實做在SoC環境下考慮狀態之網路內容分類的語言、編譯器與執行引擎（1/2）

Design and Implementation of the Specification Language, Compiler and Engine for Stateful Content-based Processing in SoC Environment

計畫編號：94-2213-E-002-054
執行期限：93/8/1-94/7/31
整合型計畫：總計畫主持人：李程輝教授
子計畫主持人：孫雅麗教授
執行單位：國立台灣大學資訊管理學系

一、中文摘要（封包分類、狀態、內容檢視）

在這一年的計畫當中，我們研究與實作一個稱為 SConPaC 的狀態化網路封包內容分類器引擎以及其描述語言與編譯器。SConPaC 的設計目標：不同於傳統的封包分類器，本系統具有應用層內容檢視、動態的通訊協定狀態紀錄與維持，以及能夠同時處理 IPv4/IPv6 **封包的特點。**

本系統 a) **根據多種常見的通訊協定與應用服務之規格開始，歸納出檢視封包標頭與應用層內容時會使用的比對特徵；進而設計合適的描述語言** (script language)**。描述語言包含足夠的比對參數種類，以滿足便利與彈性的需求**; b) **描述語言編譯器**(script language compiler)**將描述語言翻譯成程式碼，自動地將規則寫進規則資料庫內，以提供狀態化內容分類引擎** (stateful content-based classification engine) **分類封包時的依據**; c) **分類引擎將分類程序切割成多個階段，每個階段由不同的建構模組來進行分類程序**; **以及** d) **維持與追蹤通訊協定狀態。**

本計畫所提出的 SConPaC 封包分類架構，不但符合現今封包分類的要求(狀態化及內容檢視)，同時也提出了獨創的概念及設計。

英文摘要 （Packet Classification, Packet Classifier, Stateful, Content Inspection）

In this year of the project, we developed a new stateful content-based packet classification called **SConPaC**. The motivation for developing **SConPaC** is stated as follow: compared to the traditional packet classifier, this architecture is capable to inspect the packet application content, maintain and track protocol state transition dynamically, and handle both IPv4 and IPv6 packets.

We study numerous applications and protocols in wide-spread use. We generalize their features which are commonly utilized when inspecting the application content and packet header. Our ***Script Language*** covers

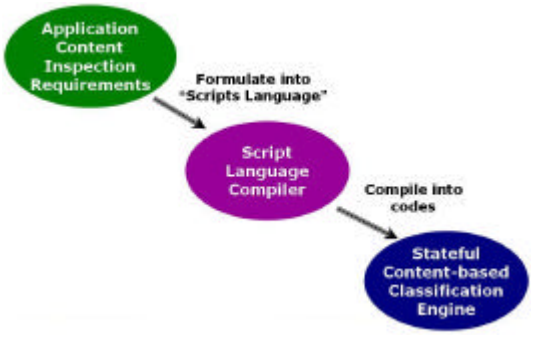
sufficient types of matches to satisfy the requirement of convenience and flexibility. ***Script Language Compiler*** compiles the script language into codes and stores the rule specifications into a rule table. Along with the rules, **SConPaC *Engine*** can perform the procedure of packet classification. **SConPaC *Engine*** comprises several functional components. Separating the filtering procedure into multiple stages is one of the features of Classification Engine. Each stage is implemented as different building blocks consistent with the characteristics of the matches. In addition, classification engine maintains and tracks the state transition of protocols in order to understand the evolution of connections.

The architecture we proposed not only meets the requirements of current packet classification (stateful and content inspection), but also brings up some original ideas and design.

二、研究方法與結果

A. *Building* SConPaC *System*

SConPaC 建構方式如下：



圖一、SConPaC 系統建構程序

研究多種目前常被使用到的通訊協定與應用服務，了解訊息格式在封包標頭與應用層內容中的特性與規格。歸納出分類封包時需要使用到的比對特徵，進而設計出合適的描述語言(Script Language)**，以涵蓋足夠的比對參數**(Match)**種類，才能描述出最多種類的通訊協定或應用服務。**

描述語言編譯器(Script Language Compiler)**將描述語言進行直譯的動作，將網路政策規則編譯成程式碼，並將規則存放至規則表內，以提供狀態化內容分類引擎**(Stateful Content-based Classification Engine)**作為封包分類時的依據。**

狀態化內容分類引擎是由多個功能元件堆砌而成。本系統的核心功能：連線的狀態維持以及封包過濾，即是分別透過分類引擎中不同的功能元件所負責。本系統封包過濾功能的設計特色是，依照分類性質的不同，將封包的分類程序切割成多個階段。IPv4/IPv6 **標頭中的固定欄位、**IPv6 **的延伸標頭，或是封包應用層內容檢視等，都具有不同的分類特性。每階段使用最適合其分類特性的演算法來進行分類的動作，以期增加整體的分類效率、節省不必要的資源耗費。**

B.1 Script Language

描述語言的設計包含了指令(Commands)**與比對參數**(Match)**，如下：**

command [match/matches] ***policy***

表一、描述語言的指令

Command	Description
--insert_rule	insert a new rule
--delete_rule	delete a reul
--list_rule	list all rules

表二、描述語言的比對參數

Match	Description
--ver	IP **的版本**
--src	**來源** IP **位址**

--dst	目的地 IP 位址
--proto	傳輸層的通訊協定
--tc	IPv6 Traffic Class
--f_label	IPv6 Flow Label
--len	IP 封包長度
--hop_limit	IPv6 Hop Limit； IPv4 Time To Live
--sport	來源通訊埠
--dport	目的地通訊埠
--tcp_flags	TCP Flags
--exist	是否存在延伸標頭
--conn_state	連線狀態
--l4_state	傳輸層協定狀態
--app_state	應用層協定狀態
--app_proto	應用層通訊協定
--Content [start..end]	應用層 字串比對 I
--match_exactly (from, shift, len)	應用層 欄位字串比對
--match_string	應用層字串比對 II
--match_string_within(start, end)	應用層 字串比對 III
--user_define(from, shift, len)	應用層 自定欄位比對

表三、描述語言的政策

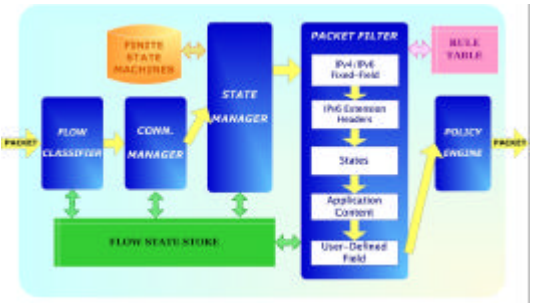
Policy	Description
ACCEPT	接受封包通過
DROP	丟棄封包
LOG	紀錄封包的詳細資訊

B.2 Script Language Compiler

描述語言編譯器負責將描述語言翻譯為成可執行的程式。描述語言編譯器包含了兩個主要元件：字彙分析器(*Lexical Analyzer*，或 Scanner)、語法分析器(*Syntax Analyze*，或 Parser)。

字彙分析器讀入以描述語言所敘述的規則，進行字彙的分析並將描述語言翻譯成辭彙(Token)。翻譯後的辭彙交由語法分析器分析出此規則所代表的語意。最後，語法分析器將規則中比對的值，儲存至分類引擎的規則資料庫(Rule Table)當中，作為分類引擎在進行封包分類時的依據。

B.3 Classification Engine



封包首先會進入資料流分類器(*Flow Classifier*)，搜尋此 IP 封包所屬之資料流，並且根據資料流的出現情況將資料流分類。然後連線管理器(Connection Manager)會為連線進行對應的處理，並且將相關資訊儲存於資料流狀態儲存庫(*Flow State Store*)中。狀態管理器(*State Manager*)會為 IP 封包所屬連線進行狀態追蹤的動作。狀態管理器分析 IP 封包的內容，辨別出所使用的通訊協定，然後再透過對應的有限狀態機(*Finite State Machines*)的協助、進行狀態追蹤的功能。得到了最新的連線狀態後，封包過濾器(*Packet Filter*)即可根據封包的標頭與內容、進行過濾的動作。最後，政策引擎(*Policy Engine*)根據訂定的規則，進行決策的動作。

Flow Classifier 資料流分類器是依封包標頭的五個欄位(Source IP, Destination IP, Protocol, Source Port, Destination Port)來分類資料流。資料流分類器根據此條資料流的識別碼，可至資料流狀態儲存庫中尋找對應的資料流紀錄。資料流狀態儲存

庫中存放了每條流經資料流的基本資料以及狀態的相關資訊。

Connection Manager 連線管理器負責連線資訊的紀錄與相關處理，並且為每個連線進行連線狀態的維持。一個連線(connection)中，可能包含數個以上的資料流(Flow)，要視協定或應用程式的設計。

State Manager 狀態管理器會辨識每個連線所使用的通訊協定，並且透過有限狀態機(*Finite State Machines*)的協助，維持並紀錄每個連線的狀態。使用非標準埠口(port)的連線會再次地進行通訊協定辨識的動作，以保證內容辨識率的增加。

Finite State Machines 有限狀態機儲存了通訊協定狀態轉換的規則，以支援狀態管理器的通訊協定狀態追蹤功能。本系統使用了包含：SMTP、HTTP、SIP、eDonkey2000、TCP、UDP、ICMP，以及屬於 IPv6 的 Mobile IPv6 協定。

Packet Filter 封包過濾器根據規則資料庫中所訂定的規則，檢視封包的標頭、應用層內容、與目前所屬資料流的狀態，找出符合的規則。根據描述語言中比對參數歸納出五種類型：封包位於網路層與傳輸層的固定欄位、IPv6 延伸標頭、連線狀態與通訊協定狀態、應用層內容檢視、使用者自行定義欄位。每一個類型都可以用該類型適用的分類比對演算法，以增加比對的效率，而避免分類特性不相同而導致單一的演算法效能低落。最後本系統利用位元向量(bit vector)的概念，在每一階段中產出一組位元向量，位元向量的索引(Index)代表在規則資料庫中的規則識別碼。位元向量的值表示是否為成功的比對。因此我們可以將不同階段的結果綜合起來交給決策引擎。

Policy Engine 則根據比對的結果，選擇優先等級最高的規則來執行決策動作。

C Examples and Result

我們實做了該引擎並以下述的例子作為測試，達成狀態化封包內容分類器的目標。描述語言如下：

- **限制來源及目的地 IP 位址**
  --insert_rule --src 140.112.107.41 --proto icmp --action drop
- **限制非 TCP 封包不可通過**
  --insert_rule --proto !tcp --action drop
- **設定 TCP Flags**
  --insert_rule --proto tcp --tcp_flags ALL:SYN,ACK --action drop
- **設定 ports**
  --insert_rule --proto tcp --sport >1024 --action drop
- **設定 IPv6 延伸標頭**
  --insert_rule --ver 6 --DstHdr.OptType 201 --action log
- **內容檢視**
  - --insert_rule --Content[0..88] “HTTP 200 OK” --action drop
  - --insert_rule --match_exactly(PktPayload, 0, 88) “HTTP 200” --action drop
  - --insert_rule --match_string “virus” --action drop
- **應用層的通訊協定**
  --insert_rule --ver 6 --src 3ffe::100:2:b3ff:fe15:fca7/64 --app_proto MobileIPv6 --action accept
- **設定狀態**
  --insert_rule --proto tcp –conn_state ESTABLISHED,RELATED --action accept
- **使用者自訂欄位**
  --ver 6 --DstHdr.Opt 201 --define_field (DstHdr.Opt, 16, 128)

3ffe:0501:0008:0:0260:97ff:fe40:efab

三、參考文獻

1. Application Layer Packet Classifier for Linux, http://l7-filter.sourceforge.net/
2. F. Baboescu and G. Varghese, “Scalable Packet Classification”, In proceedings of ACM SIGCOMM Conference, 2001
3. Andrew Begel, Steven McCanne, and Susan L. Graham, “BPF+: Exploiting Global Data-flow Optimization in a Generalized Packet Filter Architecture”, In Proceedings of ACM SIGCOMM Conference, 1999
4. “Check Point Software: Stateful Inspection Technology”, Tech Note, http://www.checkpoint.com/products/downloads/Stateful_Inspection.pdf
5. S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, December 1998
6. S. Dharmapurikar, P. Krishnamurthy, T. Sproull, and J. Lockwood, “Deep Packet Inspection using Parallel Bloom Filters”, In Proceedings of the IEEE Symposium on High Performance Interconnects, 2003
7. Knuth D. E., Morris (Jr) J. H., and Pratt V. R., “Fast pattern matching in strings”, SIAM Journal on Computing, 1977
8. eDonkey Protocol Specification, http://sourceforge.net/projects/pdonkey/
9. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1”, RFC 2616, June 1999
10. C. Georgopoulos, G. Konstantoulakis, T. Orphanoudakis, N. Nikolaou, J-A. Sanchez-P., N. Mouratidis, K. Pramataris, and N. Zervos. A Protocol Processing Architecture Backing TCP/IP-based Security Applications in High Speed Networks (PRO3) http://www.pro3-processor.com/
11. P. Gupta and N. McKeown, “Packet classification on multiple fields”, In proceedings of ACM SIGCOMM Conference, 1999
12. P. Gupta and N. McKeown, “Packet classification using hierarchical intelligent cuttings”, In Proceedings of the IEEE Symposium on High Performance Interconnects, 1999
13. M. Handley, and V. Jacobson, “SDP: Session Description Protocol”, RFC 2327, April 1998
14. Hifn Inc., “Why You Need Flow Classification”, Technical White Paper, http://www.hifn.com/docs/a/WP-0001-00-Why-You-Need-Flow-Classification.pdf, September 2001
15. IPtables: Connection Tracking, http://www.sns.ias.edu/~jns/security/iptables/iptables_conntrack.html
16. D. B. Johnson and C. Perkins, “Mobility Support in IPv6”, Internet Draft, draft-ietf-mobileip-ipv6-13.txt, November 2000
17. I. Kang and H. Kim, “Determining Embryonic Connection Timeout in Stateful Inspection”, IEEE International Conference on Communications (ICC), 2003

18. J. Klensin, "Simple Mail Transfer Protocol", RFC 2821, April 2001
19. T.V. Lakshman and D. Stiladis, "High-Speed Policy-based Packet Forwarding Using Efficient Multi-dimensional Range Matching", In proceedings of ACM SIGCOMM Conference, 1998.
20. J. Moscola, J. Lockwood, R. P. Loui, and M. Pachos, "Implementation of a Con-tent-Scanning Module for an Internet Firewall", 2003
21. Netfilter, http://www.netfilter.org/
22. Thomas Niemann, "A Guide to LEX & YACC" http://www.lugbe.ch/action/reports/lex_yacc.pdf
23. M. Norton and D. Roelker, "SNORT 2.0 Hi-performance Multi-rule Inspection Engine", http://www.sourcefire.com/technology/whitepapers/sf_snort20_HPMRIE.pdf, April 2004
24. M. Norton and D. Roelker, "SNORT 2.0 Protocol Flow Analyzer" http://sourcefire.com/whitepapers/sf_snort20_protoflow.pdf, April 2004
25. N. A. Noureldien and I. M. Osman, "A Stateful Inspection Module Architecture", In Proceedings of TENCON Conference, 2000
26. Rahul Patel, and Hifn Inc., "Stateful vs. Stateless Traffic Analysis", In Proceedings of IIC-China/ESC-China Conference, 2002
27. J. Postel , "User Datagram Protocol", RFC 768, August 1980
28. J. Postel , "Transmission Control Protocol", RFC 793, September 1981
29. J. Postel , "Internet Control Message Protocol", RFC 792, September 1981
30. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002
31. Boyer R. S. and Moore J. S., "A fast string searching algorithm", Communications of the ACM, 1977
32. D. V. Schuehler and J. Lockwood, "TCP-Splitter: A TCP/IP Flow Monitor in Reconfigurable Hardware", In Proceedings of the IEEE Symposium on High Performance Interconnects, 2002
33. D. V. Schuehler, J. Moscola, and J. Lockwood, "Architecture for a Hardware Based, TCP/IP Content Scanning System", In Proceedings of the IEEE Symposium on High Performance Interconnects, 2003
34. S. Singh, F. Baboescu, G . Varghese, and J. Wang, "Packet Classification Using Multi-dimensional Cutting", In Proceedings of ACM SIGCOMM Conference, 2003
35. V. Srinivasan, G. Varghese, S. Suri and M. Waldvogel, "Fast and Scalable Layer Four Switching", In Proceedings of ACM SIGCOMM Conference, 1998
36. V. Srinivasan, S. Suri and G. Varghese, "Packet Classification using Tuple Space Search", In proceeding of the ACM SIGCOMM Conference, 1999
37. Yeali S. Sun, Chung-Chun Huang, Cheng-Wei Lee, and Fan Liu,

“Hierarchical Mobile IPv6”, National Computer Symposium, 2003