Traffic Flow Management in Next Generation Service Provider Networks - Are We There Yet?

Ryan Goss
Institute for ICT Advancement & School of ICT
Nelson Mandela Metropolitan University
Port Elizabeth, South Africa
Email: ryan@goss.co.za

Reinhardt Botha
Institute for ICT Advancement & School of ICT
Nelson Mandela Metropolitan University
Port Elizabeth, South Africa
Email: reinhardta.botha@nmmu.ac.za

***Abstract*—For years a number of savvy Internet users have avoided firewalls and traffic engineering measures by directing traffic through ports seemingly unrelated to the application. These ports are those often marked by firewall administrators as "safe" or those given a higher priority on quality of service systems. This problem has been effectively managed by implementing deep packet inspection techniques, giving the administrators a view into the underlying layer 7 protocol of each flow. The reliance on transit payload to be in plain text format in order to reliably match the underlying content has put this method of classification at a major disadvantage.**

The use of encryption by users to render the contents of a data packet opaque is, therefore, of major concern to network administrators who rely heavily on deep packet inspection. Without the ability to interrogate the underlying payload of traffic flows, a new method to identify this type of traffic needs to be discovered in order to retain control of the network. As an increasing number of users turn to IP tunneling to secure their data transfers, network service providers need to ensure their systems are ready to handle this type of traffic. A failure to do so would result in them facing the reality of a badly managed network.

This paper highlights the challenges faced by network service providers in opaque traffic classification for both existing and future, next generation networks. It investigates and evaluates the various solutions implemented in order to manage network traffic "in the dark".

I. Introduction

The Internet has, since its inception as a public access communications facility, become the universal communications infrastructure in business [1]. The classification and identification of network applications is therefore an important requirement for network administrators, allowing them to detect and mitigate security threats, protecting the network from unwanted applications [2], [3], [4]. Although the requirement for traffic management is in place, an accurate method for reliably identifying applications associated with network traffic is still to be developed [2]. The original design for the TCP/IP protocol was based on trust, where communications would not be seen as a threat [1]. With new means of communication constantly emerging, some of which constituting resource misuse, the development of such management methodologies has become increasingly urgent [5]. The following section provides a background of various mechanisms which have been employed over the years in an attempt to classify various applications.

II. Network Traffic Classification

On a network, a flow or session is defined as the communication between two hosts on the network, described by a quintuple comprising of a source and destination address, protocol, source and destination ports [1], [6]. Internet Protocol (IP) packets sharing the same quintuple information generally belong to a single session [1]. Multiple sessions can therefore be identified by determining unique quintuples of IP packets on the network at any given time. The first packet seen, the synchronize or SYN packet, determines the beginning of the flow whilst the termination of a flow could be due to either a time-out or protocol based termination mechanism [6]. Traffic within a particular flow can be viewed in two directions, namely from source to destination or from destination to source [6]. The Transport Control Protocol (TCP) requires that for every transmitted packet, a return acknowledgement (ACK) packet be sent. These ACK packets do not carry any payload and are simply used to confirm the successful reception of the last set of packets.

A. Early Classification Techniques

The first classification systems made use of protocol and port information to classify traffic using known ports [2], [7]. The fact that port and protocol information is determined entirely by end hosts allows them to easily be changed by the user to disguise or conceal traffic [4]. For this reason, standard port and protocol matching technologies, although fast and efficient, became increasingly inaccurate [2].

Devious applications began developing their own application level protocols, operating at layer 7 of the OSI model. These protocols gave them the ability to use dynamic port selection mechanisms to communicate, providing a mechanism for the violation of network policies, allowing previously prohibited applications to traverse firewalls and enter the network [4]. The evasion of firewalls and Quality of Service (QoS) classifiers became a major concern for network administrators, who needed a way to interrogate the layer 7 protocol of such flows.

The answer to this problem was realized with the introduction of Deep Packet Inspection (DPI) technologies, a mecha-

nism for interrogating the underlying layer 7 information of traffic flows.

B. Deep Packet Inspection

Inspecting the payload of every packet in the network is a traditional approach to classifying network traffic, instituted by a number of network administrators [3]. Deep packet inspection techniques, when applied to plain-text flows, can be very accurate and provide network administrators a view into dubious traffic on the network [2], [3], [7].

DPI has become an essential tool for network security engineers, enabling them to search both packet header and payload (content) for predefined protocol signatures [8]. These signatures can be created to match legitimate, permissible applications as well as potentially malicious attacks on the network. In order for a network to be managed effectively, both types of traffic, good and bad, need to be identified.

The art of matching patterns within strings can be expanded using regular expressions, which allows for more flexible string searching and allows access control policies (ACP) the ability to match on mutable content [7]. Two of the most famous string matching algorithms include the Aho-Corsasick and Commetz-Walter algorithm [8]. These and other classification algorithms have limitations, both in what traffic can be classified and challenges in their implementation.

III. Challenges in Flow Classification

As in the case of classic port/protocol matching becoming obsolete, so too is the wide spread use of DPI as an alternative to network flow classification. DPI may be successfully implemented in enterprise or small to medium sized service provider networks, however, these algorithms were proposed as suitable for line speeds not exceeding 100Mbit/s. This speed is sub-standard for both current and future medium to large network service providers.

The speed limitation is but one drawback of DPI systems. Another to consider is the privacy regulations associated with user data in transit [9]. Many users, if made aware of the use of DPI, may raise concerns with the examination of the data within their network flows [2]. Service providers and enterprise networks need to protect the confidentiality of data transmitted across their networks. If sensitive information is in any way intercepted, the trust between end-users and the service provider will be broken.

A further consideration to take into account is the high computational and storage overhead associated with DPI [2]. This overhead increases exponentially when interrogating every packet traversing the network; a problem compounded as developments toward higher-speed network links continues. The development of new applications contributes to the problem as the number of signatures required to match various protocols proportionately increase. The storage of these signatures will increase over time, putting an increased load on the memory requirements of the classification systems [8]. As network speeds increase, the bottleneck of DPI becomes apparent as the Deterministic Finite Automation (DFA) algorithm, used by many DPI systems, starts to strain [8]. This strain results in increased queuing on the router interfaces, increasing the network latency experienced by the user and degrading the overall performance of the network connection. DPI systems will therefore be faced with high-performance challenges as link rates and traffic volumes on service provider networks continues to increase [8]. To illustrate this point, consider the string matching algorithm used in the Snort Network Intrusion Detection System (NIDS) [10]. The algorithm accounts for up to 70% of execution time and 80% of instruction cycles of the application [8].

Recent application development has introduced one final, potentially devastating challenge to DPI systems. In an effort to avoid DPI systems matching their protocols, applications are using encryption to secure their underlying payload [2].

A. Managing Opaque Network Traffic

By encrypting or otherwise altering the state of the payload of any data packets, the contents thereof become opaque and are thus unmanageable by DPI systems. Application protocols are therefore difficult to detect if the underlying protocol is encrypted [5].

Applications such as Secure Shell (SSH), Secure Socket Layer (SSL) and various Virtual Private Network (VPN) technologies make use of encryption to secure the data packet whilst in transit. These opaque network flows are of great concern for network administrators attempting to manage traffic flows on their network. Unfortunately for these administrators, an increasing number of applications on the Internet are moving toward encryption technologies, making it difficult to accurately match the their traffic [4], [6].

The problem of identifying encrypted network traffic is not limited to DPI systems. The identification of encrypted network traffic by any means has been the subject of many research projects of late in the field of network flow classification [1], [3], [6], [9]. In each case, the opacity of data in transit presents as possibly the most challenging obstacle to overcome when managing traffic flows.

Trojan horse or virus programs, along with other malicious applications, have been known to encrypt their traffic in an attempt to deter any payload detection signatures from matching their communications [4]. Other applications, including popular P2P file sharing protocols, have altered their protocols to support bi-directional encryption to avoid detection. Bittorrent, arguably one of the most popular P2P file sharing platforms today [11], is one such application which offers users the option to encrypt communications between peers. Encryption scrambles the payload of the packet, preventing string matches from taking place using predefined signatures. Encryption therefore renders DPI systems ineffective in matching strings within the payload.

Applications themselves need not provide encryption facilities natively in order to encrypt their data in transit. In order for any IP enabled application to render their payload opaque to DPI systems, the user need only direct the traffic flow through a Virtual Private Network (VPN) or IP tunnel.

B. Tunneling to Increase Privacy of Data Flow Payloads

Although the wide-spread, global reach of the Internet provides communication facilities unrivaled by others, securing communications across such a public medium has become very important to end-users and enterprises alike. One solution many are turning to is the use of Virtual Private Networks (VPN). A VPN is a private, secure network running over public network infrastructure, such as the Internet [1]. A VPN establishes a logical tunnel, secured using various cryptographic mechanisms [1]. Tunneling is the encapsulation of one network protocol within another, with the former the payload of the latter, illustrated in Figure 1.

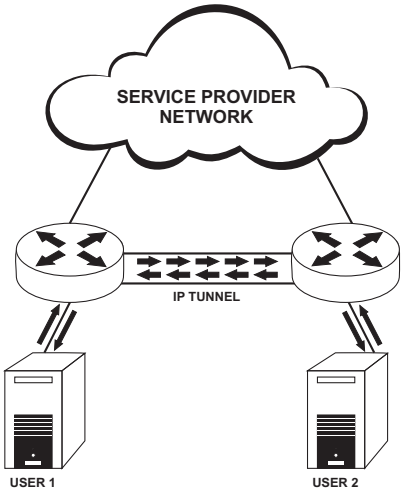

Fig. 1. IP Tunnel

The tunnel, from the perspective of end-users and their applications, is transparent. They are often, therefore, unaware they are being directed through a tunnel [1]. Tunnels are designed to mimic implementation features of physical media, thus permitting multiple protocols and traffic flows the ability to operate within a single instance, at the same time. Whereas in the past a specific flow could be managed based on the information read from its content for each quintuple, a single quintuple could potentially host a number of flows simultaneously. This fact makes it very difficult for service provider's classifiers to manage, especially when encryption is present, skewing the payload.

Packet payload encryption occurs when one or more protocols are encapsulated within the tunneling protocol, such as in the case of a secure shell (SSH) tunnel [5]. SSH, as an example, not only supports remote terminals, but also supports secure remote copy (SCP) and tunneling of TCP/IP traffic. The ability for protocols such as SSH to exhibit multiple behavioural patterns makes them increasingly difficult to manage. [4]. There are many implementations of IP tunneling available today, including IP Security (IPSec), Microsoft's Point to Point Tunneling Protocol (PPTP), Cisco's Layer 2 Tunneling Protocol (L2TP) and the afore mentioned tunneling via SSH.

Encrypted tunnels form one of the most complicated traffic engineering mechanisms for service providers to manage. For this reason, the following section discusses this topic in more detail, highlighting various attempts which have been made to mitigate the problem and enable a view for administrators into the traffic profiles within such tunnels. This paper uses the SSH protocol as an example tunneling mechanism to illustrate certain facts about encrypted IP tunnels and the effectiveness achieved in managing their opaque traffic flows.

IV. Managing Opaque Network Traffic Flows

SSH is typically used as a remote shell access mechanism to Unix/Linux based systems [2], [12]. The SSH protocol also supports a number of other functions, including the transfer of files using the Secure Copy protocol (SCP), forwarding arbitrary TCP ports over a secure channel between 2 hosts and tunneling. Traffic running on the SSH protocol is encrypted, rendering payload analysis based classification techniques ineffective in the identification of underlying flows [2]. Although the SSH protocol is encrypted, the initial SSH handshake between client and server is served in plain text, making it possible to correctly identify an SSH communication using DPI techniques [6]. A simple telnet session to an SSH server illustrates this point:

```
telnet host.domain.com 22
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
SSH-2.0-OpenSSH_4.3p2 Debian-9
```

The SSH request for comment (RFC) dictates any communications past this point need to be encrypted in order to avoid a "Protocol mismatch" error being returned and the session being closed [12]. From the initial handshake received, it is trivial to determine the SSH protocol version (2.0), server application (OpenSSH daemon), version information (4.3p2) and the host operating system (Debian Linux). A deep packet inspection classifier could therefore match the SSH protocol by using a simple regular expression signature, illustrated below.

```
# SSH - Secure SHell
# Pattern attributes: great veryfast fast
# Protocol groups: remote_access secure ietf_draft_standard
# Wiki: http://www.protocolinfo.org/wiki/SSH
# Copyright (C) 2008 Matthew Strait, Ethan Sommer; See ../LICENSE
#
# usually runs on port 22
#
# http://www.ietf.org/internet-drafts/draft-ietf-secsh-transport-22.txt
#
# This pattern has been tested and is believed to work well.

ssh
^ssh-[12]\.[0-9]
```

The SSH protocol states that immediately after the handshake process, the encryption keys for the session are exchanged between the client and server [12]. At this point and further, all communications between client and server are encrypted and thus all future payload, including tunneled flows, opaque.

In the case of applications being routed through SSH and other encrypted tunnels, the only remaining discernible

characteristics of the flow is its direction, approximate size and timing between packets [5]. Using this information, it is possible to construct a behavioural profile or signature for various protocols [5]. The signature, consisting of meta information relating to the direction, approximate size and timing of packets within the flow, can be used to match characteristics of certain flows as they traverse the network. A characteristic, or feature, is a descriptive statistic that can be calculated from one or more packets of a flow [6].

As many applications move toward encryption, the ability to identify network flows "in the dark" would be of tremendous practical value to network administrators [4]. Whilst much early research in traffic classification did not take encrypted traffic into account, a number of recent research papers focus primarily on that topic [3]. This research often resulted in the ability to classify flows into broad categories such as "peer to peer", "bulk data transfer" or "interactive" and did not provide a turn key solution to in-depth flow classification [4]. The use of Machine learning was investigated by researchers in an effort to have the machines themselves understand the traffic they were switching and thus attempt to abstract various flows from opaque.

A. Protocol Idenfication using Machine Learning and Statistical Analysis

The use of machine learning techniques was researched in an attempt to classify encrypted traffic, based on certain characteristics unique to each protocol flow [6]. The extracted features from each flow form the input vectors for these machine learning algorithms [6]. These features needed to be characteristics which survived the encryption process [4], thus features such as IP addresses, source and destination ports and layer 4 protocols are excluded from the input vector; removing the dependency on such features [5], [6]. Of the machine learning algorithms observed during a literature survey conducted by the authors, the most prominent were found to be AdaBoost, RIPPER, C4.5, Rough Set with Genetic Algorithms and Hidden Markov models.

In order for a machine learning algorithm to be effective in classifying flows on service provider networks, it should have the ability to deal with a large degree of noise and skewed data, typical of real Internet traffic [4]. Adaboost, short for Adaptive Boosting, is one meta-algorithm used in machine learning [13]. The algorithm creates new classifiers by examining previous misclassifications performed by existing classifiers and adapting to such mistakes. These new classifiers are developed by combining the performance of potentially hundreds of existing, more simple classifiers using a voting scheme [2]. One crucial problem with AdaBoost, like many machine learning algorithms, is its inability to handle noisy datasets [2], a requirement for any successful traffic flow classification algorithm.

RIPPER, or Repeated Incremental Pruning to Produce Error Reduction algorithm, is another machine learning algorithm [2]. Unlike Adaboost, the RIPPER algorithm has the ability to manage noisy, real-world Internet traffic schemes with a fair degree of accuracy. The RIPPER algorithm was found to manage noisy datasets with a lower computational overhead and higher accuracy than Adaboost and other algorithms, such as C4.5 [14].

There are a number of statistical based algorithms used in machine learning systems. C4.5, often referred to as a statistical classifier, is a machine learning algorithm which generates a decision tree used for classification. This decision tree employs a "divide-and-conquer" strategy for attribute based model building [6]. These models become classifiers in flow detection systems. Other examples of statistical algorithms include Hidden Markov models, which are used to build statistical models for the sequence of packets produced by each protocol of interest [4]. These models are then used to categorize future TCP connections observed on the network with a fair degree of accuracy.

Another machine learning algorithm used for flow classification is rough set theory. Rough set theory is a branch of set theory, a major area of research in mathematics [6]. The rough set attributes are reduced using genetic algorithms (GA). These GAs have a fitness function, which calculate a score for each attribute and thus optimize the classifier. Genetic Algorithms are believed to be able to reduce complexity in large decision systems, all the while lowering the computational overhead to do so [6]. More information on the algorithm is available in [15].

The accuracy of machine learning algorithms is complicated by the ability for certain protocols to exhibit multiple behavioural patterns. Protocols, such as SSH, are able to perform real-time, interactive communications as well as secure bulk data transfer operations [4]. Sometimes certain protocols exhibit similar characteristics, such as in the case of the simple mail transfer protocol (SMTP) and file transfer protocol (FTP), which behave the same way in many regards [4].

There are a number of other algorithms which have been considered by researchers in the quest to find the perfect classification system for network traffic flows. One commonality exhibited by all of them is their dependency on the extraction of certain characteristics from each flow, encrypted or plain text. Each algorithm requires these attributes to describe and therefore classify the flows. It is therefore important to correctly determine which attributes should be used as input vectors for the various algorithms.

B. Flow Characteristics Influence Detection Accuracy

Each flow, or quintuple, has a specific signature for each of the two directions of traffic flow [6]. Table I depicts the attributes used by [6] as valid input vectors for machine learning algorithms. The inputs used by various machine learning algorithms may be different as there is no hard and fast rule as to which are the best ones and which are not. The input choices made for the machine learning algorithm may very well depict the accuracy the algorithm in the traffic identification process. Although the use of machine learning techniques to identify encrypted traffic have proven effective, much of the research has shown that it is easier to apply

Protocol
Bytes in forward direction
Packets in forward direction
Bytes in backward direction
Packets in backward direction
Min backward inter-arrival time
Min forward inter-arrival time
Std Deviation of backward inter-arrival times
Std Deviation of forward inter-arrival time
Mean backward inter-arrival time
Mean forward inter-arrival time
Max backward inter-arrival time
Max forward inter-arrival time
Min backward packet length
Min forward packet length
Max backward packet length
Max forward packet length
Std deviation of backward packet length
Std deviation of forward packet
Mean forward packet length
Mean backward packet length
Duration of flow

TABLE I
LIST OF FLOW BASED FEATURES

such techniques to well known application traffic, including web browsing and email. More work is required in order to successfully classify unknown, encrypted applications [6]. Furthermore, the problem of encrypted protocol detection becomes compounded when the encrypted payload consists of multiple application protocols, running in parallel across a single tunnel [3]. Tunneling allows multiple flows to simultaneously operate at any time, interweaving the protocols within the tunnel [5]. This interweaving of simultaneously communicating protocols makes it extremely difficult for system classifiers to differentiate between sessions, thus providing a potentially lower hit ratio for the classification algorithm [5]. The interweaving of protocols within an opaque flow is a consideration to take into account when selecting the characteristics to be used as input vectors for machine learning.

As IP networks continue to grow, the once seemingly limitless resource, IPv4 address space, has become strained, forcing service providers to seek alternative addressing schemes. The use of a new numbering scheme for large networks may result in a complete overhall of existing thoughts with regard to flow based characteristic selection as they bring new packet header fields and flow information. One numbering protocol designed to accommodate the future growth of large networks, gaining widespread popularity, is Internet Protocol version 6.

V. INTERNET PROTOCOL VERSION 6 - SAVIOUR AND COCONSPIRITOR

The Internet Assigned Numbers Authority (IANA) is responsible for assigning the public Internet Protocol (IP) address space space to registered parties across the globe. In order to accomplish this mammoth task, the geographic areas have been broken down and specific delegations controlled by the Regional Internet Registry (RIR) to whom the requester belongs. IANA assigns additional IP allocations to the five RIRs as their available resources begin to deplete. This process has worked well over the years, however in recent times, the once seemingly limitless resource of IPv4 address space at IANA has been exhausted.

Registry	Geographic Region	Unallocated %
AFRINIC	Africa	57.7%
APNIC	Asia Pacific	2.2%
ARIN	America	7.9%
LACNIC	Latin America and Caribbean	29.3%
RIPE	Europe, Middle East and Parts of Asia	9.5%

TABLE II
RIR UNALLOCATED SPACE

The exhaustion of public Internet address space has long been predicted by specialists. With the final five /8 delegations (each consisting of 16,777,214 addresses) being made by IANA to each of the Regional Internet Registries (RIR) in February 2011, it is now more important than ever to consider addressing alternatives. One such alternative gaining momentum in many service provider and enterprise networks is IP version 6 (IPv6).

IPv6 is the latest version of the Internet protocol numbering system, designed to accommodate vastly more devices than its predecessor, IPv4. Although the implementation process of IPv6 across service provider networks has been slow, many are now being forced to implement the protocol due to the exhaustion of IPv4 address space.

The latest unallocated address space reported by Potaroo at the time of writing is shown in Table II [16].

Based on existing allocation or "burn" rates, the existing unallocated IPv4 address space is predicted to run out sometime within the next 2 to 3 years [16]. For certain RIRs, including AFRINIC and LACNIC, this IPv4 exhaustion may not occur for some time after that. Requesters in these regions may therefore have the facility to acquire new IPv4 address space for a longer period than those in other regions. For others, such as APNIC, this may come sooner than expected.

On 15th April 2011, APNIC announced that they are in the process of delegating IPv4 space from their final /8 block [17]. APNIC have thus begun advocating the immediate implementation of IPv6 support for operators within their region.

As regions under the control of rapidly depleting RIRs start to turn on IPv6, it becomes necessary for other regions to follow suite. IPv6 and IPv4 networks cannot communicate natively, thus the drive toward IPv6 implementation will be global, rather than isolated to particular regions. Those who do not participate in the IPv6 deployment process run the risk of becoming an isolated "island" on the Internet, unreachable by those who have.

IPv6 promises to bring, in addition to increased numbering facilities, new developments and applications to the Internet [18]. This, in turn, means more classifiers will be required to accurately control such traffic.

In order to provision IPv6 to an end-user, both the provider

edge and core of the service provider network need the ability to switch IPv6 traffic. The lack of IPv6 implementation by many service providers have forced end-users to seek alternatives for their IP requirements. One solution is the tunneling of IPv6 over an existing IPv4 network. This speeds up the deployment process and makes significantly more public address space available to users immediately. The use of IPv6 in IPv4 tunnels is on the rise, troubling news for network administrators already battling to manage tunneled traffic. Hurricane Electric, the largest IPv6 network, boasts more than 61,886 IPv6 tunneled connections in over 179 countries [19]. The number of tunnels hosted by Hurricane Electric is on the rise, more so now that IPv4 address space is becoming less available.

VI. Conclusion

Networks have, over the years, evolved from their once simple design, where an application could be identified by identifying the ports the communication was undertaken on. Port and protocol matching was overcome by applications who developed their own layer 7 based protocols, enabling them to communicate on dynamic, random ports and protocols. This provided them with a mechanism to avoid firewalls and QoS devices which would otherwise potentially restrict them. Classification systems adapted by looking deeper into the data packet's payload, accomplished by applying regular expressions to match strings of data in the early stages of layer 7 protocol initialization.

Layer 7 DPI was countered by applications who incorporated various levels of encryption in their protocol, making the underlying payload opaque to DPI techniques. Encryption can be seen as a two-edged sword; firstly, it creates a secure medium to transfer sensitive information across a public network infrastructure, reassuring the end-user of information confidentiality whilst in transit. This will become more common as network user become more security savvy [4]. On the other hand, the inability to view such traffic may lead to a network service provider incorrectly marking the traffic and negate their ability to correctly manage such traffic.

Much research was performed to detect applications using encryption, by building a set of characteristics based on behavioural analysis of each protocol. This system was used in conjunction with machine learning algorithms, which proved effective until faced with multiple flows running across an encrypted tunnel on a single quintuple. The dissemination of each flow from an encrypted tunnel stream proved difficult and accuracies varied. This problem was further compounded by the fact that some protocols exhibit the same behaviour as others, such as the case between file transfer protocol (FTP) and simple mail transfer protocol (SMTP). Other problems relate to one protocol being able to perform multiple functions, such as SSH remote shell and secure copy processes. Of all the research considered by the authors, no sure fire way of accurately detecting applications within tunnels was discovered. This translates to a reduction in management capabilities for network administrators faced with users who tunnel their applications across the network.

With the increase in popularity of the new IPv6 protocol, due to the exhaustion of IPv4 resources from RIRs, an increasing number of users will start creating tunnels to route new IPv6 address space over IPv4 infrastructure. Unless network service providers can get a handle on the identification of flows within such tunnels, the management of next generation service provider networks appears dismal at best.

References

[1] Y. Zhang, Z. Li, S. Mei, and C. Fu, "Session-based tunnel scheduling model in multi-link aggregate IPSec VPN," in *Third International Conference on Multimedia and Ubiquitous Engineering*, 2009.

[2] R. Alshammari and A. N. Zincir-Heywood, "A flow based approach for ssh traffic detection," in *Proceedings of the IEEE International Conference on System, Man and Cybernetics*. The IEEE Computer Society, 2007, pp. 296–301.

[3] R. Alshammari and A. Zincir-Heywood, "Investigating two different approaches for encrypted traffic classification," in *Sixth Annual Conference on Privacy, Security and Trust*. The IEEE Computer Society, 2008, pp. 156 – 166.

[4] C. V. Wright, F. Monrose, and G. M. Masson, "On inferring application protocol behaviors in encrypted network traffic," *Journal of Machine Learning Research*, vol. 7, pp. 2745–2769, 2006.

[5] M. Gebski, A. Penev, and R. K. Wong, "Protocol identification of encrypted network traffic," in *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society, 2006, pp. 957–960.

[6] R. Alshammari, A. N. Zincir-Heywood, and A. A. Farrag, "Performance comparison of four rule sets: An example for encrypted traffic classification," in *World Congress on Privacy, Security, Trust and the Management of e-Business*. The IEEE Computer Society, 2009, pp. 21–28.

[7] J. Moscola, J. Lockwood, R. P. Loui, and M. Pachos, "Implementation of a content-scanning module for an internet firewall," in *11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, 2003, pp. 31–38.

[8] K. Huang and D. Zhang, "A byte-filtered string matching algorithm for fast deep packet inspection," in *The 9th International Conference for Young Computer Scientists*. The IEEE Computer Society, 2008, pp. 2073 – 2078.

[9] P. Dorfinger, "Real-time detection of encrypted traffic based on entropy estimation," Master's thesis, Salzburg University of Applied Sciences, August 2010.

[10] "SNORT open source IDS/IPS." [Online]. Available: http://www.snort.org/

[11] K. K. Nam, "Analysis of bittorrent protocol and its effect on various networks," Simon Fraser University, Tech. Rep., 2011.

[12] "The secure shell (SSH) transport layer protocol - RFC 4253." [Online]. Available: http://www.ietf.org/rfc/rfc4253.txt

[13] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Lecture Notes in Computer Science*, vol. 904/1995, pp. 23–37, 1995.

[14] W. W. Cohen, "Fast effective rule induction," in *In Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995, pp. 115–123.

[15] J. Wroblewski, "Finding minimal reducts using genetic algorithms," in *Second Annual Join Conference on Information Sciences*. Warsaw University of Technology, 1995, pp. 186–189.

[16] "Projected RIR unallocated address pool exhaustion," 2011. [Online]. Available: http://www.potaroo.net/tools/ipv4/

[17] APNIC, "APNIC IPv4 address pool reaches final /8." [Online]. Available: http://www.apnic.net/publications/news/2011/final-8

[18] J. Tian and Z. Li, "The next generation internet protocol and its test," in *The IEEE International Conference on Communications*, vol. 1. The IEEE Computer Society, 2001, pp. 210 – 215.

[19] H. Electric, "Tunnelbroker service," 2011. [Online]. Available: http://www.tunnelbroker.net/usage/tunnels_by_country.php