# A NetFlow Scoring Framework For Incident Detection

Michael Sweeney*, Barry Irwin*†
*Department of Computer Science, Rhodes University, Grahamstown, South Africa
†Council for Scientific and Industrial Research (CSIR), Pretoria, South Africa
Email: g16s8255@campus.ru.ac.za, b.irwin@ru.ac.za

***Abstract*—As networks have grown, so has the data available for monitoring and security purposes. This increase in volume has raised significant challenges for administrators in terms of how to identify threats in amongst the large volumes of network traffic, a large part of which is often background noise. In this paper we propose a framework for scoring and coding NetFlow data with security related information. The scores and codes are added through the application of a series of independent tests, each of which may flag some form of suspicious behaviour. The cumulative effect of the scoring and coding raises the more serious potential threats to the fore, allowing for quick and effective investigation or action. The framework is presented along with a description of an implementation and some findings that uncover potentially malicious network traffic.**

## I. Introduction

The question of how to find and measure security related incidents has gained considerable attention in recent years. However, this has often lead to solutions that produced large amounts of seemingly valuable yet static output only useful to niche audiences [1]. Add to this the ever-increasing challenge of data volumes and we are presented with the proverbial problem of trying to find a needle in a haystack. Technologies such as NetFlow have in the past been found to be especially useful for the detection of DOS attacks, network scans, worms and botnets [2]. In addition, because organisations typically store flow data for some period of time the flows can assist in forensic investigations. The commonality of these attack forms is that they all affect network metrics such as flows, packet counts, byte counts, etc. But analysing and reporting on these attributes alone is not enough to give insight into all the attacks an organisation may experience.

NetFlow was originally developed at Cisco in 1996 as a packet switching technology [3]. Flow export technology is now well understood and has become widely used for security analysis, capacity planning, accounting and traffic profiling [2]. The data captured by NetFlow is exported as a flow [4] describing the packet data passing though the collector. A flow is identified by a tuple of 5 to 7 IP packet attributes. These are:

- IP source address
- IP destination address
- Source port
- Destination port
- Layer 3 protocol type
- Class of Service (optional)
- Router or switch interface (optional)

Various implementations of NetFlow may have additional information in the records. For example, routing information such as AS numbers and next hop, VLAN data, input/output interfaces etc. All packets with the same attributes are grouped together into a flow record along with the packets and bytes transmitted. It is important to note that flows are unidirectional, so the data is recorded for traffic flowing in one direction only. TCP or UDP conversations will result in two flow records being recorded. In high-volume environments the data may be sampled recording only a fixed percentage of the network traffic seen. A NetFlow deployment typically has three components [5]:

- *Flow capture and export*: These can be processes in network devices such as routers or switches or they can be applications deployed on a server. They capture network data traffic and export the flow information to collectors.
- *Flow collector*: Collectors are responsible for collecting, pre-processing and storing exported flow data from one or more flow exporters.
- *Flow analyser*: This component is responsible for process, summarising and reporting on the captured data.

The architecture allows for multiple flow exporters to be deployed across a network all sending data from different points on the network to a single collector. With the ever-increasing speeds in network interfaces, the analysis of NetFlow data for useful information is fast becoming a challenge.

This paper considers the challenges faced by network and security administrators when presented with large volumes of NefFlow output and no idea where to start. The challenge is that it is easy to find things you know about, but what about the things you don't know about but suspect are there - the "known unknowns" (with apologies to Donald Rumsfeld). It is the known unknowns that are of greatest concern to administrators and the hardest to find. The solution suggested in this paper is the implementation of a scoring system. Instead of a binary approach to threat identification that requires significant pre-knowlege and classifies traffic as a threat or non-threat, we propose a scaled view of threats where issues are viewed on a continuum and not in isolation. A series of tests are applied to flows, each of which by itself may not indicate serious problems, but as some flows fail and are scored higher and higher, the serious issues emerge. The application of many different tests also means that issues can be viewed in context. For example, network traffic on unknown ports may not be considered an issue, but when this traffic also includes high volumes of traffic from sites with low reputational scores, this may be an indication of something more serious.

### A. Structure

This paper is organised as follows: section II is a review of related research on NetFlow analysis; section III presents the proposed scoring framework; and section IV discusses an implementation of this framework along with the results and some findings. Finally, section V looks at future work and section VI concludes the paper.

## II. Related Work

In this section, we review the work that has been done in order to get insight into what approaches may be applicable to analyse. A common theme in prior research has been the use of blacklists and whitelists to preprocess the data (see Hofstede et al. [2], Verandi & Pihelgas [1] and Amini et al. [6]). Blacklists are lists of IP hosts suspected to be involved in malicious behaviours such as acting as botnet C&C nodes, known compromised node, known port scanners, etc. Often the blacklist may include reliability or reputation scores to give an indication of confidence in the hosts perceived threat. These lists are compiled and maintained by numerous security institutions such as Emerging Threats and AlienVault and are made available as regularly updated feeds. Whitelists are lists of IP hosts that are known to be reliable or common destinations that are not likely to be a risk. Examples of these are Google, Facebook and Amazon. The suggested approach is that flows with blacklisted hosts are marked as suspicious, while flows with whitelisted hosts can be ignored from any further investigation.

The use of traffic profiling for anomaly detection is discussed in [1] and [7]. This approach requires profiling either host or network traffic over a period of time and then using various methods for the detection of anomalies. This however requires that data that has been collected over a significant period of time in order to establish a baseline against which to look for deviations. Similar approaches look for network patterns on a much smaller scale. For example, scan detection can be done by counting distinct connection attempts made by a source within a particular time interval [8]. The performance and accuracy of these approaches depend on the time interval chosen. Probabilistic models have been used to try and improve this approach. Simple methods for detecting the spread of worms have also been used such as looking for unusual top-N connections or top-N bytes within a short time interval [9].

Flow data contains the TCP flags from the network conversation which allows for simple analysis to detect DOS attacks or scans. In [7] the authors describe the detection of DOS attacks by looking for unusual traffic targeting single hosts that has only the SYN or RST flag set. Flows with only the SYN flag set are typically the result of port scanning and can be recorded and reported on [6]. Finally, flows with illegal flag combinations (e.g. TCP FIN without an ACK) can also be detected and marked for further investigation [1]. A simple method of reducing the number of flows that require investigation is the filtering out of traffic to known hosts and ports [1]. An example of this is filtering out flows for traffic to and from port 25 from a known mail server from the dataset. Conversely, traffic to and from unknown ports or servers can be flagged for further investigation.

ICMP flow data can be used to detect scans or malicious connection attempts such as those caused by worms [9]. High numbers of ICMP destination unreachable or ICMP port unreachable messages to a host or hosts may indicate scanning or malicious connection attempts on a network. In [10] the authors propose an attack detection methodology whereby they compare flow characteristics to expected behaviour. In particular, they consider brute force SSH attacks. By taking into account the network conversation traits of a failed SSH login attempt (i.e. session setup, number of login attempts, etc) they are able to reliably detect brute force attacks by looking for large number of SSH flows which only have between 11 and 51 packets. In [11] the authors propose a similar approach for the detection of botnets.By taking into account the characteristics of botnets and the related C&C activity they propose monitoring for patterns that could indicate a potential infection. For example, observing a large number of DNS requests from many hosts at the same time followed by other synchronised network traffic from the same hosts could be strong indicator of the existence of a botnet on a network.

A number of approaches using scoring of hosts based on flow attributes for automated threat detection have been proposed. As early as 2004 the MINDS system was proposed with the aim of solving the dependency in pre-knowledge for intrusion detection [12]. At the time threat detection relied heavily on threat signatures and manual intervention. The MINDS system aimed to solve this problem through the automatic assigning of an anomaly score to network connections. The higher this score the more likely the connection was suspect. High ranking connections could then be assigned to a network operator for further investigation. In [13] the authors propose using a similar approach to scoring hosts on a large internal network in order to detect APTs. They apply various statistical algorithms that use flow attributes (specifically number of bytes sent, number of flows, number of destinations) to score a host over time with a specific focus on detecting data exfiltration. Commercial vendors such as Cisco's IronPort have also implemented host scoring for threat detection and applications that can work with external reputation feeds to aid in scoring and alerting [14].

## III. Proposed Scoring Framework

As can be seen in the previous section there are many approaches to threat detection using NetFlow data. Rather than pick one or two and attempt to implement them, a decision was made to try a variation on the data scoring approach. The aim of this new approach was a framework that could be used for identifying malicious traffic without any prior knowledge of what is being looked for. Each flow is scored in two ways: a goodness factor and a badness factor. These are purposefully kept as two individual scores in order to allow each attribute to be separately recorded and measured. This separation also allows for the identification of attacks against trusted hosts. For example if a host is involved in many good flows and then suddenly a burst of bad flows is detected, alarms can be raised.

It is important to note that flows are scored rather than hosts. A host's reputation may contribute to a flow score (for example, if it appears on a blacklist), however, the scores are applied to the flows. This distinction is important because it is the attributes of the flow or flows that we will use to identify suspicious traffic. In this model, a flow is scored by viewing its attributes from many different angles.

'*Goodness*' refers to how likely a flow is to be a normal network conversation, while '*badness*' is an indication of how likely a flow is to be suspect. By applying many different tests to flows and scoring them accordingly, an incremental measure of goodness or badness can be established. In addition to scoring, a record is kept on each flow of the tests which resulted in a score being increased. This coding of a flow score provides additional information that can be used to filter out different classes of malicious traffic. As more tests are applied the more suspect flows (with the highest badness score and lowest goodness score) can be easily identified and investigated. Different tests may increase a score by different amounts. For example, a SYN only flow may increase a badness score by 10 points, while a network flow between unknown ports may only score a five in terms of badness. The cumulative nature of the scoring and coding allows for information from multiple disparate sources to contribute to a score, thereby increasing the confidence levels in the outcome. Badness can also be used as a filter to exclude flows from further investigation. For example, all scanning traffic can be removed in order to focus on DOS activity. For the implementation of the scoring four attributes were added to each flow record:

- *Goodness Score*: an integer value used to score how 'good' a flow is.
- *Badness Score*: an integer value used to score how 'bad' a flow is.
- *Goodness Detail*: a list of reason codes that indicate which tests contributed to the goodness score.
- *Badness Detail*: a list of reason codes that indicate which tests contributed to the badness score.

To illustrate how the scoring works consider a test that first checks for flows with only the SYN TCP flag set. The test has a badness value of 10 and a code of 'SYN_ONLY'. All flows that match the condition have their badness score increased by 10 and the code value appended to the badness detail attribute. As flows match test conditions their scores will go up with the detail attributes listing the tests that the flow matched. For example, a flow that matches the SYN only test with one of the hosts found in a threat intelligent blacklist would end up with a badness score of 20 and a detail string of 'SYN_ONLY ALIEN_VAULT'.

## IV. Implementation

An implementation of the scoring framework was undertaken and applied to a sample data set and the resulting scores reviewed. Based on the outcomes of the scoring a sample of score profiles was investigated in order to identify possible security related issues.

### A. Test Dataset

A sample of data was used from an small network, consisting of 60 active hosts on a /24 netblock consisting of mostly servers providing caching, mail, DNS and IP gateway services to a number of schools in the Eastern Cape. Two days worth of NetFlow traffic was collected from October and November 2016 totaling just over 37 million flows. The flow data was segmented into inbound, outbound and internal traffic (where the 'inside' network was considered to be the /24 netblock). Only inbound traffic was scored as this was assumed to be most likely to contain potentially malicious traffic. The scored data set consisted of 17.8 million rows of incoming traffic the majority of which was TCP (72%) followed by UDP (27%) and ICMP (1%).

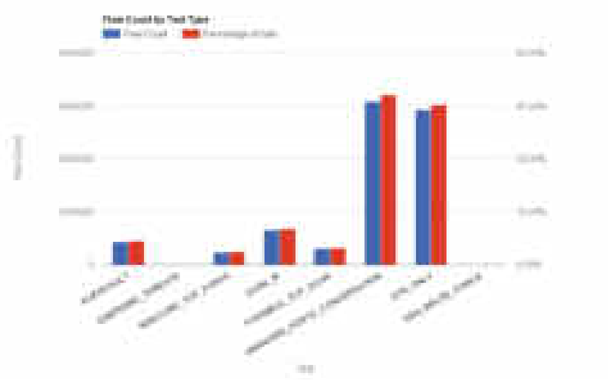

Fig. 1. Score Reason Code Totals

### B. Scoring Model

In order to test the scoring, a number of different classes of tests were applied to the flows. The first class of tests, shown in table I used reputation data to score the flows. One source was used for whitelisting and two for blacklisting. The next class of scoring, shown in table II, used features of TCP/IP in order to score the flows. Finally, the last scoring used specialised tests to score the data (table III). Because the tests are very specific, any flows that match are given a high score as a result.

Other than the last test (POSSIBLE_TCP_SCAN) all of the tests are executed on a per process level and could be done as part of regular flow processing. The custom TCP scan algorithm requires a large set of data to work on and flags a group of flows when the conditions match, making it more suitable for periodic batch processing. As can be seen, the emphasis in the scoring was on badness. This was done to see how effective the approach was in identifying possible threats. Once the tests had been applied and the flows scored, the flow data was analysed using the scoring data to look for suspicious traffic.

### C. Score Outcomes

Table IV presents the top 10 count of flows and their detail for all flows with a badness score greater than 0. While not deemed a serious threat, the number of incomplete flows on unknown ports stands out as a significant issue. The AlienVault blacklist scoring appears to have a lot more impact than the Emerging Threats one and traffic flows to dark IP address space is also prominent in the flow list.

In table V the reason codes of the scoring are presented, along with the top 10 flow count for all flows scoring a badness of 50 or higher. From the first entry in this table we can see strong evidence of TCP port scanning. A total of 317,380 flows are directed at non-existent hosts, at insecure TCP ports, TCP SYN only, and the custom flow matches the scan algorithm. It is also interesting to note that all top 10 entries in the list contain some indication of TCP port scanning. The custom SSH brute force test has also featured highly in the results coming in at number 6.

Figure 2 shows the top sources for the SYN only traffic.

Figure 1 breaks down the flow count of each badness reason code. The flow counts are mutually exclusive in that the same flow may be counted multiple times due to failing a number of tests. From the chart we note that unknown TCP conversations and flows with only the TCP SYN flag set each account for

TABLE I
List based scores

| Class | Code | Score | Notes |
|---|---|---|---|
| *Goodness* | NDPI_IP | 10 | A list of known hosts were extracted from the nDPI source code. This is a open source deep packet inspection library which uses, amongst other things, a list of known networks to identify traffic. The list includes netblocks for Google, Twitter, etc. Flows involving any of these hosts had their goodness score increased by 10 [15]. |
| *Badness* | ALIENVAULT | 10 | IP addresses from the AlienVault threat intelligence database was used to flag flows with a badness score of 10 [16]. |
| *Badness* | EMERGING_ THREATS | 10 | IP addresses from the AlienVault threat intelligence database was used to flag flows with a badness score of 10 [17] |

TABLE II
IP Feature based scores

| Class | Code | Score | Notes |
|---|---|---|---|
| badness | INSECURE_TCP _PORTS | 10 | Any flow that included traffic to a list of TCP or UDP ports deemed to be insecure protocols (e.g. telnet, ftp, rsh, etc) was scored as bad. See [18] for examples. |
| badness | DARK_IP | 10 | Traffic directed at dark IP addresses on the internal network was considered to be suspicious and scored as bad. In the initial data analysis phase it was determined that there was only 60 active hosts on the internal network. The rest of the /24 netblock was dark IP address space. |
| badness | UNKNOWN_ PORTS _CONVERSATION | 5 | Any traffic flows between unknown or unassigned source and destination TCP or UDP ports was flagged as an anomaly and scored as bad. The list of known ports was taken from the services file on an Ubuntu server. A low badness score was used, as traffic of this nature is not always an issue. |
| badness | SYN_ONLY | 10 | . |

TABLE III
Custom scoring

| Class | Code | Score | Notes |
|---|---|---|---|
| badness | SSH_BRUTE_ FORCE | 20 | By applying a simple version of the methodology described in [10] possible SSH brute force attacks are scored. The test looks for any flows to port 22 with between 11 and 51 packets in the flow. |
| badness | POSSIBLE_TCP_ SCAN | 20 | For this test flows are grouped by destination port and counted along with a average number of packets per flow. Any group with more than 1000 flows and less than 4 packets per flow was considered to indicate possible scanning of the destination port associated with the group. The reasoning being that a large number of flows to a port with very few packets per flow was an indication of general scanning to that port. |

| Coding Detail | Flow Count |
|---|---|
| UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 3,579,403 |
| UNKNOWN_PORTS_CONVERSATION | 1,131,551 |
| DARK_IP UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 830,488 |
| ALIENVAULT | 784,267 |
| ALIENVAULT UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 770,071 |
| DARK_IP INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 317,380 |
| INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 98,835 |
| POSSIBLE_TCP_SCAN UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 60,206 |
| DARK_IP POSSIBLE_TCP_SCAN UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 51,958 |
| ALIENVAULT DARK_IP INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 41,034 |

TABLE IV
Flow counts by category

more than 50% of the total inbound traffic. The custom TCP scan detection algorithm has only flagged 5% of flows as possible scans in contrast to the SYN test.

Figure 3 compares the top destination ports for the flows identified as possible scans using both the SYN only approach and the custom scan detection algorithm. It is interesting to note that while the the total ports flagged in each case are

| Scores 50 and Higher with Coding | |
|---|---|
| Coding Detail | Flow Count |
| DARK_IP INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 317,380 |
| ALIENVAULT DARK_IP INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 41,034 |
| ALIENVAULT INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 12,914 |
| ALIENVAULT DARK_IP POSSIBLE_TCP_SCAN UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 6,926 |
| ALIENVAULT DARK_IP POSSIBLE_TCP_SCAN SYN_ONLY | 4,599 |
| ALIENVAULT POSSIBLE_TCP_SCAN SSH_BRUTE_FORCE | 2,251 |
| EMERGING_THREATS DARK_IP POSSIBLE_TCP_SCAN SYN_ONLY | 851 |
| ALIENVAULT EMERGING_THREATS DARK_IP POSSIBLE_TCP_SCAN UNKNOWN_PORTS_CONVERSATION SYN_ONLY | 647 |
| ALIENVAULT EMERGING_THREATS DARK_IP INSECURE_TCP_PORTS POSSIBLE_TCP_SCAN SYN_ONLY | 496 |
| ALIENVAULT EMERGING_THREATS DARK_IP POSSIBLE_TCP_SCAN SYN_ONLY | 381 |

TABLE V
Flow counts by category with badness score 50 and higher (Top 10)

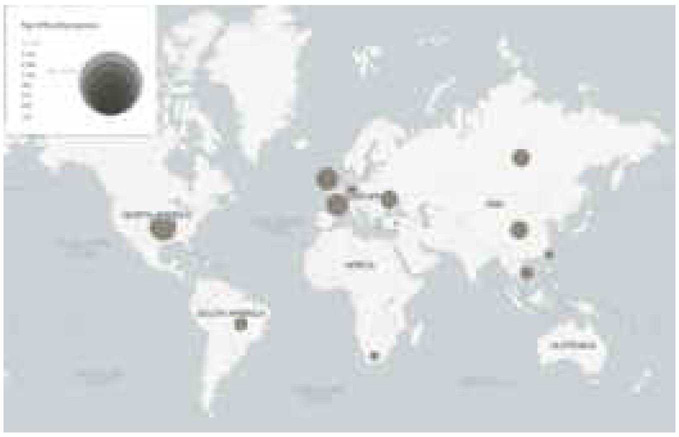

Fig. 2. Top Sources for SYN Only Flows

significantly different in number, there is a lot of commonality in the top 10 scanned ports identified. Reviewing the flows in detail we find that a large number of the SYN only flows appears to be non-malicious traffic sent in low volumes from a large number of hosts across a wide range of ports making it unlikely to be a targeted scan. This indicates that a SYN only test has a large number of false positives. The custom algorithm by its nature looks for a large number of flows of a specific type and therefore has a different profile. The overlap of the two tests has the potential to be the strongest indicator of focused scanning.

### *D. Sample Findings*

The four potential threats identified through the scoring process are:

- Highest scoring flows. These have the highest badness and therefore are obvious candidates for investigation.
- Potential TCP scans of insecure ports. Two different scoring approaches have identified evidence of scanning of the network.
- High number of TCP conversations between unknown source and destination ports. This bears investigation due to the large flow counts.
- Potential SSH brute force attempts to test the custom scoring approach.

*1) Top Score:* A total of 496 flows scored a badness of 70. Looking at the flows in question we note that both reputational tests have contributed to their badness scores as a result (ALIENVAULT and EMERGING_THREATS). Looking at the source IP addresses of these flows, we find that 468 of the flows are from two hosts. Both hosts are from Chinese IP addresses and more specifically from the securityresearch.360.cn domain. The ports in question are MSQL (1433) and MySQL (3306). By widening our analysis to include all traffic from these hosts we find that they are scanning 13 different ports on the network (81, 443, 1080, 1433, 3306, 3306, 3389, 3389, 4899, 5800, 8009, 8081, 27017).

*2) TCP Scans:* By searching for all inbound flows with the POSSIBLE_TCP_SCAN and SYN_ONLY score code and grouping by destination port, it was found by far the majority of flows (443,187) were for traffic to port 23 (Telnet). Looking at the target IP addresses it was found that each of the 253 addresses on the internal network had on average 1,753 connection attempts. A maximum of 1,979 attempts and minimum of 1,627 indicated that the scan was fairly evenly spread across the internal network. Looking at the source host's IP address the traffic was found to originate from 253,788 unique hosts with the majority of source hosts being participants in under 10 flows. Looking at source ASNs it was found that the top three sources of the scans, accounting for 30% of the total flows, were a Chinese ISP, a Taiwanese ISP and a Vietnamese ISP. The source organisation of these flows, the large number of sources of the flows, the target port and the dates led to the conclusion that this traffic profile matched that of the the Mirai botnet as described in [19] and [20]. Further investigation revealed 68,002 flows to port 2323 that appeared to be connection attempts related to Mirai.

*3) Unassigned TCP Port Conversations:* Although it is not unusual to have traffic flows between two non-assigned TCP ports an unusually high number of flows were identified to a single port on a host on the local network. A total of 700,768 flows were found from the outside network to port 45554 on the inside network. Of these 698,382 were to 196.21.242.189 (ap-music.kc.ecap.eschool.za). These flows accounted for over two gigabytes of outbound traffic and almost 500 megabytes of inbound traffic. The flows originated from 6,195 different unique hosts. The only possible explanation for this traffic comes from an entry found on a blog advertising open public SOCKS proxies (https://b0ylasr0.blogspot.co.za/2016/11/vip-socks-5-free-05112016_80.html). The host and port number in question is listed a post dated 5th November 2016. Without knowledge of whether or not this was an approved service, it should probably be investigated further.

*4) SSH Brute Force Attempts:* Looking at the hosts on the network flagged as potential targets of SSH brute force attacks (SSH_BRUTE _FORCE code) it was found that three hosts where the target of the attacks, 196.21.242.130 (gateway.sdpschool.org), 196.21.242.93 (kcweb.async.org.za) and 196.21.242.68 (lair.moria.org). The last host was targeted more than the first two combined (1,264 flows). Over half of the attempts on 196.21.242.68 came from just two IP addresses located in China.

## V. Conclusion

On the one hand NetFlow provides a rich source of information, while on the other it can also be seen as a source of too much information. Because of the sheer number of permutations possible in the attributes that identify a unique flow, the number of flows in a data set can be overwhelming and finding points of interest can be a challenge. Researchers in the field have taken numerous approaches, but generally they tend to focus on identifying specific threats or classes of threats. In this work we have taken a different approach and proposed a general scoring framework that allows for the application of a number of diverse methodologies for threat detection and then applying incremental scoring and coding of flows based on the outcome of the test. A benefit of this approach is that many different potential threat identification approaches can be combined in order to identify high-risk flows with a strong level of confidence. In addition, the scoring has allowed us to apply and compare different detection methods in order to filter out false positives as in the case of potential scans. Finally, the reason codes and score can be used to remove known but uninteresting data, allowing us to focus on more unusual events. The approach has shown

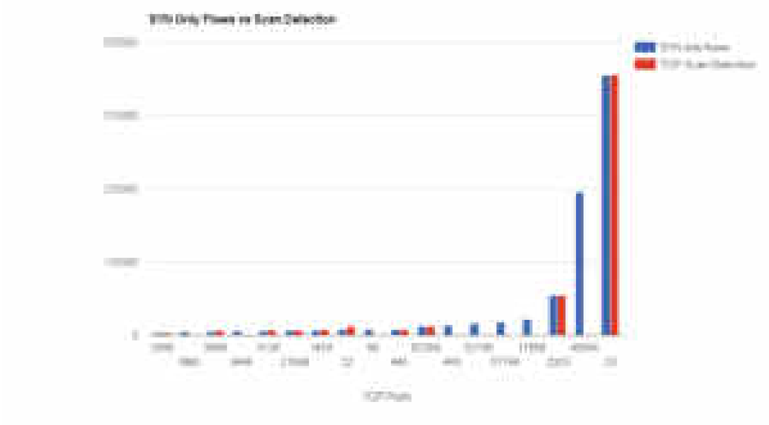

Fig. 3. TCP Scan Detection vs SYN only - Ports

promise with a limited data set and a small number of simple tests and has set the groundwork for further research into the application of the framework.

## A. Further Work

A number of potential improvements to the framework present opportunities for further work. These include realtime scoring, champion-challenger approaches to refining the scoring, the addition of more sophisticated scoring criteria and the use of machine learning to evaluate scores and reason codes to more readily identify threats.

Most of the tests presented are applied on a per flow basis and are therefore suitable for realtime scoring. There is scope for implementation of an event stream processing solution that can analyse flows as they are collected applying scoring and reason codes in realtime using a pipeline of event processing nodes before persisting the flows.

Champion/challenger testing is an approach using the financial world to test new models. This approach involves applying new strategies to evaluating data by comparing the outcome of the new approach against the old one. In practice the new strategy is applied to a subset of the data and compared to the existing approach. If the new one performs better, it replaces the old one. This can be applied to the NetFlow scoring framework by re-scoring old data with new methodologies and comparing the outcomes in terms of anomalies found.

The current set of tests are simplistic in nature. As seen in the prior work section, there are many different approaches to threat detection many of which could and should be included in the framework.

Finally, there is an opportunity to introduce a machine learning component to analyse the scores and reason codes in order to identify the combinations that are most likely to represent threats or non-issues.

## REFERENCES

[1] R. Vaarandi and M. Pihelgas, "Using security logs for collecting and reporting technical security metrics," *Proceedings - IEEE Military Communications Conference MILCOM*, pp. 294–299, 2014.

[2] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras, "Flow Monitoring Explained : From Packet Capture to Data Analysis with NetFlow and IPFIX," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, 2014.

[3] D. R. Kerr and B. L. Bruins, "Network flow switching and flow data export," 1996.

[4] Cisco, "Introduction to Cisco IOS NetFlow - A Technical Overview," http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html, 2012, [Accessed:2016-12-09].

[5] ——, "NetFlow Collector User Guide," http://www.cisco.com/c/en/us/td/docs/net_mgmt/netflow_collection_engine/6-0/tier_one/user/guide/user/overview.html#wp1038820, 2014, [Accessed:2016-12-09].

[6] P. Amini, R. Azmi, and M. Araghizadeh, "Botnet Detection using NetFlow and Clustering," *Advances in Computer Science: an ...*, vol. 3, no. 2, pp. 139–149, 2014.

[7] S. Choudhary, "Usage of Netflow in Security and Monitoring of Computer Networks," *Interntional Journal of Computer Applications*, vol. 68, no. 24, pp. 17–24, 2013.

[8] P. Chandrashekar, S. Dara, and V. Muralidhara, "Feasibility Study of Port Scan Detection on Encrypted Data," *2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, no. November, pp. 109–112, 2015.

[9] M. Khule, M. Singh, and D. Kulhare, "Enhanced Worms Detection By NetFlow," vol. 3, no. 3, pp. 5123–5127, 2014.

[10] R. Hofstede and A. Sperotto, "SSH Compromise Detection using NetFlow / IPFIX," *ACM SIGCOMM Computer Communication Review*, 2014.

[11] V. KrmiCek, "Inspecting DNS Flow Traffic for Purposes of Botnet Detection," pp. 1–9, 2011.

[12] L. Ertoz, E. Eilertson, A. Lazarevic, P.-n. Tan, V. Kumar, J. Srivastava, and P. Dokas, "Minds-minnesota intrusion detection system," *Next Generation Data Mining*, pp. 199–218, 2004.

[13] M. Marchetti, F. Pierazzi, M. Colajanni, and A. Guido, "Analysis of high volumes of network traffic for Advanced Persistent Threat detection," *Computer Networks*, vol. 0, pp. 1–15, 2016.

[14] M. Schiffman, "Correlating NetFlow Data for Proactive Security: Network Notoriety," http://blogs.cisco.com/security/correlating-netflow-data-for-proactive-security-network-notoriety, 2012, [Accessed:2016-12-04].

[15] NDPI. (2016) nDPI Source. https://github.com/ntop/nDPI/blob/dev/src/lib/ndpi_content_match.c.inc. [Accessed:2016-12-01].

[16] AlienVault, "IP Reputation Explained," https://www.alienvault.com/documentation/usm-v5/kb/2015/05/ip-reputation-explained.htm, 2015, [Accessed:2016-12-01].

[17] Emerging Threats, "Emerging Threats," http://rules.emergingthreats.net/fwrules/, 2016, [Accessed:2016-12-01].

[18] D. Mashburn and R. VandenBrink, "NetFlow Collection and Analysis Using NFCAPD, Python, and Splunk," Tech. Rep., 2015.

[19] R. Dobbins, "Mirai IoT Botnet Description and DDoS Attack Mitigation," https://www.arbornetworks.com/blog/asert/mirai-iot-botnet-description-ddos-attack-mitigation/, 2016, [Accessed:2016-12-10].

[20] B. Herzberg, D. Bekerman, and I. Zeifman, "Breaking Down Mirai: An IoT DDoS Botnet Analysis," https://www.incapsula.com/blog/malware-analysis-mirai-ddos-botnet.html, 2016, [Accessed:2016-12-10].

**Michael Sweeny** is completing his Masters at Rhodes University in the field of network security with a focus on netflow processing.