Research Journal of Information Technology

ISSN 1815-7432

Academic Journals Inc.

www.academicjournals.com

OPEN ACCESS Research Journal of Information Technology

ISSN 1815-7432
DOI: 10.3923/rjit.2016.55.65

Research Article

Conditional Hybrid Approach for Intrusion Detection

H. Alaidaros and M. Mahmuddin

Department of Computer Science, College of Arts and Sciences, University Utara Malaysia, 06010 UUM Sintok, Kedah, Malaysia

Abstract

Background and Objective: Inspecting all packets to detect intrusions faces challenges when coping with a high volume of traffic. Packet-based detection processes every payload on the wire, which degrades the performance of intrusion-detection systems. This issue requires the introduction of a flow-based IDS approach that reduces the amount of data to be processed by examining aggregated information of related packets in the form of flow. However, flow-based detection still suffers from the generation of false positive alerts due to lack of completed data input. This study proposed a model to improve packet-based performance and reduce flow-based false positive rate by combining flow-based with packet-based detection to compensate for their mutual shortcomings. This proposed model is named as conditional hybrid intrusion detection. **Materials and Methods:** In this model, only malicious flows marked by flow-based must be further analyzed by packet-based detection. For packet-based detection to communicate with flow-based detection, input framework approach was used. To evaluate the proposed detection methods, public datasets were replayed in different traffic rates into both the proposed method and default Bro implementations in a testbed controlled environment. **Results:** Experimental evaluation shows that the proposed approach was able to detect all infected hosts reported and corresponding datasets. At 200 Mbps rate, proposed approach can save 50.6% of memory and 18.1% of CPU usage compared with default Bro packet-based detection. Experiments demonstrated that the default Bro packet-based can handle bandwidth up to 100 Mbps without packets drop, while 200 Mbps in the proposed approach. **Conclusion:** Experimental evaluation showed that the proposed model gains a significant performance improvement, in term of resource consumption and packet drop rate compared with a default Bro packet-based detection implementation. The proposed approach can mitigate the false positive rate of flow-based detection and reduce the resource consumption of packet-based detection, while preserving detection accuracy. This study can be considered as skeleton model to be applied for intrusion or monitoring detection systems.

Key words: Flow-based detection, packet-based detection, Bro IDS, false positive, input framework

Received: July 15, 2016 Accepted: August 08, 2016 Published: September 15, 2016

Citation: H. Alaidaros and M. Mahmuddin, 2016. Conditional hybrid approach for intrusion detection. Res. J. Inform. Technol., 8: 55-65.

Corresponding Author: H. Alaidaros, Department of Computer Science, College of Arts and Sciences, University Utara Malaysia, 06010 UUM Sintok, Kedah, Malaysia

INTRODUCTION

Fast increase of devices, users and services connected to the internet has resulted serious threats against the security countermeasures such as network Intrusion Detection Systems (IDSs). Inspecting every packet to detect intrusions faces challenges when coping with a high volume of traffic. As stated by Liao *et al.*[1], packet-based detection degrades the performance of intrusion-detection systems. This challenge requires the introduction of a flow-based IDS approach that reduces the amount of data to be processed by examining only aggregated information of related packets in the form of flow. According to Golling *et al.*[2], since flow-based detection has lack of complete processed information, it reports significant number of false positive alerts. However, enhancing the detection algorithm accuracy of IDS has been a hot issue in study, while relatively less for reducing the false positive[3]. Several studies intended to reduce false positive rates via variety approaches. Such approaches include specializing NIDS to detect certain types of attacks and signature[4], building profile for understanding the environment of the networks[5-7], cleaning and filtering traffic to relevant data[8,9], verifying and correlating alerts[10,11] and combining signature-based and anomaly-based using packet-based only[12-14].

Limmer and Dressler[15] proposed a new monitoring technique that combines the flow records and their corresponding (part of) payloads in pre-processing stage. However, false positive rate are not studied in their analysis. In addition, in their approach, every portion payload of every flow, whether suspicious or not is processed by signature-base, which means more consumption is needed for processing. Another study involve in combining flow-based and packet-based was conducted by Schaffrath and Stiller[16]. The researchers developed only a conceptual framework for combining the approaches to to reduce the amount of packets to be processed by NIDS and enhance alert confidence. Golling *et al.*[2] also attempt to take the advantages of combining flow-based and packet-detection. However, no results were reported since the implementation is under deployment. This study proposed a model to reduce the false positive rate of flow-based detection by combining flow-based with packet-based detection to compensate for their mutual drawbacks. In other words, only packets corresponding to the alerts generated by flow-based detection are further analyzed by packet-based detection. This combination approach can reduce resource consumption of packet-based detection while preserving detection accuracy. Open-source Bro IDS[17] for intrusion detection is used for each flow-based and packet-based detection method. For packet-based detection to communicate with flow-based detection, input framework which is provided by the Bro is used.

The IRC-bot and P2P-bot scenarios for the proposed model are implemented. To evaluate the proposed detection method, different recent labeled datasets that represent real traffic are replayed into Bro with different traffic rates ranging from 100-1000 Mbps. The experimental evaluation shows that the hybrid approach gains a significant performance improvement when using input framework and BPF techniques compared with default Bro packet-based detection implementation.

MATERIALS AND METHODS

Packet-based and flow-based overview: In packet-based, a detecting engine has to inspect per packet header and payload in order to determine the existence of intrusion. This approach is mostly used by signature-based IDS which compares the incoming traffic to the given list of signatures in the database. On the other hand, a flow-based IDS does not inspect payload content for inspection and analysis, however, it depend on information and statistics of network flows. Such information includes number of packets and bytes transferred over a particular time and duration of a flow.

A flow can be defined as a unidirectional data stream between two nodes where all transmitted packets of this stream share the following 5-tubles; source and destination IP address and source and destination port number and protocol type[18]. This approach is mostly used by statistical and anomaly based IDS which identifies any significant deviations fall against the predefined normal profile or threshold. Nowadays, routers are equipped with ability to be configured to generate flow statistics records in form of so called netflow[18].

Promising results are achieved by researchers to detect attacks (such as denial of service, worms, SSH etc.) with focusing only on flow-based detection[19]. Table 1 summarizes the comparison between packet-based and flow-based detection. Alaidaros *et al.*[20] presented more details and overview on the comparison between these approaches.

Proposed approach: This explains how the combination of two approaches, flow detection and packet detection can reduce false positives of flow-based detection and reduce the resource consumption of packet-based detection while maintaining the level of detection accuracy. The idea of the proposed approach design is to obtain the advantages of a flow-based NIDS approach by having a small amount of data to be processed and the advantages of a packet-based NIDS approach by having a higher detection accuracy rate. With these two approaches, the model is based on the following strategy: One approach is used for first inspection (to mark traffic as suspicious) and the second one is used for further inspection to confirm the decision made by the first inspection.

In the combined approach theory, two candidates are possible. The first candidate is to set flow-based detection as first inspector and packet-based as second inspector. The second candidate is the opposite. The first approach is anticipated to have better scalability and alert verification level. On the other side, when placing packet-based detection in the second inspector, low scalability level might occur with low alert verification from flow-based detection. Thus, the first approach is chosen as candidate for the combined approach and named as Conditional Hybrid Intrusion Detection (CHID).

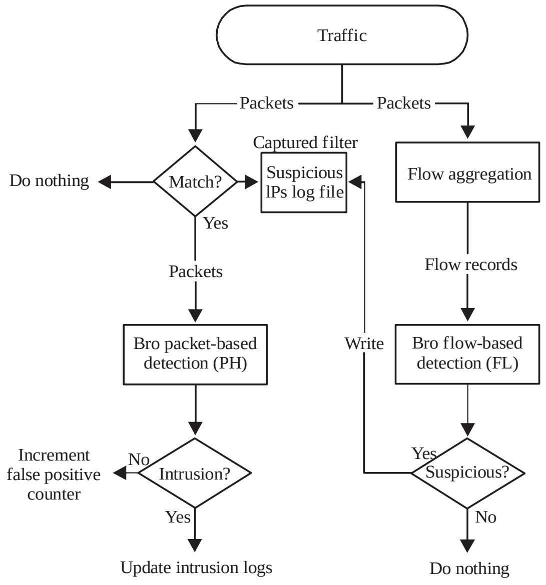

Fig. 1: Conditional Hybrid Intrusion Detection (CHID) approach. This method can reduce the false positive alerts generated from flow-based detection. When flow-based detection generates suspicious IP addresses, these IP addresses are added into the capture filter so that from now on, only incoming traffic that matches these suspicious IP addresses is subject to inspection by packet-based detection for further analysis

Design: The practical requirements for designing and operating the CHID approach are presented. The goals of this approach are to verify alerts that are produced by flow-based detection and to reduce the resource consumption of packet-based detection. Bro is used as platform for detection system. This approach is designed in conditional combination. In other words, only packets corresponding to (or triggered) the flow alerts are retrieved for further inspection by packet-based detection. In fact, repetitive attacks frequently send similar traffic and patterns in future connections[21]. This result was verified for IRC-bot and P2P-bot traffic in the experiments. Moreover, this result implies that these malicious activities can be detected and verified by inspecting their future packets using packet-based detection. With this fact in mind, these activity characteristics are utilized in order to verify the flow-based alerts by inspecting related packets in the corresponding future traffic.

Figure 1 illustrates how the CHID approach works. Initially, Bro in packet-based is adjusted with the Berkeley Packet Filter (BPF) to exclude all traffic. Later, when flow-based detection generates suspicious IP addresses, the packet-based detection adds these IP addresses into the capture filter so that from now on, only incoming traffic that matches these suspicious IP addresses is subject to inspection by packet-based detection for further analysis. If packet detection finds an intrusion, it updates the intrusion log file. Bro is used in the implementation. Bro is a unix-based, open-source network intrusion-detection system that monitors, analyzes and inspects all traffic to detect suspicious activity, even in high-speed networks. Henceforth in the remaining of this study, PH represents packet-based detection in CHID, FL represents flow-based detection and PO represents the default packet-based only detection that inspects all packets.

Table 1: Comparison between packet-based and flow-based detection system

Comparison items	Packet-based IDS	Flow-based IDS
Data to be analyzed	Header and payload	Flow records
Size of data to be analyzed	Mostly large	Small
Detection method mostly used	Signature-based	Anomaly-based
Size of network preferred	Small network	Small and large network
Extra device needed	No need extra device	Extra device is required to reform the traffic
Delay before detection analysis	No delay	Has delay since packets need to be aggregated
Allow to access raw packet data for further analysis	Can access raw packet	Cannot access raw packet
Privacy	Confidential data in payload is read	Confidential data in payload is not available

Implementation: The communication mechanism between PH and suspicious flow information that is generated from flow-based detection (FL)are presented. Filtering mechanism in the PH is also explained in this section. Basically, when FL detects suspicious flows, it writes corresponding IP addresses into a log file. In the log file, every entry has an expiry time. Once this expiry time is reached, the entry (suspicious IP address) is deleted. This process ensures that entries are updated with only recent suspicious traffic. To ensure that PH is only capturing the suspicious IP addresses, PH should communicate with FL to read the suspicious log file. Although broccoli API can be used for this purpose, because it subscribes to events of other Bro instances, its high overhead consumption is reported. Alternatively, the most recent novel framework developed by the Bro team, which is called "Input framework[22]" is used in this study. Input framework generally integrates external information in real-time into an IDS source without negatively affecting the IDS's main task, even in high-volume environments. For CHID case, it allows PH Bro instance to import the suspicious log file from FL Bro instance and store it as a table.

Because the suspicious log file is in dynamic mode-which means it is continuously being changed by the addition of new suspicious IP addresses-a re-reading mechanism is required to update the table in PH. Input framework provides a file reader that can read a file once at startup and then continuously monitor it for any changes and trigger the update operation automatically. In other words, input framework performs a modification by (1) Adding any values to the corresponding table, (2) Updating any entries whose values have changed and (3) removing entries that no longer exist.

Figure 2 illustrates how the filter reads the suspicious log file through input framework communication. To minimize PH Bro consumption, this re-reading process should be called only when there is a new entry added into the log file. The advantage of using input framework for re-reading processes is avoiding PH Bro instances being launched multiple times to update the table, causing extra overhead. After input framework updates its table, PH Bro captures the packets that correspond to the IP addresses found in the table. When the table is empty, BPF excludes all traffic. The BPF filter frequently reads the entries from the table and updates the filter to add them to the capturing process. To perform this function, compilation is needed. To avoid unnecessary filter compiling overhead occurring when updating the filter, this update (or filter re-compilation) should occur only when the table is changed. In order to implement CHID approach, attack scenarios should be selected and FL and PH should be individually implemented which are explained in the following sections.

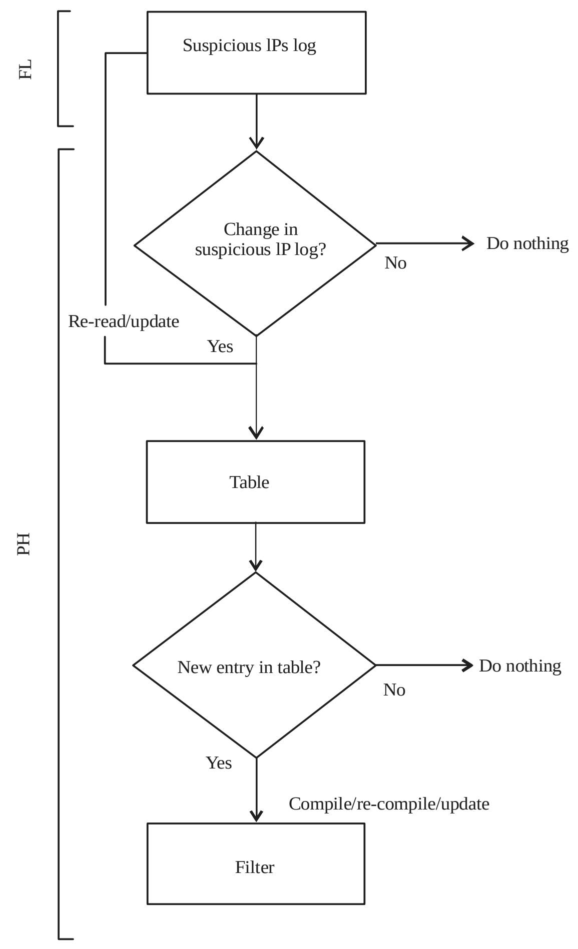

Fig. 2: Input framework as intermediate between suspicious log and filter. To study illustrates how the filter reads the suspicious log file through input framework communication. To minimize PH Bro consumption, the re-reading process should be called only when there is a change in suspicious log file. After input framework updates the table, filter captures the packets that correspond to the IP addresses found in the table. The FL represents flow-based detection while PH represents packet-based in CHID approach

Attack scenarios: To evaluate the proposed model, attack scenarios that fit the CHID model should be prepared. Based on literature, flow-based detection yields promising results when detecting botnet activities that perform repetitive traffic patterns[23]. Repetitive attacks imply that the attacks generate similar traffic patterns in future connections. With these points in mind and rather than review general botnet detection in this study, the following bot-related malicious types are considered in this study: IRC-bot and P2P-bot. These type are the most popular active botnets. More details on the characteristics of these malicious activities are presented by Zhu *et al.*[21].

The other reason behind selecting these attacks is that they match the model strategy. Thus, an attack can be marked as suspicious in the first inspector (flow-based detection or FL) and can be detected and verified in the second inspector (packet-based detection or PH), which can access data (payloads) not available in FL. However, in this study, flow-based detection scripts for IRC-bot and P2P-bot were adopted from the previous study[24]. For packet-based detection scripts, rules obtained from literature[25,26] are combined to form "Signatures". For example, one of the common payload signature for P2P-bots is such information form as "*.mpg; size = *", where, * represent decimal numbers[27]. These signatures are then converted into Bro format syntax. Basically, payloads of incoming packets are compared with these signatures. If these payload match with these signatures, then it is considered as intrusion. Signature engine of Bro provides high-performance pattern matching that is separately from the normal script processing. In addition, Bro provides default and built-in signature-based detection scripts[28] which are useful for the implementations.

IRC-bot scenario: With reference to IRC-bot, flow-based includes significant information about IRC and IRC-bot connections[21]. An IRC connection such as a ping-pong message exchange is easily identified in flow-level information. A ping-pong message or so-called keep-alive function is used by an IRC server to see whether the user computer is alive. This ping-pong message generally holds a fixed amount of packets and bytes per flow. In addition, typically no messages are exchanged when a bot is communicating with the controller other than ping-pong messages because IRC-bot is mostly in idle mode; it usually waits for commands from the controller. The absence of any messages other than ping-pong messages could be a significant indication of the existence of an IRC-bot. For packet level or PH, the payload information for IRC-bot also contains potential information for detecting such attacks. With this information, it is possible to identify whether an IRC connection is used for benign communication such as chatting or for malicious communication such as connecting to an IRC-bot command and control server.

P2P-bot scenario: Similarly to IRC-bot, P2P-bot communication can be marked by FL as suspicious and then detected by PH. However, in P2P-bots, peers do not receive commands from a central server as they do in IRC-bot; instead, they receive commands from peers. Based on the experiments, P2P-bot involves higher numbers of connections between peers compared with the normal communication in P2P such as e-Donky and bit-torrent applications[21]. A higher number of connections means higher flows occur in P2P-bot, which can be easily marked by an FL process. For PH, payloads include significant information for P2P-bot intrusion detection. For example, in command and control (C and C) communication, a payload could contain the instructions (e.g., what task to perform) sent to peers. If this communication is encrypted, spam mails, sent by a bot for the distribution botnet could be a sign of a P2P-bot net.

Evaluation: Having shown the implementation of the combined approach in the selected scenario, presents how this implementation is evaluated. This is to ascertain the possibility of CHID to reduce resource consumption while maintaining detection accuracy when compared with the default Bro packet-based detection (PO). It is challenging to evaluate an IDS with realistic traffic with an approach that is repeatable and reproducible. Since series of measurements are run in the experiments, running the experiments on live traffic approach is not possible because that will not yield fair comparisons when different configurations run in sequence. In other words, live traffic approach has limitations, in terms of repeatability of experiments for any systematic performance evaluation study.

Executing the experiments in offline-mode approach to receive datasets (traces) also not practical. This is because Bro would process the packets as quickly as possible at 100% CPU utilization. To address the aforementioned issues, a method that combines the best of both mentioned approaches is used. In this method, Bro reads the input from a dataset but in live mode, which can be achieved by replaying the traces into Bro. This approach results in a reproducible procedure that is comparable to using the data in the experiment on live traffic. All of the tests were run three times to avoid any anomalies or noise in results.

However, compare the new results with other's studies were proven hard to accomplished[29]. However to obtain the original implementation of other methods is difficult due to copyright issues. In addition, most of the study in IDS field do not share their dataset due to privacy reasons. In addition, Tavallaee *et al.*[30] stated that most of detection proposals lack of proper documentation of their methods and experiments.

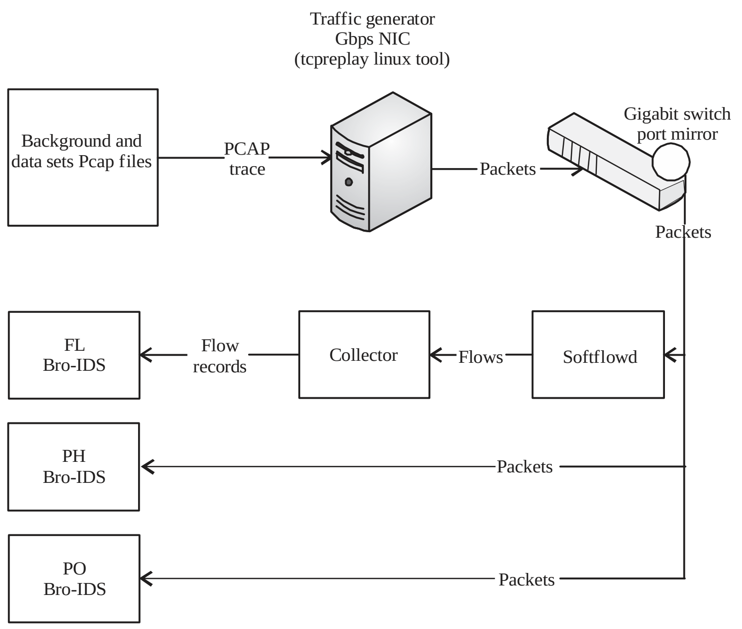

Fig. 3: Experimental testbed. The experimental environment run on two machines interconnected through the Gigabit switch. The first machine is used for traffic generation; the machine replays PCAP trace using tcpreplay to the second machine. The second machine is installed with softflowd and Bro

In this study, local comparative analysis is performed between CHID approach (FL+PH) and PO in terms of detection rate, memory and CPU usage and packet drop rate. In other words, total resource consumption for Bro in the combined approach is calculated to perform a direct comparison with Bro PO. However, PH and PO share the same signature detection. The difference between them is that PH is involved in input framework and BPF techniques, while PO is left with default configurations. The following subsections initially introduce the network testbed environment in which the evaluation is performed and then present the dataset used in this measurement.

Experimental testbed: The proposed implementation run on a testbed depicted in Fig. 3. The experimental environment run on two machines interconnected through the Gigabit switch. Both machines are running 12.04 Linux-based Ubuntu desktop 64 bit with intel i7 3.1 GHz with 32 GB of RAM. Both machines NICs support Gbps. The first machine is used for traffic generation; the machine replays real (previously captured) network traffic datasets (presented in the next section) using tcpreplay[31] v 4.0.5 and sends the traffic to the second machine for further analysis. The second machine is installed with softflowd[32] v 0.9.9 and Bro 2.3.

The experiments are repeated by replaying the datasets at the following rates: 100, 200, 500 and 1000 Mbps using tcpreplay. The switch supports Gigabit speed with port mirror enabled to forward all traffic to the analyzing (second) machine. Softflowd is used as the flow aggregator with default parameters to generate flow records from the dataset packets received and to export those records to the collector. Softflowd is also capable of generating Cisco netflow export format. In the evaluation, the resource consumption of netflow aggregation is not considered because it is assumed to exist in a production network[2]. Bro is used for malicious detection using the policy script. In addition, Bro is configured to collect the flow records by reading the flows for Bro analysis. All Bro instances experimented in this study in identical environment in term of input source, platform and traffic speed.

Dataset: The greatest challenge in validating a detection method is the lack of standard public datasets[29]. The DARPA 1999, which is believed to be the most standard public trace was criticized due to its age and inability to reflect real-world traffic by Shiravi *et al.*[33]. At this writing, only a small number of datasets is publicly available. Datasets presented in Table 2 are collected which have been made public for the research community and will be used in the experiments. The first dataset used is the information Security and Object Technology (ISOT) dataset[34], generated by the University of Victoria in 2011. The ISOT dataset has a combination of malicious P2P-bot and normal traffic (Table 3). The other datasets were generated by Czech Technical University (CTU) in different scenarios and were published in 2013[29]. Three individual scenarios are selected, CTU-51, CTU-52 and CTU-53, from the CTU datasets[29].

These datasets consist of real traffic in the PCAP format. However, these datasets are labeled, i.e., IP addresses of malicious and non-malicious hosts are known. Labeling this traffic is useful to validate the accuracy of the detection methods. However, these existing recent public datasets are limited to certain types of attacks. For IRC-bot, it was performed on CTU-51 and CTU-52 datasets, whereas P2P-bot activities are generated in both ISOT and CTU-53 datasets. For the non-malicious traffic, unfortunately none of the datasets mentioned in Table 4 (except for the ISOT dataset) contains full-payload background traffic for privacy reasons. Therefore, a one-day complete payload trace is used. The trace was captured at Alfaisal University, Prince Sultan College Jeddah (PSCJ), Information Technology Center, at the main gateway link that connects hundreds of hosts with an educational network to the Internet. The size of this trace is 8 GB and contains 32 million packets, corresponding to approximately 1.9 million flows. This trace is named as "PSCJ" and it contains a variety of network activities. The PSCJ dataset involves everyday activity usage such as HTTP web behavior, popular sharing file packets, IRC traffic, emails and streaming media. However, PSCJ trace is combined and injected along with each of the datasets listed in Table 2.

Table 2: Total number of packets and flows on data sets

Dataset	Total No. of packets	Total No. of exported flows
ISOT*	157 millions	5.2 millions
CTU-51**	66.3 millions	31.7 millions
CTU-52***	3.9 millions	1.7 millions
CTU-53****	351,537	11,117

#Information security and object technology dataset, generated for P2P-bot scenario, **Czech technical university CTU-51 dataset, generated for IRC-bot scenario, ***Czech technical university CTU-52 dataset, generated for IRC-bot scenario and ****Czech technical university CTU-53 dataset, generated for P2P-bot scenario

Table 3: Packet drop rate percentage in P2P-bot scenario

	100	200	500	1000
Traffic rates	----------------	(Mbps)	----------------	
PH#	0.00	0.16	5.40	21.12
PO**	5.04	9.32	29.79	48.45

#PH: Packet-based detection in the approach and **PO: Default packet-based only detection that inspects all packets

Table 4: Packet drop rate percentage in IRC-bot scenario

	100	200	500	1000
Traffic rates	----------------	(Mbps)	----------------	
PH#	0.00	0.00	7.98	27.47
PO**	2.50	5.12	26.02	48.55

#PH: Packet-based detection in the approach and **PO: Default packet-based only detection that inspects all packets

RESULTS

Evaluation results of the measurement performance are presented. In all measurements, similar to PO, PH in CHID was able to detect all IRC-bot and P2P-bot infected IP addresses that were reported and labeled in the traces. With this result in mind, PH yields a high accuracy rate with a zero false positive rate. Concerning resource consumption, in Fig. 4 and 5 plotted the average percentage of memory and

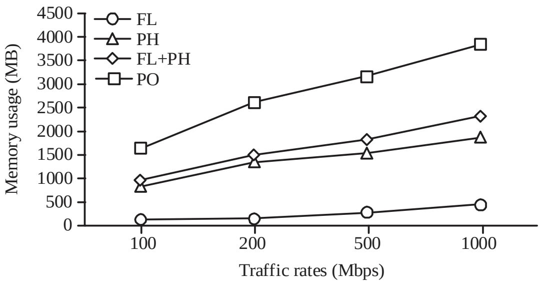

Fig. 4: Average memory usage with different traffic rates in P2P-bot scenario. To study how resource consumption effect the proposed approach, traffic were replayed at 100, 200, 500 and 1000 Mbps, FL: Flow-based detection, PH: Packet-based detection in theapproach, FL+PH: Total memory consumption of FL and PH and PO: Default packet-based only detection that inspects all packets and PO generates generally higher memory usage compared with the hybrid method (F+PH)

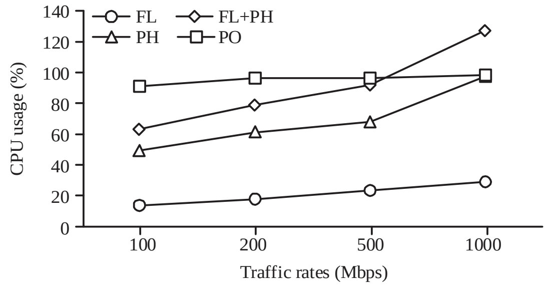

Fig. 5: Average CPU usage with different traffic rates in P2P-bot scenario. To study how resource consumption effect the proposed approach, traffic were replayed at 100, 200, 500 and 1000 Mbps, FL: Flow-based detection, PH: Packet-based detection in the approach, FL+PH: Total CPU consumption of FL and PH and PO: Default packet-based only detection that inspects all packets and PO generates generally higher CPU usage compared with the hybrid method (F+PH) until traffic rate reaches 500 Mbps

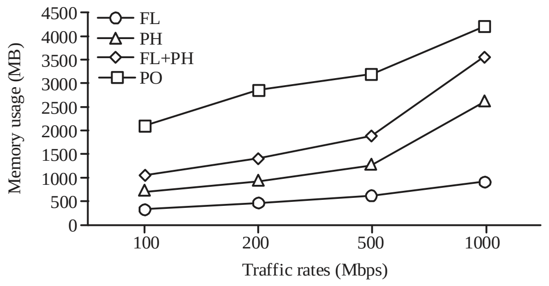

Fig. 6: Average memory usage with different traffic rates in IRC-bot scenario. To study how resource consumption effect the proposed approach, traffic were replayed at 100, 200, 500 and 1000 Mbps, FL: Flow-based detection, PH: Packet-based detection in approach, FL+PH: Total memory consumption of FL and PH and PO: Default packet-based only detection that inspects all packets and PO generates generally higher memory usage compared with the hybrid method (F+PH)

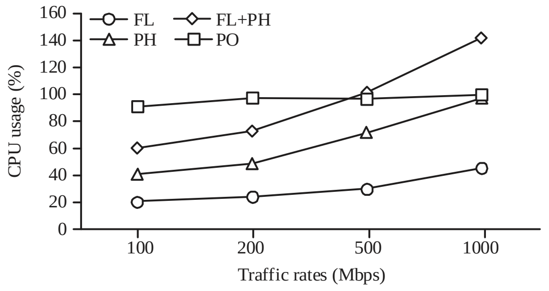

Fig. 7: Average CPU usage with different traffic rates in IRC-bot scenario. To study how resource consumption effect the proposed approach, traffic were replayed at 100, 200, 500 and 1000 Mbps, FL: Flow-based detection, PH: Packet-based detection in the approach, FL+PH: Total CPU consumption of FL and PH and PO: Default packet-based only detection that inspects all packets and PO generates generally higher CPU usage compared with the hybrid method (F+PH) until traffic rate reaches 500 Mbps

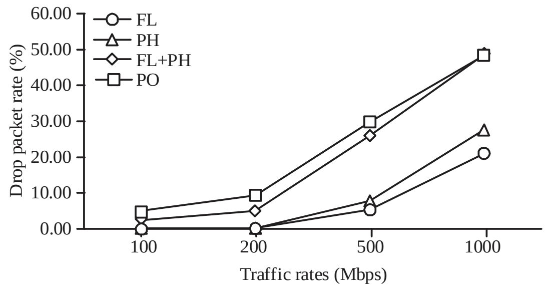

Fig. 8: Drop packet rate with different traffic rates. To study how packet drop rate effect the proposed approach, traffic were replayed at 100, 200, 500 and 1000 Mbps, PH-IRC and PH-P2P: Packet-based detection in the approach in IRC-bot and P2P-bot scenarios, respectively, PO-IRC and PO-P2P: Default packet-based only detection that inspects all packets in IRC-bot and P2P-bot scenarios, respectively and PO generates generally higher packet drop rate compared with PH

CPU usages while replaying the P2P-bot trace with speeds ranging from 100-1000 Mbps, respectively. When sending packets at a constant rate of 200 Mbps for example, CHID method (or FL+PH) dropped CPU usage average from 97.7% with PO to 72.4% with the CHID approach. Figure 6 and 7 showed average percentage of memory and CPU usages in IRC-bot scenario. At a 200 Mbps rate, CHID approach can save 50.6% of memory and 18.1% of CPU usage (Fig. 4 and 5). In both scenarios as shown in Fig. 5 and 7, PH had less CPU resource consumption compared with PO until 1000 Mbps rate. However, the total CPU consumption of CHID approach (FL+PH) reached at almost the full CPU capacity when the traffic rate was about 500 Mbps.

With concerning packets drop issues (Fig. 8, Table 3, 4) compared the packet drop with traffic rates from 100-1000 Mbps for P2P-bot and IRC-bot. Drop packet rate was calculated in relation to the total packet received. At 500 Mbps in IRC-bot scenario for example, PH reported 7.98% of drop packet while 26% in PO. Experiments demonstrated that when P2P-bot scenario is measured, Bro PH solution could handle bandwidth up to 200 Mbps with little drop of 0.16%. With the same traffic rate, no drop was reported when IRC-bot was tested.

DISCUSSION

Since CHID was able to detect all infected hosts reported in the datasets, PH is not only able to detect intrusion activities but also means that FL detection plays an important role for detection accuracy as supported by Sperotto *et al.*[19]. This is because PH filtered incoming packets depend primarily upon the suspicious list generated from FL detection. According to Alaidaros and Mahmuddin[24], implemented flow-based detection only and reported significant number of false positive alerts. However, when the same dataset input into CHID approach, no false positive alerts were identified. This is due to the extra layer (PH) added after flow-based detection in CHID.

Findings also show that PO generates higher memory and CPU usage compared with CHID approach. Unfortunately, this holds true until traffic rate reaches 500 Mbps. This means that the CHID approach can save resource consumption until 500 Mbps rate take place. Before traffic rate reach 500 Mbps, the mentioned observation is expected because PH in CHID processes only potential (filtered) packets rather than inspecting all incoming traffic. In addition, resource consumption of packet-based in CHID can be reduced compared with default packet-based that inspects all packets. This implies that the resource consumption of input framework mechanism, which is used for multi-Bro instances communication does not affect negatively on intrusion detection as stated by Amann *et al.*[22].

Number of filtered hosts in PH also affects the resource consumptions. When comparing between P2P-bot and IRC-bot cases, it was observed that the memory and CPU usage of P2P-bot scenario are higher compared with IRC-bot scenario. This holds until the traffic rate reaches about 500 Mbps. The reason for this is the increased number of filtered hosts reported from the P2P-bot scenario. With the same reason, when receiving packets at 500 Mbps onwards, IRC-bot scenario consumes higher resources compared with P2P-bot case. The more hosts that are in the filter, the more traffic is received and consumed and vice versa. However, these filtered hosts require more filter compilation overhead. This compilation is needed to apply filters in the packet capturing operation. In other words, the filter compilation or BPF update resource usages increases when the number of hosts added to filter increases, which might destroy the advantage of the combination method.

In addition, input framework might contribute a negative effect to resource consumption if the filtered host number is high. More filtered hosts require more updates to the corresponding table. Moreover, the re-read function must be called every time a new host is added to the log file generated by FL. Unfortunately, an optimum value of filtered hosts cannot be easily identified. This is because such identification largely depends upon the data in the network traffic characteristics in the traces themselves, i.e., session length, protocols and the ratio of malicious traffic to benign traffic.

Concerning packet drop rate, it is expected that PO has higher drop rate compared with PH even at 1 Gbps rate. This is because of light CPU usage in PH as stated early in this section. Based on findings, PO detection can handle up to 80 Mbps rate. This observation was also stated by Pihelgas[35] and Bro official documentation[36]. Comparing with these studies, on the other hand, Bro in CHID approach can handle higher rate up to 200 Mbps.

It is also observed that PH in P2P-bot starts to drop packets faster than when IRC-bot scenario is tested. The reasons behind this may refer to (1) The higher CPU consumption at the early traffic rates when P2P-bot scenario is measured as mentioned earlier in this section and (2) Per-packet processing time which might occur when Bro spends much time on a single packet.

However, both Bro PO and PH have shown to dropped packets increase when the speed rate of the packets increase. This is because Bro is single-threaded technique which means Bro only fully utilize one processor core and is not taking advantage of multi-core CPU. This will lead to overloading the Bro with a big amount of traffic. Bro provides an option to spread workload across many cores using cluster-mode[36]. Another solution should be used to decrease the packet drop rate is PF_RING[31] instead of standard pcap library which was being used in this study. The PF_RING is packet capture mechanism that improves the packet capture speed. Pihelgas[35] studied the packet drop of Bro when PF_RING was implemented, the study concluded better packet drop rate. However, in order to enable PF_RING in Bro, cluster-mode must be run.

However, CHID is not designed as a suitable solution for enhancing the performance of IDSs in all scenarios. The CHID just demonstrates the possible approach of combining flow-based and packet-based detection in specific scenarios. Similar to IRC-bot and P2P-bot attacks, there are other attacks that can be detected in both flow and packet level. Such attacks include SPAM, HTTP-bot and brute-force attacks. These attacks with combined approach can be considered in the future.

CONCLUSION

In this study, a model named CHID for improving IDS scalability was presented. The CHID is based on combining flow-based and packet-based detections to build on their advantages and overcome their drawbacks to reduce the resource consumption of IDS. The CHID approach used Bro IDS and utilize the input framework to communicate between flow-based and packet-based detection modules. The CHID approach was evaluated by replaying labeled datasets with traffic rates from 100 Mbps through 1 Gbps.

The experimental evaluation shows that CHID approach could gain a significant performance improvement compared with a default Bro packet-based detection implementation. This improvement hold true until traffic rate reaches 500 Mbps. When an IRC-bot scenario is implemented at 200 Mbps, CHID approach can save 50.6% of memory and 18%

of CPU usage. With this saving in mind, the resource usage of packet-based detection in CHID approach were enhanced. With scalability improving, CHID implementation maintains detection accuracy level, manages to eliminate false positive alerts generated from flow-based detection and detects all reported malicious hosts.

Input framework and BPF compilation resources depend upon the number of suspicious hosts generated from flow-based detection. The more hosts identified by flow-based detection, the more resources are needed in packet-based detection in CHID approach. A huge number of filtered hosts would result in compilation process overhead that might destroy the advantage of CHID. For flow-based detection resources, the number of concurrent hosts processed in the FL table can affect the overall performance of proposed approach because this number requires table updating and dynamic threshold calculations. The CHID can be considered a skeleton model facilitating the application of other intrusion or monitoring detection systems in the future.

ACKNOWLEDGMENT

This study has been partially supported by Al-Faisal University, Prince Sultan College Jeddah (PSCJ), Research Centre.

SIGNIFICANCE STATEMENTS

- Along with the wonderful benefits that the Internet gives, it also has its dark face. Since the Internet becomes bigger and bigger, network security attack threats have become more serious
- Considering the damage cost caused by the attacks, it is important to detect attacks as soon as possible. For this purpose, Network Intrusion Detection Systems (NIDSs) have been developed
- With the increase in network speed and number and types of attacks, existing NIDSs, face challenges of capturing every packet to compare them to malicious signatures. These challenges will impact on the efficiency of NIDSs, mainly the performance and accuracy power
- The expected contribution of this research is a new model that enhances the NIDS scalability without compromising detection accuracy

REFERENCES

1. Liao, H.J., C.H.R. Lin, Y.C. Lin and K.Y. Tung, 2013. Intrusion detection system: A comprehensive review. J. Network Comput. Applic., 36: 16-24.
2. Golling, M., R. Hofstede and R. Koch, 2014. Towards multi-layered intrusion detection in high-speed networks. Proceedings of the 6th International Conference on Cyber Conflict, June 3-6, 2014, Tallinn, pp: 191-206.
3. Guo, G.F., 2014. The study of the ontology and context verification based intrusion detection model. Applied Mech. Mater., 644-650: 3338-3341.
4. Vykopal, J., M. Drasar and P. Winter, 2013. Flow-based Brute-Force Attack Detection. In: Advances in IT Early Warning, Schoo, P. and E. Hermann (Eds.). Fraunhofer IRB Verlag, Germany, Switzerland, ISBN: 9783839604748, pp: 41-51.
5. Sourour, M., B. Adel and A. Tarek, 2009. Environmental awareness intrusion detection and prevention system toward reducing false positives and false negatives. Proceedings of the IEEE Symposium on Computational Intelligence in Cyber Security, 30 March-April 2, 2009, Nashville, TN., pp: 107-114.
6. Shimamura, M. and K. Kono, 2006. Using attack information to reduce false positives in network IDS. Proceedings of the 11th IEEE Symposium on Computers and Communications. June 26-29, Cagliari, Sardinia, Italy, pp: 386-393.
7. Xiao, M. and D. Xiao, 2007. Alert verification based on attack classification in collaborative intrusion detection. Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, Volume 2, 30 July-August 1, 2007, Qingdao, pp: 739-744.
8. Kumar, G.S., C.V.K. Sirisha, R.K. Durga and A. Devi, 2012. Robust preprocessing and random forests technique for network probe anomaly detection. Int. J. Soft Comput. Eng., 1: 391-395.
9. Bhatti, D. and P.V. Virparia, 2012. Data preprocessing for reducing false positive rate in intrusion detection. Int. J. Comput. Applic., 57: 15-15.
10. Bhatti, D.G., P.V. Virparia and B. Patel, 2012. Conceptual framework for soft computing based intrusion detection to reduce false positive rate. Int. J. Comput. Applic., 44: 1-3.
11. Spathoulas, G.P. and S.K. Katsikas, 2009. Using a fuzzy inference system to reduce false positives in intrusion detection. Proceedings of the 16th International Conference on Systems, Signals and Image Processing, June 18-20, 2009, Chalkida, pp: 1-4.
12. Aydin, M.A., A.H. Zaim and K.G. Ceylan, 2009. A hybrid intrusion detection system design for computer network security. Comput. Electr. Eng., 35: 517-526.
13. Hussein, S.M., F.H.M. Ali and Z. Kasiran, 2012. Evaluation effectiveness of hybrid IDS using Snort with Naive Bayes to detect attacks. Proceedings of the 2nd International Conference on Digital Information and Communication Technology and it's Applications, May 16-18, 2012, Bangkok, pp: 256-260.

14. Day, D.J., D.A. Flores and H.S. Lallie, 2012. Condor: A hybrid IDS to offer improved intrusion detection. Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, June 25-27, 2012, Liverpool, pp: 931-936.
15. Limmer, T. and F. Dressler, 2009. Flow-based front payload aggregation. Proceedings of the 4th IEEE LCN Workshop on Network Measurements, October 20-23, 2009, Zurich, Switzerland, pp: 1102-1109.
16. Schaffrath, G. and B. Stiller, 2008. Conceptual integration of flow-based and packet-based network intrusion detection. Proceedings of the 2nd International Conference on Autonomous Infrastructure, Management and Security, July 1-3, 2008, Bremen, Germany, pp: 190-194.
17. Mehra, P., 2012. A brief study and comparison of snort and bro open source network intrusion detection systems. Int. J. Adv. Res. Comput. Commun. Eng., 1: 383-386.
18. Estan, C., K. Keys, D. Moore and G. Varghese, 2004. Building a better NetFlow. Proceedings of the ACM SIGCOMM, August 4, 2004, San Diego California, USA., pp: 245-256.
19. Sperotto, A., G. Schaffrath, R. Sadre, C. Morariu, A. Pras and B. Stiller, 2010. An overview of IP flow-based intrusion detection. IEEE Commun. Surv. Tutorials, 12: 343-356.
20. Alaidaros, H., M. Mahmuddin and A. Al Mazari, 2011. An overview of flow-based and packet-based intrusion detection performance in high speed networks. Proceedings of the International Arab Conference on Information Technology, December 11-14, 2011, Saudi Arabia.
21. Zhu, Z., G. Lu, Y. Chen, Z.J. Fu, P. Roberts and K. Han, 2008. Botnet research survey. Proceedings of the 32nd Annual IEEE International on Computer Software and Applications, 28 July-August 1, 2008, Turku, pp: 967-972.
22. Amann, B., R. Sommer, A. Sharma and S. Hall, 2012. A lone wolf no more: Supporting network intrusion detection with real-time intelligence. Proceedings of the 15th International Symposium, September 12-14, 2012, Amsterdam, The Netherlands, pp: 314-333.
23. De Ocampo, F.B., T.M. del Castillo and M.A. Gomez, 2013. Automated signature creator for a signature based intrusion detection system (pancakes). Proceedings of the 2nd International Conference on Cyber Security, Cyber Peace Fare and Digital Forensic, March 4-6, 2013, Kuala Lumpur, Malaysia, pp: 198-205.
24. Alaidaros, H. and M. Mahmuddin, 2015. Flow-based approach on bro intrusion detection. Proceedings of the 2nd Advancement on Information Technology International Conference, December 3-5, 2015, Krabi.
25. Stover, S., D. Dittrich, J. Hernandez and S. Dietrich, 2007. Analysis of the storm and Nugache Trojans: P2P is here. USENIX: Login, 32: 18-27.
26. Wurzinger, P., L. Bilge, T. Holz, J. Goebel, C. Kruegel and E. Kirda, 2009. Automatically generating models for botnet detection. Proceedings of the 14th European Symposium on Research in Computer Security, September 21-23, 2009, Saint-Malo, France, pp: 232-249.
27. Stover, S., D. Dittrich, J. Hernandez and S. Dietrich, 2007. Analysis of the storm and Nugache Trojans: P2P is here. USENIX: Login, 32: 18-27.
28. Sommer, R. and V. Paxson, 2003. Enhancing byte-level network intrusion detection signatures with context. Proceedings of the 10th ACM Conference on Computer Communication Security, October 27-30, 2003, Washington, DC., USA., pp: 267-271.
29. Garcia, S., M. Grill, J. Stiborek and A. Zunino, 2014. An empirical comparison of botnet detection methods. Comput. Security, 45: 100-123.
30. Tavallaee, M., N. Stakhanova and A.A. Ghorbani, 2010. Toward credible evaluation of anomaly-based intrusion-detection methods. IEEE Trans. Syst. Man. Cybernetics, Part C (Applic. Rev.), 40: 516-524.
31. Undy, M., 1999. Tcpreplay. Software Package, May 1999.
32. Miller, D., 2012. Softflowd: A software netflow probe. http://www.mindrot. org/projects/softflowd/.
33. Shiravi, A., H. Shiravi, M. Tavallaee and A.A. Ghorbani, 2012. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Comput. Secur., 31: 357-374.
34. Zhao, D., I. Traore, A. Ghorbani, B. Sayed, S. Saad and W. Lu, 2012. Peer to peer botnet detection based on flow intervals. Proceedings of the International Information Security and Privacy Conference, June 4-6, 2012, Crete, Greece, pp: 87-102.
35. Pihelgas, M., 2012. A comparative analysis of open-source intrusion detection systems. Master's Thesis, Tallinn University of Technology, Tallinn.
36. Weaver, N. and R. Sommer, 2007. Stress testing cluster bro. Proceedings of the DETER Community Workshop on Cyber Security Experimentation and Test, August 6-7, 2007, Boston, MA., USA., pp: 9-9.