An Aggregative Approach for Scalable Detection of DoS Attacks

Alireza Hamidi, Sudhakar Ganti, Kui Wu
Department of Computer Science, University of Victoria, Victoria, BC, Canada
e-mail:{alireza, sganti, wkui}@cs.uvic.ca

***Abstract*— In Voice Over IP (VoIP) systems, intruders can launch DoS attacks by establishing a large number of open connections to prevent the system from serving legitimate users. Existing defenses against DoS attacks on VoIP systems maintain full state information and thus are not scalable to implement at core routers. To this end, we adopt a two-layer aggregation scheme, termed Advanced Partial Completion Filters (APCF), to defend against DoS attacks without tracking state information of each individual connection. APCF provides adjustable control parameters so that both false alarms and detection rate can be controlled.**

I. Introduction

Voice-over-IP (VoIP) is an application that provides cheap voice communications over the Internet to millions of customers. SIP is a widely used signaling protocol in VoIP communications and has naturally become the target of many Denial-of-Service (DoS) attacks. As a result, the malicious signaling traffic will finally exhaust the resources of a VoIP server which maintains each connection state [1]. Detection of DoS attacks on VoIP networks is thus of great importance.

When a detection system is deployed at the edge of a network, it should be able to operate at high link speeds (e.g. 1 Gb/s or higher). Detection systems that maintain per-flow state information and handle huge number of flows can create implementation problems at these speeds. Scalability of detection system has become an important and hard problem to solve. Currently, the detection of several types of attacks such as evasion and TCP hijacking has been proved impossible to perform in a scalable fashion [3], and some attacks such as bandwidth attacks already have scalable solutions [4]. Distributed DoS (DDoS) and scanning attacks, however, are still unattended [2]. Existing solutions like *Partial Completion Filters* (PCF) [2] introduce a scalable approach by aggregating incoming flows and by not tracking each individual connection. PCF, however, is still not efficient with respect to false detection and sensitivity. The question is: *Is it possible to design a scalable detection system that works at high link rates and at the same time provides high detection ratio and low error rate?*

The contribution of this paper is to provide a positive answer to the above problem. Specifically,

- The structure of PCF is changed to make it more accurate and easy to control.
- Our new detection model is more sensitive to attacks and more robust against behavioral aliasing (see Section II).
- In noisy environment, our new model behaves more reliably than PCF since it can preserve same efficiency while its sensitivity can be adjusted.

II. Background Knowledge

A. Partial Completion Filter (PCF)

Authors in [2] have introduced a scalable *aggregation* method in detecting partial completion attacks, termed Partial Completion Filter (PCF), that avoids maintaining per-flow state and at the same time keeps false detection rate under control. We must deal with the *behavioral aliasing* problem if flows are aggregated, i.e., aggregation of bad behaviors may appear as innocent or vice versa.

PCF aggregates connections based on some fields of each packet that can be easily extracted without any reassembly. PCF is mainly comprised of a hashing function (or several hashing functions in multi stage PCF [2], [5]) which hashes incoming connections into some *buckets*, and assigns a counter to each of them to hold the balance between connection establishments and terminations. Fig. 1 presents a hashing scheme in PCF. In this example, all TCP packets originated from the same source IP address are aggregated into a bucket. Each bucket has been assigned a counter associated with a special feature. For example, the counter can be configured to increase for every SYN packet and decrease for every FIN packet passing through a monitoring point for TCP connections.

Intuitively, under normal system traffic, SYN and FIN packets should be roughly balanced, which has been proved right via mathematical analysis and experiments in [2]. The major concern of PCF is the errors caused by behavioral aliasing. It has been shown that the errors resulted from behavioral aliasing are bounded by a Normal distribution [2]. We have found that the errors in [2] were under-estimated.

B. The Binomial Behavior of PCF

Let us assume that X_i is the value assigned (e.g., $+1$ for initiation and -1 for termination) for i^{th} incoming message, and the value of the PCF counter is maintained as $X = \sum_{i=0}^{n} X_i$. Then the value of PCF counter follows a Binomial distribution [2]. Since we expect the number of connection initiation messages to be equal to that of the terminations in a legitimate traffic scenario, the Binomial values of 1 and -1 are assigned with an equal probability of 0.5. Hence the mean of the Binomial distribution, $\mu_b = 0$, and the standard

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

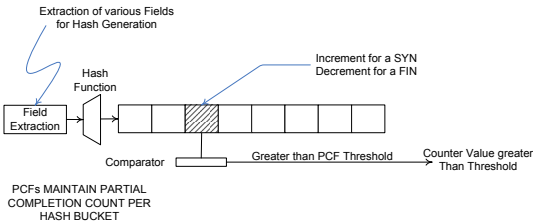


Fig. 1: Partial Completion Filter [2]

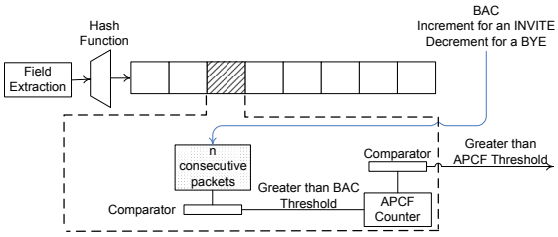


Fig. 2: APCF Functional Diagram

deviation of the Binomial distribution, $\sigma_b = 1$. According to the Central Limit theorem, for a large enough n, this Binomial distribution tends to a Normal distribution with $\mu_n = n\mu_b$ and $\sigma_n = \sigma_b\sqrt{n}$. Here, n is the number of valid messages[1] hashed to each bucket. Therefore, with confidence bounds of a and b, the following equation can be derived [2]:

$$P\left[a \leq \frac{X - \mu_n}{\sigma_n} \leq b\right] = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-\frac{z^2}{2}} dz \qquad (1)$$

With $a = -3$, $b = 3$, $\mu_b = 0$, and $\sigma_b = 1$ the above equation implies that $Pr\left[|X| \leq 3\sqrt{n}\right] = 0.9987$. If there are $3\,000$ packets hashed to each bucket, then the equality indicates that the probability of a counter value lies in between 164 and -164 is 0.9987, given that all of the connections are benign.

C. Error Estimation

We call a bucket as a "bad bucket" if its counter value falls outside a given bound. Because PCF raises alarms for all flows hashed to a bad bucket, two types of errors occur in PCF alarms: the first type occurs if all connections are *benign* in a bad bucket; the second occurs when a bad bucket *includes* both legitimate and attack connections. The probability that the first type of errors occur is smaller than $1 - 0.9987 = 0.0013$ if we set the bound to the counter value as $3\sigma_n$, as shown above. For the second type of errors [2], if we assume that there are b attacks and the total number of buckets is m, then the number of bad buckets should be close to b if $m >> b$. Therefore, the number of legitimate connections mapped to a bad bucket by chance alone will be $\left(\frac{b}{m}\right) \cdot f$, where f is the total number of benign connections. The number of this type of errors is not negligible since f could be very large. The authors in [2] did not provide any experimental assessment.

III. Advanced PCF

A. APCF Introduction

Our new detection method, the *Advanced PCF* (APCF) improves the detection accuracy by breaking the PCF counter into two different counters. The first one, *Behavior Alarming Counter* (BAC) counts a certain number of packets that arrive consecutively from the connections and are hashed to a certain bucket. It alarms a second counter called *APCF counter*, if past stream of signals is found suspicious. APCF counter holds the number of alarms it has received from BAC. If the APCF counter value exceeds a preset threshold (τ_{APCF}), it recognizes the flows hashed to the bucket as intrusions. Fig. 2 shows the general functional diagram of APCF. In this figure, connections are hashed into buckets for each of which there is an APCF counter. Dashed area shows the functionality of BAC along with APCF counter. Note that each bucket has its own APCF counter and BAC.

B. Behavior Alarming Counter

BAC is a bin of n consecutive messages (n is called BAC *population*) and holds the balance between the connection initiation and connection termination signals. If the balance is more than a preset threshold τ_{BAC} (named BAC *alarm threshold* and stated as a percentage of n), BAC identifies that stream of packets as a suspicious one. BAC is a counter similar to PCF in the sense that it is incremented/decremented for every connection initiation/termination packets. However, the difference between them is that BAC stops counting after every n packets, alarms APCF if BAC is more than the threshold, and resets itself to zero. This alarm causes APCF to increment its counter. A behavioral alarming counter with population of n and alarm threshold of τ_{BAC} is denoted as BAC(n, τ_{BAC}). BAC(n, τ_{BAC}) signals an alert to the APCF counter if the count exceeds $n.\tau_{BAC}$.

Fig. 3 gives an insight into BAC behavior for a single bucket, assuming connection initiation packets are marked as "$+$" and connection terminations as "$-$". Note that BAC resets its value to zero for the next window of n packets. If parameters are adjusted appropriately, the imbalance in the buckets will be certainly recognized in one or more packet groups.

IV. APCF Analysis

A. Theoretical Analysis

First we discuss the probability for BAC to alarm an APCF in the absence of any attack traffic. In a benign traffic scenario, BAC value has a binomial distribution as BAC is updated with values of -1 and $+1$ with equal probability. Therefore, BAC alarm probability with certain population size and alarm threshold (n and τ_{BAC}), **given that there is no intrusion**,

[1]Initiation and termination messages are referred to as valid messages. In TCP they are SYN and FIN packets, in SIP they are INVITE and BYE/CANCEL messages.

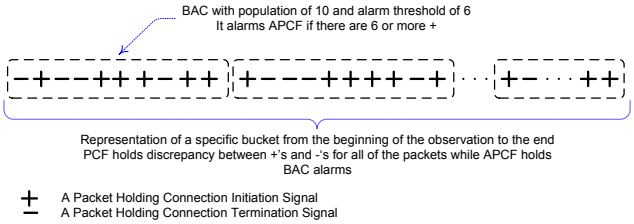


Fig. 3: BAC Behavior

can be calculated as,

$$P_{BAC} = P\left[BAC\left(n, \tau_{BAC}\right) \geq \alpha\right] = \sum_{k=\alpha}^{n} \binom{n}{k} \cdot \left(\frac{1}{2}\right)^{n} \quad (2)$$

where $\alpha = \lfloor n.\tau_{BAC} \rfloor$. APCF counter value is always updated with values of 0 and 1 and therefore, the APCF counter value follows a Binomial distribution. That is, APCF counter counts BAC alarms as 1 with success probability of P_{BAC} and no alarm as 0 with failure probability of $(1 - P_{BAC})$. This fact helps to bound the APCF counter for a non-attacked bucket, similar to PCF. As an example, according to (2), $P[BAC(20, 0.75) \geq 15] = \binom{20}{15}0.5^{20} + \binom{20}{16}0.5^{20} + \cdots + \binom{20}{20}0.5^{20} = 0.02069473$. This means that BAC(20, 0.75) alerts APCF with relatively small probability of 0.0269473, given that it investigates every 20 packets for 15 or more invitations. This value is the success probability for the Binomial distribution of the affiliated APCF counter value.

As the BAC threshold nears 1.0, the BAC alarm probability reduces and APCF becomes less sensitive. On the other hand, when the BAC threshold is smaller, the BAC alarm probability increases and sensitivity grows. Note that APCF counting units are not for individual connections, but for BAC alarms. In other words, if the total number of packets hashed to a bucket is C and BAC population is n, then the maximum possible value of APCF counter is $m = \frac{C}{n}$.

Similar to PCF, the Binomial distribution of APCF counter can be approximated by a Normal distribution with mean $\mu_{APCF} = P_{BAC} \cdot m$ and a standard deviation $\sigma_{APCF} = P_{BAC}(1-P_{BAC})\sqrt{m}$. Substituting these values into (1) yields a tighter alarming bound for APCF in a fully benign traffic scenario. For example consider a bucket with 3 000 packets and a BAC population of $n = 20$. Thus, m would be 150, $P_{BAC} = P[BAC(20, 0.75) \geq 15] = 0.02069473$, $\mu_{APCF} = 3.10421$ and $\sigma_{APCF} = 0.24821$. Similar to (1), it is easy to calculate the probability that APCF won't raise an alarm for a bucket will be $P\left[|X - \mu_{APCF}| \leq (3\sigma_{APCF})\right] = 0.9987$.

This result shows that if all the flows are not attacks, the APCF counter deviation will be less than $3\sigma_{APCF}$, or approximately 0.74463, with a probability of 0.9987. Hence, APCF has significantly smaller counter deviation than PCF.

B. Parameter analysis

APCF introduces three controlling parameters as opposed to single counter as in PCF. These parameters are APCF threshold, BAC population and BAC alarm threshold. Among them, the alarming probability of BAC (P_{BAC}) is derived from the population and alarm threshold. It means that if we have a proper value of P_{BAC}, we can find a relation between population and alarm threshold. Moreover, BAC alarm probability is the key variable to calculate a bound for APCF threshold as well as its mean and standard deviation in the non-attack situation.

Adjusting APCF parameters for a given network relies upon three statistical attributes:

1) Average number of connection initiation packets or *mean number of initiation packets* or M_{in} in each bucket
2) Probability that APCF raises an alarm for a bucket or P_{ATTACK}
3) Average number of packets in each bucket or C

If the traffic contains intrusions, then (1) can be rewritten as:

$$P\left[a \leq \frac{X - P_{BAC} \cdot m}{P_{BAC}(1 - P_{BAC})\sqrt{m}} \leq b\right] = 1 - P_{ATTACK} \quad (3)$$

Consequently, if we relate P_{BAC} to some available network attributes, we can derive the relation between necessary parameters and adjust APCF accordingly. If we assume that a group of initiation messages are together in an intrusion traffic, such attacks would be very easy to detect. Instead, we assume more intelligent attacks that create signaling traffic with initiation and termination packets scattered randomly to evade detection. We point out that APCF will be more accurate if accurate models of signaling traffic are used. Clearly, this traffic modeling is beyond the scope of this work. Because BAC value is updated for every n packets, it can be approximated to a Poisson distribution [6] with $\mu_p = \frac{M_{in}}{m}$, where M_{in} is the average number of initiation packets received. Here we refer the BAC alarm probability as P'_{BAC} for this Poisson distributed BAC, to distinguish it from the Binomial distributed one. Subsequently, (4) (Poisson Cumulative Distribution Function) gives the value of P'_{BAC} based on μ_p, and two BAC parameters (population and threshold).

$$\begin{aligned} P'_{BAC} &= \frac{\Gamma(n+1, \mu_p)}{n!} - \frac{\Gamma(\lfloor n \cdot \tau_{BAC} \rfloor + 1, \mu_p)}{\lfloor n \cdot \tau_{BAC} \rfloor !} \\ &= e^{-\frac{M_{in}}{m}} \sum_{k=\lfloor n \cdot \tau_{BAC} \rfloor}^{n} \frac{\left(\frac{M_{in}}{m}\right)^k}{k!} \end{aligned} \quad (4)$$

Assuming a and b in (3) are symmetric, we can directly reformulate (3) as (5):

$$P\left[b \leq \frac{X - P'_{BAC} \cdot m}{P'_{BAC}(1 - P'_{BAC})\sqrt{m}}\right] = \frac{P_{ATTACK}}{2} \quad (5)$$

We can find b by looking at the statistical tables for Z-ratios and $P[\tau_{APCF} \leq X] = P_{ATTACK}/2$. Remember that X is the final value of the APCF counter, and P_{ATTACK} is a given control value. By substituting τ_{APCF} in (5), we get:

$$b = \frac{\tau_{APCF} - P'_{BAC} \cdot m}{P'_{BAC}(1 - P'_{BAC})\sqrt{m}} \quad (6)$$

We know that P'_{BAC} can be derived from n and τ_{BAC}, and $m = \frac{C}{n}$. As b is known from the table, (6) leaves us with three unknown variables. However, if there is no intrusion, τ_{APCF} can be bounded by (3) with probability of 0.98 and $b = -a = 3$. The result is (7) which leads to a relation between BAC population, BAC threshold and APCF threshold in a benign traffic scenario.

$$3 = \frac{\tau_{APCF} - P_{BAC} \cdot m}{P'_{BAC}(1 - P_{BAC})\sqrt{m}} \tag{7}$$

By substituting P_{BAC} and P'_{BAC} from (2) and(4) into (6) and (7), we come up with two equations that relate three APCF parameters with the affiliated network. The above analysis serves as a guideline heuristic to select APCF parameters for a given network.

V. APCF Experiments

A. Measurement parameters

We used general classification terms [7] for categorizing attacks in this paper. By the term *positive*, we refer to those single connections (<SIP, SP, DIP, DP>) detected as intrusions. *Negative* means a connection not detected as an attack. *False positives* are those innocent single connections that have been detected as intrusions and *true positives* are connections correctly detected as intrusions.

The tool we used to measure the APCF efficiency is Relative Operating Characteristics Diagram or ROC diagram [7]. ROC curve shows how a particular detector behaves in terms of the ratio of true detections over false detections, when a specific feature (e.g. APCF threshold) varies. The larger the area beneath the curve, the better balance the detector holds between false positives and false negatives [5]. Here are some definitions for the measures:

- *TP Rate(recall)*, also called *hit ratio* or *sensitivity*, is defined as the number of detected attacks over the number of all attacks.
- *FP Rate* is defined as the number of connections falsely detected as attacks over the number of all benign connections.
- *Precision* is defined as the number of detected attacks over the total number of detection alarms.
- *Efficiency* (F_α) is a weighted measure which gives a different priority to *precision* and *recall*, using a *weight factor* α:

$$F_\alpha = \frac{(1+\alpha) \cdot (\textit{precision} \times \textit{recall})}{\alpha \cdot \textit{precision} + \textit{recall}} \tag{8}$$

In our simulations, we used $\alpha = 1$ which is the *harmonic mean* of precision and recall.

B. Simulations

We used OMNET++ to simulate a network with more than 45 000 peers in 15 WANs and connected through 7 routers. Each WAN is considered as a domain network with a unique address for each proxy server, as in VoIP networks. Henceforth we call each WAN as an ISP, since they point to a VoIP proxy server. Each user selects one client, in the same domain or in a different one and randomly establishes a connection for a call period which is also selected randomly from an exponential distribution. Each ISP has been assigned an attack probability with which they lunch partial completion attacks on specified ISPs. Thereby we control the P_{ATTACK}. The whole network is exposed to a drop/retransmission mechanism which is set randomly with a controllable rate. Any user who is set to be an attacker is known and so we can keep track of false and true detections.

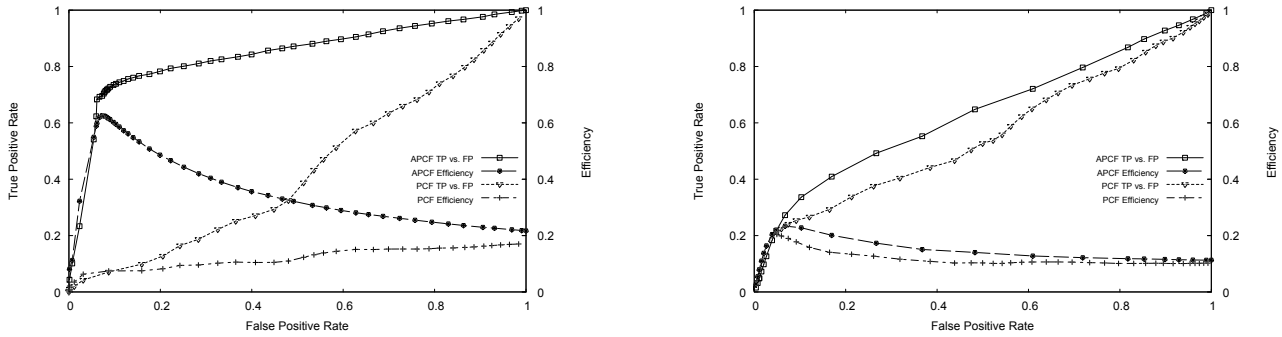
Two agents of PCF and APCF are placed at the single entry point of a pre-determined ISP that they guard. Detection ratios are based on single clients, not ISPs, which is an important difference with the result provided in [2] for PCF. Two scenarios are simulated: Scenario 1 (high attack) and Scenario 2 (low attack) with parameters listed in Table I.

C. Simulation results

Fig. 4 demonstrates how APCF and PCF detection ratios increase as their thresholds decrease. Figs. 4a and 4b show the massive attack scenario and low attack scenario, respectively. When APCF or PCF thresholds are large enough (near the origin) both false positive and true positive ratios are almost zero; this is for larger thresholds that prevent incoming connections to be declared as intrusion. As thresholds decrease, both ratios grow until they tend to one, since more and more incoming packets are declared as intrusion. The area below the ROC diagram shows efficiency of each detector.

ROC graphs demonstrate that in both cases, APCF true detection ratio outperforms that for PCF with the same false positive ratio, with notable significance in the high attack situation. Based on the analysis provided in Section IV-B, for the massive attack scenario with $P_{ATTACK} = 0.5$, $M_{in} = 22\,000$, $C = 36\,000$ and $n = 25$, we expect to have the best result from APCF with $\tau_{BAC} = 0.75$ and τ_{APCF} between 32 and 37. For the second scenario, n is the same, $P_{ATTACK} = 0.2$, $M_{in} = 11\,000$ and $C = 20\,000$ on average and the best results should be attained with 0.7 for τ_{BAC} and 39 to 47 for τ_{APCF}. Accordingly, APCF threshold is changed from 42 down to 0 for the first scenario and from 90 to 0 for the second. The bending point in Fig. 4 for both a and b falls in the expected area for τ_{APCF} ($\tau_{APCF} = 35$ for the first and $\tau_{APCF} = 51$ for the second case), the same point where efficiency starts to decline. For PCF the threshold is changed from 350 down to 10 for both scenarios which covers the best threshold stated in [2], i.e. $\tau_{PCF} = 150$.

Fig. 5 provides a different efficiency comparison with τ_{BAC} varying from 0.6 down to 0.9 and with best τ_{APCF} selected from (1) for both scenarios. Efficiency is a good comparison measure since it is calculated based on both precision and recall; hence, it encompasses all false positive, false negative and true positive rates implicitly. From Fig. 5a, obviously all APCF parameters yield better efficiency than the best PCF results. The second scenario contains intense blend of good and bad connections with low attack density. In this scenario,



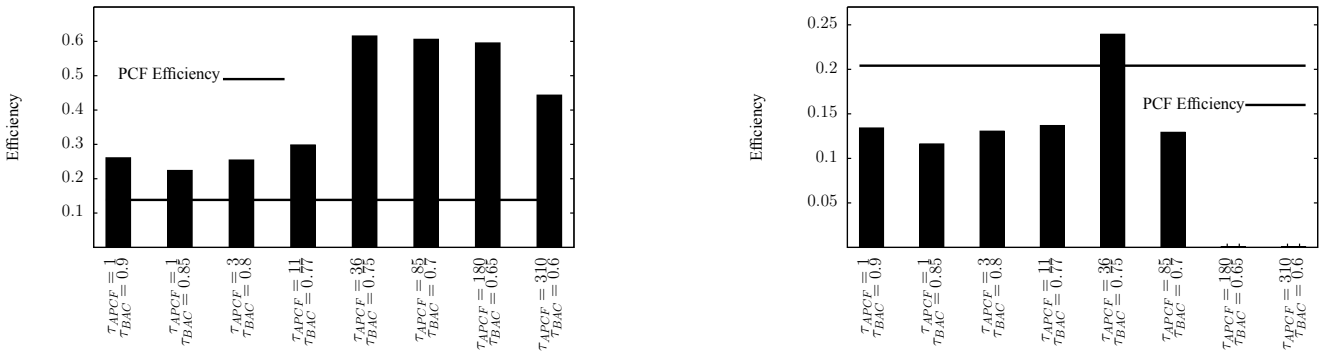
(a) High Attack Scenario (Scenario 1)

(b) Low Attack Scenario (Scenario 2)

Fig. 4: ROC diagram PCF and APCF. (a) High attack scenario, τ_{BAC} = 0.75, τ_{APCF} varying from 42 down to 0, τ_{PCF} varying from 350 down to 10 (b) Low attack scenario, τ_{BAC} = 0.7, τ_{APCF} from 90 down to 0, τ_{PCF} from 350 down to 10

TABLE I: Two simulated scenarios

	Average Attack Probability	M_{in}	BAC Threshold	BAC Population	Pkt drop Rate	APCF Hash	PCF Hash
Scenario 1	%50	22 000	0.75	25	0.1	<SRC ISP, DEST ISP>	<SRC ISP>,<DEST ISP>
Scenario 2	%20	11 000	0.7	25	0.01	<SRC ISP, DEST ISP>	<SRC ISP>,<DEST ISP>



(a) High Attack Scenario (Scenario 1)

(b) Low Attack Scenario (Scenario 2)

Fig. 5: Efficiency comparison of PCF and APCF for (a) High Attack Scenario (b) Low Attack Scenario

the best APCF efficiency may be poorer than the best PCF efficiency, but the gap between them is very small.

VI. Conclusion

DoS attacks are hazardous to current Internet services. Many existing solutions need to track and maintain per-flow state information, and thus are not scalable at high-speed networks. For scalable detection of DoS attacks, a recent approach [2] is to aggregate flows by using one or more hash functions to avoid maintaining per-flow information. We point out that flow aggregation may result in behavior aliasing and impose a large number of detection errors over the network because of the lack of balance between sensitivity and precision. We design and test a new detection algorithm, termed Advanced Partial Completion Filters (APCF), that is scalable, accurate, and easy to adjust.

References

[1] D. Sisalem, J. Kuthan, T. S. Ehlert, F. Fokus, *Denial of Service Attacks Targeting a SIP VoIP Infrastructure: Attack Scenarios and Prevention Mechanisms*, IEEE Network, September/October 2006.

[2] R. R. Kompella, S. Singh, G. Varghese, *On Scalable Attack Detection in the Network*, IEEE/ACM transactions on networking, vol. 15, no. 1, February 2007.

[3] K. Levchenko, R. Paturi, and G.Varghese, *On the difficulty of scalably detecting network attacks*, in Proc. 11th ACM Conf. Computer and Communications Security, 2004, pp. 12-20.

[4] T. M. Gill, M. Poletto, *MULTOPS: A data-structure for bandwidth attack detection*, in Proc. 10th USENIX Security Symp., 2001, pp. 23-38.

[5] C. Estan and G. Varghese, *New directions in traffic measurement and accounting*, in Proc. ACM SIGCOMM, 2002, pp. 271-282.

[6] M. Mitzenmacher, E. Upfal, *Probability and Computing*, Cambridge University Press, 2005.

[7] T. Fawcett, *ROC Graphs: Notes and Practical Considerations for Data Mining Researchers*, Intelligent Enterprise Technologies Laboratory HP Palo Alto, January 7th 2003.