SyFi: A Systematic Approach for Estimating Stateful Firewall Performance

Yordanos Beyene, Michalis Faloutsos, and Harsha V. Madhyastha

Department of Computer Science and Engineering, UC Riverside
{yordanos,michalis,harsha}@cs.ucr.edu

Abstract. Due to the lack of a standardized methodology for reporting firewall performance, current datasheets are designed for marketing and provide inflated throughput measurements obtained under unrealistic scenarios. As a result, customers lack usable metrics to select a device that best meets their needs.

In this paper, we develop a systematic approach to estimate the performance offered by stateful firewalls. To do so, we first conduct extensive experiments with two enterprise firewalls in a wide range of configurations and traffic profiles to identify the characteristics of a network's traffic that affect firewall performance. Based on the observations from our measurements, we develop a model that can estimate the expected performance of a particular stateful firewall when deployed in a customer's network. Our model ties together a succinct set of network traffic characteristics and firewall benchmarks. We validate our model with a third enterprise-grade firewall, and find that it predicts firewall throughput with less than 6-10% error across a range of traffic profiles.

1 Introduction

Which firewall will meet the throughput requirement of our network?

Currently, a customer shopping for a firewall cannot find a good answer to this basic question. Since there is no systematic methodology for evaluating and reporting firewall performance, firewall vendors report (a) unrealistically high performance obtained with unspecified or arbitrarily chosen traffic profiles [1], and (b) performance metrics that can be easily "gamed" [2] as we discuss later. As a result, customers have to either rely on word-of-mouth recommendations or go through the laborious process of testing each firewall themselves.

Thus far, characterizing firewall performance has received limited attention from both industry and researchers. To counter approaches taken by firewall vendors to report unrealistic high throughput numbers (e.g., by using maximum-sized UDP packets), third-party testing agencies such as NSS [23] measure firewall performance using a more realistic pre-defined traffic mix. Though a step in the right direction, this approach is limited in that its results are not applicable for a customer with a different traffic mix. Traffic characteristics vary from site to site, and as we examine later, firewall throughput significantly varies across traffic profiles. On the other hand, the focus of the research community

N. Taft and F. Ricciato (Eds.): PAM 2012, LNCS 7192, pp. 74–84, 2012.

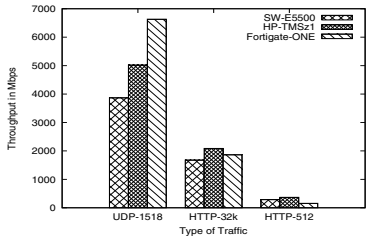


Fig. 1. Variation in maximum throughput across firewalls on different traffic profiles

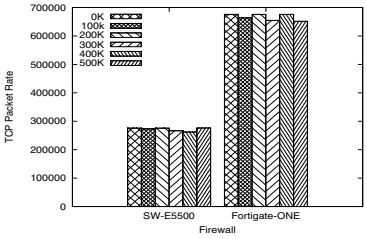


Fig. 2. Effect of the number of active concurrent sessions on maximum packet rate

has mainly been on improving firewall performance by optimizing the firewall rule set [7,12,17], detecting firewall rule conflicts [18,9,8], and developing firewall architectures that make firewalls efficient and prevent such rule conflict errors from occurring [16,15,19,20].

Our goal instead is to develop a meaningful way to characterize the performance of stateful firewalls. Our approach is motivated by the observation that the performance a customer can expect from a firewall depends on characteristics of both network traffic at the customer's site and the firewall's hardware and software. For example, Figure 1 shows the throughput obtained with three enterprise-grade firewalls with three different traffic profiles. We see that the choice of the best firewall depends on the traffic profile, and there is no firewall that performs best in all cases.

In this paper, we first experiment with two enterprise firewalls to determine the characteristics of network traffic that impact their performance. We find that packet sizes and the number of active firewall sessions have minimal impact on the firewall's performance. On the other hand, our measurements show that the protocol and packet type make a significant difference; both firewalls incur much higher processing overhead with 1) TCP packets as compared to UDP, and 2) packets that create new sessions on the firewall compared to data packets that belong to an existing session.

We use these observations to develop a simple model of stateful firewall performance. Our model ties together two inputs—1) a profile of the traffic at the customer's network, and 2) computational costs incurred by a firewall on different types of packets. We prescribe the format for the first input based on our observations of the resource requirements imposed on firewalls by different types of packets. We believe firewall vendors should be specifying the latter input in their datasheets.

We validate our model with a third firewall, different from the two used to derive our model. We apply our model to a range of traffic profiles and find that in each case, our model's estimate of the throughput is within 6% of measured values. We also evaluate our model's ability to predict the firewall's performance when subjected to a SYN flood attack, and here too, its throughput estimates are within a 10% error across traffic profiles.

2 Understanding Firewall Performance

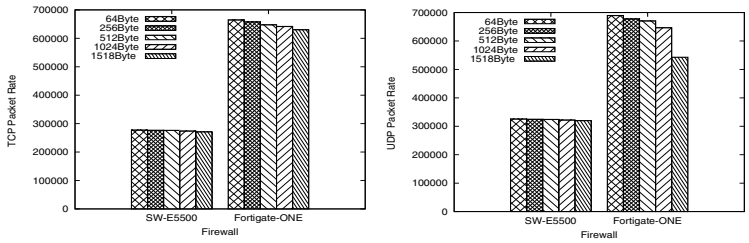
In this section, we provide a brief background on stateful firewalls and then describe our measurement-based approach to characterize them.

Background: A stateful firewall secures a private network by keeping track of flows and enforcing security policies. We use the term flow (interchangeably with session) in its commonly used sense—a stream of packets with the same five tuple: source and destination IP address, source and destination port, and protocol type. A stateful firewall inspects all incoming and outgoing packets and decides to discard or accept a packet based on the sequence of rules in the firewall rule set and its session table. A packet that belongs to a new session is allowed to enter the network if permitted by the firewall rule set, whereas packets corresponding to previously established sessions are let through by looking up the session table.

Measurement-driven characterization. Here, we examine which factors affect the performance of a stateful firewall in practice. We conduct focused experiments for each factor in isolation on two commercial enterprise-level firewalls: (a) SonicWall E5500 [6], and (b) Fortinet Fortigate-ONE [3]. Though both devices come with additional security features such as intrusion detection, here we focus on one of the common deployment scenarios for these devices where they are configured to run as a stateful firewall. This reflects the current status quo in which most customers rely on separate application-aware filtering devices to sit behind stateful firewalls [5], rather than bundling both features on the same device. Note that the devices are not selected based on cost or hardware specification, and hence the purpose of this study is not to compare the firewalls, but rather to ensure that our observations are not unique to any one firewall.

Traffic generator: We use the traffic generation tool from BreakingPoint Systems (BPS) [10] to generate synthetic traffic for our experiments. BPS is a powerful test tool used to measure and analyze the performance, security, and stability of network devices. The BPS version that we use can generate up to 30 million simultaneous sessions, 1.5 million sessions per second, and 16 Gbps of stateful blended application traffic with over 130 application protocols, sufficient to stress-test all the firewalls we considered. It provided us enormous flexibility to simulate conditions needed to characterize the products.

Test setup: In our experiments, we used eight 1 Gbps interfaces on each firewall, matched in pairs as input and output. Our aggregate maximum rate of 8 Gbps sufficed to reach the processing capacity of either firewall. Our test traffic generator serves as the source and destination for all generated flows, and keeps track of the number of transmitted and received packets and reports packet drops. We adopt the industry-wide convention to calculate the performance as the sum of the packets and bytes across all interfaces, irrespective of their direction (i.e. from inside the network going outside or vice versa). Though a firewall's performance can be affected by its rule set (ACL) size, given the operation on an ACL (it only affects the first packet of every flow) and the optimizations that one can



(a) TCP Packet Rate (b) UDP Packet Rate

Fig. 3. Maximum packet rate with different packet sizes

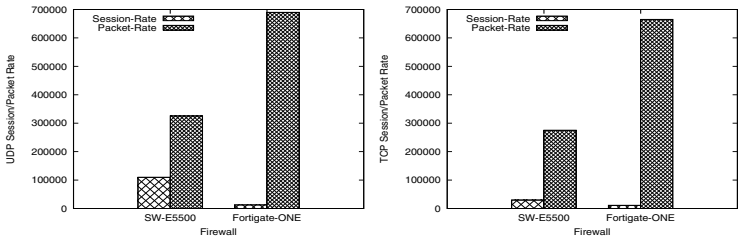
perform on these rules (e.g., reordering of rules) to lower the number of ACL hits, we isolate the performance of the firewall from the ACL as a first order of approximation. We do a separate series of experiments in Section 4 to identify the effect that the firewall's ruleset has on its performance.

Impact of concurrent sessions. First, we investigate whether the number of concurrently active firewall sessions affects a firewall's performance. To study this, we introduce a set of *background* sessions; these are flows on which we do not send any packets once they are setup. These flows occupy space in the firewall's session table without adding traffic flowing through the firewall. We ensure that the session timeout on the firewall is long enough that these background sessions do not expire early. In addition, we have *regular* sessions on which we do send packets after they are created.

In each test, we start with one flow that sends 10,000 packets/second, and every 60 seconds we add a new flow that sends 10,000 packets/sec. Note that every flow stays active until the experiment stops, so the number of packets increases every 60 seconds. We need to add new flows to increase packet rate because our traffic generating tool can at most generate 10,000 packets/sec per flow. We stop creating more flows when the device can handle no more load, i.e., when we see an onset of persistence packet drops, and record the maximum packet rate that the firewall can handle. We found that the specific numbers used here (1 flow every 60 seconds, 10k packets/sec/flow) provide enough granularity and time for the system to stabilize and allows us to detect the point of load saturation with reasonable accuracy.

We measured the maximum packet rate as above with the creation of the background sessions and the regular sessions interspersed, with the background sessions created first, and with the background sessions created last to capture the performance impact of the order in which sessions are created. In each case, we experimented with varying number of background sessions ranging from 0 to 500K in increments of 100K. In all cases, as shown in Figure 2, we found that the number of active sessions in the firewall table had minimal impact on the maximum packet rate on either device.

Impact of packet size on packet rate. Next, we measure the impact of packet size on firewall performance. We vary the packet size across 64, 256, 512,



(a) UDP (b) TCP

Fig. 4. Session rate versus packet rate

1024 and 1518 bytes, and in each case, measure the maximum packet rate as above. Figures 3a and 3b illustrate the maximum packet rates for TCP and UDP packets. Both figures show that packet size has negligible impact; the maximum packet rate declines by less than 3% for TCP and by less than 5% for UDP when packet size increases from 64 bytes to 1518 bytes. Thus, packet size has minimal impact on packet rate; *a firewall vendor can inflate the throughput in bytes/second by using the maximum packet size possible.*

We also note that though our approach is agnostic to the specific bottleneck resource as it treats the firewall as a black box, we observed the CPU to be the bottleneck in all cases and packet loss occurred when CPU utilization reached nearly 100%. We observed an isolated incident with Fortigate-ONE where maximum packet rate with UDP drops by nearly 20% when the packet size is 1518 bytes. This could be due to an implementation inefficiency in the tested firmware version.

Cost of creating a session. We next compare the overhead of processing packets that create sessions on the firewall with that of processing subsequent packets that belong to an active session. We do this by computing the maximum session rate as described next and comparing it with the previously measured maximum packet rates. We begin by creating 5K new flows every second and increase the rate by 1K flows every 60 seconds until the firewall resources are exhausted and we start observing persistent packet drops. Thus, the firewall establishes 5K new sessions every second in the first cycle, 6K new sessions every second in the second cycle, and so on. For every flow, we stop sending packets after it is created. We also lowered the session timeout on the firewall to force it to flush sessions regularly; packet loss is thus caused due to the firewall's resource limitation, and not because the session table is full. We record the maximum session rate as the maximum rate of session creation reached before we start observing packet drops.

Figures 4a and 4b compare the maximum session rates and packet rates for UDP and TCP. The large difference between the session and packet rates is indicative of the much higher cost of establishing a session compared to processing subsequent packets. Though the extent of variation between creating sessions and processing subsequent packets varies across the two firewalls, this is to be expected since each firewall runs its own proprietary software.

Impact of transport protocol. We make two other observations from our measurements of maximum packet and session rates. First, we see that establishing a TCP session is more expensive than a UDP session. Second, we find that processing a TCP data packet is more expensive than a subsequent UDP packet. These findings show that packet processing costs depend on the transport protocol of packets.

Finally, we varied application level properties, such as the generating application or sender and receiver port numbers, but we observed no significant performance impact. Note that the application type may have significant impact for security devices that inspect payload which is beyond the scope of this paper.

3 SyFi: Model for Firewall Performance

Based on the insights from our measurements, we develop a systematic approach for estimating the performance of any stateful firewall. As summarized in Figure 5, given a firewall and a traffic profile as input, our model outputs the maximum throughput that the firewall can sustain.

Traffic profile. First, our model represents any input traffic profile as the mix of four packet types, which we denote with T_t, $t = 1, 2, 3, 4$. These four packet types are: a) packets associated with establishing TCP sessions (TCP SYN packets), (b) TCP data packets, (b) the first packets of a UDP flow, and (d) the subsequent packets of a UDP flow [1]. Thus, we represent a traffic profile with the probability P_t that a packet of type T_t is seen. These probabilities can be directly computed from a traffic trace gathered at the target network. There may be some challenges in isolating UDP packets that create sessions from those that belong to an existing session as firewalls clear sessions that have been inactive for longer than the firewall time-out. The subtle but reasonable assumption that can be made is to consider flow with inter-packet intervals longer than firewall time-outas the beginning of a new flow. On the other hand, if the traffic description is in terms of the application layer (e.g., 80% HTTP and 20% FTP), it is straightforward to calculate the probabilities if (a) the fraction of flows per application, and (b) the average number of packets per flow is known.

Firewall profile. Second, based on the measurements we described in Section 2, we measure the overhead or cost of each packet type T_t, denoted as C_t. In our case, we use the CPU utilization associated with processing each packet type as the cost, but more generally, this is the utilization of the firewall bottleneck. We compute the cost C_t for each of the four packet types T_t as $\frac{1}{Max_t}$, where Max_t is the maximum rate measured using the steps described in Section 2. Though this step of characterizing costs is resource intensive, it needs to be performed only once for each firewall device and the measurements can be reused across all traffic scenarios. Recall that packet sizes, firewall session table size, and application level protocol type were seen to have minimal effect on performance.

[1] We focus on UDP and TCP packets as they account for over 95% of Internet traffic [25], but the model can be extended to include packets such as ICMP.

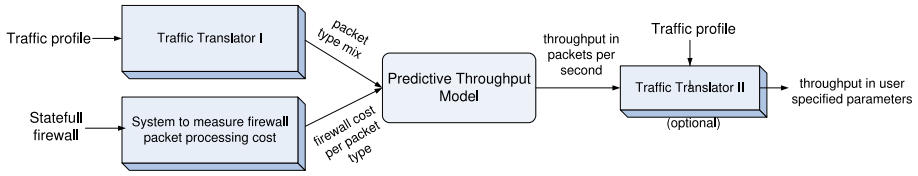


Fig. 5. Systematic Firewall Throughput Approach

3.1 Analytical Model

Finally, we calculate the expected throughput of the firewall on the input traffic profile in terms of packets per second, which we denote by N. Given the probability P_t and the cost C_t for each of the four packet types, we compute the cost (CPU utilization) for N packets per second as follows.

$$CPU = N \times \sum_{t=1}^{4} (C_t \times P_t) \tag{1}$$

The maximum number of packets occurs when the CPU is fully utilized, i.e., when CPU utilization is 1 (= 100%). Thus, we compute the maximum number of packets per second as follows.

$$N = \frac{1}{\sum_{t=1}^{4} (C_t \times P_t)} \tag{2}$$

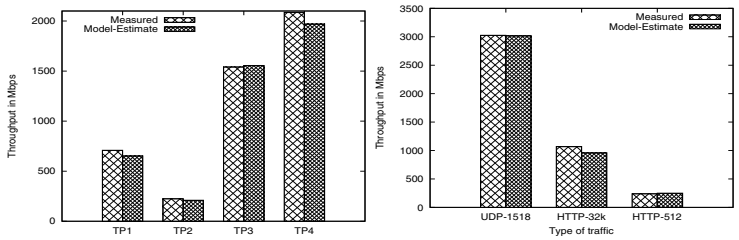
Though the final equation that lies at the heart of our model may appear simple, note that the simplicity of the model stems from our experimental observations, before which how to estimate a firewall's performance was unclear. For example, the facts that packet size and number of concurrent sessions have minimal impact on performance are key observations that enables our approach to be simple, yet highly accurate (as we show later in Section 4).

Once we have the firewall performance in packets per second, it is straightforward to calculate the throughput in any metric that the user prefers. For example, the Traffic Translator II in our approach can be used to compute firewall throughput in bytes per second, as long as average packet size is available.

4 Validation of the Model

In this section, we validate the accuracy of our model by comparing its predictions with actual measurements. We perform this validation when using the model in two scenarios—1) to estimate a firewall's performance on normal traffic, and 2) to estimate a firewall's performance when under a SYN flood attack.

In the first part of our validation, we present here results from the HP TM-Szl firewall [4]—a different device from the ones we used to develop our model in Sections 2 and 3; we measure packet processing costs using the mechanism described in Section 2 for the four types of packets considered in our model.



(a) Normal traffic (b) SYN flood attack

Fig. 6. Comparison of measured throughput with throughput estimates from our model

We evaluate our model on four disparate profiles of normal traffic, which test different variations with respect to protocol type, packet size, and packet counts per flow. In the second part, we evaluate on the ability of our model to predict the performance of the SonicWall firewall when under a SYN flood attack.

Normal traffic. The four traffic profiles we consider are as follows. In our baseline traffic profile *TP1*, 20% of flows correspond to TCP traffic and the remaining 80% of flows are UDP. UDP flows send 100 packets/second and the average packet size is 64 bytes, while TCP flows send 1000 packets/second, and the average packet size is 512 bytes. In our second traffic profile *TP2*, we keep all parameters the same as in *TP1* but make all flows shorter; we decrease the number of packets per TCP and UDP flow to 10 and 1, respectively. Shorter flows result in a higher rate of session creation, which as we have shown earlier is an expensive operation for stateful firewalls. Third, in *TP3*, we change the average packet sizes in our baseline profile to make packets larger; we make average packet sizes for TCP and UDP flows to be 1024 bytes and 512 bytes respectively. Lastly, we use the standard HTTP-32K traffic profile generated by our BPS traffic generation tool. We analyzed the packets generated by our test tool and found that every HTTP flow contains 52 packets which includes 3 packets each for TCP connection setup and teardown, 1 HTTP GET, 1 ACK for the HTTP GET, 22 TCP data packets, and 22 TCP ACK packets. We use this information to compute the probabilities for our fourth traffic profile *TP4*.

For each of the four traffic profiles, we measure the maximum throughput that the HP TMSzl firewall can support by generating the corresponding traffic from our traffic generation tool and scaling up the rate of the traffic until we begin to observe packet drops. We then compare the measured values with the corresponding estimates provided by our model. Figure 6 shows that our model's estimates are within 6% of the measured values for all four traffic profiles. These results highlight our model's ability to correctly predict the effects of variations in flow duration, packet sizes, and protocol types.

Attack traffic. Beyond estimating the throughput that a firewall can support on the normal traffic at a customer's site, customers may also want to ensure that the firewall they choose can support their traffic even when under attack. To evaluate the utility of our model in such a scenario, we consider the problem

of estimating firewall throughput in the face of a SYN flood attack. For this experiment, we consider three different standard traffic patterns in our traffic generation tool—UDP flows with 1518 bytes of data, HTTP flows with 32 KB of data, and HTTP flows with 512 bytes of data on average. We subject the SonicWall firewall to these traffic patterns in turn, and in each case, use our traffic generation tool to launch 10K SYN packets per second in parallel. Figure 6(b) compares the measured values of the throughput sustained by the firewall with the corresponding estimates from our model. Our model's estimates are accurate with less than 10% error in all three traffic profiles.

The experiments reveal significant throughput change when traffic profile varies. Thus the 5 Gbps data sheet through for the HP TMSzl [4], and the 3.9 Gbps throughput for SonicWall [6] is over-optimistic.

Finally, we performed a preliminary investigation of the effect of the firewall's ruleset, i.e., the set of ACLs used to filter traffic, on session rate. We ran an experiment wherein we populated the firewall's ruleset with several DENY rules that do not match our test traffic, and added a single rule at the bottom to permit the test traffic. We repeated this experiment on all three firewalls, varying the number of rules and the test traffic. In all cases, we found that the maximum session rate significantly declines with an increase in the number of rules.

5 Related Work

RFC 2544 [11] is the industry leading network interconnecting device benchmarking methodology since 1999. RFC 2647 [22] extends the terminology established with definitions specific to firewalls. It specifies device throughput to be tested with frames sized 64, 128, 256, 512, 1024 and 1518 bytes. Firewall vendors exploit RFC 2544 by publishing throughput measurements in bytes/sec conducted with full-sized (1,518 bytes) UDP packets only. Clearly, this does not reflect a typical enterprise traffic mix, which involves different protocols and packet sizes.

Third-party testing agencies such as NSS [23] have challenged vendors by evaluating and comparing firewalls with respect to their performance, security, and stability. NSS measure throughput using a more realistic traffic mix. Though a step in the right direction, it doesn't address a customer's basic question of how the firewall will perform in her network since the firewall's throughput varies significantly across traffic profiles.

The focus of the research community has been on 1) optimizing firewall rules either by reordering rules or removing redundant rules [7,12,17], 2) detecting policy errors in a firewall's ruleset [21,18,9], 3) detecting anomalies and vulnerabilities in firewalls [9,24,14], and 4) improving firewall design [16,15] to prevent the policy errors and anomalies from happening in the first place.

To the best of our knowledge, there is no prior work to answer the question—*what is the performance a given customer can expect from a particular firewall?* Customers may therefore end up buying a low performing device which could potentially introduce a bottleneck into their network, or a high performing device that is more expensive than necessary. In this paper, we resolve this issue in a

systematic way, and our model accurately computes the expected performance of a firewall using a profile of the traffic profile from the deployment environment.

Perhaps the most closely related work in terms of resource based performance analysis is that of Dreger et al. [13]. They developed `nidsconf`, a tool that examines Intrusion Detection Systems resource utilization on a sample traffic profile and derives configurations that prevent bursts of packet drops due to spikes in CPU and memory utilization.

6 Conclusions

Since performance numbers in the datasheets of firewalls are often inflated, customers of stateful firewalls today are left needing to rely on word of mouth recommendations to choose a firewall. To make the process more scientific, in this paper, we examined two state-of-the-art enterprise-level stateful firewalls to highlight the factors that affect their performance. Based on our observations that protocol and packet type matter for performance but packet sizes and number of concurrent sessions do not, we developed a model of firewall performance that takes as input characteristics of the particular firewall and the traffic at a target network. Our evaluations on a third firewall showed that our model can estimate throughput across different traffic profiles with over 90% accuracy. In the future, we will study the performance impact of payload inspection and connection teardown packets.

References

1. Comparison shopping for scalable firewall products, `http://tinyurl.com/7smaqet`
2. Data sheets lie: How to measure the performance, security and stability of network devices,
   `http://resources.breakingpoint.com/acton/form/567/0024:d-0004/0/`
3. Fortinet FortiGate-ONE,
   `http://www.fortinet.com/products/fortigate/one.html`
4. HP Threat Management Services zl module,
   `http://h20195.www2.hp.com/v2/GetPDF.aspx/4AA2-6512ENN.pdf/`
5. Next Generation Firewalls not ready to replace all legacy firewalls,
   `http://searchnetworking.techtarget.com/news/1520651/`
   `Next-generation-firewalls-not-ready-to-replace-all-legacy-firewalls/`
6. SonicWALL E-class network security appliance E5500,
   `http://www.firewalls.com/sonicwall/`
   `sonicwall-firewall/sonicwall-e-class-series/`
7. Acharya, S., Wang, J., Ge, Z., Zane, T.F., Greenberg, A.: Traffic-aware firewall optimization strategies. In: ICC (2006)
8. Al-Shaer, E., Hamed, H., Boutaba, R., Hasan, M.: Conflict classification and analysis of distributed firewall policies. In: IEEE JSAC (2005)
9. Baboescu, F., Varghese, G.: Fast and scalable conflict detection for packet classifiers. In: IEEE ICNP (2002)
10. BreakingPoint firewall performance testing,
    `http://www.breakingpointsystems.com/solutions/firewall-testing/`

11. Bradner, S., McQuaid, J.: Benchmarking methodology for network interconnect devices. RFC 2544 (1999)
12. Cohen, E., Lund, C.: Packet classification in large ISPs: Design and evaluation of decision tree classifiers. In: ACM SIGMETRICS (2005)
13. Dreger, H., Feldmann, A., Paxson, V., Sommer, R.: Predicting the Resource Consumption of Network Intrusion Detection Systems. In: Lippmann, R., Kirda, E., Trachtenberg, A. (eds.) RAID 2008. LNCS, vol. 5230, pp. 135–154. Springer, Heidelberg (2008)
14. El-Atawy, A., Al-Shaer, E., Tran, T., Boutaba, R.: Adaptive early packet filtering for protecting firewalls against DoS attacks. In: IEEE INFOCOM (2009)
15. Gouda, M.G., Liu, A., Jafry, M.: Verification of distributed firewalls. In: IEEE GLOBECOM (2008)
16. Gouda, M.G., Liu, A.X.: Structured firewall design. Computer Networks (2007)
17. Hamed, H., Al-Shaer, E.: Dynamic rule-ordering optimization for high-speed firewall filtering. In: ASIACCS (2006)
18. Hari, A., Suri, S., Parulkar, G.: Detecting and resolving packet filter conflicts. In: IEEE INFOCOM (2000)
19. Liu, A.X.: Change-impact analysis of firewall policies. In: European Symp. Research Computer Security (2007)
20. Liu, A.X.: Firewall policy verification and troubleshooting. In: ICC (2008)
21. Liu, A.X., Gouda, M.G.: Firewall policy queries. IEEE Trans. on Parallel and Distributed Systems (2009)
22. Newman, D.: Benchmarking terminology for firewall devices. RFC 2647 (1999)
23. NSS Labs. IPS, UTM, Web application firewall testing lab, `http://nsslabs.com`
24. Shaer, E.A., Hamed, H.: Discovery of policy anomalies in distributed firewalls. In: IEEE INFOCOM (2004)
25. Caceres, R.: Measurements of Wide-Area Internet Traffic, UCB/CSD.89/550, Univ. CA, Berkeley (1989)