Adaptive Selective Verification: An Efficient Adaptive Countermeasure to Thwart DoS Attacks

Sanjeev Khanna, Santosh S. Venkatesh, *Member, IEEE*, Omid Fatemieh, Fariba Khan, and Carl A. Gunter, *Senior Member, IEEE, Member, ACM*

***Abstract*—Denial-of-service (DoS) attacks are considered within the province of a shared channel model in which attack rates may be large but are bounded and client request rates vary within fixed bounds. In this setting, it is shown that clients can adapt effectively to an attack by increasing their request rate based on timeout windows to estimate attack rates. The server will be able to process client requests with high probability while pruning out most of the attack by selective random sampling. The protocol introduced here, called *Adaptive Selective Verification* (ASV), is shown to use bandwidth efficiently and does not require any server state or assumptions about network congestion. The main results of the paper are a formulation of optimal performance and a proof that ASV is optimal.**

***Index Terms*—Bandwidth, distributed denial of service (DDoS), performance analysis, selective verification, shared channel model, theorem.**

I. Introduction

DENIAL-OF-SERVICE (DoS) attacks are a growing concern as they continue to pose an elevated threat to the reliability of the Internet. Such attacks can occur at all levels in the protocol stack and threaten both routers and hosts. Many attacks aim to deplete scarce resources (e.g., CPU, memory, disk) by generating illegitimate requests from one or many, possibly compromised, attacker-controlled hosts [14], [17], [2], [15], [13], [16]. The time required to process these requests degrades the service to available clients to an unacceptable degree or forces costly overprovisioning by the service provider. Instances of potentially vulnerable services include IKE key exchanges for gateway security association setup [9], legacy and digitally signed DNS services [7], large file retrievals from Web servers, and computationally expensive query processing at database front ends.

A variety of countermeasures have been proposed to address these problems. *Currency-based mechanisms* are ones in which a server under attack demands some type of payment from clients in order to raise the bar for provoking work by the server. Classic currency examples in this context are *money* [12] and *CPU cycles* [22], [3], [6]. Our focus in this paper is on the use of *bandwidth* as currency. In order to get service, the clients are encouraged to spend more bandwidth by either sending repeated requests from which the server selectively verifies (processes) some requests [4], [8], [19] or sending dummy bytes on a separate channel to enable a bandwidth auction [21]. Currency-based mechanisms impose a cost on the system, so it is desirable to have *adaptive* countermeasures that are deployed dynamically and proportionally to blunt attacks at minimal cost. Auction-based bandwidth payments accomplish this by an accounting system in which clients to build credit by sending dummy bytes in congestion-controlled streams, and the server periodically takes requests from clients that have built the most credit. This may require significant server state and is vulnerable to adversaries who are able to create network congestion that causes legitimate clients to back off while attackers ignore backoffs. Selective verification requires no server state or congestion assumptions. However, the existing state of the art does not provide any precisely analyzed strategy for adaptation.

In this paper, we introduce *Adaptive Selective Verification* (ASV), which is a distributed adaptive mechanism for thwarting attackers' efforts to deny service to legitimate clients based on selective verification. Our scheme uses bandwidth as currency, but the level of protection employed by the clients dynamically adjusts to the current level of attack. At a high level, the clients exponentially ramp up the number of requests they send in consecutive time-windows, up to a threshold. The server implements a reservoir-based random sampling to effectively sample from a sequence of incoming packets using bounded space. This enables adaptive bandwidth payments with server state whose size remains small and constant regardless of the actions of the attacker. While the protocol itself is both natural and simple, analyzing its performance turns out to be a rather intricate task. A primary contribution of this work is a novel theoretical analysis of ASV whereby we evaluate its performance as compared to an "omniscient" protocol in which all attack parameters are instantaneously made known to all clients as well as the servers. Surprisingly, we show that ASV closely approximates the performance of this omniscient protocol. The performance is measured in terms of the success probability of each client and the total bandwidth consumed by the clients. We also perform an empirical evaluation of the adaptive selective verification protocol with the aim of understanding its performance in practice. Besides validating our theoretical guarantees, our simulations show that under

Manuscript received March 18, 2010; accepted August 11, 2011; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor M. Kodialam. Date of publication October 20, 2011; date of current version June 12, 2012. This work was supported in part by the NSF CNS under Grant 05-24516 and a grant from Boeing. The views expressed are those of the authors only.

S. Khanna and S. S. Venkatesh are with the University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: sanjeev@cis.upenn.edu; venkatesh@seas.upenn.edu).

O. Fatemieh, F. Khan, and C. A. Gunter are with the University of Illinois at Urbana–Champaign, Urbana, IL 61801 USA (e-mail: sfatemi2@illinois.edu; fkhan2@illinois.edu; cgunter@illinois.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2011.2171057

a time-varying attack, the performance of the ASV protocol adjusts extremely quickly to the prevailing attack parameters.

II. Related Work

Protection mechanisms that assure availability remain a difficult challenge for the Internet. Attacks can come from many sources in distributed DoS (DDoS) or from a single source where spoofed addresses create the appearance of a DDoS attack. They can occur at link, network, transport, or application layers. They can be sudden and dramatic or gradual and subtle. Intentional attacks, aimed at disabling services, are easy to confuse with misconfiguration errors or heavy legitimate use. Most network protocols were designed without DoS protection and are vulnerable by design. This rich collection of attack vectors combines with various options for what can be changed to effect a countermeasure. For instance, is it possible to involve the routers, or do solutions need to work exclusively at hosts? Is it possible to change packet formats and protocols, or must these be handled in a transparent manner? Can one assume the attackers have limited knowledge of network state (such as eavesdropping on packets), or must they be assumed to have global knowledge? Solutions vary according to the types of attacks envisioned and these options for countermeasures. For instance, there are trace-back methods [18] to find attackers so action can be taken to cut them off and filter mechanisms to remove attack traffic from the network [11]. There are capability-based mechanisms to allow parties that can prove their legitimacy to gain priority or to limit the impact of unproven parties [10], [23], [24]. Moreover, there are currency-based schemes that aim to limit attackers by making them sacrifice a valuable resource like money or CPU cycles in order to get access to server resources [12], [22], [6], [3].

Perhaps the most counterintuitive DoS countermeasure strategy is the use of *bandwidth* as payment. In such a scheme, clients use additional bandwidth to get access. The idea is that attackers are using all of the bandwidth available to them (or the maximum bandwidth they can afford to use without being detected by other mechanisms) to execute an attack, whereas legitimate clients are using only the resources they require to accomplish their less-demanding objectives. Hence, legitimate clients have bandwidth to spare and can use this fact to differentiate themselves from attackers. This strategy was introduced in [8] in the context of authenticated broadcast using selective verification and extended to general Internet protocols in [21] using bandwidth auctions. *Selective verification* allows clients to send extra requests and the server samples from these requests probabilistically. This technique is very effective in diminishing the effects of a DoS attack if a sufficient level of client redundancy is employed. It is usable in unidirectional communications, requires no server state, and makes no assumptions about network congestion. However, extra client requests are a cost that should ideally be avoided when there is no attack and used proportionately to the strength of attack when there is one. The aim of this paper is to do this by surrendering unidirectional communication capabilities and developing an acknowledgment-based adaptive technique.

There are several works that address adaptive measures for DoS protection in other contexts. Zou *et al.* [26] provide ideas that are effective for filter schemes, although it is unclear how they can be applied to bandwidth payments. Srivastsa *et al.* [20] show how to use information available in the application layer to identify and differentiate between low- and high-utility clients to provide better service to more valuable customers. Their solution requires more feedback from the application than selective verification and is more applicable to scenarios where the clients have a history of interactions with the server. Wang *et al.* [22] show how to provide adaptation for client puzzles. Because of the nature of the client puzzle schemes, where the cost factor of the defense on the server is minimal, their proposal mainly focuses on cost minimization for the clients. However bandwidth payment schemes must account for costs to the server and network as well as the client. Finally, Yau *et al.* [25] propose an adaptive solution for installing router throttles in the network. The main focus of their approach is network flooding attacks and router-based distributed defense against them, but it shares many of the same high-level adaptation concerns as bandwidth payment.

III. Setting

Consider the following one-round client–server protocol. The first step of the protocol is an REQ packet from a client $\mathscr{C}$ to the server $\mathscr{S}$. In response, the server sends back an ACK to the client. Each client employs a *timeout window* of duration T determined by the worst-case expected round-trip delay between the clients and the server: If after transmission of an REQ, a client does not receive an ACK within T seconds, he assumes that the attempt has failed. The parameter T is known to the clients as well as the server.

It will be convenient to partition time into a sequence of windows $W_1, W_2, \ldots$, each of duration T. We suppose that the server $\mathscr{S}$ can process requests at a mean rate of S REQ packets per second so that, in any window, the mean number of requests that it can process is ST. In any given window W, new clients arrive at a rate of $R(W) = \rho(W)S$ clients per second. The *client request factor* $\rho(W) = R(W)/S$ determines the fraction of the server's (computational) bandwidth that is required to process new clients in the window W. We suppose that the client request factors are uniformly bounded above by $0 \leq \rho(W) \leq \rho_{\max} \leq 1$, for some fixed $\rho_{\max}$ in the unit interval.

We will assume that a diffuse, distributed, denial-of-service attack $\mathscr{A}$ on the server takes the form of a potentially time-varying flood of spurious REQ packets aimed at overwhelming the server's capacity to process new REQs. We suppose that, in any given window W, the attack $\mathscr{A}$ sends spurious REQs at a rate of $A(W) = \alpha(W)S$ packets per second. The *attack factor* $\alpha(W) = A(W)/S$ determines the excess bandwidth that will be required of the server to process the illegitimate requests in window W. In keeping with the guiding philosophy of the shared channel model that was articulated by the authors to model DoS attacks [8], we assume that the attack factors are uniformly bounded, $0 \leq \alpha(W) < \alpha_{\max}$, for some fixed $\alpha_{\max}$, though the upper bound $\alpha_{\max}$ on the attack factors may be very large. Clearly, when $\alpha(W) > 1$, the attack overwhelms the server's capacity to process all requests unless there is a mechanism to efficiently handle the attack packets. Our interest is in

the case where $\alpha_{\max} \gg 1$ and the attack can occur on a scale much larger than the available server bandwidth.

In order to focus on the DDoS attack at the receiver, we idealize the situation and assume that REQ and ACK packets are transmitted instantaneously, the round-trip delay occasioned solely by processing time at the server, and that no REQ or ACK packets are lost in transmission. Packet drops at the server are then occasioned only because the arriving request stream from clients and attackers combined exceeds the server's computational bandwidth. Thus, if $\rho_{\max} + \alpha_{\max} > 1$, then it cannot be guaranteed that an individual client's REQ will be processed by the server. If $\alpha_{\max} \gg 1$, it is in principle then possible to almost completely throttle the clients of service and effect a successful DoS attack.

On Notation: As a default convention, when the base is not explicitly specified, logarithms are to be assumed to be to the Napier or natural base e; there are a few places, especially in the proof of Theorem 8, where base 2 is more natural, but in these cases we explicitly specify the base.

IV. Omniscient Protocol

Consider any timeout window W. Suppose that $0 < \rho = \rho(W) < 1$ and $0 < \alpha = \alpha(W)$ denote the client request factor and the attack factor, respectively, over the window W. If clients and the server clairvoyantly know ρ and α, then it is relatively easy for them to thwart the DDoS attack by having the clients send multiple identical requests, with the server implementing a random sampling scheme for processing these requests. Informally speaking, if the server samples incoming requests with probability p, then a legitimate client succeeds with high probability by sending $O(1/p)$ copies of any request. On the other hand, requests sent by an attacker now reach the server at a rate that is reduced by a factor of p. Thus, when p is chosen to be much smaller than 1, it significantly cuts down the effective attack rate.

Of course, the assumption that clients and the server know the client request factor and the current attack rate at all times is rather unrealistic. However, this simple setting is worth analyzing as it provides a performance benchmark for protocols operating under more realistic assumptions.

($\mathscr{C}$) Omniscient Client Protocol: Given $\alpha > 0$ and $\rho > 0$, each new client in a given window W transmits $\lceil \alpha/\rho \rceil$ copies of the REQ packet in that window. Clients who do not receive an ACK from the server within T seconds leave never to return.

($\mathscr{S}$) Omniscient Server Protocol: Given α and ρ, the server accepts an arriving REQ packet in the window W, independently of other arriving packets, with probability $p = \min\{1, (\alpha + \rho\lceil \frac{\alpha}{\rho} \rceil)^{-1}\}$ and discards it with probability $1-p$. The server sends out an ACK for each accepted REQ.

For comparative purposes, we begin by noting two immediate consequences of the protocol. We first ensure that the server is operating within its rated capacity.

Theorem 1: Server utilization in the omniscient protocol is within its rated mean capacity of S packets per second.

Proof: Consider any given window W. The number of attack packets received in this window is αST, where $\alpha = \alpha(W) > 0$ is assumed known to the server. For the given client request factor $\rho = \rho(W) > 0$, the total number of clients arriving during this window is ρST, with each client transmitting $\lceil \alpha/\rho \rceil$ REQs. The total number of REQs received by the server during this window is hence given by

$$N = N(W) = \left\lceil \frac{\alpha}{\rho} \right\rceil \rho ST + \alpha ST = \left(\alpha + \rho \left\lceil \frac{\alpha}{\rho} \right\rceil \right) ST.$$

Accordingly, the expected number of packets processed by the server in window W is given by $pN \leq ST$ so that the server processes REQs within its rated mean capacity. ∎

By design, the bandwidth required by the clients expands to match the attack.

Theorem 2: In any given window W with client request factor $\rho = \rho(W) > 0$ and attack factor $\alpha = \alpha(W) > 0$, the bandwidth consumed by the clients is bounded between αST and $(\alpha + \rho)ST$ REQs.

Proof: The total number of REQs transmitted by clients in window W is $\lceil \frac{\alpha}{\rho} \rceil \rho ST$. As $\alpha/\rho \leq \lceil \alpha/\rho \rceil < \alpha/\rho + 1$, the stated conclusion follows. ∎

It follows that, in the domain $\rho \ll 1$ and $\alpha \gg 1$ of interest, *the cumulative transmission bandwidth consumed by client REQs in the omniscient client–server protocol is approximately αS packets per second.*

The next two results show that the protocol works as advertised by ensuring that each client gets connected with high probability provided the client request factor does not get too large.

Theorem 3: Suppose $0 < \delta < 1$ is a given confidence parameter. If $\rho_{\max} \leq (-2\log\delta)^{-1}$, then the probability that a given client has an REQ accepted is at least $1-\delta$ under the omniscient client–server protocol.

Proof: Consider any window W with client request factor $\rho = \rho(W)$ and attack factor $\alpha = \alpha(W)$. We may as well suppose that the acceptance probability $p < 1$ as else the REQ from each client is guaranteed to be accepted.

Now, each client $\mathscr{C}$ in window W transmits $\lceil \alpha/\rho \rceil$ REQs in the window. The probability that each of these REQs is discarded by the server is given by

$$Q := (1-p)^{\lceil \alpha/\rho \rceil} \leq e^{-p\lceil \alpha/\rho \rceil} = \exp\left(\frac{-\lceil \alpha/\rho \rceil}{\alpha + \rho\lceil \alpha/\rho \rceil} \right) \quad (1)$$

in view of the elementary inequality $1 - x \leq e^{-x}$. As the function $f(x) = x/(\alpha + \rho x)$ increases monotonically with x for $x \geq 0$, it follows that

$$Q \leq \exp\left(\frac{-(\alpha/\rho)}{\alpha + \rho(\alpha/\rho)} \right) = \exp\left(\frac{-1}{2\rho} \right) \leq \exp\left(\frac{-1}{2\rho_{\max}} \right) \leq \delta$$

if $\rho_{\max} \leq (-2\log(\delta))^{-1}$. ∎

The choice $\delta = 0.05$, for instance, yields the standard 95% confidence interval: Any given client has an REQ accepted with confidence at least 95% if $\rho_{\max} \leq 0.1669$.

Thus, for all sufficiently small client request factors $\rho_{\max}$, the omniscient client–server DDoS protocol accepts REQs from all but a small fraction of at most δ of all clients at a cost in transmission bandwidth of (about) αS client packets per second. Somewhat sharper results can be stated asymptotically in the limit of large server bandwidths.

Theorem 4: Suppose $S \to \infty$ and let c be any fixed real constant. Suppose further that, in any given window W, the attack factor satisfies $\alpha/\log(ST) \to \infty$ and the client request factor satisfies

$$\rho = \frac{1}{2\log(ST)}\left[1 + \frac{\log\log(ST)}{\log(ST)} - \frac{c}{\log(ST)} + \mathcal{O}\left(\frac{\log\log(ST)^2}{\log(ST)^2}\right)\right]. \quad (2)$$

Let $M = M_{ST}$ denote the number of clients that are rejected by the server in the window W. Then, under the omniscient client–server protocol, the number of rejected clients $M = M_{ST}$ converges in law to the Poisson distribution with mean e^{-c}. More specifically, for every nonnegative integer m

$$\Pr\{M_{ST} = m\} \to e^{-e^{-c}}\frac{(e^{-c})^m}{m!}$$

as $S \to \infty$, and, a fortiori, the probability that no new clients are rejected in a given window tends asymptotically to $e^{-e^{-c}}$.

Proof: The number of rejected clients M_{ST} is binomially distributed and corresponds to the number of successes obtained in ρST tosses of a coin whose success probability is $Q = (1-p)^{\lceil \alpha/\rho \rceil}$. In view of the given conditions on α and ρ which ensure that $\alpha\rho \to \infty$, we may refine the estimate given in (1) to obtain the sharper result

$$Q = (1-p)^{\lceil \alpha/\rho \rceil} = \exp\left[\left\lceil \frac{\alpha}{\rho} \right\rceil \log(1-p)\right] = \exp\left[\left\lceil \frac{\alpha}{\rho} \right\rceil \{-p + \mathcal{O}(p^2)\}\right]$$

asymptotically by Taylor's formula for the logarithm. As $\alpha p \to 1/2$, it follows that $Q/e^{-1/2\rho} \to 1$. But with ρ given asymptotically by (2), we have $\rho ST e^{-1/2\rho} \to e^{-c}$ as $S \to \infty$, whence the expected number of clients rejected in the window satisfies $\rho STQ \to e^{-c}$. The claimed result follows in view of the classical convergence of the binomial to the Poisson. ∎

The fine order of infinity manifested in the expression (2) is worthy of note: the first term $[2\log(ST)]^{-1}$ is dominant (a *phase transition*), while the asymptotically emergent behavior for the blocking probability is squirrelled away within the constant in the second subdominant term. Roughly speaking, any ρ less than $[2\log(ST)]^{-1}$ will result in almost all clients being accepted; and any ρ larger than $[2\log(ST)]^{-1}$ will result in a significant fraction of rejections.

V. Adaptive Selective Verification Protocol

The assumption in the omniscient client–server protocol that clients are continuously aware of the client request factor and the attack factor current in each window is clearly unrealistic, especially given the distributed and—until connection is established—as yet unknown location and legitimacy of the clients and, more critically, the ability of the attack to vary rates continuously and unpredictably. Designing a protocol for the worst-case attack is, of course, possible, but unnecessarily congests the network during periods when the attack is quiescent or at low levels. Our goal, hence, is to design a client–server DDoS protocol that adapts to the behavior of the attack $\mathscr{A}$ *without clients having access to explicit current information about the nature and intensity of the attack.*

In view of our experience with the omniscient protocol, on the client side we are led to seek a replicating protocol where the replication rate used by the clients should ideally be proportional to the current attack factor (and inversely proportional to the current request factor though this is likely to be under better regulation). While the client does not have direct access to this information, he can infer the state of the attack indirectly based on whether he receives an ACK or not in response to REQ(s) sent in the previous window. The failure to receive an ACK in response to transmitted REQ(s) can be construed provisionally as evidence of an attack in progress, and the client can then ramp up his replication rate in an effort to counter current attack conditions. Experience with doubling algorithms (or, on the flip side, exponential backoff in TCP protocols) suggests that it would be profitable to have the replication rate grow exponentially with repeated connection failures (up to a worst-case maximum).

On the server side, a more detailed picture about current conditions can be directly obtained from the ensemble of packets arriving in each timeout window. The server can now very simply maintain the advertised mean service rate by reservoir sampling to generate a random sample of the sequentially arriving packets. The randomized sampling of incoming packets helps obviate timing attacks or the exercise of other overt control by the adversary over the decision-making process at the server, while the adaptive changes in sampling rates that reservoir sampling accords allows the server to respond to changes in attack factors across windows while staying within the budgeted mean service bandwidth.

These considerations lead to our ASV protocol.

($\mathscr{C}$) ASV Client Protocol: Given $\rho_{\max}$, $\alpha_{\max}$, and T, after each unsuccessful attempt, the protocol adaptively increases the number of REQs sent in the succeeding timeout window up to a maximum number specified by the given parameters.

C1. [*Initialize replication count.*] Set $j \leftarrow 0$ and $J \leftarrow \lceil \log(\frac{\alpha_{\max}}{\rho_{\max}})/\log(2) \rceil$.

C2. [*Double replication.*] Send 2^j REQ packets to the server.

C3. [*Timeout.*] If no ACK packet is received within T time units, set $j \leftarrow j+1$; if an ACK packet is received, exit the initiation protocol and proceed to the next phase of communication.

C4. [*Iterate till exit condition.*] If $j \le J$, go back to step C2; else exit without communicating with the server.

($\mathscr{S}$) ASV Server Protocol: The server performs reservoir sampling on incoming REQ packets during each timeout window. Given S and T, the server processes a random subset of the arriving REQs at a mean rate not exceeding S packets per second.

S1. [*Initialize window count.*] Set $k \leftarrow 1$.

S2. [*Form reservoir.*] Store the first $\lfloor ST \rfloor$ REQ packets arriving in window W_k in a reservoir. If timeout expires without filling the reservoir, go to step S4. Else, set REQ packet count $j \leftarrow \lfloor ST \rfloor + 1$.

S3. [*Randomly sample incoming packets.*] If there is an incoming REQ numbered j, accept it for placement into the reservoir with probability $\lfloor ST \rfloor / j$ and discard it with probability $1 - \lfloor ST \rfloor / j$. If the REQ is

accepted for placement in the reservoir, discard an REQ from the reservoir uniformly at random and replace it with the accepted packet. Set $j \leftarrow j+1$ and iterate until the timeout window expires.

S4. [*Timeout*.] Accept the packets in the reservoir and send out an ACK for each accepted REQ.

S5. [*Iterate*.] Empty the reservoir, set $k \leftarrow k+1$, and go back to step S2.

We call the quantity

$$J = \left\lceil \log\left(\frac{\alpha_{\max}}{\rho_{\max}}\right) \Big/ \log(2) \right\rceil = \left\lceil \log_2\left(\frac{\alpha_{\max}}{\rho_{\max}}\right) \right\rceil$$

the *retrial span* of a client. In the event that the attack is launched at maximum severity, a client can replicate packets over a period of J windows until he achieves a maximum replication of $\alpha_{\max}/\rho_{\max}$ matched to the peak attack.

We have streamlined the protocols to focus on the critical ideas. In particular, we adopt the convenient fiction that step S4 in the server protocol occurs *instantaneously*. Thus, there is no gap in time between the expiration of a timeout window and the identification of the random subset of packets that is accepted by the server over that window.

Reservoir sampling dates to Fan *et al.* [5] and permits a sequential random selection of arriving packets. Specializing to our setting, we obtain the following lemma.

Lemma 1: If N REQ packets arrive in a given timeout window, then each packet is accepted with probability $p = \min\{1, \lfloor ST \rfloor / N\}$ and discarded with probability $1-p$, independently of the other packets.

The expected number of accepted REQs in a given window W during which N requests were received is hence pN, and as this is bounded by ST, we obtain the following theorem.

Theorem 5: Server utilization in the adaptive selection protocol is within its rated mean capacity of S packets per second.

A. Blocking Probabilities

A first step in the analysis of the Adaptive Selective Verification Protocol is to consider the attrition rate of clients. We begin by showing that each client succeeds in establishing a connection with essentially the same confidence guarantee as in the omniscient case at the expense of some added delay.

Theorem 6: Suppose $0 < \delta < 1$ is a given confidence parameter. If $\rho_{\max} \leq (-5\log(\delta))^{-1}(1-(ST)^{-1})$, then under the ASV protocol, any given client will establish a connection with the server within $JT = T\lceil \log(\frac{\alpha_{\max}}{\rho_{\max}})/\log(2) \rceil$ seconds with probability at least $1-\delta$; the client is turned away with probability no larger than δ.

The nuisance factor $1-(ST)^{-1}$ in the upper bound accounts for the correction due to integer roundoff; in the case of interest where ST is large, it is essentially negligible. As will be clear from the proof, the upper bound on the client request rate can be improved slightly at the expense of a slightly more inscrutable condition.

We say that a client is in *generation g with respect to window W* if he initiated his protocol g windows in advance of W. We begin with two simple observations.

Lemma 2 (Traffic Bound): The total number of REQs, legitimate and illegitimate, received by the server during any window W can be bounded by

$$N(W) \leq N^* := 5\alpha_{\max} ST. \tag{3}$$

Proof: Consider any window W. Client requests in window W may then arise from clients in generations $g = 0, 1, \ldots, J$, with any client in generation g submitting 2^g REQs during W. As the number of generation g clients does not exceed $\rho_{\max} ST$, it follows that the number of generation g client REQs received by the server during window W cannot exceed $\rho_{\max} ST 2^g$, while the number of attack packets received during W cannot be in excess of $\alpha_{\max} ST$. It follows that the total number of REQs, both legitimate and illegitimate, received by the server during window W is bounded by

$$\begin{aligned} N(W) &\leq \sum_{g=0}^{J} \rho_{\max} ST 2^g + \alpha_{\max} ST \\ &= \rho_{\max} ST\left(2^{J+1}-1\right) + \alpha_{\max} ST \\ &\leq 2\rho_{\max} ST 2^J + \alpha_{\max} ST \end{aligned}$$

via the standard geometric summation. As $J = \log(\frac{\alpha_{\max}}{\rho_{\max}})/\log(2) + t$ for some $0 \leq t < 1$, we have $2^J < 2\alpha_{\max}/\rho_{\max}$, and the claimed result follows. ∎

The uniform bound on the number of packets that can be received during any window allows us to bound the probability that a client repeatedly fails to establish a connection with the server.

Lemma 3 (Blocking Probability): Suppose $\alpha_{\max} > 1/5$. Then, the probability that a client $\mathscr{C}$ in generation g with respect to a window W fails to establish a connection with the server by the end of W is bounded by

$$Q_g(W) \leq \exp\left(\frac{-2^g \lfloor ST \rfloor}{N(W)}\right) \leq \exp\left(\frac{-2^g \lfloor ST \rfloor}{N^*}\right)$$

for each $0 \leq g \leq J$.

Proof: The probability $Q_g(W)$ that $\mathscr{C}$ is blocked through $g+1$ successive windows is bounded above by the probability that he is blocked in W conditioned on being blocked in the previous g windows $W-1, W-2, \ldots, W-g+1$. However, a surviving generation g client of W will transmit 2^g REQs in W, and each of these REQs is dropped with probability $1-p = \max\{0, 1-\lfloor ST \rfloor/N(W)\} \leq 1-\lfloor ST \rfloor/N(W)$. Accordingly

$$Q_g(W) \leq \left(1 - \frac{\lfloor ST \rfloor}{N(W)}\right)^{2^g} \leq \exp\left(\frac{-2^g \lfloor ST \rfloor}{N(W)}\right)$$

by another deployment of the inequality $1-x \leq e^{-x}$. ∎

The proof of Theorem 6 is now in hand. Reusing notation, write $Q(\mathscr{C})$ for the probability that $\mathscr{C}$ fails to make a connection and leaves the protocol—this is the *blocking probability for client $\mathscr{C}$*. Setting $g = J$ in the Blocking Probability Lemma and observing that $2^J \geq \alpha_{\max}/\rho_{\max}$, we obtain

$$\begin{aligned} Q(\mathscr{C}) \leq Q^* &:= \exp\left(\frac{-2^J \lfloor ST \rfloor}{5\alpha_{\max} ST}\right) \\ &\leq \exp\left\{\frac{-1}{5\rho_{\max}}\left(1 - \frac{1}{ST}\right)\right\} \end{aligned} \tag{4}$$

as $ST - 1 < \lfloor ST \rfloor \leq ST$. Theorem 6 follows.

Tighter asymptotic results may be shown along the lines of Theorem 4 with essentially the same blocking probability rates. We shall content ourselves with a one-way result indicative of what is available.

Theorem 7: Let $M = M(W)$ denote the number of new clients arriving in window W who fail to make a connection with the server and eventually leave the protocol. If $\rho_{\max} \leq 1/(5\log(ST))$, then $\Pr\{M(W) = 0\} \to 1$ as $S \to \infty$.

Proof: Let Z_k be the indicator for the event that client $\mathscr{C}_k$ arriving during window W fails to make a connection with the server and eventually leaves. The number of connection failures for clients newly arrived in window W is then $M(W) = \sum_{k=1}^{\rho(W)ST} Z_k$ as there are $\rho(W)ST$ new arrivals during window W. As $\mathbf{E}(Z_k) = Q(\mathscr{C}_k) \leq Q^*$ by (4), by additivity of expectation, $\mathbf{E}(M(W)) \leq \rho(W)STQ^*$. In view of (4), if $\rho_{\max} \leq 1/5\log(ST)$, then $\mathbf{E}(M(W))$ is bounded above by $(ST)^{1/ST}/5\log(ST) = \mathscr{O}(1/\log(ST))$. As $M(W)$ is nonnegative and integer-valued, we have $\Pr\{M(W) \geq 1\} \leq \mathbf{E}(M(W))$. ∎

B. Bandwidth Considerations

The proof of Lemma 2 shows that if the attack factor is maintained at $\alpha_{\max}$, then the cumulative mean transmission bandwidth consumed by client REQs is $\mathscr{O}(\alpha_{\max}S)$ packets per second. The estimate is rather pessimistic, however, and we anticipate that the adaptive protocol does much better in periods of lulls in attack.

The key idea is provided by the Blocking Probability Lemma, which shows that the probability that a client is blocked decreases very rapidly at each successive generation. Indeed, for moderate attacks, any given client is likely to be accepted by the server well before the end of his retrial span J. Indeed, he is likely to form a connection within $J + \log(\rho_{\max})/2\log(2)$ generations. (Bear in mind that $\rho_{\max} < 1$ so that $\log \rho_{\max} < 0$.) This suggests that there is relatively little client traffic build up due to unconsummated connections near the end of the retrial span. It follows that the upper bound N^* on traffic given by (3) may be much too generous in periods where attack rates are low. We will show here that, indeed, the transmission bandwidth required by clients in the adaptive protocol is essentially of the same order as that commandeered in the omniscient protocol.

We formalize the intuition that client requests are typically accepted relatively quickly by considering successive windows forming *generational slices* of width

$$\lambda := \left\lfloor -\frac{1}{2}\log_2(\rho_{\max}) \right\rfloor$$

windows. For later reference, we observe that

$$\log_2 \rho_{\max}^{-1/2} - 1 < \lambda \leq \log_2 \rho_{\max}^{-1/2} \tag{5}$$

and, in particular, $2^{-\lambda} \geq \rho_{\max}^{1/2}$.

For any window W, we call the swath of

$$\sigma := \left\lceil \frac{1}{\lambda - 1}\log_2(\alpha_{\max}) \right\rceil$$

windows preceding it the *segment preceding* W and denote it $\mathscr{S}(W)$. It will suffice to suppose that $\rho_{\max} \leq 1/16 = 0.0625$ to ensure that $\lambda \geq 2$ and $\sigma < \infty$. Again, for later reference, we note that

$$\log_2 \alpha_{\max}^{1/(\lambda-1)} \leq \sigma < \log_2 \alpha_{\max}^{1/(\lambda-1)} + 1$$

and, in particular, that

$$2^{\sigma} < 2\alpha_{\max}^{1/(\lambda-1)} \leq 2\alpha_{\max} \qquad \text{if } \rho_{\max} \leq 0.0625. \tag{6}$$

Finally, we write $\bar{\alpha} = \bar{\alpha}(W)$ to denote the largest attack factor of any window in the segment $\mathscr{S}(W)$. (To obviate trivialities, we will, at need, replace $\bar{\alpha}$ by the larger of 1 and $\bar{\alpha}$.) We are now ready for the main result.

Theorem 8: Suppose $\alpha_{\max} \geq e^{1/15} = 1.06894\ldots$ and $\rho_{\max} \leq [60\log(\alpha_{\max})]^{-2}$. Then, the cumulative mean transmission bandwidth consumed by client REQs in window W under the adaptive client–server protocol is bounded above by

$$4\bar{\alpha}ST + \frac{84\log(2\alpha_{\max})^2}{\log(2)\alpha_{\max}}ST.$$

In the asymptotic regime where $S \to \infty$ and $\alpha_{\max}$ increases unboundedly with S, the upper bound becomes $4\bar{\alpha}ST + \mathfrak{o}(ST)$.

Corollary 1: Under the conditions of Theorem 8, the expected bandwidth consumed by clients in the adaptive client–server protocol is larger than the bandwidth consumed by the omniscient selective verification protocol only by a multiplicative factor of order not exceeding $\log(\alpha_{\max})/[-\log(\rho_{\max})]$.

Remarks:

1) If the attack factor in a given window is $\bar{\alpha}$, omniscient clients aware of both the current attack and request factors will occupy a bandwidth of $\bar{\alpha}S$ REQs. The adaptive protocol achieves essentially this order working from *tabula rasa* with no specific state knowledge.
2) It is easy to verify that $\rho_{\max} \leq 0.0625$ for the specified ranges of parameters, whence the width of the generational slices satisfies $\lambda \geq 2$ over the entire specified range.
3) We have opted for conservative bounds to keep the proof as uncluttered as possible. It will be apparent that the attack and client request factors may be expanded at the cost of increased algebraic tedium.
4) The adaptive scheme can *exponentially* improve bandwidth consumption over a nonadaptive approach that stays in a high protection mode at all times.

An elementary preliminary result smooths the way.

Lemma 4: Suppose m is a nonnegative integer and $0 \leq p \leq 1$. If $mp \leq 1$, then $(1-p)^m \geq 1 - mp$.

Proof: By the binomial theorem

$$(1-p)^m = 1 - mp + \binom{m}{2}p^2 - \binom{m}{3}p^3 + \cdots + (-1)^m\binom{m}{m}p^m. \tag{7}$$

The ratio of the absolute values of successive terms in the alternating sum is given by

$$\frac{\binom{m}{k}p^k}{\binom{m}{k-1}p^{k-1}} = \frac{(m-k+1)p}{k}.$$

Accordingly, $\binom{m}{k}p^k \le \binom{m}{k-1}p^{k-1}$ if, and only if, $(m+1)p \le k(1+p)$ or, equivalently, $k \ge (mp+p)/(1+p)$. As $mp \le 1$, it follows that $\binom{m}{k}p^k \le \binom{m}{k-1}p^{k-1}$ for $k \ge 1$, and consequently the terms in the alternating sum on the right of (7) decrease monotonically in absolute value. Truncating the alternating series at any point hence yields an error whose sign is that of the first neglected term. It follows *a fortiori* that $(1-p)^m \ge 1-mp$, the first neglected term $\binom{m}{2}p^2$ being positive.■

Proof of Theorem 8: We will suppress integer roundoff factors to keep the burgeoning notation compact. In the asymptotic domain, these factors become negligible.

1) Notation and Proof Structure: We may assume, without loss of generality, that the windows in the segment $\mathscr{S}(W)$ are numbered $W_0, W_1, \ldots, W_\sigma = W$. To estimate the accumulated traffic in window W, we may conservatively consider clients arriving in the segment windows W_j for $0 \le j \le \sigma$ together with those originating in windows W_{-j}, where $1 \le j \le J$ can range over at most the retrial span J.

While the nominal number of active generations in play for W is given conservatively by the retrial span $J = \log_2(\frac{\alpha_{\max}}{\rho_{\max}})$, we may anticipate that, starting from W_0, as one progresses down the segment toward $W_\sigma = W$, in any target window sufficiently far along in the segment, generations beyond

$$\bar{J} := \log_2\left(\frac{\bar{\alpha}}{\rho_{\max}}\right)$$

will have been absorbed with high probability. The key idea in the proof is a successive refinement of active generations in play: Starting from the nominal J generations at W_0, we recursively prune the number of unabsorbed generations seen by each window as we progress down the segment so that by the time we reach W_σ, the target window sees only $\overline{J}$ active generations.

It will be convenient to introduce notation for the geometrically decreasing sequence

$$\tau_k := \frac{1}{\lambda^k}\log_2\left(\frac{\alpha_{\max}}{\bar{\alpha}}\right) \qquad (k \ge 0).$$

Starting from W_0, as we progress along the segment, we identify the sequence of *marker windows*

$$W^{(k)} := W_{\tau_1+\cdots+\tau_k} \qquad (k \ge 0)$$

with separations decreasing exponentially along the segment. The nominal number of active generations seen by the initial marker window $W^{(0)} = W_0$ is given by

$$J_0 := J = \log_2\left(\frac{\alpha_{\max}}{\rho_{\max}}\right) = \log_2\left(\frac{\alpha_{\max}}{\bar{\alpha}}\right) + \log_2\left(\frac{\bar{\alpha}}{\rho_{\max}}\right) = \tau_0 + \bar{J}$$

so that τ_0 denotes the generational "excess" that we seek to prune. We begin by showing that the τ_0 ancestral generations, $J_0 - \tau_0 < j \le J_0$, of $W^{(0)}$ contribute a negligible amount of traffic to the next marker window $W^{(1)}$, and thence to all subsequent marker windows. The number of effectively active generations seen by the marker window $W^{(1)}$ is hence reduced from the nominal value $J_0 = J = \tau_0 + \bar{J}$ to the value

$$J_1 := (J_0 - \tau_0) + \tau_1 = \tau_1 + \bar{J}.$$

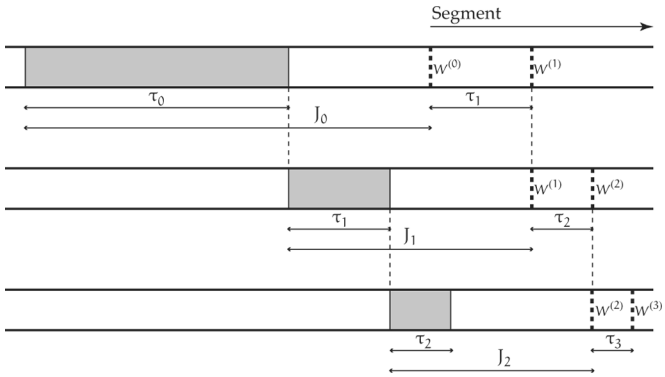

Fig. 1. Pictorial view of the sequential culling of generations. The τ_k earliest generations in the J_k active generations of the marker window $W^{(k)}$ are culled, leaving the marker window $W^{(k+1)}$, which is τ_{k+1} windows to the right of $W^{(k)}$, with a smaller number J_{k+1} of active generations.

The induction step consists in showing that given

$$J_k := \tau_k + \bar{J}$$

active generations for marker window $W^{(k)}$, the τ_k ancestral generations, $J_k - \tau_k < j \le J_k$, of $W^{(k)}$ contribute a negligible amount of traffic to the next marker window $W^{(k+1)}$, and thence to all subsequent marker windows. The marker window $W^{(k+1)}$ then effectively sees only

$$J_{k+1} = (J_k - \tau_k) + \tau_{k+1} = \tau_{k+1} + \bar{J}$$

active generations, and the excess generations are culled exponentially fast. The process is sketched in Fig. 1.

Let n be the unique integer for which $\tau_n \ge 1$ and $\tau_{n+1} < 1$, that is to say, n is the largest integer for which $\tau_n \ge 1$. It follows that

$$\frac{\log\log_2\left(\frac{\alpha_{\max}}{\bar{\alpha}}\right)}{\log\lambda} - 1 < n \le \frac{\log\log_2\left(\frac{\alpha_{\max}}{\bar{\alpha}}\right)}{\log\lambda} \tag{8}$$

in consequence of which, asymptotically in $\alpha_{\max}$

$$n = \mathscr{O}\left(\frac{\log\log\alpha_{\max}}{\log\log\log\alpha_{\max}}\right)$$

as $\lambda = \lfloor \log_2 \rho_{\max}^{-1/2} \rfloor = \Omega(\log\log\alpha_{\max})$ under the conditions of the theorem. Thus, after at most n reductive stages, the number of active generations seen by the marker window $W^{(n+1)} = W_{\tau_1+\cdots+\tau_{n+1}}$ is no more than $\bar{J}$ with all preceding generations absorbed with high probability. (Edge effects are accommodated in the final stage by adjusting the value of τ_{n+1} as necessary.) As

$$\begin{aligned}\tau_1 + \cdots + \tau_{n+1} &= \frac{1}{\lambda}\left(\frac{1-\lambda^{-n-1}}{1-\lambda^{-1}}\right)\log_2\left(\frac{\alpha_{\max}}{\bar{\alpha}}\right) \\ &\le \left\lceil \frac{1}{\lambda-1}\log_2\left(\frac{\alpha_{\max}}{\bar{\alpha}}\right)\right\rceil = \sigma\end{aligned}$$

all subsequent windows W_j with $\tau_1 + \cdots + \tau_{n+1} \le j \le \sigma$ in the segment will each see an effective number of active generations $\bar{J}$.

2) Induction Base: The analysis for the first stage of reduction provides a model for the general recurrence step. We begin

Fig. 2. To move from the first marker window $W^{(0)}$ to the next marker window $W^{(1)}$, we traverse the first τ_1 windows in the segment $\mathcal{S}$ sequentially. We proceed recursively: For $0 \le i \le \tau_1 - 1$, we successively estimate the traffic in the ith window that originates in the generational slice $\mathcal{G}_i$ of the marker window $W^{(0)}$.

accordingly by a consideration of the $J_0 = J$ generations of the marker window $W^{(0)} = W_0$. Let $Z^{(1)}$ denote the amount of traffic in the marker window $W^{(1)} = W_{\tau_1}$ originating in the τ_0 ancestral generations, $J_0 - \tau_0 < j \le J_0$, of $W^{(0)}$.

To estimate $\mathbf{E}(Z^{(1)})$, we begin by partitioning windows in the retrial span of the initial marker window $W^{(0)} = W_0$ into generational slices $\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_{\tau_1 - 1}$ where, for $0 \le i \le \tau_1 - 1$

$$\mathcal{G}_i := \{\, W_{-j} : J_0 - (i+1)\lambda + 1 \le j \le J_0 - i\lambda \,\}$$

identifies the indices of the windows W_{-j} of clients whose generations range from $J_0 - (i+1)(\lambda - 1)$ to $J_0 - i(\lambda - 1)$ with respect to W_i. The relation of the generational slices to the windows in the segment is sketched in Fig. 2.

Begin by considering the generational slice

$$\mathcal{G}_0 := \{\, W_{-j} : J_0 - \lambda < j \le J_0 \,\}$$

of the furthest λ generations with respect to window W_0. These are the windows comprising the generations at the end of the retrial span of the protocol for W_0. Let the random variable X_0 denote the number of REQ packets sent in W_0 from clients arriving in the generational slice $\mathcal{G}_0$. In view of the Blocking Probability Lemma, we may bound the expected value of X_0 by

$$\begin{aligned}
\mathbf{E}(X_0) &\le \sum_{g=J_0-\lambda+1}^{J_0} 2^g \rho_{\max} ST \exp\left(\frac{-2^g ST}{N^*}\right) \\
&\le \lambda 2^{J_0} \rho_{\max} ST \exp\left(\frac{-2^{J_0-\lambda}}{5\alpha_{\max}}\right) \\
&\le 2\lambda \alpha_{\max} ST \exp\left(\frac{-2^{-\lambda}}{5\rho_{\max}}\right) \\
&\le \frac{2\alpha_{\max} ST}{\log(2)} \log\left(\rho_{\max}^{-1/2}\right) \exp\left(-\rho_{\max}^{-1/2}\Big/ 5\right)
\end{aligned}$$

as $\alpha_{\max}/\rho_{\max} \le 2^{J_0} < 2\alpha_{\max}/\rho_{\max}$ and $\lambda \le \log(\rho_{\max}^{-1/2})/\log(2)$. Identifying $x = \rho_{\max}^{-1/2}$, we may write the bound more compactly in the form

$$\mathbf{E}(X_0) \le \frac{2\alpha_{\max} ST}{\log(2)} \log(x) e^{-x/5}.$$

Differentiation shows that the function $f(x) := \log(x)e^{-x/5}$ decreases monotonically with x for $x \ge 3.7686\ldots$ or, equivalently, $\rho_{\max} \le 0.0704\ldots$. It follows that the bound on the right decreases monotonically as $\rho_{\max}$ decreases over the entire range of values for $\rho_{\max}$ specified in the theorem (see Remark 2 following the statement of the theorem).

We constrain x so that $f(x) \le \alpha_{\max}^{-6}$ or, equivalently, $x \ge 30\log\alpha_{\max} + 5\log\log x$. Simple estimates suffice here: Tthese bounds are automatically achieved if we select $\alpha_{\max} \ge e^{4/60} = e^{1/15}$ and $x \ge 60\log\alpha_{\max}$ as the interval $[60\log\alpha_{\max}, \infty)$ is contained in the interval $[4, \infty)$ and is hence in the monotonic range of f and simple computations serve to verify that

$$60\log\alpha_{\max} \ge 30\log\alpha_{\max} + 5\log\log(60\log\alpha_{\max})$$

for $\alpha_{\max} \ge e^{1/15}$. (As $\rho_{\max} = x^{-2}$, these conditions are equivalent to the stated condition $\rho_{\max} \le (60\log\alpha_{\max})^{-2} \le 1/16$.) It follows then that, under the conditions of the theorem, we have

$$\mathbf{E}(X_0) \le \frac{2ST}{\log(2)\alpha_{\max}^5}.$$

Markov's inequality now allows us to limit the excursions of X_0 quite sharply. With $\sigma = \frac{1}{\lambda - 1}\log_2(\alpha_{\max})$ denoting the segment swath width as before

$$\Pr\{X_0 \ge 2^{-\sigma} ST\} \le \frac{\mathbf{E}(X_0)}{2^{-\sigma} ST} = \frac{2}{\log(2)} \cdot \frac{2^\sigma}{\alpha_{\max}^5} \le \frac{4}{\log(2)\alpha_{\max}^4}$$

in view of the upper bound (6) on 2^σ. The probability of a large excursion for X_0 decays to zero satisfactorily fast.

For $0 \le i \le \tau_1 - 1$, let X_i now denote the number of REQ packets sent in W_i by clients who arrive in the generational slice $\mathcal{G}_i$. Let A_i denote the "bad" event that $X_i > 2^{-\sigma} ST$. We work inductively.

Conditioned on the event that none of the bad events A_0, $A_1, \ldots, A_{i-1}$ have occurred, the total client traffic in W_i originating in the generational slices $\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_{i-1}$ is bounded by

$$\sum_{j=0}^{i-1} 2^{i-j} X_j \le 2^{-\sigma} ST \sum_{j=0}^{i-1} 2^{i-j} \le 2^{-\sigma+i+1} ST \le ST. \quad (9)$$

On the other hand, the client traffic in W_i that originates from clients in generations $0 \le g < J_0 - i(\lambda - 1)$ is, by the usual geometric summation, no more than $2^{J_0 - i(\lambda-1)}\rho_{\max} ST$. As W_i is part of the segment, the attack factor in the window is no larger than $\bar{\alpha}$. Consequently, conditioned on none of the bad events A_0, $A_1, \ldots, A_{i-1}$ occurring, the total traffic in window W_i may be bounded by

$$N(W_i) \le \bar{\alpha} ST + ST + 2^{J_0 - i(\lambda-1)} \rho_{\max} ST. \quad (10)$$

However, as $i \le \tau_1 - 1$, we have

$$J_0 - i(\lambda - 1) \ge J_0 - \lambda\tau_1 = J_0 - \tau_0 = \bar{J} = \log_2\left(\frac{\bar{\alpha}}{\rho_{\max}}\right)$$

whence we observe that

$$2^{J_0 - i(\lambda-1)} \rho_{\max} \ge \frac{\bar{\alpha}}{\rho_{\max}} \rho_{\max} = \bar{\alpha}.$$

Consequently, the conditional traffic seen in window W_i is bounded by

$$N(W_i) \le 3\rho_{\max} ST 2^{J_0 - i(\lambda-1)} \quad (11)$$

where, if necessary, we replace $\bar{\alpha}$ by the larger of 1 and $\bar{\alpha}$ to eliminate trivial cases from consideration. As window W_i is situated i windows to the right of W_0, with respect to W_i the generation number of the left edge of the generational slice $\mathscr{G}_i$ is $J_0 - i\lambda + i = J_0 - i(\lambda - 1)$, and the right edge of the generational slice $\mathscr{G}_i$ with respect to W_i is $J_0 - i(\lambda - 1) - \lambda + 1 = J_0 - (i+1)(\lambda - 1)$. The Blocking Probability Lemma tells us then that the conditional expected value of X_i given that none of the bad events $A_0, A_1, \ldots, A_{i-1}$ occur may be bounded by

$$\begin{aligned}
&\mathbf{E}\left(X_i \mid A_0^{\mathrm{c}}, A_1^{\mathrm{c}}, \ldots, A_{i-1}^{\mathrm{c}}\right) \\
&\quad \le \sum_{g=J_0-(i+1)(\lambda-1)}^{J_0-i(\lambda-1)} 2^g \rho_{\max} ST \exp\left(\frac{-2^{g-J_0+i(\lambda-1)}}{3\rho_{\max}}\right) \\
&\quad \le \lambda 2^{J_0-i(\lambda-1)} \rho_{\max} ST \exp\left(\frac{-2^{-\lambda+1}}{3\rho_{\max}}\right) \\
&\quad \le \frac{2\alpha_{\max} ST}{\log(2)} \left\{\log\left(\rho_{\max}^{-1/2}\right) \exp\left(-2\rho_{\max}^{-1/2}\Big/3\right)\right\}.
\end{aligned}$$

However, the quantity in curly brackets on the right is certainly no larger than $\log(\rho_{\max}^{-1/2}) \exp(-\rho_{\max}^{-1/2}/5)$, which, in our estimate of $\mathbf{E}(X_0)$, we have already seen is bounded above by $\alpha_{\max}^{-6}$ under the given conditions on $\alpha_{\max}$ and $\rho_{\max}$. It follows that

$$\mathbf{E}\left(X_i \mid A_0^{\mathrm{c}}, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1}\right) \le \frac{2ST}{\log(2)\alpha_{\max}^5}.$$

Markov's inequality for conditional probabilities hence yields

$$\begin{aligned}
&\Pr\left(A_i \mid A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1}\right) \\
&\quad = \Pr\left\{X_i \ge 2^{-\sigma} ST \mid A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1}\right\} \\
&\quad \le \frac{\mathbf{E}(X_i \mid A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1})}{2^{-\sigma} ST} = \frac{2}{\log(2)} \cdot \frac{2^{\sigma}}{\alpha_{\max}^5} \\
&\quad \le \frac{4}{\log(2)\alpha_{\max}^4}.
\end{aligned}$$

On the other hand, the probability that any of the bad events $A_0, A_1, \ldots, A_{i-1}$ occur is given by

$$\begin{aligned}
\Pr(A_0 \cup A_1 \cup \cdots \cup A_{i-1}) &= 1 - \Pr\left(A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1}\right) \\
&= 1 - \prod_{j=0}^{i-1} \Pr\left(A^{\mathrm{c}}{}_j \mid A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{j-1}\right) \\
&= 1 - \prod_{j=0}^{i-1} \left[1 - \Pr\left(A_j \mid A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{j-1}\right)\right].
\end{aligned}$$

Simple bounds suffice to bound the right-hand side, and we may bound $\Pr(A_0 \cup A_1 \cup \cdots \cup A_{i-1})$ from above by the string of inequalities

$$\begin{aligned}
1 - \left(1 - \frac{4}{\log(2)\alpha_{\max}^4}\right)^i &\le 1 - \left(1 - \frac{4}{\log(2)\alpha_{\max}^4}\right)^{\sigma} \\
&\le 1 - \left(1 - \frac{4}{\log(2)\alpha_{\max}^4}\right)^{\lceil \log_2 \alpha_{\max} \rceil}
\end{aligned} \tag{12}$$

as $i \le \sigma = \lceil \frac{1}{\lambda - 1} \log_2 \alpha_{\max} \rceil \le \lceil \log_2 \alpha_{\max} \rceil$, the final step following because $\lambda \ge 2$ for the given range of $\rho_{\max}$. We now claim that the bound

$$1 - \left(1 - \frac{4}{\log(2)\alpha_{\max}^4}\right)^{\lceil \log_2 \alpha_{\max} \rceil} \le \frac{4\lceil \log_2 \alpha_{\max} \rceil}{\log(2)\alpha_{\max}^4} \tag{13}$$

holds for all $\alpha_{\max} \ge e^{1/15}$. Indeed, if $e^{1/15} \le \alpha_{\max} \le 2$, then $\lceil \log_2 \alpha_{\max} \rceil = 1$ and

$$1 - \left(1 - \frac{4}{\log(2)\alpha_{\max}^4}\right)^1 = \frac{4}{\log(2)\alpha_{\max}^4} = \frac{4\lceil \log_2 \alpha_{\max} \rceil}{\log(2)\alpha_{\max}^4}$$

so that equality in the claimed bound holds for trivial reasons. If $2 < \alpha_{\max} \le 4$, then $\lceil \log_2 \alpha_{\max} \rceil = 2$ and

$$\begin{aligned}
1 - \left(1 - \frac{4}{\log(2)\alpha_{\max}^4}\right)^2 &= \frac{8}{\log(2)\alpha_{\max}^4} - \frac{16}{\log(2)^2 \alpha_{\max}^8} \\
&\le \frac{8}{\log(2)\alpha_{\max}^4} = \frac{4\lceil \log_2 \alpha_{\max} \rceil}{\log(2)\alpha_{\max}^4}
\end{aligned}$$

and the bound is verified in this range of $\alpha_{\max}$ as well. Finally, if $\alpha_{\max} > 4$, temporarily set $m = \lceil \log_2 \alpha_{\max} \rceil$, $p = 4/[\log(2)\alpha_{\max}^4]$, and $\zeta(\alpha_{\max}) = [4\log(2\alpha_{\max})]/[\log(2)^2 \alpha_{\max}^4]$. Simple bounds suffice again: We have

$$m < \frac{\log(\alpha_{\max})}{\log(2)} + 1 = \frac{\log(2\alpha_{\max})}{\log(2)}$$

and it follows that $mp \le \zeta(\alpha_{\max})$. Differentiation shows that $\zeta(\alpha_{\max})$ decreases monotonically for $\alpha_{\max} \ge \frac{1}{2} e^{1/4} = 0.642013\ldots$ and *a fortiori* also for the range $\alpha_{\max} \ge 1.06894\ldots$ specified in the theorem. Numerical computation now shows that $\zeta(\alpha_{\max}) = 1$ for $\alpha_{\max} = 1.80883\ldots$, and so for $\alpha_{\max} > 4$, we have $mp \le \zeta(\alpha_{\max}) \le 1$. Lemma 4 is hence in force, and the claimed result is proved for $\alpha_{\max} > 4$, and hence for all $\alpha_{\max} \ge e^{1/15}$.

In view of (12) and (13), we obtain the compact bound

$$\Pr(A_0 \cup A_1 \cup \cdots \cup A_{i-1}) \le \frac{4\lceil \log_2 \alpha_{\max} \rceil}{\log(2)\alpha_{\max}^4}.$$

For future reference, let $\mathfrak{A}^{(1)}$ be the event that one or more of the inequalities $X_i \le 2^{-\sigma} ST$ is violated for the windows preceding the marker window $W^{(1)}$ up to (and including) the previous marker window $W^{(0)}$. Thus, $\mathfrak{A}^{(1)} = A_0 \cup A_1 \cup \cdots \cup A_{\tau_1 - 1}$. The superscript in $\mathfrak{A}^{(1)}$ denotes that this event is associated with the inductive step for marker window $W^{(1)}$. As the previous inequality holds for each i, specializing in particular to the case $i = \tau_1$ shows that

$$\Pr\left(\mathfrak{A}^{(1)}\right) \le \frac{4\lceil \log_2 \alpha_{\max} \rceil}{\log(2)\alpha_{\max}^4}$$

so that the probability of occurrence of one or more bad events A_j is $\bar{\mathcal{O}}(\log(\alpha_{\max})/\alpha_{\max}^4)$ and vanishes asymptotically.

To return to the analysis, if any of the bad events A_0, $A_1, \ldots, A_{i-1}$ occurs, we can hence afford to be cavalier and bound the client traffic in window W_i via the Traffic Bound Lemma. Thus, by conditioning on the two cases, the expected

client traffic in window W_i that originates in generational slice $\mathscr{G}_i$ may be bounded by

$$\begin{aligned}\mathbf{E}(X_i) &\le \mathbf{E}\left(X_i \mid A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1}\right) \\ &\quad + 5\alpha_{\max} ST\left[1 - \Pr\left(A^{\mathrm{c}}{}_0, A^{\mathrm{c}}{}_1, \ldots, A^{\mathrm{c}}{}_{i-1}\right)\right] \\ &= \frac{2ST}{\log(2)\alpha_{\max}^5} + \frac{20\lceil \log_2 \alpha_{\max}\rceil ST}{\log(2)\alpha_{\max}^3} \\ &\le \frac{21\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^3}\end{aligned}$$

for each $0 \le i \le \tau_1 - 1$ and $\alpha_{\max} \ge e^{1/15}$.

Thus, the contribution of each of the generational slices $\mathscr{G}_i$ to client traffic is essentially extinguished by window W_{τ_1}. Indeed, the total expected contribution to traffic in the marker window $W^{(1)} = W_{\tau_1}$ from clients in the generational slices $\mathscr{G}_0, \ldots, \mathscr{G}_{\tau_1 - 1}$, that is to say, the traffic originating in the τ_0 ancestral generations, $J_0 - \tau_0 < g \le J_0$, of the previous marker window $W^{(0)}$, may now be bounded by

$$\begin{aligned}\mathbf{E}\left(Z^{(1)}\right) &\le \sum_{i=0}^{\tau_1 - 1} 2^{\tau_1 - i}\mathbf{E}(X_i) \le 2^{\tau_1}\frac{21\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^3} \\ &\le 2^{\sigma}\frac{21\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^3} \le \frac{42\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^2}.\end{aligned}$$

The τ_0 ancestral generations of the previous marker window $W^{(0)}$ hence contribute asymptotically small amounts of traffic to $W^{(1)}$. The effective number of active generations for the marker window $W^{(1)}$ has now been reduced from the nominal number $J = J_0 = \tau_0 + \bar{J}$ corresponding to the retrial span of the protocol to

$$J_1 = (J_0 - \tau_0) + \tau_1 = \tau_1 + \bar{J}.$$

We may now begin anew with the marker window $W^{(1)}$ as the new origin and consider the contribution to the traffic in the next marker window $W^{(2)}$ that originates in the ancestral τ_1 generations, $J_1 - \tau_1 < g \le J_1$, of the marker window $W^{(1)}$. The stage is set for an induction.

3) Induction Step: The analysis for the base case may now be systematically reproduced, stage by stage, with minor modifications: At the kth stage, the kth marker window $W^{(k)}$ serves as the new origin and effectively sees only $J_k = \tau_k + \bar{J}$ active generations. Let $Z^{(k+1)}$ denote the contribution to the traffic in the marker window $W^{(k+1)}$ that originates in the τ_k ancestral generations, $J_k - \tau_k < g \le J_k$, of the previous marker window $W^{(k)}$.

As before, the τ_k ancestral generations, $J_k - \tau_k < g \le J_k$, of $W^{(k)}$ are partitioned into τ_{k+1} generational slices, each of width λ. Reusing notation, X_i now denotes the traffic due to the ith generational slice $\mathscr{G}_i$ in the ith window following $W^{(k)}$; as before, A_i is the "bad" event that $X_i > 2^{-\sigma}ST$; and $\mathfrak{A}^{(k)}$ is the event that one or more of the inequalities $X_i \le 2^{-\sigma}ST$ is violated for the windows preceding the marker window $W^{(k+1)}$ up to (and including) the previous marker window $W^{(k)}$. As induction hypothesis we suppose that, for each $1 \le j \le k$

$$\begin{aligned}\Pr\left(\mathfrak{A}^{(j)}\right) &\le \frac{4\lceil \log_2 \alpha_{\max}\rceil}{\log(2)\alpha_{\max}^4} \\ \mathbf{E}\left(Z^{(j)}\right) &\le \frac{42\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^2}.\end{aligned}$$

For the estimate for the total traffic in window $W_{\tau_1+\cdots+\tau_k+i}$ corresponding to (10), we now also have to take into account the contribution due to the earlier ancestral slices carried forward from the earlier marker windows. By conditioning now on the event that none of $\mathfrak{A}^{(1)}, \ldots, \mathfrak{A}^{(k)}$, A_0, $A_1, \ldots, A_{i-1}$ occurs, we guarantee that none of the ancestral generational slices contributes an excessive amount. With this conditioning then, corresponding to the estimate (9) the total client traffic in $W_{\tau_1+\cdots+\tau_k+i}$ originating from all previous ancestral slices is bounded by

$$2^{-\sigma+\tau_1+\cdots+\tau_k+i}ST \le ST$$

(as $\sum_j \tau_j \le \sigma$) the upper bound unchanged from (9). The expression corresponding to the conditional traffic bound (11) hence becomes

$$N(W_{\tau_1+\cdots+\tau_k+i}) \le 3\rho_{\max}ST2^{J_k - i(\lambda-1)}.$$

With the additional conditioning on the joint occurrence of $\mathfrak{A}^{(1)^{\mathrm{c}}}, \ldots, \mathfrak{A}^{(k)^{\mathrm{c}}}$, the analysis for the base case now goes through with little more than the replacement of J_0 and J_1 by J_k and J_{k+1}, respectively, and τ_0 and τ_1 by τ_k and τ_{k+1}, respectively, to yield

$$\begin{aligned}\Pr\left(\mathfrak{A}^{(k+1)}\right) &\le \frac{4\lceil \log_2 \alpha_{\max}\rceil}{\log(2)\alpha_{\max}^4} \\ \mathbf{E}\left(Z^{(k+1)}\right) &\le \frac{42\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^2}.\end{aligned}$$

This completes the induction.

4) Steady-State Bandwidth Consumption in the Segment: The expected contribution to the marker window $W^{(n+1)}$ due to the cumulative contributions from the ancestral τ_k generations, $J_k - \tau_k < g \le J_k$, of $W^{(k)}$ for each $1 \le k \le n$ exhausts all generations beyond $\bar{J}$ of $W^{(n+1)}$. The total expected contribution from these ancestral generations may be bounded by

$$\begin{aligned}\sum_{k=1}^{n}\mathbf{E}\left(Z^{(k)}\right)2^{\tau_{k+1}+\cdots+\tau_{n+1}} &\le n2^{\sigma}\frac{42\log_2(2\alpha_{\max})ST}{\log(2)\alpha_{\max}^2} \\ &\le \frac{84\log_2(2\alpha_{\max})^2 ST}{\log(2)\alpha_{\max}}\end{aligned}$$

where we can afford to crudely bound the upper estimate for n in (8) by $n \le \log_2(2\alpha_{\max})$. Thus, the entire expected contribution to the traffic in $W^{(n+1)} = W_{\tau_1+\cdots+\tau_{n+1}}$ due to generations beyond $\bar{J}$ is $\mathfrak{o}(ST)$ asymptotically. It only remains to estimate the traffic from the remaining $\bar{J}$ generations. However, this, by geometric summation, is no larger than $2^{\bar{J}+1}\rho_{\max}ST \le 4\bar{\alpha}ST$ (the extra factor of 2 in the upper bound reintroduced here to take into account integer roundoff in the exponent).

By shifting all windows right one step at a time, it is clear that the analysis holds for all windows W_j with $\tau_1 + \cdots + \tau_{n+1} \le j \le \sigma$ in the segment. Moreover, *a fortiori*, the expected client traffic in the target window $W = W_\sigma$ is bounded above by $4\bar{\alpha}ST + \mathfrak{o}(ST)$ as claimed. ■

VI. Extensions

We now turn to mechanisms by which the basic adaptive protocol may be extended to cover: 1) the possibility of an unre-

liable server and network; and 2) a more flexible regulation of bandwidth by server and client.

If a server being protected by ASV goes down, the basic ASV protocol would escalate a flood of requests from the clients, which would only aggravate the situation. A simple remedy is for the server to provide a special type of ACK, Drop ACKs (DACKs) at step S3 in Section V, for every request it receives but is not able to process. DACKs serve as an encouragement mechanism that communicates a "please retry more aggressively" message to the clients. A client in a given round (operational timeout window) establishes connection upon receipt of an ACK; else, if he receives any DACK in T time units, he moves on to the next round; failing these two possibilities, he quits. Smurf-type attacks [1] can be inhibited by using nonces (as a weak authenticator) in REQs and DACKs.

A slight modification of the client protocol can also handle lossy networks in which REQs or (D)ACKs are dropped. If no DACK is received for K consecutive packets sent by the client, he quits. This check is only performed at the beginning of each round. Therefore, if the path from a client to the server experiences a drop rate of d, this modification reduces the probability of a client incorrectly quitting to the order of d^K.

Some network situations may also call for a more flexible allocation of bandwidth under ASV. For instance, if a number of colocated services share bandwidth, it may not be economical for one service to consume too much bandwidth just to prevent DoS attacks. In another instance, a server may wish to reduce ASV bandwidth consumption in favor of having more bandwidth available for a period of heavy bulk transfers. Concerns of this type may be addressed by having the server inform the clients *not* to send too much traffic by providing clients with a new retrial span J when it is desired that the clients reduce bandwidth consumption. In the general setting, the server computes a maximum bandwidth consumption and client request success probability pair $(B(r), P(r))$ for each retrial span r and, for a given parameter vector of *system priorities* S, selects a retrial span to maximize a utility function, $J = \arg\max_r U(B(r), P(r), S)$. On the client side, on receipt of a (maximum) retrial span J from the server, clients can use a client cost–benefit analysis inspired by [22] to select a retrial span within the permitted range by optimizing a *client valuation function* V based on the nondecreasing success function $P(r)$ (which the client can obtain from the server through DACKs).

VII. Experimental Evaluation

The simulations described in this section test the full adaptive protocol in settings reflecting real world situations. We: 1) verify our analytical predictions by evaluating the effectiveness of the adaptive scheme vis à vis nonadaptive counterparts; 2) study ASV's behavior in the presence of network congestion; and 3) consider its effect on TCP-based cross traffic.

Simulation Setup: The simulations were performed using the *NS-2* network simulator for the topology shown in Fig. 3. The topology is dynamic with clients arriving at a fixed rate of 50 per second, each arriving client needing to get one REQ served. Varying attack rates are simulated by having the number of attackers vary between 1 and 100, each attacker constantly issuing 400 REQs/s. The attack rate hence ranges between 400 and 40 000 REQs/s. We set the number of requests that the server can process in a second to $S = 600$ so that, in our notation, $\rho_{\max} = \rho_{\min} = 0.08$ and $\alpha_{\max} = 66$. For the topology, we fix RTT at 60 ms and T to 0.4 s. All communications are over UDP unless otherwise noted, REQs are 200 B, DACKs and ACKs are 50 and 200 B, respectively. The capacity of the bottleneck link is overprovisioned to 100 Mb/s to avoid any network congestion in all the experiments except those where we evaluate the performance of ASV in a lossy network. Arrival times of clients and attackers and interpacket intervals for attackers are randomized to avoid creating undesirable deterministic patterns.

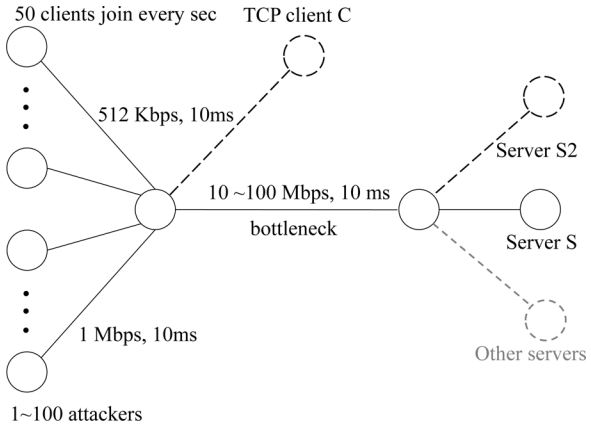

Fig. 3. Simulation topology.

Comparing Adaptive and Nonadaptive: For the solid line topology in Fig. 3, clients send one or more REQs every T seconds until either a connection is established with the receipt of an ACK (success) or timeout expires after JT seconds (failure). In this setting, we compare ASV with the following two static client behaviors.

- *Naive*: Send one REQ every T seconds.
- *Aggressive*: Send 2^J REQs every T seconds.

For these experiments, we chose $J = 7$. Each experiment is performed with one type of client, varying attack rates from 1 to 100 attackers, and a fixed average attack period of 30 s, which proves to be sufficiently long for the system to stabilize. Comparative results for the three distinct client behaviors are shown in Figs. 4–6.

As predicted analytically, the experimental results provide clear evidence for the effectiveness of the adaptation strategy in raising the costs (bandwidth) in accordance with the attack rate. ASV provides clear benefits in success ratios and client bandwidth consumption over the high protection *Aggressive* strategy with concomitant service latencies of at most 2.3 s for the fiercest attacks. Given ASV's marked benefits in success ratios and bandwidth consumption, these worst-case latencies for severe attacks may be considered acceptable. At the other extreme, *Naive* clients, as expected, suffer serious failure rates, again underscoring the effectiveness of ASV. The results also quantify the overhead of ASV—a factor of 16 in terms of bandwidth and 1.5 in terms of service latency in the worst attack scenarios.

Variable Rate Attacks: In each of the previous experiments, the attack rate was fixed during the simulation. We now explore the effect of varying attack rates on clients implementing ASV.

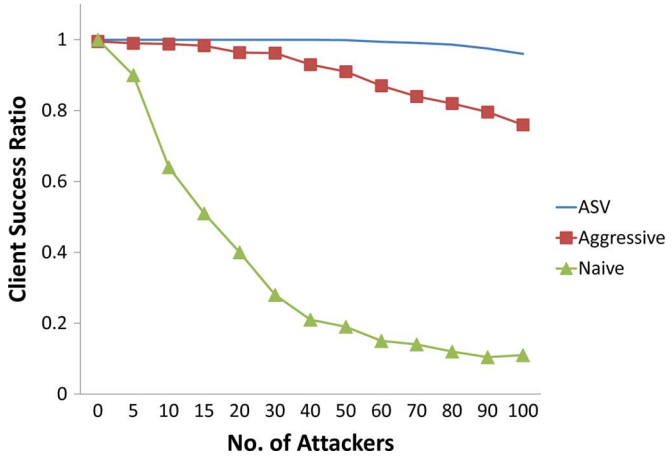

Fig. 4. Ratio of the successful clients to all clients (1500 in 30 s) versus attack rate.

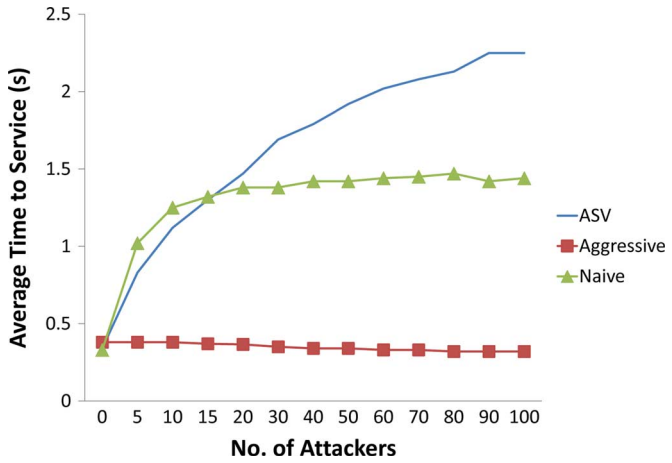

Fig. 5. Average time to service (for clients that succeed in getting service) versus attack rate.

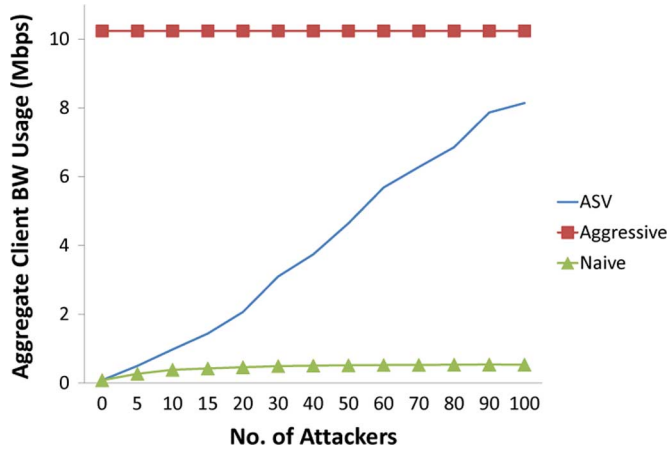

Fig. 6. Aggregate bandwidth consumption for all the clients versus attack rate.

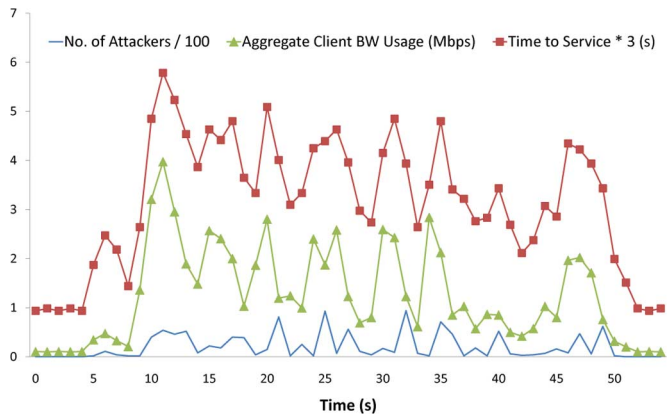

Fig. 7. Effect of 45 s of variable rate attacks on success ratio and aggregate client bandwidth consumption. Success ratio for clients is always 1. Clients that join the system between times i and $i+1$ are represented in front of time i.

In the first set of experiments, we subject the system to pulse attacks where we observe the system's response to a 5-s no-attack period, followed by 10 s of a heavy (but fixed rate) attack, followed by another 10 s with no attack. We performed this experiment for 25, 50, and 100 attackers. While we omit the detailed results here in view of space considerations, the most important observed outcome was that in all three scenarios, in less than 2 s the ASV implementation fully adapts itself to attack conditions with the success ratio, time to service, and bandwidth consumption numbers converging to the corresponding values in Figs. 4–6. In addition, after the attack stops, the system relaxes to its pre-attack conditions in less than 2 s.

To better understand the effect of highly variable rate attacks, we also simulated 45 s of variable rate attacks, preceded and followed by 5-s periods with no attack. During each attack period, the number of attackers was randomly set at each second to $\lceil \exp(A) \rceil$, where A is a floating point number chosen at random from $[0, \ln(100))$. The results depicted in Fig. 7 show how quickly the system adapts and then recovers to the pre-attack pattern in the presence of pulse attacks. These experiments show how ASV preserves success ratio, time to service, and bandwidth consumption in consonance with our analytically predicted bounds, even in the face of highly variable rate attacks. This significantly reduces the attackers' ability to disrupt the operation (and bandwidth consumption) of multiple ASV protected servers at the same time by attacking them in rotation.

Lossy Network: So far, we have assumed links are overprovisioned and that there is thus no packet loss in the network. In order to assess the effect of a lossy network, with 50 attackers present, we made the bottleneck link drop packets at different rates and modified the ASV protocol using DACKs as outlined in Section VI for $K = 3$ and $K = 7$. To summarize our findings, in both cases, there is almost no quitting, and client bandwidth consumption stays approximately fixed for drop rates of up to 30%. However, for network drop rates of 40%–80%, the quit ratio ranges from 0.08 to 0.71 for $K = 3$, and from 0.01 to 0.32 for $K = 7$. The corresponding client bandwidth consumption ranges from 4.26 to 1.04 Mb/s and 4.62 to 4.08 Mb/s, respectively.

Even though enforcing a cap on the maximum number of outstanding REQs (with no DACK) is not meant to be a full-fledged congestion control mechanism, it would still be desirable for ASV clients to react to serious network congestion by backing off. Additional simulations that we do not elaborate on here in view of space considerations provide evidence that if (for any

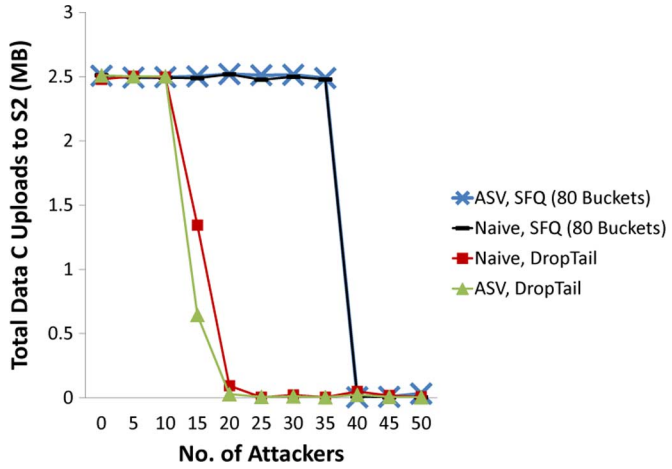

Fig. 8. Amount of data that client $\mathscr{C}$ can upload to sever $\mathscr{S}2$ in 30 s. The lines that are close to the horizontal axis represent values in the 7.5–15 kB range. Queueing disciplines in the bottleneck link are DropTail and SFQ with 80 buckets. The clients use ASV and *Naive* to connect to server $\mathscr{S}$.

reason) clients face heavy congestion in the network, they eventually react and stop aggravating the situation.

Effect on TCP Cross Traffic: To measure the effect of ASV on cross traffic, we set up the following simulation scenario. We created a client $\mathscr{C}$ communicating with a data-backup server $\mathscr{S}2$, colocated with an ASV-protected server $\mathscr{S}$ behind the bottleneck link (capacity 100 Mb/s), as illustrated in the shaded lines in Fig. 3. Client $\mathscr{C}$ is backing up data on $\mathscr{S}2$ at the rate of 512 kb/s over TCP. In parallel, we simulated DDoS attacks on $\mathscr{S}$ with a clientele of 50 clients per second (clients protocols are either *Naive* or ASV with $K = 3$). As before, *Naive* behavior represents a no-defense base for comparative purposes. The attack rates and the queueing disciplines used in the bottleneck link vary in different scenarios. The amount of data that $\mathscr{C}$ can upload to $\mathscr{S}2$ in 30 s in each scenario is plotted in Fig. 8. The figure shows that when TCP cross traffic shares a bottleneck link with non-congestion-controlled traffic from attackers, it could be seriously throttled. It confirms that unless the network links around a UDP-based service are highly overprovisioned and protected against network link attacks, TCP cross traffic would be seriously harmed in the face of fierce attacks. In addition, we observe that Stochastic Fair Queueing (SFQ) provides better guarantees compared to DropTail until client $\mathscr{C}$'s traffic is hashed into the same bucket as attackers' packets. This results in $\mathscr{C}$'s traffic being dropped, which in turn causes it to backoff. Our main conclusion is that the attack traffic is the major cause of the throttling of the TCP client and not the particular client protocol in place: Compared to *Naive* (which represents a no-defense attack-only scenario), SASV does not induce any significant *extra* harm to TCP cross-traffic.

VIII. Conclusion

In conclusion, ASV advances the state of the art in bandwidth-based DDoS defense mechanisms by introducing a distributed adaptive solution based on selective verification. In ASV, the clients exponentially ramp up the number of requests they send in consecutive time-windows, up to a threshold. The server implements a reservoir-based random sampling to effectively sample from a sequence of incoming packets using bounded space. The novel theoretical analysis of the protocol proves that the performance of ASV (in terms of client success probability and bandwidth consumption) closely approximates an "omniscient" protocol in which all attack parameters are known to clients and the server. NS-2 network simulations of the protocol verify and quantify the effectiveness of ASV against its nonadaptive counterparts and illustrate that under highly variable rate attacks, the performance of ASV adjusts extremely quickly to prevailing attack parameters. In addition, it is shown that the effect of ASV on Internet cross traffic is minimal and comparable to that of its naive nonadaptive counterpart, which represents no-defense attack-only scenarios.

Acknowledgment

The authors would like to thank Dr. M. Alturki, Dr. M. Greenwald, and Prof. J. Meseguer for the their inputs and support throughout this work. Thanks to the Editor-in-Chief, Associate Editor, and IEEE/ACM Transactions on Networking staff for seeing the paper through the long review process.

References

[1] CERT CC, "smurf attack," CERT, Pittsburgh, PA, 1998 [Online]. Available: http://www.cert.org/advisories/CA-1998-01.html
[2] "Three botnets responsible for half of all computer infections," *Infosecurity Mag.*, Feb. 11, 2010 [Online]. Available: http://www.infosecurity-us.com/view/7242/three-botnets-responsible-for-half-of-all-computer-infections/
[3] M. Abadi, M. Burrows, M. Manasse, and T. Wobber, "Moderately hard, memory-bound functions," *Trans. Internet Technol.*, vol. 5, no. 2, pp. 299–327, 2005.
[4] M. AlTurki, J. Meseguer, and C. A. Gunter, "Probabilstic modeling and analysis of DoS protection for the ASV protocol," *Electron. Notes Theoret. Comput. Sci.*, vol. 234, pp. 3–18, 2009.
[5] C. T. Fan, M. E. Muller, and I. Rezucha, "Development of sampling plans by using sequential (item by item) selection techniques and digital computers," *J. Amer. Statist. Assoc.*, vol. 57, pp. 387–402, 1962.
[6] C. Dwork, A. Goldberg, and M. Naor, "On memory-bound functions for fighting spam," in *Proc. CRYPTO*, 2003, pp. 426–444.
[7] D. Eastlake, "Domain name system security extensions," RFC 2535, Mar. 1999.
[8] C. A. Gunter, S. Khanna, K. Tan, and S. S. Venkatesh, "DoS protection for reliably authenticated broadcast," presented at the NDSS, 2004.
[9] E. C. Kaufman, "Internet key exchange (IKEv2) protocol," RFC 4306, Dec. 2009.
[10] X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: Network-layer DoS defense against multimillion-node botnets," *Comput. Commun. Rev.*, vol. 38, no. 4, pp. 195–206, 2008.
[11] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *Comput. Commun. Rev.*, vol. 32, no. 3, pp. 62–73, 2002.
[12] D. Mankins, R. Krishnan, C. Boyd, J. Zao, and M. Frentz, "Mitigating distributed denial of service attacks with dynamic resource pricing," in *Proc. 17th Annu. IEEE ACSAC*, Washington, DC, 2001, pp. 411–421.
[13] R. McMillan, "Two root servers targeted by botnet," *PC Advisor* Feb. 7, 2007 [Online]. Available: http://www.pcadvisor.co.uk
[14] E. Mills, "Twitter, Facebook attack targeted one user," *CNET News*, Aug. 6, 2009 [Online]. Available: http://news.cnet.com/8301-27080_3-10305200-245.html
[15] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," *Trans. Comput. Syst.*, vol. 24, no. 2, pp. 115–139, 2006.
[16] K. Poulsen, "FBI busts alleged DDoS mafia," Security Focus, Symantec, Mountain View, CA, Aug. 26, 2004 [Online]. Available: http://www.securityfocus.com

[17] J. Richards, "Georgia accuses Russia of waging 'cyber-war'," *Times Online* Aug. 11, 2008.
[18] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," in *Proc. ACM SIGCOMM*, 2000, pp. 295–306.
[19] M. Sherr, M. B. Greenwald, C. A. Gunter, S. Khanna, and S. S. Venkatesh, "Mitigating DoS attacks through selective bin verification," in *Proc. IEEE ICNP NPSec*, 2005, pp. 7–12.
[20] M. Srivatsa, A. Iyengar, J. Yin, and L. Liu, "A middleware system for protecting against application level denial of service attacks," in *Proc. Middleware*, 2006, pp. 260–280.
[21] M. Walfish, M. Vutukuru, H. Balakrishnan, D. Karger, and S. Shenker, "DDoS defense by offense," in *Proc. ACM SIGCOMM*, 2006, pp. 303–314.
[22] X. Wang and M. K. Reiter, "Defending against denial-of-service attacks with puzzle auctions," in *Proc. IEEE Symp. Security Privacy*, Washington, DC, 2003, pp. 78–92.
[23] A. Yaar, A. Perrig, and D. X. Song, "SIFF: A stateless internet flow filter to mitigate DDoS flooding attacks," in *Proc. IEEE Symp. Security Privacy*, 2004, pp. 130–143.
[24] X. Yang, D. Wetherall, and T. Anderson, "TVA: A DoS-limiting network architecture," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1267–1280, Dec. 2008.
[25] D. K. Y. Yau, J. C. S. Lui, F. Liang, and Y. Yam, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles," *IEEE/ACM Trans. Netw.*, vol. 13, no. 1, pp. 29–42, Feb. 2005.
[26] C. C. Zou, N. Duffield, D. Towsley, and W. Gong, "Adaptive defense against various network attacks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 10, pp. 1877–1888, Oct. 2006.

Sanjeev Khanna received the Ph.D. degree in computer science from Stanford University, Stanford, CA, in 1996.

He is a Professor and Rosenbluth Faculty Fellow with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia. His primary research interests are in algorithms and complexity.

Santosh S. Venkatesh (S'81–M'86) received the Ph.D. degree in electrical engineering from the California Institute of Technology, Pasadena, in 1986.

He is a Professor with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia. His research interests are in probabilistic models, random graphs, epidemiology, and network and information theory.

Omid Fatemieh received the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign in 2011.

He is a Postdoctoral Researcher with the University of Illinois at Urbana-Champaign. His areas of research interest include wireless and network security, security for white space networks, machine learning in security, and human-centric sensing.

Fariba Khan received the B.S. degree in computer science and engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2004, the M.S. degree in computer science from the University of Illinois at Urbana–Champaign in 2006, and is expected to receive the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign by December 2011.

Her research expertise includes experimentation on the Internet such as queueing and bandwidth for DDoS defense and design of attribute-based messaging. She has enjoyed two full years of teaching at the University of Illinois at Urbana–Champaign.

Carl A. Gunter (SM'95) received the Ph.D. degree from the University of Wisconsin, Madison, in 1985.

He is a Professor of computer science with the University of Illinois at Urbana–Champaign and Director of the Illinois Security Lab and the Center for Health Information Privacy and Security. His research interests are programming-language semantics, formal analysis of networks and security, and privacy, including privacy issues for the power grid and healthcare.

Prof. Gunter is a member of the Association for Computing Machinery (ACM) and the American Medical Informatics Association (AMIA).