Exploiting the Transients of Adaptation for RoQ Attacks on Internet Resources

MINA GUIRGUIS msg@cs.bu.edu
AZER BESTAVROS best@cs.bu.edu
IBRAHIM MATTA matta@cs.bu.edu

Computer Science Department
Boston University
Boston, MA 02215

BUCS-TR-2004-005

Abstract

In this paper, we expose an unorthodox adversarial attack that exploits the transients of a system's adaptive behavior, as opposed to its limited steady-state capacity. We show that a well orchestrated attack could introduce significant inefficiencies that could potentially deprive a network element from much of its capacity, or significantly reduce its service quality, while evading detection by consuming an unsuspicious, small fraction of that element's hijacked capacity. This type of attack stands in sharp contrast to traditional brute-force, sustained high-rate DoS attacks, as well as recently proposed attacks that exploit specific protocol settings such as TCP timeouts. We exemplify what we term as Reduction of Quality (RoQ) attacks by exposing the vulnerabilities of common adaptation mechanisms. We develop control-theoretic models and associated metrics to quantify these vulnerabilities. We present numerical and simulation results, which we validate with observations from real Internet experiments. Our findings motivate the need for the development of adaptation mechanisms that are resilient to these new forms of attacks.

1. Introduction

Motivation and Scope: Over the past few years, Denial of Service (DoS) attacks have emerged as a serious vulnerability for almost every Internet service. An adversary bent on limiting access to a network resource could simply marshal enough client machines to bring down an Internet service by subjecting it to sustained levels of demand that far exceed its capacity, making that service incapable of adequately responding to legitimate requests. In the most recent of these attacks, MyDoom earned its malevolent moniker by crashing SCO Group's Web site as the email-carried W32/Novarg.A, W32/Shimg, and W32/Mydoom worms mounted a widespread and record-setting Distributed DoS (DDoS) attack in the first minutes of February 1st.

While such attacks may be viewed by some as mere nuisances, the impact of these attacks on critical resources and services may cripple our increasingly Internet-dependent economy. Already, the MyDoom attack is estimated to have cost the global economy over $26.1B [34]. Luckily, DoS attacks that overwhelm a service beyond its capacity are not easy to mount because they do require control of a fairly large base [33], *e.g.,* 100K-200K zombie clients in the case of MyDoom. More importantly, by their very nature, DoS attacks are easily discovered, making it possible for counter measures to be taken, including the collection of information that could be used to prosecute the attack perpetrators. [1]

The ability to anticipate a DoS attack and/or to traceback perpetrators thereof are powerful deterrents. But, what if victims of an attack cannot anticipate or even detect that they are under an attack? What if the attack's purpose is not to necessarily cripple a service, but rather to inflict significant degradation in some aspect of the service—*e.g.,* resource utilization, system stability, or service quality?

In this paper, we expose an attack that exploits system dynamics—*i.e.,* the characteristics of a system's transient behavior as opposed to its limited steady-state capacity—to achieve the above adversarial goals. In particular, we show that a determined adversary could bleed a system's capacity or significantly reduce service quality by subjecting the system to a fairly low-intensity (but well orchestrated and timed) request stream that causes the system to become very inefficient, or unstable. We give examples of such attacks—which we term Reduction of Quality (RoQ; as in "rock") attacks—on a number of common adaptive components in modern computing and networking systems. RoQ attacks stand in sharp contrast to traditional brute-force, sustained high-rate DoS attacks [5], as well as recently proposed "shrew" attacks [22] that exploit specific protocol settings such as TCP timeouts. Indeed, as the results in this paper show, RoQ attacks are potentially more potent than both DoS and shrew attacks. Our contributions can be summarized as follows:

+ We formalize a notion of attack "potency", which exposes the tradeoff between the "damage" inflicted by an attacker (*e.g.,* waste in bandwidth) and the "cost" of the attack (*e.g.,* average attack rate).

[1] As of the writing of this paper, SCO and Microsoft were each offering $2.5M rewards for informants of MyDoom's originators.

+ We parameterize our definition of potency to capture the aggressiveness of the attacker (*i.e.*, the level of exposure risk that the attacker is willing to take). We then introduce families of DoS attacks based on aggressiveness.

+ We focus on less aggressive DoS attacks that exploit the transients of the system's underlying adaptation mechanisms. We term these "RoQ attacks," and we identify different attack goals based on the service quality targeted by the attacker for degradation.

+ Unlike existing DoS attacks which target a specific host or a set of flows, we are able to define RoQ attacks which can continually degrade a network element's performance (and thus *all* flows passing through it). We introduce RoQ attacks which target various service qualities such as rate and delay jitter.

+ We crystallize the effect of RoQ attacks by analytically deriving effect of attack traffic on the network element's efficiency-load curve. For instance, an attacker can continually disturb the stability of a router by affecting the congestion signals (prices) fed back to rate-adaptive sources.

+ In addition to analytical, numerical, and simulation results, we present real Internet experiments that confirm the premise of RoQ attacks. For example, we show that a RoQ attack can achieve higher potency than a "shrew" attack [22] whose attack period is chosen so as to target a specific initial TCP timeout value. The shrew attack showed a potency of only 4.3, where as the potency of the RoQ attack was over 12.25 (close to three times that achieved by the shrew attack). The lower potency (*i.e.* damage per unit-cost) of shrew-type attacks is due to its exploitation of the specific timeout behavior of TCP, which can only be induced at a much higher cost (attack traffic).

Paper Outline: We start this paper in Section 2 by introducing the premise and definition of RoQ attacks. Also, we propose a general metric for the quantification of the impact of RoQ attacks. In Section 3, we present an illustrative analytical model whereby the transients of adaptation are simply the result of an optimization process which forces a network to converge to a stable operating point. In Section 4, we present experimental results obtained numerically using a more detailed control-theoretic model in which the dynamics of queue management (*e.g.*, RED) as well as TCP's AIMD adaptation are explicitly modeled. These numerical results are validated in Section 5 using extensive ns simulations in which other phenomena that are not captured in our analytical models are present (*e.g.*, TCP slow-start and timeouts). In Section 6, we present results from Internet experiments we have conducted, which confirm the feasibility of RoQ attacks and provide further validation of the insights we gained from analysis and simulations. In Section 7, we discuss how RoQ attacks could target end systems. In Section 8, we briefly discuss related work, noting that throughout this paper, we point to various pieces of research work as appropriate. We conclude in Section 9 with a summary and future directions.

2. RoQ Attack Premise and Definition

Network Adaptation Mechanisms and Vulnerabilities: End system protocols (*e.g.*, TCP) rely on feedback mechanisms to adapt their sending rates to match their "fair share" of network resources. TCP reduces its sending rate on packet loss/marking and increases its rate on successful packet transmission. Typically, the decrease in rate, which is needed to protect against wasting network utilization, is drastic—*e.g.*, by halving the sending rate—whereas the increase in rate, which is needed to probe for available bandwidth, is slow—*e.g.*, by linearly increasing the sending rate over time. Additive-Increase-Multiplicative-Decrease (AIMD) rules[2] ensure that flows react adequately to congestion in a "friendly" manner to one another—hence the TCP-friendly label [15]. Moreover, these protocols react even more swiftly to excessive losses by completely shutting off their sending rates for a long period of time (*e.g.*, timing out in TCP).

Buffer management schemes play an important role in the effectiveness of transmission control mechanisms as they constitute the feedback signal (by marking or dropping packets) to which such mechanisms adapt. In DropTail, an incoming packet to a full queue is dropped otherwise, its is queued. DropTail doesn't try to achieve any performance improvements, nor does it try to stabilize the queue size. Other Active Queue Management (AQM) techniques have been developed that try to maintain the queue size at a target level and employ probabilistic dropping—*e.g.*, RED and its many variants [13, 23, 30], PI [17] and REM [1]). Such techniques improve fairness and allow flows to send small bursts of packets without experiencing packet drops. Stabilizing the queue at a low target guarantees efficiency while minimizing jitter and round trip time in general.

The adaptation strategies of transmission control protocols such as TCP, while crucial for alleviating congestion, make them vulnerable to losses that are generated through other processes—namely losses that are not the result of congestion (*e.g.*, wireless losses). The impact of such losses on TCP performance was considered in many studies; examples include [2]. In these studies, however, the processes interfering with TCP's adaptation could be considered "non adversarial" in the sense that the losses were more or less the result of (say) a random process as opposed to a calculated attack. Indeed, in recent work, it was shown that an attacker could potentially shut off the communication between two parties (*e.g.*, Alice and Bob) by mounting what is termed as a "shrew" attack [22]—an attack that exploits TCP's time-out mechanism, which is how TCP *adapts* to persistent congestion. In the following sections of this paper, we show that the vulnerabilities resulting from the dynamics of adaptation are

[2]Other TCP-friendly increase/decrease rules have also been proposed and evaluated [3]. All would be susceptible with various degrees to the same issues we consider in this paper.

potentially more serious than shrew attacks in that an attacker could in effect target a potentially large set of connections utilizing a single resource (or set of resources as we will argue later) such as a network link.

Attack Definition: The feedback signal from a given link carries important information to senders about how they should adjust their sending rates. Carrying false signals would cause the senders to back off at times of non-persistent congestion, or to increase when congestion in fact exists! For the purposes of this paper, we focus on attack techniques that would hinder an AQM from stabilizing its queue, and hence resulting in a noisy feedback signal to the end-system transmission controllers, which in turn would lead to high jitters due to oscillations, as well as inefficiencies due to queue drainage, *i.e,* the input rate can't saturate the link capacity.

Consider a bottleneck link of capacity C and a buffer size B shared by m TCP connections and a single CBR connection, representing the attack traffic. For simplicity, we consider an attack comprising a burst of M packets (or bytes) transmitted at the rate of δ packets (or bytes) per second over a short period of time τ, where $M = \delta\tau$. This process is repeated every T units of time. We call M the *magnitude* of the attack, δ the *amplitude* of the attack, τ the *duration* of the attack, and T the *period* of the attack.

Attack Goal: For the above attack, we define Π, the *attack potency*, to be the ratio between the *damage* caused by that attack and the *cost* of mounting such an attack. Clearly, an attacker would be interested in maximizing the damage per unit cost—i.e., maximizing the attack potency.

$$\text{Potency} = \Pi \quad = \quad \frac{\text{Damage}}{\text{Cost}^{\frac{1}{\Omega}}} \tag{1}$$

The above definition does not specify what constitutes "damage" and "cost". In the remainder of this paper, we will consider various instantiations of these metrics. For example, for an attacker aiming to minimize the utilization of a link, a natural metric of "damage" would be the difference between the total bandwidth through the link before and after the attack (excluding the attacker's traffic). If the attacker aims to maximize a link's jitter, then a natural metric of "damage" would be the difference between the standard deviation of the queue size before and after the attack. Similarly, there could be a number of different metrics for what constitutes "cost". Examples include the effective attack bit-rate (*i.e.*, M/T), the attack amplitude δ, the attack duration τ, *etc.*

The above definition uses a parameter Ω to model the aggressiveness of the attacker. A large Ω reflects the highest level of aggression, *i.e.*, an attacker bent on inflicting the most damage and for whom cost is not a concern. Mounting a DoS attack is an example of such behavior. A small Ω reflects an attacker whose goal is to maximize damage with minimal exposure. Unless specified otherwise, for the remainder of this paper we take Ω to be 1.

Detection and Traceback of RoQ Attacks: As we have hinted in the introduction and as the results in this paper will show, an adversary mounting a RoQ attack does not have to overwhelm the network resource under attack in order for its attack to be effective. Moreover, the transients induced by the attack are not much different from those that are possible under normal operation (except that they do not subside). These dimensions of RoQ attacks make it challenging for a network resource to even realize that it is under attack.

Even if the network element is known to be under attack [28], tracing back the perpetrators and/or taking counter measures[3] is much more challenging than in traditional DoS attacks. Specifically, packets generated by an adversary do not even have to use specific destination addresses since the target of the attack is not the destination but rather a router along the path. In effect, this flexibility provides the adversary with two degrees of freedom for "evading" detection.

The first degree of freedom—as in traditional DoS attacks on web servers/sites—is through the use of many sources to mount the attack (*i.e.*, mounting a Distributed RoQ attack). For instance, these sources (possibly zombies) could take turn in sending the attack traffic. The second degree of freedom—and unlike traditional DoS—is through the use of many destinations in the attack traffic. Indeed, every packet sent from a source of the attack could be sent to a different destination. Moreover, as long as they are known to be routed through the resource under attack, these destinations do not even have to be legitimate or live addresses.[4]

3. Network Model and RoQ Exploits

In this section, we illustrate how the transients of adaptation could be exploited in a RoQ attack. We do so by analytically deriving the effect of a RoQ attack on a set of rate-controlled connections, each with transmission rate x_r. The value of x_r for a given connection is adapted based on congestion feedback (equivalently, prices) from links along the route r of that connection. We use the following differential equations [21]:

$$\frac{d}{dt}x_r(t) = \kappa\left(w_r - x_r(t)\sum_{l\in r} p_l(\sum_{l\in s} x_s(t))\right) \tag{2}$$

where κ represents the gain of the system; the first term represents the additive rate increase and the second term represents the multiplicative decrease (as in the AIMD transmission rules of TCP).

The link function $p_l(.)$ reflects the prices (or, costs) fed back to the sources as the input load on the link varies. Figure 1 shows an example of a pricing function. Given

[3] For example using network ingress filtering [11], traceback [31], or new router functionalities, such as those envisioned in [8].

[4] Notice that this destination flexibility makes the need for "spoofing" source addresses to evade detection less important.

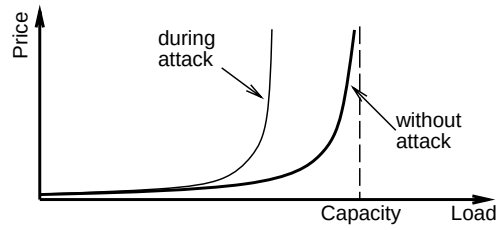


Figure 1: An example of a link pricing function

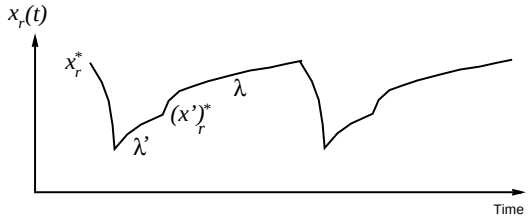


Figure 2: RoQ attack on the stability of the system

such positive, continuous, and increasing function, one could show that the Lyapunov function [29] of the system is given by [21]:

$$U(x) = \sum_{r \in R} w_r \log x_r - \sum_{l \in L} \int_{y=0}^{\sum_{l \in s} x_s} p_l(y) dy \qquad (3)$$

where $U(x)$ represents the net gain—the first term represents the gain in sending rates, while the second term represents the associated costs. A Lyapunov stability analysis shows that the system converges to a stable state x_r^* that maximizes $U(x)$, i.e. $\frac{d}{dt}U(x(t)) > 0$ if $x_r(t) \neq x_r^*$ and equals zero when $x_r(t) = x_r^*$ for all r.

The steady-state rates x_r^* can be obtained by equating to zero the following partial derivatives:

$$\frac{\partial}{\partial t} U(x) = \frac{w_r}{x_r} - \sum_{l \in r} p_l(\sum_{l \in s} x_s) \qquad (4)$$

Given that this system is guaranteed to converge from any starting state $x_r(0)$, one would be interested in that rate of convergence. If the system is perturbed around its steady-state, a linearized model—in terms of new variables $y_r(t)$, such that $x_r(t) = x_r^* + \sqrt{(x_r^*)} y_r(t)$—yields [21]:

$$\begin{aligned} \frac{d}{dt} y(t) &= -\kappa \left(W X^{-1} + X^{1/2} A^T P' A X^{1/2} \right) y(t) \\ &= -\kappa \Gamma^T \Phi \Gamma \; y(t) \end{aligned} \qquad (5)$$

In the above, W, X, and P' are diagonal matrices, where the diagonal elements are w_r, x_r^*, and the derivatives of $p_l(\sum_{l \in s} x_s^*)$, respectively. The matrix A is an $L \times R$ matrix, where entry $a_{l,r}$=1 if connection r is using link l. The diagonal matrix Φ gives the eigen values of the system along the diagonal. The smallest eigen value, call it λ, determines the rate of convergence of the system—a higher value indicates faster convergence, as the transient response of the system decays more rapidly.

The above analysis could be used to provide insights into the effect of RoQ attacks on such a system.

Assume that the system had already stabilized to its steady-state x_r^* values. Since a link is used to its almost maximum capacity, the additional attack load is likely to push the link towards saturation where the fed-back prices are extremely high (cf. Figure 1). Since the RoQ attack involves a sustained rate of δ for τ units of time, the system will be pushed to a new stable point, say $(x')_r^*$. Let λ' refer to the new smallest eigen value, representing the convergence rate to the new stable point (*i.e.*, from x_r^* to $(x')_r^*$). Since during the attack duration τ, the capacity of the attacked link is effectively reduced, the link pricing function is pushed to the left, as shown in Figure 1. Such higher prices result in faster convergence (*i.e.*, higher λ') and lower $(x')_r^*$.[5]

As soon as the system stabilizes to the new $(x')_r^*$, an optimized RoQ attack would cease its attack so as to allow the system to return to its original state x_r^*. This attack pattern then repeats. Figure 2 illustrates the effect of such RoQ attack, when the system spends its time oscillating between different states, in the presence and absence of the attack traffic. Note that in general, the attack traffic may destroy the "contractive" mapping property of the pricing function and so the system may not converge to a fixed point while under attack.

To assess the impact of the RoQ attack, we turn our attention to the attack potency as defined in Equation 1. In terms of the above analytical model parameters, one may capture the "damage" caused by the attack using the expression $\delta(\frac{1}{\lambda'} + \frac{1}{\lambda})$. Intuitively, this expression represents the wasted bandwidth (and other service qualities such as delay and rate jitter, as we discuss later) during instability. Also, one may capture the "cost" of the attack by $(\delta/(\frac{1}{\lambda'} + \frac{1}{\lambda}))$. Intuitively, the cost increases with increasing peak rate and decreases with longer attack period. Figure 3 shows a single measure of attack potency versus attack peak rate, where potency in this model is defined by:

$$\begin{aligned} \text{Potency} = \Pi &= \frac{\delta(\frac{1}{\lambda'} + \frac{1}{\lambda})}{\left(\delta/(\frac{1}{\lambda'} + \frac{1}{\lambda})\right)^{1/\Omega}} \\ \Pi &= \delta^{1-\frac{1}{\Omega}} (\frac{1}{\lambda'} + \frac{1}{\lambda})^{1+\frac{1}{\Omega}} \end{aligned} \qquad (6)$$

As we eluded before, Ω reflects the relative values that an attacker attributes to "damage" versus "cost", or equivalently the desired level of aggression.

Figure 3 shows the potency plots for a 2-link tandem network used by three rate-controlled sources. The pricing

[5] Since during the attack, the pricing function includes δ, the convergence rate λ' depends on δ.

function of the first link is $p_1(y) = 0.2/(10 - y)$, and that of the second link is $p_2(y) = 0.5/(5 - y)$. One connection crosses both links, while each of the other two crosses only one link. The additive-increase parameters w_r are taken to be (0.5, 1.5, 1.0)—the increase rate is lowest for the longest 2-link connection.

We observe that for a given value of Ω, there is an optimal attack peak rate δ that optimizes potency Π. On the one hand, a low attack peak rate, while less costly, results in minimal damage, and thus results in low potency. On the other hand, a higher attack peak rate, while resulting in higher damage, may be too costly that it results in lower potency. This suggests that an optimized RoQ attacker can achieve higher potency (i.e. higher damage per unit-cost) by forcing the system into instabilities at the right times, injecting only the right amount of attack traffic. This is true for any level of attacker's aggressiveness, Ω.

Note that in this model, a RoQ attacker attempts to cause damage by specifically exploiting the AIMD-like dynamics of the system. Such dynamics are essential in driving the system to fairness. If one were to protect the system against RoQ attacks through other increase-decrease rules as AIAD or MIMD (to avoid drastic reductions at the time of the attack or to ramp the rate back up more quickly when the attack ceases), then fairness can't be achieved. Such tradeoff between protection against attacks and performance during normal operating conditions is very important to highlight. In the remainder of this paper, we confirm these analytical observations using more elaborate models, and also using simulations and Internet experiments. These more detailed models will include other dynamics, such as those resulting from TCP timeouts. Nevertheless, the simple model presented here unifies important general dynamics of many adaptive computing and networking systems. For example, the x_r variables may represent the sending rates of TCP sources, or the admission rates of different classes of web requests. Additionally, the pricing function $p_l(.)$ may represent the congestion feedback signals (e.g. from RED-like AQM) to TCP sources, or the degradation in the service rate of a web server as it thrashes under high load.

4. Vulnerability Assessment

Wasted Bandwidth as the Target of RoQ Attacks: Consider an attacker bent on maximizing the bandwidth wasted as a result of a RoQ attack.[6] Thus, we define b_w, the *wasted bandwidth*, to be the difference between the achievable throughput under normal conditions and the achievable throughput under a RoQ attack, both measured as the number of packets (or bytes) going through the link for legitimate traffic. Thus, b_w quantifies the absolute "damage" resulting from the attack. Let b_a, the *attack bandwidth*, denote the

[6]Later, we consider other attack objectives—*e.g.*, maximize jitter.

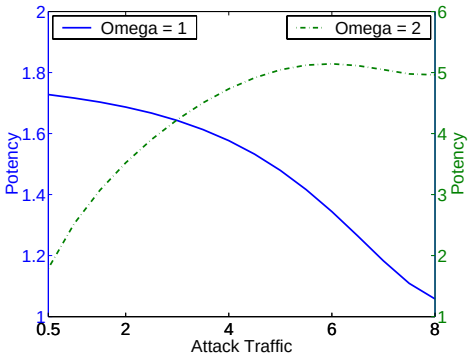


Figure 3: Attack potency versus attack peak rate for Ω=1 (labels on left y-axis) and for Ω=2 (labels on right y-axis).

bandwidth consumed by the attacker over the link under attack. Clearly b_a could be construed as the cost of the attack—for instance because the higher the value of b_a the more likely that the attacker would be identified. Substituting in Equation 1, we get the following definition of attack potency:

$$\Pi = \frac{b_w}{b_a^{\frac{1}{\Omega}}} \quad (7)$$

The above definition doesn't account for the total traffic sent by the attacker but only for the attack traffic that is observable at the link under attack. We have also used an alternative measure of cost that accounts for the total traffic injected by the adversary; the resulting potencies were indistinguishable.

Detailed Analysis: We extend an analytical fluid model similar to that proposed in [16, 20, 24, 32] to assess the potential damage that could be inflicted by an attacker exploiting the dynamics of adaptation for TCP + AQM (namely, AIMD + RED).

We consider a dynamic fluid model of m TCP connections and a single CBR connection, representing the attacker's traffic, traversing a single bottleneck of capacity C. The round trip time $r_i(t)$ at time t for connection i is equal to the round-trip propagation delay D_i between the sender and the receiver for connection i, plus the queuing delay at the bottleneck router. Thus $r_i(t)$ can be expressed by

$$r_i(t) = D_i + \frac{b(t)}{C} \quad (8)$$

where $b(t)$ is the backlog buffer size at time t at the bottleneck router. We denote the propagation delay from sender i to the bottleneck by D_{s_ib}, which is a fraction α_i of the total propagation delay.

$$D_{s_ib} = \alpha_i D_i \quad (9)$$

The backlog buffer $b(t)$ evolves according to the equation

$$\dot{b}(t) = \sum_{i=1}^{m} x_i(t - D_{s_ib}) + y(t - D_{ab}) - C \quad (10)$$

which is equal to the input rate $x_i(.)$ from the m connections plus the attacker's traffic $y(.)$, minus the output link rate. Notice that the input rates are delayed by the propagation delay from the senders and the attacker to the bottleneck D_{s_ib} and D_{ab}.

The losses seen by the connections depend on our choice of queue management at the bottleneck router. For RED [14], the congestion loss probability $p_c(t)$ is given by:

$$p_c(t) = \begin{cases} 0 & v(t) \leq B_{min} \\ \sigma(v(t) - \varsigma) & B_{min} < v(t) < B_{max} \\ 1 & v(t) \geq B_{max} \end{cases} \quad (11)$$

where σ and ς are the RED parameters given by $P_{max}/(B_{max} - B_{min})$ and B_{min}, respectively, and $v(t)$ is the average queue size, which evolves according to the equation:

$$\dot{v}(t) = -\beta C(v(t) - b(t)), \qquad 0 < \beta < 1 \quad (12)$$

Notice that in the above relationship, we multiply β by C since RED updates the average queue length at every packet arrival, whereas our model is a fluid model [16, 24].

The throughput of TCP, $x_i(t)$ is given by

$$x_i(t) = \frac{w_i(t)}{r_i(t)} \quad (13)$$

where $w_i(t)$ is the size of the TCP congestion window for sender i.

According to TCP's AIMD rules, the dynamics of TCP throughput for each of the m connections can be described by the following differential equations:

$$\begin{aligned} \dot{x}_i(t) &= \frac{x_i(t - r_i(t))}{r_i^2(t)x_i(t)}(1 - p_c(t - D_{bs_i}(t))) - \\ &\quad \frac{x_i(t)x_i(t - r_i(t))}{2}(p_c(t - D_{bs_i}(t))) \\ i &= 1, 2, .., m \end{aligned} \quad (14)$$

The first term represents the additive increase rule, whereas the second represents the multiplicative decrease rule. Both sides are multiplied by the rate of acknowledgments for the last window of packets $x_i(t - r_i(t))$. In the above equations, the time delay from the bottleneck to sender i, passing through the receiver i, is given by:

$$D_{bs_i}(t) = r_i(t) - D_{s_ib} \quad (15)$$

The attack traffic, $y(t)$, follows a square wave is given by:

$$y(t) = \begin{cases} \delta & t \bmod T \leq \tau \\ 0 & otherwise \end{cases} \quad (16)$$

where δ is the attack amplitude, τ is the attack duration and T is the attack period. All defined above.

The fluid model presented above is only capable of capturing the dynamics due to AIMD. Thus the effects of slow start and timeouts are ignored. Despite these limitations, the model is useful as it provides a "lower-bound" assessment of vulnerability to RoQ attacks. By lower bound, we mean that in reality, as we will show in simulation and Internet experiments, the impact of a RoQ attack is likely to be *even worse* than the model predicts. This is so because it is reasonable to assume that the attack duration will be long enough for many connections not only to back off, but also to go into timeout/slow start, which would increase the "damage" from the attack.

Quantitative Assessment of Attack Potency: We are now ready to put the fluid model just developed to work by numerically solving for the attack potency. In the next two sections, we present results of ns simulations and Internet experiments that relax the simplistic assumptions of our model to capture various effects (*e.g.*, slowstart, timeouts, cross traffic on multiple hops) and other queue management policy (*e.g.*, DropTail).

Based on the fluid model developed above, we quantify the impact of RoQ attacks by considering an example parameterization of the model. Specifically, we consider a bottleneck RED link of 2,000 packets/sec (=16Mbps) capacity traversed by 19 TCP connections and a single CBR connection representing the attacker's traffic—a total of 20 connections. The bottleneck's buffer size is 250 packets. The RED parameters, minimum and maximum thresholds, were tuned to 50 and 120, respectively. The weight parameter was chosen to be 0.00001, and the maximum loss probability was chosen to be 0.1. The propagation delays of the connections were generated uniformly between 80 and 120 msec. We start the attack at time $t = 40$ with a δ = 16Mbps, $\tau = 0.2$ seconds, and $T = 5$ seconds. The value T was chosen to roughly match the time it takes the queue to converge to its steady state operating point. As we will discuss later, an attacker could "discover" this value using a number of methods.

Figure 4(a) shows the queue size predicted by our analytical model for $0 \leq t \leq 100$. Clearly, after a short while, RED's queue size stabilizes, indicating that the system converges to an efficient operating point. However, as soon as the attack is started, one can see the wide oscillations in queue size, which includes periods of time during which RED's queue is empty, implying an inefficient operation (*i.e.*, a loss of capacity, which is precisely the goal of the attack). Figure 4(b) takes a closer look at the period of time $35 \leq t \leq 50$. It shows how the oscillations resulting from two attack cycles at time $t = 40$ and $t = 45$ cause the queue size to reach zero, leading to under utilization.

Figure 4(c) shows the average throughput for each flow traversing the link from $t = 40$ to $t = 60$ (*i.e.*, while the link is under attack). The flow with ID=20 represents the attacker's flow. Clearly, the average throughput consumed

by the attacker is indistinguishable from that consumed by the remaining 19 legitimate flows. Thus, by simply looking for flows that use more than their fair share over the time scale of the attack period (or even shorter if the attacker sends its packets with different destinations, as we pointed out in Section 2), it would be impossible to identify the attacker—not to mention realizing that the system is under attack in the first place! Indeed, as we mentioned earlier in the paper, one of the dangerous aspects of exploits that capitalize on system dynamics is that they are harder to detect.

5. Simulation Experiments

As we mentioned before, the analytical model used in the quantitative evaluation in Section 4 does not capture many aspects of the system dynamics (*e.g.*, variability in RTT, effects of timeouts, etc.) To validate that (simple) model, we present results from ns simulations [10] in which such limitations are not present.

Similar to the setting we chose for our numerical evaluation, we consider a bottleneck link with 2,000 packets/sec (=16Mbps) capacity, 250-packet buffer size, and a 20ms propagation delay shared by a number of TCP connections with unlimited data to send, and a CBR connection representing the attack traffic. The propagation delay for the access links are chosen uniformly at random between 15 and 25 msec. So the average round trip propagation delay is around 120 msec. We run the experiments with 10, 20, 30 and 120 connections, including the attack traffic.

Results of RoQ Attack on a RED Queue: Figures 5(a), 5(b), and 5(c) show the results obtained from ns simulations when the queue management is RED. The RED parameters, minimum and maximum thresholds, were tuned to 50 and 120, respectively. The weight parameter was chosen to be 0.0001, and the maximum loss probability was chosen to be 0.1. Figures 5(a), 5(b), and 5(c) show the queue size (and average queue size as estimated by RED) for $0 \leq t \leq 100$, a closeup of the period $35 \leq t \leq 50$ showing the stabilized queue followed by two attack cycles at $t = 40$ and $t = 45$, and the average throughput achieved for the 19 legitimate flows and for the 20th attack flow. Clearly, the results of our ns simulations shown in Figures 5(a), 5(b), and 5(c) match fairly well those obtained using the analytical model of Section 4 in Figures 4(a), 4(b), and 4(c).

Results of RoQ Attack on a DropTail Queue: Figure 6(a), 6(b), and 6(c) show the results obtained from ns simulations when the queue management is DropTail. The difference between these results and those obtained for RED is that the queue size (prior to the attack) does not stabilize around a target value. This is expected given that DropTail does not aim to stabilize the queue. Also, the average throughput for the different flows is less uniform under DropTail than under RED. Again, this is expected by virtue of RED's design principles. Surprisingly, though, our results suggest that both RED and DropTail are equally susceptible to the attack. In other words, RED's attempt to more fairly drop/mark packets across all flows based on buffer occupancy, while helping it with fairness, does not protect it from being exploited by a RoQ attack. Since DropTail recovers faster than any AQM (since it doesn't generate early congestion signals after the attack is over), a RoQ attack basically degenerates any AQM to DropTail making them all susceptible to being exploited.

RoQ Attack Potency: Table 1 shows the achievable potency of the RoQ attack predicted by our numerical model of Section 4 and validated by the simulation experiments presented above. For DropTail, the table shows that a well orchestrated RoQ attack with a rate of 44 packets/sec, representing less than 2.5% of the link's capacity, is capable of robbing that link of over 25% of that capacity—a potency of over 10.

	Damage b_w	Cost b_a	Potency Π
RED (model)	177	79	2.15
RED (ns)	467	56	8.30
DropTail (ns)	475	44	10.7

Table 1: Potency values for Bandwidth (with $\Omega = 1$)

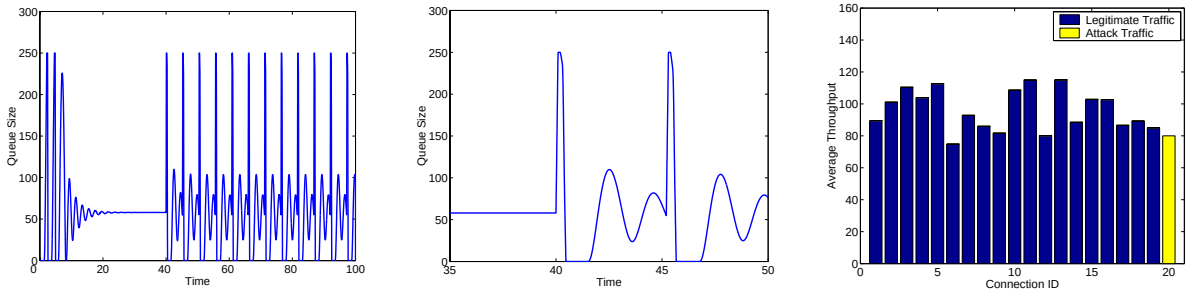
Jitter as a Goal of RoQ Attacks: In our discussions so far, we have focused on bandwidth as the subject of the RoQ attack. This need not be the case.[7] As both of our numerical and simulation results shown in Figures 4, 5, and 6 clearly show, RoQ attacks result in other undesirable effects, including a larger delay jitter as evident from the larger oscillations in queue size induced by the attack for both RED and DropTail. Such reduction in service quality could well be the target of a RoQ attack. Table 2 shows the achievable potency of the RoQ attack we have considered above, if increasing jitter is the goal of the attack.

One interesting observation from the results shown in Table 2 is that the impact on jitter of RoQ attacks on a RED queue is much more pronounced than that on a DropTail queue. This is expected since RED actively aims to reduce jitter (by stabilizing the queue size) whereas DropTail does not. Thus, a RoQ attack on a RED queue could be seen as robbing RED of its advantage over DropTail.

The results in Tables 1 and 2 confirm what we mentioned earlier–namely that the analytical model of Section 4 provides us with a "lower bound" on the achievable potency in practice.[8] For example, the model predicted a potency of 2.15 for an attack aiming to bleed link bandwidth when our ns simulations resulted in a potency of 8.3. In other words, in addition to the bandwidth it consumes, an attacker can bleed more than eight times that bandwidth by exploiting system dynamics. Similarly, the model predicted a potency of 0.36

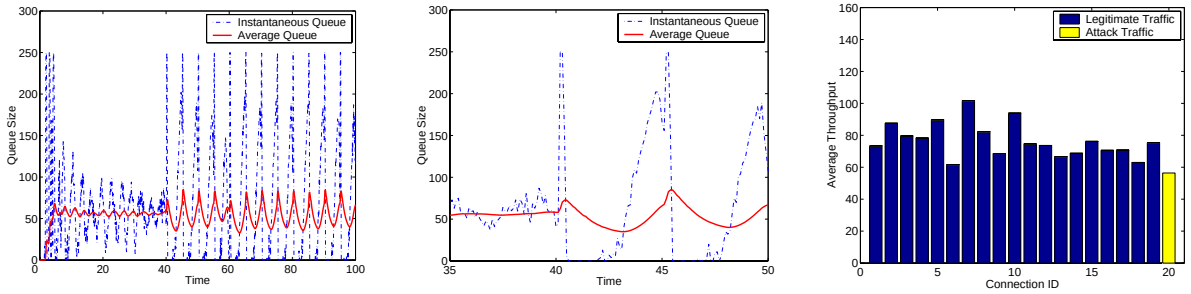
[7] Indeed, this may be a difficult goal to achieve in practice.

[8] Our Internet experiment will further validate this in the next section.



(a) (b) (c)

Figure 4: Assessment of vulnerability to RoQ attacks: Results from analytical model of TCP+RED showing queue size over time (left and center) and and the throughput achieved per flow in the presence of the attack (right).



(a) (b) (c)

Figure 5: Assessment of vulnerability to RoQ attacks: Results from ns simulations over a RED showing queue size over time (left and center) and the throughput achieved per flow in the presence of the attack (right).

	Delay Jitter		Damage	Cost	Potency
	Before	After	(msec)	b_a	Π
RED (model)	0.0	28.5	28.5	79	0.36
RED (ns)	8.50	37.5	29.0	56	0.52
DropTail (ns)	32.0	42.0	10.0	44	0.23

Table 2: Potency values for Delay Jitter (with $\Omega = 1$)

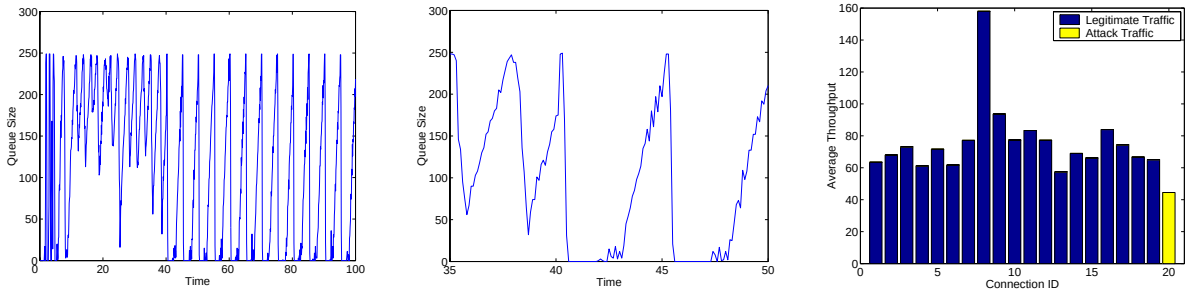
for an attack aiming to increase delay jitter when our ns simulations resulted in a potency that is 44% higher at 0.52. The meaning of this potency is that by exploiting network dynamics, a RoQ attacker is able to inflict 0.52 milliseconds of added jitter for every extra packet it injects in the network per second.[9]

Tuning the RoQ Attack Parameters: The parameters of the RoQ attack used to produce the results shown in Tables 1 and 2 were chosen rather arbitrarily. Specifically, δ, τ, and T were "educated guesses" given the *a priori* known settings of the experiment—namely, RTT, link capacity, *etc.* Are these attack parameters optimal though? Could a different set of parameters lead to higher potency?

For a chosen burst of size $M = \delta \times \tau$, the attacker has a choice of using a large amplitude δ over a short duration τ, or else using a smaller amplitude over a longer duration. Figure 7(a) shows the potency of a RoQ attack with $M = 5$ for various δ (or equivalently for various $\tau = M/\delta$) values and for different number of flows. The figure shows that there is an "optimal" choice of δ that maximizes the attack potency. Figure 7(b) shows that this "optimal" setting changes as a function of M, for a given number of flows (namely 20). Thus, for any value of M and for any number of flows traversing the link under attack, an adversary could pick the value that will maximize the potency of its attack.

Next, we consider T, the attack period. Intuitively, one would think that the smaller the value of T, the more the damage inflicted on the link. But, as Figure 7(c) shows, the attack potency decreases fairly rapidly once the period T decreases below some threshold (*i.e.*, moving towards 0 on the x-axis). For example, with 10 flows traversing the link under attack, the potency of an attack of magnitude $M = 5$ quickly diminishes when T is less than 5 seconds. In other words, there is a point of diminishing returns (to the attacker) beyond which increasing the rate of the attack does not payoff in terms of the harm done per packet of attack traffic. Thus, an adversary bent on causing the maximum harm with the minimum attack traffic would chose the minimum value of T that is larger than

[9]Notice that this 0.52 msec is the best an attacker can do since this value represents the transmission time of a packet.



(a) (b) (c)

Figure 6: Assessment of vulnerability to RoQ attacks: Results from ns simulations over a DropTail queue showing queue size over time (left and center) and the throughput achieved per flow in the presence of the attack (right).

that point of diminishing returns.

As discussed above, for each value of M there are "optimal" values of δ (τ) and T. Assuming that these optimal values are selected, Figure 7(d) shows the attack potency as a function of M. The figure suggests that for a given number of flows traversing the link under attack, there is a point of diminishing returns beyond which increasing the magnitude of the attack does not buy the attacker more potency.

Notice that the above tuning of the RoQ attack parameters depends very much on the aggressiveness Ω of the attacker (assumed to be 1 above). Figures 7(e)-(h) show how potencies change when the attack parameters are changed, when $\Omega = 2$. Clearly, the optimal parameters of a RoQ attack with $\Omega = 2$ are *different* from those obtained with $\Omega = 1$.

The above tuning of the attack parameters was performed with an eye on maximizing the wasted bandwidth per attack "byte". Would this set of parameters be also optimal for a RoQ attack aiming to maximize delay jitter? Figures 8(a-h) show the parameter space of a RoQ attack aiming to maximize delay jitter. Clearly, the set of parameters that optimize such an attack are *different* from those optimizing an attack on link bandwidth.

Measurement-Based On-Line Tuning of RoQ Attacks: In the above tuning of RoQ attacks (to optimize potency), it was assumed that system parameters such as link capacity, number of flows, and RTTs are known *a priori*. This is a meaningful assumption if the adversary is an oracle (or an author :), but not otherwise. Even if optimal attack parameters are possible to derive, these parameters are likely to depend on the profile of the traffic going through the router (*e.g.*, the mix of RTTs). This suggests that any set of fixed parameters may not be optimal at all times. One alternative to fixing the values of δ, τ, and T is to use a measurement based approach to orchestrating the attack. Specifically, it is reasonable to assume that an adversary would have at its disposal the arsenal of measurement tools available publicly—tools that could be used to estimate the bandwidth or the buffer size of the targeted link, for example. Alternatively, an attacker could use a TCP connection as a "probe" via which to measure the damage caused by its attack. Using that as the feedback signal to a simple controller, the attacker could easily adjust the parameters of the RoQ Attack to maximize potency.

6. Internet Experiments

Experimental Setup: Figure 9 depicts the experimental setup we used for our Internet experiments. It consists of a router (R), a local content server ($S0$), three client machines (C_1, C_2, and C_3), a source of attack traffic (A_s) and a sink of attack traffic (A_k). The router's server-side interface is connected to a 100 Mbps switch that connects the whole setup to the Internet, the content server machine ($S0$) and the attack source (A_s). The router's client-side interface is connected to another 100 Mbps switch that connects it to the local subnet where client machines (C_1, C_2, and C_3) and attack sink (A_k) reside. The network interface cards on all machines run at 100 Mbps except for the router's client-side interface, representing the bottleneck link, which runs at 10 Mbps. All machines run Linux RedHat version 2.4.20. The router uses iproute2 and tc [19] to run different packet scheduling disciplines. In all experiments we report on in this paper, we used a packetized version of FIFO (called pfifo).

A client (C_i) is configured to request local data transfers from the local server $S0$ or via HTTP 1.1 from a remote transatlantic Internet server ($S1$). As described in Section 2, the attack source (A_s) injects UDP packets destined to sink (A_k)[10] following the RoQ attack square wave pattern with parameters δ, τ and T. Since the traffic injected by A_s passes through the bottleneck link, the attacker's main goal is to adjust its parameters to maximize potency.

In calculating results from a given experiment we always discard the data from the first 10 seconds to allow for

[10]Unlike traditional DoS attacks, A_k is not the target of the attack, but rather a co-conspirator of A_s, or even a bystander which does not even have to be on-line (as long as packets destined to it are routed through the target of the attack—namely router R).

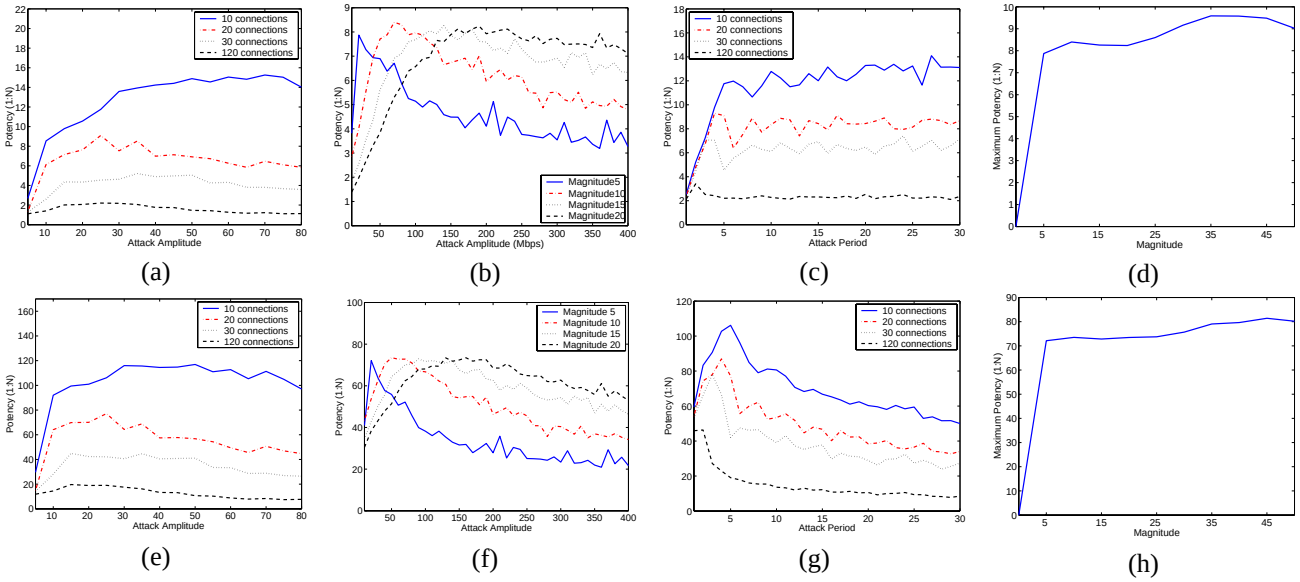


Figure 7: Tuning attack parameters to maximize potency (for link bandwidth) using $\Omega = 1$ (top) and $\Omega = 2$ (bottom): Effect of changing δ for a fixed magnitude ($M = 5$) under different number of flows (a & e). Effect of changing δ for a fixed number of 20 flows under different attack magnitudes (b & f). Effect of changing T for a fixed magnitude ($M = 5$) with corresponding optimal δ under different number of flows (c & g). Effect of changing M with corresponding optimal δ and T (d & h).

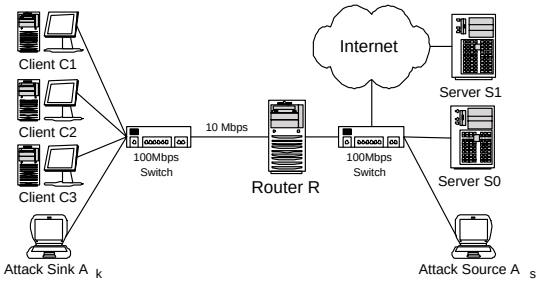


Figure 9: Setup for Internet Experiments.

throughput to ramp up. Unless otherwise specified, each experiment lasted for two minutes. As one would expect, experiments that only involved machines in our laboratories produced almost identical results every time they were run. When our experiments involved remote servers (*e.g.*, $S1$), results were not consistent when experiments were conducted over different times (*e.g.*, mornings versus evenings). However, when experiments involving remote servers were conducted at the same time-window (on different days), the results were fairly consistent. All such experiments we report on in this section were performed in a three-hour window of time from 4:00am to 7:00am GMT.

RoQ Attacks on Flows with Short RTTs: In this set of experiments, each client C_i opens two TCP connections to the server $S0$ for a total of 6 TCP flows traversing the bottleneck link. Under no RoQ attack traffic, the 6 flows achieve a total throughput of 8.45 Mbps, with the average round-trip time between the clients and the server measured to be 15msec. To ensure queuing and induce drops at the bottleneck link R, the attack rate δ was fixed at 9.5 Mbps, and the attack duration τ and period T were varied to assess the potency of various RoQ attacks as was done numerically in Section 4 and in simulation in Section 5.

With the attack duration τ fixed at 40 msec, Figure 10(a) shows the effect of varying the attack period T. With the attack idle time (*i.e.*, $T - \tau$) fixed at 250 msec, Figure 11(a) shows the effect of changing the attack duration τ. As these figures suggest, the maximum potency was achieved when the attack period was set to 270msec and when the attack duration was set to 20msec. With these settings, the total throughput achieved by the 6 flows dropped from 8.45 Mbps to 4.47 Mbps with only 0.45Mbps of attacker's traffic—a potency of almost 9 (using $\Omega = 1$).

RoQ Attacks on Flows with Long RTTs: In this experiment, each client issues two HTTP requests for a very large file to the remote server $S1$. Thus, in total, 6 TCP connections traverse R. With no RoQ attacks present, the total bandwidth grabbed by the 6 flows is 8.08Mbps,[11] with the average round-trip time between the clients and the server measured to be 120msec. A characterization of the Internet path from R to $S1$ revealed a 21-hop route with a bottleneck bandwidth well over the 10Mbps capacity of R.

With the attack duration τ fixed at 40 msec, Figure 10(b)

[11] Notice that this value is slightly lower than the 8.45Mbps total throughput obtained with local connections. This could be explained by virtue of the longer RTT of transatlantic connections.

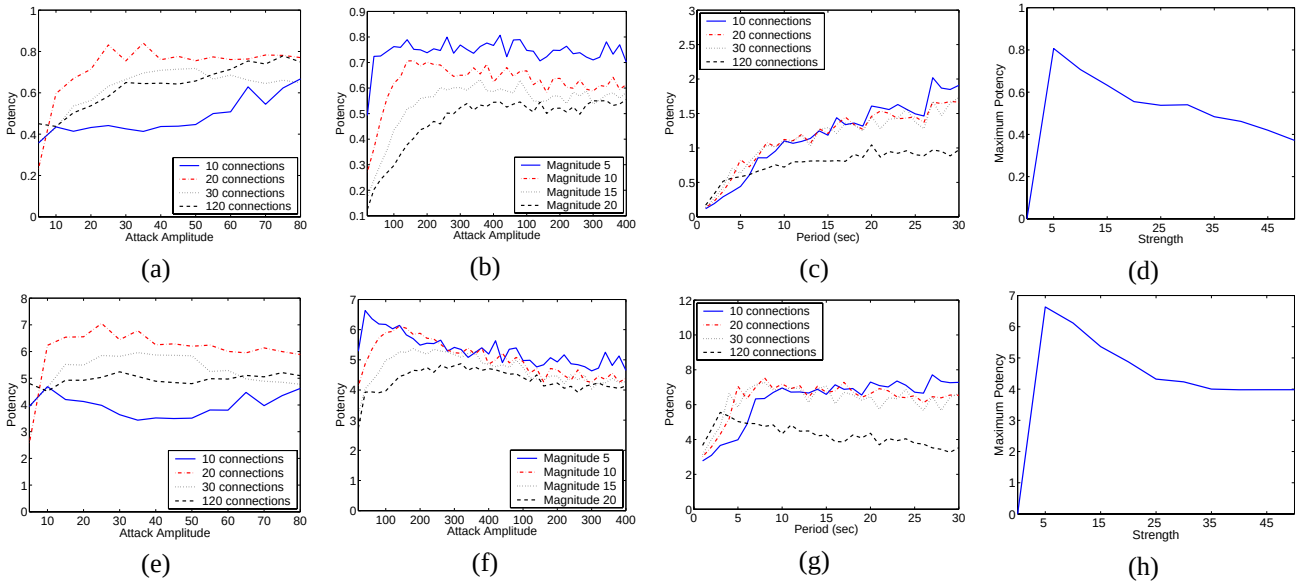


Figure 8: Tuning attack parameters to maximize potency (for delay jitter) using $\Omega = 1$ (top) and $\Omega = 2$ (bottom): Effect of changing δ for a fixed magnitude ($M = 5$) under different number of flows (a & e). Effect of changing δ for a fixed number of 20 flows under different attack magnitudes (b & f). Effect of changing T for a fixed magnitude ($M = 5$) with corresponding optimal δ under different number of flows (c & g). Effect of changing M with corresponding optimal δ and T (d & h).

shows the effect of varying the attack period T. With the attack idle time (*i.e.*, $T - \tau$) fixed to 1,000 msec, Figure 11(b) shows the effect of changing the attack duration τ. The trends in these figures are consistent with those observed in simulations—for example the trends in Figure 10(b) match those in Figure 7(c & g). As these figures suggest, the maximum potency was achieved when T was set to 1,040 msec and when the attack duration was set to 40msec. With these settings, the total throughput achieved by the 6 flows dropped from 8.08 Mbps to 3.6 Mbps with only 0.37 Mbps of attacker's traffic—a potency of almost 12 (using $\Omega = 1$).

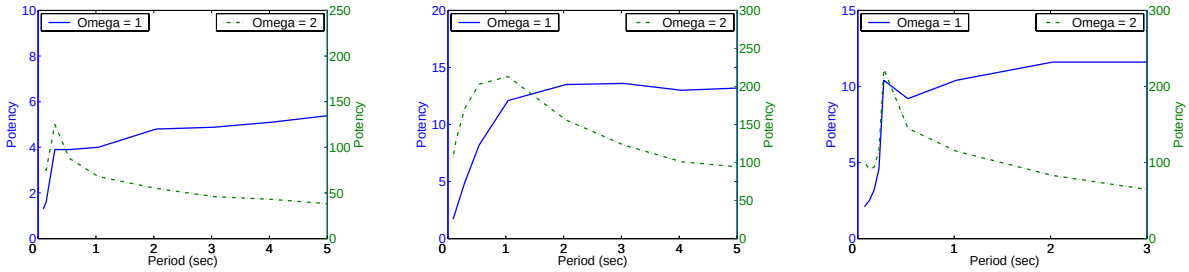
It is worth noting that setting the attack duration τ to 240msec, which is close to twice the round-trip time, and setting the attack period T to around 1 sec reduces the RoQ attack to the shrew attack [22], which would cause TCP to perpetually timeout. This setting is indeed one of the points shown in Figure 11(b). Interestingly enough, relying on TCP timeout mechanism to cause the maximum damage per unit of attack traffic is not effective. Indeed, if we examine the results of the experiment with settings resulting in a shrew attack, we observe that the total throughput of the 6 flows was brought down from 8.08 Mbps to 1.25 Mbps with an average attack traffic of 1.58 Mbps—a potency of only 4.3. Notice that with a period T of around 1 second, an attack with a much lower rate of 0.3 Mbps would achieve a potency of over 12.25 (close to three times that achieved by the shrew attack), bringing down the throughput achieved by the six flows to 4.4 Mbps.

Another point worth mentioning is that a "flooding" approach to hijacking a link's capacity (a la DoS attacks) is also quite inefficient if maximizing the damage caused per byte of attack traffic is the goal. While flooding would indeed shut down the TCP connections, its potency will also approach 0 as the attacker must inject a lot of traffic (not to mention increasing its exposure).

These results demonstrate that to maximize the marginal utility of attack traffic, exploiting the transients of adaptation is the way to go; it is a much more efficient strategy than exploiting specific protocol properties (such as timeouts) or simply blasting packets at the highest rate.

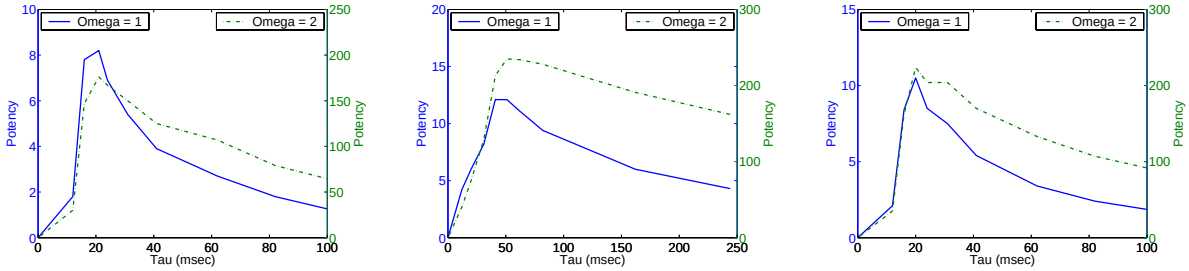
RoQ Attack on a Mix of Long and Short RTT Flows: In this experiment, each client opens two connections: one on the remote server $S1$ and one on the local server $S0$. Thus, in total, 6 TCP connections traverse R—three with RTT of around 15 msec and three of around 120 msec. With no RoQ attacks present, the total bandwidth for the 6 flows is 8.45Mbps. With the attack duration τ fixed to 20 msec, Figure 10(c) shows the effect of varying the attack period T. With the attack idle time (*i.e.*, $T - \tau$) fixed to 250 msec, Figure 11(c) shows the effect of changing the attack duration τ.

As these figures suggest, the maximum potency is achieved when the attack period was set to 270msec and when the attack duration was set to 20msec. With these settings, the total throughput achieved by the 6 flows dropped from 8.45 Mbps to 3.7 Mbps with only 0.45 Mbps of attacker's traffic—a potency of over 10.5. These results are quite similar to those obtained with six short RTT flows—



(a) (b) (c)

Figure 10: Potency of Internet RoQ attack as the attack period T (in seconds) is changed while keeping the attack duration τ constant. Two potency curves are shown one for $\Omega = 1$ (labels on left y-axis) and one for $\Omega = 2$ (labels on right y-axis). Results shown are for (a) RoQ attack on flows with short RTTs with τ=40msec, (b) flows with long RTTs with τ 40msec, (c) a mix of long and short RTT flows with τ = 20msec.



(a) (b) (c)

Figure 11: Potency of Internet RoQ attack as the attack duration τ (in milliseconds) is changed while keeping the attack duration τ constant. Two potency curves are shown one for $\Omega = 1$ (labels on left y-axis) and one for $\Omega = 2$ (labels on right y-axis). Results shown are for (a) RoQ attack on flows with short RTTs with $T - \tau$=250msec, (b) flows with long RTTs with $T - \tau$ 1,000msec, (c) a mix of long and short RTT flows with $T - \tau$ = 250msec.

suggesting that a RoQ attack on the lower-end of the RTT profile of flows going through the bottleneck is effective.

7. RoQ Attacks on End Systems

So far we have focused on the vulnerability of network adaptation to RoQ attacks. Clearly, adaptation mechanisms pervade the design of many other types of systems and services, including network servers and end systems, such as web servers, DNS servers, and BGP servers. The insights gained from our work, as well as the framework we adopted in our analysis (*e.g.*, models, techniques, and metrics) are quite applicable in evaluating similar RoQ vulnerabilities. Indeed, as part of this project, we have also examined the vulnerability of a variety of admission control strategies that are often deployed in end hosts for overload protection [35], reaching very similar conclusions to the ones presented in this paper. Space limitations prohibit us from presenting these results with at any meaningful depth. Nevertheless, in this section we present an illustrative example and "sample" numeric results obtained using a control theoretic model—similar in nature to those presented earlier in this paper—to evaluate RoQ attacks on an admission controller, which is widely deployed in front-ends of Internet web servers and web server farms.

The feedback delay inherent in the design of any admission controller constitutes the "Trojan Horse" through which a RoQ attack could be mounted. Consider an admission controller that sets its admission rate of incoming requests as a function of the utilization of its backend. Now, consider a point in time when offered load is low enough for the admission controller to allow a large percentage of all requests to go through. At this point, a surge in demand (*e.g.*, a large number of requests) in a very short period of time would push the system into overload conditions. This, in turn, would result in the admission controller shutting off subsequent legitimate requests for a long time given the fact that under overloaded conditions, the system operates in an inefficient region (*e.g.*, due to thrashing). Once the system "recovers" from the ill-effects of this unsuspected surge in demand, an attacker would simply repeat the process, resulting in a RoQ attack pattern akin to that we defined in Section 2.

Figure 12 shows results from such an attack, which were obtained numerically from a control theoretic model (not

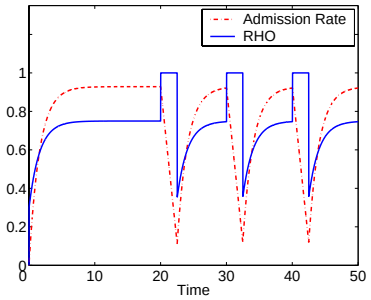


Figure 12: RoQ attack on a proportional admission controller and its impact on admission rate and on utilization.

shown) of the admission controller and of the relationship between service rate and utilization of the backend server. Figure 12 shows the admission rate as well as the utilization of the backend system over time. Clearly, within 5 seconds of operation, the system converges to an efficient operating region with admission rate at 0.85 and utilization around 0.75. The RoQ attack is mounted at time 20 for a duration of τ=1 second and is repeated every T=10 seconds, producing a potency of over 100, *i.e.*, one request from the attacker results in over 100 legitimate requests being denied service.

8. Related Work

To our knowledge, this work is the first to investigate the adversarial exploitation of network dynamics for the purpose of reducing one or more aspects of service quality or of efficiency. Clearly, there is a huge literature on many other forms of adversarial attacks, which range from attacks that compromise security (*e.g.*, Trojan horse attacks) to those that target system availability (*e.g.*, DoS attacks). In this section, we put our work in context by comparing it to existing work, which while not explicitly looking at exploiting the transients of adaptation, are nevertheless related.

Modeling of Network Dynamics: There have been a number of recent works that have focused on the study of network dynamics using an arsenal of tools and techniques. Most of these studies have concentrated on the negative impact of interference by non-adversarial processes. Examples include works in which the negative impact of non-congestion packet losses (*e.g.*, due to wireless links) is evaluated [2] and new adaptation strategies to circumvent such impact (*e.g.*, in large bandwidth-delay product networks) are proposed [4, 12]. While we have employed some of the same analytical tools and techniques to pursue our study of RoQ attacks, our work is the first to consider the use of such techniques in assessing vulnerabilities due to adversarial processes.

Dealing With Misbehaving Flows: There is a large body of literature that looked at the impact of misbehaving flows and on detection and policing techniques thereof [25]. Misbehaving flows are loosely defined as those who do not adapt their sending rate in a manner consistent with an established norm—in particular TCP friendliness [15]. In this body of literature, the basic assumption is that the goal of misbehaving flows is to get more than their fair share of bandwidth, typically by blasting packets at some constant high rate. Indeed, many of the detection techniques proposed in the literature are based on that assumption—*e.g.*, by proposing efficient data structures for the identification of high-bandwidth flows [9]. While not friendly, misbehaving flows that merely attempt to get more than their fair share of a resource are not adversarial, in the sense that they are not aiming primarily at reducing the efficiency or service quality of the resource. Resources targeted by such flows could well be operating efficiently, delivering acceptable service quality to other flows (*e.g.*, low delay and jitter). In this paper, our focus was on adversarial flows whose sole purpose is precisely to disturb these aspects of the resource's operation, while preferably being "under the radar" as far as their fair use of the resource. Indeed, the results we show in Section 4 suggest that a flow consuming only its fair share of bandwidth could be quite adversarial in its impact.

RoQ versus DoS: DoS attacks [6, 5] and its many variants [7] could be characterized as targeting one dimension of a system's service quality–namely, its availability. There are a number of papers that classify various forms of DoS attacks; examples include [18, 27, 26]. Using our model, DoS attacks could be classified as RoQ attacks with an infinite aggressiveness index (defined in Section 3), which imply that the attacker's ultimate goal is to maximize the damage at any cost. In this paper, we have focused on attacks whose perpetrators are not focused on denying access (*i.e.*, targeting availability), but rather they are focused on bleeding the system of its capacity, or simply pushing it to operate in inefficient operating regions to reduce some aspect of service quality. More importantly, in this paper, we have focused on the harder-to-detect, low-intensity attacks, *i.e.*, with modest aggressiveness compared to the aggressiveness required for DoS attacks.

RoQ versus the "Shrew": The "Shrew" attack proposed in [22] is an example of a low-intensity, harder to detect attack which is targeted at a subset of flows going through a network link, with the intension of shutting off these flows by synchronizing the attack traffic in such a way to cause these flows to perpetually timeout. Shrew attacks could be viewed as RoQ attacks which target service quality extended to a *specific set of flows*. As explained in [22] Shrew attacks could result in shutting off targeted flows without much of an effect on the link's utilization. In this paper, we have focused on RoQ attacks that do not target a specific subset of flows, but rather target the network element itself.

9. Conclusion

This paper highlights the importance of a systematic examination of the dynamics of systems and networks as possible vulnerabilities to adversarial attacks. Towards that end,

we formalized a notion of potency that exposes the trade-offs between the damage inflicted by an attacker, the cost of mounting the attack, and the willingness to pay such costs (*i.e.*, aggressiveness). We identified RoQ attacks as those attempting to maximize the marginal utility of attack traffic by optimally exploiting the transients of the underlying system adaptation mechanisms. We uncovered susceptibilities to RoQ attacks that could compromise the efficient operation and the service quality of Internet resources, by maximizing wasted bandwidth or delay jitter, for example. Using the potency metric, the extent of such RoQ exploits can be quantitatively assessed, enabling a solid basis for comparing the trustworthiness of competing designs. We used a control-theoretic model to underline the complex interplay between the efficiency-load behavior of a resource and the adaptation mechanisms of both the resource and its consumers. Our conclusions are confirmed analytically, numerically, as well as through simulations and Internet experiments. We believe this paper to be a first step towards developing a general understanding of design principles that could be adopted to protect against RoQ exploits, and applying such principles to the design and implementation of common adaptive resource management components.

References

[1] S. Athuraliya, S. Low, V. Li, and Q. Yin. REM: Active Queue Management . *IEEE Network*, 15(3):48–53, 2001.

[2] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Kartz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. In *ACM SIGCOMM 1996*, 1996.

[3] D. Bansal and H. Balakrishnan. Binomial congestion control algorithms. In *Proceedings of INFOCOM'2001*, 2001.

[4] D. Wei C. Jin and S. Low. Internet Draft: FAST TCP for high-speed long-distance networks . *draft-jwl-tcp-fast-01.txt*, 2003.

[5] CERT Coordination Center. CERT Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attacks Original issue date: September 19, 1996. http://www.cert.org/advisories/CA-1996-21.html.

[6] CERT Coordination Center. Denial of Service Attacks. http://www.cert.org/tech_tips/denial_of_service.html.

[7] CERT Coordination Center. Trends in Denial of Service Attack Technology, October 2001. http://www.cert.org/archive/pdf/DoS_trends.pdf.

[8] T. Doeppner, P. Klein, and A. Koyfman. Using router stamping to identify the source of ip packets. In *Proceedings of the 7th ACM conference on Computer and communications security*, pages 184–189. ACM Press, 2000.

[9] C. Estan and G. Varghese. New directions in traffic measurement and accounting. In *Proceedings of the ACM SIGCOMM'02*, San Francisco, CA, November 2001.

[10] E. Amir et al. UCB/LBNL/VINT Network Simulator - ns (version 2). Available at http://www.isi.edu/nsnam/ns/.

[11] P. Ferguson and D. Senie. Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. RFC 2267 available from ftp://ftp.isi.edu/in-notes/rfc2267.txt.

[12] S. Floyd. Internet Draft: HighSpeed TCP for Large Congestion Windows. *draft-ietf-tsvwg-highspeed-01.txt*, 2003.

[13] S. Floyd, R. Gummadi, and S. Shenker. Adaptive red: An algorithm for increasing the robustness of red, 2001.

[14] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[15] The PSC Networking group. The TCP-Friendly Website. http://www.psc.edu/networking/tcp_friendly.html.

[16] C. Hollot, V. Misra, D. Towsley, and W. Gong. A Control Theoretic Analysis of RED. In *Proceedings of IEEE INFOCOM 2001*, Anchorage, AL, April 2001.

[17] C. Hollot, V. Misra, D. Towsley, and W. Gong. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. In *Proceedings of IEEE INFOCOM 2001*, Anchorage, AL, April 2001.

[18] A. Hussain, J. Heidemann, and C. Papadopoulos. A framework for classifying denial of service attacks. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 99–110. ACM Press, 2003.

[19] *iproute2 and tc*. http://snafu.freedom.org/linux2.2/iproute-notes.html.

[20] F. Kelly. Mathematical Modelling of the Internet. *Mathematics Unlimited - 2001 and Beyond*, pages 685–702, 2001.

[21] F. Kelly, A. Maulloo, and D. Tan. Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability. *Journal of Operations Research Society*, 1998.

[22] A. Kuzmanovic and E. Knightly. Low-Rate TCP-Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants). In *Proceedings ACM SIGCOMM'03*, karlsruhe , Germany, August 2003.

[23] D. Lin and R. Morris. Dynamics of Random Early Detection. In *Proceedings of ACM SIGCOMM'97*, Cannes, France, September 1997.

[24] S. Low, F. Paganini, J. Wang, S. Adlakha, and J. Doyle. Dynamics of TCP/RED and a Scalable Control. In *Proceedings of IEEE INFOCOM 2002*, New York, NY, June 2002.

[25] R. Mahajan, S. Floyd, and D. Wetherall. Controlling high-bandwidth flows at the congested router. In *Proceedings of ICNP'01*, November 2001.

[26] C. Meadows. A formal framework and evaluation method for network denial of servic. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, June 1999.

[27] J. Mirkovic, J. Martin, and P. Reiher. A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms. *Technical report 020018 Computer Science Department, University of California, Los Angeles* .

[28] D. Moore, G. Voelker, and S. Savage. Inferring Internet Denial-of-Service Activity. In *Proceedings of the 2001 USENIX Security Symposium (Security)*, 2001.

[29] K. Ogata. Modern Control Engineering, 4th Ed. Prentice Hall, 2002.

[30] T. Ott, T. Lakshman, and L. Wong. SRED: Stabilized RED. In *INFOCOM 1999*, New York, USA, March 1999.

[31] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP Traceback. In *Proceedings of ACM SIGCOMM'00*, August 2000.

[32] S. Shenker. A Theoretical Analysis of Feedback Flow Control. In *Proceedings ACM SIGCOMM'90*, Philadelphia, PA, September 1990.

[33] S. Staniford, V. Paxson, and N. Weaver. How to 0wn the Internet in Your Spare Time. In *Proceedings of the 11th USENIX Security Symposium (Security)*, 2002.

[34] The Salt Lake Tribune. As forecast, worm takes SCO offline (February 2, 2004). Available from http://www.sltrib.com/2004/Feb/02022004/utah/134908.asp.

[35] M. Welsh and D. Culler. Adaptive Overload Control for Busy Internet Servers. In *Proceedings of the 4th USENIX Conference on Internet Technologies and Systems (USITS)*, March 2003.