A Chain Reaction DoS Attack on 3G Networks: Analysis and Defenses

Bo Zhao*, Caixia Chi†, Wei Gao*, Sencun Zhu* and Guohong Cao*
*Department of Computer Science and Engineering, The Pennsylvania State University
†Bell Laboratories, Alcatel-Lucent Technologies
* {bzhao,wxg139,szhu,gcao}@cse.psu.edu, † chic@alcatel-lucent.com

***Abstract*—The IP Multimedia Subsystem (IMS) is being deployed in the Third Generation (3G) networks since it supports many kinds of multimedia services. However, the security of IMS networks has not been fully examined. This paper presents a novel DoS attack against IMS. By congesting the presence service, a core service of IMS, a malicious attack can cause chained automatic reaction of the system, thus blocking all the services of IMS. Because of the low-volume nature of this attack, an attacker only needs to control several clients to paralyze an IMS network supporting one million users. To address this DoS attack, we propose an online early defense mechanism, which aims to first detect the attack, then identify the malicious clients, and finally block them. We formulate this problem as a change-point detection problem, and solve it based on the non-parametric GRSh test. Through trace-driven experiments, we demonstrate that our defense mechanism can throttle this DoS attack within a short defense time window while generating few false alarms.**

I. Introduction

The IP Multimedia Subsystem (IMS) is being deployed in the Third Generation (3G) networks since it supports many kinds of multimedia services for end users [1]. One example is the *presence service* which allows a user to be informed whether other users are online. If they are online, it provides information such as whether they are idle or busy, whether they have audio, video, instant messaging capabilities or not, etc. The presence service is viewed as an indispensable feature to help service providers generate new revenue [2]. Thus, presence services have been well supported in the IMS architecture and many telecom equipment vendors such as Nokia, Alcatel-Lucent, Nortel, Ericsson and Cisco have provided presence service products and solutions. Many service providers such as America Online, AT&T Wireless and Sprint have launched presence services to end users.

Session Initiation Protocol (SIP) [3] has been selected as the session control protocol in IMS and a presence service based on SIP is shown in Figure 1. As shown in the figure, Bob has several devices which provide information about Bob's presence. All the devices send their pieces of information to the *Presence Server* (PS). The PS gathers all the received information and obtains a complete picture of a user's presence. Figure 1 also shows how Bob's friend (Alice) obtain the presence information about Bob.

This work was supported in part by NSF CNS-0643906 and MURI/ARO W911NF-07-1-0318.

When a user, say Alice, wants to subscribe the presence service about her friends, say Bob, she needs to exchange a SUBSCRIBE transaction and a NOTIFY transaction with the PS which maintains Bob's presence information. If Alice wants to know the presence information of her friends Bob, Jeff, and Peter, she sends a SUBSCRIBE to each of their presence servers (1),(3),(5), as shown in Figure 2, and later receives a NOTIFY from each of them (7), (9), (11). It's obviously that this mechanism does not scale well if the subscriber needs to subscribe the presence service of many users. To address this problem, IETF creates a new function called *Resource List Server* (RLS) which is used to reduce the number of messages that the subscriber has to send and receive as shown in Figure 3.

Although RLS can reduce the message overhead, we discover that the presence service still can generate a high volume of signaling traffic, as one subscription request can result in multiple SUBSCRIBE and NOTIFY messages to the users whose presence information is subscribed. Furthermore, we find that the use of RLS introduces DoS attacks to PS. For example, a malicious attacker can compromise a client whose user list consists of hundreds of users. This subscription request will then cause the RLS to send hundreds of messages to the subscribed PSs. As long as the attacker can compromise a number of presence service clients, it can launch distributed DoS attacks to the PS and congest it. In addition, the IMS network is more vulnerable to DoS attacks, as the IMS inherits the DoS attack vulnerability from the Internet and the users of IMS network are easy to be compromised [4].

In IMS, many services such as presence service, VoIP service, Push-to-Talk services all go through the Call Session Control Function (CSCF) for routing or charging purpose. When the PS suffers from DoS attacks, many normal traffic will be delayed, which in turn will result in many message retransmissions according to the retransmission mechanism in SIP. For example, each SIP request could be re-transmitted ten times if no response is received within 32 s [3]. The retransmitted benign traffic and malicious traffic could together congest the CSCF, and thus other IMS services that have to go through the CSCF will be disabled. We demonstrate that this DoS attack only requires several compromised clients with low-volume traffic to crush an IMS network supporting one million users in a metropolitan area such as Washington D.C. and New York City. Note that once IMS is out of service

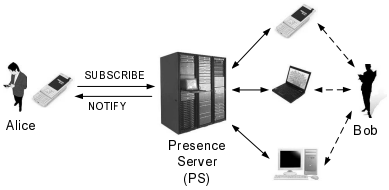

Fig. 1. SIP Presence Architecture

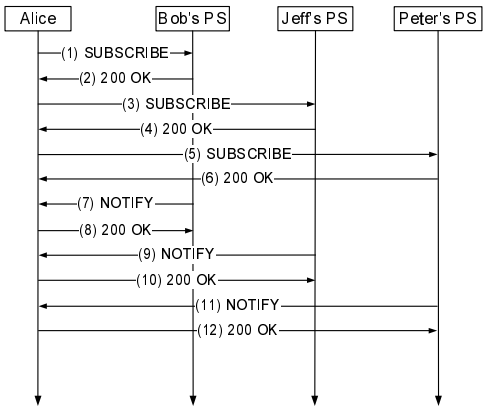

Fig. 2. Long Presence List without a RLS

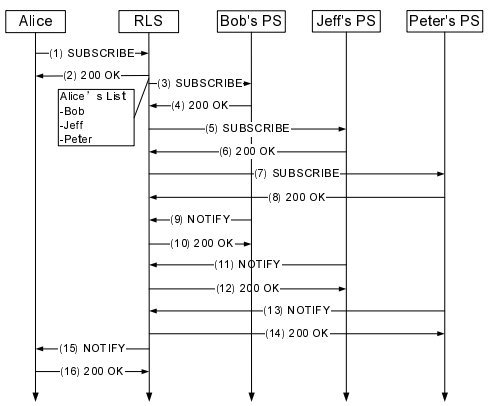

Fig. 3. Resource List through a RLS

by the DoS attack, most data services and all the multimedia services of a 3G network will be blocked.

To address this DoS attack, we have to stop it before the CSCF is congested by the PS traffic load. We formulate online early detection as a change-point detection problem, which aims to detect when the system is under attack. This is achieved by monitoring the system resource usage. We can further identify the sources of this DoS attack by monitoring any abnormal behavior of a user. Our detection mechanism is based on the non-parametric *GRSh test* [5]. Using trace-driven simulation, we demonstrate that our defense mechanism can stop this DoS attack, and it has a short detection time and a low false alarm rate.

The remainder of this paper is organized as follows: Section II presents the attacks against IMS and explains how malicious clients deny all the IMS services by exploiting them. Section III offers defense mechanism to defend these attacks before they cause any major damage. In Section IV, we evaluate the effectiveness of our detection mechanism. Section V discusses related works. Section VI presents concluding remarks.

II. The DoS Attack against IMS

This section describes this chain reaction DoS attack, and shows how it may block all services of IMS. To simplify the presentation, we divide the entire attack process into two attacks, namely the first attack and the second attack, where the first attack automatically triggers the second one. In Figure 4, the first attack corresponds to the message type (1), the malicious traffic from clients to a presence server (PS); the second attack is indeed caused by benign traffic, including those from benign clients to PS (message type (2)) and those to other servers (message type (3)).

A. The Analysis of the First Attack

This section will illustrate how RLSs can be exploited by malicious clients to launch the first attack in order to congest a PS as well as the CSCF, and evaluate the impact and efficiency of the first attack.

As shown in Figure 3, the RLS is designed to decrease the traffic for presence service, but it may also multiply the traffic by many times. Even when a client only establishes one session to RLS, RLS has to establish with the remote PS as many sessions as the number of users included in the list. Thus, the malicious clients can use RLSs to inject many more messages to a remote PS by using low-volume traffic from RLSs located in multiple networks, as shown in Figure 5. Although the PS capacity may be more than that of the RLS, the attacker can use many RLSs of other networks to attack the PS of one network. Moreover, when RLSs do not find presence information from the PS of its own network, they will send requests to the PS of another network.

Next we calculate the number of malicious clients needed to paralyze a PS. This calculation requires the knowledge of user behaviors and network deployment specifications. However, because IMS was deployed only recently, even the telecom equipment vendors do not have this information. As such, in this paper, we rely on the existing user behavior model of the presence service [6] to demonstrate the efficiency of this attack. We assume that in an IMS network, a presence service user generates at least 2 messages per-hour ($M_{min} = 2$) and at most 8 messages per-hour ($M_{max} = 8$) when the user is online. The total number of users is $N = 10^6$, and the peak traffic load of a PS is estimated as $P = 8/3600 * N$. Suppose 80% of the PS capacity is occupied in the peak time, that is, $C = 8/3600 * N/80\%$ is the server capacity. The extra message rate to block a PS is:

$$\begin{aligned} T_{mal} &= C - M_{min} \times N \\ &= (8/3600/80\% - 2/3600) \times 10^6 = 2222/s. \end{aligned}$$

T_{mal} is the number of malicious messages sent to a PS per second in order to congest it.

Suppose a malicious client constructs a resource list with N_l contacts, and it sends a subscription request to RLS every I_{mal} seconds. Suppose one session between the client and RLS triggers at least $N_l * 8$ messages to the PS. When $N_l = 100$ and $I_{mal} = 5$ s. the total number of malicious clients to be

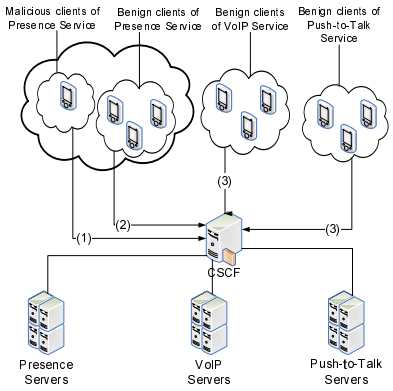

Fig. 4. The chain reaction DoS attack

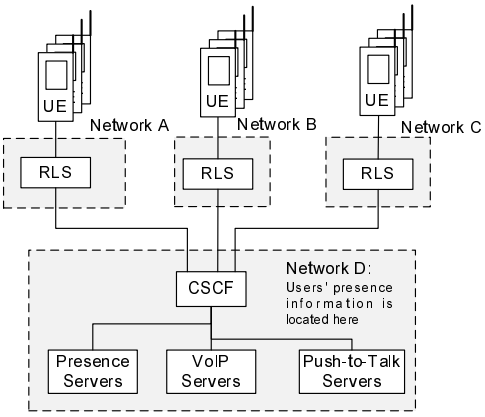

Fig. 5. The compromised malicious clients use multiple RLSs of different networks to attack one PS

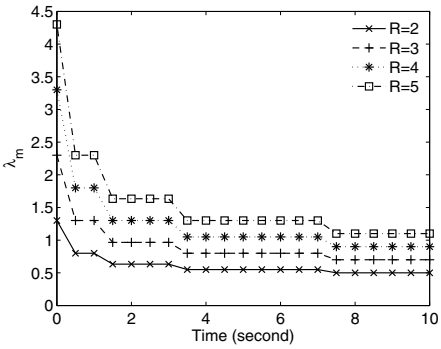

Fig. 6. The traffic rate λ_m needed to congest a CSCF in time t

compromised to congest the PS is:

$$N_m = T_{mal}/(N_l * 8/I_{mal}) = 2222/(100 * 8/5) = 13.9.$$

From these numbers, we can see that this is a powerful low-volume low rate DoS attack. In a metropolitan area, such as Washington D.C. and New York City, where a presence service has one million users, the attacker only needs 14 such clients to block the presence service of the IMS network. Note that the peak time of an IMS network is usually known, e.g., the morning time when people begin to work, so the attackers can easily attack the PS at the peak time.

B. The Analysis of the Second Attack

1) Attack description: The second attack exploits the time-out mechanism of the SIP protocol. In the SIP protocol [3], after a User Agent Client (UAC) sends out the request message to the User Agent Server (UAS), it sets timer F to fire in $64 * T1$ seconds and another timer E to fire in $T1$ seconds. Every time when the timer E fires, a request is retransmitted. If timer E fires while UAC still has not received the response, the timer will be reset to a value of $MIN(2 * T1, T2)$. When the timer fires again, it is reset to $MIN(4 * T1, T2)$. This process continues so that retransmissions occur at an exponentially increasing interval that caps at $T2$. The default value of $T1$ is 500 ms. The default value of $T2$ is $4s$ and it represents the maximum response delay for a transaction. Under the default values of $T1$ and $T2$, this results in intervals of 500 ms, $1s$, $2s$, $4s$, $4s$, etc. If timer F fires but the UAC still has not received the response, this transaction should be terminated. Thus, the request of UAC is retransmitted 10 times during $32s$ $(64 * T1 = 32s)$. Therefore when the PS is congested by the first attack, all requests to this PS will be retransmitted 10 times, and the CSCF has to handle 10 times more requests.

When the CSCF becomes very busy, the benign clients of other application services (such as VoIP and push-to-talk services) may not receive responses from the CSCF and could also generate 10 times of retransmission requests to the CSCF in the next $32s$ [3]. Consequently, all the clients of IMS, malicious or benign, effectively participate in the DoS attacks on the CSCF, making the CSCF fully congested. After that, the requests of all the services through CSCF will be timed out and denied.

2) Traffic model of PS after it is under attack : Next we model the traffic load change of the PS after it is under attack. We will show that even if some spare system resource is available, the DoS attack is still very significant. This analysis will provide us some guidance for designing an effective defense mechanism.

Let λ_m be the total traffic rate of malicious clients to the PS. Let λ_b be the traffic rate of benign clients to the PS, and $\lambda_P = \lambda_m + \lambda_b$ is the traffic rate of all the clients to the PS. The capacity of the PS is denoted as C_P and the capacity of CSCF is denoted as C_C, which should be R times larger than that of PS, that is, $C_C = R \times C_P$. Let r be the capacity utilization rate of a PS, and it is usually set to $\lambda = 70\% C_P$ to accommodate traffic spikes resulted by benign users. Without loss of generality, we normalize $\lambda_m, \lambda_b, \lambda_P$ over C_P. Thus, $\lambda_m = 1$ indicates that the traffic rate of malicious clients is just high enough to occupy the capacity of the PS, and $\lambda_m = k$ means the rate of kC_P. Set the time when malicious clients congested PS as $t = 0$.

The problem is:

Given C_C, C_P, what is the relationship between λ_m and t? To congest PS, there must be $\lambda_P > C_P$, that is $\lambda_m > C_P - \lambda_b$. When PS is congested at $t = 0$ s, the traffic rate to be retransmitted at $t = 0.5$ s is $\lambda_r = \lambda_m + \lambda_b - C_P$. Then the total traffic rate to CSCF and PS at $t = 0.5s$ is:

$\lambda_P(t) = \lambda_m + \lambda_b + \lambda_r$.

Suppose that initial requests to a PS has priority over the retransmitted traffic, then there will be another λ_r newly arrived traffic rejected and retransmitted at the next time.

Set $T = \{(0, 0.5), (0.5, 1), (1, 1.5), \cdots (31.5, 32)\}$. The traffic, which arrived at the time interval $(0, 0.5)$ but received no responses, will be retransmitted at time interval $(0.5, 1)$. If the retransmitted traffic still does not receive any response, it will be retransmitted at time interval $(1.5, 2)$,

$(3.5, 4)$, $(7.5, 8)$, $(11.5, 12)$, $(15.5, 16)$, $(19.5, 20)$, $(23.5, 24)$, $(27.5, 28)$, $(31.5, 32)$. Similarly, for the new traffic arriving at time $(t - 0.5, t), \forall t \in [0.5, 32]$, it will be retransmitted at $(t, t + 0.5)$,$(t + 1, t + 1.5)$,$(t + 3, t + 3.5)$,$(t + 7, t + 7.5)$....

The following gives the total traffic rate of a PS:

$$\lambda_P(t) = \begin{cases} \lambda_m + \lambda_b & 0 \leq t < 0.5 \\ \lambda_m + \lambda_b + \lambda_r & 0.5 \leq t < 1.5 \\ \lambda_m + \lambda_b + 2\lambda_r & 1.5 \leq t < 3.5 \\ \lambda_m + \lambda_b + & \\ (\lfloor (t - 3.5)/4 \rfloor + 3)\lambda_r & 3.5 \leq t < 31.5 \\ \lambda_m + \lambda_b + 10\lambda_r & 31.5 \leq t \end{cases}$$

With $\lambda_P(t) = C_C$ and $\lambda_r = \lambda_m + \lambda_b - C_P$, we have $\lambda_m(t)$, the rate of total traffic generated by malicious clients to fill up C_C in order to congest the CSCF, as a function of time t.

$$\lambda_m(t) = \begin{cases} C_C - \lambda_b & 0 \leq t < 0.5 \\ (C_C + C_P)/2 - \lambda_b & 0.5 \leq t < 1.5 \\ (C_C + 2C_P)/3 - \lambda_b & 1.5 \leq t < 3.5 \\ \frac{C_C + (\lfloor (t-3.5)/4 \rfloor + 3)C_P}{\lfloor (t-3.5)/4 \rfloor + 4} - \lambda_b & 3.5 \leq t < 31.5 \\ (C_C + 10C_P)/11 - \lambda_b & 31.5 \leq t \end{cases}$$

With $r = 70\%$, $C_P = 1$ and different values of R, we plot the relationship between λ_m and t in Figure 6. It indicates that the traffic rate of malicious clients required to congest the CSCF decreases dramatically with time within the first 4 seconds. After that, the required malicious traffic does not vary much. Therefore, after balancing the effectiveness of malicious traffic and the delay t to congest the CSCF, $\lambda_m(4)$ looks a good point for the attacker. On the other hand, this also indicates that the length of the time window of a defense mechanism is 4 seconds. In addition, Figure 6 shows by increasing the capacity of CSCF, the malicious client also need to increase its data rate to congest the CSCF.

III. Defense Mechanism

In this section, we aim to design an efficient mechanism to defend against the DoS attack at the earliest stage. To achieve this goal, an extension of the standard Girshick-Rubin-Shiryaev (GRSh) algorithm is used to detect the change-points in the stochastic characteristics of system resources and user behaviors. Generally speaking, the DoS attack is done through several compromised malicious clients, each of which is assigned a unique user ID in telecom network. They subscribe to the users on their lists to generate malicious traffic. The users are considered as *explored users* if they are in the malicious clients' lists, e.g. Bob, Jeff and Peter in Figure 3. Our detection mechanism includes three stages: *detection*, *identification* and *blocking*.

In the detection stage, we monitor the system resources such as bandwidth, traffic load, CPU usage in real time, e.g., every $0.2s$. In this paper, we only select CPU usage as the system resource to monitor and measure. Let the measured data sequence be $X = \{x_1, x_2, ..., x_n\}$, which is considered as a random process. GRSh is then used to detect the change-point of the stochastic characteristics of X, which is defined in Section III-A. If a change-point is detected, we know that the DoS attack has happened.

In the identification stage, we first identify who are the explored users and obtain their IDs, then group them by the RLSs of remote networks where the traffic of those IDs comes from, and finally send each group information to its corresponding RLS. Specifically, we first identify the set of users who accessed a PS recently, e.g., within the past 60 seconds, and get the time interval sequence for each user in the set. Each value in the sequence is the arrival time interval between two subscription requests to the PS for the user. This sequence is denoted as $Y = \{y_1, y_2, ..., y_n\}$. Given Y, GRSh is used to detect any change point, and a user is highly likely to be an explored user if a change-point exists in his sequence Y. The false positive probability of such identification is investigated through experiments in Section IV-D.

In the blocking stage, after we identify the explored users, the remote RLSs will be responsible for finding out the malicious clients who subscribed to those IDs. Finally, the authentication and authorization module of IMS will block these malicious clients. This stage is relatively straightforward and is not the focus of this paper.

Since the DoS attack could exhaust the system resources of a PS, the defense mechanism should be executed on a machine which does not share any resource with the PS. Further, in PS, a very high priority process records and transfers the data to a remote machine. Hence, even if the PS is totally congested or out of service because of the attack, our detection mechanism still gets the data and works to find out when this attack happens and who generates these malicious traffic.

Another simple fast detection solution is to add a rate-limitation service directly into the RLS. However, since only with rate information of a client on RLS, we can not confirm if this client is attacking our system, The high transmission rate of a client could be caused by various reasons, such as unstable connection, the interference or user's abnormal behaviors. Thus we do not consider this solution in this paper.

Next, we first give an overview of the GRSh method, followed by the details of our detection and identification algorithms.

A. Overview of the GRSh Method

The sequential change-point detection problem GRSh method [5] is defined as follows. Suppose that $X = \{x_1, x_2, ...\}$ is a sequentially observed sequence of independent random variables with the following probability density function

$$f(x_i) = \begin{cases} f_0(x_i) & 1 \leq i \leq m \\ f_1(x_i) & m \leq i \end{cases} \quad (1)$$

where $f_0(.) \neq f_1(.)$ and m is the change-point, i.e., the probabilistic characteristics of X changes at m. $f_0(.)$ is the probability density function before the change-point happens, while $f_1(.)$ is the probability density function after the change-point happens. Let

$$
\begin{aligned}
w_i &= \frac{f_1(x_i)}{f_0(x_i)}, i \geq 1, \\
W_i &= w_i(1 + W_{i-1}) = w_i + w_i w_{i-1} \\
&\quad +... + w_i w_{i-1} ... w_1 \\
W_0 &= 0
\end{aligned}
$$

then the change-point is detected at W_n when $W_n > g$, where g is the detection threshold.

Basically, GRSh captures the change-points by comparing the values of $f_0(x_i)$ and $f_1(x_i)$ for observations x_i, and measuring the differences. Therefore, w_i can be defined as the *step-wise difference indicator*, and W_i as the *cumulative difference indicator*. Since $1 + W_{i-1} \geq 1$, the calculation of each W_i can be considered as a combination of current and past value differences between $f_0(\cdot)$ and $f_1(\cdot)$ on the observations. Note that, due to the randomness of observations, normally the GRSh method will not detect the change-point immediately after it happens. That is, the "detection point", denoted as n, will be $n > m$. Clearly, smaller g is helpful to reduce the detection delay $n-m$, but it also increases the false-positive ratio of change-point detection [7]. The choice of g is therefore a tradeoff between false alarm rate and detection delay, which is going to be evaluated in Section IV.

B. Detection Mechanism

In our detection mechanism based on GRSh, we need to provide appropriate $f_0(x)$ and $f_1(x)$. In Figure 8, we compare the probability mass function of the call rates in a real trace with that of a Poisson distribution with the mean 17.7683. We find that our real trace follows the Poisson distribution very well. Suppose each call consumes the same amount of system resources, then this call rate distribution also makes the PS CPU usage follow a Poisson distribution with mean λ. Therefore, we have:

$$f_0(x) = \frac{\lambda^x e^{-\lambda}}{x!} \tag{2}$$

For $f_1(x)$, we do not have the trace to get the probability mass function of the CPU usage when PS is under attack. Even if the call rate of the malicious traffic is constant, the increasing trend of CPU usage of the PS also depends on the implementation of the PS, such as the policy and size of its queue, which could vary among different equipment vendors. Thus, we decide to choose the Pareto distribution for $f_1(x)$. Pareto distribution was originally used to describe the allocation of wealth among individuals where a larger portion of the wealth of any society is owned by a smaller percentage of the people in that society. Here we adopt Pareto distribution because the larger the value of CPU usage is, the higher possibility that this system is under DoS attack. At the attacking time, the value of CPU usage in most time is distributed among a small range near 100%. So we assume $101 - x$ (x is the value of CPU usage) follows a Pareto distribution when PS is under DoS attack, and the minimum possible value of $101 - x$ is 1. λ is the mean of $f_0(x)$.

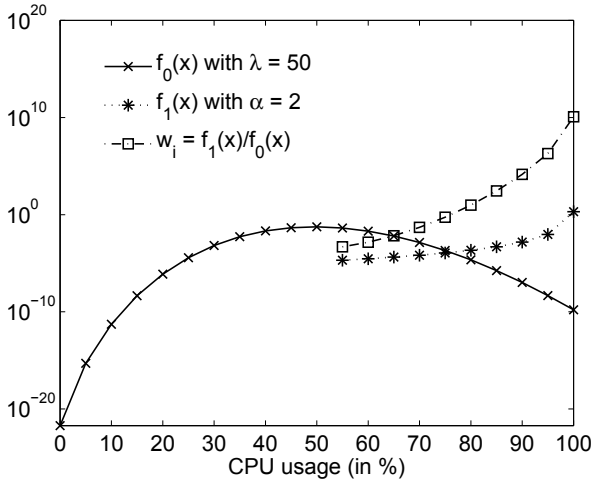

Fig. 7. The change of $f_0(x)$, $f_1(x)$ and w_i with the increase of CPU usage

$$f_1(x) = \begin{cases} \alpha/(101-x)^{\alpha+1} & \lambda < x \leq 100 \\ 0 & 0 \leq x \leq \lambda \end{cases} \tag{3}$$

Based on Eq. (2) and (3), w_i can be calculated as follows:

$$w_i = \begin{cases} 0 & 0 \leq x \leq \lambda \\ \frac{x!\alpha}{(101-x)^{\alpha+1}\lambda^x e^{-\lambda}} & \lambda < x \leq 100 \end{cases} \tag{4}$$

The tunable parameter α, pareto index, is a measure of the breadth of the distribution. It impacts how dramatically the corresponding $f_1(x)$ increases when the CPU usage increases. We will find out the proper value of α in our experiment.

When the system CPU usage is below $\lambda\%$, we do not care if it is under attack. When our system is under a severe DoS attack, the CPU usage should rapidly pass $\lambda\%$. Note that when x is much smaller than λ, $f_0(x)$ in a Poisson distribution is much smaller than $f_1(x)$ in a Pareto distribution under the same x. Accordingly, w_i would be very large. To reduce false alarms, we set $f_1(x)$ to 0. That is why we have two cases based on λ in Eq.(3).

Even if $f_1(x)$ does not match the real probability mass function of the CPU usage when PS is under attack, our detection mechanism still works. The reason is that with the increase of CPU usage, $f_1(x)$ becomes much larger, whereas $f_0(x)$ becomes much smaller. In the worst case, when CPU usage increases to 100%, $w_i = \frac{f_1(x_i)}{f_0(x_i)}$ becomes huge. Thus, $W_i = w_i(1+W_{i-1})$ will easily exceed g, as shown in Figure 7 (note that in this figure the y axis is in the power scale).

Algorithm 1 shows the details of using GRSh to detect attacks.

C. Identification Mechanism

In the identification phase, given the time interval sequence of each user, $Y = \{y_1, y_2, ...\}$, we want to find if this user is an explored user, that is, if the subscription rate to this user has a dramatic increase. When there is no attack, user subscriptions often follow a Poisson process. Therefore, we can assume

Algorithm 1 : Detecting attacks

Get the α and g_1 for this system
Set $W_{last} = 0$
for each time unit **do**
  Set x = current CPU usage.
  Get $w = \frac{f_1(x)}{f_0(x)}$ with Eq. (4)
  Get $W_{curr} = w(1 + W_{last})$
  if $W_{curr} > g_1$ **then**
    System under attack, go to Algorithm 2
  else
    $W_{last} = W_{curr}$
  end if
end for

$f_0(y)$ of the sequence follows an exponential distribution and derive the mean λ from this sequence.

$$f_0(y) = \frac{1}{\lambda} e^{-y/\lambda} \tag{5}$$

When the PS is under attack, the smaller y/λ is, the higher possibility that this user is explored. Here our detection is based on y/λ instead of y because we are more concerned with the relative change of the subscription rates. In this case, assuming y/λ follows a Pareto distribution, we have:

$$f_1(y) = \begin{cases} \lambda\beta/y^{\beta+1} & 0 \le y \le \lambda \\ 0 & \lambda < y \end{cases} \tag{6}$$

Note that our detection mechanism should still work even if $f_1(y)$ does not match the actual one when PS is under attack. The reason is that with the decrease of interval y, $f_1(y)$ becomes much larger, whereas $f_0(y)$ becomes much smaller. So $w_i = \frac{f_1(y_i)}{f_0(y_i)}$ increases, and $W_i = w_i(1 + W_{i-1})$ keeps increasing until $W_i > g$. In our experiment, we will demonstrate that our $f_1(y)$ works well even if the real intervals change randomly when the PS is under attack.

Based on Eq. (5) and (6), w_i is calculated as follows:

$$w_i = \begin{cases} 0 & \lambda < y \\ \frac{\lambda^2\beta}{e^{-y/\lambda} y^{\beta+1}} & 0 \le y \le \lambda \end{cases} \tag{7}$$

The meaning of the tunable parameter β is the same as α. When the call interval of a client is larger than λ seconds, we do not care if it is an exploited client, since it cannot inject a significant number of traffic into the system. In Section IV, we will investigate the values of α and β to detect and identify attacks earlier with low false positive rate. Note that when the system is under DoS attack, W_i values should increase continuously; otherwise, they should keep stable. Algorithm 2 shows the details of our identification scheme.

Algorithm 2 : Identify the exploited users

Get the set S of users who send messages in the past
for each user in S **do**
  Get time interval sequence of establishing sessions to the PS of this client $Y = \{y_1, y_2, ..., y_n\}$
  Get the λ, β and g_2 for this user
  Set $W_{last} = 0$,
  for each y_i of Y **do**
    Set y = y_i,
    Get $w = \frac{f_1(y)}{f_0(y)}$ with Eq. (7)
    Get $W_{curr} = w(1 + W_{last})$
    if $W_{curr} > g_2$ **then**
      "malicious traffic" $\leftarrow$ true
      record user ID and break
    end if
    $W_{last} = W_{curr}$
  end for
end for
if "malicious traffic" == true **then**
  send identified users' IDs to RLSs in order to find out and forbid the malicious clients.
end if

IV. Performance Evaluation

In this section, we evaluate our detection mechanism against the DoS attack based on our test bed and trace-driven simulation. Our following evaluation focuses on the real traces collected from SIP traffic traces captured in an operational network providing a commercial presence service.

A. Test Bed

We use Open SIP Express Router version 1.2.2 (OpenSER) [8] and SIPp version 3.0 [9] to build our test bed. The OpenSER is a freely-available, open source SIP proxy server. It is configured to act as a CSCF. SIPp is a free Open Source test tool and traffic generator for the SIP protocol. It may be configured to act as PS, VoIP and clients. It is also modified to realize the functions of a RLS. Each of PS and CSCF is installed on a machine with an Intel(R) Pentium(R) 2.60GHz (4) CPU, 1GB memory , and Linux OS version 2.6.18-8.el5PAE. Clients are installed on several different machines. Because we do not evaluate the performance of clients, the configuration of their machines is not mentioned here. The test bed works in the same way as depicted in Figure 4.

B. Overview of Metrics

In evaluating an intrusion-detection system, three fundamental metrics are often considered: (1) false positive ratio: the number of times when the system is mistakenly identified as under attack *over* all the time (in the detection mechanism), or the fraction of benign clients that are mistakenly identified as malicious *over* all the observed clients (in the identification scheme); (2) false negative ratio: the fraction of the time when the system is under attack but undetected over all the time, or the fraction of explored users that are not identified over all the observed clients; (3) detection time: the delay of Algorithm 1 to detect the attack after the chain reaction starts, or the delay of Algorithm 2 to identify an explored user.

Here, we do not consider the false negative ratio. The GRSh decision depends only on the additional load due to the attack, regardless of what the attack strategy is. If there exists a false negative, it implies that the traffic load is within an acceptable level. In other words, the attack only has little impact. Hence, our following evaluation only focuses on the false positive ratio and the detection time. Because our detection mechanism is composed of two algorithms, we first evaluate "Algorithm 1 : Detecting attacks" in section IV-C, followed by the evaluation of "Algorithm 2 : Identifying who are exploited users" in section IV-D.

C. Performance Evaluation of Algorithm 1

1) Experiment Setup: The total call rate of the benign clients group follows the probability mass function provided by the real trace in Figure 8. Its mean is 17.8 calls/s. Because the duration of our trace is only 33 minutes (*15:59:02-16:32:16*), we can not get the call duration from this trace. So we set the mean call duration of benign clients as 1800 s and it follows an exponential distribution. To warm up the system to get stable measurement data and to evaluate the performance of Algorithm 1 among a whole day, we build a 24-hour trace by repeating this 33-minute trace.

2) False Positive Ratio vs. Different Tunable Parameters: In this section, we analyze the false positive ratio of our detection mechanism at different values of α and detection threshold g. For a given 24-hour trace, we consider the parameter $\alpha = 1, 2, 3$ and 4. We then compute the false positive ratio of our detection mechanism as a function of g.

Figure 9 illustrates the false positive ratio versus different values of g, together with the values of α. In general, at most times, the benign clients generate stable traffic. Thus, the false positive ratio in most cases is very small. For instance, when $\alpha \geq 2$, the false positive ratio is less than 0.11353% for $g \geq 10^3$.

3) Detection Time vs. Different Tunable Parameters: To assess the detection time, we add malicious clients to generate low-volume packets to RLSs in order to trigger a huge traffic to overload the PS. We randomly choose the time to attack. We repeat the experiment 10 times with different times, and the measured detection time is averaged over the experimental instances.

Figure 10 illustrates the detection time versus different values of g. For larger values of α and g, the detection mechanism requires longer time to detect if it is under attack, but in the meantime it produces fewer false positives (see Figure 9). This suggests the trade-off between different combinations of α and g.

4) Evaluation for defending against the first attack: We now evaluate our detection mechanism for defending against the first attack. Since the false positive ratio is low according to our previous evaluation, we fix $\alpha = 2$ and $g = 10^3$.

With fixed α and g, we study detection time vs. traffic rate of malicious clients. As they become more aggressive by generating more traffic load, we need less time to detect this first attack, but the defense time window becomes smaller. The detection time should always be smaller than the time window so that our second level detection mechanism can have enough time to identify the explored users and further find out the malicious clients. In this experiment, we want to find how much time our detection mechanism leaves for Algorithm 2. With 70% CPU usage of the PS used by benign clients, Figure 11 illustrates that our detection time is very short, comparing to the defense time window. However, when the traffic rate of malicious clients increases to $\lambda_m \geq 1.75$, the time left to the identification stage is less than $1s$. Further, when the $\lambda_m \geq 2.5$, our system has no time to identify the explored users.

In this worst situation, we should use traffic overload control techniques [10] [11] to reduce the traffic load. Then our defense mechanism has more time to identify who are the attackers and block them to stop the attack traffic. After our system recovers from this attack, it can respond to retransmission requests of benign clients. Finally all the clients impacted by this DoS attack can get services.

D. Performance Evaluation of Algorithm 2

1) Experiment Setup: Algorithm 2 analyzes the session time interval of each individual user to find out who is the explored user. For calculating the parameter β for the change-point detection defined in Eq. (7), from a time interval sequence $Y = \{y_1, y_2, ..., y_n\}$, we simply extract a sub-sequence of Y, named $Y_s = \{y_{n-k}, y_{n-k+1}, ..., y_{n-l}\}$ $(n > k > l)$. Such sub-sequence represents the past history of time intervals. When the system is under attack, the current time intervals decrease dramatically compared to that in the past.

Because the duration of our real trace is only 33-minute, we can not get the session time interval sequences for all the users. However, this does not limit our experiment, as the change-point in a session time interval sequence is detected through the relative changes of the intervals before and after the change-point, which is irrelevant to the interval values themselves. Thus, the mean of session time interval before the change-point is set to $\lambda = 1800s$ following the exponential distribution.

In order to indicate that our Algorithm 2 can detect explored users with low false positive ratio and short detection time no matter how their session time intervals are changed, for a given Y, we analyze the effects of different value of β and g. In our experiment, $k = min(100, n)$ and $l = 10$.

2) False Positive Ratio vs. Different Tunable Parameters: In this section, we analyze the false positive ratio of our detection mechanism with several values: $\beta = 1$, 2, 3 and 4. We first generate a Y of 24-hour trace of a user whose mean λ of session time interval is $1800s$. We then compute the false positive ratio versus different values of g from the simulation by using our identification mechanism. We repeat the experiment 10 times with different seeds to generate Y and the measured false positive ratio is averaged.

Figure 12 illustrates the false positive ratio versus different values of g, together with the values of β. In general, at most times, benign clients generate stable traffic. Thus, the false

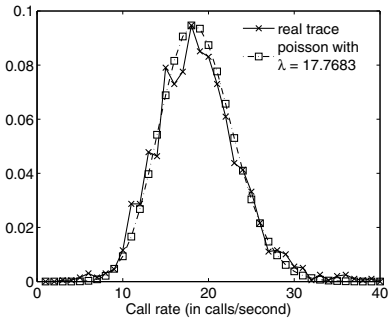

Fig. 8. The probability mass function of the call rates of real trace and that of Poisson distribution with $\lambda = 17.7683$

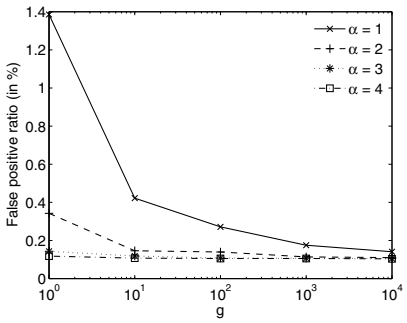

Fig. 9. Alg 1: False positive ratio vs. g

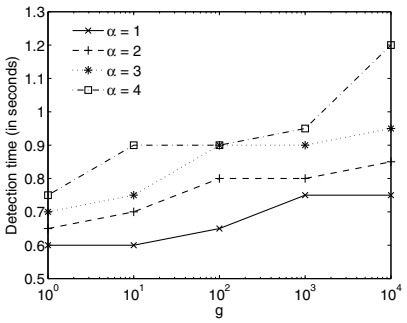

Fig. 10. Alg 1: Detection time vs. g

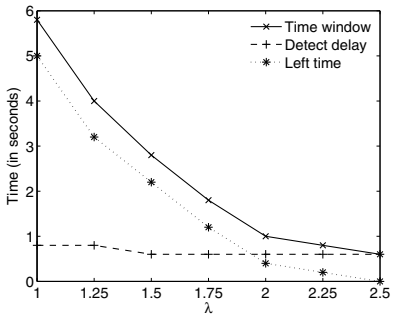

Fig. 11. Time window and the time of defense mechanism for Alg 2

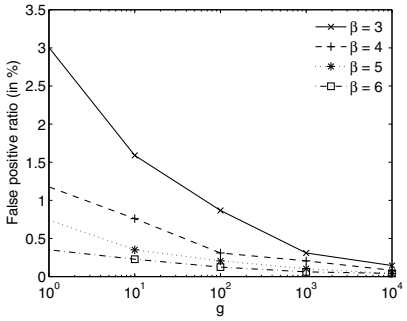

Fig. 12. Alg 2: False positive ratio vs. g

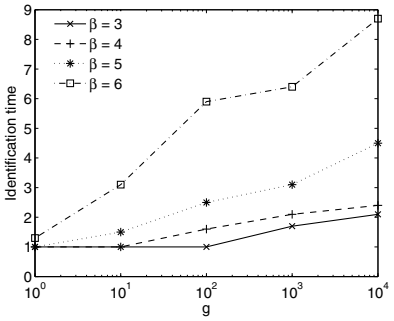

Fig. 13. Alg 2: Identification time vs. g

positive ratio in most cases is very small. For instance, when $\beta \geq 4$, the false positive ratio is less than 0.3095% for $g \geq 10^2$.

3) Identification Time vs. Different Tunable Parameters: The identification time is the number of session intervals we need to identify if this user is explored. To assess the identification time for each user, we make explored users establish sessions in random intervals to illustrate that our $f_1(y)$ following Pareto distribution is good enough to detect it, no matter what the actual sequence of attack intervals is. We randomly choose the time to attack and repeat the experiment 10 times with different times, and the measured detection time is averaged over the experimental instances.

Figure 13 illustrates the identification time versus different values of g. For larger values of β and g, the identification mechanism requires longer identification time to identify the client, but in the meantime it produces fewer false positives (see Figure 12). This suggests $\beta = 4$ and $g = 10^2$ is a good trade-off combination.

V. Related Work

In this section, we review the related work on DoS attacks and their defense mechanisms, especially in the context of 3G networks.

Recently the vulnerabilities of telecom networks get much attentions. Traynor *et al* [12] identified and proposed solutions to eliminate or extensively mitigate the vulnerabilities of SMS services. Based on social relationships between mobiles, Zhu *et al* [13] proposed a counter-mechanism to contain the propagation of mobile worms at the earliest stage.

The IMS of 3G telecom networks recently began to attract much attention on its security issues, as it bears the kind of vulnerabilities inherent to Internet Protocol-based solutions, indicating it is also subject to attacks like worms, viruses, DoS, and so on. 3GPP2 [14] and 3GPP [15] proposed solutions for IP-based services of IMS. Researchers at Bell Labs [16] examined the threats and vulnerabilities of IMS implementations. Several companies also provided their security solutions for IMS in 3G networks [17] [18]. However, these works do not consider how to protect the IMS network from DoS attacks. When IETF designed the SIP protocols [3] [19] [20] [21] that are used in IMS networks, it studied the DoS attack problems for each protocol. However, it neither considered the vulnerabilities of their protocols in the IMS circumstance, nor studied how to defend the DoS attacks when those protocols work together. In this paper, the identified DoS attack exploits the unique vulnerabilities of IMS such as the chain reaction.

There are many kinds of DoS attacks. They either have different special characteristics or work in different situations. Besides the traditional high-rate high-volume DoS attacks, a low-rate high-volume TCP attack [22] could periodically generate a high volume of packets in a burst to force TCP flows to retransmit. The DoS attack identified in this paper and the DoS attack identified by Lee [23] are both based on

low-rate, low-volume attacks, which exploit the vulnerabilities of 3G telecom network to trigger much traffic; However, Lee's work cannot address the chain reaction problem, which has been addressed by our solution.

Many techniques have been developed to detect and defend against DoS attacks. For instance, [24] [25] and [23] provide statistical online detection mechanisms for DoS detection. In order to increase the robustness of the channels to overloads or DoS attacks, Zang and Bolot [26] develop a family of profile-based paging techniques to reduce signaling load. A very nice classification of DoS attacks and defenses can be found in [27]. The defense mechanisms of ours and Lee's [23] are both based on users' session intervals to identify the attackers, but we use user's history information individually for attacker identification, not generalizing the information from all the users.

In order to stop an attacker, localizing the source of the attack is a proper solution. Traceback [28] and ICMP [29] trace attacks back towards their origin. Pi [30] embeds a path fingerprint in each packet to enable a victim to identify packets traversing the same paths through the Internet. However, those approaches can not localize the malicious clients in this DoS attack, since they attack the IMS indirectly.

VI. Conclusion and Future Work

In this paper, we have presented a new DoS attack that targets at 3G and IMS networks. The attack exploits the vulnerabilities of IMS and SIP protocols in 3G networks. We have showed through experiments in a test bed that this attack can trigger chain reaction DoS attacks to substantially congest an IMS network and block all its services with limited traffic. Its low-volume, low-rate nature and the small defense time window make it hard to defend. We proposed a defense mechanism composed of two statistical GRSh-based algorithms to deal with this attack. We have shown that our mechanism can detect this attack before the entire network is congested with very few false positives; our identification mechanism can help the remote RLSs to identify and block the sources of this attack in a timely manner.

As an initial work, we do not expect to solve all the problems. So far we have used CPU usage as the monitoring factor for DoS detection. In our future work we will study how to monitor other system resources for attack detection. We will also study the effectiveness of chain reaction attacks on IMS networks of different scales.

References

[1] C. Urrutia-Valds, A. Mukhopadhyay, and M. El-Sayed, "Presence and availability with ims: Applications architecture, traffic analysis, and capacity impacts," *Bell Labs Technical Journal*, vol. 10(4), pp. 101–107, Mar 2006.

[2] M. Poikselka, A. Niemi, H. Khartabil, and G. Mayer, *The IMS IP Multimedia Concepts and Services*. Wiley, March 2006.

[3] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session initiation protocol. rfc 3261," *IETF*, June 2002.

[4] R. Zhang and X. Wang, "Billing attacks on SIP-based VoIP systems," *In Proceedings of the 1st USENIX Workshop on Offensive Technologies (WOOT 2007)*, 2007.

[5] M.A.Girshick and H.Rubin, "A bayes approach to a quality control model," *Ann.Math.Statist*, vol. 23 1, pp. 114–125, 1952.

[6] Z. Cao, C. Chi, and R.Hao, "User behavior modelling and traffic analysis of IMS presence servers," in *GLOBECOM '08: Proceedings of the 2008 Global Telecommunications Conference,*, 2008.

[7] B. E. Brodsky and B. S. Darkhovsky, *Non-parametric methods in Change-Point Problems*. Springer, 1993.

[8] OPENSER, "The open sip express router (OpenSER)."

[9] R. Gayraud and O. Jacques., "SIPp," 2008.

[10] C. Shen, H. Schulzrinne, and E. Nahum, "Session initiation protocol (SIP) server overload control: Design and evaluation," in *IPTComm'08: Proceedings of Principles, Systems and Applications of IP Telecommunications 2008*, Heidelberg, Germany, 2008.

[11] E. C. Noel and C. R. Johnson, "Initial simulation results that analyze SIP based VoIP networks under overload," in *IPTComm'07: Proceedings of International Teletraffic Congress 2007*, Ottawa, Canada, 2007, pp. 54C–64.

[12] P. Traynor, W. Enck, P. McDaniel, and T. L. Porta, "Mitigating attacks on open functionality in SMS-capable cellular networks," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2006, pp. 182–193.

[13] Z. Zhu, G. Cao, S. Zhu, S. Ranjany, and A. Nucciy, "A social network based patching scheme for worm containment in cellular networks," in *Proceedings of IEEE INFOCOM*, Rio de Janeiro, Brazil, April 2009.

[14] 3GPP2, "IMS security framework," www.3gpp2.org/Public_html/specs/S.R0086-A_v1.0_040614.pdf 2004.

[15] 3GPP, "Feasibility study on IMS security extensions 3GPP TR 33.802," *3GPP*, vol. V0.1.0, 2005.

[16] E. E. Anderlind, D. W. Faucher, E. H. Grosse, D. N. Heer, A. R. McGee, D. P. Strand, and R. J. T. Jr., "IMS security," *Bell Labs Technical Journal*, vol. 11(1), pp. 37–58, 2006.

[17] FORTINET, "Exploring ims network security for next generation network (NGN) carriers," www.fortinet.com/doc/whitepaper/IMS_Security_Carriers.html 2007.

[18] HUAWEI, "Solution–building E2E IMS network security," www.huawei.com/publications/view.do?id=1144&cid=1802&pid=61 2006.

[19] A. B. Roach, "Session initiation protocol (SIP)-specific event notification. rfc 3265," *IETF*, June 2002.

[20] J. Rosenberg, "A presence event package for the session initiation protocol (SIP). rfc 3856," *IETF*, August 2004.

[21] A. B. Roach, B. Campbell, and J. Rosenberg, "A session initiation protocol (SIP) event notification extension for resource lists. rfc 4662," *IETF*, August 2006.

[22] A. Kuzmanovic and E. W. Knightly, "Low-rate TCP-targeted denial of service attacks and counter strategies," in *IEEE/ACM Transactions on Networking (TON)*, vol. 14, August 2006.

[23] P. P. Lee, T. Bu, and T. Woo, "On the detection of signaling DoS attacks on 3G wireless networks," in *Proceedings of IEEE INFOCOM*, Anchorage, Alaska, May 2007.

[24] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast portscan detection using sequential hypothesis testing," in *IEEE Symposium on Security and Privacy 2004*, Oakland, CA, May 2004.

[25] "Change-point monitoring for the detection of DoS attacks," *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 4, pp. 193–208, 2004, member-Haining Wang and Member-Danlu Zhang and Fellow-Kang G. Shin.

[26] H. Zang and J. C. Bolot, "Mining call and mobility data to improve paging efficiency in cellular networks," in *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2007, pp. 123–134.

[27] A. Hussain, J. Heidemann, and C. Papadopoulos, "A framework for classifying denial of service attacks," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2003, pp. 99–110.

[28] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," vol. 9, no. 3, JUNE 2001.

[29] S. Bellovin, "ICMP traceback messages. ietf internet draft," Feb 2003.

[30] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2003, p. 93.