Modeling and Mitigating the Coremelt Attack*

Guosong Yang[1], Hossein Hosseini[2], Dinuka Sahabandu[2], Andrew Clark[3], João Hespanha[1], and Radha Poovendran[2]

***Abstract*— This paper studies the Coremelt attack, a link-flooding Distributed Denial of Service attack that exhausts the bandwidth at a core network link using low-intensity traffic flows between subverted machines. A dynamical system model is formulated for analyzing the effect of Coremelt attack on a single-link Transmission Control Protocol (TCP) network and developing mitigation methods. For the case with a limited number of subverted sources, a modified TCP algorithm is developed for the attackers to achieve a desired congestion level. A mitigation method is proposed to improve the link usage of legitimate users when the link is under attack. The network performance under different attack and mitigation scenarios is illustrated through simulation results.**

I. Introduction

A major threat to Internet security today is the Distributed Denial of Service (DDoS) attack, in which an adversary attempts to interrupt legitimate users' access to certain network resources by sending superfluous traffics from a vast number of subverted sources (called a botnet). Since the first incident of DDoS attack reported by the Computer Incident Advisory Capability (CIAC) in 1999 [1], many large-scale DDoS attacks have been launched against various infrastructures and organizations (see, e.g., [2], [3]). Moreover, the frequency and size of DDoS attacks grows rapidly every year. The 11th annual Worldwide Infrastructure Security Report (WISR) [4] from Arbor Network shows that 44% of the service provider respondents have seen more than 21 DDoS attacks per month from November 2014 to November 2015, up from 38% in the previous year; moreover, nearly one-quarter of them experienced attacks with size over 100 Gbps, whereas 20 percent reported attacks over 50Gbps the year before.

As pointed out in [3], most DDoS attacks exploit one or both of the following two methods: i) disrupting legitimate users' service by exhausting server resources; ii) disrupting legitimate users' connectivity by exhausting link bandwidth. The second method (called link-flooding DDoS) proves to be particularly effective and stealthy, as it can be executed without sending traffic to the victims at all. One of the first and foremost link-flooding DDoS attacks proposed is the Coremelt attack [5] shown in Fig. 1, in which an adversary attempts to congest a core link of the network using low-intensity traffics between subverted machines (called bots). The Coremelt attack is of high interest as it allows individual sources participating in the coordinated attack to keep the flow rates well below the detection thresholds, yet effectively exhausts the bandwidth of the target link in the intersection of attack flows. Moreover, it has been further extended in the security literature to demonstrate attacks such as the Crossfire attack [6] that have no known countermeasure.

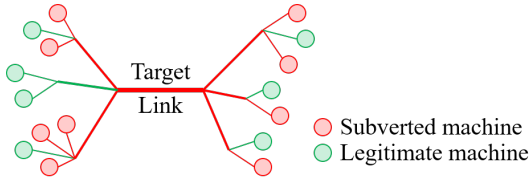

Fig. 1. The Coremelt Attack [5]: a coordinated link-flooding DDoS attack which allows the flow rate of each subverted source to be kept well below the detection threshold, yet effectively congests/shuts down a target link in the intersection of attack flows.

This paper studies the Coremelt attack proposed in [5] on a single-link Transmission Control Protocol (TCP) network. Our goal is to build a dynamical system framework for modeling and developing mitigation methods for the Coremelt attack. We make the following specific contributions:

- A dynamical system model is formulated for analyzing the effect of Coremelt attack on a single-link TCP network. Using this model, stability and convergence of flow rates are established via Lyapunov analysis.
- For a case with a limited number of subverted sources, a modified TCP algorithm is developed for the attackers to achieve a desired congestion level.
- A mitigation method against the Coremelt attack is proposed. With this mitigation, the link usage by legitimate users remains the same as in the case where all sources follow the standard TCP, thus increasing the number of bots the adversary needs to occupy the same bandwidth.
- Simulation results are provided to illustrate different attack and mitigation scenarios.

This paper is organized as follows: Section II provides the backgrounds on TCP and Coremelt attack. The network model and the dynamics of TCP sources are formulated in Section III. In Section IV, we analyze the effect of the Coremelt attack using the dynamical system model, and present a modified TCP algorithm through which a desired

*This paper is based upon work supported by the Office of Naval Research grants N00014-14-1-0029 and N00014-16-1-2710, the Army Research Office grant W911NF-16-1-0485, and the National Science Foundation grant CNS-1446866.

[1]G. Yang and J. Hespanha are with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 U.S. {guosongyang, hespanha}@ucsb.edu.

[2]H. Hosseini, D. Sahabandu, and R. Poovendran are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 U.S. {hosseinh, sdinuka, rp3}@uw.edu.

[3]A. Clark is with the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, Worcester, MA 01609 U.S. aclark@wpi.edu.

congestion level can be achieved. Mitigation methods are discussed in Section V. Simulation results are presented in Section VI. Section VII concludes the paper.

II. Background

Transmission Control Protocol (TCP). TCP is a congestion control algorithm which defines how to establish a connection between a source-destination pair and reliably transmit data over the Internet [7]. A TCP source controls its flow rate by transmitting one congestion window of packets per round-trip time (RTT), and avoids congestion by adjusting the window size based on acknowledgments (ACKs) received from the destination. The adjustment follows the additive-increase/multiplicative-decrease (AIMD) feedback control algorithm [8], which linearly increases the window when all ACKs are accounted for, and halves it if congestion is detected through a missing ACK. Many studies have been devoted to the development of dynamical system models for TCP (see, e.g, [9] and references therein). In this paper, we consider TCP-NewReno [10], one of the most widely used TCP variants in modern Internet.

Coremelt Attack. The Coremelt attack [5] is a DDoS attack in which a large number of bots communicate with each other in such a way that the traffic of each source-destination pair traverses a core link of the network and their aggregation exceeds the link capacity. It has been shown that such an attack can disrupt the operation of the entire Internet by deploying only an average-size botnet [5]. Conventional authenticity-based mitigation methods fail against the Coremelt attack due to the use of legitimate-looking traffic by the bots. Also, the distributed nature of the botnet makes defense techniques such as path tracking or IP traceback more complex.

Several papers proposed defense mechanisms against the Coremelt attack [11]–[16]. A new network architecture was proposed in [11] as a defense against DDoS attacks. In [12], a protocol for edge gateways (routers at the both ends of the link) was developed to prevent flows that use IP spoofing for link-flooding attacks. In [13], the authors suggested bandwidth isolation between different traffic flows to minimize the collateral damage caused by link-flooding attacks. A collaborative defense mechanism which enables routers to distinguish low-intensity attack flows from legitimate ones was developed in [14]. In [15], observable statistics at routers were used to infer the presence of an attack. In [16], the authors proposed to detect attacks by rate change tests.

In this paper, we model the Coremelt attack in the scope of the current network architecture, and propose a mitigation method based on penalizing the sources that are most responsible for the congestion. This mitigation method does not require identifying malicious sources/traffics or modifying the transmission protocols used by sources. Simulation results show that it successfully protects legitimate users and suppresses aggressive flows.

III. Preliminaries

In this section, we define the network model and formulate the dynamics of TCP sources.

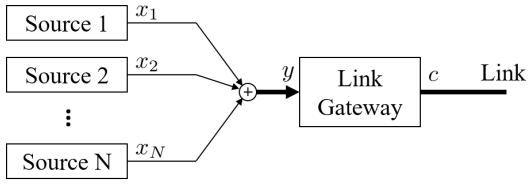

Fig. 2. A network with the "dumbbell" topology: N sources compete for the finite bandwidth c of a single link. When the aggregate flow rate $y = \sum x_k > c$, a fraction p of the traffic is dropped so that $(1-p)\,y \le c$.

Denote by $\mathbb{R}_+ := [0, \infty)$ the set of nonnegative real numbers. Given $a \in \mathbb{R}$ and $b \in \mathbb{R}_+$, define the projection

$$(a)_b^+ := \begin{cases} a, & \text{if } b > 0, \text{ or } b = 0 \text{ and } a \ge 0; \\ 0, & \text{if } b = 0 \text{ and } a < 0. \end{cases}$$

A. Network model

Consider the computer network shown in Fig. 2. In this network, traffic flows from N sources compete for the finite bandwidth C of a single link. Such a configuration is known as the "dumbbell" topology, and is typically used to analyze network congestion control [17]. Denote by x_k the flow rate from source $k \in \mathcal{N} := \{1, \dots, N\}$, and by y the aggregate flow rate at the link gateway, that is,

$$y = \sum_{k \in \mathcal{N}} x_k. \tag{1}$$

The link gateway manages the throughput via the "tail drop" technique: if the aggregate flow rate $y > C$, the gateway buffers the excess data in a queue, waiting to be transmitted; when the queue is filled to its maximum capacity, the newly arriving flows will be dropped until there is enough room to accept incoming flows. In this case, the network is said to be congested.

We assume that packets from different sources arrive at the link gateway in a random order, so that they all suffer from the same packet drop probability p. The value of p is approximated via a simple model

$$p = g(y) := \begin{cases} 1 - C/y, & \text{if } y > C; \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

which has been used in, e.g., [18], [19]. This model ensures that $p \in [0, 1]$ at all times and the throughput never exceeds the bandwidth, that is, $(1-p)\,y \le C$. In the subsequent analysis, we also use the property that g is a nondecreasing function, that is

$$(y - y^*)(g(y) - g(y^*)) \ge 0 \qquad \forall\, y, y^* \ge 0. \tag{3}$$

This property implies that the system defined by the memoryless function g in (2) with input $y - y^*$ and output $p - p^*$ is passive (here y^* is the steady-state aggregate flow rate and $p^* = g(y^*)$ is the steady-state packet drop probability.).

B. Dynamics of TCP

We consider TCP-NewReno [10], which was designed to improve the performance of its original version TCP-Reno [20] when there are multiple packet drops in one congestion window, and has become one of the most widely used TCP variants. In TCP-NewReno, source k sends a congestion window of w_k packets per RTT τ_k, and the window size w_k is governed by the following AIMD algorithm:

$$\begin{cases} w_k^+ \leftarrow w_k + 1, & \text{without congestion;} \\ w_k^+ \leftarrow w_k/2, & \text{with congestion.} \end{cases}$$

Here by congestion we mean that at least one of the w_k packets is dropped (detected by a missing ACK), and denote by q_k the congestion probability. Also, w_k is halved only once when there are multiple packet drops in one RTT. We assume that the RTT τ_k is constant for each source k, and define the flow rate x_k as the average number of packets transmitted per unit time, that is, $x_k = w_k/\tau_k$. This approximation is valid as our model is not intended to provide accurate description at finer time scales than the RTT. In each RTT, the window increases by 1 with probability $1-q_k$, and decreases by $w_k/2$ with probability q_k. Hence the average change of window size is $(1-q_k)/\tau_k - w_k q_k/(2\tau_k)$ per unit time, which, combined with $x_k = w_k/\tau_k$, yields the dynamical system model

$$\dot{x}_k = \frac{1-q_k}{\tau_k^2} - \frac{q_k x_k}{2\tau_k}. \tag{4}$$

In particular, if $x_k = 0$ then $\dot{x}_k > 0$, which ensures that the flow rate x_k is always nonnegative. We model the congestion probability q_k by

$$q_k = \tau_k x_k p, \tag{5}$$

which follows from the first-order approximation $q_k = 1 - (1-p)^{w_k} \approx w_k p$.

Remark 1. In analyzing loss-based TCP algorithms, many papers adopt the dynamical system model

$$\dot{x}_k = \frac{1-p}{\tau_k^2} - \frac{1}{2} p x_k^2,$$

which is derived via the approximation that, on average, the window increases by $(1-p)\,x_k/w_k$ per unit time due to the $(1-p)\,x_k$ ACKs received, and decreases by $p x_k w_k/2$ per unit time due to the $p x_k$ packets dropped [9]. However, this formulation is developed for TCP-Reno [20], and is not suitable for analyzing TCP-NewReno. More specifically, if multiple packets are dropped in the same window, the window is halved only once in TCP-NewReno, but multiple times in TCP-Reno. See [21] for derivations of similar dynamical system models for TCP-Reno and TCP-NewReno in the discrete-time setting.

IV. Analysis of Coremelt attack

In this section, we analyze the Coremelt attack in two different scenarios using tools from dynamical systems theory. In the first scenario, we assume all sources follow TCP-NewReno, and compute the number of sources the adversary needs to achieve a certain drop probability. In the second scenario, we propose a modified TCP algorithm for attackers through which a desired congestion probability can be achieved. In both cases, Lyapunov analysis is used to establish stability and convergence of the flow rates.

Suppose there is an adversary who has taken control of M sources (called attackers), and is trying to congest the core link so that the other $S := N - M$ sources (called users) suffer from low throughputs. We only consider the case of static Coremelt attacks, in which the attackers (and thus their transmission paths) are fixed. Without loss of generality, denote by $\mathcal{A} := \{1, \dots, M\}$ and $\mathcal{U} := \{M+1, \dots, N\}$ the sets of attackers and users, respectively. Hence $\mathcal{N} = \mathcal{A} \cup \mathcal{U}$. We assume that all users follow TCP-NewReno. Attackers, however, can follow a different algorithm, which is the same for all attackers and is distributed in the sense that the attackers do not coordinate with each other and adjust their flow rates independently. Each source (user or attacker) is not aware of the total number of sources, link capacity or instantaneous link usage, and adjust its flow rate only based on the feedback (i.e., the ACKs received).

A. Attack with TCP-NewReno

Suppose that all sources (users and attackers) follow TCP-NewReno. Then all flow rates $x_k \geq 0$ satisfy the dynamical system model (4) and (5), which can be rewritten as

$$\dot{x}_k = G_k(x_k)(D_k(x_k) - q_k), \qquad k \in \mathcal{N}, \tag{6}$$

where

$$D_k(x_k) := \frac{2}{\tau_k x_k + 2}$$

is a strictly positive, strictly decreasing function on $\mathbb{R}_+$, and

$$G_k(x_k) := \frac{\tau_k x_k + 2}{2\tau_k^2} = \frac{1}{\tau_k^2 D_k(x_k)}$$

is a strictly positive function on $\mathbb{R}_+$. Hence the network becomes a dynamical system defined by (1), (2), (5), and (6) with the state $x := (x_1, \dots, x_N) \in \mathbb{R}^N$. Note that y and p are uniquely determined by x through (1) and (2). Setting $\dot{x}_k = 0$, we obtain from (5) and (6) that

$$D_k(x_k^*) = q_k^* = \tau_k x_k^* p^* > 0 \qquad \forall\, k \in \mathcal{N}. \tag{7}$$

Consequently, the steady-state flow rate x_k^* and drop probability p^* are both positive, which also implies the steady-state aggregate flow rate $y^* > C$. Next, we establish stability and convergence of flow rates.

Theorem 1. *The system defined by* (1)*,* (2)*,* (5)*, and* (6) *is globally asymptotically stable w.r.t. a unique equilibrium.*

Proof. The property that all D_k are strictly decreasing functions ensures that there is a unique equilibrium x^*, which can be calculated by combining (1), (2), (5), and (7). Consider the function $V : \prod_{k\in\mathcal{N}} [-x_k^*, \infty) \to \mathbb{R}_+$ defined by

$$V(x - x^*) := \sum_{k\in\mathcal{N}} \frac{V_k(x_k - x_k^*)}{\tau_k x_k^*}$$

with

$$V_k(x_k - x_k^*) := \int_{x_k^*}^{x_k} \frac{s - x_k^*}{G_k(s)}\,\mathrm{d}s, \qquad k \in \mathcal{N}.$$

The construction of V is inspired by [22]. As all $x_k^* > 0$ and all K_k are strictly positive functions, V is a positive definite function. Also, all V_k, and therefore V, are radially unbounded. For each $k \in \mathcal{N}$,

$$\begin{aligned}
\dot{V}_k &= \frac{\mathrm{d}V_k}{\mathrm{d}x_k}\,\dot{x}_k \\
&= (x_k - x_k^*)(D_k(x_k) - q_k) \\
&= (x_k - x_k^*)(D_k(x_k) - D_k(x_k^*)) - \tau_k p (x_k - x_k^*)^2 \\
&\quad - \tau_k x_k^* (p - p^*)(x_k - x_k^*) \\
&\le (x_k - x_k^*)(D_k(x_k) - D_k(x_k^*)) \\
&\quad - \tau_k x_k^* (p - p^*)(x_k - x_k^*),
\end{aligned}$$

where the last equality follows partially from (5) and (7), and the inequality follows from $\tau_k p (x_k - x_k^*)^2 \ge 0$. Hence

$$\begin{aligned}
\dot{V} &= \sum_{k\in\mathcal{N}} \frac{1}{\tau_k x_k^*}\dot{V}_k \\
&\le \sum_{k\in\mathcal{N}} \frac{(x_k - x_k^*)(D_k(x_k) - D_k(x_k^*))}{\tau_k x_k^*} \\
&\quad - (p - p^*) \sum_{k\in\mathcal{N}} (x_k - x_k^*) \\
&= \sum_{k\in\mathcal{N}} \frac{(x_k - x_k^*)(D_k(x_k) - D_k(x_k^*))}{\tau_k x_k^*} - (p - p^*)(y - y^*) \\
&\le \sum_{k\in\mathcal{N}} \frac{(x_k - x_k^*)(D_k(x_k) - D_k(x_k^*))}{\tau_k x_k^*},
\end{aligned}$$

where the last equality follows from (1), and the last inequality follows from (2) and (3). As D_k is a strictly decreasing function, $(x_k - x_k^*)(D_k(x_k) - D_k(x_k^*))/(\tau_k x_k^*)$ is a negative definite function of $x_k - x_k^*$. Hence V is a Lyapunov function, and x^* is globally asymptotically stable. □

Theorem 1 ensures that the drop probability p will converge to the equilibrium value p^*. Combining (1), (2), (5), and (7), we obtain that

$$\sum_{k\in\mathcal{N}} \frac{1}{\tau_k} = \frac{\sqrt{1 + 2/p^*} + 1}{2(1 - p^*)}\, p^* C. \tag{8}$$

Suppose that every source has the same RRT τ, which is reasonable because the wait in the congested link dominates the network latency. Then the minimum number of attackers needed to achieve a certain p^* is given by

$$M \ge \frac{\sqrt{1 + 2/p^*} + 1}{2(1 - p^*)}\, \tau p^* C - S,$$

in which the number of users S is usually unknown to the adversary but can be estimated empirically. Even without knowing S, it will be sufficient to deploy $(\sqrt{1 + 2/p^*} + 1)\,\tau p^* C/(2(1 - p^*))$ bots.

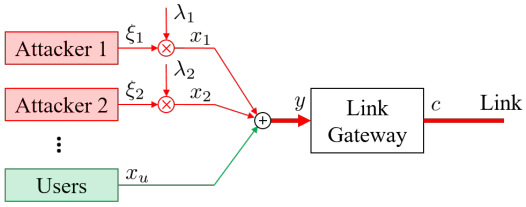

Fig. 3. Attack with a modified TCP: each attacker sets its flow rate x_j by amplifying an internal state ξ_j that emulates a TCP-NewReno flow with a gain $\lambda_j \ge 0$.

B. Attack with a modified TCP

Consider the case where the adversary intends to achieve a target congestion probability $q_0 \in (0, 1)$ with a limited number of attackers. On the one hand, simply following standard TCP may not be sufficient, as the achievable drop probability is constrained by (8). On the other hand, letting the attackers send as much traffic as they can will greatly increase the risk of being detected. In this subsection, we develop a carefully orchestrated algorithm for the attackers to adjust their flow rates based on the feedback, so that the target congestion probability can be achieved without generating excessive traffic.

Suppose that all users follow TCP-NewReno. Then their flow rates satisfy

$$\dot{x}_i = G_i(x_i)(D_i(x_i) - p), \qquad i \in \mathcal{U}. \tag{9}$$

Each attacker sets its flow rate x_j by "amplifying" an internal state ξ_j that emulates a TCP-NewReno flow with a gain $\lambda_j \ge 0$. The gain λ_j is adjusted according to the following algorithm:

$$\begin{cases} \lambda_j \leftarrow \lambda_j + \gamma_j \tau_j q_0 \xi_j, & \text{without congestion;} \\ \lambda_j \leftarrow \lambda_j - \gamma_j \tau_j (1 - q_0)\xi_j, & \text{with congestion.} \end{cases} \tag{10}$$

Based on the same averaging arguments used in formulating the dynamical system model for TCP-NewReno in Section III-B, we obtain

$$\begin{aligned}
\dot{\xi}_j &= G_j(\xi_j)(D_j(\xi_j) - q_j), \\
\dot{\lambda}_j &= \gamma_j \xi_j (q_0 - q_j)^+_{\lambda_j}, \qquad j \in \mathcal{A}, \\
x_j &= \lambda_j \xi_j,
\end{aligned} \tag{11}$$

where $\gamma_j > 0$ is a constant used to adjust the convergence rate, and the projection ensures $\lambda_j \ge 0$. (In particular, if $\lambda_j \equiv 1$ then the attacker's behavior is the same as TCP-NewReno.) Hence the network (shown in Fig. 3) becomes a dynamical system defined by (1), (2), (5), (9), and (11) with the state $(x_u, \xi, \lambda) \in \mathbb{R}^{N+M}$, where $x_u := (x_{M+1}, \ldots, x_N)$, $\xi := (\xi_1, \ldots, \xi_M)$ and $\lambda := (\lambda_1, \ldots, \lambda_M)$. Note that y and p are uniquely determined by (x_u, ξ, λ) through (1), (2), and (11). Setting $\dot{x}_i = \dot{\xi}_j = \dot{\lambda}_j = 0$, we obtain from (5), (9) and (11) that the steady-state congestion probability $q_j^* = q_0$, and the steady-state vector $(x_u^*, \xi^*, \lambda^*)$ satisfies

$$\begin{aligned}
D_i(x_i^*) &= \tau_i x_i^* p^* > 0 \qquad && \forall\, i \in \mathcal{U}, \\
D_j(\xi_j^*) &= q_0 = \tau_j \lambda_j^* \xi_j^* p^* > 0 \qquad && \forall\, j \in \mathcal{A}.
\end{aligned} \tag{12}$$

Consequently, the steady-state values $x_i^*, \lambda_j^*, \xi_j^*$ and the steady-state drop probability p^* are all positive, which also implies that the steady-state aggregate flow rate $y^* > C$. Next, we establish stability and convergence of the network, which ensures that the the target congestion probability q_0 can be achieved.

Theorem 2. *The system defined by* (1)*,* (2)*,* (5)*,* (9) *and* (11) *is globally asymptotically stable w.r.t. a unique equilibrium.*

Proof. The property that all D_i and D_j are strictly decreasing functions ensures that there is a unique equilibrium $(x_u^*, \xi^*, \lambda^*)$, which can be calculated by combining (1), (2), (5), and (12). Consider the function $V : \prod_{i\in\mathcal{U}}[-x_i^*, \infty) \times \prod_{j\in\mathcal{A}}[-\xi_j^*, \infty) \times \prod_{j\in\mathcal{A}}[-\lambda_j^*, \infty) \to \mathbb{R}_+$ defined by

$$\begin{aligned} &V(x_u - x_u^*, \xi - \xi^*, \lambda - \lambda^*) \\ &:= \sum_{i\in\mathcal{U}} \frac{V_i(x_i - x_i^*)}{\tau_i x_i^*} + \sum_{j\in\mathcal{A}} \frac{V_j(\xi_j - \xi_j^*, \lambda_j - \lambda_j^*)}{\tau_j \lambda_i^* \xi_j^*}, \end{aligned}$$

where

$$V_i(x_i - x_i^*) := \int_{x_i^*}^{x_i} \frac{s - x_i^*}{G_i(s)}\,\mathrm{d}s, \qquad i \in \mathcal{U},$$

and for $j \in \mathcal{A}$,

$$V_j(\xi_i - \xi_i^*, \lambda_j - \lambda_j^*) := \frac{(\lambda_j - \lambda_j^*)^2}{2\gamma_j} + \lambda_j^* \int_{\xi_j^*}^{\xi_j} \frac{s - \xi_j^*}{G_j(s)}\,\mathrm{d}s.$$

Based on the proof of Theorem 1, we see that V is positive definite and radially unbounded, and that

$$\dot{V}_i \le (x_i - x_i^*)(D_i(x_i) - D_i(x_i^*)) - \tau_i x_i^* (p - p^*)(x_i - x_i^*)$$

for each $i \in \mathcal{U}$. For each $j \in \mathcal{A}$,

$$\begin{aligned} \dot{V}_j &= \frac{\partial V_j}{\partial \xi_j}\,\dot{\xi}_j + \frac{\partial V_j}{\partial \lambda_j}\,\dot{\lambda}_j \\ &= \lambda_j^*(\xi_j - \xi_j^*)(D_j(\xi_j) - q_j) + (\lambda_j - \lambda_j^*)\,\xi_j(q_0 - q_j)^+_{\lambda_j} \\ &\le \lambda_j^*(\xi_j - \xi_j^*)(D_j(\xi_j) - q_j) + (\lambda_j - \lambda_j^*)\,\xi_j(q_0 - q_j) \\ &= \lambda_j^*(\xi_j - \xi_j^*)(D_j(\xi_j) - D_j(\xi_j^*)) - \tau_j p(\lambda_j\xi_j - \lambda_j^*\xi_j^*)^2 \\ &\quad - \tau_j \lambda_j^* \xi_j^* (p - p^*)(\lambda_j\xi_j - \lambda_j^*\xi_j^*) \\ &\le \lambda_j^*(\xi_j - \xi_j^*)(D_j(\xi_j) - D_j(\xi_j^*)) \\ &\quad - \tau_j \lambda_j^* \xi_j^* (p - p^*)(\lambda_j\xi_j - \lambda_j^*\xi_j^*), \end{aligned}$$

where the first inequality follows from the fact that if $\lambda_j = 0$ and $q_0 < q_j$ then $(0 - \lambda_j^*)\,\xi_j(q_0 - q_j) \ge 0$, the last equality follows partially from (5) and (12), and the last inequality follows from $\tau_j p(\lambda_j\xi_j - \lambda_j^*\xi_j^*)^2 \ge 0$. Hence

$$\begin{aligned} \dot{V} &= \sum_{i\in\mathcal{U}} \frac{1}{\tau_i x_i^*}\dot{V}_i + \sum_{j\in\mathcal{A}} \frac{1}{\tau_j \lambda_i^* \xi_j^*}\dot{V}_j \\ &\le W(x_u - x_u^*, \xi - \xi^*) \\ &\quad - (p - p^*)\Bigg(\sum_{i\in\mathcal{U}}(x_i - x_i^*) + \sum_{j\in\mathcal{A}}(\lambda_j\xi_j - \lambda_j^*\xi_j^*)\Bigg) \\ &= W(x_u - x_u^*, \xi - \xi^*) - (p - p^*)(y - y^*) \\ &\le W(x_u - x_u^*, \xi - \xi^*) \end{aligned}$$

with

$$\begin{aligned} W(x_u - x_u^*, \xi - \xi^*) &:= \sum_{i\in\mathcal{U}} \frac{(x_i - x_i^*)(D_i(x_i) - D_i(x_i^*))}{\tau_i x_i^*} \\ &\quad + \sum_{j\in\mathcal{A}} \frac{(\xi_j - \xi_j^*)(D_j(\xi_j) - D_j(\xi_j^*))}{\tau_j \xi_j^*}, \end{aligned}$$

where the last equality follows from (1), and the last inequality follows from (2) and (3). As all D_i and D_j are strictly decreasing, W is a negative definite function of $x_u - x_u^*$ and $\xi - \xi^*$. Hence V is a weak Lyapunov function, and LaSalle's invariance principle implies that the the unique equilibrium $(x_u^*, \xi^*, \lambda^*)$ is globally asymptotically stable. □

Theorem 2 ensures that the congestion probability converges to the target value q_0. When targeting a large q_0, the attackers need to engage relatively large flow rates compared to the users'. From the perspective of link gateway, this means that the attackers are advertising small RTTs, and could potentially be identified if their perceived RTTs become less than a predefined bound. This type of defense mechanism is not explored here, but a topic for future research. In the next section, we propose a mitigation method which improves the performance of legitimate users even if the attackers cannot be identified.

V. Mitigation

One solution to DDoS attacks is to distinguish malicious sources/traffics from legitimate ones. Defense mechanisms using this approach include source authentication [23] and packets inspection [24]. However, in the Coremelt attack the attackers use low-intensity, legitimate-looking traffics, which makes it difficult to detect the attack flows. In this section, we consider the case where the attackers cannot be identified, and propose a mitigation method solely based on the flow rates from each source.

When attackers cannot be distinguished from legitimate users, one option for mitigation is to punish aggressive flows by dropping more of their packets. Following this approach, our mitigation method is formulated by letting the link gateway monitor the source flow rates and assign an individual drop probability for each source. For implementation, the flow rate from each source can be approximated by counting the packets it transmitted within a certain time (in the order of a few RTTs); the individual drop probability can be enforced by dividing the bandwidth and dropping packets accordingly. In our mitigation method, the ideal individual drop probabilities p_k are assigned so that the bandwidth C is evenly shared by all sources, that is, $p_k = \max\{1 - C/(Nx_k), 0\}$. Consequently, the sources with higher flow rates will experience higher drop probabilities. The steady-state throughput of every source is $(1 - p_k^*)\,x_k^* = C/N$ and the steady-state link usage ratio of legitimate users is $1 - M/N$, which is equivalent to the case where all sources follow TCP-NewReno and have the same RTT.

The proposed mitigation method does not require modifying the transmission protocols used by sources; it only

involves the link gateway adjusting the individual drop probability for each user based on its flow rate, the link capacity, and the number of sources accessing the link. Moreover, it ensures that, regardless of the attack strategy and even if the attackers are coordinated, the steady-state link usage ratio of users will always be at least $1 - M/N$. Consequently, to occupy the same bandwidth as in the case without mitigation, the adversary needs to deploy a significantly larger number of bots. Limitations of this method include that legitimate users with smaller RTTs will also be penalized, and it has little effect when all attackers follow the same TCP algorithm as users.

VI. Simulation Results

In this section, we provide simulation results for the scenarios where the attackers follow TCP-NewReno or the modified TCP algorithm, and the link employs no mitigation or the proposed mitigation method. In all cases, we assume that the legitimate users follow TCP-NewReno. In the modified TCP algorithm, each attacker amplifies its flow rate by the gain λ_j, which is updated according to (10). In the proposed mitigation method, the link sets an individual drop probability for each source based on its flow rate. In simulations, assuming the source flows arrive in a random order, the link gateway drops them one by one with the corresponding probability, until the sum of remaining flows becomes less than or equal to the link capacity.

We consider the scenario where 2000 users and 1000 attackers are transmitting over a link with a capacity of 10^6 packets per RTT. (For an average packet size of 500 bytes and an average RTT of 50 ms, the link bandwidth is around 10 GBps.) We assume that all RTTs are the same and use that value as the time unit in plots. We also set $\gamma_j = 10$ in (11), and initialize congestion windows and gains λ_j randomly.

Figure 4 shows the congestion probability of attackers and the link usage ratio of users when the attackers follow TCP-NewReno or the modified TCP algorithm. When the attackers follow TCP-NewReno, the congestion probability is almost zero and the link usage ratio of users is about $1 - M/N = 2/3$. Meanwhile, with the modified TCP algorithm, the congestion probability of attackers converges to the target value q_0 and the link usage ratio of users is almost zero.

Figure 5 shows the congestion probability of attackers and the link usage ratio of users when the link deploys the proposed mitigation method. When the attackers follow TCP-NewReno, the congestion is almost zero and the link usage ratio of users is still about $1 - M/N = 2/3$. Therefore, the mitigation does not disrupt the performance of users when all sources follow TCP-NewReno. When the attackers follow the modified TCP algorithm, their congestion probability can still converge to the target value, but the link usage ratio of users becomes higher. Moreover, with a higher target value q_0 the attackers will suffer from an even lower link usage ratio. Therefore, under our mitigation method, the better strategy for attackers is to use the same transmission protocol as users. As a result, the adversary needs to deploy a significantly larger number of bots to occupy the same bandwidth as in the case without mitigation.

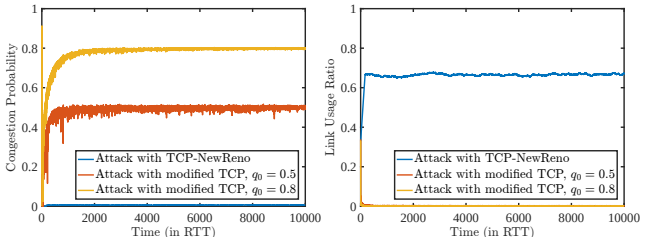

Fig. 4. Effect of the modified TCP algorithm on the congestion probability of attackers and the link usage ratio of users. When the attackers follow TCP-NewReno, the congestion probability is almost zero and the link usage ratio of users is about $2/3$. Meanwhile, with the modified TCP algorithm, the congestion probability of attackers converges to the target value q_0 and the link usage ratio of users is almost zero.

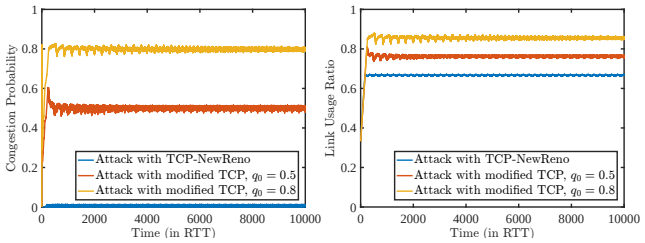

Fig. 5. Effect of the proposed mitigation method on the congestion probability of attackers and the link usage ratio of users. When the attackers follow TCP-NewReno, the mitigation does not disrupt the performance of users. When the attackers follow the modified TCP algorithm, their congestion probability can still converge to the target value, but the link usage ratio of users becomes larger. Therefore, the mitigation successfully protects the sources following TCP-NewReno.

VII. Conclusion

In this paper, we studied the Coremelt attack on a single-link TCP network using a dynamical system model and Lyapunov analysis. We considered two attack scenarios, one with standard TCP and the other with a modified form of TCP, and proposed a mitigation method that increases the cost of attack. Future research directions include extending the results to more complex network configurations, other transmission protocols such as User Datagram Protocol (UDP) [25], and dynamic Coremelt attacks; and generalizing the dynamical system framework for modeling and mitigating more link-flooding DDoS attacks such as the Crossfire attack [6].

References

[1] P. J. Criscuolo, "Distributed Denial of Service Trin00, Tribe Flood Network, Tribe Flood Network 2000, and Stacheldraht," Lawrence Livermore National Laboratory, Tech. Rep. CIAC-2319, 2000.

[2] S. Specht and R. Lee, "Taxonomies of Distributed Denial of Service Networks, Attacks, Tools, and Countermeasures," Princeton University, Tech. Rep. CE-L2003-03, 2003.

[3] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of defense mechanisms against Distributed Denial of Service (DDoS) flooding attacks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[4] "Worldwide Infrastructure Security Report, Volume XI," Arbor Networks, Tech. Rep., 2016.

[5] A. Studer and A. Perrig, "The Coremelt attack," in *16th European Symposium on Research in Computer Security*, 2011, pp. 37–52.

[6] M. S. Kang, S. B. Lee, and V. D. Gligor, "The Crossfire attack," in *2013 IEEE Symposium on Security and Privacy*, 2013, pp. 127–141.

[7] J. Postel, "Transmission Control Protocol," Information Sciences Institute, University of Southern California, Tech. Rep. RFC 793, 1981.

[8] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1–14, 1989.

[9] S. H. Low, F. Paganini, and J. C. Doyle, "Internet congestion control," *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 28–43, 2002.

[10] T. Henderson, S. Floyd, A. Gurtov, and Y. Nishida, "The NewReno Modification to TCP's Fast Recovery Algorithm," IETF, Tech. Rep. RFC 6582, 2012.

[11] C. Basescu, R. M. Reischuk, P. Szalachowski, A. Perrig, Y. Zhang, H.-C. Hsiao, A. Kubota, and J. Urakawa, "SIBRA: Scalable internet bandwidth reservation architecture," in *2016 Network and Distributed System Security Symposium*, 2016.

[12] Y. Gilad and A. Herzberg, "LOT: A defense against IP spoofing and flooding attacks," *ACM Transactions on Information and System Security*, vol. 15, no. 2, p. 30, 2012.

[13] J. C.-Y. Chou, B. Lin, S. Sen, and O. Spatscheck, "Proactive surge protection: A defense mechanism for bandwidth-based attacks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 6, pp. 1711–1723, 2009.

[14] S. B. Lee, M. S. Kang, and V. D. Gligor, "CoDef: Collaborative defense against large-scale link-flooding attacks," in *9th ACM Conference on Emerging Networking Experiments and Technologies*, 2013, pp. 417–428.

[15] A. P. Athreya, X. Wang, Y. S. Kim, Y. Tian, and P. Tague, "Resistance is not futile: detecting DDoS attacks without packet inspection," in *10th Web Information System and Application Conference*, 2013, pp. 174–188.

[16] M. S. Kang, V. D. Gligor, and V. Sekar, "SPIFFY: Inducing cost-detectability tradeoffs for persistent link-flooding attacks," in *2016 Network and Distributed System Security Symposium*, 2016, p. 15.

[17] J. P. Hespanha, S. Bohacek, K. Obraczka, and J. Lee, "Hybrid modeling of TCP congestion control," in *4th International Conference on Hybrid Systems: Computation and Control*, 2001, pp. 291–304.

[18] R. J. Gibbens and F. P. Kelly, "Resource pricing and the evolution of congestion control," *Automatica*, vol. 35, no. 12, pp. 1969–1985, 1999.

[19] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: utility functions, random losses and ECN marks," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 689–702, 2003.

[20] M. Allman, V. Paxson, and E. Blanton, "TCP Congestion Control," IETF, Tech. Rep. RFC 5681, 2009.

[21] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 525–536, 2003.

[22] J. T. Wen and M. Arcak, "A unifying passivity framework for network flow control," *IEEE Transactions on Automatic Control*, vol. 49, no. 2, pp. 162–174, 2004.

[23] C. Jin, H. Wang, and K. G. Shin, "Hop-count filtering: An effective defense against spoofed DDoS traffic," in *10th ACM Conference on Computer and Communication Security*, 2003, pp. 30–41.

[24] A. Yaar, A. Perrig, and D. Song, "StackPi: New packet marking and filtering mechanisms for DDoS and IP spoofing defense," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1853–1863, 2006.

[25] J. Postel, "User Datagram Protocol," Information Sciences Institute, University of Southern California, Tech. Rep. RFC 768, 1980.