Towards Behavioral Control in Multi-Player Network Games

Andrey Lukyanenko
Helsinki Institute for Information Technology
Email: andrey.lukyanenko@hiit.fi

Andrei Gurtov
Helsinki Institute for Information Technology
Email: gurtov@hiit.fi

***Abstract*— Congestion in the routers as well as certain types of resource-exhaustion DoS attacks at the servers can be treated by differentiating packet processing according to previous history of its source. Since it is often difficult to correctly classify packets as legitimate or attack traffic, the scheduling algorithm should tolerate imprecise labeling of packets as long as on the average it punishes misbehaving sources. In this paper, we propose a game-theoretic model based on player rating and formulate the problem in terms of optimal control theory. Applying the Pontryagin maximum principle, we derive necessary control functions to encourage good behavior of network players. As an application of results, we suggest two algorithms for differentiating packet treatment in congested servers and routers.**

***Index Terms*— DDoS attack, fairness, game theory, traceback, player behavior, misbehavior.**

I. Introduction

DENIAL-OF-SERVICE (DoS) attacks on servers as well congestion in network routers represent a serious problem to the health of future Internet [4]–[8], [13], [16]–[20]. Network users behave maliciously to deny the service from others or simply selfishly just trying to obtain the maximum possible resources for itself without considering interests of others. Certain types of resource-exhaustion DoS attacks [12] could be treated by prioritizing packet processing from sources known for good behavior. However, as network users can change their behavior over time, simple approaches such as blacklisting often do not work, especially because it is difficult to classify packets as good or bad reliably.

Game theory is a useful tool to model such "tussle" in user behavior [1]–[3], [9], [10], [14], [15], [21], [22]. In this paper, we suggest a game-theoretic model of network resource sharing based on ranking of the players [11]. The model is specified in terms of optimal control theory. Applying the maximum principle of Pontryagin, we analyze the model and derive the control functions that encourage social behavior of network users. We consider the application of the results in a form of two algorithms for processing packets in network routers and servers.

The rest of the paper is organized as follows. In Section II, we make an abstract formulation of multi-player resource sharing problem. Section III provides a more focused statement of the problem in terms of optimal control theory and outlines our approach with the Pontryagin maximum principle. In Section IV, we derive the optimal state trajectory for a game of N players. Section V offers two algorithms that enable protection against DoS attacks and user selfish behavior using results of the optimal control. Simulations results are presented in Section VI. Section VII concludes the paper with a discussion on applicability of results.

A. Lukyanenko and A. Gurtov are with Helsinki Institute for Information Technology, Helsinki University of Technology, P.O. Box 9800, FIN-02015 TKK, Finland.

II. A game-theoretic model

Fist we want to formulate the problem in abstract terms of game theory. In the following section, we define a specific problem for the analysis from this abstraction. Let us have a system of N players; we number them from 1 to N. Every player interacts with other players of the system and wants to have a gain from this interaction (most likely a player wants the maximum possible gain). Our goal is to achieve fairness in interactions, and we seek not the local fairness at some moment but fairness over whole period of time. For simplicity we assume, without loss of generality, that the system starts executing at the initial moment of time 0, stops at the terminal moment of time T, and during the execution all players remain in the system.

To achieve fairness during whole period of time the system needs to maintain history of players' actions. The history of player i at the time moment t, where $i \in 1, N$, will be $h_i(t)$. At the moment 0, the history for every player is empty. With the progress of the time the history aggregates all player's actions. Let $a_i(t)$ be the action of player i at the moment t and let $A_i(t)$ be an action space ($a_i(t) \in A_i(t)$). Player's action in a sense resembles an event (in theory of probability) occurring at the moment t. $h_i(t)$ aggregates all events occurred until time t, with the corresponding order of the events ($h_i(t) = a_i(t')|_{\{t'<t\}}$, i.e., $h_i(t)$ is restriction of $a_i(t')$ to $S = \{t' < t\}$).

Now let us consider the players' interactions. Every player of the system provides some service to the system and, hence, to other players or demands some service from the system and, hence, from other players. Let's focus on a situation, when a player (provider) supplies a set of other players (consumers) with a service. We know that at a moment t any consumer i asks the provider for an action $d_i(t)$, where $d_i(t)$ is some subaction of $a_i(t)$ ($d_i(t) \in a_i(t)$). If a provider can satisfy all the demands at a moment t $(\dots, d_i(t), \dots, d_j(t), \dots)$ then there is no conflict. If the provider is not capable of serving all consumers, then this provider decides how much every consumer should get.

978-1-4244-4177-8/09/$25.00 ©2009 IEEE

The decision comes from the consumer history information ($h_i(t)$, for some i), thus consumers receive the service by the history of their behavior, proportional to their goodness for the system. The latter can be written as some function from the previous history and current event $(f_i(h_1(t), \dots, h_N(t), d_1(t), \dots, d_N(t))$ for some i).

The abstract definition (though not very strict) of the problem is thereby given. Player i wants to maximize

$$\int_0^T F_i(h_1(t), \dots, h_N(t), a_1(t), \dots, a_N(t))dt,$$

whilst the player chooses $a_i(t)$ on every step and has $h_i(t)$ – history of actions before current moment of time, where $F_i(h_1(t), \dots, h_N(t), a_1(t), \dots, a_N(t))$ the overall profit for player i at the moment t using strategy $a_i(t)$.

It is clear that dealing with such abstract definition is complicated, because player's strategies $(a_i(t))$ are defined as events, and, thereby, are dependent on the actual problem statement and the nature of the problem. We cannot expect that any real system would remember whole history of all players. Therefore, we introduce variables that let us split the task into two parts. For every player i, let $z_i(t)$ be a function that maps strategy space $A_i(t)$ onto the sector $[-1, 1]$. This function in some way resembles a random process; for every t it transfers the problem from action space to the real-value space.

This function must be constructed separately for each specific problem. We only assume that this function is well defined; and that a value -1 corresponds to the most negative actions of the player, a value 0 corresponds to neutral actions, and a value 1 corresponds to the most positive actions. By analogy, we define $k_i(t)$ to be a function that maps the space of all possible histories onto interval $(0, 1)$. We are call this function a goodness function. It represents player's history as a value from the interval $(0, 1)$ which we can easily deal with, and this function can be used to compare goodness of two players of the system. Hence, this function presents a rating of a player.

At last we can define the problem in the necessary form: Every player i wants to maximize the income function

$$\int_0^T F_i(z_1(t), \dots, z_N(t), k_1(t), \dots, k_N(t))dt \to \max,$$

where $\dot{k}_i(t) = g_i(z_1(t), \dots, z_N(t), k_1(t), \dots, k_N(t))$, $k_i(t) \in (0, 1)$, $z_i(t) \in [-1, 1]$ and $k_i(0) = k_0$. Here $z_i(t)$ can be defined as a vector to split interactions of player i with other players of the system. We, however, treat this variable here only as an accumulated function of all interactions onto $[-1, 1]$. We assume that $z_i(t)$ and $k_i(t)$ accurately correspond to real task events for every i.

III. Problem statement

In this section, we state the specific problem we are interested in. Let us have N players, each player interacts only with a central server. Every player has a goodness variable $k_i(t)$ at time moment t. Initially $k_i(0) = k_i^0$. The central server is ready to give a player i a service value proportional to the player's goodness value (i.e. $\frac{k_i(t)}{X(t)}$, where $X(t) = \sum_{i=1}^N k_i(t)$ and, for simplicity, we assume that server capacity is 1). This is not a strict rule; a player can get at most twice more than what the server suggests. Hence, the goodness value changes depending on player's behavior $u(t) \in [0, 2]$, where $u(t)$ corresponds to the factor of what server suggests to take. Furthermore, we define the change of goodness value by the following formula $\dot{k}_i(t) = k_i(t)(1 - k_i(t))(1 - u(t))$. Note, that if $1 - u(t) \in [-1, 1]$ for all t and $k_i(0) \in (0, 1)$, then $k_i(t) \in (0, 1)$ is true for all t.

Equation $\dot{k}_i(t) = k_i(t)(1 - k_i(t))(1 - u(t))$ and goodness value $k_i(t)$ have the same meaning as equation $\dot{z}_i(t) = 1 - u(t)$ and the goodness value $\frac{1}{1+\exp(-z_i(t))}$. In the later form, $\dot{z}_i(t) = 1 - u(t)$ can be considered as accumulation of individual behaviors over time. Function $\frac{1}{1+\exp(-z_i(t))}$ maps this accumulated history from $(-\infty, \infty)$ space to $(0, 1)$ space as goodness value. It is possible to define other such mapping functions from $(-\infty, \infty)$ space to $(0, 1)$ space, but such functions are a subject of our future work.

We obtain the following differential game to solve for all i:

$$\begin{cases} \int_0^T \frac{k_i(t)}{X(t)} u(t)dt \to \max \\ \dot{k}_i(t) = k_i(t)(1 - k_i(t))(1 - u(t)) \\ 0 \le u(t) \le 2 \\ k_i(0) = k_i^0 \\ 0 < k_0 < 1. \end{cases} \tag{1}$$

Formulation above is game-theoretic, but we want to solve a more specific problem. We wish to observe player behavior under a DoS attack on the server when the number of players is high and proportion $\frac{k_i(t)}{X(t)}$ which player i receives is not so high. In this case, we expect that with increasing number of players N, the value of $X(t)$ tends to some specific function of t (and, loses the dependency on individual player behavior).

Consider a situation when some player sees the system under a DoS attack. Then we can put $X(t) = x(t) + Y_{N-1}(t)$, where $x(t)$ is behavior for the player, and $Y_{N-1}(t)$ is the behavior for the rest. Hence, $Y_{N-1}(t)$ should be much bigger than $x(t)$ for this situation to be a DoS attack. Therefore, we are not interested in situations when $Y_{N-1}(t)$ is similar to $x(t)$. On the other hand, $Y_{N-1}(t)$ is not easy to maintain on a high level. Initially, it is on the level of $k_0 * (N - 1)$ and with bad behavior for the rest players it tends to zero fast. Hence, for an attacker it is good to maintain $Y_{N-1}(t)$ on a high level (consider it C), that every individual player suffers from it. We leave the task to maintain the high value for $Y_{N-1}(t)$ to an attacker and take a look at the best strategy for an individual player to achieve the best possible service.

From the discussion above, we will analyze (1) with an assumption that $X(t)$ is a constant (i.e., $X(t) = x(t) + Y_{N-1}(t) \approx x(t) + C \approx C$, as $C >> x(t)$). The complexity of the analysis does not seem to increase significantly if we take $X(t)$ as a function of time. Assuming $X(t)$ is a constant,

the problem becomes a control theory problem:

$$\begin{cases} \int_0^T x(t)u(t)dt \to \max \\ \dot{x}(t) = x(t)(1-x(t))(1-u(t)) \\ 0 \le u(t) \le 2 \\ x(0) = k_0 \\ 0 < k_0 < 1, \end{cases} \tag{2}$$

which is possible to solve with Pontryagin maximum principle.

IV. Analysis

We define Hamiltonian as $H(x(t), u(t), \psi(t)) = -\psi_0 x(t)u(t) + \psi_1 x(t)(1-x(t))(1-u(t))$, where $\psi(t)$ is a vector of costate variables, $x(t)$ is a vector of coordinates (with an additional coordinate x_0, such that $\dot{x}_0(t) = f_0(x,u) = -x(t)u(t)$), and $u(t)$ is a control function. Costate variables are strictly connected to the coordinates with the following equations

$$\dot{x}(t) = \frac{\partial H}{\partial \psi} \tag{3}$$

$$\dot{\psi}(t) = -\frac{\partial H}{\partial x} \tag{4}$$

Additional initial conditions are given for costate variables; we fix the initial and the terminal time moments and leave free the final position of the coordinate. These conditions — transversality conditions, are defined as $\psi(T) = \{-1, 0\}$. Immediately we observe that as H is independent of x_0 and $\psi_0(T) = -1$, then $\psi_0(t) = -1$ for any t.

Following the maximum principle of Pontryagin we assume knowledge of the optimal control variable $u^*(t)$. Thus, from the optimal control we can strictly determine the optimal state trajectory $x^*(t)$ from (2), and therefore from $u^*(t)$ and $x^*(t)$ we obtain the optimal costate variable ψ^*. Then maximum principle states that

$$H(x^*, u^*, \psi^*) = \sup_{u \in U} H(x^*, u, \psi^*); \tag{5}$$

additionally the permissible control U can have a finite number of points of discontinuity. As u^* we can take any control that satisfy condition (5), i.e.,

$$u^*(t) = \begin{cases} 0, \text{ if } 1 - (1-x^*(t))\psi_1^*(t) \le 0 \\ 2, \text{ if } 1 - (1-x^*(t))\psi_1^*(t) > 0; \end{cases} \tag{6}$$

as $H(x^*, u, \psi^*) = x^*(1 - (1-x^*)\psi_1^*)u + x^*(1-x^*)\psi_1^*$. Recall, that if we know u^* then we can precisely find the optimal trajectory x^*

$$\dot{x}^*(t) = \begin{cases} x^* - x^{*2}, \text{ if } (1-x^*)\psi_1^* \ge 1 \\ x^{*2} - x^*, \text{ if } (1-x^*)\psi_1^* < 1. \end{cases} \tag{7}$$

Theorem 1 (Optimal trajectory): For the task above, the motion starts from the point $x^*(0) = k_0$ with control $u^*(t) = 0$. Reaching the coordinate $x^*(t^*) = 2x(T)$ the control adjusts to $u^*(t) = 2$ and the coordinate starts to decrease till $x(T)$ if $t^* > 0$, otherwise the control is $u^*(t) = 2$ on $[0, T]$.

Proof: See Appendix . ∎

Thus we get $z^* = 1 + \sqrt{e^T \frac{k_0}{1-k_0} + 1}$ and during the time $t^* = T - \ln(z^*)$ the control is $u^* = 0$, after that during the time $t^* = \ln(z^*)$ the control is $u^* = 2$. The payoff in that case is $2\ln(z^*)$, i.e., $J = 2\ln\left(1 + \sqrt{e^T \frac{k_0}{1-k_0} + 1}\right)$. Therefore, the average of relations of the payoff to the time is $J/T \to 1$ while $T \to \infty$. For small values of T if the player stays in the system for a short period of time, then the relation goes to 2, i.e., the player has selfish behavior. If we fix the smallest time $T > \ln 2$, which the player stays in the system before starting to be dealt with, then we can find the corresponding value of k_0 which restricts player's selfish behavior.

V. Game-theoretic algorithms

In this section, we introduce two algorithms that realize the benefits shown in the model analysis in Section IV. We deal with misbehavior of users who communicate with one central resource, including DoS attacks and selfish behavior. These algorithms represent congestion control mechanisms for the server or router.

The first algorithm provides the direct use of the model. The central server has N users who communicate with it, a user i has goodness k_i; for new users the goodness value becomes k_0. u is the function that transfers from user's action space into domain $[-1, 1]$ as was described earlier.

We assume that all incoming messages are put into a priority queue not directly depending on the goodness value, but on the elapsed time before serving, which initially depends on the goodness value. To be precise, if we know the average load of the system (including loads that come from new unidentified users), then we can estimate how long should every user with corresponding goodness value wait for service. Knowing this time we put the request from the user into the priority queue, where it waits for this time before the receiving service.

The process of the input queue (lines 5-13) should work fast as long as we only rearrange messages from one queue to another. It can be also outsourced to an external device.

The events are served in the central resource depending on user goodness value k. If a user is already known then priority queue gets the message with k of this user. If a user is unknown then the user gets new goodness value k_0. If the system suffers from a DDoS attack or congestion then this initial value will be small. The new user who gets the goodness value k_0 needs to wait until the first attempt to be served, but if the events that come from the user are good enough for the system (for example, not a part of DDoS attack), then the user's k-value goes up and during next interactions the user will be served faster. All good users of the system that is under DDoS attack are served quite fast. If a new user comes during DDoS attack or congestion, then the user should wait some warm-up time to be served faster, of course, if the user behaves well for the system. As

Algorithm 1 Direct congestion control mechanism.

Require: N, $k_1, \ldots, k_N$, k_0, u

loop
  if server terminates it work **then**
    return
  end if
  if input queue has an event e from a user i at the top **then**
    if i is a new user **then**
      $N \leftarrow N + 1$
      $i \leftarrow N$
      $k_i \leftarrow k_0$
    end if
    $u_i \leftarrow u(e)$
    $k_i += k_i(1 - k_i)u_i$
    put event e from user i to the priority queue with k_i
  else
    process an event at the top of the priority queue
  end if
  recompute k_0 depending on the current load of the server
end loop

we can see all users who spoof their IP addresses always served with goodness value k_0, it is important to keep this value reflecting the load of the system.

The second congestion mechanism is related to the first but instead of dealing with users it deals with network routers. The idea is to extend the marking algorithm [7]. Law et al. suggest to construct an attack graph with help of routers. When a router has a message for the victim host (in case of a DDoS attack), it puts a router's mark with some probability to the message. The victim host collects messages and during the warm up creates the attacking graph using routers as the nodes. After warming up it continues to update the graph. We want to add additional information to the attacking graph: we will add the k_i value for every router i and it will again represent the goodness value: the goodness not of the end host but of the router itself. Hence, we can serve less messages that come from the routers that send more attacking/misbehaving traffic.

The Algorithm 2 can be run on a server, which wants to deal with DDoS attacks and congestion, but we can go further and add this mechanism to the routers in the network. Then these routers will start to propagate all traffic depending on user behavior, which prevents misbehavior of users from the start.

VI. Simulation

During simulation we created a network consisting of M autonomous systems (AS) with N routers in each using BRITE network generator (with additional code modifications to get special network settings). We placed one server in a random node with service time 0.1 seconds per message. Based on the place of this server we created an attacking

Algorithm 2 Congestion control mechanism using marking in routers.

Require: u

collect attacking graph information for some time, get N, $k_0, \ldots, k_N$
loop
  if server terminates its work **then**
    return
  end if
  if input queue has an event e from a router i at the top **then**
    if i is a new router **then**
      extend the attacking graph with a new router
      $N \leftarrow N + 1$
      $i \leftarrow N$
      $k_i \leftarrow k_0$
    end if
    $u_i \leftarrow u(e)$
    $k_i += k_i(1 - k_i)u_i$
    put event e from router i to the priority queue with k_i
  else
    process an event at the top of the priority queue
  end if
  recompute k_0 depending on the current load of the server
end loop

graph, where we placed zombie nodes (25% of the network) with message generation time 10 ms and 5 clients with message generation time 10 s.

The purpose of zombie nodes was to make server suffer from congestion (hence making it not reachable for clients). We implemented 4 main server models. First is a normal server, which puts every incoming message in the server queue, and takes messages only from the top of the queue. The second model implements our first algorithm with fixed k_0 value. Hence every client coming to the server sees the same initial value for k. The third model implements the first algorithm with changing k_0 value (which we set to average k value for all clients communicating with server). The last model considers situation when we do not count messages that are not taken into the queue.

In models 2-4 we implement a priority queue based on values k for every client. We run simulation during 12000 seconds for network of 20 AS with 20 routers each. In future we plan to extend our simulation to larger networks (here to compensate for the network scale we made the server run quite slow).

We left simulation study of the second algorithm for the future; here we are not considering spoofing of IP addresses. Thus, we always know the message source and know if these messages are in the queue or not. We are interested to see if our algorithm allows a good user to maintain access to a server. We are studying three situations of incoming message analysis. The good one is when we almost clearly

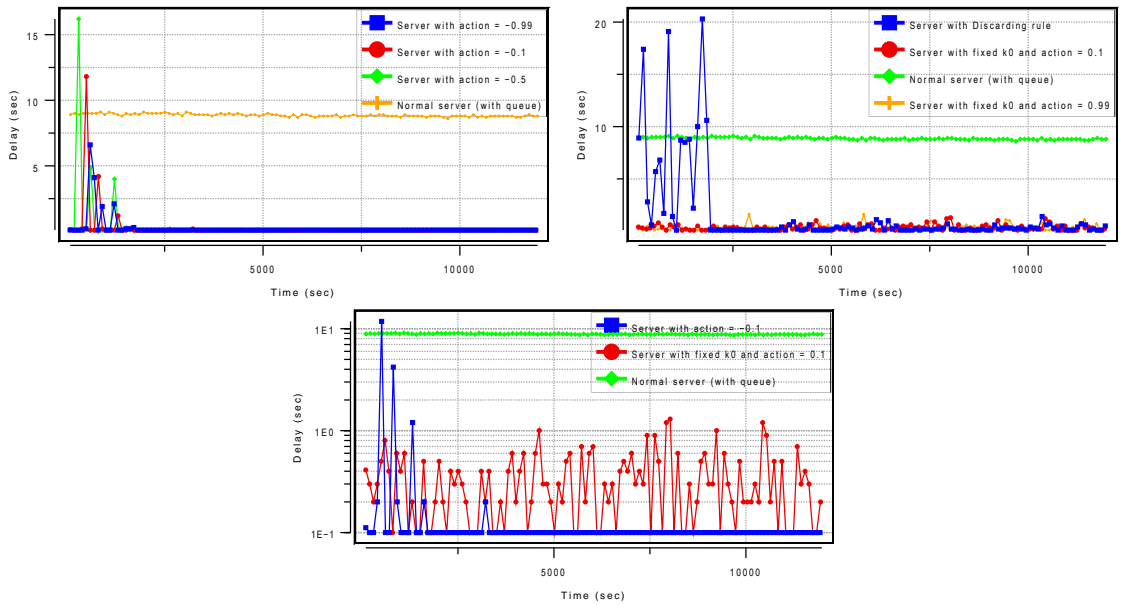

Fig. 1. Simulation results.

can distinguish a good packet from a bad one (we put action $= -0.99$ for misbehaving users and action $= 0.99$ for good users. The average one is where we put action $= -0.5$ for bad users and action $= 0.5$ for good users. In the worst one, the action is $= -0.1$ for bad users and $= 0.1$ for good users.

The results of our simulations are shown in Figure 1. The first graph shows the speed of response for Algorithm 1 with changeable k_0. An the beginning, good clients need some time to start operating faster than in a normal case. It is a result of bad clients coming first to the uncongested system and receiving high initial k values. After some operating time, good clients get more access to the system and start to receive a response almost immediately (in less than a second, compared to 8 seconds in normal server case).

On the second graph, by Algorithm 1 clients get a response, but the initial value k_0 is fixed at 0.5 (red and orange lines). Additionally, there is a blue line for an algorithm when we do not count packets that were sent from clients already in the queue (hence less information received for recomputation of k). This works worse than protocols on the first graph, while it still converges to some small value.

Algorithms with a fixed initial k_0 value show better performance in the beginning (bad clients do not get high initial ranks when coming first), but in this case the algorithm is not as stable as in adjustable k_0 case. The third graph with logarithmic scale shows the difference in performance better. We can see that the protocol from Algorithm 1 with adjustable k_0 (the blue line) responds slower in the beginning but afterwards converges to some minimal value. The protocol with fixed k_0 (the red line) does not converge to the minimal value with significant variation. The normal server works on the same level during the simulation (the green line).

VII. Conclusion

In this article we proposed two algorithms that force users to behave fairly — selfish behavior leads to restrictions in user actions. Analysis shows that if a user wants to stay in the system during long period of time then the behavior of the user tends to be fair, but short presence leads to misbehavior. If we know that the user wants to stay more than at least $\ln 2$ units of time (which corresponds to $\ln 2$ number of steps in an iteration model) then we can find such small k_0 that prevents selfish behavior.

The observation corresponds to a reasonable logical result; if the user wants to gain maximum from a system during long enough period of time (for a small initial goodness value) then the user should first increase the goodness value till some point, and only after that gain maximum from it. This can help to overcome the whitewashing problem. If a user coming to a system increases the goodness value first (otherwise the user would not be served) then there is no gain from leaving the system and returning with a smaller goodness value and increasing it after that again. Whitewashing in some sense resembles a DDoS attack when the source IP address is spoofed, and hence the system sees new users in the input queue. The idea, also, suits to defend against Sybil attacks. In a Sybil attack, when a user creates multiple nodes in the system, the increase of goodness value from k_0 for every node can be made very hard.

At last it is worth mentioning, that when two users directly interact in the system there should not be a possibility that they can increase goodness values endlessly. In other words, during user interactions an increase of the goodness value of the first decreases the same amount (or greater) the goodness value of the second user. In our algorithm the players interact only with a central server (non-player), hence no actual player interactions occur. However, a P2P system should

prevent a user from adding two nodes and using the false interaction between them to increase the goodness value of one of the nodes endlessly. Thus the optimal control problem suits better for central server architecture, while for P2P system it should be studied more as an original game theory model.

REFERENCES

[1] A. Akella, S. Seshan, R. Karp, S. Shenker, and C. Papadimitriou. Selfish behavior and stability of the Internet: a game-theoretic analysis of TCP. *SIGCOMM Comput. Commun. Rev.*, 32(4):117–130, 2002.

[2] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 228–236, New York, NY, USA, 2004. ACM.

[3] R. Garg, A. Kamra, and V. Khurana. A game-theoretic approach towards congestion control in communication networks. *SIGCOMM Comput. Commun. Rev.*, 32(3):47–61, 2002.

[4] M. T. Goodrich. Efficient packet marking for large-scale IP traceback. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 117–126, New York, NY, USA, 2002. ACM.

[5] J. Ioannidis and S. M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. In *Proceedings of Network and Distributed System Security Symposium, Catamaran Resort Hotel San Diego, California 6-8 February 2002*, 1775 Wiehle Ave., Suite 102, Reston, VA 20190, February 2002. The Internet Society.

[6] C. Jin, H. Wang, and K. G. Shin. Hop-count filtering: an effective defense against spoofed DDoS traffic. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 30–41, New York, NY, USA, 2003. ACM.

[7] T. K. Law, J. C. Lui, and D. K. Yau. You can run, but you can't hide: An effective statistical methodology to trace back DDoS attackers. *IEEE Transactions on Parallel and Distributed Systems*, 16(9):799–813, 2005.

[8] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: Source address validity enforcement protocol, 2001.

[9] P. Liu, W. Zang, and M. Yu. Incentive-based modeling and inference of attacker intent, objectives, and strategies. *ACM Trans. Inf. Syst. Secur.*, 8(1):78–118, 2005.

[10] A. Mahimkar and V. Shmatikov. Game-based analysis of denial-of-service prevention protocols. In *CSFW '05: Proceedings of the 18th IEEE workshop on Computer Security Foundations*, pages 287–301, Washington, DC, USA, 2005. IEEE Computer Society.

[11] V. Mazalov, I. Falko, A. Gurtov, and A. Pechnikov. Equilibrium in a P2P-system. In *Proc. of AMICT'07*, June 2007.

[12] J. Mirkovic and P. Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *SIGCOMM Comput. Commun. Rev.*, 34(2):39–53, 2004.

[13] T. Peng, C. Leckie, and K. Ramamohanarao. Adjusted probabilistic packet marking for ip traceback. In *NETWORKING '02: Proceedings of the Second International IFIP-TC6 Networking Conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; and Mobile and Wireless Communications*, pages 697–708, London, UK, 2002. Springer-Verlag.

[14] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. On selfish routing in internet-like environments. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 151–162, New York, NY, USA, 2003. ACM.

[15] S. J. Shenker. Making greed work in networks: a game-theoretic analysis of switch service disciplines. *IEEE/ACM Trans. Netw.*, 3(6):819–831, 1995.

[16] M. Sung and J. Xu. IP traceback-based intelligent packet filtering: A novel technique for defending against internet ddos attacks. In *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 302–311, Washington, DC, USA, 2002. IEEE Computer Society.

[17] M. Waldvogel. Gossib vs. ip traceback rumors. In *ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference*, page 5, Washington, DC, USA, 2002. IEEE Computer Society.

[18] M. Walfish, M. Vutukuru, H. Balakrishnan, D. Karger, and S. Shenker. DDoS defense by offense. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 303–314, New York, NY, USA, 2006. ACM.

[19] J. Xu and W. Lee. Sustaining availability of web services under distributed denial of service attacks. *IEEE Transactions on Computers*, 52(2):195–208, 2003.

[20] D. K. Y. Yau, J. C. S. Lui, F. Liang, and Y. Yam. Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles. *IEEE/ACM Trans. Netw.*, 13(1):29–42, 2005.

[21] H. Zhang, D. Towsley, and W. Gong. TCP connection game: A study on the selfish behavior of TCP users. In *ICNP '05: Proceedings of the 13TH IEEE International Conference on Network Protocols*, pages 301–310, Washington, DC, USA, 2005. IEEE Computer Society.

[22] Y. Zhou and H. Sethu. On achieving fairness in the joint allocation of processing and bandwidth resources: principles and algorithms. *IEEE/ACM Trans. Netw.*, 13(5):1054–1067, 2005.

APPENDIX
PROOF OF THEOREM 1.

Proof: In contrast to the control variable, the trajectory variable should be absolutely continuous; in points of discontinuity of control ($(1-x^*)\psi_1^* = 1$) we define the value of $x^*(t)$ as the right-hand limit.

We saw that the trajectory changes depending on the control variable. If $u^* = 0$ then the trajectory monotonically increases, if $u^* = 2$ then the trajectory monotonically decreases. If after an interval of monotonicity, behavior continues with an interval of absolutely same monotonous behavior (those intervals are separated only by the point of discontinuity of the control), then we can concatenate these intervals changing the value of the control variable in the discontinuity point to the value on the intervals of monotonous behavior. Recall, that a change of the control in a finite set of points does not affect the optimality of that control.

Thus, consider a sequence of points $t_1, \dots t_{N-1}$ where monotonicity changes. Define $t_0 = 0$ and $t_N = T$. Additionally we suppose that the control remains unchanged on half-intervals $(t_i, t_{i+1}]$, $0 \le i \le N-1$. Let $x_i = x^*(t_i)$, then for $t \in (t_i, t_{i+1}]$ $\forall i$ is fulfilled

$$u^*(t) = \begin{cases} 0, \text{ if } (1-x^*)\psi_1^* \ge 1 \\ 2, \text{ if } (1-x^*)\psi_1^* < 1. \end{cases} \tag{8}$$

$$x^*(t) = \begin{cases} \dfrac{x_i}{x_i + (1-x_i)e^{-(t-t_i)}}, \text{ if } (1-x^*)\psi_1^* \ge 1 \\ \dfrac{x_i}{x_i + (1-x_i)e^{t-t_i}}, \text{ if } (1-x^*)\psi_1^* < 1. \end{cases} \tag{9}$$

The control u^* and the corresponding trajectory x^* thoroughly define the costate variable. From (4) we have

$$\dot{\psi}_1^* = -u^* - (1-2x^*)(1-u^*)\psi_1^*,$$

provided that $\psi_1(T) = 0$.

$$\left(\psi_1^* e^{\int_0^t (1-2x^*)(1-u^*)d\tau}\right)'_t = -u^* e^{\int_0^t (1-2x^*)(1-u^*)d\tau}. \tag{10}$$

In turn knowing the connection between x^* and u^* we get an equality

$$\frac{dx^*}{x^*(1-x^*)} = (1-u^*)dt.$$

which simplifies the following equation

$$e^{\int_0^t (1-2x^*)(1-u^*)d\tau} = e^{\int_{k_0}^{x^*} \frac{(1-2x^*)dx}{x^*(1-x^*)}} = \frac{x^*(1-x^*)}{k_0(1-k_0)}.$$

If we apply previous equation to equation (10) we obtain

$$(\psi_1^* x^*(1-x^*))_t' = -u^* x^*(1-x^*).$$

$$\psi_1^* x^*(1-x^*) = C - \int_0^t u^*(\tau)x^*(\tau)(1-x^*(\tau))d\tau.$$

Taking into account the transversality condition, we obtain

$$\psi_1^*(1-x^*) = \frac{1}{x^*(t)} \int_t^T u^*(\tau)x^*(\tau)(1-x^*(\tau))d\tau.$$

The value of integral on half-interval $(t_i, t_{i+1}]$ is calculated as following

$$\begin{aligned} &\int_{t_i}^{t_{i+1}} u^*(\tau)x^*(\tau)(1-x^*(\tau))d\tau = \\ &\qquad u^*(t_{i+1}) \int_{t_i}^{t_{i+1}} x^*(\tau)(1-x^*(\tau))d\tau = \\ &\qquad u^*(t_{i+1}) \int_{t_i}^{t_{i+1}} \frac{x_i(1-x_i)e^{\tau-t_i}d\tau}{(x_i+(1-x_i)e^{\tau-t_i})^2}. \end{aligned} \tag{11}$$

The latter is possible since if the control $u^*(t_{i+1}) = 0$ then the expression under the integral sign, as a multiplier, does not change the result. Again making the change of variable in the expression under the integral sign $z = x_i + (1-x_i)e^{\tau-t_i}$, we get $dz = (1-x_i)e^{\tau-t_i}d\tau$ and

$$\begin{aligned} &\int_{t_i}^{t_{i+1}} \frac{x_i(1-x_i)e^{\tau-t_i}d\tau}{(x_i+(1-x_i)e^{\tau-t_i})^2} \\ &= \int_{z(t_i)}^{z(t_{i+1})} \frac{x_i dz}{z^2} = \frac{x_i}{z(t_i)} - \frac{x_i}{z(t_{i+1})} \quad = x_i - x_{i+1}. \end{aligned}$$

Thus

$$\int_{t_i}^{t_{i+1}} u^*(\tau)x^*(\tau)(1-x^*(\tau))d\tau = u^*(t_{i+1})(x_i - x_{i+1}).$$

By analogy if $t \in (t_i, t_{i+1}]$

$$\int_{t}^{t_{i+1}} u^*(\tau)x^*(\tau)(1-x^*(\tau))d\tau = u^*(t_{i+1})(x^*(t) - x_{i+1}).$$

Uniting the latest equation with equations of the optimal control and the optimal trajectory, we obtain a solution on every half-interval $(t_j, t_{j+1}]$

$$u^*(t) = \begin{cases} 0, \text{ if } u^*(t_{j+1})(x^*(t) - x_{j+1}) \\ \qquad + \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1}) \geq x^*(t) \\ 2, \text{ if } u^*(t_{j+1})(x^*(t) - x_{j+1}) \\ \qquad + \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1}) < x^*(t). \end{cases} \tag{12}$$

Since we know the value of the control variable on that half-interval, we get

$$u^*(t) = \begin{cases} 0, \text{ if } x^*(t) \leq \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1}) \\ 2, \text{ if } x^*(t) < 2x_{j+1} - \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1}). \end{cases} \tag{13}$$

and the equation of the optimal trajectory

$$x^*(t) = \begin{cases} \frac{x_j}{x_j + (1-x_j)e^{-(t-t_j)}}, \text{ if } x^*(t) \leq \\ \qquad \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1}) \\ \frac{x_j}{x_j + (1-x_j)e^{t-t_j}}, \text{ if } x^*(t) < 2x_{j+1} - \\ \qquad \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1}). \end{cases} \tag{14}$$

What remains is to find a set of all points $t_1, \dots t_{N-1}$ (we recall that $t_0 = 0$ and $t_N = T$). We analyze the motion backward from the end. Consider the last half-interval $(t_{N-1}, t_N]$. If we assume that the control on this half-interval is $u^*(t) = 0$, then the only condition to satisfy is $x^*(t) \leq 0$ leading to contradiction since $x^*(t) > 0$ by definition. It means that on the last half-interval $u^*(t) = 2$.

Let t_{N-1}^* be the moment of time, that $t_{N-1}^* < T$ and for $t \in (t_{N-1}^*, T]$ $u^*(t) = 2$, while for the point itself t_{N-1}^* the condition $x^*(t_{N-1}^*) < 2x_{j+1} - \sum_{i=j+1}^{N-1} u^*(t_{i+1})(x_i - x_{i+1})$ breaks, where $j = N-1$ for the last half-interval. This condition breaks because of the continuity of x^* when $x^*(t_{N-1}^*) = 2x(T)$. Hence we can get t_{N-1}^*, if $t_{N-1}^* \leq 0$; then we put $t_{N-1}^* = 0$ and consider that for $t \in [0, T]$ $u^*(t) = 2$. We explore now the case when $t_{N-1}^* > 0$. There exists previous half-interval $t \in (t_{N-2}^*, t_{N-1}^*]$ where the control $u^*(t) = 0$. Let us find the point t_{N-2}^*, that $t_{N-2}^* < t_{N-1}^*$, $u^*(t) = 0$ for $t \in (t_{N-2}^*, t_{N-1}^*]$ and inequality $x^*(t_{N-2}^*) \leq 2(x_{N-1} - x_N)$ breaks. I.e., $x^*(t_{N-2}^*) \leq 2X(T)$ breaks. We know that on the right side of half-interval $(t_{N-2}^*, t_{N-1}^*]$ $x^*(t_{N-1}^*) = 2x(T)$, and for $u(t) = 0$ the decrease of time leads only to decrease of coordinate, i.e. $t_{N-2}^* = 0$.

Omitting all calculus, we can say that the motion starts from the point $x^*(0) = k_0$ with control $u^*(t) = 0$. Reaching the coordinate $x^*(t^*) = 2x(T)$ the control adjusts to $u^*(t) = 2$ and the coordinate starts to decrease till $x(T)$ if $t^* > 0$, otherwise the control is $u^*(t) = 2$ on $[0, T]$.

Let us find the case when $t^* \leq 0$. We know that moving from the point $2x(T)$ to the point $x(T)$ the control is $u^*(t) = 2$. Thus,

$$x(T) = \frac{2x(T)}{2x(T) + (1-2x(T))e^{T-t^*}} \tag{15}$$

from where

$$t^* = T - \ln\left(1 + \frac{1}{1 - 2x(T)}\right). \tag{16}$$

The condition $t^* \leq 0$ is fulfilled if

$$x(T) \geq \frac{e^T - 2}{2(e^T - 1)}. \tag{17}$$

Recalling that

$$x(T) = \frac{k_0}{k_0 + (1 - k_0)e^T}, \tag{18}$$

we set restriction

$$k_0 \geq 1 - \frac{1}{e^T - 1} \tag{19}$$

wherein the optimal control will be $u^* = 2$, i.e., selfish.

Let us find the point t^* under the condition that $k_0 < 1 - \frac{1}{e^T - 1}$ and the condition that we have the following control $u^* = 0$ on $[0, t^*]$ and control $u^* = 2$ on $(t^*, T]$.

$$x(t^*) = 2x(T) = \frac{k_0}{k_0 + (1 - k_0)e^{-t^*}}. \tag{20}$$

Substituting (16) into the last equation and producing a change of variable $z = 1 + \frac{1}{1 - 2x(T)}$ (and $x(T) = \frac{z-2}{2(z-1)}$) we get

$$\frac{z - 2}{z - 1} = \frac{k_0 e^T}{k_0 e^T + (1 - k_0)z}. \tag{21}$$

The solution of the latest equation is equal to the solution of quadratic equation

$$z^2 - 2z - e^T\left(\frac{k_0}{1 - k_0}\right) = 0. \tag{22}$$

■