Proceedings of the 2010 Winter Simulation Conference
B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, eds.

METHODOLOGIES FOR EVALUATING GAME THEORETIC DEFENSE AGAINST DDOS ATTACKS

Tanmay Khirwadkar, Kien C. Nguyen, David M. Nicol, Tamer Başar

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 W Main St.
Urbana, IL 61801, USA

ABSTRACT

Distributed Denial of Service (DDoS) attacks on the Internet are used by attackers to be a nuisance, make a political statement (e.g. the 2009 attack against Estonia), or as a weapon of an Internet extortionist. Effective defense against these is a crucial study area, where advanced simulation techniques play a critical role, because of the enormous number of events involved. This paper considers a methodology for evaluating a game-theoretic defense against DDoS. We first describe a basic form of the defense, note the performance limitations suffered by a naive implementation, and then consider methodologies in which a parallelized approach may accelerate performance.

1 INTRODUCTION

A Distributed Denial-of-Service attack (DDoS attack) is an attack launched from multiple computers in a network to flood the resources of a targeted system, and thus making it less accessible to the intended users. The computers launching attacks are called zombies, which could be regular hosts that have been compromised by the attacker. Many network-based countermeasures against DDoS have been proposed and simulated; a survey can be found in (Peng, Leckie, and Ramamohanarao 2007). In this work we consider issues related to evaluating the *pushback* defense, a mechanism first proposed in (Mahajan et al. 2002).

As described in (Mahajan et al. 2002, Ioannidis and Bellovin 2002), pushback is a mechanism that allows routers in a network to cooperate in *aggregate-based congestion control* (ACC). An aggregate is defined to be a collection of packets that share a common property or parameter, such as ICMP ECHO packets or packets with the same destination IP address. The properties or parameters used to identify an aggregate are called *attack signatures*. Based on aggregates, traffic and packets are divided into three different categories: "bad", "poor", and "good". Bad traffic is that generated by the attackers. Poor traffic is from legitimate users but shares the same attack signatures. Finally, good traffic does not match the attack signatures but may suffer from the congestion. In local ACC, an individual router identifies the aggregates that causes the congestion and tries to cut down the throughput of these aggregates. In pushback, a router can request adjacent upstream routers to rate-limit some aggregates. This way, the system can save the bandwidth that would otherwise be wasted if packets in these aggregates were dropped downstream. Furthermore, if the DDoS attack traffic comes from a few upstream links, pushback helps protect poor traffic from congestion due to attack traffic.

From now on, we will use "Attacker" to refer to the DDoS attacker and all the zombies under its control, and "System" to refer to all the routers taking part in the pushback mechanism. When the Attacker launches DDoS attacks, it has at its disposal a number of strategies to choose from. Among these are the set of zombies, the set of targeted computers, and the attack protocols and traffic patterns. Similarly, the System can also change the pushback parameters such as the congestion checking time, the target drop rate, and the aggregate pattern. For each pair of strategies of the Attacker and the System, the payoffs for each of them can be formulated based on the bandwidth occupied by Attacker, the bandwidth used by the legitimate users, and the costs of attacking and defending. It thus can be seen that there is a game situation between the Attacker and the System, where each player tries to maximize its own payoff against all the possible strategies of the opponent.

In (Liu, Zang, and Yu 2005), DDoS attacks are modeled as a Bayesian game among the Attacker, the System, and legitimate users. With such a game formulation, in order to compute a Nash equilibrium pure or mixed strategy, each player has to have full knowledge of payoff. The paper also mentions a repeated mechanism where at each step, each player makes the best response to current strategies of other players. Although this mechanism allows each player to

proceed without necessarily knowing others's payoff matrices, it works well only when the game has a pure strategy Nash equilibrium.

We examine a repeated game model based on the *fictitious play* (FP) process for pushback defense. In a FP process, each player observes all the actions up to present and makes estimate of the mixed strategy of the opponent's actions. At each stage, she updates this estimate and plays the pure strategy that is the best response to the estimate. It can be seen that in a FP process, if one plays a fixed strategy (either of the pure or mixed type), the opponent's strategy will converge to the best response to this fixed strategy. Furthermore, it has been shown that, for many classes of games, such a FP process will finally render both players playing the Nash equilibrium.

The rest of the paper is organized as follows. In Section 2, we present the setup and mathematical basis of the fictitious play model. Next, we present the implementation details of the simulations in Section 3. The simulation results are shown in Section 4, and a discussion concerning methodology for parallelization in Section 5. Finally, some concluding remarks will end the paper.

2 FICTITIOUS PLAY MODEL

In this section we present an overview of static games and fictitious play, where player P_1 has m and player P_2 has n possible actions (pure strategies) (Shamma and Arslan 2004, Shamma and Arslan 2005, Nguyen, Alpcan, and Başar 2009, Nguyen, Alpcan, and Başar 2010b).

2.1 Static Games

We first introduce a one-shot nonzero-sum version of the games, which we will refer to as static games. In equations written for the generic player P_i, $i = 1,2$, we use k to denote m or n. Denote by $p_1 \in \Delta(m)$ and $p_2 \in \Delta(n)$ a pair of mixed strategies for P_1 and P_2, respectively, where $\Delta(k)$ is the simplex in $\mathbb{R}^k$ ($\mathbb{R}$ is the set of real numbers), i.e.,

$$\Delta(k) \equiv \left\{ s \in \mathbb{R}^k | s_j \geq 0, j = 1,\ldots,k, \ \sum_{j=1}^{k} s_j = 1 \right\}. \tag{1}$$

For a static game, player i selects an integer action v_i according to the mixed strategy p_i. The (instant) payoff for player P_i is $v_i^T M_i v_{-i}$, where we use $v_i^{(j)}, j = 1,\ldots,k$, to indicate the jth vertex of the simplex $\Delta(k)$ (For example, when $k = 2$, $v_i^{(1)} = [1\ 0]^T$ for the first action, and $v_i^{(2)} = [0\ 1]^T$ for the second action)[1]. For a pair of mixed strategies (p_1, p_2), the utility functions are given by the expected payoffs:

$$\begin{aligned} U_i(p_i, p_{-i}) &= E\left[v_i^T M_i v_{-i}\right] \\ &= p_i^T M_i p_{-i}, \end{aligned} \tag{2}$$

where M_i is the payoff matrix of $P_i, i = 1,2$ (Note that M_1 is of dimension $m \times n$ and M_2 $n \times m$.) This standard formulation of the payoff functions leads to a FP process that will be referred to as *classical FP*.

Now, the *best response* mappings $\beta_1 : \Delta(n) \to \left\{v_1^{(1)}, v_1^{(2)}, \ldots, v_1^{(m)}\right\}$ and $\beta_2 : \Delta(m) \to \left\{v_2^{(1)}, v_2^{(2)}, \ldots, v_2^{(n)}\right\}$ are defined as:

$$\beta_i(p_{-i}) = \arg \max_{v_i \in \left\{v_i^{(1)}, v_i^{(2)}, \ldots, v_i^{(k)}\right\}} U_i(v_i, p_{-i}). \tag{3}$$

Note that given p_{-i}, the best response mapping can be set-valued, i.e., there may be multiple vertices of the simplex $\Delta(k)$ that yield the maximum value of the payoff function.

Finally, a (mixed strategy) Nash equilibrium is defined to be a pair $(p_1^*, p_2^*) \in \Delta(m) \times \Delta(n)$ such that for all $p_1 \in \Delta(m)$ and $p_2 \in \Delta(n)$

$$U_i(p_i, p_{-i}^*) \leq U_i(p_i^*, p_{-i}^*). \tag{4}$$

In a static game, as both players act simultaneously, each player cannot observe the action of the other. Their choice of actions (or probabilities of actions – mixed strategies) is based on their knowledge of both payoff matrices.

[1]As standard in the game theory literature, the index $-i$ is used to indicate those of other players, or the opponent in this case.

2.2 Fictitious Play Process

From the static game described in Subsection 2.1, we define the (discrete-time) FP process as follows. The game is now repeated at times $\tau \in \{0,1,2,\ldots\}$. The empirical frequency $p_i(\tau)$ of player P_i is given by

$$p_i(\tau+1) = \frac{1}{\tau+1}\sum_{j=0}^{\tau} v_i(j) \tag{5}$$

Using induction, we can prove the following recursive relation:

$$p_i(\tau+1) = \frac{\tau}{\tau+1}p_i(\tau) + \frac{1}{\tau+1}v_i(\tau). \tag{6}$$

Player i, $i = 1,2$, then employs the following algorithm:

1: Given payoff matrix M_i.
2: **for** $\tau \in \{0,1,2,\ldots\}$ **do**
3: Update the empirical frequency of the opponent $p_{-i}(\tau+1)$ using (6).
4: Pick the optimal pure strategy (strategies) using (3).
5: If there are multiple optimal strategies, randomize over the optimal strategies with equal probabilities.
6: **end for**

Algorithm 1: Fictitious Play Algorithm.

Thus, in terms of information, there are two main features that distinguish a FP process from the corresponding static game (the static game with the same payoff formulation). First, each player can make decisions without necessarily knowing the other's payoff matrix. Second, each player has to be able to observe the other's actions. For several classes of games, however, it has been shown that such a FP process will finally render both players playing their Nash equilibrium strategies.

3 IMPLEMENTATION

3.1 Test Topology

A straightforward simulation based approach is packet-based; most discrete-events move packets through routers and switches. While this reflects the real-world, it does call for the generation and transmission of many many packets. We set to out implement pushback using a common networking simulator, PRIME (Liu 2006, Liu 2007, Liljenstam et al. 2005). We configured a network that contains essentially only the routers involved in moving traffic from attackers to the victim, illustrated in Figure 1. Of the 64 hosts, $U1$ to $U64$, there are 8 zombies and 56 legitimate users. Both zombies and users send packets to servers $S1$ and $S2$. The routers $Ri.j$, $i = 0,1,2,3$, are organized in a hierarchical manner where the subscript i denotes the level, and the subscript j denotes the router in a level. In the Attacker's side, a central controller controls all the zombies in the network using control messages. Similarly, in the System's side, a master router controls the (slave) routers taking part in the pushback mechanism with pushback control messages.

Each router employs a version of the IP protocol with modifications for enforcing pushback (Mahajan et al. 2002), (Ioannidis and Bellovin 2002).

Every router checks for congestion after each specified time interval which we refer to as the *Congestion_Checking_Interval*. A router is considered to be in congestion if $Incoming\_Data\_Rate > (1 + Target\_Drop\_Rate) \times Outgoing\_Bandwidth$. Here, *Target_Drop_Rate* is the acceptable rate of dropping packets for the router. If a router detects congestion, it looks through the log of dropped packets that it maintains to identify an attack signature. Since the source ip-address of a packet can be spoofed by the Attacker, we only use the destination ip-address as the attack signature. Thus, the router identifies the most frequently occurring destination ip-address in the dropped packets log as the signature. For the sake of efficiency, the log is of a fixed size and new log records overwrite older ones if the log is full. In subsequent checks for congestion, if the router detects that incoming traffic which does not match the signature is still greater than $(1 + Target\_Drop\_Rate) \times Outgoing\_Bandwidth$ then each time it adds the next-most frequently occurring destination ip-address in the log to the current signature. Each signature has a timestamp which is updated every time the router detects congestion. A router also sends the identified signatures to its immediately upstream routers and uses signatures received from downstream routers as attack signatures. Traffic through the router which matches the signature (*Signature Traffic*) is filtered out. The maximum signature traffic allowed to pass through the router is $Outgoing\_Bandwidth \times (1 + Target\_Drop\_Rate) - Non\_Signature\_Traffic$. A router also periodically checks if any of the attack signatures has expired after a specified time interval (which we refer to as *Refresh_Interval*). Routers periodically send update messages for signatures to upstream routers. Routers use update messages from downstream routers to update the timestamp of the signatures received from downstream routers.

Networks	A_1	A_2	A_3	A_4	A_5	Users' Datarates
Network I	91.73 Mbps	55.04 Mbps	27.52 Mbps	2.75 Mbps	0.275 Mbps	1.93 Mbps
Network II	13.76 Gbps	5.504 Gbps	2.752 Gbps	0.275 Gbps	30.58 Mbps	124.42 Mbps

Table 1: Attacker's actions (Datarates generated by all 8 zombies) and collective users' datarates.

Experiments revealed that the volume of simulation traffic forced selection of unrealistically low bandwidths for the links the simulation were to run fast enough to evaluate the defense in a timely fashion (2 Mbps for the Edge bandwidths, 20 Mbps for the Backbone bandwidths). To address this, we created another model where all traffic is described in terms of flows; in these an event occurs when a flow's rate changes at a router, not when a packet crosses it. Flow rates will be constructed to represent aggregated background flows. This choice also makes the simulation more friendly to parallel execution, in a way that we later describe. In this case we were able to use much more realistic bandwidths, OC3 for the Edge (155.52 Mbps) and OC48 on the Backbone links (2.488 Gbps).

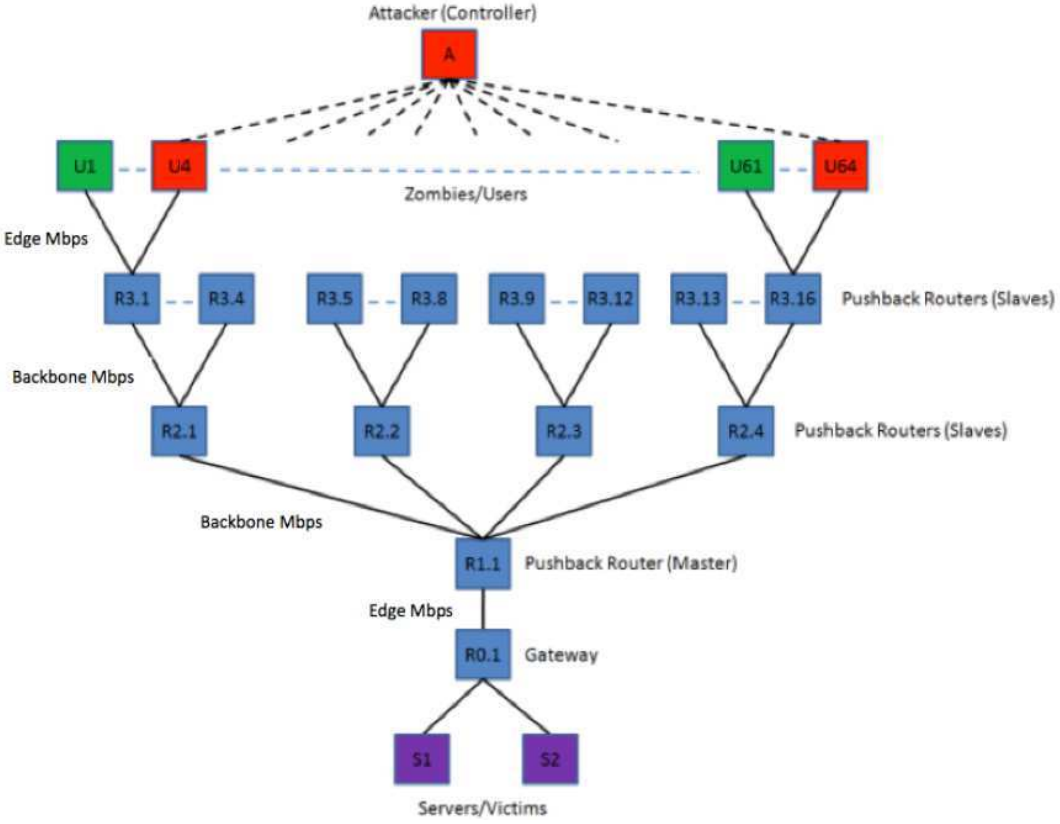

Figure 1: Network Topology.

3.2 Game Formulation

The Attacker's pure strategies are given by $A_{att} = \{A_1, \ldots, A_5\}$, where A_i, $i = 1, \ldots, 5$ are the collective attack datarates (generated by all 8 zombies). The System consists of all the routers taking part in pushback defense, $\{R_1, \ldots, R_r\}$. The pushback behavior of a router is represented by three parameters: $Congestion\_Checking\_Interval$, $Refresh\_Interval$, and $Target\_Drop\_Rate$. The action space of the System, $A_{sys} = \{S_1, \ldots, S_6\}$, is the same for both Networks I and II, and is specified in Table 2. For each pair (A_i, S_j), $i = 1, \ldots, 5,\ j = 1, \ldots, 6$, the payoff of the Attacker is given by

$$U_{att} = \alpha \frac{B_{ao}}{B_N} + (1-\alpha)\left(1 - \frac{\sum_{l=1}^{L} B_{lo}^{(l)}}{\sum_{l=1}^{L} B_{lw}^{(l)}}\right), \tag{7}$$

Actions	Congestion_Checking_Interval (s)	Refresh_Interval (s)	Target_Drop_Rate
S_1	2	5	0.05
S_2	2	10	0.05
S_3	4	5	0.05
S_4	2	5	0.03
S_5	6	10	0.05
S_6	2	5	0.07

Table 2: System's actions.

where $B_{lo}^{(l)}$ is the bandwidth occupied by the legitimate user l, and $B_{lw}^{(l)}$ is the bandwidth required by the legitimate user l, $l = 1,\ldots,L$, where L is the number of legitimate users (56 in our simulations), B_{ao} is the bandwidth occupied by the Attacker, and B_N is the bandwidth capacity ($B_N = 2$ Mbps for Network I and $B_N = 155.52$ Mbps for Network II). $\alpha \in [0,1]$ is used to balance between the damage the Attacker does to the System and the damage it causes to the legitimate users; α is chosen to be 0.2 throughout our simulations. The payoff of the System is given by

$$U_{sys} = \omega \frac{\sum_{l=1}^{L} B_{lo}^{(l)}}{\sum_{l=1}^{L} B_{lw}^{(l)}} + (1-\omega)\left(1 - \frac{B_{ao}}{B_N}\right), \tag{8}$$

where $\omega \in [0,1]$ is used to balance between the utility the System can provide for the legitimate users and the pushback it applies against the Attacker; ω is chosen to be 0.8 throughout our simulations. The costs of attacking and defending can also be included in the payoff functions.

For the Attacker, the action to be taken is determined by the controller and sent to the zombies. The zombies then adjust their datarates and pick their victims accordingly. Similarly, for the System, the action to be taken is determined by the master router and sent to the slave routers. The slave routers then adjust their pushback parameters accordingly.

Our simulations consists of two steps: payoff measurement and fictitious play. In the first step, the System and the Attacker are forced to take each pair of actions. The attack traffic, good traffic, and poor traffic at router $R0.0$ are then measured. These measurements are used to calculate the payoffs for the Attacker and the System using Equations (7) and (8). In the second step, both the System and the Attacker use a fixed time interval as a "time step", during which the action taken by the opponent is identified. At the end of each time interval, both players choose the next action to be taken (which is the best response to the empirical frequencies of the opponent's actions (using Algorithm 1 with the payoff matrices obtained from Step 1). The time step is chosen to be 50s, which allows enough time for the pushback mechanism to stabilize.

3.3 Flow Generation

In a packet-based simulation the Attacker's traffic consists of fixed length IP packets generated at a constant rate by the zombies. Users' traffic consists of fixed length IP packets with inter-packet times being exponentially distributed. For determining the parameter of the exponential distribution, we set the average user datarate to be the bandwidth of the router $R0.0$ divided by the number of hosts in the network. The rationale is that if all users send out data with this rate, there should be no congestion in the network. The datarates generated by the each zombie ranges from around 300 to 1 times the legitimate user datarate.

In order to increase simulated bandwidth to realistic levels, we developed a flow-based version (Nicol and Yan 2006), and study it on the same architecture (but with OC3 and OC48 level bandwidths for the Edge and Backbone links). In this approach, we push "rate events" through paths, not packets. We model two different types of traffic: background traffic and Attacker traffic. Attacker traffic is the traffic generated and controlled by the Attacker. It is deterministic in nature, i.e., the Attacker can precisely control these flows. Background traffic is the aggregate of all other traffic in the network and is stochastic in nature. For background traffic, we assume that different flows are statistically independent.

The input flow at a router port at any instant is the sum of deterministic traffic (from the Attacker) and the stochastic traffic (background traffic). The input deterministic traffic at a given router is the sum of outgoing deterministic flows from all routers connected to it. The input stochastic flows are generated using Gaussian copulas (Nelsen 2006). The idea is to change the input flows in an auto-correlated way, in order to model the dependence in time in flow measurements, owing to the flow representing an aggregate of point-to-point flows. Specifically, let Z be a discrete-time stochastic process describing a flow; the random variable Z_t is the flow rate at time t. Let ρ_{st} be the correlation coefficient between

Z_s and Z_t; we assume that $\rho_{st} = 0$ for $|s-t| > n$, where n is a positive integer. Let

$$\begin{aligned} \Sigma &\equiv \{\rho_{st}\},\ s,t = i-1, i-2, \ldots, i-n,\ (n \times n \text{ matrix}), \\ \underline{\rho}_{ji} &\equiv (\rho_{i,i-1}, \rho_{i,i-2}, \ldots, \rho_{i,i-n}),\ (1 \times n \text{ vector}), \\ \underline{\rho}_{ij} &\equiv \underline{\rho}_{ji}^T. \end{aligned}$$

Let $\underline{Z}_i \equiv (Z_{j-1}, \ldots, Z_{j-n})^T$. We have that $Z_j \sim \mathbb{N}(\mu, \sigma^2)$, i.e., Z_j is a Gaussian random variable with mean μ and variance σ^2. Then we have

$$(Z_j | \underline{Z}_i = \underline{z}_i) = \mathbb{N}(\tilde{\mu}, \tilde{\rho}\sigma^2), \tag{9}$$

where

$$\begin{aligned} \tilde{\mu} &= \underline{\rho}_{ji} \Sigma^{-1} \underline{z}_i \\ \tilde{\rho} &= 1 - \underline{\rho}_{ji} \Sigma^{-1} \underline{\rho}_{ij}. \end{aligned}$$

In this simulation, we choose $n = 5$, $\rho_{st} = (0.75)^{|s-t|}$, $\mu = 0.8 \times outgoing\_bandwidth / number\_of\_flows$, and $\sigma^2 = \mu/3$.

Given the input flows at a router the *Drop_Rate* is defined as $Outgoing\_Bandwidth \times (1 + Target\_Drop\_Rate) - (Input\_Flowrate)$. In absence of filtering, the number of dropped packets from any flow at time t is $Drop\_Rate \times t$. If there is a filter on any flow, then the flow rate is scaled by the filter rate before the *Drop_Rate* is calculated. In the simulation each router periodically samples its stochastic input flows and recalculates its output flows. Any changes in deterministic output flows are sent to routers connected to the given router. A router on receiving an update in deterministic input flow, recalculates its output flows and updates connected routers. The dropped packets set is also updated at each recalculation of flows at the routers. Since the pushback mechanism only needs the dropped-packets set and the test for congestion to work, the mechanism works well with the flow-based simulation.

4 SIMULATION RESULTS

In this section, we present the simulation results of both packet-based simulation and flow-based simulation.

4.1 Packet-based simulation results

The results from payoff measurements are presented in Table 3. We use the parameters given in Section 3. The payoff matrices of the System and the Attacker are shown in Tables 4 and 5.

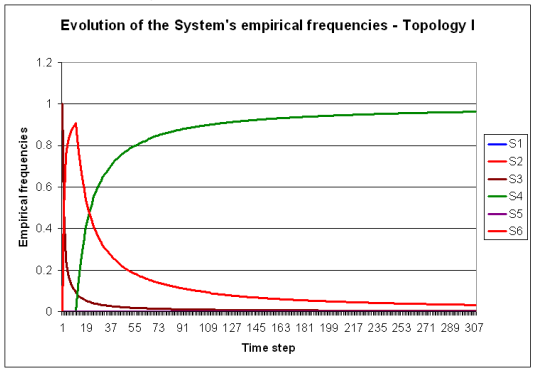

Figure 2: System's empirical frequencies - packet-based.

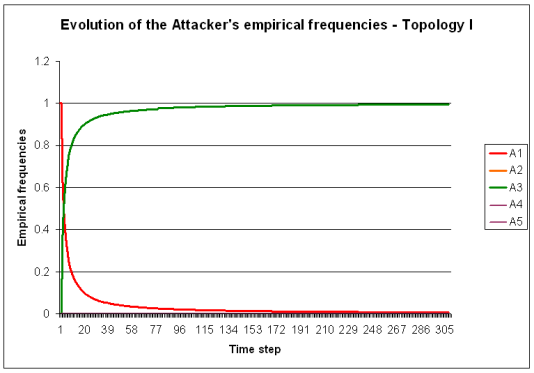

Figure 3: Attacker's empirical frequencies - packet-based.

The Nash Equilibrium calculated using Gambit (McKelvey, McLennan, and Turocy 2006) is $(0,0,1,0,0)$ for the Attacker and $(0,0,0,1,0,0)$ for the System. The fictitious play simulation results are given in Figures 2 and 3. The number of time steps simulated is about 300. It can be seen that the frequency of pure strategy $S4$ of the System goes to 1 while those of other pure strategies go to 0. Similarly, the frequency of pure strategy $A3$ goes to 1, while those of the others go to 0. This coincides with the Nash Equilibrium obtained from Gambit.

It is however worth noting that in general this mechanis does not guarantee convergence to a Nash equilibrium. For two-player zero-sum FP, the convergence proof was obtained for arbitrary numbers of actions for each player $(m \times n)$ (Robinson 1951). For nonzero-sum games, the proofs for two-player FP have been found for the case where

Actions	Attacker Traffic	Good Traffic	Poor Traffic	Attacker Payoff	System Payoff
$A1, S1$	$2.42E+05$	$5.15E+05$	$8.70E+03$	0.6067	0.3933
$A1, S2$	$2.42E+05$	$5.12E+05$	$8.48E+03$	0.6083	0.3917
$A1, S3$	$3.07E+05$	$5.00E+05$	$9.22E+03$	0.6194	0.3806
$A1, S4$	$2.40E+05$	$5.33E+05$	$1.55E+04$	0.5961	0.4039
$A1, S5$	$3.66E+05$	$5.49E+05$	$8.52E+03$	0.6051	0.3949
$A1, S6$	$2.40E+05$	$5.58E+05$	$7.66E+03$	0.5892	0.4108
$A2, S1$	$2.41E+05$	$5.21E+05$	$8.04E+03$	0.6044	0.3956
$A2, S2$	$2.41E+05$	$5.17E+05$	$8.27E+03$	0.6058	0.3942
$A2, S3$	$3.06E+05$	$5.05E+05$	$8.43E+03$	0.6174	0.3826
$A2, S4$	$2.41E+05$	$5.33E+05$	$1.50E+04$	0.5967	0.4033
$A2, S5$	$3.68E+05$	$5.54E+05$	$7.03E+03$	0.6039	0.3961
$A2, S6$	$2.39E+05$	$5.77E+05$	$6.70E+03$	0.5817	0.4183
$A3, S1$	$2.42E+05$	$5.12E+05$	$9.17E+03$	0.6080	0.3920
$A3, S2$	$2.43E+05$	$5.09E+05$	$9.26E+03$	0.6090	0.3910
$A3, S3$	$3.07E+05$	$5.06E+05$	$1.07E+04$	0.6159	0.3841
$A3, S4$	$2.41E+05$	$5.24E+05$	$1.65E+04$	0.5995	0.4005
$A3, S5$	$3.68E+05$	$5.46E+05$	$1.16E+04$	0.6053	0.3947
$A3, S6$	$2.45E+05$	$5.30E+05$	$7.99E+03$	0.6012	0.3988
$A4, S1$	$2.44E+05$	$9.78E+05$	$1.51E+05$	0.3556	0.6444
$A4, S2$	$2.44E+05$	$9.78E+05$	$1.51E+05$	0.3556	0.6444
$A4, S3$	$2.66E+05$	$9.68E+05$	$1.64E+05$	0.3563	0.6437
$A4, S4$	$2.35E+05$	$9.82E+05$	$1.46E+05$	0.3551	0.6449
$A4, S5$	$2.87E+05$	$9.56E+05$	$1.79E+05$	0.3575	0.6425
$A4, S6$	$2.52E+05$	$9.74E+05$	$1.56E+05$	0.3560	0.6440
$A5, S1$	$2.17E+05$	$8.98E+05$	$7.78E+05$	0.1256	0.8744
$A5, S2$	$2.17E+05$	$8.98E+05$	$7.78E+05$	0.1256	0.8744
$A5, S3$	$1.77E+05$	$1.04E+06$	$6.50E+05$	0.1172	0.8828
$A5, S4$	$2.19E+05$	$8.64E+05$	$7.80E+05$	0.1393	0.8607
$A5, S5$	$2.21E+05$	$8.33E+05$	$7.80E+05$	0.1521	0.8479
$A5, S6$	$2.15E+05$	$9.32E+05$	$7.76E+05$	0.1122	0.8878

Table 3: Payoff measurements - Network I.

System \ Attacker	A_1	A_2	A_3	A_4	A_5
S_1	0.3933	0.3956	0.3920	0.6444	0.8744
S_2	0.3917	0.3942	0.3910	0.6444	0.8744
S_3	0.3806	0.3826	0.3841	0.6437	0.8828
S_4	0.4039	0.4033	0.4005	0.6449	0.8607
S_5	0.3949	0.3961	0.3947	0.6425	0.8479
S_6	0.4108	0.4183	0.3988	0.6440	0.8878

Table 4: System's payoff matrix - Network I.

Attacker \ System	S_1	S_2	S_3	S_4	S_5	S_6
A_1	0.6067	0.6083	0.6194	0.5961	0.6051	0.5892
A_2	0.6044	0.6058	0.6174	0.5967	0.6039	0.5817
A_3	0.6080	0.6090	0.6159	0.5995	0.6053	0.6012
A_4	0.3556	0.3556	0.3563	0.3551	0.3575	0.3560
A_5	0.1256	0.1256	0.1172	0.1393	0.1521	0.1122

Table 5: Attacker's payoff matrix - Network I.

Actions	Attacker	Background Traffic	Attacker Payoff	System Payoff
$A1, S1$	$1.25E+08$	$3.03E+07$	0.9665	0.2335
$A1, S2$	$1.25E+08$	$3.03E+07$	0.9665	0.2335
$A1, S3$	$1.27E+08$	$2.86E+07$	0.9796	0.2204
$A1, S4$	$1.25E+08$	$3.03E+07$	0.9659	0.2341
$A1, S5$	$1.27E+08$	$2.86E+07$	0.9796	0.2204
$A1, S6$	$1.25E+08$	$3.01E+07$	0.9676	0.2324
$A2, S1$	$1.25E+08$	$3.03E+07$	0.9665	0.2335
$A2, S2$	$1.25E+08$	$3.03E+07$	0.9664	0.2336
$A2, S3$	$1.27E+08$	$2.86E+07$	0.9796	0.2204
$A2, S4$	$1.25E+08$	$3.03E+07$	0.9659	0.2341
$A2, S5$	$1.27E+08$	$2.86E+07$	0.9796	0.2204
$A2, S6$	$1.25E+08$	$3.01E+07$	0.9676	0.2324
$A3, S1$	$1.25E+08$	$3.03E+07$	0.9664	0.2336
$A3, S2$	$1.25E+08$	$3.03E+07$	0.9664	0.2336
$A3, S3$	$1.27E+08$	$2.86E+07$	0.9796	0.2204
$A3, S4$	$1.25E+08$	$3.03E+07$	0.9659	0.2341
$A3, S5$	$1.27E+08$	$2.86E+07$	0.9796	0.2204
$A3, S6$	$1.25E+08$	$3.01E+07$	0.9676	0.2324
$A4, S1$	$1.26E+08$	$2.94E+07$	0.9729	0.2271
$A4, S2$	$1.25E+08$	$3.02E+07$	0.9666	0.2334
$A4, S3$	$1.25E+08$	$3.02E+07$	0.9666	0.2334
$A4, S4$	$1.26E+08$	$2.94E+07$	0.9729	0.2271
$A4, S5$	$1.25E+08$	$3.02E+07$	0.9666	0.2334
$A4, S6$	$1.25E+08$	$3.02E+07$	0.9669	0.2331
$A5, S1$	$3.06E+07$	$3.46E+07$	0.8169	0.3831
$A5, S2$	$3.06E+07$	$3.46E+07$	0.8169	0.3831
$A5, S3$	$3.06E+07$	$3.46E+07$	0.8169	0.3831
$A5, S4$	$3.06E+07$	$3.46E+07$	0.8169	0.3831
$A5, S5$	$3.06E+07$	$3.46E+07$	0.8169	0.3831
$A5, S6$	$3.06E+07$	$3.46E+07$	0.8169	0.3831

Table 6: Bandwidth - Network II.

one player is restricted to 2 actions (See (Berger 2003) for classical FP and (Shamma and Arslan 2004) for stochastic FP). Nevertheless, there are counter examples (e.g., for 3×3 games) where FP does not converge to the mixed strategy NE (Shamma and Arslan 2005). Several techniques that can be used to enhance convergence are discussed in (Shamma and Arslan 2005, Nguyen, Alpcan, and Başar 2010a).

One assumption in a fictitious play process is each player has to be able to observe the actions of the opponent. As mentioned earlier, the Attacker's central controller has to send messages to the zombies to coordinate the attacks. On the System's side, the master router also controls all the routers in the pushback mechanism using pushback control messages. In this work, we assume each player has access to its opponent's control messages, and thus can observe the opponent's actions. In practice, players are subject to both decision errors (e.g., control messages may get corrupted, delayed, or lost) and observation errors (due to incomplete information on the other's actions). Such complications may affect the convergence and the Nash equilibria of the game (Nguyen, Alpcan, and Başar 2009, Nguyen, Alpcan, and Başar 2010b).

4.2 Flow-based simulation results

The results from payoff measurements and the payoff matrices of the System and the Attacker are presented in Tables 6, 7, and 8, respectively. Again, we use the parameters given in Section 3. From Gambit, there are three Nash equilibria For Network II:

- Attacker $(0,0,0,1,0)$, System $(0,0.992,0,0,0.008,0)$.
- Attacker $(0,0,0,1,0)$, System $(0,0.992,0.008,0,0,0)$.
- Attacker $(0,0,0,1,0)$, System $(0,1,0,0,0,0)$.

Again, the fictitious play simulation results are in agreement with the Nash equilibria obtained from Gambit. Note that the first two mixed-strategy Nash equilibria (MSNE) shown above are very close to pure strategies. In the case where

System \ Attacker	A_1	A_2	A_3	A_4	A_5
S_1	0.2335	0.2335	0.2336	0.2271	0.3831
S_2	0.2335	0.2336	0.2336	0.2334	0.3831
S_3	0.2204	0.2204	0.2204	0.2334	0.3831
S_4	0.2341	0.2341	0.2341	0.2271	0.3831
S_5	0.2204	0.2204	0.2204	0.2334	0.3831
S_6	0.2324	0.2324	0.2324	0.2331	0.3831

Table 7: System's payoff matrix - Network II.

Attacker \ System	S_1	S_2	S_3	S_4	S_5	S_6
A_1	0.9665	0.9665	0.9796	0.9659	0.9796	0.9676
A_2	0.9665	0.9664	0.9796	0.9659	0.9796	0.9676
A_3	0.9664	0.9664	0.9796	0.9659	0.9796	0.9676
A_4	0.9729	0.9666	0.9666	0.9729	0.9666	0.9669
A_5	0.8169	0.8169	0.8169	0.8169	0.8169	0.8169

Table 8: Attacker's payoff matrix - flow-based.

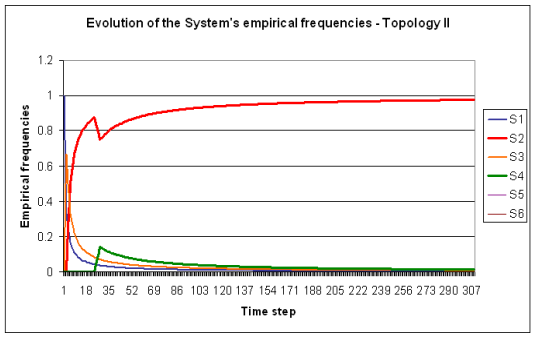

Figure 4: System's empirical frequencies - flow-based.

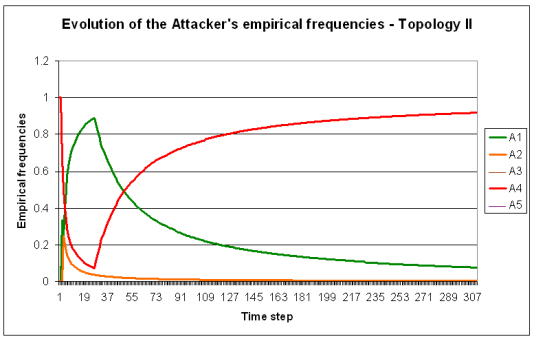

Figure 5: Attacker's empirical frequencies - Network II.

there are mixed-strategy Nash equilibria, if the first action (at time $\tau = 0$) of each player is chosen appropriately (which is necessary only if there are both mixed-strategy NE and pure-strategy NE), the empirical frequencies of the player's actions will converge to a mixed-strategy Nash equilibrium, which means each player will alternate among the pure strategies constituting the MSNE with proportional numbers of times.

5 METHODOLOGY AND PARALLELISM

Both the packet and flow based models were implemented in PRIME, which can exploit parallelism automatically. An important point is that in the packet based approach the temporal separation between distributed components is on the order of the latency across a link. For packet-based simulation this is small. In the flow-based simulation the temporal separation is on the order of the delay between successive updates to the background flow rates. This is considerably larger, and furthermore these updates can be done synchronously. This fact implies that the flow-based approach has potential for a much higher level of performance both because of the model abstraction, but also by reducing the overhead associated with synchronization. The nature of the game-theoretic defense allowed us to develop a methodology for evaluating it that is friendly to parallel processing.

6 CONCLUSION

In this paper we have developed a methodology suitable for evaluating the effectiveness of a game-theoretic defense against DDoS, using parallel simulation. We have modeled DDoS attacks and the pushback mechanism as a 2-player game between the Attacker and the System. We have used a simple fictitious play mechanism (classical FP) that allows players to learn their Nash equilibrium strategies without necessarily knowing the opponent's payoff function. Simulation experiments forced us to develop a flow-based simulation of traffic, because the packet-based approach takes

too much computation time. The flow-based approach is much more efficient, due to its aggregation of flows, and its allowance for larger temporal separation between submodels in parallelized simulation.

ACKNOWLEDGMENTS

This work was supported by the Boeing Company.

REFERENCES

Berger, U. 2003, March. Fictitious play in 2xn games. Game theory and information, EconWPA.

Ioannidis, J., and S. M. Bellovin. 2002. Implementing pushback: Router-based defense against ddos attacks. In *NDSS*.

Liljenstam, M., J. Liu, D. Nicol, Y. Yuan, G. Yan, and C. Grier. 2005. Rinse: the real-time immersive network simulation environment for network security exercises. In *In Proceedings of the 19th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation (PADS*.

Liu, J. 2006. Parallel real-time immersive modeling environment (prime) scalable simulation framework (ssf). Technical report, Colorado School of Mines.

Liu, J. 2007. The design of prime ssfnet: A tutorial for developers of the network simulator. Technical report, Colorado School of Mines.

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

Mahajan, R., S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. 2002. Controlling high bandwidth aggregates in the network. *ACM Computer Communication Review* 32:62–73.

McKelvey, R. D., A. M. McLennan, and T. L. Turocy. 2006. Gambit: Software tools for game theory. Technical report, Version 0.2007.01.30.

Nelsen, R. B. 2006. *An introduction to copulas*. Springer Series in Statistics. New York, NY: Springer-Verlag. 2nd Edition.

Nguyen, K. C., T. Alpcan, and T. Başar. 2009, June. Security games with incomplete information. In *Proc. of IEEE Intl. Conf. on Communications (ICC 2009)*. Dresden, Germany.

Nguyen, K. C., T. Alpcan, and T. Başar. 2010a, September. Fictitious play with time-invariant frequency update for network security. In *Proc. of the 2010 IEEE Multi-Conference on Systems and Control (MSC10)*. Yokohama, Japan.

Nguyen, K. C., T. Alpcan, and T. Başar. 2010b, June-July. Security games with decision and observation errors. In *Proc. of the 2010 American Control Conference (ACC 2010)*. Baltimore, Maryland, USA.

Nicol, D. M., and G. Yan. 2006. High-performance simulation of low-resolution network flows. *Simulation* 82 (1): 21–42.

Peng, T., C. Leckie, and K. Ramamohanarao. 2007. Survey of network-based defense mechanisms countering the dos and ddos problems. *ACM COMP. SURV* 39 (1).

Robinson, J. 1951. An iterative method of solving a game. *Ann. Math.* 54:296–301.

Shamma, J. S., and G. Arslan. 2004, July. Unified convergence proofs of continuous-time fictitious play. *IEEE Transactions on Automatic Control* 49 (7): 1137–1142.

Shamma, J. S., and G. Arslan. 2005, March. Dynamic fictitious play, dynamic gradient play, and distributed convergence to nash equilibria. *IEEE Transactions on Automatic Control* 50 (3): 312–327.

AUTHOR BIOGRAPHIES

TANMAY KHIRWADKAR received a B.Tech degree in Computer Science from Indian Institute of Technology (IIT) Bombay in 2009. He is currently pursuing a M.S. Degree in Computer Science at the University of Illinois at Urbana-Champaign. He works as a Research Assistant at the Information Trust Institute with Professor David Nicol. His research interests include Network Security, Game Theory, and Databases. His email address is <khirwad1@illinois.edu>.

KIEN C. NGUYEN received a B.Eng. in Electronics and Telecommunications from Hanoi University of Technology, Vietnam, an M.Eng.Sc. in Telecommunications from the University of New South Wales, Australia, and an M.S. in Electrical and Computer Engineering (ECE) from the University of Illinois at Urbana-Champaign (UIUC). He is currently a Ph.D. Candidate in ECE at UIUC. His research interests include network security, hypothesis testing, and game theory. He was a recipient of an Australian Development Scholarship (2001) and a Vietnam Education Foundation Fellowship (2004). His email address is <knguyen4@illinois.edu>.

David M. Nicol is Professor of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He holds a B.A. in mathematics from Carleton College (1979), and M.S. and Ph.D. degrees in computer science from the University of Virginia (1983,1985). Prior to joining UIUC, he taught at the College of William & Mary, and Dartmouth College. He has served in many roles in the simulation community (e.g. Editor-in-Chief of ACM TOMACS,

General Chair of the Winter Simulation Conference Executive Board of the WSC), was elected Fellow of the IEEE and Fellow of the ACM for his work in discrete-event simulation, and was the inaugural recipient of the ACM SIGSIM Distinguished Contributions award. His current research interests include application of simulation methodologies to the study of security in computer and communication systems. His email address is <dmnicol@illinois.edu>.

TAMER BAŞAR received the BSEE degree from Robert College, Istanbul, and the MS, MPhil, and PhD degrees in engineering and applied science from Yale University. After holding positions at Harvard University and Marmara Research Institute (Gebze, Turkey), he joined the University of Illinois at Urbana-Champaign (UIUC) in 1981, where he currently holds the positions of Swanlund Endowed Chair, Center for Advanced Study Professor of Electrical and Computer Engineering, Research Professor at the Coordinated Science Laboratory, and Research Professor at the Information Trust Institute. He has published extensively in systems, control, communications, and dynamic games, and has current research interests in modeling and control of communication networks; control over heterogeneous networks; resource allocation, management and pricing in networks; game-theoretic tools for networks; mobile computing; security issues in computer networks; and robust identification, estimation and control. He has received several awards and recognitions over the years, among which are the Medal of Science of Turkey (1993); Distinguished Member Award (1993), Axelby Outstanding Paper Award (1995), and Bode Lecture Prize (2004) of the IEEE Control Systems Society (CSS); Millennium Medal of IEEE (2000); Tau Beta Pi Drucker Eminent Faculty Award of UIUC (2004); the Outstanding Service Award (2005) and the Giorgio Quazza Medal (2005) of the International Federation of Automatic Control (IFAC); and the Richard E. Bellman Control Heritage Award (2006) of the American Automatic Control Council (AACC). He is a member of the National Academy of Engineering (of USA), a member of the European Academy of Sciences, a Fellow of IEEE, a Fellow of IFAC, President of AACC, a past president of CSS, and the founding president of ISDG. His email address is <basar1@illinois.edu>.