

Intelligence, not Integration: Distributed Regret Minimization for IDS Control

Martin Reháč
Czech Technical University
Prague, Czech Republic
Email: rehakm@agents.felk.cvut.cz

Jan Stiborek
Czech Technical University
Prague, Czech Republic
Email: stiborek@agents.felk.cvut.cz

Martin Grill
Czech Technical University
Prague, Czech Republic
Email: grillm@agents.felk.cvut.cz

Abstract—We present an empirical study of regret minimization procedure used in a distributed Intrusion Detection System (IDS) to independently adapt the self-contained components of the system without any explicit coordination. We show that the regret minimization methods can be used to build survivable distributed security systems that can only communicate using standard data-transfer protocols (NetFlow, selective traffic mirroring or alerts) and do not need to rely on explicit communication required by more elaborate coordination protocols. The intended impact is dramatically easier integration, maintenance and repair of IDS systems, with only a small impact on system characteristics.

I. INTRODUCTION

One of the principal advantages of security systems based on the distributed intelligent behavior is their ability to perform runtime reconfiguration, such as the ability to react to changing environmental conditions by introducing novel agents or changing their behavior. However, it is a well known fact that even if the individual autonomous components (agents) in the system optimally react to the *local* environment changes (e.g. react to the parts of the environment they are able to perceive), the global reaction can still be suboptimal. Therefore, numerous collaboration mechanisms based on communication protocols were developed in the multi-agent and web services research community. Some of the recent mechanisms are based on the methods from the fields of Game Theory [1], [2] and Algorithmic Game Theory [3] that are designed to ensure robust behavior in environments where the players don't have complete information, as their incentives and goals are not aligned or are even contradictory.

However, many collaboration and coordination mechanism introduce undesirable complexity and dependencies between the otherwise separate and well-defined agents. Such dependencies not only make the system more fragile, but can be a significant problem when designing the systems that are compatible with networks that operate in a multi-level security [4] architectures - bi-directional communication may be prohibited by system policy.

More typically, the collaboration will suffer from technical difficulties: distributed protocols are notoriously difficult to implement, the agents need to share **mutually compatible models** of their internal state, environment and preferences, and need to be able to interpret the models and information shared by the others. We argue that the additional complexity

is rarely justified, and that the full lifecycle costs of maintaining the compatibility for such mechanisms can be prohibitive, as the example of CORBA standardization suggests.

Both above concerns: *multi-level security* and *complex integration and lifecycle management* are of primary importance in the networks used in tactical environments. With the progress towards more ad-hoc networking and mesh architectures, the likelihood of deploying the same or similar network configuration twice is almost zero. Each network is likely to be a novel, continuously changing amalgamate of devices, network equipment, services and applications that will evolve in real time together with the operations in the field. We need to expect that the opponent will attack this amorphous network, and we need to prepare a set of intrusion detection/prevention methods well adopted to ensure its security under the operational constraints.

We address the limitations inherent to the domain by carefully selecting very simple, yet robust and well described methods from the field of game theory. In essence, **we replace the explicit coordination/communication between the agents by parallel, independent learning of agents** that individually observe at least partially correlated environments (or the same environment in the ideal case). After the careful analysis of available computational approaches to game solution, we have selected the *regret minimization* (Section II), implemented it into an actual anomaly-based intrusion detection system [5] and measured whether the theoretical properties of the method hold in the practical deployment in a highly dynamic environment. Our contribution consists of:

- Empirical comparison of **distributed** (multiplayer) regret minimization problem **vs.** the **centralized** (two player, attacker vs. defender problem) formulation of the same problem, specifically verifying whether they converge towards the same results (both are defined in Section III).
- Empirical study of **convergence** of regret minimization algorithms in the **dynamic environment**, verifying whether the algorithms identify the equilibria in accordance with the theory (or beyond the theory, as the convergence was shown for static environments only).
- Experimental deployment of regret minimization in an **operational environment** on the university network. The regret minimization algorithms in both distributed and centralized form were integrated with an industrial

intrusion detection system (CAMNEP, as described in Section IV) and used to optimize this system’s performance in a real production environment.

Globally, this is an experimental paper that verifies whether we can push the regret minimization algorithms outside of the domain where their properties are verified theoretically (sequences of static games) into the dynamic, real world problems, where they can help us solve real distributed security concerns. We assume that the reader has basic knowledge of game theory and we don’t introduce the basic terms as game, player, equilibria or information set – references [2] and [3] provide excellent introductions and full, technically correct definitions with proper background.

II. REGRET AND REGRET MINIMIZATION

Regret minimization is an algorithm that attempts to achieve long-term optimality in a sequence of identical games. It does not require any knowledge of other players’ utility functions or the communication between players (players are denoted as P), but is based on independent loss minimization performed by the individual players given the payoffs received for their past moves.

The regret, or more specifically the *external regret*, is defined [6], [7] as an ex-post evaluated *loss of utility* due to the suboptimal strategy selection - thus the term regret. Due to the properties of our problem as imposed by the design of our system, we assume *full information model*, i.e. we assume that player P has access to all payoff values of all strategies x from the set X in any time step in the past¹. It shall be noted that the regret minimization method *does not* require the players to know the utility function and/or goals of the opponents – the strategy played by the agents is based only on the feedback that they receive.

We will denote l^t the loss related to the use of one defender’s strategy in the time step t – note that the strategy is not explicitly denoted² in the notation. Then, we define the loss in time step T aggregated over time as:

$$L^T = \sum_{t=T-N}^T l^t \quad (1)$$

and the regret as follows:

$$R^T = L^T - \min(L^T) \quad (2)$$

It represents the difference of (hypothetical) payoff encountered upon (hypothetical) playing of this strategy (or using a particular strategy selection algorithm) and the best payoff available over the set of player’s strategies X .

The regret value is then used in a *polynomial weights regret minimization* (PWM) algorithm [3]. In the PWM algorithm, the player P selects the strategy x with a probability that

¹This contrasts with a *partial information model*, where the players only have access to the results of strategies actually played in the past

²Some authors use index i to emphasize the loss l_i^t relevance to a particular strategy x_i . In our case, we omit this index to make the notation lighter in context of multi-player game as it will be introduced in Section III.

is inversely proportional to the regret. Formally, we define weight w^t assigned to each strategy x at time t and p^t , the corresponding probability of playing the strategy x . Initially at time 1, all the strategies x from the set X receive the identical weights and probabilities:

$$w^1 = 1 \text{ and } p^1 = \frac{1}{|X|}, \forall x \in X \quad (3)$$

With increasing time, more and more games are played and the values w^t and p^t assigned to individual strategies are determined as follows (with $\eta = 0.2$, as it needs to be $\eta < 0.5$ for the convergence proof to hold [6]):

$$w^t = w^{t-1}(1 - \eta L^{t-1}) \quad (4)$$

$$W^t = \sum_{x \in X} w^t \quad (5)$$

$$p^t = \frac{w^t}{W^t} \quad (6)$$

Amortized loss for dynamic environment. The loss/regret function used in our experiments includes time-amortized loss accumulated over N last time periods, rather than over the totality of history. The Eq. 1 is thus modified as follows:

$$L^T = \omega L^{T-1} + (1 - \omega)l^T \quad (7)$$

The use of Eq. 7 with $\omega = 0.5$ instead of simple loss accumulation accounts for the fact that the real-world players can join the games and leave them dynamically and that the characteristics of the game can change dynamically. This is a generalization of traditional regret definition, but is consistent with previous work and key properties of the PWM method still hold as shown in [7].

A. Properties

Regret minimization is a robust method that yields predictable results in a wide range of games. In a static environment, it can be shown that the use of the proper regret minimization algorithm (as the one described above) bounds the maximal loss achieved using external regret minimization by the term

$$O(\sqrt{T \log |X|}) \quad (8)$$

relative to the best loss achievable [6]. This is a general result that can be applied outside of game theoretic framework.

In the specific case of two player, zero-sum games, the use of regret minimization will make the player’s payoffs converge towards the value of the game, with the speed of convergence bounded by the term above (Eq. 8). This result can be generalized for a far broader class of games. Hart and Mas-Colell show that if *all* players play regret minimization in a sequence of static games, the joint distribution of play converges [8], [7] to the set of *correlated equilibria* [3], [9] of the stage game. One of the important corollaries is that the probability of strategy switching decreases as well, making the players reach increasingly longer sequences of constant strategy play.

The correlated equilibrium is an extension of the well-known Nash equilibrium (i.e. stable point in the strategy space where none of the players benefits from unilateral deviation) by assuming that the players can either communicate, or can observe a common variable(s) or share a history of gameplay. All these specific examples are special cases of a *correlating device* [6]. Such device produces a set of (correlated) signals, one for each player, and the players use the signals for strategy selection in the game. It can be shown that when the signals are fully correlated (e.g. when all players share a single public signal), the correlated equilibria set equals the convex hull of Nash equilibria.

On the other hand, the convergence to the smaller set of Nash equilibria is possible, but is guaranteed only in very specific types of game, and not guaranteed at all in the general case [10]. In particular, in the two player, zero sum game example mentioned above the game converges, but counterexamples of non-convergent games can be found even for simple three player games, such as Shapley game [11]. In the non-zero sum games with more than two players, the robust regret-minimizing algorithm provides robust results when other approaches may fail.

The results of [12] suggest that regret minimization is also robust in zero-sum finite extensive-form games with perfect recall when applied across independent information sets in the game. The regret (counterfactual regret) is measured and minimized on information sets in the game and the authors show that minimization of the counterfactual regret in individual game stages bounds the overall regret of the global game – albeit only in a very specific class of games. This is an extremely important property, as it hints that at least some of the regret minimization properties may hold when we split one of the players into several partial players, each of them selecting a fraction of the original player’s strategy.

It is also important to note that we do not oblige all players to use regret minimization (even if we do so in the experiments in Section V-A). Regret minimization performs robustly also against the rational players who simply play their own equilibria strategies obtained by other methods. When the opponents play stable non-equilibria strategies, regret minimization is likely to benefit from the sub-optimality.

The question of convergence is further complicated in the dynamic environment. The speed of algorithm’s convergence becomes critical, as it needs to converge within the time interval when the environment (which defines the structure of the game) is relatively stable, making the set of correlated equilibria also stable. This also implies that the value of the past history decreases with increasing time difference, making us adopt the amortized loss function as defined by Eq. 7 to find the balance between the robustness and speed of convergence.

III. FORMAL PROBLEM STATEMENT

We assume that one distributed intrusion detection system, which can be decomposed into autonomously acting agents, plays against one opponent. The **defender** system P_D consists of two layers P_D^1, P_D^2 , each layer consists of one or more

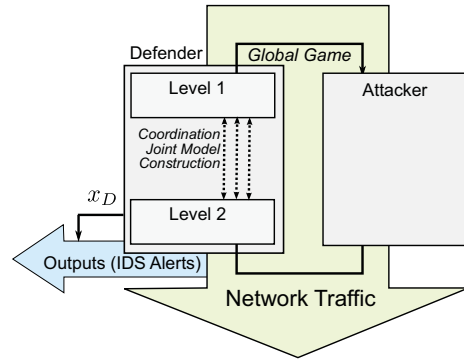


Fig. 1. Problem formulated as a global game.

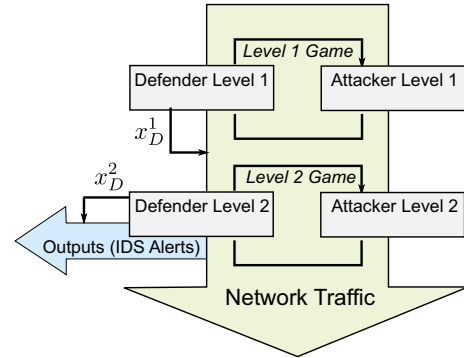


Fig. 2. Problem formulated as two games correlated through the effect on the network traffic and detection results. Note that both layers apply their strategies independently and that the strategy applied by the first layer influences the inputs of the second layer.

agents that implement one functionality/activity. Defender’s strategies x_D are drawn from the set X_D and can be decomposed into distinct individual layer’s strategies: x_D^1, x_D^2 from the mutually orthogonal subspaces X_D^1 and X_D^2 of the space X_D . We have implemented the **attacker** P_A (and P_A^1, P_A^2) as a *model of the attacker* implemented within the system [13], [14], typically also consists of one or more agents that realistically represent the goals, utility functions and actions use by the real attackers. This is by no means necessary for the regret minimization approach to work: we have opted for this method to be able to easily determine the Nash equilibria and correlated equilibria in the game. The details of attacker modeling will be discussed in Section IV. Attacker’s strategies x_A (same in both game variants below) are drawn from the set X_A and correspond to the realistic attack actions that may harm the protected network. Each strategy corresponds to one individual attack technique available to the attacker. For simplicity, we assume that attacker’s action spaces are identical (X_A) in both layers when the game is played in the distributed form.

Two game variants are defined as follows:

- **Global game:** In this traditional formulation (Fig. 1) used as a benchmark, we build and solve a two player game of the attacker P_A against the defender P_D . Defender

selects full strategies x_D for both layers simultaneously, and has full access to all relevant system and environment state variables to build and update the global utility function u_D . In this setup, the defender is able to directly identify global optima in the global strategy space. On the other hand, finding the equilibria of the global game and reasoning about the selection of the optimal strategy becomes increasingly computationally difficult with growing dimension of strategy spaces. The game is formulated as follows:

$$G = (\{P_A, P_D\}, \{X_A, X_D\}, \{u_A, u_D\}) \quad (9)$$

$$u_A : x_A \times x_D \rightarrow \mathbb{R} \quad (10)$$

$$u_D : x_A \times x_D \rightarrow \mathbb{R} \quad (11)$$

The utility functions of the attacker u_A and the defender u_D are defined as functions of the global strategies.

- **Distributed game:** In our formulation of the game (Fig. 2), each of the layers P_D^1 and P_D^2 (and P_A^1 and P_A^2) plays as an independent player. Therefore, we have a four player game, with the players $P_A^1, P_A^2, P_D^1, P_D^2$, where the couples of the players **should** be ideally optimizing a shared utility function. In practice, though, the utility functions (on the defender's side) differ, as the internal state of the players P_D^1 and P_D^2 is mutually inaccessible (for integration reasons), meaning that their dynamically updated utility functions can only rarely be identical in practice. This defines the distributed version of the game as follows:

$$G = (\{P_A^1, P_A^2, P_D^1, P_D^2\}, \{X_A, X_D^1, X_D^2\}, \{u_A^1, u_A^2, u_D^1, u_D^2\}) \quad (12)$$

$$u_A^1 : x_A^1 \times x_D^1 \rightarrow \mathbb{R} \quad (13)$$

$$u_D^1 : x_A^1 \times x_D^1 \rightarrow \mathbb{R} \quad (14)$$

$$u_A^2 : x_A^1 \times x_A^2 \times x_D^1 \times x_D^2 \rightarrow \mathbb{R} \quad (15)$$

$$u_D^2 : x_A^1 \times x_A^2 \times x_D^1 \times x_D^2 \rightarrow \mathbb{R} \quad (16)$$

For higher generality, we break down both players and also **both** player's utility functions into u_A^1 and u_A^2 for the attacker and u_D^1 and u_D^2 for the defender. In practice, most attackers will just use a single strategy and utility function in both levels of the game, resulting in $u_A^1 = u_A^2$ and $x_A^1 = x_A^2$, and the game will degenerate into a three player game³. Note the distinction between the two layer's utility functions, where the first layer's results are independent of the second layer's actions, while the results of the second layer depend on both. Effectively, the utility function u_D^2 would typically be

³The three player formulation would be:

$$G = (\{P_A, P_D^1, P_D^2\}, \{X_A, X_D^1, X_D^2\}, \{u_A, u_D^1, u_D^2\}) \quad (17)$$

$$u_A : x_A \times x_D^1 \times x_D^2 \rightarrow \mathbb{R} \quad (18)$$

$$u_D^1 : x_A \times x_D^1 \rightarrow \mathbb{R} \quad (19)$$

$$u_D^2 : x_A \times x_D^1 \times x_D^2 \rightarrow \mathbb{R} \quad (20)$$

Graphically, this would correspond to the attacker from Fig 1 playing against the defenders from Fig. 2

the same as u_D , with the notable difference that only the action x_D^2 is under the control of the player P_D^2 , while the action x_D^1 is selected independently by the player P_D^1 who optimizes its own utility function u_D^1 . We do not require the players P_D^1 and P_D^2 to know each other's utility functions, but their consistence with u_D determines whether the distributed game results in the same equilibria as the global game.

The notion of independent optimization in the distributed game where only one of the layers depends on the actions of the other, and is not explicitly informed about the action played by the first layer, makes the game consistent with the formalization based on extensive-form games introduced in [12] and discussed in Sec. II-A. As the defender's second layer is not informed about the strategy x_D^1 played by the first layer, it performs the selection of x_D^2 in the same information set regardless of the x_D^1 . Likewise, the first defender player is typically not able to infer the strategies played, payoffs received or the internal variable states of the second level player.

Both formulations will be solved by regret minimization algorithm introduced in Section II, in one case by two players, in the other by four players playing independently. We are facing a dynamic optimization problem, where the environmental conditions imposed by the external environment can change rapidly, making the game similar to a sequence of static games with unpredictable length. In the next section, we will introduce a specific instantiation of the above model on a specific intrusion detection system.

IV. EXPERIMENTAL SYSTEM

The experiments were performed inside the CAMNEP [5] Intrusion Detection System which was de-coupled into two layers to allow the evaluation of the above-described principles. The first layer is the **preprocessing layer**, where the NetFlow/IPFIX [15]⁴ data from the network is received, pre-processed and possibly sampled using a dynamically selected **sampling strategy**. The preprocessing layer is followed by the **detection layer** where a set of anomaly detection methods performs a collective analysis of the received data using the statistical and information theoretical models of network traffic. The detection layer of CAMNEP has been designed to adapt to the network situation and to select the optimal **aggregation strategy**: the mix of weights to assign to the outputs of the anomaly detection methods, so that the system can dynamically find the best combination of opinions for the given environment. Selection of both the sampling algorithm and rate and the appropriate aggregation strategy (that are obviously linked) are crucial for system performance.

⁴The NetFlow data was originally created for traffic management and accounting purposes. It does not contain the content of the communication, but is formed by a set of simple records, one record created for each flow. The flow is a unidirectional stream of packets with a shared source and destination IP address and port, and its record also contains the number of packets/bytes transferred, timestamps and duration. NetFlow is widely available from routers, switches and other network devices.

Characterization of the immediate system performance is a difficult problem, as the manual feedback is nearly impossible because of the sheer volume and rate of the input data and general unavailability of any ground truth about their nature. Therefore, we use a challenge insertion mechanism that modifies the input data of the system by inserting a small number of classified events from the past, belonging to both legitimate traffic and various attack classes. These **challenges** [13] are processed alongside of the real input data, used for system component/strategy characterization/evaluation and then removed from the output data produced by the system.

The game as described below is actually played between **two opponents inside the system**. One of them, the defender P_D uses the challenge processing results to select the best sampling strategy x_D^1 and aggregation strategy x_D^2 (whose combination in the global game is denoted x_D) plays against the attacker's model P_A that determines and exploits the system vulnerabilities in the same manner as the worst-case real attacker would by playing its attack strategies represented by challenges: x_A^1, x_A^2 or the global strategy x_A . The dynamic optimization achieved by this method is crucial, as the system constantly reconfigures itself to counter the worst case attacker (with full access to its internal state) on the background of the current network traffic.

Specifically, the lower layer has to select a sampling method and ratio from the predefined set of 5 various sampling methods, each of them with two different ratio settings. The detection layer has to select a single aggregation function from a static finite set of aggregation functions.

The system operates sequentially, processing one batch of data every 5 minutes. Each data set defines one game in the sequence, when the challenges are inserted, all players use the results of their processing to determine the regret value for each available strategy and then commit to the strategy selected by the algorithm presented in Sec. II. The final strategies of the defender (or defenders) is then used to process the real network data from the current batch.

Defender (CAMNEP) has to satisfy the safety demands, i.e. it tries to lower the probability of the undetected attacks, and at the same time has to optimize the usage of the computational resources. On the other side, the attacker tries to attack the system while avoiding the detection. Thus the attacker has to find best attack type and size to fit under the detection threshold, by exploiting the flaws either in the sampling or aggregation strategy.

A. Global Game

In the global game the defender performs the global optimization and identifies the best strategy x_D from the set X_D , effectively looking for the best combination of input data sampling and anomaly detector's aggregation scheme.

The utility function of the defender in the global game has three principal components: the first term deals with successfully detected attacks, the second term represents the loss associated with undetected attacks and the third term

describes the overhead of the monitoring, which consists of the false positive costs and the fixed cost of monitoring.

Individual utility functions of the global game for both players (adapted from [16] and made more realistic) are defined as follows. Defender's utility is:

$$u_D(x_D, x_A) = \alpha(D_D(x_D) - C_{TP}) + (1 - \alpha)\gamma P_D(x_A) - \beta V C_{FP} - C_M \quad (21)$$

where the α denotes the probability that a particular attack strategy x_A is detected when the defender selects the defense strategy x_D ; $D_D(x_A)$ denotes the defender's payoff for attack detection; C_{TP} - denotes the (average) cost of processing of each successfully detected incident (true positive) for the defender; γ denotes the probability of attack success; $P_D(x_A)$ denotes the defender's payoff/loss on attack success; β denotes the probability that a given detection strategy, combined with current system state and background traffic, will result in a false positive; V denotes the background traffic volume that is used to estimate the number of false positives in combination with the parameter β ; C_{FP} - denotes the average cost of a false alarm for the defender, used in conjunction with β and V to estimate the false positive cost; C_M denotes the fixed cost of monitoring infrastructure, independent on attack or traffic intensity.

Attacker's utility can be described as:

$$u_A(x_D, x_A) = \alpha D_A(x_A) + (1 - \alpha)\gamma P_A(x_A) - C_A(x_A) \quad (22)$$

where α and γ are the same as described above; $D_A(x_A)$ denotes the attacker's payoff/loss on detection; $P_A(x_A)$ denotes the expected utility the attacker receives upon successful realization of given attack action from the attack class corresponding to strategy x_A ; $C_A(x_A)$ denotes the cost of the attack performance on the part of attacker.

B. Distributed Game

As we have defined in Sec. III, the optimization problem in distributed game is separated into two different games solved independently. Thus, we have to define four different utility functions for four independent players P_A^1, P_D^1 and P_A^2, P_D^2 .

On the first layer the player P_D^1 , i.e. the defender, optimizes the selection of the best sampling method which preserves as much information necessary for detection as possible and at the same time optimizes the volume of the traffic. These contradictory requirements can be summarized into following equation:

$$u_D^1(x_D^1, x_A^1) = \xi_D \left(1 - \frac{|\Phi_s(x_D^1)|}{|\Phi|}\right) + \vartheta_D \frac{|I_s(x_D^1)|}{|I|} + \mu_D |\{c \in C \mid |c_s(x_D^1)| > \varepsilon\}| \quad (23)$$

where $|\Phi_s(x_D^1)|$ is a number of flows in the dataset after the sampling using the strategy x_D^1 ; $|\Phi|$ represents the number of flows in the whole dataset before sampling; $|I_s(x_D^1)|$ is the

number of distinct IP addresses in the sampled dataset; $|I|$ is a number of distinct IP addresses in the unsampled dataset; C is the set of challenges inserted into the traffic, as described in Section IV and $|c_s(x_D^1)|$ represents a number of flows from inserted challenge c after the sampling, using sampling method x_D^1 . The first of the additive terms emphasizes the need to sample out as much traffic as possible, the second, on the other hand, emphasizes the need to preserve as much diversity as possible, using the source IP as the most important feature, and the third term represents the need to keep enough information from each challenge (here used as a representative of a typical incident) for further analysis. All three terms are contradictory, and $\xi_D = 1$, $\vartheta_D = 2$ and $\mu_D = \frac{1}{2}$ are constants assigning the relative importance to the different parts of the utility.

Attacker’s utility function can be summarized as follows:

$$u_A^1(x_D^1, x_A^1) = \frac{1}{|C|} \sum_{c \in C} \left(1 - \frac{|c_s(x_D^1)|}{|c|}\right) \quad (24)$$

where C is the set of all challenges inserted in the current dataset, term $|c|$ represents the volume of the challenge c , i.e. the number of flows and $|c_s(x_D^1)|$ again represents the number of sampled flows from challenge c obtained using the sampling strategy x_D^1 . Thus, from the Eq. 24 can be seen that the goal of the attacker in the first level of the game is to perform an attack with volume of traffic as high as possible (thus increasing the speed of probing or brute force operations) which will be at the same time sampled as little as possible.

Utility functions in the detection layer are nearly identical as in the global game for both attacker and defender. This is logical, as their values determine the same output, the only differentiation is formal. It is due to the fact that the first level sampling strategy is imposed by the first layer, rather than optimized together with the aggregation (or attack) strategy.

The second layer of neither the attacker, nor the defender gets no information regarding the strategies used in the first layer and the only implicit information passed between the layers is the shape of the data after the application of the sampling strategy.

V. EXPERIMENTAL EVALUATION

The experiments were performed on the CAMNEP system deployed on the university network and processing the data from one campus building – 200-300 users, with approximate bandwidth in hundreds of Mb/s and about 30-50 thousand flows in each 5-minute long dataset. We have used 70 datasets in a series, capturing roughly 6 hours of traffic.

A. Experimental Results

The first property to verify is that the distributed versions of the game actually do converge into some (possibly mixed) equilibria state and would not be cycling between incompatible states as in some of the theoretically known cases [11]. In Fig. 3 and Fig. 4, for the sampling game and the detection game respectively, we can see that the games do indeed converge into mixed equilibria where 3-4 strategies are selected

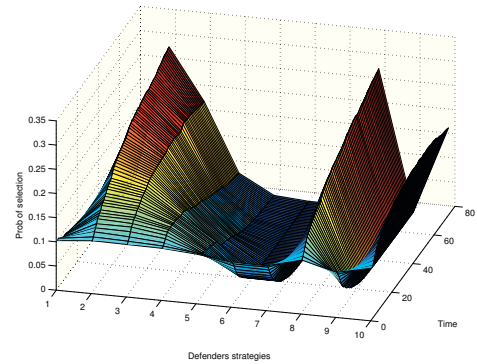


Fig. 3. Selection probabilities of individual strategies in the defender’s sampling game. Note the progressive disappearance of most strategies and rapid convergence to the strategies 2,3,8 and 10.

with high probability, while the other states are selected with smaller frequencies.

In Fig. 4, we can note that the strategy 15 was one of the strategies appropriate at the beginning, but its selection probability decreased constantly – this is due to the changing characteristics of the traffic during the experiment and the convergence of sampling strategies in the lower layer.

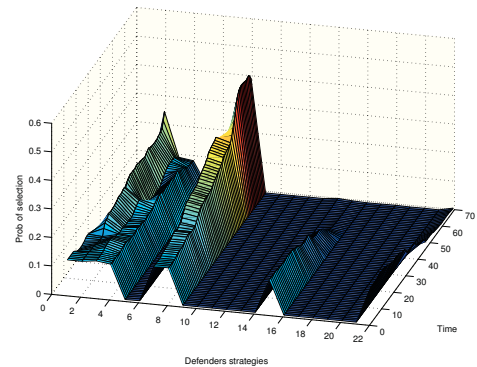


Fig. 4. Selection probabilities obtained by PWM algorithm in the defender’s detection game. Note the progressive dominance of the strategy 7/8, then decrease of the strategy 7 after the $t=50$ and corresponding improvement of the strategy 1.

The Fig. 3 and Fig. 4 do indeed show a robust convergence, but the question remains regarding where do they converge. In Fig. 5 and Fig. 6, we visualize the history of gameplay. For each dataset (i.e. one game in the sequence), we show the results of regret minimization (the payoff of the strategy selected) and the payoff of the Nash equilibria in that given game. Both values are shown for both the attacker and defender, each on its own axis.

Fig. 5 shows that the convergence in the sampling game is fast and that we converge into the region very close to the actual Nash equilibria, defender being slightly closer. The payoff differences from the equilibria are in the order of 5% to 10% of the potential payoff in this game stage. It is also interesting to note that most games do have a single Nash equilibria, with the set of correlated equilibria also collapsing to very small region around Nash – therefore, we only show

Nash equilibria in the following to preserve clarity.

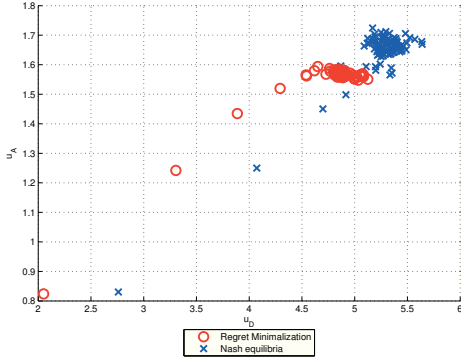


Fig. 5. Regret payoff values vs. the Nash equilibria (mostly co-incident with correlated equilibria) payoffs for the sampling game. Not rapid convergence to the region near the Nash equilibria.

In Fig. 6, we can see that the convergence of the detection game in terms of payoffs is less robust for the defender. Given the fact that the values of the Nash equilibria also fluctuate, this shows that the detection game is far more sensitive to the choice of the challenges inserted into the traffic [13] and that the payoffs of the same strategy (which is in itself robust, as we can see in Fig. 4) can vary widely.

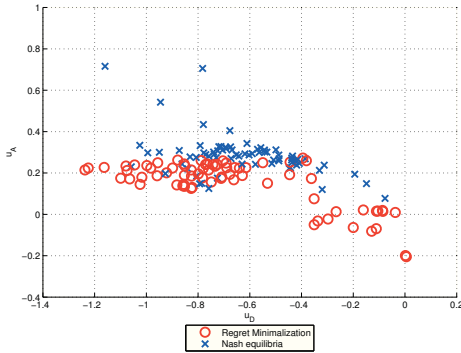


Fig. 6. Regret payoff values vs. the Nash equilibria (also mostly co-incident with correlated equilibria) payoffs for the detection game. Convergence of payoffs for the defender is worse than in Fig. 5, as their immediate values are influenced by the selection of challenges inserted in the current dataset.

Slower convergence in Fig. 6 can be also attributed to the influence of the co-learning with the sampling layer, as the results of the u_D^2 depend on the unknown strategy x_D^1 . We actually consider the robust convergence of the detection layer’s strategies despite the important noise in the payoff as an encouraging property, as this is precisely the difference between the real systems and synthetic experiments.

The Figure 7 shows the evolution of the strategies in the global game - note that the strategy space is far larger in this case. The strategies converge to mixed equilibria of about 10 strategies. **Most of the top detection strategies selected are identical to the ones identified in Fig. 4** – and this provides us with another hint about the stability of distributed regret minimization. On the other hand, the strategies selected for the sampling are different: the global view allows the

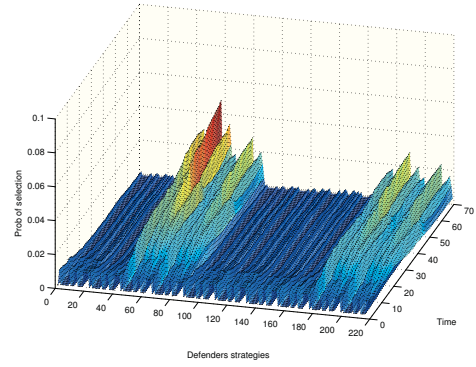


Fig. 7. Convergence of regret minimization in the global game. Note far larger strategy space and the larger size of the mixed strategy identified: it consists of roughly 12 strategies from the set of 220, with the dominant one being selected in about 5% of cases.

system to opportunistically select higher sampling rates (of the same sampling algorithms), as it can determine that their impact on attack detection is small. Reaching the near-identical equilibria would be possible had we manually tweaked the coefficients in Eq. 23, but we have preferred a conservative approach preserving more data – and got actually outsmarted by the optimization algorithm. However, the actual difference between the payoffs can be safely ignored in operational settings.

For the lack of space, we do not make any quantified claims regarding the performance of the system, which remains nearly equivalent in most of the actually selected strategies. We will include these in a subsequent more detailed publication which will also include some key details about the IDS settings, the analyzed traffic and other important parameters. The goal of our experiments is merely to assess the differences between the distributed and global regret minimization and to show that being social does not always pay off.

VI. RELATED WORK

The problem of reconfiguration and parametrization (both runtime and offline) of intrusion detection systems has been addressed from several perspectives, even if the direct use of game-theoretic principles is relatively rare and most contributors present the solutions applied for offline use [17], [18], [16], [19], when they identify the environment parameters static system and adapt the IDS to this static viewpoint. Roy et al. [1] present a nice overview of game-theoretical models of network intrusion detection problem.

The initial work of Alpcan [17] analyzes the IDS game as a sequence of interactions between a strategically reasoning opponents and a network of IDS sensors, in the format of two player, single act finite game with dynamic information. In [18], Alpcan and Basar extend [17]. Their formalism, based on a combination of Markov games and Q-learning, actually links the agent’s performance as a detector/learner to its game performance by representing the imperfect information.

The utility function that we use is based on the work of Chen [16], but we have included supplementary terms

that represent the real world concerns – and make the game decidedly a non-zero sum game.

One of the principal problems for the use of presented methods is the problem system characterization, that we address by use of challenges. The challenges need to be generated and they need to be representative of the real system. Vejandla, Dasgupta et. al address this problem in [20], where they use multi-objective evolutionary techniques on simulated network traffic to build complex strategies in the anticipation games.

VII. CONCLUSIONS

This paper contributes to the development of robust and survivable distributed intrusion detection systems (and also other systems where the same abstract model can be fitted) by introducing the collective adaptation paradigm which is not based on explicit communication, but on the individual adaptation of components working on the shared data. From the research perspective, our work is based on the results regarding the convergence of regret minimization approaches to the set of correlated equilibria [7][6] and a more restricted result regarding the bounds of regret minimization in information sets of extended games [12]. Both results suggest that explicit communication and collaboration only brings limited benefit, at the expense of far worse system flexibility.

We may be tempted to achieve better synchronization at least by signalling from the lower layers to the others, hinting on the strategy adopted by them and thus reducing the size of the information sets in which the second layer (and by extension subsequent layers) performs. However, while this is a valid possibility for static systems deployed on stable networks, this would typically require amending the current de-facto data format standards (tcpdump, NetFlow, various formats of syslog messages) and reducing the opportunity to replace a damaged component by opportunistically deployed replacement. Therefore, **we argue that strict adherence to standard data formats and individual adaptation of components processing, modifying and producing the data in standard formats is a valid approach for construction of survivable intrusion detection systems**, and the results presented in this paper show that such systems do not take significant performance hit when compared to centrally optimized versions.

ACKNOWLEDGMENT

This material is based upon work supported by the ITC-A of the US Army under Contract No. W911NF-10-1-0070. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the ITC-A of the US Army. Effort is also sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-10-1-3016. Also supported by Czech Ministry of Education grants 6840770038 and AMVIS-AnomalyNET, MSMT ME10051.

REFERENCES

- [1] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu, "A survey of game theory as applied to network security," in *HICSS*. IEEE Computer Society, 2010, pp. 1–10.
- [2] M. Osborne and A. Rubinstein, *A Course in Game Theory*. MIT Press, 1994.
- [3] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic Game Theory*. New York, NY, USA: Cambridge University Press, 2007.
- [4] R. J. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, January 2001.
- [5] M. Reháč, M. Pechoucek, M. Grill, J. Stiborek, K. Bartoš, and P. Celeda, "Adaptive multiagent system for network traffic monitoring," *IEEE Intelligent Systems*, vol. 24, no. 3, pp. 16–25, 2009.
- [6] A. Blum and Y. Mansour, "learning, regret minimization and equilibria," in *Algorithmic Game Theory*, N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, Eds. Cambridge University Press, 2007, ch. 4, pp. 79–101.
- [7] S. Hart, "Adaptive Heuristics," *Econometrica*, vol. 73, no. 5, pp. 1401–1430, Sep. 2005. [Online]. Available: <http://www.blackwell-synergy.com/doi/abs/10.1111/j.1468-0262.2005.00625.x>
- [8] S. Hart and A. Mas-Colell, "A simple adaptive procedure leading to correlated equilibrium," *Econometrica*, vol. 68, no. 5, pp. 1127–1150, September 2000.
- [9] R. Aumann, "Correlated equilibrium as an expression of Bayesian rationality," *Econometrica: Journal of the Econometric Society*, 1987. [Online]. Available: <http://www.jstor.org/stable/1911154>
- [10] S. Hart, "Nash equilibrium and dynamics," Center for Rationality and Interactive Decision Theory, Hebrew University, Jerusalem, Discussion Paper Series dp490, Sep. 2008.
- [11] J. Shamma and G. Arslan, "Dynamic fictitious play, dynamic gradient play, and distributed convergence to Nash equilibria," *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 312–327, Mar. 2005. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1406126>
- [12] M. Zinkevich, M. Johanson, M. H. Bowling, and C. Piccione, "Regret minimization in games with incomplete information," in *NIPS*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. MIT Press, 2007.
- [13] M. Reháč, E. Staab, V. Fusenig, M. Pechoucek, M. Grill, J. Stiborek, K. Bartos, and T. Engel, "Runtime monitoring and dynamic reconfiguration for intrusion detection systems," in *Recent Advances in Intrusion Detection, 12th International Symposium, RAID 2009, Saint-Malo, France, September 23-25, 2009. Proceedings*, E. Kirda, S. Jha, and D. Balzarotti, Eds., 2009, pp. 61–80.
- [14] M. Rehak, E. Staab, M. Pechoucek, J. Stiborek, M. Grill, and K. Bartos, "Dynamic information source selection for intrusion detection systems," in *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '09)*, K. S. Decker, J. S. Sichman, C. Sierra, and C. Castelfranchi, Eds. IFAAMAS, May 2009, pp. 1009–1016.
- [15] B. Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC 5101 (Proposed Standard), Internet Engineering Task Force, Jan. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5101.txt>
- [16] L. Chen and J. Leneutre, "A game theoretical framework on intrusion detection in heterogeneous networks," *Information Forensics and Security, IEEE Transactions on*, vol. 4, no. 2, pp. 165–178, June 2009.
- [17] T. Alpcan and T. Başar, "A game theoretic approach to decision and analysis in network intrusion detection," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, December 2003, pp. 2595–2600. [Online]. Available: papers/cdc03_alpcan_WeP03-1.pdf
- [18] —, "An intrusion detection game with limited observations," in *12th Int. Symp. on Dynamic Games and Applications*, Sophia Antipolis, France, July 2006. [Online]. Available: <papers/isdg06.pdf>
- [19] G. Wagener, R. State, A. Dulaunoy, and T. Engel, "Self adaptive high interaction honeypots driven by game theory," in *SSS*, 2009, pp. 741–755.
- [20] P. Vejandla, D. Dasgupta, A. Kaushal, and F. Nino, "Evolving gaming strategies for attacker-defender in a simulated network environment," *Social Computing / IEEE International Conference on Privacy, Security, Risk and Trust, 2010 IEEE International Conference on*, vol. 0, pp. 889–896, 2010.