1

Nonfunctional requirements validation using nash equilibria

Vicky Papadopoulou and Andreas Gregoriades
European University Cyprus
Cyprus

1. Introduction

The *Network security* aims to protect the network and the network-accessible resources from unauthorized access. However, the dynamic characteristics of contemporary networks combined with their increased size makes the vision of absolute network security almost impossible. Specifically, networks are vulnerable to infection by different types of electronic attacks such as viruses, Trojan horses or eavesdroppers that exploits the loopholes in the security mechanisms of networks [FAGY00]. Guaranteeing an acceptable level of security for a prospective system represents a common problem in systems engineering. Network security, is defined as a *Non-Functional Requirement* (NFR) that is influenced by functional aspects of the system such as the specification of antivirus and firewall mechanism employed on the network. This area of research has gained considerable popularity due to the implications it has on users' satisfaction, business reputation and performance. Therefore, being able to quantify the security level of a future network early in the design phase is of vital importance to its sustainability. The need to validate security requirements early has been addressed also by Lamsweerde [CILN02] and Crook [L04].

Unlike functional requirements, which can be deterministically validated, NFRs are soft variables that cannot be implemented directly; instead, they are satisfied by a combination of functional requirements. NFRs define the overall qualities or attributes of the resulting system and as such place restrictions on the software product being developed. Examples of NFR include safety, security, usability, reliability and performance requirements. Typical approaches to validating NFRs include, formal methods, prototypes, system simulations [AG05] and use of scenarios.

Model-checking techniques have been used extensively to verify and validate requirements. However, when its comes to NFR model checking is not adequate. Scenario-based requirements analysis methods, pioneered by Potts [P99] , Potts and Anton [P98], and Potts et al [P94], proposed that obstacles or difficulties which might prevent a goal being achieved should challenge requirements and, hence, promote refinement of the requirements specification to deal with such obstacles. This approach was developed by van Lamsweerde [L01] and van Lamsweerde and Letier [L00] , who applied formal reasoning to requirements

specifications to infer whether goals could or could not be achieved given constraints imposed by obstacles. Hierarchical goal decomposition produced specifications of the states to be achieved and the system behavior required to reach those states, so considerable problem refinement was necessary before automated reasoning could be applied. These approaches also assumed that a limited number of scenarios and their inherent obstacles are tested. This raises the question of test data coverage, i.e., just what is a sufficient set of scenarios to enable validation to be completed with confidence? While we believe there is no quick answer to this vexing problem, one approach is to reduce the set of scenarios that needs to be tested to achieve adequate validation.

This chapter addresses the aforementioned problem of generating large numbers of test scenarios during a typical scenario-based requirements validation process through Game Theory. Specifically, we reduce the complexity of the solution space to a manageable set by focusing only on combinations of strategies that satisfy the both defenders and attackers of a network. In this work, we apply game theory to assess the security NFR of a prospective network prior to its implementation and as such provide a validation of the security NFR. The assessed security NFR represents the minimum level of security guarantee for a prospective network, given a number of immunity requirements to be implemented in the network. These requirements correspond to antivirus software and their location on the network. Specifically, in the problem scenario we address in this chapter we assume that a number of harmful entities or *attackers* (or an upper bound of this number) may hit anywhere in the network. Attacks target nodes of the network. When, there is no information on how the attackers are placed on the network nodes, one may assume that they follow a *uniform distribution*. The immunity functional requirements of the network describe its defence mechanisms and are expressed by a set of *defenders*; software security systems that should guarantee an acceptable level of security to a part of the network (a link, a path, or a subnetwork). Attackers damage targeted nodes unless these are guarded by a defence software. Lamsweerde in [L04] also refers to the need to analyze the rational of the attacker in an attempt to become proactive rather than reactive in network security management. Lamsweerde refers to anti goals and anti requirements that define the attacker's strategies based on which the network designers specified functional requirements to tackle these.

1.1 Network Security NFR

Network Security is considered an important non-functional requirement needed to be guaranteed in a prospective computer network. Thus, it should be validated early in the design phase. Maintaining acceptable level of security in a network is analogous to preventing attacks on a country by deploying appropriate defences. Network security NFR corresponds to the ability of a network to successfully prevent attackers from maliciously exploiting its' information technology resources. With adequate security, attacks could be stopped at their entry points before they spread into the network. This requirement however, is impossible to achieve most of the times, due to the level of complexity, size and dynamic nature of contemporary computer networks. As a result designers seek to identify the best network configuration given the desire security level to be achieved using different configurations of immunity requirements.

Recent work by [KO04, ACY05] and [MPPS05b, MPPS05c], initiated the introduction of strategic games on graphs (and the study of their associated Nash equilibria) as a means of studying security problems in networks with selfish entities. By selfish we mean that each entity in the game aims to maximize its utility. In the security games studied in [KO04], a large number of players must make individual decisions related to security. The ultimate safety of each player may depend in a complex way on the actions of the entire population. [MPPS05b, MPPS05c] considers a security problem on a distributed network modeled as a multi-player non-cooperative game with *attackers* (e.g., viruses) and a *defender* (e.g., a security software) entities. More specifically, there are two classes of confronting randomized players on a graph: α *attackers*, each choosing vertices and wishing to minimize the probability of being caught, and a single *defender*, who chooses edges and gains the expected number of attackers it kills. A subsequent work [MMPPS06] introduced the *Price of Defense* in order to evaluate the loss in the provided security guarantees due to the selfish nature of attacks and defenses. This notion can be also seen as a (negative) measurement of the network security. A collection of polynomial computable Nash equilibria with guarantee defense ratio (i.e. security level) is presented.

1.2 Road Map

The paper is organised as follows. Firstly, we illustrate the principles of game theory, followed with a description of the approach. The important question that arises here is the following: '' Given the limited capabilities of the system security software, which part of the network should it choose to clean or protect from possible attack, so that the security level achieved is at least equal to the required level specified by the network designer?''

2. Game Theory

Game Theory is a branch of applied mathematics that attempts to analytically model the rational behavior of intelligent agents in strategic situations, in which an individual's success depends on the decisions of others. While initially developed to analyze competitions in which one individual does better at another's expense, it evolved into techniques for modeling a wide class of interactions, characterized by multiple criteria.

Most of the existing and foreseen complex networks, such as the Internet, are operated and built by thousands of large and small entities (autonomous *agents*), which collaborate to process and deliver end-to-end flows originating from and terminating at any of them. Recently, Game Theory has been proven to be a powerful modeling tool to describe such *selfish*, rational and at the same time, decentralized interactions [C01, O94]. In particular, Game Theory was successfully utilized for analyzing and most importantly *evaluating* the performance of *existing* networks in various aspects. Examples of such performance aspects include makespan, throughput, latency, resource utilization, users' satisfaction as well as security guarantees [R05, R02, ACY05, ADTW03, KP99, T04]. At the same time, a significant branch of Game Theory, *Mechanism Design* [NR99] is used to design future networks given a number of functional requirements specifications.

Game Theory has been used to understand selfish rational behaviour of complex networks, e.g. the Internet, of many "agents" (consisting the *players* of the game). In such domains,

Game Theory models players with potentially different goals (*utility functions* or *payoffs*), that participate under a common setting with well prescribed interactions (*strategies*), e.g. TCP/IP protocols. More importantly, it helps finding the *best strategy* of each player that will guarantee the best result. The core concept of Game Theory is the notion of *equilibrium* that is defined as the condition of a system in which competing influences are balanced.

2.1 Fundamental Components of Game Theory

The fundamental component of game theory is the notion of a *game*, expressed in normal form as $G=(M, A, \{u_i\})$, where G is a particular game, M is a finite set of players (decision makers) {1,2,…,m}, A_i is the set of actions available to player i, $A = A_1 \times A_2 \times \cdots \times A_m$ is the action space, and $\{u_i\} =\{ u_1 , u_2 , u_i , u_m \}$ is the set of objective functions that the players wish to maximize. For every player i, the objective function, u_i, is a function of the particular action chosen by player i, a_i, and the particular actions chosen by all of the other players in the game, a_{-i}.

A *profile* or strategy of a game σ is defined as the a setting of its players in term of possible actions or the probability distribution on a set of actions for each player of the game in setting σ. The action of player $i \in$ M is denoted by σ_i, where $\sigma_i \in A_i$.

The core concept of Game Theory is the notion of equilibrium that is defined as the condition of a system in which competing influences are balanced, i.e. steady-state conditions. More informally, in any game, a profile σ is a *Nash equilibrium* [Nash50, Nash51] if in σ no player would unilaterally choose to deviate from his chosen action as this would diminish his payoff. Intuitively speaking, Nash equilibria model well stables states of a network, since if the network reaches such a configuration, most probably it would remain in the same configuration, since none of the involving entities has a motivation to change his status in order to be more satisfied. Thus, identifying Nash equilibria configuration of a network and evaluating them has been the main approach in order to analyze, evaluate networks performance [ACY05, ADTW03 , CK05, KP99, MMP08, RT02].

Summing up, Game Theory and its various concepts of equilibrium provide a rich framework for modeling the behavior of selfish agents in distributed or networked environments. Moreover, it offers mechanisms to achieve efficient and desirable global outcomes given the selfish behavior of agents.

2.2. An Example Game: The Prisoners' Dilemma

The Prisoners' Dilemma [O94] game has two players (the prisoners): Bob and Al. Each of them has two possible strategies: to confess the other or not. Each of them should *simultaneously* decide which one of his strategies to follow (without knowing the choice of the other). Their choices determine their gain: If they both confess, each gets 10 years in prison , but if Al (resp., Bob) confesses and Bob (resp., Al) does not, Bob (resp., Al) gets 20 and Al (resp., Bob) goes free. Finally, if they both do not confess they both get 1 year in prison.

		Al	
		Confess	Don't Confess
Bob	Confess	10, 10	0, 20
	Don't Confess	20, 0	1, 1

Table 1. The Prisoners' Dilemma game.

Table 1 shows the players, the strategies and their payoffs (gain) for each of their strategy selections. Each prisoner can choose among one of the two strategies. In effect, Al chooses a column and Bob chooses a row. The two numbers in each cell tell the outcome for the two prisoners when the corresponding pair of strategies is chosen. The number to the left of the comma tells the payoff to the person who chooses the rows (Bob) while the number to the right of the column tells the payoff to the person who chooses the columns (Al). Thus (reading down the first column) if they both confess, each gets 10 years, but if Al confesses and Bob does not, Bob gets 20 and Al goes free.

Consider the following pair of strategies (*profile*) of the two players (*confess*, *confess*) corresponding to the strategy of Al and Bob respectively. Concerning Al, he gets 10 years in prison if he adopts this strategy, while he would get 20 years if he would not confess. Therefore his choice to confess is best for him. But the same reasoning holds also for Bob. Thus, the profile (*confess*, *confess*) consists best response strategies for all players of the game. This constitutes a Nash equilibrium of the game. Since all players use a single strategy in this profile, it is called *pure profile*.

Finding Nash equilibrium in this game seems to be not a difficult task. But in general games, there are more than two players involved with much more complicate payoff functions. This results to a significant increase of the difficulty to find Nash equilibrium. In particular, there are significant hardness results in finding pure Nash equilibria [FPT04], pointing to a whole complexity class (the PLS complexity class) which includes such searching tasks.

With regards to our approach to network security evaluation, a game is represented by a number of attackers and defenders that both aim to maximize their utility on the network, the former by maliciously degrading its performance and the latter by protecting it against attacks.

3. The Method

Assessing network security NFR is not a trivial task. An increasingly popular approach is to express this problem in the form of a game between attacker and defenders [AB04, B99, W08]. The former correspond to malicious software and the latter to defence software. When the designer starts thinking like an attacker, in essence he/she engages in a game with the attacker. Finding and evaluating equilibriums between attackers and defenders' strategies provide the mechanism to assess network's security. Therefore, this critical information can be provided during the design phase of a prospective network and hence, enable the designer to optimise network features accordingly.

The approach described in here is based on identifying Nash equilibria between attacker and defender strategies and in this way provide the means to assess the security level of prospective networks. These estimates can be subsequently used to validate security.

However, to validate a prospective network security NFR early in the design phase, prerequisite capturing its behaviour for all possible types of assaults. These combinations however, constitute a large number of possible test scenarios. Therefore, to evaluate the security performance of a prospective network we need to assess it against each of these possible test scenarios. Scenarios became a popular method for validating NFR [AS02, Car00] where each corresponds to a set of situations that might occur during the operation of a system. Application of scenarios in requirements validation has been performed by a number of researchers [AG05, AS02, AD93, ZJ00]. However, the main problem in requirements validation through scenarios is the specification of an adequate number of test cases. This however is a tedious and time consuming task. On the other hand, automated support for the scenario generation proved to be a vexed problem due to the exponentially large set of possible variations that needs to be examined [AG05] for the NFR to be guaranteed.

An approach that makes this problem tractable is described in here and is based on the application of game-theoretic analysis. In particular, we manage to reduce the number of scenarios needed to validate the NFRs by investigating only stable network states (configurations). This method is of polynomial time complexity compared to the size of the proposed network. Stable configurations describe the most likely states that a network could reside. Thus, by assessing security NFR in such states, we ensure the validity of the NFR almost always. Such states are very well captured through Nash equilibria profiles of the resulting game. Thus, we only utilize Nash equilibria in order to assess network security.

Our approach is composed of the following steps:

1) ***Functional and non-functional security requirement specification:*** Initially the network designer specifies quantitatively the required level of security of the future network as a percentage value. Moreover, the designer explicitly specifies the functional specification of the network in terms of security software capabilities and topology coverage.

2) ***Modeling of the functional security and network requirements:*** Model functional security requirement in the prospective network as a game played on a graph. In particular, we represent the network's topology using a graph and adopt a security game introduced in [MPPS05c]. According to this approach, the security threats and the potential defence mechanisms are realized by a set of confronting players on a graphical game.

3) ***Validation of the non-functional security requirement:*** We utilize the Nash equilibria identified and evaluated in [MPPS05c] to measure the security guarantee in the prospective network for both approaches. These represent a reduced set of test scenarios to be evaluated. Since Nash equilibria model well the stable configurations of the network, we ensure the validity of the NFR in the most probable states of the network. Evaluating of the Nash equilibria of the resulting game [MPPS05c] provides a novel validation method of the security NFR of prospective networks.

3.1. Case-study

We next illustrate the application of our method in an example network. The method is applicable in any network that fulfills the functional requirements specified a priori. The corresponding security NFR is initially defined as a percentage of the required level of security. Finding equilibria through Game Theory enables the designer to identify "stable" network configurations and subsequently evaluate whether these can archive the required level of security. The security NFR is satisfied if the assessed security meets the initial requirement. Therefore, the core problem in validating security is to firstly provide the means to assess it.

Our approach is based on the notion of *scenarios* [Car00], each describing possible configurations of attackers and defenders on the network. The use of Game Theory enables us to reduce the complexity of this process by analysing only scenarios that both attackers and the defender would choose given that they act *rationally*-they act in a way that aims to maximizes their benefit. Through game-theoretic analysis, strategies of both attackers and defenders on a network are modeled accordingly to assess the network's security.

Next we illustrate the application of the method for a network characterized by a set of functional requirements.

3.1.1. Functional Security Requirement Specification

A precondition for the method is that the network is of type "hit-all". This means that the network N consists of an arbitrary number of nodes, n and a set of communication links E between the nodes of the network. Moreover, there exists a subset of the links $E' \subseteq E$ such that each node υ of the network is "*hit*" (incident) to *exactly* one link of the set E'. Note that a network with this property can be build and identified (that has fulfills the property) in polynomial time [LP86] (such a set is called a *Perfect Matching* of the network). We call such a network a *hit-all* network. For example, in the network of Figure 1, node υ_1 is hit by links e_1, e_2 and e_3 shown with thick lines. Moreover, the thick links constitutes a hit-all set for that network.

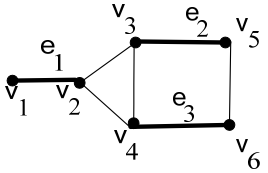

Fig. 1. An example of a network with a hit all set of links shown with thick lines.

We specify *network security specification* using a common process utilized in critical systems specifications [S05]. The process consists of the following components:

1. *Asset identification:* The assets of the network are the nodes of the network. In the most general case, all nodes are of the same importance. A node is considered protected or secure if a security software is installed on that node. Otherwise it is considered vulnerable to attacks.

2. *Threat analysis and assignment:* The prospective network may witnessed threats, such as viruses, Trojan horses and eavesdroppers [FAGY00] which are described as attacks that target the nodes of the network. At any time there is a maximum number of attackers, α, that may be present in the network. Each of them damages nodes that are not protected. In the most general case, we have no information on the distribution of the attacks on the nodes of the network. So, we assume that attacks will follow a uniform distribution [T01], which is quite common in such cases. So, we assume that each attacker decides to attack or not a node of the network with the same probability. We call such attacks *uniform attacks*.

3. *Technology analysis:* One major security mechanism for protecting network attacks are the firewalls, that we refer to as defenders. Furthermore, in distributed firewalls [17] the network that is protected includes the links spanned by the nodes that participate in the distribution of the defenders. However, due to financial costs (e.g., the prohibitive cost of purchasing global security software) or from performance bottlenecks (e.g., the reduced usage of the protected part of the network) distributed mechanisms are only able to clean a limited part of the network. There are two possibilities with regards to the functional specification of the protection mechanism:

(a) The simplest case is when the security mechanism resides on a single link of the network and hence protects the two nodes that the link connects. We call this specification as *single-edge–protection* specification.

In this case we assume that the prospective network is supported by a *single* security software, denoted as d, which is able to clean a *single* link between two nodes from possible attackers at the endpoints of that link.

The distribution of defenders on the network's nodes exploits the topological property of the network as presented in the specification. That is, there is a set of links E' in the network such that any node is hit by (exactly) one link of that set. In particular, we assume defense mechanism chooses one link among that set E' with the same probability that is uniformly at random. We call this placement of the defense mechanism as *uniform-hit-all*.

(b) In the general case when the security mechanism covers a set of links k, where $k > 1$ but $k < E$. We call this specification as *multiple-edge–protection* specification.

So, in this case we assume that the network is supported by a security mechanism, denoted by d_k, which is able to clean a *set k* of links between two nodes from possible attackers at the endpoints of any link in the set.

In this case, there is a set of links E' in the network such that any node is hit by (exactly) one link of that set. It is assumed that the defense mechanism is placed on a set of k links among the set E'. We call this placement of the defense mechanism as *k-edges-hit-all*.

In this work we consider both *uniform-hit-all* and *k-edges-hit-all* that correspond to *single-edge–protection* and *multiple-edge–protection accordingly* security specification.

3.1.2. Modelling scenarios using Security and Network properties

This activity aims to assess the security NFR of the prospective network using a number of scenarios. A game theoretical model of the proposed network is presented and subsequently the necessary tools and notions that enable its security quantification are explained.

We model both network and security specifications presented in section 3.1.1. using two graph-theoretic games introduced and investigated in [MPPS05c, MPPS05b, MMPPS06]. The game is played on a graph *G* representing the network *N*. The players of the game are of two kinds: the *attackers* players and the *defender* players, representing the attacks and the security software of the network. The attackers play on the vertices of the graph, representing the nodes of the network. We consider two scenarios for the defenders:

a) The defender plays on *the edges* of the graph, representing the links of the network. This case models the *single-edge–protection* security specification and calls this model *single-edge-protection game*.

b) The defender plays on *sets of k edges* of the graph, representing sets of links of the network. This case models the *multiple-edge–protection* security specification and calls this model *k-edges-protection game*.

3.1.2.1 Network Configurations

A *network configuration s* models the location (nodes) of attackers and defense mechanism (link or a set of links) on the network. The positioning of attackers and defenders may follow a probability distribution. That is, each attacker can target more than one node according to some probability distribution and similarly, the defense mechanism may protect more than one link according to another probability distribution. In such a case, have a *mixed* configuration of **s**. Otherwise, the configuration is said to be *pure*; one attacker on one node and the sole defender on one link. This constitutes another property of the scenario specification.

Example of the Single-edge-protection game.

Figure 2 illustrates a *mixed* configuration for an example network, N consisting of 8 nodes (n=8). It can be seen that the network is a hit-all type. We assume that there exists 3 different attackers (α=3). According to the threat analysis of the security specification, the attacks are uniform; and hence, the probability of an attacker assaulting any node of the network is equal to 1/n which is equal to 1/8. In the Figure, attacker *i* is indicated by X_i.

Next, in the technology analysis of the security specification we designate that the security software mechanism is a *single-edge–protection.* Hence, modeled using the single-edge-protection game. Moreover, according to the security specifications, the security mechanism uses a uniform-hit-all probability distribution on a set of links E'. Recall that E' is such that any node of the network is hit by (exactly) one link of that set. So, the defender chooses each links of this set with probability $1/|E'| = 1/4$. In Figure 2, the links, as well as their corresponding visiting probabilities, are indicated by Y and thick lines.

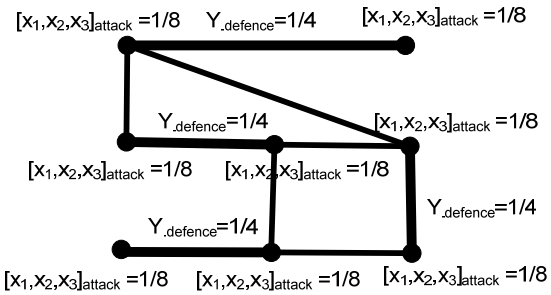

Fig. 2. An example of a network configuration for the Single-edge-protection game. We assume that there exists 3 different attackers (α=3). Each attacker is indicated by X. Each attacker targets any node of the network with probability 1/8. The security software chooses among a subset of links E' to clean them from possible attacks, uniformly at random. The links consisting the set E', and their corresponding visiting probabilities, are indicated by Y in thick lines. So, each link in the set is visited by the security software with probability 1/4. The assessed security level of this scenario is equal to 25%.

Example of the k-edges-protection game.

Figure 3 illustrates a network configuration for the same sample network of Figure 2 and the same scenario assumptions for the attackers. The scenario specification for the security software mechanism is defined as a *multiple-edge–protection.* Hence, modeled in a k-edge-protection game. Here, we assume that k=n/2. Moreover, according to the security specifications, the set of edges E′, that the defense mechanism can clean simultaneously, constitute a *k-edges-hit-all* set. That is, any node of the network is hit by (exactly) one link of the set *E*′. In Figure 3, the links of the set E′ are indicated by thick lines.

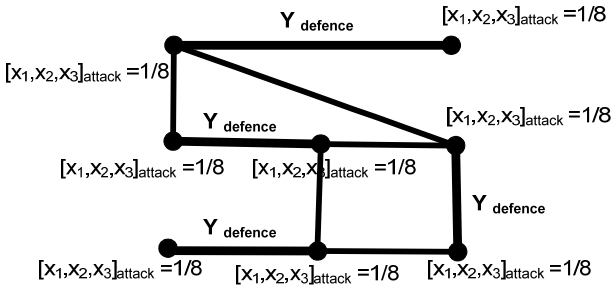

Fig. 3. An example of a network configuration for the k-edges-protection game. In this case the defense mechanism can clean k links at the same time; that is k=n/2. Also, the defense mechanism is placed on a set of links E′ such that the set is a *k-edges-hit-all* indicated with thick lines. The assessed security level of this scenario is equal to 100%.

3.1.3. Validation of the Non-functional Security Requirement

3.1.3.1 A Game-Theoretic Security Measurement

To evaluate network security it is necessary to assess the security level of an arbitrary profile (configuration) of the defined game of the prospective network similarly with [MPPS05c, MPPS05b, GMPPS06]. Therefore, consider a pure network configuration $\boldsymbol{s}$. Let $\boldsymbol{s}_d$ be the edges defended by the security software. For each attacker $i \in [\alpha]$, let $\boldsymbol{s}_i$ be the node in which the attacker strikes. We say that the attacker i is *killed* by the security mechanism if the node $\boldsymbol{s}_i$ is one of the two endpoints of the link $\boldsymbol{s}_d$ being defended by the security software. Then, the *defense ratio* [MMPPS06] of the configuration $\boldsymbol{s}$, denoted by $\boldsymbol{r}_s$ is defined to be as follows, when given as a percentage:

$$r_s = \frac{\text{number of attackers killed in } s}{a} \times 100. \quad (1)$$

For a mixed network configuration, the *defense ratio* [MMPPS06] of the configuration, $\boldsymbol{r}_s$ is defined as:

$$r_s = \frac{\text{expected number of attackers killed in } s}{a} \times 100. \quad (2)$$

From the above, the optimal defense ratio of a network equals to 100 if the security software manages to kill all attackers. In such a case we specify that the network configuration obtains 100% security level. The larger the value of $\boldsymbol{r}_s$ the greater the security level obtained. Through this approach, we assess the security level of perspective networks by only examining *stable* configurations and hence limited scenarios. Given that, whenever the network reaches a stable a configuration it *tents* to remain in that configuration, highlights the significance of evaluating scenarios that emerge from this to assess its security NFR. This is because in such configurations no single player has an incentive to unilaterally deviate from its current strategy. So, such configurations constitute the most probable states of the network and hence we use these to define the test scenarios based on which to assess security. Therefore, we escape from the NP-hard problem of having to assess each possible configuration or scenario. We identify such stable configurations evaluate the network security on them. Thus, this measurement constitutes a *representative* assessment of the security level of prospective networks.

Considering that the network designer wishes to achieve a security level of 90%, the following procedure is used to assess the security level for different network configurations. The main constrain of the approach is that it limits its scope to hit-all type networks.

Initially, we identify stable configurations resulting from the specifications by the Nash equilibria found in the game of [MMPPS06]. Thus, in order to evaluate network security we evaluate the Nash equilibria of the game of [MPPS05c, MPPS05b]. Indeed they showed a result which is interpreted in our terms as follows:

Theorem 1. [MMPPS06] *Consider a network N with n nodes such that the network and security and functional and non-functional specifications of Section 3.1.1 (case (a) of Technology analysis of Section 3.1.1) are satisfied. Then the network contains a stable configuration (i.e. a mixed Nash equilibrium)* $\boldsymbol{s}$ *where the expected number of attackers killed is 2/n. So, the defense ratio here is :*

$$r_s = \frac{2}{n} \times 100 \quad (3)$$

The result combined with equation (1) above implies that the network of Figure 1 has security level equal to 2/n×100=2/8×100=25%, since n=8. This designates that the level of security is 25% given the functional requirements specified in configuration $\boldsymbol{s}$. This assessment however indicates that the initial NFR specified by the designer is not satisfied using the prescribed functional requirements of the network as is. Hence, the network specification needs to be revised and the security NFR revalidated, prior to implementation.

We also use the following result:

Theorem 2. [GMPPS06] *Consider a network N with n nodes such that the network and security and functional and non-functional requirements given in section 3.1 (b) are satisfied and k=n/2. Then the network contains a stable configuration (i.e. a Nash equilibrium)* $\boldsymbol{s}$ *where all attackers are killed. So, the defense ratio is*

$$r_s = \frac{a}{a} \times 100 = 100 \quad (4)$$

The result implies that the network of Figure 2 has security level equal to 100% (recall that k=n/2 here) given the functional requirements specified in configuration $\boldsymbol{s}$. This assessment indicates that the NFR specified by the designer a priori is now satisfied using the prescribed functional requirements of the network.

4. Conclusion

Security requirements validation is traditionally performed through security-specific testing. Ideally, validation should be performed on all possible network conditions expressed by test scenarios. However, examining all possible scenarios [AD93, AS02] to validate security requirement early in the design phase of a prospective network, constitutes a highly complex and sometimes infeasible task. In this work we manage to accomplish this process in only polynomial time. This is achieved by considering only stable configurations of the system, that we model using Nash equilibria. This yields in a limited set of test scenarios that guarantee the assessment of network's security level. In this context, the method presented in this paper constitutes a novelty in validating security NFR through game theory.

5. References

[AB04] T. Alpcan and T. Basar, ``A Game Theoretic Analysis of Intrusion Detection In Access Control Systems,'' in Proceedings of *the 43rd IEEE Conference on Decision and Control* , Vol. 2, pp. 1568-1573, 2004.

[AD93] J. S. Anderson, B. Durley, ``Using Scenarios in Deficiency-Driven Requirements Engineering,'' in Proceedings of *the Requirements Engineering (RE'99)*, pp. 134-141, 1993.

[ADTW03] E. Anshelevich, A. Dasgupta, É. Tardos, and T. Wexler, ''Near-Optimal Network Design with Selfish Agents,'' in Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC), pages 511–520, 2003.

[ACY05] J. Aspnes, K. C hang, and A. Yampolskiy, `` Inoculation Strategies for Victims of Viruses and the Sum-of-squares Partition Problem,'' in Proceedings *of the 16th Annual A CM-SIAM Symposium on Discrete Algorithms (SODA 2005)* , pages 43--52. Society for Industrial and Applied Mathematics, 2005.

[B99] D. Burke, A game theory model of Information Warfare, USAF Air Force Institute of Technology, Air University, Master's thesis, 1999.

[Car00] J.M. Carroll, Making Use: Scenario-Based Design of Human-Computer Interaction, MIT Press, Cambridge, MIT, 2000.

[CHK05] G. Christodoulou and E. Koutsoupias, ''The Price of Anarchy of Finite Congestion Games,'' in Proceedings of *the 37th Annual ACM Symposium on Theory of Computing (STOC 2005)*, pages 67–73, ACM Press, 2005.

[CILN02] R. Crook, D. Ince, L. Lin and B. Nuseibeh, ``Security requirements Engineering: When Anti-Requirements Hit the Fan,'' in Proceedings of *the 10th Anniversary IEEE Joint International Conference of Computing (STOC 2004)* , pages 604—612, ACM Press, 2004.

[FPT04] A. Fabrikant, C. H. Papadimitriou, and K. Talwar, ''The Complexity of Pure Nash Equilibria,'' in *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC 2004)*, pages 604–612, ACM Press, 2004.

[FAGY00] M. Franklin, Z. Galil, and M. Yung, `` Eavesdropping Games: a Graph- Theoretic Approach to Privacy in Distributed Systems,'' *Journal of the ACM* , 47(2):225--243, 2000.

[GMPPS06] M. Gelastou, M. Mavronicolas, V. G. Papadopoulou, A. Philippou and P. G. Spirakis, "The Power of the Defender", CD-ROM Proceedings of the 2nd International Workshop on Incentive-Based Computing (IBC 2006), in conjunction with the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06), pp. 37, July 2006.

[AG05] A. Gregoriades and A. Sutcliffe, ``Scenario-Based Assessment of Non-Functional Requirements,'' Proceedings of the *IEEE Transactions on Software Engineering*, Vol. 31, no. 5, pp. 392-409, 2005.

[KO04] M. Kearns and L. Ortiz, ''Algorithms for Interdependent Security Games,'' in *Proceedings of the 16th Annual Conference on Neural Information Processing Systems (NIPS 2004)*, pages 288–297, MIT Press, 2004.

[KP99] E. Koutsoupias and C. H. Papadimitriou. ``Worst-Case Equilibria,'' in Proceedings of the *16th Annual Symposium on Theoretical Aspects of Computer Science* , pp. 404--413, Springer-Verlag, March 1999.

[L01] A. van Lamsweerde, ``Goal-Oriented Requirements Engineering: A Guided Tour,'' Proc. *Fifth IEEE Int'l Symp. Requirements Eng.* (RE '01), 2001.

[L00] A. van Lamsweerde and E. Letier, ``Handling Obstacles in Goal-Oriented Requirements Engineering,'' *IEEE Trans. Software Eng.*, vol. 26, pp. 978-1005, 2000.

[L04] A. van Lamsweerde, ``Elaborating Security Requirements by Construction of Intentional Anti-Models'', in Proceedings of *the 26th International Conference on Software Engineering*, pp. 148--157, 2004, IEEE Press.

[LP86] L. Lovasz and M. D. Plummer, *Matching Theory*, North-Holland Mathematics Studies, 121, 1986.

[NR99] N. Nissan, A. Ronen, "Algorithmic Mechanism Design," Proceedings of *the 31st Annual ACM Symposium on Theory of computing (STOC '99)*, pp. 129–140, 1999.

[O94] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, MIT Press, 1994.

[MPPS05c] M. Mavronicolas, V. G. Papadopoulou, A. Philippou, and P. G. Spirakis, A Graph- Theoretic Network Security Game, in Proceedings of *the 1st International Workshop on Internet and Network Economics (WINE 2005)* , volume 3828 of Lecture Notes in Computer Science , pages 969—978, Springer, 2005.

[MPPS05b] M. Mavronicolas, V. G. Papadopoulou, A. Philippou, and P. G. Spirakis, ''A Network Game with Attacker and Protector Entities", in *Proceedings of the 16th Annual International Symposium on Algorithms and Computation (ISAAC 2005)*, volume 3827 of *Lecture Notes in Computer Science*, pages 288–297. Springer, 2005.

[MMP08] M. Mavronicolas, B. Monien, and V. G. Papadopoulou, ''How Many Attackers Can Selfish Defenders Catch?" in *CD-ROM Proceedings of the 41st Hawaii International Conference on System Sciences, Software Technology Track, Algorithmic Challenges in Emerging Applications of Computing Minitrack*, January 2008

[MMPPS06] M. Mavronicolas, L. Michael, V. G. Papadopoulou, A. Philippou and P. G. Spirakis, "The Price of Defense", *Proceedings of the 31st International Symposium on Mathematical Foundations of Computer Science,* pp. 717–728, Vol. 4162, Lecture Notes in Computer Science, Springer-Verlag, August/September 2006.

[Nash50] J. F. Nash. ``Equilibrium Points in n-Person Games,'' Proceedings of *the National Academy of Sciences of the United States of America* , Vol 36, pp 48-49, 1950.

[Nash51] J. F. Nash, ``Non-cooperative Games'', *Annals of Mathematics* , 54(2):286--295, 1951.

[C01] C. H. Papadimitriou: ``Algorithms, games, and the internet``, Proceedings of the *33rd Annual ACM Symposium on Theory of Computing*, pp. 749-753, 2001.

[P99] C. Potts, ``ScenIC: A Strategy for Inquiry-Driven Requirements Determination,'' *Proc. Int'l Symp. Requirements Eng.*, 1999.

[P98] C. Potts and A. Anton, ``A Representational Framework for Scenarios of System Use,'' *Requirements Eng.*, vol. 3, pp. 219-241, 1998.

[P94] C. Potts, K. Takahashi, and A. Anton, ``Inquiry-Based Requirements Analysis,'' *IEEE Software*, vol. 11, pp. 21-32, 1994.

[RT02] T. Roughgarden and É. Tardos, ''How Bad is Selfish Routing?" *Journal of the ACM*, 49(2): 236–259, 2002.

[R05] T. Roughgarden, *Selfish Routing and the Price of Anarchy*. MIT Press, 2005.

[S05] I. Summerville, "Software Engineering", Seventh Edition, Addison Wesley, 2005.

[AS02] A.G. Sutcliffe and A. Gregoriades, ``Validating Functional System Requirements with Scenarios'', Proceedings of *the First IEEE Joint International Conference of Requirements Engineering (RE '02)* , Sept. 2002.

[T04] É. Tardos, "Network games, *Proceedings of the thirty-sixth Annual ACM symposium on Theory of computing*, pp. 341–342,2004

[T01] K.S. Trivedi, Probability and Statistics with Reliability, *Queuing, and Computer Science Applications*, John Wiley and Sons, New York, 2001, ISBN number 0-471-33341-7.

[W08] M. Wing "Scenario Graphs Applied to Network Security'', *Information Assurance: Survivability and Security in Networked Systems* , Chapter 9, Yi Qian, James Joshi, David Tipper, and Prashant Krishnamurthy, editors, Morgan Kaufmann Publishers, Elsevier, Inc., 2008, pp. 247-277.

[ZJ00] H. Zhu, L., Jin, ``Scenario Analysis in an Automated Tool for Requirements Engineering'', *Journal of Requirements Engineering*, 5 (1), 2-22, 2000.