Allocating Resources to the Search for Vulnerabilities In Information Infrastructures

Fabrizio Baiardi
Dipartimento di Informatica, Università di Pisa

Abstract A fundamental step of any security assessment of an information infrastructure is the search for vulnerabilities of ICT components. To optimise the investment in the search, first of all we introduce a mathematical model that evaluate the impact for the infrastructure owner as a function of the resources allocated to the search of vulnerabilities in distinct infrastructure components. Then, we introduce a zero sum game between an attacker and a defender, each managing a fixed amount of resources to be allocated to the search To prevent attacks, the resources allocated by the defender search for vulnerabilities to patch the infrastructure to remove them. Instead, the attacker resources search for vulnerabilities to exploit them and attack the infrastructure. Attacks results in a loss for the defender that, in the simplest case, is proportional to the time in-between the discovery of a vulnerability by an attacker resource and the discovery of the same vulnerability by a defender one. A loss is avoided only if and when a defender resource discovers a vulnerability before than an attacker one. In another case of interest, there is a delay between the discovery of vulnerability and the patching or the attack. We define conditions for Nash equilibrium where a player cannot improve its utility by changing its move only. We show that the corresponding allocation requires a large defender investment with a low return. Lastly a condition is introduced to evaluate when the adoption of open code components may reduce the defender investment and result in a better security.

Keywords: vulnerability, vulnerability window, open source, equilibrium

1. Introduction

A fundamental component of most critical infrastructures is an ICT network that interconnects the infrastructure components to a set of control rooms that manage the overall infrastructure according to the predefined goals set by the infrastructure owner. This ICT network will be referred to as the **information infrastructure** paired with the considered critical infrastructure. An information infrastructure consists of several interconnected components, some developed for the infrastructure and other commercial of the shelf, cots, components adopted to reduce the overall cost. A vulnerability [7, 8, 24, 25] is any defect of a component that enables an attack against the information infrastructure. If the attack is successful, the information infrastructure is controlled, to some extent, by the attacker rather than by the owner. When the attacker controls the information infrastructure, it can control and manage the whole infrastructure. Hence, the overall security of the infrastructure strongly depends upon that of the information infrastructure. Any security assessment of the whole infrastructure cannot neglect the corresponding information infrastructure [1-6].

By adopting the guidelines in [10], this paper considers cyber attacks against the information infrastructure and the search for vulnerabilities that enable such attacks. In particular, we assume that for each component of the information infrastructure there are two sets of people searching for component vulnerabilities. The goal of people in one set is to patch the component to remove any vulnerability that has been found. Instead, people in the other for vulnerabilities to attack the infrastructure. We assume that the patching [11] of the infrastructure invalidate any previous attack so that the loss due to an attack is proportional to the vulnerability window [8, 24, 25] of the vulnerability V enabling the attack, i.e. to time in-between the discovery of V by the attacker and the discovery of V by the defender. The window is larger than zero iff those interested in attacking the infrastructure discover V before than those that patch the infrastructure. We introduce a mathematical model that defines the average impact as a function of the vulnerability window and of the number of people searching for a vulnerability. Then, we define a strategy to allocate resources, i.e. people, to the search. To define this strategy, we introduce a zero sum game between an attacker and a defender, each managing a set of resources to be allocated to the components of the information infrastructure. Each

instance of the game consists of a pair of allocations, one for each player. Each allocation defines the resources the player allocates to each component. By exploiting the mathematical model previously defined, the allocations are mapped into the average vulnerability windows and the corresponding loss of the defender. This loss is the utility of the attacker and the inverse of the defender one. We define the condition for Nash equilibrium of the game in the case where the probability that a player finds a vulnerability in a component is geometrically distributed in the number of resources it has allocated to the component [12, 16]. In Nash equilibrium neither player can improve its utility by changing its allocation only, i.e. both allocations have to be changed to improve the utility of any player.

This paper is structured as follows. At first, it introduces the framework of the mathematical model to that relates the player utilities to the allocations, i.e. to the number of resources assigned to search for vulnerabilities in the components. We also discuss the optimisation of an allocation if the one of the opponent is known. Then, equilibrium conditions are discussed together with some preliminary applications to the adoption of open code components and some generalization of the player utility functions. Lastly we draw some conclusions and outline some future developments.

The importance of a quantitative evaluation of the relation between attack impacts and the investments in the search for vulnerabilities has often been stressed [5, 17, 18, 22, 26]. [23] presents a survey of current approaches to evaluate attack impacts and introduces the notion of market price of vulnerability. In an infrastructure, this price mostly depends upon the service supported by the infrastructure. [17] applies game theory to information warfare. [20] defines an insurance inspired allocation of resources to minimize the impact of a terrorist attack. The competition between defenders and attackers in the search for vulnerabilities and an optimal disclosure policy is considered in [9, 24, 25]. [9] considers the search for vulnerabilities and a social planner that decides when a vulnerability is disclosed. However, it neglects the case where a vulnerability is discovered by an attacker.

2. Modelling Resource Allocation As a Game

After discussing the main underlying assumptions and constraints, we present in some details the mathematical model that will be used to define the game to evaluate alternative allocation of resources searching for vulnerabilities in the components of the information infrastructure.

2.1. Underlying Assumptions

The proposed mathematical framework is based upon the *zero delay* model [10] as it assumes that both the patching and the attacks on the infrastructure occur **as soon as a vulnerability has been found**. A further significant assumption of the proposed framework is that the impact of a successful attack, i.e. the loss of the infrastructure owner, increases with the size of the window of a vulnerability V, i.e. with the time in between the discovery of V by those interested in attacking the infrastructure and the discovery of V by those interested in patching the infrastructure. This assumption is satisfied any time the goal of the attackers is an economic benefit rather the disruption of the infrastructure or the loss of human lives. Furthermore, for the moment being we assume that in each component of the information infrastructure there is only one vulnerability. This is relaxed in the following.

Our model assumes that, for each component there are two sets of people searching for the vulnerability V of the component. Let us consider at first the search of one set. The size of the set is N and it is fixed. The probability that each of the people in the set finds V in one time unit is p and it does not change with time. The assumption that p does not change with time is realistic provided that people searching for the vulnerability have been trained before starting the search. Furthermore, two people in the set have the same probability of finding V because they can access the same information.

Since the probability that at last one people in the set finds V in a time unit is $1-(1-p)^N$, the one that V is found after exactly t unit of time is geometrically distributed and equal to $q^{t-1}(1-q)$ where $q=(1-p)^N$. Hence, on the average, V is found at time AT where

$$AT = \frac{1}{1-q} = \frac{1}{1-(1-p)^N}$$

We consider now that there are two sets, S_1 and S_2 including, respectively, N_1 and N_2 people with probabilities p_1 and p_2 of finding V. The two searches for V are independent because we assume that the sets are disjoint and there is no exchange of information between them, hence the average difference between the times when V is discovered by the two sets is

$$\frac{1}{1-(1-p_1)^{N_1}} - \frac{1}{1-(1-p_2)^{N_2}}$$

Since no information flows between the two sets, no information from other people is available to speed up the search. The average difference between the two times can be larger or smaller than zero according to the set sizes. By exploiting the approximation

$$(1-p)^N \approx (1-pN)$$

the average difference may be rewritten as

$$\frac{1}{(N_1 p_1)} - \frac{1}{(N_2 p_2)}$$

if this value is larger than zero then people in S_2 discover V before than those in S_1.

2.2. Allocating Resources to Infrastructure Components

Consider now an information infrastructure including C components that are instances of *n* component types T_1, … , T_n. Let M_i, $1 \le i \le n$ denotes the number of components that are instances of T_i. According to the previous assumption, for each type T_i:

1. there is just one vulnerability V_i,
2. two sets of peoples, A_i and D_i, search for V_i. Let Na_i and Nd_i be the sizes of the two sets and pa_i and pd_i the probabilities that a person in the corresponding set finds V_i.

Peoples in A_i, the attackers, exploit V_i to immediately attack the infrastructure while those in D_i the defenders, immediately patch the infrastructure. If an attacker discovers V_i before than the defenders, then the attack is successfully executed and it has an impact, i.e. a loss for the infrastructure owner. Besides than to the number of attacks, the impact is proportional to M_i, to the window size and to the role of the component in the infrastructure. According to the proposed model, the size of the window for V_i is

$$\frac{1}{(pd_i Nd_i)} - \frac{1}{(pa_i Na_i)}$$

On the average, if $pa_i\ Na_i \le pd_i\ Nd_i$ then the defenders discover V_i before than the attackers so that $TypeLoss(T_i)$, the loss of the owner due to V_i is zero. Instead, anytime

$$pd_i\ Nd_i < pa_i\ Na_i \quad (1)$$

the attackers find V_i before than the defenders. In this case, the average impact is equal to

$$TypeLoss\ (T_i) = \delta_i \rho_i M_i \left(\frac{1}{pd_i Nd_i} - \frac{1}{pa_i Na_i} \right) \qquad \text{(a)}$$

where:

- ρ_i is the percentage of the M_i components that are attacked
- δ_i is the profit for unit of time of a successful attack to a component of type T_i.

Both constants depend upon the considered infrastructure. In the following, α_i denotes $\rho_I \delta_i$.

For some types, the definition of the attack and/or that of the patch may be so complex that the assumption of a zero delay in between the discovery of a vulnerability and either an attack or the patching may not be realistic. Taking into account that there is a loss anytime the infrastructure is not patched when the attack is ready, the impact for each of these types may be defined as:

$$TypeLoss(T_i) = \alpha_i M_i \left(\left(\frac{1}{pd_i Nd_i} - \frac{1}{pa_i Na_i} \right) + Tpa_i - Tatt_i \right)$$

where Tpa_i and $Tatt_i$ are the times to define, respectively, the patch and the attack and the last factor is the new size of the vulnerability window. Hence, there is a loss for T_i if

$$\frac{1}{pd_i Nd_i} + Tpa_i > \frac{1}{pa_i Na_i} + Tatt_i \quad (1')$$

The overall owner average loss AoLo is the sum of the average losses for all the types,

$$AoLo\ = \sum_{i=1}^{n} TypeLoss\ (T_i)$$

Obviously, only the types satisfying either (1) or (1') contribute to the sum.

3. Equilibrium and Optimal Allocations

At first, this section defines the allocation of resources to the search of vulnerability as a zero sum game that involves two players, the attacker and the defender, i.e. the infrastructure owner. Then we

consider how a player can optimise its allocation as soon as it knows the opponent one. Taking into account this strategy, we consider Nash equilibrium of the game and the condition for such equilibrium.

To define the resource allocation as a game, we introduce two players, the attacker and the defender, each managing a resource pool. The one of the attacker includes A resources, the defender one D resources. A player allocates each resource to the search for the vulnerability of one type. Obviously, the overall number of resources a player may allocate is equal to the size of its pool. An instance of the game consists of a pair of moves, i.e. of allocations, one for each player. The pair of allocations defines both Na_i and Nd_i, for each i, $1 \leq i \leq n$ and, as a consequence, a loss that corresponds to the attacker utility AU, i.e. AU=AoLo. If DU is the utility of the defender, then $DU=-AoLo$. Notice that D, the size of the defender pool should be at least equal to n, the number of types. In fact, anytime the defender cannot allocate at least one resource to each type, the loss it suffers may be unbounded.

3.1. Optimal Allocation

Consider an instance of the game we have defined and assume that both player moves are known. We say that the allocation of a player P is an optimal one if P cannot improve its utility given the N resources it manages and the opponent allocation.

With reference to definition (a) of the size of the vulnerability window and of the loss, we outline a procedure to compute an optimal allocation for a player P starting from the current player allocation. After determining an initial allocation in a first step, the procedure iteratively updates the current allocation by shifting one resource managed by P from a type T_i to a type T_j $i \neq j$ where i and j are chosen according to a local condition. By local we mean that it depends upon the resources allocated to T_j and T_i only. The number of iterations is bounded by the product of the number of resources and that of types.

At first, let us assume that P is the defender and consider an allocation where there is a type T_i such $pd_i\, Nd_i \geq pa_i\, Na_i$. In this case there is an **excess of defender resources** for T_i equal to

$$Nd_i - \frac{pa_i Na_i}{pd_i}$$

If in an allocation there is an excess of defender resources for T_i, then T_i does not contribute to the overall loss of the defender. When the excess for T_i is zero, there is equilibrium for T_i between the two allocations. While an excess of resources is always possible, the feasibility of equilibrium for T_i depends upon pa_i and pd_i. The following theorems hold.

Theorem 1

In any allocation where two types T_i and T_j exist such that there is an excess of defenders resources for T_j larger than one, i.e.

$$Nd_i - \frac{pa_i Na_i}{pd_i} > 1, \quad pd_j\, Nd_j < pa_j\, Na_j,$$

the shift of one resource from T_i to T_j reduces the loss.

Theorem 2

An allocation where there is an excess of defender resources for any type is optimal for the defender.

Theorem 2 can be exploited to define an optimal defender allocation any time the corresponding pool includes at least $OD = \sum_{k=1}^{n} pa_k \frac{Na_k}{pd_k}$ resources because in this case the defender can define an allocation where there is an excess of resources for any type. As proven in the following, OD is the lower amount of resources that guarantees a zero loss for the defender.

To optimise the defender allocation, at first we define $DBenefit(T_i)$, the **benefit of a defender resource for T_i** as the difference in the overall loss achieved by assign a further resource T_i:

$$DBenefit(T_i) = \frac{\alpha_i M_i}{pd_i}\left(\frac{1}{Nd_i + 1} - \frac{1}{Nd_i}\right)$$

An increase of the defender resources allocated to T_i reduces the overall loss of the defender, provided that there is not an excess of resources for T_i. The change in the defender utility due to the further

resource is equal to *-DBenefit(T_i)*, a monotone decreasing function of Nd_i. A further definition is that of *Dloss(T_i)*, the **loss of a defender resource for T_i**, the increase in the loss if a defender resource allocated to T_i is shifted to a distinct type. We have that

$$DLoss\ (T_i) = \frac{\alpha_i M_i}{pd_i}\left(\frac{1}{Nd_i - 1} - \frac{1}{Nd_i}\right)$$

If there is not an excess of resources for T_i, Dloss(T_i) is always positive because the reduction of the defender resources for T_i increases the loss. Since Dloss(T_i) is proportional to Nd_i, the following theorem holds.

Theorem 3

Both the benefit and the loss of a defender resource allocated to T_j decrease with Nd_j, the number of defender resources currently allocated to T_j .

We define *DShift(i ,j)* as the overall loss difference if the defender shifts one resource from T_i to T_j:

$$DShift(i, j) = -DBenefit(T_j) + DLoss(T_i)$$

We are interested in negative values of *DShift(i, j)* because they reduce the overall loss. Because of Theorem 1, a negative value is achieved in the following case:

there is not an excess of resources for both types before or after the shift

$$\frac{\alpha_i M_i}{pd_i}\frac{1}{(Nd_i - 1)Nd_i} > \frac{\alpha_j M_j}{pd_j}\frac{1}{(Nd_j + 1)Nd_j}$$

If no pair of types satisfies condition b), the allocation is optimal for the defender. Instead, if there is a pair of types that satisfies both conditions, then a resource shift improves the defender allocation. A particular case is the one where the shift from T_i eliminates an excess of defender resources and *Dloss(T_i)* is the contribution of T_i to the overall loss after the shift:

$$Dloss(T_i) = \alpha_i M_i\left(\frac{1}{pd_i(Nd_i - 1)} - \frac{1}{pa_i Na_i}\right)$$

Since, the shift of one resource removes the excess, we have that

$$pd_i\,(Nd_i\text{-}1) \le pa_i\,Na_i \le pd_i\,Nd_i$$

This implies that

$$\frac{\alpha_i}{pd_i}\frac{M_i}{(Nd_i - 1)Nd_i} > \frac{\alpha_j}{pd_j}\frac{M_j}{(Nd_j + 1)Nd_j}$$

is an upper bound on *Dloss(T_i)* and the previous condition still guarantees that the shift decreases the overall loss. To define, in the general case, the resource shifts that improve the defender allocation, consider that the following cases are possible:

1) $Dloss(T_i) > 0$, $DBenefit(T_j) < 0$,
2) $Dloss(T_i) = 0$, $DBenefit(T_j) < 0$,
3) $Dloss(T_i) > 0$, $DBenefit(T_j) = 0$,
4) $Dloss(T_i) = 0$, $DBenefit(T_j) = 0$.

Case 1) has been previously discussed. In case 2), the shift always increase the loss because it moves a resource from a type where there is not an excess to one where already there is an excess. In case 3), a resource is shifted from a type where there is an excess before and after the shift to one where there is not an excess. Since in case 4) $Dloss(T_i) = 0$ and $DBenefit(T_j)=0$, there is an excess of resources for both types before and after the shift. Hence, the shift does no influence the overall loss. Only case 1) can improve the defender allocation because a resource shift can reduce the loss provided that there is a pair of types satisfying condition b).

However, we have neglected a shift involving several resources and types simultaneously. To be able to neglect these shift, together with those that do not improve the utility but are useful an elementary step of a larger shift, we transform each defender allocation *DA* into Min(*DA*), the minimization of *DA*. Min(*Da*) is an allocation that minimizes the excess of resources so that, for each type T_k there is an excess of resources for T_k in *DA* there is also an excess in Min(*DA*) but the amount of resources allocated to T_k is the smallest one resulting in an excess. The saved resources are assigned, one at the time, to a type T_w such that the value of

$$\frac{\alpha_w M_w}{pd_w Nd_w (Nd_w + 1)}$$

is the largest one among all the types such that there is not an excess of resources for the type. Distinct resources may be assigned to distinct types. Only if there is an excess of resources for any type, the excess for some type can be increased. The minimization operator saves some resources from types where there is an excess to transfer them to types that contribute to the overall loss and to the attacker utility. After applying the minimization operator, we can consider only the shift of one resource.

In any case where the attacker allocation is known, the procedure to compute the optimal defender allocation may be outlined as follows:

1. if $\sum_{h=1}^{n} \frac{pa_h Na_h}{pd_h} \leq D$ then allocate to each type T_k the smallest amount of resources larger than $\frac{pa_k Na_k}{pd_k}$
2. otherwise
   a. apply the minimization operator to shift the defender resources from types with an excess of resources to those without an excess;
   b. shift one at the time a defender resource between types satisfying both conditions of Theorem 1. If there are not two types satisfying both conditions, then an optimal allocation for the defender has been computed.

The overall complexity is $O(D^3 \log D)$, because each resource may be sequentially assigned to, at most, all the types. Furthermore, if step b) of the procedure is applied, the complexity of each assignment is $O(D \log D)$ because types have to be ordered. Lastly, the number of types is bounded by D.

To prove the correctness of the procedure, we prove that distinct steps of the procedure cannot shift a resource back and forward between the same types. To this purpose, consider that a resource may be shifted from T_i to T_j and then to T_i only if condition b) is satisfied between T_i and T_j before the first shift and between T_j and T_i before the second one. This occurs only if

$$\frac{\alpha_i M_i}{pd_i (Nd_i - 1) Nd_i} > \frac{\alpha_j M_j}{pd_j (Nd_j + 1) Nd_j} > \frac{\alpha_i M_i}{pd_i (Nd_i - 1) Nd_i}$$

that is impossible.

We do not discuss in details the optimisation of the attacker allocation because the cases to be considered are similar to the previous ones as the only difference between players is the sign of the utility function, as *AD=-DU*. As an example, we can define the notion of excess of attacker resources as for the defender ones and introduce the shift of attacker resources.

4. Equilibrium

A pair of allocations defines a game equilibrium if no shift of resources can improve a player utility. This implies that both allocations are optimal for the players because they cannot increase their utility by shifting some resources.

To define conditions for equilibrium, consider an optimal allocation for the defender and assume the attacker shift all its resources to one type only. In general, the utility of each player changes. It is trivial to see that a case where the utility function does not change is the one where the defender has allocated to each type an amount of resources such that there is an excess of defender resources independently of the attacker allocation, i.e. both before and after the shift of the attacker, that is where

$$\frac{D_i}{A} \geq \frac{pa_i}{pd_i}$$

holds for any $i \in 1..n$. This is possible only if $D \geq A \sum_{i=1}^{n} \frac{pa_i}{pd_i}$.

If this condition is satisfied, the defender can allocate to each type a number of resources that guarantees that, on the average, it will find any vulnerability before than the attacker. In the following, this allocation is denoted as the **overflow allocation**. The overflow allocation is not cost effective for the defender because it assumes the worst case where the attacker focuses all its resources on just one

type. Hence, if the attacker distributes its resources across several types, some of the resources the defender allocates to a type are ineffective, i.e. they could be removed from the defender pool without increasing the overall loss.
Hence, any pair of allocations where the defender allocation is an overflow one always defines a game equilibrium. Any pair of allocations with an excess of defender resources for all the types but where the defender pool is too small for an overflow allocation is not a game equilibrium because the attacker may improve its utility by assigning all its resources to just one type. This always increases the attacker utility. In this case there is not equilibrium because one allocation is optimal for a player but the opponent on is not optimal.

Let us consider now an equilibrium that not corresponds to an overflow allocation. Since any update of one configuration only cannot improve the corresponding utility, any resource shift for the attacker or the defender does not change the corresponding utility. If we consider that the shift of some resources by either the attacker or by the defender should not result in a change of the corresponding utility function conditions for equilibrium may be deduced. As an example, a shift that involves one resource does not change the utility function if, for any pair of types T_i and T_j the following condition holds:

$$\frac{pa_i(Na_i-1)Na_i}{pd_i(Nd_i-1)Nd_i}=\frac{pa_j(Na_j+1)Na_j}{pd_j(Nd_j+1)Nd_j}$$

The condition can be generalized by taking into account the shift of several resources. However, from our point of view, the important property is that a player can exploit conditions such this one to play a move that results in equilibrium only if it knows the opponent allocation. Instead, an overflow allocation results in equilibrium independently of the opponent allocation and it can be played provided that the player knows the size of the opponent pool.

Taking into account that the notion of overflow allocation also applies to the attacker, and that a move of a player **forces equilibrium** if it defines a game **equilibrium independently of that of the other player**, the previous discussion may be resumed by the following theorem.

Theorem 4

A player is sure of having forced a game equilibrium if and only if it knows the size of the pool of its opponent and it can play an overflow allocation.

4.1. Examples

First of all we notice that in the fairly common case where, for any type, the attacker and the defender resources have the same probability of finding the vulnerability, an overflow allocation is possible if $D \geq nA$. This implies that the defender is sure of having avoid a loss only if the size of its pool is *n* times that of the attacker.

Let us consider now a simple case with two types T_1 and T_2 and where:

1. M_1= 10, M_2=15,
2. D=30, A=20
3. $\alpha_1=\alpha_2$
4. $pd_1=pd_2=1/10^5$
5. $pa_1=pa_2=5/10^6$

Consider a pair of allocations where $Na_1=Na_2=10$, $Nd_1=Nd_2=15$. In this case, the overall average loss for the defender is zero, because there is an excess for both types. However, a loss is possible because the size of the defender pool cannot force a game equilibrium. As an example, there is a loss for the defender if $Na_1=0$ and $Nd_1=25$, because $\frac{1}{pa_2Na_2}=10^{-4}$ while $\frac{1}{pd_2Nd_2}=5\cdot10^{-4}$.

If, instead D=A=20, $pd_1=pa_1$ and $pd_2=pa_2$, the optimal allocation for the defender is the one where it matches the resources of the attacker, i.e. where $Nd_1=Na_1$ and $Nd_2=Na_2$.

Consider now the case where

1. M_1= 10, M_2=15, M_3=20,
2. D = 45,
3. A = 30,
4. $\alpha_1=\alpha_2=\alpha_3$
5. $pd_1 = pa_1 = 1/10^5$,
6. $pd_2 = pa_2 = 5/10^6$,
7. $pd_3 = pa_3=1/10^4$.

Also in this case, the defender cannot force equilibrium because $D < A\sum_{i=1}^{3}\frac{pa_i}{pd_i}$. Consider a defender allocation where Nd_1=20, Nd_2=20 and Nd_3=5, while in the attacker one Na_1=Na_2=Na_3=10. In this instance of the game, there is an excess of resources for both T_1 and T_2 that do not contribute to the overall loss because there are at least six resources allocated to each type. Hence, to optimize the defender allocation, the minimization operation shifts resources from either T_1 or T_2 to T_3 till Nd_3>10.

5. Generalization and Preliminary Applications

This section generalizes the model as far as concerns the number of vulnerabilities of a type and alternative utility functions. Then, a first application of the results is illustrated.

5.1. Generalization

At first, we consider the presence of several vulnerabilities in a component, a case that can be handled in several ways. As an example, if we assume that all the vulnerabilities enable attacks with the same severity, i.e. attacks with similar impacts, then the existence of several vulnerabilities can be handled by properly defining the probability value of finding just one vulnerability. If, instead, the vulnerabilities have distinct severities we can model the search by considering the goal of the attackers, Some attackers will be satisfied even if they find a vulnerability that enable low impact attacks only. Instead, other attackers will neglect such vulnerabilities and focus their search for those that enable high impact attacks. In this case, several searches occur simultaneously. The proposed model can be exploited to model each search in isolation and the information it returns on the optimal allocation of resources for a single search can be used to allocate resources among the various searches. In this way, we define a two level hierarchical model. The highest level allocates resources to distinct pools, one for each severity class. The second level allocates the resources of a pool to the types of the information infrastructure components.

5.2. Alternative Utility Functions

As a first alternative utility function, assume that the defender is interested in minimizing the probability of a successful attack, i.e. the probability that the attacker resources find a vulnerability before the defenders one. To handle this case, at first we define $DFirst_i$ as the probability that a defender resource finds V_i before an attacker one. It may be proved [10] that

$$Dfirst_i = \frac{pd_i Nd_i}{pd_i Nd_i + pa_i Na_i}$$

We can now define the defender utility as $DU = \sum_{i=1}^{n} Dfirst_i$

Also in this case, the attacker utility is the inverse of the defender one, i.e. $AU = -DU$. $Dfirst_i$ is important because it allows us to define utility functions that take catastrophic attacks into account. In fact, the only defender strategy that can prevent such attacks is the one that finds any vulnerability of a component before than the attacker. The corresponding allocations can be evaluated by utility functions defined in terms of $Dfirst_i$. Even in the case of these utilities, we can define an optimisation strategy based upon resource shifts and excess of resources. Consider, as an example, the shift of a resource from T_i to T_j such that there is no excess before and after the shift. In this case,

$$DLoss(T_i) \cong \frac{pd_i}{pd_i Nd_i + pa_i Na_i} \qquad DBenefit(T_j) \cong -\frac{pd_j}{pd_j Nd_j + pa_j Na_j}$$

The update to the overall utility depends upon the sign of $-DBenefit(T_j) + Dloss(T_i)$.

A further definition of the defender utility may consider the probability that the defender finds all the vulnerabilities before than the attacker. Hence, the attacker utility is the probability that it finds at least one vulnerability before than the defender. The resulting game is not a zero sum one because

$DU = \prod_{i=1}^{n} DFirst_i$ while $AU=1-DU$.

In this case, a shift from T_i to T_j increases the defender utility if $\frac{Nd_i - 1}{Nd_j + 1} > \frac{Nd_i}{Nd_j}$

5.3. Some Preliminary Applications

A first important application of the previous results concerns the adoption of open code, cots, components in an information infrastructure. A component is **open code** if its code is available for peer review and analysis; otherwise the component is a closed code one. These notions may be applied also to hardware-firmware components because even in this case the component behaviour depends upon some code in an internal memory. Obviously, open source components are open code ones but the inverse is not true. When evaluating the benefit of adopting an open code component vs. that of a closed code one, the main difference to be considered is the large number of resources that search for a vulnerability to patch the component but that are not managed by the defender. In the most general case, if T_i is an open code type, there are two sets of people, Sa_i and Sp_i searching for the vulnerability. People in Sa_i are interested in selling the information on the vulnerability to those interested in attacking the infrastructure, instead the results of people in Sp_i is public and may be exploited by both the attacker and the defender. If Nao_i and Ndo_i are, respectively, the sizes of Sa_i and Sp_i, we can apply the results of the previous sections to compute the size of the vulnerability window by taking into account that the size of the attacker pool is Nao_i+Ndo_i while that of the defender one is Ndo_i. In fact, any information found by a defender resource is public and the attacker can use it as well. To compare the vulnerability window of open code against that of closed code, we assume that

1. Nac_i, the size of the attacker pool for a closed code component is a fraction of that of an open code one so that $Nac_i = \frac{Nao_i}{k}, k \geq 1$
2. an attacker and a defender resource find a vulnerability in an open code component with the same probability p. In a closed code component, the probability that a defender resource finds a vulnerability does not change, since it can access the same information. Instead, that of an attacker one is $\frac{p}{h}, h \geq 1$ due to the lower amount of available information.

Under these assumptions, the adoption of an open code component results in a smaller vulnerability window anytime

$$\left(\frac{1}{pNdo_i} - \frac{1}{p(Nao_i + Ndo_i)} \right) < \left(\frac{1}{pNdc_i} - \frac{1}{\frac{p}{h}\frac{Nao_i}{k}} \right)$$

If $w=h*k$, $r_{oc} = \frac{Nao_i}{Ndo_i}$, $r_{cs} = \frac{Nao_i}{Ndc_i}$, $r_{cs}>r_{oc}$, the previous condition may be rewritten as:

$$\frac{r_{os}^2}{r_{os}+1} < r_{cs} - w$$

In general, $Ndo_i>>Ndc_i$ because the number of people that search for vulnerability in an open code component is much larger than the one that can be afforded by the infrastructure owner. This condition may not be satisfied in the case of very specialized components only. The condition to determine when an open code component is more convenient can be applied anytime we can assume that an owner that adopts an open code will apply the patch produced by any defender resource as it were produced by its own resources. If, instead, the owner neglects the results of people in Sp_i because, as an example, it does not monitor the mailing lists or forum where these results are published, then the adoption of an open code component cannot be convenient.

Anytime an open code component is more convenient, the owner may reallocate some of its resources to the closed code ones. In this way, an overflow allocation for closed code components may be convenient. Hence, the owner should consider the following alternatives

- adopt closed code components only. This is obviously the best solution anytime the owner manages a pool so that an overflow allocation is feasible;
- if an overflow allocation is not possible, replace some closed code components through open code ones so that available resources can be partitioned among a lower number of types.

6. Conclusion and Future Developments

We have considered the allocation of resources to the search for vulnerabilities and presented some formal results on optimal allocations and game equilibrium. In particular, we have shown that the allocation that surely results in an equilibrium is not cost effective for the infrastructure owner. The results have also been applied to define the conditions where the adoption of open code components in information infrastructures results in a more secure infrastructure. The adoption of open code components allows the owner to focus its resources on closed code components only so that equilibrium may be achieved in a more cost effective way. Other applications are possible for specific infrastructures or infrastructure components.

Among the possible developments of this work, we have the more accurate evaluation of the allocations chosen by the attacker and the defender by taking into account the distribution of the impact values rather than the average values only. Also other models for the search should be considered. As an example, the probability that a resource finds a vulnerability may increase with time because of the experience it has accumulated. A further issue to be considered is the adoption of distinct utility functions for distinct components.

References

[1] A. Acquisti, Privacy and security of personal information. Economic incentives and technological solutions, *Workshop on Economics of Information Security*, University of California, Berkley, CA, USA, 2002

[2] J.Akins, An Insurance Style Model for Determining the Appropriate Investment Level against Maximum Loss arising from an Information Security Breach, *Workshop on Economics of Information Security*, University of Minnesota, Twin Cities, USA, 2004

[3] J.Alberts, J. A Dorofee, A.J, An introduction to the OCTAVE Method. http://www.cert.org/octave/methodintro.html , 2000

[4] R.J.Anderson, Why Information Security is Hard-An Economic Perspective, *17th Applied Computer Security Applications Conference*, New Orleans, Louisiana, USA, 2001.

[5] R.J.Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons, Inc, ISBN: 0-471-38922-6, 2001.

[6] R.J.Anderson, Security in Open versus Closed Systems - The Dance of Boltzmann, Coase and Moore, *Conf. on Open Source Software Economics*, Toulouse (France), 2002-

[7] S. Anton, R.H. Anderson, R. Mesic, M. Scheiern, Finding and fixing vulnerabilities in information systems, MR-1601, Rand Corporation, 2003.

[8] A. Arbaugh, W.L. Fithen, J. McHugh, Windows of Vulnerability: A Case Study Analysis, *IEEE Computer*, vol.12, p. 52-59, 2002.

[9] R. Arora, R. Telang, H. Xu, Optimal Policy for Software Vulnerability Disclosure, *Workshop on Economics of Information Security*, University of Minnesota, Twin Cities, USA, 2004

[10] F.Baiardi, C. Telmon, Theoretical Model for the Average Impact of Attacks Against Billing Infrastructures. *Mathematical Methods and Model for Advance Computer Network Security*, S.Petersburg. LNCS, Vol. 3685, ISBN: 3-540-29113-X, 2005

[11] S.Beattie, S. Arnold. et al. Timing the Application of Security Patches for Optimal Uptime. *16th USENIX Sys. Administration Conf.* (LISA 2002), Berkley, CA, USA,2002.

[12] D.A. Burke, Towards a game theory model of information warfare, Master Thesis, Air Force Institute of Technology. USA,1999.

[13] B.Carini, Dynamics and Equilibria of Information Security Investments, Workshop on Economics of Information Security, University of California, Berkley, CA, USA, 2002

[14] B.S. Frey, S. Luechinger., A. Stulzer, Calculating Tragedy: Assessing the Cost of Terrorism, Inst. for Empirical Research in Economics, University of Zurich, 2004

[15] L.A. Gordon, M.P. Loeb, The Economics of Information Security Investment, *ACM Trans. on Information and System Security*, Vol. 5. No.4, pp. 438-457, 2002

[16] S.N. Hamilton, W.L. Miller, A. Ott, O.S. Saydjari, The Role of Game Theory in Information Warfare. *4th Information Survivability Workshop*, Vancouver, B.C., Canada, 2002.

[17] K.S. Hoo, How Much Is Enough? A Risk Management Approach to Computer Security, Ph.D. Thesis, Standford University, Standford CA, USA, 2002.

[18] K. Kannan, R. Telang, An Economic Analysis of Market for Software Vulnerabilities, *Workshop on Economics of Information Security*, University of Minnesota, Twin Cities, USA, 2004

[19] I.V. Krsul , Software Vulnerability Analysis, Ph.D. Thesis , Purdue University Purdue West Lafayette, IN,USA,1998

[20] J.A. Major, Advanced Techniques for Modeling Terrorism Risk, *Journal of Risk and Finance*, Vol. 4, No. 1, pp. 15-24, 2002

[21] G.Owen, *Game Theory*, Academic Press, 1995, Third Edition, ISBN 0-12-531151-6.

[22] E. Rescorla Is Finding Security Holes a Good Idea?, *Workshop on Economics of Information Security*, University of Minnesota, Twin Cities, USA, 2004

[23] S.E. Schechter. Computer Security Strength & Risk: A Quantitative Approach, Ph.D. thesis, Harvard University, Boston USA, 2004

[24] B. Schneier, B Full disclosure and the window of vulnerability, Crypto-Gram http://www.counterpane.com/crypto-gram-0009.html 2000

[25] B. Schneier, Closing the Window of Exposure: Reflections on the Future of Security, Securityfocus.com,http://www.securityfocus.com., 2000

[26] G. Schudel, .B Wood, Adversary work factor as a metric for information assurance, *Workshop on New Security Paradigms*, Ballycotton, County Cork, Ireland, 2000