Using CP-nets as a guide for countermeasure selection*

Stefano Bistarelli†, Fabio Fioravanti, Pamela Peretti
Dipartimento di Scienze
Università "G.d'Annunzio", Pescara, Italy
{bista, fioravanti, peretti}@sci.unich.it

ABSTRACT

In this paper we present a qualitative approach for the selection of security countermeasures able to protect an IT system from attacks. For this purpose, we model security scenarios by using defense trees (an extension of attack trees) and preferences over countermeasure using Conditional Preference networks (CP-nets for short). In particular, we introduce two different methods for the composition of preferences: the `and`-composition and the `or`-composition. The first one is used to determine a preference order in the selection of countermeasures able to mitigate the risks produced by conjunct attacks. The second one is used to determine a preference order over sets of countermeasures able to mitigate the risks produced by alternative attacks.

1. INTRODUCTION

In order to focus the real and concrete threat that could affect an enterprise's assets, a risk management process is needed in order to identify, describe and analyze the possible vulnerabilities that must be eliminated or reduced. The final goal of the process is to make security managers aware of the possible risks, and to guide them towards the adoption of a set of countermeasures which can bring the overall risk under an acceptable level.

The use of *quantitative* evaluations in the analysis of an IT system is often very difficult because many factors can influence the attacks and the selection of countermeasures. In real life a system administrator bases his choice on experience and knowledge.

An instrument that can be used to determine the possible attacks that can harm a system, and the necessary countermeasures are *defense trees*. Defense trees [1] allow us to analyze the possible attack/defense scenarios: they represent the vulnerabilities of the system as leaf nodes of an and/or tree. Each leaf node is then decorated with the countermeasures which are able to mitigate the damage of threats using the specific vulnerability corresponding to that node.

Defense trees allow us to represent a set of possible countermeasures to defend the system, but we also need to select the best countermeasures among the possible ones. In this paper we propose for this purpose the use of CP-nets.

In this paper we use CP-nets [2] (a formalism to represent and reason with qualitative and conditional preference) to model preference over attacks and countermeasures. In particular, we introduce two different operations for the composition of preferences with the purpose of determining preference order over *sets* of countermeasures.

The paper is structured as follows: we introduce defense trees and CP-nets in Sec. 2 and 3 respectively. The innovative part is in Sec. 4 where we propose two different methods to compose preferences: the `and`-composition and the `or`-composition. In Sec. 5 we present an application of this method for the selection of countermeasures. In Sec. 6 we present our conclusions and some proposals for future work.

2. DEFENSE TREE

Defense trees [1] are an extension of attack trees [11] and represent an attack against a system and how it can be mitigated by a set of countermeasures.

The idea of integrating countermeasures into threat trees (another name for attack trees), and more generally into directed acyclic graphs, is not new. Caelli *et al.* [3] in the 90's integrate safeguards by representing them as nodes, placed throughout the diagram. Even in the popular Microsoft text by Howard and LeBlanc, "Writing Secure Code", one can find threat trees in which countermeasures are integrated [6]. One may also see examples of countermeasures in direct acyclic graphs in both Foster's thesis and Schechter's thesis [4, 10], both of which include discussions and histories of the evolution of these structures.

The Srivatanakul's thesis [12] introduces fault tree analysis [13] which consists in a deductive top-down method used to determine the fault that could result in the occurrence of a specific undesired event. It is a good method for a detailed analysis of a single event, but it requires a detailed knowledge of the system. There is also a tool [5] that use fault trees for the risk analysis.

Event tree analysis [9], instead, illustrates the possible sequences of outcome of an event, considering success and/or failure of the system components that respond to it. They are useful to identify different scenarios that can be produced from a single event.

The attack trees analysis method was invented by Bruce Schneier [11] in the late 1990's to facilitate the process of security threat modeling. The basic principle of the approach is that, if we know the possible attacks to an asset, the selec-

*This paper is supported by the MIUR PRIN 2005-015491.
†Partially supported by: Istituto di Informatica e Telematica, C.N.R., Pisa, Italy (`stefano.bistarelli@iit.cnr.it`).

SAC '07 March 11–15, 2007, Seoul, Korea

tion of the countermeasures is easier and more applicable. One important characteristic of attack trees is the ability to reuse the tree. Moore *et al.* [7] has organized attack patterns into attack profiles.

The main difference between an attack tree and a defense tree is that the first one represents only the action that an attacker can perform, while the second one adds the set of countermeasures that can be introduced into the system to mitigate the possible damages produced by an attack action.

The root of the tree is associated with an asset of the IT system under consideration and represents the attacker's goal. Leaf nodes in the attack tree represent simple subgoals which lead the attacker to (partially) damage the asset by exploiting a single vulnerability. Non-leaf nodes (including the tree root) can be of two different types: `or`-nodes and `and`-nodes. Subgoals associated with `or`-nodes are completed as soon as any of its child nodes is achieved, while `and`-nodes represent subgoals which require all of its child nodes to be completed (in the following we draw an horizontal line between the arcs connecting an `and`-node to its children to distinguish it from an `or`-node). The standard attack tree is then enriched by decorating every leaf node with a set of countermeasures. Each countermeasure associated with a leaf node represents a possible way of mitigating risk in an attack scenario where that specific vulnerability is used.

Notice that in order to mitigate the risks deriving from an `or`-attack, the system administrator has to introduce into the system a countermeasure for each possible action of the branch. To mitigate the risks associated with an `and`-attack, instead, it is enough introducing a countermeasure for an attack action in the `and`-attack is sufficient to stop the entire attack.

In the following example we use a defense tree to model an attack/defense scenario for an organization's IT system.

EXAMPLE 1. *An enterprise's server used to store information about customers. To steal these data an attacker can follow different attack strategies: To steal this data an attacker needs to perform, while square box denote the different countermeasures the system administrator can adopt.* □

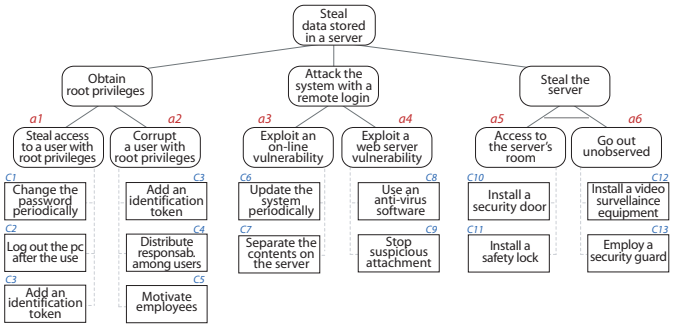


Figure 1: An example of defense tree.

tacker can follow different attack strategies: obtain the privileges on the server, attack the system with a remote login or steal the server itself.
For each attack strategy the system administrator can use different countermeasures like for example: introduce security policies, update the system periodically, use an antivirus software, control the email traffic or he can protect the server's room using a security door or a safety look, install a video surveillance equipment or employ a security guard.
Fig. 1 shows the scenario described above: rounded-box nodes denote the attack strategies and the different actions the at-

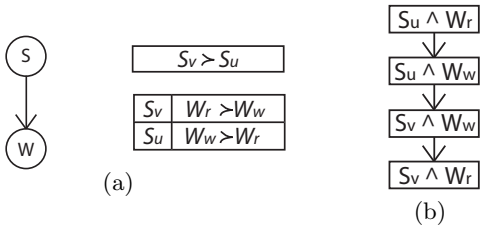


Figure 2: An example of CP-net and the corresponding induced preference graph.

3. CP-NETWORKS

Conditional preferences networks (CP-net for short) [2] are a graphical formalism for specifying and representing qualitative conditional preference relations. CP-nets capture *ceteris paribus* preference statements like, for example: *"I prefer red wine to white wine if vegetable soup is served"*. This preference states that, given two meals both containing a vegetable soup, the meal with red wine is preferred to the meal with white wine, *all else being equal*. By using only a sentence we are able to express a conditional preference over two variables, the soup, S, and the wine, W. This preference can be represented more formally using a CP-net. During preference elicitation, for each variable x in the variable set V, a user specifies the parent variable $Pa(x)$ that can affect his preferences over the values of x. This information is used to create the CP-net graph in which each node x has $Pa(x)$ as its immediate predecessor. So, given a particular value assignment to $Pa(x)$, the user can determine a preference order over the domain of x (denoted as $\mathcal{D}(x)$), all other things being equal. This conditional preference over the values of X is captured by a *conditional preference table* (CPT). For each assignment to $Pa(X)$, $CPT(X)$ specifies a strict partial order (denoted $\succ$) over $\mathcal{D}(X)$[1].

More formally, a CP-net can be defined as follows.

DEFINITION 1. **(CP-net)** *A CP-net is a directed graph $\mathcal{N} = (V, E)$, where $V = \{x_1, \ldots, x_n\}$ is a set of variables and $E = \{(x_i, x_j) : x_i, x_j \in V\}$ is a set of edges between variables. The function $Pa(x)$ gives for each node $x \in V$, the node $x' \in V$ s.t. $(x', x) \in E$. The conditional preference table of the CP-net describes a strict partial order $(\mathcal{D}(x_i), \succ_u^i)$ where $\mathcal{D}(x_i)$ is the domain of the variable x_i, and $\succ_u^i$ represents the conditional preference of the instantiations of variable x_i given an instantiation u of the variable $Pa(x_i)$.* △

So using this definition we can represent the previous sentence "I prefer red wine to white wine if vegetable soup is served" using a CP-net: the variables are the soup and the wine $V = \{S, W\}$, and S is a parent of W because the type of soup influences the selection of wine, $Pa(W) = S$. The $CPT(W)$ says that if a vegetable soup is served (S_v), the red wine is preferred to the white wine ($W_r \succ W_w$). However, if another type of soup is served the white wine is preferred

[1]Notice that the right side of a conditional preference table is composed by a set of admissible values and by a partial order over these values. When the set is not specified, we assume that it is composed only by the values used in the partial order.

to the red wine ($W_w \succ W_r$). Fig. 2(a) shows the resulting CP-net.

The preference relations are then used to build an *induced preference graph*, that is an acyclic directed graph where the nodes corresponds to the complete assignments of the variables of the network, and there is an edge from node o' to node o if and only if the assignments at o' and o differ only in the value of a single variable X and o is preferred to o'. Fig. 2(b) is the induced preference graph corresponding to the CP-net of Fig. 2(a). It shows that the assignment $S_u \wedge W_r$, the root of the induced preference graph, is the less preferred while $S_v \wedge W_r$, the leaf of the graph, is the most preferred assignment.

The preference information captured by an acyclic CP-net can be viewed as a set of logical assertions about a user's preference ordering over complete assignments to variables in the network.

4. CP-NET AND DEFENSE TREE: COMPOSITION OF PREFERENCES

We can combine the use of defense trees and CP-nets to analyze attacks to the assets that compose an IT system. The first one allows us to determine, in a qualitative manner, the attack strategies that an attacker can follow to damage a system, the different actions that compose each attack and the security measures that a system administrator can introduce into the system. The second one can be used to determine the system administrator's preference order in the selection of sets of security mechanisms.

For each attack action a the system administrator can specify a preference order in the adoption of n countermeasures ($a : \{c_i \succ c_j \text{ s.t } i,j \in [1,\dots,n]\}$) and collect this preference relation in a CPT.

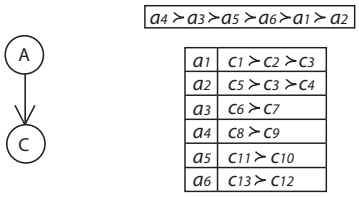


Figure 3: An example of CP-net

Fig. 3 shows a possible CP-net and CPTs corresponding to the defense tree of Fig. 1: the preference over countermeasures is conditioned by the type of action, so $Pa(C) = A$, and $CPT(A)$ describes the preferences over actions $a_i \in A$. For example, attack actions can be ordered according to the impact produced on the system, so we can say that the action a_4 is preferable to the action a_3 ($a_4 \succ a_3$) because it is less dangerous. $CPT(C)$, instead, describes the conditional preference relations over the countermeasures $c_i \in C$. For example we can say that for the action a_1, c_1 is preferable to c_2 because it is less expensive than c_1.
The induced preference graph associated with this CP-net can be integrated in a defense tree, as shown in Fig. 4. We represent the preference order over actions described in $CPT(A)$ with dotted arrows, while conditional preferences over countermeasures described in $CPT(C)$ are represented by using solid arrows[2].

[2]Remember that the arrows of the induced preference graph are directed from the less preferred to the more preferred outcome.

At this point the system administrator doesn't know which particular attack will affect his system, so he has to determine the best defense strategy that can protect the system from all the possible attack strategies. So using the preference table associated with each action, he has to combine the preference orders and obtain a new preference order over *sets* of countermeasures.

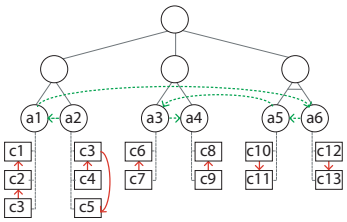


Figure 4: CP-nets on a defense tree.

4.1 and-composition

As already mentioned in Sec. 2, an **and**-attack is an attack composed by a set of actions that an attacker has to successfully achieve to obtain his goal. To protect the system from this type of attack the system administrator can select a countermeasure for any of the actions composing the **and**-attack. In fact, it is enough to stop one of the actions to stop the attack[3].

DEFINITION 2. *We say that a set of countermeasures C covers an **and**-attack A if there exists a countermeasure $c \in C$ covering at least an attack action $a \in A$.* △

So given an **and**-attack composed by two actions x and y, we have to consider the CP-nets associated with both actions and combine the order of preferences for the countermeasures associated actions x and y thereby obtaining a new partial order considering the actions $x \wedge y$.
The **and**-composition of preferences can be defined more formally as follows:

DEFINITION 3. **(Preference table and-composition)** *Consider a CP-net $\mathcal{N} = (V, E)$, where $V = \{x_1, ..., x_n\}$ is a set of variables, $\mathcal{D}(x_i)$ is the domain of variable x_i and the conditional preference table is described by the partial orders $(\mathcal{D}(x_i), \succ^i_u)$ and $(\mathcal{D}(x_i), \succ^i_v)$, given two instantiations u and v of the variable $x_j = Pa(x_i)$.*

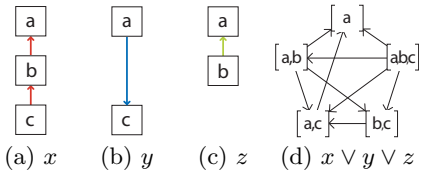
*The **and**-composition of the preference tables described by the partial orders $(\mathcal{D}(x_i), \succ^i_u)$ and $(\mathcal{D}(x_i), \succ^i_v)$, is described by the partial order $(\mathcal{D}(x_i), \succ^i_{u\wedge v})$ where $\succ^i_{u\wedge v}$ represents the conditional preference of the instantiations of variable x_i given an instantiation $u \wedge v$. So given $a, b \in dom(x_i)$ and $x_j = Pa(x_i)$:*

$$a \succ^i_{u\wedge v} b \iff \begin{cases} a \succ^i_u b \text{ or } a \succ^i_v b \text{ or} \\ \forall x, y \in \mathcal{D}(x_i),\ x \not\succ^i_u a \text{ and } b \not\succ^i_v y \\ \quad \text{and } v \succ^j u \end{cases}$$

△

The countermeasure a is preferred to the countermeasure b if it is preferred in, at least, one of the partial orders $\succ^i_u$, $\succ^i_v$. Otherwise, we consider the preferences over the values of the parent variable $x_j = Pa(x_i)$: if $v \succ^j u$ then the countermeasures able to mitigate the risk of u are preferable to the

[3]We suppose that the system administrator is able to select correctly the countermeasure able to stop the **and**-attack.



(a) x (b) y (c) z (d) $x \vee y \vee z$

Figure 5: An example of `or`-composition.

countermeasures able to mitigate the risk of v. For example, given three attack actions x, y and z and the corresponding CPT $x : a \succ b \succ c$, $y : b \succ c$ and $z : a \succ b$ if we consider the action $x \wedge y \wedge z$ we obtain the preference order $x \wedge y \wedge z : a \succ b \succ c$. In this case the countermeasure a is the best choice for this `and`-attack. As another example, given two attack actions x and y, the corresponding CPT $x : a \succ b$, $y : c \succ d$ and the preference relation $x \succ y$, we obtain the preference order $x \wedge y : c \succ d \succ a \succ b$ so the countermeasure c is the best choice for this `and`-attack.

4.2 `or`-composition

An `or`-attack is an attack that can be performed with different and alternative actions: the attacker can complete successfully any of its actions to obtain his goal. To protect the system from this type of attack, the system administrator has to select one countermeasure for each of the actions composing the `or`-attack.

Definition 4. *We say that a set of countermeasures C covers an `or`-attack A if for each action $a \in A$ there exists a countermeasure $c \in C$ covering such attack.* △

Example 2. *For example, let us consider a set of attacks actions $A = \{x, y, z\}$, a set of countermeasures $C = \{a, b, c\}$ with the following conditional preference order for the countermeasure: $x : a \succ b \succ c$, $y : c \succ a$ and $z : a \succ b$. The set $\langle a \rangle$ covers the set A because it is able to mitigate the risk associated with attack actions x, y and z. The set $\langle b \rangle$, on the contrary, doesn't cover A because it is not able to protect the system from the attack action y.* □

In summary, given an `or`-attack composed by two actions x and y, we have to consider the CP-net associated with both the attack actions, create all the possible couples of countermeasures and then determine a new partial order of preference, considering the preference associated with each action that compose the attack.

Definition 5. **(Preference table `or`-composition)** *Consider a CP-net $\mathcal{N} = (V, E)$ and the conditional preference tables for the variable $x \in V$, described by the partial order $(\mathcal{D}(x), \succ_{u_1}), \ldots, (\mathcal{D}(x) \succ_{u_k})$ for the instantiations $u_1, \ldots, u_k$ of the variable $Pa(x)$.*
Given two sets of countermeasure $C = \{c_1, \ldots, c_k\}$ and $C' = \{c'_1, \ldots, c'_{k'}\}$ covering the attacks $u_1, \ldots, u_k$, the `or`-composition conditional preference table $(\mathcal{D}(x), \succ_{u_1 \vee \cdots \vee u_k})$ is defined as follows:

$$C \succ_{u_1 \vee \cdots \vee u_k} C' \iff \begin{cases} C \subset C' \text{ or} \\ k = k' \text{ and } \exists \text{ a permutation } \pi \\ \quad s.t.\ \forall i \in [1, \ldots, k],\ c_i \not\prec c'_{\pi(i)} \end{cases}$$

△

In an `or`-composition of preferences we have to consider tuples of countermeasures.

In particular: if two different tuples of countermeasures, C and C', are both able to mitigate the risk associated with the same same attack, the tuple C is preferred to the tuple C' because it uses less countermeasures. If, instead, the number of countermeasures that compose the two tuples is the same ($k = k'$), we have to compare the tuples element by element (considering all the possible permutations).

The following example shows how the `or`-composition works.

Example 3. *We consider a set of attack actions $A = \{x, y, z\}$, a set of countermeasures $C = \{a, b, c\}$ and the following conditional preference order for the countermeasures: $x : a \succ b \succ c$, $y : c \succ a$ and $z : a \succ b$ (respectively Fig. 5(a), 5(b) and 5(c)). We want to determine the preference order over sets of countermeasures considering the attack $x \vee y \vee z$. First of all we have to determine the possible tuples of countermeasures that cover this `or`-attack and group it into equivalence classes.*

$$\begin{aligned} [a] &= \langle a, a, a \rangle \\ [a, b] &= \langle b, a, a \rangle, \langle a, a, b \rangle, \langle b, a, b \rangle \\ [a, c] &= \langle a, c, a \rangle, \langle c, c, a \rangle, \langle c, a, a \rangle \\ [b, c] &= \langle b, c, b \rangle, \langle c, c, b \rangle \\ [a, b, c] &= \langle b, c, a \rangle, \langle a, c, b \rangle, \langle c, a, b \rangle \end{aligned}$$

Notice that the elements that compose each equivalence class are only the sets of countermeasures that cover an attack. For instance, the equivalence class $[a, b]$ is composed by the sets $\langle b, a, a \rangle$, $\langle a, a, b \rangle$ and $\langle b, a, b \rangle$: the set $\langle b, b, a \rangle \notin [a, b]$ cause the countermeasure b in the second position is not able to mitigate the risks produced by the second action y and, for this reason, is not in the domain of y.
Now we have to compare these sets: for instance the set $\langle b, c, b \rangle$ is preferable to the set $\langle b, a, a \rangle$ because $b \not\succ_x b$, $a \not\succ_y c$ and $a \not\succ_z b$ so we can draw in the graph an edge from the class $[a, b]$ to the class $[b, c]$. Fig. 5(d) shows the resulting induced preference graph, where the set $[a]$ is the preferable solution for the attack $x \vee y \vee z$. □

5. AN EXAMPLE

In this section we present an application of CP-nets and the operation of `and`/`or` composition for analyzing the attack/defense scenario of an enterprise's server used to store information about customers as we described in Example 1.

Fig. 3 shows the CP-nets and the conditional preference for this scenario. We want to determine the set of countermeasures to protect the system from all the actions that compose the scenario: $a_1 \vee a_2 \vee a_3 \vee a_4 \vee (a_5 \wedge a_6)$.

In this example, to make the presentation easier, we omit to compose the preferences associated in all five branches. Notice that in this specific case the only shared countermeasure is c_3 (between actions a_1 and a_2). For the other attack actions a_3, a_4, a_5 and a_6 there is no shared countermeasure so it suffices to just take the best countermeasure associated with each attack.

The actions a_1 and a_2 can be covered by the countermeasure $\langle c_3 \rangle$ so it is interesting to analyze in detail how to combine the preference associated with both attacks. Using the definition of `or`-composition we determine the set of countermeasures and the corresponding preference order for the attack $a_1 \vee a_2$. We obtain the induced preference graph of Fig. 6 where we can see that the preferable sets are: $\langle c_1, c_5 \rangle$ and $\langle c_3 \rangle$. The first one is preferable because it is composed by the best countermeasure for each attack, the second one

is preferable because by using just one countermeasure we cover both the attacks

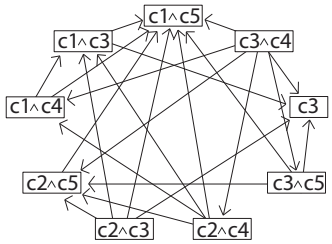


Figure 6: The induced preference graph associated with the `or`-attack $a_1 \vee a_2$.

The attack actions a_3 and a_4 are covered respectively by the countermeasures $\langle c_6 \rangle$, $\langle c_7 \rangle$ and $\langle c_8 \rangle$, $\langle c_9 \rangle$: each one is able to mitigate only one attack, so we can simply select the best countermeasure for each action: $a_3 : \langle c_6 \rangle$, $a_4 : \langle c_8 \rangle$.

The attack actions a_5 and a_6 constitute an `and`-attack so we have to compose the preference over both the actions using Def. 3. Remember that sometimes to compose the conditional preferences over countermeasures we have also to take into account the preference order between attacks. In this case we can see in the CPT of Fig. 3 that $a_5 \succ a_6$. Using this information in the `and`-composition we obtain the following preference order (where the countermeasures associated with the most dangerous attacks are preferable to the others): $a_5 \wedge a_6 : c_{13} \succ c_{12} \succ c_{11} \succ c_{10}$. The preferable solution for this attack is the countermeasure $\langle c_{13} \rangle$.

Now, considering all the attacks together we can say that the system administrator can select two different defense strategies: the first one is composed by the countermeasures $\{\langle c_1, c_5 \rangle, \langle c_6 \rangle, \langle c_8 \rangle, \langle c_{13} \rangle\}$ while the second one is composed by the countermeasures $\{\langle c_3 \rangle, \langle c_6 \rangle, \langle c_8 \rangle, \langle c_{13} \rangle\}$.

6. CONCLUSION AND FUTURE WORKS

In this paper we propose the use of two qualitative instruments for the selection of defense strategies to protect an IT system from the risk of attacks: we use defense trees to model attack/defense scenarios and CP-nets to model qualitative conditional preference over attacks and countermeasures.

Defense trees allow us to represent the possible attack actions that a person can perform to damage an asset of an IT system, and the possible countermeasures able to stop each action.

CP-nets, instead, allow us to model conditional preferences over attacks and countermeasures, and the qualitative preferences of a system administrator in the selection of countermeasures for each attack.

Our idea is to use CP-net to model the selection of the countermeasures represented in a defence tree. We propose two methods for the composition of these preferences: an `and`-composition to model the preference order in the case of an `and`-attack, and an `or`-composition in the case of an `or`-attack.

The first operation we propose is an `and`-composition, we combine the preference order over the countermeasures represented in the CP-nets associated to attack actions composing the `and`-attack. In this case we also take into consideration the preference between attacks with the purpose of determining the preferred countermeasures.

The second operation we propose is an `or`-composition of the CP-nets: we are interested in determining a preference order over sets of countermeasures which are able to cover all actions of an `or`-attack. In particular, we prefer countermeasures which are able to mitigate the risks associated with more than one action.

The methodology presented in this paper provides a basis for future work along the following directions.

We faced the problem of determining defense strategies composed by sets of countermeasures (one for each attack) and only in the case of an `or`-attack we studied how a single countermeasure can cover more than one attack. A possible extension of this work is to investigate how to determine sets of countermeasures able to mitigate sets of attacks.

In this paper we use CP-nets considering only *strict* partial orders, a possible extension can be the use of *non strict partial orders* to model the preference over countermeasures and attacks, and the use of indifference between this variables (for instance $a_1 \succeq a_2$).

We also plan to consider attacks as uncertain variables. In [8] an approach is described to model a real-life problem as a set of variables with finite domains and a set of soft constraints among subsets of the variables. A variable is uncertain if we cannot decide its value. In this case, they associated a possibility degree to each value in its domain, which will tell how plausible it is that the variable will get that value.

7. REFERENCES

[1] S. Bistarelli, F. Fioravanti, and P. Peretti. Defense tree for economic evaluations of security investment. In *ARES06*, pages 416–423, 2006.

[2] C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole. Reasoning with conditional ceteris paribus preference statements. In *UAI-99*, pages 71–80, 1999.

[3] W. J. Caelli, D. Longley, and A. B. Tickle. A methodology for describing information and physical security architectures. In *IFIP/Sec'92*, pages 277–296. North-Holland, 1992.

[4] N. L. Foster. *The application of software and safety engineering techniques to security protocol development*. PhD thesis, Univ. of York, Dep. of Computer Science, 2002.

[5] J. Gordon. Vurisk. http://www.conceptlabs.co.uk/.

[6] M. Howard and D. C. LeBlanc. *Writing Secure Code*. Microsoft Press, 2002.

[7] A. Moore, R. Ellison, and R. Linger. Attack modeling for information security and survivability. Technical report, Soft. Eng. Inst. CMU/SEI-2001-TN-001, 2001.

[8] M. S. Pini, F. Rossi, and K. B. Venable. Possibility theory for reasoning about uncertain soft constraints. In *ECSQARU*, pages 800–811, 2005.

[9] N. C. Rasmussen. Reactor safety study: An assessment of accident risks in us commercial nuclear power plants. Technical report, Nuclear Regulatory commission, 1975.

[10] S. E. Schechter. *Computer Security Strength & Risk: A Quantitative Approach*. PhD thesis, Harvard University, 2004.

[11] B. Schneier. Attack trees: Modeling security threats. Dr. Dobb's Journal, 1999.

[12] T. Srivatanakul. *Security Analysis with deviational techniques*. PhD thesis, Univ. of Cork, Dep. of Computer Science, 2005.

[13] H. A. Watson. Launch control safety study. Technical report, Bell Telephone Laboratories, 1961.