Cyber Security Risks Assessment with Bayesian Defense Graphs and Architectural Models

Teodor Sommestad
Royal Institute of Technology
teodors@ics.kth.se

Mathias Ekstedt
Royal Institute of Technology
mek101@ics.kth.se

Pontus Johnson
Royal Institute of Technology
pj101@ics.kth.se

Abstract

To facilitate rational decision making regarding cyber security investments, decision makers need to be able to assess expected losses before and after potential investments. This paper presents a model based assessment framework for analyzing the cyber security provided by different architectural scenarios. The framework uses the Bayesian statistics based Extended Influence Diagrams to express attack graphs and related countermeasures. In this paper it is demonstrated how this structure can be captured in an abstract model to support analysis based on architectural models. The approach allows calculating the probability that attacks will succeed and the expected loss of these given the instantiated architectural scenario. Moreover, the framework can handle the uncertainties that are accompanied to the analyses. In architectural analysis there are uncertainties acquainted both to the scenario and its properties, as well as to the analysis framework that stipulates how security countermeasures contribute to cyber security.

1. Introduction

The capability to assess the current cyber security posture as well as the cyber security posture after potential security investments is vital for efficient security management. For decision makers that deal with security investments, knowledge of the expected consequence of attacks prior and after a cyber investment is something that enables rational decision making [21].

When assessing these numbers the decision maker is faced with a great deal of uncertainty. Firstly, there is an uncertainty in how the cyber security mechanisms influence each other and how, or if, they contribute to security of a system as a whole. As stated in [24], there is today no algebra on perimeter security. Secondly, there is an issue regarding the accuracy of information on which the assessment is based upon. As soon as enterprises become moderate in size, the number of security mechanisms and issues becomes very large, highly diverse, and interconnected in complex manners. Information and indicators collected for security assessments are thus inevitably never fully credible, nor complete [25]. Still however, the cyber security decision maker needs to make choices that are rational from a holistic point of view.

This paper describes a framework for assessing the security of information systems while taking both these types of uncertainty into consideration. The framework bases its analyses on system architecture models. Software, system, and enterprise architecture is commonly proposed as tool for managing system complexities on a high level, holistic, level of abstraction. Unfortunately many architecture languages and frameworks fail to explain how different kinds of analysis should be performed on an architecture model. This paper describes an information system analysis framework that is well equipped for dealing with uncertainty can be merged with architecture metamodels by using a concept we call Abstract models.

The structure of this paper is as follows. In section two and three related work is described. This includes a description and an example of how extended influence diagrams can be used to model defense graphs. Section four describes abstract models and how these can be used to capture the type of defense graph that was presented in section three. Section five presents an example of how this abstract model can be instantiated into a concrete model and how it thereby facilitates security assessment. Finally, in section six and seven, some topics are discussed and conclusions are drawn.

2. Attack graphs and defense graphs

A great number of metrics have been proposed to capture the security of information technology. Typically those metrics start analyzing the information system from an external perspective by considering what kinds of attacks that would harm the system. An example is the attack surface metric [29] that measures how attackable various resources of a system are to enable relative comparison with similar systems. Other examples are the weakest adversary metric [27] and the mean time-to-compromise metric [28]. These metrics capture the difficulty or effort an adversary faces when attacking a system. To some extent this requires specification of the paths an adversary can use to compromise a system. Attack trees [15][16][17] (sometimes called threat trees) is an approach commonly used to model these attack paths.

In an attack tree, the attacker's main goal is depicted as the root of the tree and the steps to reach this goal are broken down into sub-goals of the attack through "AND" and "OR" relationships. Usually, several different attacker goals are of relevance and create a forest of attack trees.

Tree and graph structures of attack paths have been applied in several ways to assess security of systems and to assess system vulnerabilities and risks. Both [17] and [13] has proposed the use of attack trees during system development to analyze the security. In addition to this, plenty of analysis techniques based on graphs over attacks has been suggested, for example [18] [19][20] and [30].

While it is typically difficult to directly control what actions an attacker will chose and how frequent their attempts are, decision makers can to great extent control the difficulty to perform undesired actions by imposing countermeasures. Hence, a natural extension of attack graphs is to include these controllable countermeasures in the graph. In [13] countermeasures are modeled together with trees depicting threats, and in the theses by Foster [17] and Schechter [15] countermeasures are included in the tree structures. The concept of including countermeasures in the tree structure has also been used in [2], to create something called "Defense trees" (cf. Figure 1).

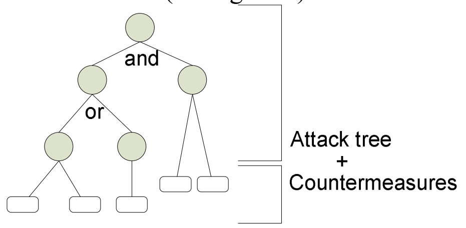


Figure 1 – The defense tree concept, from [26].

Techniques has been presented which use defense trees for strategic evaluation of security investments [2], modeling strategic games in security [26], as well as modeling of conditional preference of defense techniques using conditional preference nets [3].

The statistical mathematics apparatus of Bayesian networks is well suited for combining disparate concepts and managing the uncertainty present in assessments [9]. Bayesian networks can also be used as a formal underpinning for attack graphs [1]. In [31] a method for expressing defense graphs using Extended Influence Diagrams, a flavor of Bayesian networks, has been presented. This paper builds on that work.

3. Using extended influence diagrams for expressing defense graphs

3.1 Extended influence diagrams

An influence diagram is a Bayesian network extended with special nodes indicating decisions and utilities, in addition to the so called Bayesian chance nodes [5][8]. Extended influence diagrams are in addition equipped with a new kind of edge in order to clearly separate between causal relations and definitions [4], cf. Figure 2.

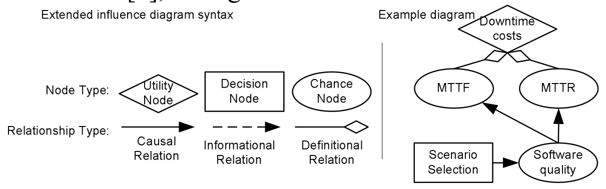


Figure 2 -The notation of extended influence diagrams.

Extended influence diagrams feature random variables associated with chance nodes that may assume values, or states, from a finite domain such as {High, Medium, Low} or {True, False}. A chance node could for example be "encryption strength" or "use of digital signatures".

The nodes in extended influence diagrams are connected to each other through causal or definitional arcs. Causal arcs capture relations of the real world, such as "stronger encryption yield higher confidentiality". Definitional relationships are on the other hand defined by the modeler, who also specifies how the defined property is defined by its parents [4]. The concept of security can for example be defined through preservation of confidentiality, integrity and availability. Probabilistic inference is performed by propagating values through the network; given the value of one node, the values of related nodes can be statistically inferred.

Decision nodes may as chance nodes assume one of several predefined and mutually exclusive states and can be coupled with chance nodes to express the capability to influence, or be influenced by, chance nodes. Utility nodes are used to express the utility associated with a combination of states in chance and decision nodes. A value expressing utility, positive or negative, is assigned to each states of influencing chance nodes and decision nodes. A utility node could for instance be "Economic loss" and this could be influenced by whether an attack is successful or not. For more comprehensive treatments on Bayesian networks, influence diagrams and extended influence diagrams see [4][5][6][7] and [8].

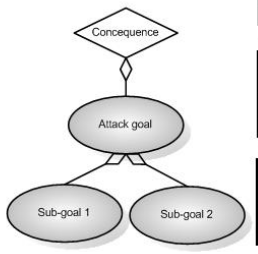
Similar to the development of Bayesian networks, the creation of an extended influence diagram involves two parts: one qualitative part where the structure of the extended influence diagram is constructed and definitional relationships are quantified; and one quantitative part where causal influences are quantified.

In the qualitative part various sources can be used as input, including literature, statistical data and expert judgment. A heuristic method developing Extended Influence Diagrams from scientific texts is presented in [12]. The quantitative definition of conditional probability tables are often considered the more difficult part of modeling Bayesian networks [10] [11], and consequently also extended influence diagrams. The commonly used sources to quantify these conditional probabilities are also literature, statistical data, and human experts [11]. Methods have also been developed to base probabilities on combinations of these sources [10][11].

3.2 Expressing defense graphs with extended influence diagrams

Extended influence diagrams can be used to express attack trees [31]. The steps in an attack can be illustrated by chance nodes with the states "Success" and "Failure". The AND-relationships and OR-relationships in the attack graph can be expressed using deterministic definitional relations and specified through conditional probability tables (cf. Figure 3). The impact from consequences of a successful attack can be taken into consideration and expressed through utility nodes where the negative impact of an attack is quantified.

By extending this structure with the countermeasures influencing how probable an attack is to succeed, a defense graph can be created [31]. In the defense graph the countermeasures are included as attributes. These countermeasures have an influence the on probability that an attacker succeeds with his or her sub-goals, and indirectly with his ultimate goal (cf. Figure 4).



Consequence

Attack goal 1	Success	Failure
Utility	-1000	0

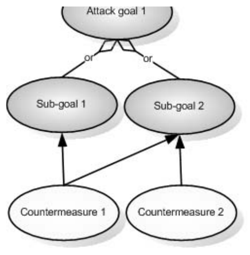
OR-relationship:

Sub-goal 1	Success		Failure	
Sub-goal 2	Success	Failure	Success	Failure
True	1	1	1	0
False	0	0	0	1

AND-relationship:

Sub-goal 1	Success		Failure	
Sub-goal 2	Success	Failure	Success	Failure
True	1	0	0	0
False	0	1	1	1

Figure 3 – Attack trees expressed through extended influence diagrams.



Sub-goal 1:

Countermeasure 1	State1	State2
Success	P_5	P_6
Failure	$(1- P_5)$	$(1- P_6)$

Sub-goal 2:

Countermeasure 1	State1		State2	
Countermeasure 2	State1	State2	State1	State2
Success	P_1	P_2	P_3	P_4
Failure	$(1- P_1)$	$(1- P_2)$	$(1- P_3)$	$(1- P_4)$

Figure 4 – The influence of countermeasures on the difficulty in succeeding attacks.

The capabilities of the attacker may also be included in the model. These can be assessed by attributes such as the time he or she have to spend; the prior knowledge about the system that he or she attacks; and the skill of the attacker in terms of compromising systems [23]. For the example used in this paper the attacker's capabilities are however excluded in the model. Instead assume that the model is designed for specific type of attacker, e.g. an outsider with world-class skill and prior system knowledge.

3.3 An example defense graph over password protection

In order to further explain the approach we here present a simple example borrowed from [31]. The example models a defense graph over access control of a standalone computer with password protection. Three general strategies are here assumed to exist to compromise a password protection mechanism. Firstly and secondly the attacker may performs a brute force attack or a dictionary attack against the system, and that way guess the password. A third strategy would be to find out the password by other means, for example by social engineering. An attack tree depicting these goals is given in Figure 5.

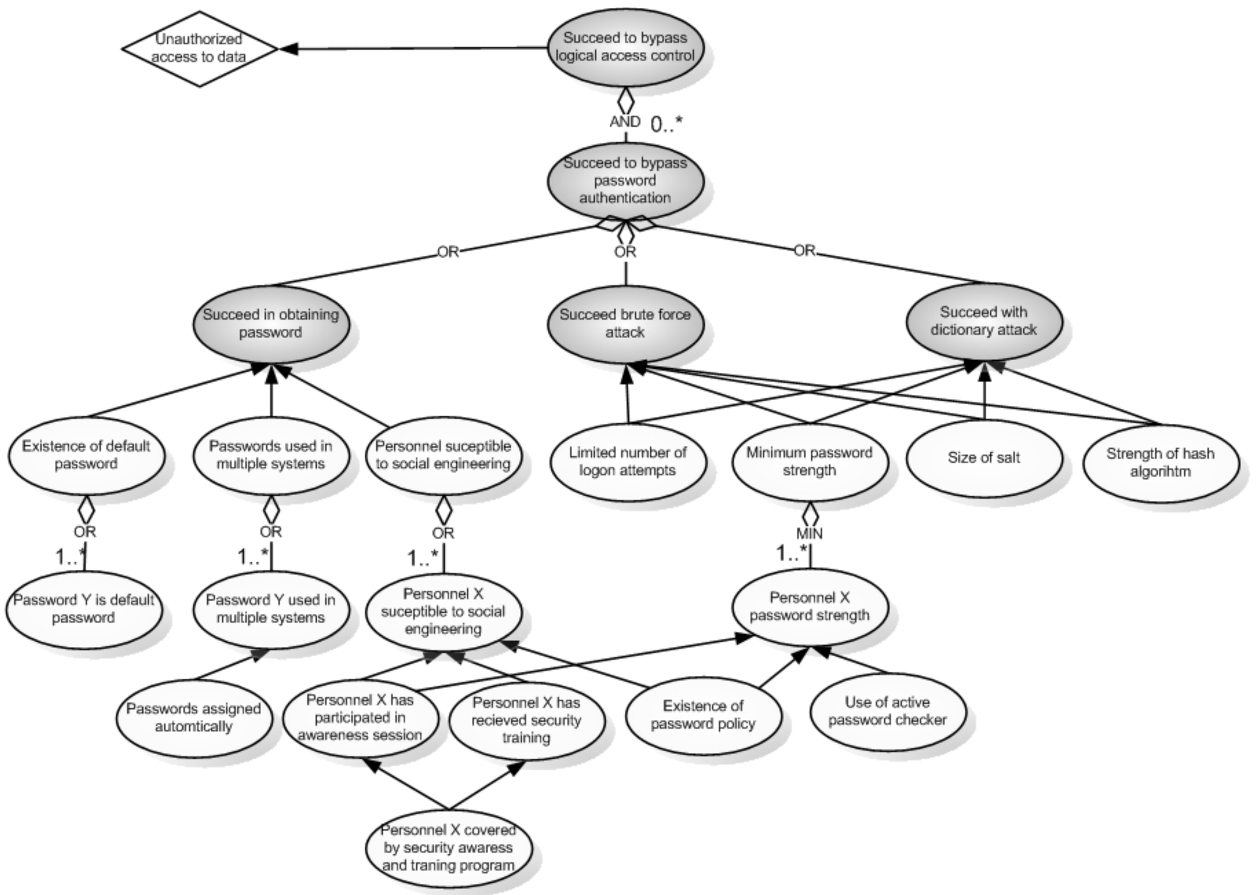

Unauthorized access to data
Succeed to bypass logical access control
AND 0..*
Succeed to bypass password authentication
OR
OR
OR
Succeed in obtaining password
Succeed brute force attack
Succeed with dictionary attack
Existence of default password
Passwords used in multiple systems
Personnel suceptible to social engineering
Limited number of logon attempts
Minimum password strength
Size of salt
Strength of hash algorihtm
OR 1..*
OR 1..*
OR 1..*
MIN 1..*
Password Y is default password
Password Y used in multiple systems
Personnel X suceptible to social engineering
Personnel X password strength
Passwords assigned automtically
Personnel X has participated in awareness session
Personnel X has recieved security training
Existence of password policy
Use of active password checker
Personnel X covered by security awaress and traning program

Figure 5 – Defense graph for access control expressed through an extended influence diagram. Grey nodes represent the attack graph and white nodes represent countermeasures.

The figure also contains some defense mechanisms relevant for the mentioned attacks. It is assumed that the difficulty of obtaining passwords is directly dependent on three attributes: the existence of default passwords that grant access; if the passwords are used in other systems; and if password holders (personnel) are susceptible to social engineering or not. Two factors that influence the difficulty of cracking passwords in a brute force attack are the strength of passwords, and if there is a limitation to the number of attempts that an attacker can try passwords using standard logon functionality. The size of salt and the strength of hash algorithms are also to be important for the difficulty of succeeding with brute force attacks. The same attributes are also of relevance to the difficulty of performing dictionary attacks, but most likely in another way.

The efficiency, functioning and strength of technical countermeasures are in many cases dependent on the quality of functions, processes and humans surrounding them. For instance, passwords do not offer strong protection if they are not kept confidential, if they are default passwords, or if they are weak. The presence of weak passwords may in turn reflect whether the employees have received security training or not, and if password policies are existent.

Figure 5 depicts the qualitative structure of an extended influence diagram expressing the abovementioned. In order to infer a numeric value to the top node unauthorized access to data, the qualitative properties of the extended influence diagram is also needed. Hence, to each attack goal and influencing attribute, a conditional probability table is assigned. The relationships between attack goals these are of either AND- or OR-tables, as exemplified above (cf. Figure 3). The conditional probability tables of attack graph-leafs specify the probability of an attack succeeding given a certain state in the influencing chance nodes (cf. Figure 4). An example of such a conditional probability table for the difficulty of obtaining passwords is given in Table 1. Conditional probability tables for how countermeasures influence each other are also specified. An example of such a table is given in Table 2 for the attribute "Password Y used in multiple systems". These tables can, as described above, be specified by experts, based on literature, or based on empirical studies.

Table 1 – Example conditional probability tabled for "Succeed in obtaining password".

Existence of default passwords	T				F			
Passwords used in multiple systems	T		F		T		F	
Personnel susceptible to social engineering	T	F	T	F	T	F	T	F
Success	0.9	0.8	0.8	0.7	0.8	0.7	0.7	0.1
Failure	0.1	0.2	0.2	0.3	0.2	0.3	0.3	0.9

Table 2 – Example conditional probability table for "Password Y used in multiple systems".

Passwords assigned automatically	True	False
True	0.07	0.95
False	0.93	0.05

4. Using abstract models for defense graphs

4.1 Abstract models

Thus far in the paper the focus have been on the analysis framework and how to derive a value of security (or expected losses due to lack of security). We now turn to the design or management of information systems. In recent years model based design and management of information systems have gained much attention and increased in popularity. There are many propositions on how to do this, and for information systems design the Unified Modeling Language (UML) is the de-facto standard language. For information system management a number of enterprise architecture frameworks have been proposed. A common denominator for these modeling languages is that they are designed from the point of view of what exists rather than what the models should be used for. The concept of abstract models has been proposed to avoid this [14]. The purpose of these is to merge analysis frameworks, such as extended influence diagrams, and modeling languages so that the analysis can be performed on scenarios modeled according the abstract model.

An abstract model comprise of four components: entities, entity relationships, attributes and attribute relationships. The first three of these components can be recognized from standard modeling languages such as the class diagrams of the UML. Entities are a central component in most modeling languages and can as in class diagrams be used to represent concepts of relevance for the model. These can be either physical artifacts, such as "computer" and "person", or more concepts such as "data" and "procedure". Entities are represented in a similar was as classes in UML are: a rectangular box with the name of the entity specified at its top.

Entities can in abstract models be connected through entity relationships. These entity relationships are depicted as lines spanning between the entities with roles names and multiplicities declared at the endpoints.

Attributes of abstract models are as in UML held by entities and are depicted as squared boxes within the entity belong to. However, unlike in UML, they are random variables or utility variables of finite domains. In other words the attributes of the abstract model are the chance and utility nodes of the extended influence diagram.

Finally, and thus naturally, abstract models in addition to other modeling languages have the attribute relationship. This is relationship is the same as the relationship in the extended influence diagrams and is either causal or definitional. If this attribute relationship spans two entities, it is always associated with a particular entity relationship, which is denoted by the dashed line, for indicating which entity relationship that is the reason why the attribute relationship exists. Cf. Figure 6.

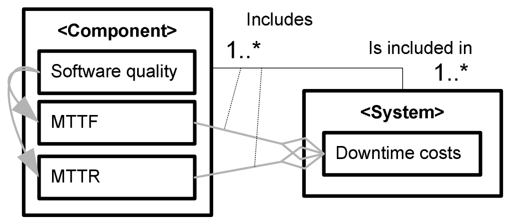


Figure 6 – Example of an abstract model.

Abstract models can thus be seen as metamodels enhanced with extended influence diagrams. This enhancement is not as straight forward as it seem. The reason for this is that the extended influence diagram does not differentiate between the instantiated and abstract modes. For instance, as a result of the multiplicities of entity relationships, the number of parents an attribute has may differ between instances of the abstract model. One way to handle this when describing the "instantiated extended influence diagram" is to use aggregation functions to specify the conditional decency an attribute has on its parents. Examples of such aggregation functions are “AND”, “OR”, “AVERAGE” and “MAX”.

4.2 Generating abstract models from defense graphs

Schechter [15] points out that the structure of attack trees depends on the system architecture and the choice of countermeasures. For example, in architectures that contain confidential data, attacks compromising the access control of this data are relevant. The countermeasures included in the architecture are also of relevance since the attack vectors that are available depend on these. The attacks bypassing an access control mechanism based on biometrics does for instance differ from attack against an access control mechanism based on passwords. Also, the multiplicity of countermeasures is of importance since additional countermeasures introduce additional hinders for adversaries.

Abstract models offer a way of handling these dependencies by dictating the attribute relationships as a consequence of an entity relationship. With this as a basis, it can be expressed how the relationship between attack-goals depend on the entities included in a model, and their relationships to each other.

The nodes of a defense graph, expressed as a extended influence diagram, are typically associated with some entity to which they belong. Based on this, the entities that are relevant for the assessment can be identified and populated with the appropriate attributes. For example, the node “Password Strength” can be interpreted as the entity “Password”, holding the attribute “Strength”.

If an entity relationship shall be included in an abstract model depend on the structure of the associated extended influence diagram. The entity relationships of relevance are those that determine if an attribute relationship shall exist in an instantiated version of the model. The example abstract model in Figure 6 does for instance have the entity relationship “includes” since this relationship between a system and component would result in attribute relationships between their attributes.

4.3 An example abstract model over password protection

The attributes of the extended influence diagram in Figure 5 refers several concepts that needs to be investigated in a security assessment. The objective is to protect some *data* and vital for this is the *password authentication mechanisms* which protects software such as *application* and *operating systems*. The password authentication mechanism uses passwords to grant or deny access and these *passwords* could or should be governed by a *password policy*. The *persons* who own the passwords have an influence on security related attributes according to the extended influence diagram and should also be considered in the assessment. Furthermore, if password holders are covered security *training and awareness program* is influencing the probability that these individuals have *received training* and *participated in awareness sessions*. Hence, this aspect should also be included.

The entities are included because they hold attributes that are of relevance to the assessment. For example, the entity *password* is relevant because they should have *strength.* The *persons* holding the passwords are relevant since their *susceptibility to social engineering* influence the difficulty to perform attacks.

For personnel, the participation *in awareness and training programs* is believed to influence their susceptibility to social engineering, hence if they are covered by such programs is also of relevance. The *password authentication mechanism* itself holds attributes such as *strength of password hash* and if there is an *active password checker* in use. These defensive attributes and the attributes representing attack goals and sub-goals give rise to the abstract model in Figure 1.

Relationships among attributes in the abstract model should exist if they will lead to an attribute relationship in an instantiated model. For instance, in an instantiated abstract model the entity relationship *owns* between a person and a password would imply that the person’s attribute “Has participated in security awareness session” influence strength of the password. If a password’s strength is influencing the minimum password strength of an authentication mechanism depends on whether it *gives access to* that particular password authentication mechanism. In the same way, a password protection mechanism’s attributes will only influence the difficulty of bypassing logical access control of software if it *protects* that specific application or operating system. The attribute relationships that these entity relationships are associated with is shown with dashed lines Figure 7.

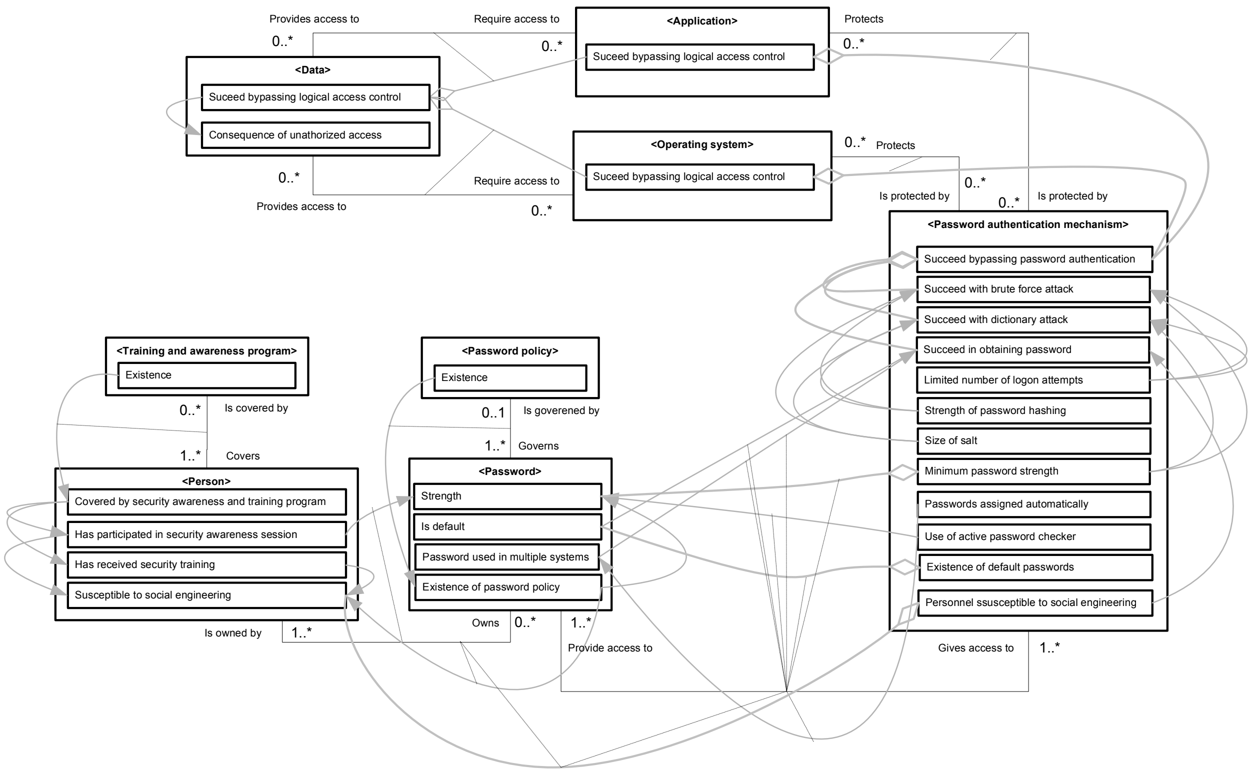

Provides access to
Require access to
<Application>
Suceed bypassing logical access control
Protects
0..*
<Data>
Suceed bypassing logical access control
Consequence of unathorized access
<Operating system>
Suceed bypassing logical access control
Protects
Provides access to
Require access to
Is protected by
Is protected by
<Password authentication mechanism>
Succeed bypassing password authentication
Succeed with brute force attack
Succeed with dictionary attack
Succeed in obtaining password
Limited number of logon attempts
Strength of password hashing
Size of salt
Minimum password strength
Passwords assigned automatically
Use of active password checker
Existence of default passwords
Personnel ssusceptible to social engineering
<Training and awareness program>
Existence
Is covered by
Covers
0..*
1..*
<Password policy>
Existence
Is goverened by
Governs
0..1
<Person>
Covered by security awareness and training program
Has participated in security awareness session
Has received security training
Susceptible to social engineering
<Password>
Strength
Is default
Password used in multiple systems
Existence of password policy
Is owned by
Owns
Provide access to
Gives access to

Figure 7 – Abstract model based to the extended influence diagram in Figure 5.

5. Instantiating the abstract model

5.1 Evidence collection

A security assessment typically involves data collection in terms of interviews, documentation studies, log reviews, deriving various metrics, penetration tests and more. The purpose of this is to collect information (evidence) about matters that are believed to influence security to facilitate analysis. One part of this information collection serves to identify the entities that need to be investigated and their relationship to each other. Another part of the data collection concerns the quality of various attributes and analyzing how these qualities influence security.

In relation to the framework presented herein the first part of information serves to scope the model based on the entities and relations available in the metamodel. The second part will add evidence to the state of attributes that influence the security. Evidence on attributes' state can be added by for example studying documents, performing interviews, from first-hand experience, or form penetration testing.

There is always some uncertainty as to whether the evidences has credibility and reflect the actual state of attributes [25]. A penetration test will for instance provide a high degree of confidence in results if the simulated attack is successful while unsuccessful attempts do not ensure the nonexistence of vulnerabilities. Documents describing systems may be old and obsolete, persons interviewed may be biased etc. Furthermore, some attributes are harder to assess directly than others and the evidence collected on these will consequently be vaguer. Abstract models allow the uncertainty of collected information to be included in the assessment by expressing evidence on the state of attributes through chance nodes. These evidence chance nodes are influenced by the state of the assessed attribute.

In Figure 8 evidence obtained from an interview on the use of automatic password checker is depicted as an ellipse. Table 3 expresses the significance of this of evidence by describing the expected outcome of the interview based on the possible states of assessed attribute. In this example, the system administrator would give the answer "true" with 95 percents probability if there is an automatic password checker. With 10 percents probability the system administrator would wrongfully answer "true" even if there was no automatic password checker.

Table 3 – A conditional probability table specifying credibility of evidence on the use of automatic password checker.

	Administrator's answer	T	F
Automatic password checker	True	0.95	0.10
	False	0.05	0.90

While there may be numerous attributes influencing the security according to an abstract model, evidence on all of these does not have to be collected. However, the more evidence that is collected, the more certain the result is. A method for dealing with the tradeoff between data collection cost and its impact in abstract models has been presented by [32].

5.2 Constructing the concrete model

By instantiating the abstract model, a concrete model can be created. The model depicted in Figure 8 is a concrete model based on the abstract model in Figure 7. In this example the data for which expected losses is assessed are customer records and strategic plans.

To base an abstract model on an extended influence diagram that expresses the defense graph has direct benefits. Firstly, it ensures that the model used for assessment, and consequently the data collected for it, contains the data needed to for generating and assessing security using defense graphs. Secondly, since the model only covers the parts that are of relevance to the assessment, the assessment will only focus on things of relevance to its result. Furthermore, it is from an instantiated abstract model straightforward and supported by tools [14] to derive the attributes, attribute relationships, and conditional probability tables. Using an abstract model, the modeler will only have to model the entities, their relationships, and the state of attributes to assess the security.

Shown in Figure 8 is the expected loss from unauthorized access to the strategic plans. This depends on the probability that an adversary will succeed bypassing the logical access control of the ERP Client. This in turn depends on the attributes of its password protection mechanism, and so on.

Based on the entities instantiated in the concrete model and the relationship among these, a network of attributes can be derived. This network will correspond to an extended influence diagram expressing a defense graph. The attributes to be included can be derived from the entities that have been instantiated. And since the attribute relationships are associated with entity relationships, these can be derived based on the entity relationships that exist in the concrete model

Together with evidence, like the interview system administrators, Bayesian inference can be used derive a value of attributes in the same way they can in an influence diagram. This would include deriving the probability that certain attacks succeed; the expected consequence of attack attempts; and an index comparing the expected losses of today's solution and the optimal one. Furthermore, this can be done even if the evidence is incoherent, or incomplete.

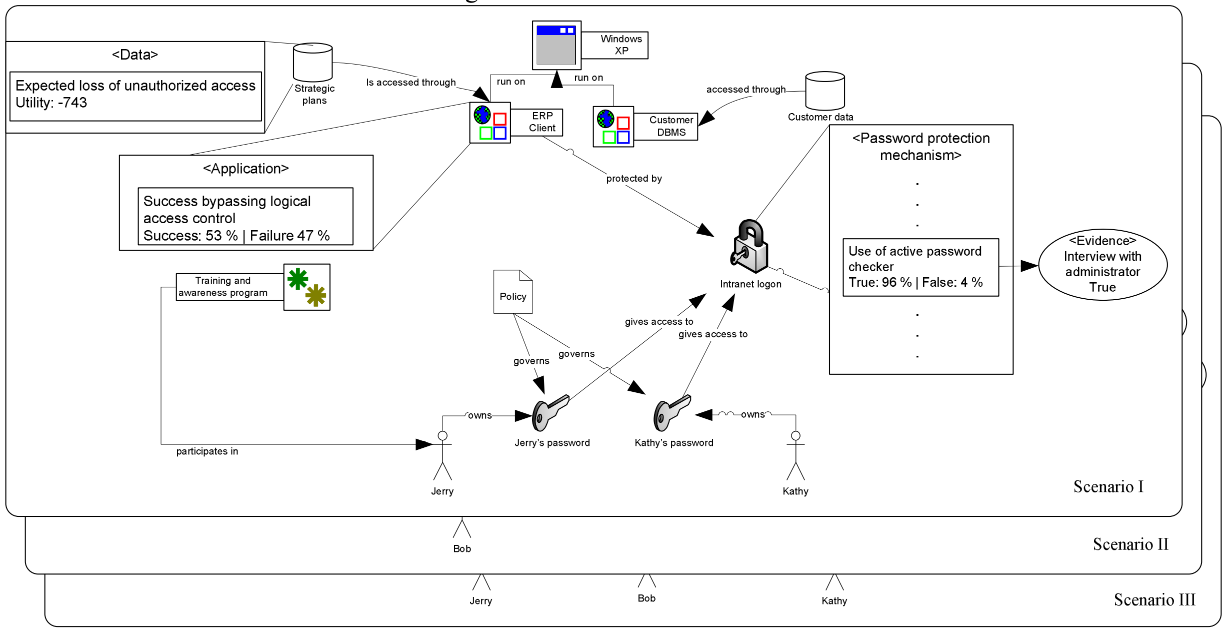


Figure 8 – A concrete model of the assessed enterprise. Entities and evidence on their attributes enable the probability of attacks success to be inferred and the expected loss to be calculated.

6. Discussion

When modeling security, or anything else for that matter, there is typically a tradeoff between completeness and feasibility on one hand, and simplicity and accuracy on the other. Models of attacks against cyber technology is no different, in fact attack graphs growing large is a known problem [33]. Hence, identifying all plausible attack steps and defense mechanisms prior an evaluation could become a daunting task. A suggested solution to the problem, proposed by Liu and Hong [1], is to use Bayesian networks to express attack graphs.

In a similar manner is the method proposed in this paper using extended influence diagrams, which also is form of Bayesian networks, to enable a compact representation of both attack graphs and defense mechanisms. By taking advantage of the probabilistic expressiveness of these Bayesian networks, the complexity of graphs can be reduced to a desirable size. This allows models to be created, without enumerating all plausible attacks or specifying them to the last detail, and instead include the uncertainty that is a result of keeping models on a high level. This also enables models to express uncertainty related to for instance access success that is a result of unknown, novel, attacks. By introducing this uncertainty to the analysis framework it is also possible to refine it iteratively. The more we learn about certain attacks and how to protect against them (by case studies, experiments or otherwise), the more we can reduce the uncertainties in the framework. In other words, the quality of results is related to how well we have prepared the analysis framework.

Except for managing the problem with knowing exactly all kinds of attacks and how to defend oneself against them, there is also always a practical problem of assessing cyber security with respect to our current knowledge. Typically in real world situations there are so many things that we know have a bearing on cyber security that there are an enormous amount of information about the state of the world that are needed for performing the security analysis. To collect all this data with high confidence take substantial efforts, if possible at all. However, the level of certainty in the results that is the highest imaginable. These two parameters thus need to be traded against each other by the decision maker. By using abstract models to generate defense graphs, and provide evidence on attribute states together with their credibility, a model based on incomplete information can provide a value on security that also provide an indication of the certainty of this value.

7. Conclusions

Model based design and is an established approach for management of information systems. For these models to support decision making relating to cyber security, the models of systems, software, and enterprises need to include the factors that influence security.

This paper has shown how defense graphs expressed with extended influence diagrams can be merged with the concept abstract models. Used as a metamodel this type of abstract model will include all the components necessary to perform a security assessment based on an architecture model. In addition to this, it will express how these components influence each other and how to derive a value on security and expected losses from an instantiated model.

8. References

[1] Y. Liu and M. Hong, Network vulnerability assessment using Bayesian networks, Proceedings of Data Mining, Intrusion detection, Information assurance and Data networks security, Orlando, Florida, USA, 2005, pp 61-71.

[2] S. Bistarelli, F. Fioravanti, P. Peretti, Defense trees for economic evaluation of security investments, Proceedings of Availability, Reliability and Security (ARES), Vienna, Austria, 2006, pp. 8.

[3] S. Bistarelli, F. Fioravanti, P. Peretti, Using CP-nets as a Guide for Countermeasure Selection, Proceedings of the 2007 ACM symposium on Applied computing, Seoul, Korea, 2007, pp. 300 - 304.

[4] P. Johnson, R. Lagerstrom, P. Narman, M. Simonsson., Enterprise Architecture Analysis with Extended Influence Diagrams. Information System Frontiers, 9(2), Springer Netherlands, 2007, pp. 163-180.

[5] R. Shachter, Evaluating influence diagrams. Operations Research, 34(6), Institute for Operations Research and the Management Sciences, Hanover Maryland, 1986, pp. 871-882.

[6] Neapolitan, R. Learning Bayesian Networks. Prentice-Hall, Inc. Upper Saddle River, NJ, USA 2003.

[7] Jensen, F.V., Bayesian Networks and Decision Graphs, Springer New York, Secaucus, NJ, USA 2001.

[8] R. Shachter, Probabilistic inference and influence diagrams. Operations Research, 36(4), Hanover Maryland, 1988, pp. 36-40.

[9] B.A Gran, Use of Bayesian Belief Networks when combining disparate sources of information in the safety assessment of software-based systems. International Journal of Systems Science, 33(6), 2002, pp. 529-542.

[10] M.J. Druzdzel and L.C. van der Gaag, Elicitation of Probabilities for Belief Networks: Combining Qualitative and Quantitative Information, Proceeding of the 11th Conference on Uncertainty in Artificial Intelligence, 1995, pp. 141-148.

[11] M.J. Druzdzel and L.C. van der Gaag, Building probabilistic networks: where do the numbers come from?, IEEE Transactions on Knowledge Data Engineering, 12(4), 2000, pp. 481–6.

[12] R. Lagerström, P. Johnson, P. Närman , Extended Influence Diagram Generation, Proceedings of the Interoperability for Enterprise Software and Applications Conference, 2007

[13] M. Howard and D. C. LeBlanc. Writing Secure Code, Microsoft Press, Redmond, WA, USA, 2002.

[14] P. Johnson, E. Johansson, T. Sommestad, and J. Ullberg, A Tool for Enterprise Architecture Analysis, In Proceedings of Enterprise Distributed Object Computing Conference, 2007, pp. 142-142.

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

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

[17] 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.

[18] S. Jha, O. Sheyner, and J. Wing. Two formal analyses of attack graphs. In Proc. of the 15th Computer Security Foundation Workshop, June 2002.

[19] O. Sheyner “Scenario Graphs and Attack Graphs,” Carnegie Mellon University, April 2004. PhD Thesis.

[20] O. Sheyner and J.M. Wing, “Tools for Generating and Analyzing Attack Graphs,” Proceedings of Workshop on Formal Methods for Components and Objects, 2004, pp. 344-371.

[21] J.J.C.H. Ryan and D.J. Ryan, Expected benefits of information security investments, Computers & Security, 25(8), 2006, pp 579-588.

[22] W.J. Caelli, D. Longley, and A.B. Tickle. A methodology for describing information and physical security architectures. Proceedings of the IFIP TC11, Eigth International Conference on Information Security, IFIP/Sec ’92, volume A-15 of IFIP Transactions, pages 277–296. Elsevier, May 27–29, 1992.

[23] S. Vidali and A. Jones, Analyzing Threat Agents & Their Attributes, School of Computing, University of Glamorgan, Technical Report CS-05-04, 2005.

[24] R. Vaughn, R. Henning, and A. Siraj, Information assurance Measures and Metrics: State of Practice and Proposed Taxonomy. Proceedings of 36th Hawaiian International Conference on System Sciences, 2003, pp. 331-334.

[25] E. Johansson, Assessment of Enterprise Information Security - How to make it Credible and Efficient, PhD Thesis, Royal Institute of Technology (KTH), 2005.

[26] S. Bistarelli, M. Dall’Aglio, P. Peretti, “Strategic games on defense trees”, Formal Aspects in Security and Trust, Springer Berlin / Heidelberg, 2007, pp. 1-15.

[27] J. Pamula, P. Ammann, A. Jajodia, V. Swarup., A weakest-adversary security metric for network configuration security analysis, Conference on Computer and Communications Security, Proceedings of the 2nd ACM workshop on Quality of protection, 2006.

[28] D.J. Leversage, E. James,, Estimating a System's Mean Time-to-Compromise, IEEE Security & Privacy, Volume 6, Issue 1, pp. 52-60. 2008

[29] P. Manadhata,, J. Wing,., M. Flynn, and M. McQueen, Measuring the attack surfaces of two FTP daemons. in Proceedings of the 2nd ACM workshop on Quality of protection, ACM Press, New York, 2006, 3-10.

[30] C. Philips, L.P Swiler , Graph-Based System for Network-Vulnerability Analysis, Proceedings of the 1998 workshop on New security paradigms, 1998.

[31] T. Sommestad, M. Ekstedt, P. Johnson, Combining defense graphs and enterprise architecture models for security analysis, Proceedings of the 12th IEEE International Enterprise Computing Conference, September 2008

[32] P. Närman, P. Johnson, R. Lagerström, U. Franke, M. Ekstedt,. Data Collection Prioritization for Software Quality Analysis, Second International Workshop on Software Quality and Maintainability, Athens, Greece, April 1, 2008.

[33] P. Ammann, D. Wijesekera, and S. Kaushik, “Scalable, graph-based network vulnerability analysis,” in Proceedings of the 9th ACM conference on Computer and communications security, pp. 217–224, November 2002.