Requirements-Driven Adaptive Security: Protecting Variable Assets at Runtime

Mazeiar Salehie*, Liliana Pasquale*, Inah Omoronyia*, Raian Ali†, and Bashar Nuseibeh*‡
* *Lero- Irish Software Engineering Research Centre, Limerick, Ireland*
{*Mazeiar.Salehie, Liliana.Pasquale, Inah.Omoronyia*}*@lero.ie*
† *School of Design, Engineering and Computing, Bournemouth University, Bournemouth, UK*
rali@bournemouth.ac.uk
‡ *Department of computing, Open University, Milton Keynes, UK*
Bashar.Nuseibeh@lero.ie

***Abstract*—Security is primarily concerned with protecting assets from harm. Identifying and evaluating assets are therefore key activities in any security engineering process – from modeling threats and attacks, discovering existing vulnerabilities, to selecting appropriate countermeasures. However, despite their crucial role, assets are often neglected during the development of secure software systems. Indeed, many systems are designed with fixed security boundaries and assumptions, without the possibility to adapt when assets change unexpectedly, new threats arise, or undiscovered vulnerabilities are revealed. To handle such changes, systems must be capable of dynamically enabling different security countermeasures. This paper promotes assets as first-class entities in engineering secure software systems. An asset model is related to requirements, expressed through a goal model, and the objectives of an attacker, expressed through a threat model. These models are then used as input to build a causal network to analyze system security in different situations, and to enable, when necessary, a set of countermeasures to mitigate security threats. The causal network is conceived as a runtime entity that tracks relevant changes that may arise at runtime, and enables a new set of countermeasures. We illustrate and evaluate our proposed approach by applying it to a substantive example concerned with security of mobile phones.**

***Keywords*-Security requirements, Adaptation, Causal reasoning**

I. Introduction

Security is concerned with protecting assets from harm. Identifying assets in the system-to-be and estimating their values are usually the initial steps of security requirements engineering [1]. Based on these assets, other security concerns, such as threats, attacks, vulnerabilities, security goals and countermeasures, are determined. Assets may evolve dynamically while a system is operating. For example, an asset's value can change, new assets can be added within the system boundary, or existing assets may no longer be under the system's protection anymore. These changes may affect the system's security concerns and the risk of harm. For example, an increase in the value of assets could increase the motivation of attackers, raise the threat level (i.e., probability of a threatening event occurring), and thus could increase the likelihood of successful (harmful) attacks. Certain security goals may become more critical if additional valuable assets need to be protected. Potential loss can also increase if an asset is compromised. Countermeasures, which are applied to mitigate security vulnerabilities, may also need to be strengthened to prevent greater losses.

Despite their central role in the security domain, assets are not treated as first class entities in the requirements engineering process, and are not used to drive system adaptation at runtime. Moreover, systems are often developed based on fixed security boundaries and assumptions, without the possibility to adapt when assets change unexpectedly, new threats arise, or undiscovered vulnerabilities are found.

For these reasons, this paper aims to integrate adaptive security (also called self-protection [2]) from the early stages of requirements modeling to the enactment of systems at runtime. We treat assets as first-class entities to enable some adaptation activities, namely, analysis and decision-making. Assets and other security concerns are modeled explicitly, together with the other requirements. Requirements, including security requirements, are represented through a goal model, which is augmented with a representation of vulnerabilities and countermeasures. Assets are expressed through an asset model, while threats and attacks are represented through a threat model.

To analyze the consequences of asset-relevant changes, we build a *causal network* (inspired by influence diagrams [3] and fuzzy cognitive maps [4]) from the asset, goal and threat models. Causal links then allow us to explicitly represent the effects of asset changes on other security concerns. This representation can express causal relationships among security concerns in a way that is amenable to change impact analysis. It also enables us to evaluate the overall utility of our countermeasures with respect to security goals.

Evaluating assets and their impact on security is usually qualitative, and these parameters could not be usually quantified in a rigorous way. Our causal network supports the qualitative nature of asset evaluation and other security concerns, such as the risk of possible attacks. We provide an analysis

mechanism for the causal network to estimate security risk and identify the most appropriate countermeasures.

Our causal network is conceived as a runtime model to track consequences of asset-relevant changes. These changes can be detected by suitable monitors or can be provided interactively by users. Some changes, such as adding or removing assets, require the causal network to be restructured. These changes need to be captured by our security model and then reflected in the causal network. At this stage of our research, the evolution of the causal network structure is performed manually. We applied and evaluated our approach on a substantive example concerning security of mobile phones. On one hand, the example provides evidence that our causal network results in the appropriate level of security, since an increase in the value of an asset moves the system to a higher level of protection (i.e. using stronger countermeasures), if necessary. On the other hand, we use simulations to demonstrate the feasibility of our analysis mechanism to generate plausible countermeasures.

The rest of the paper is organized as follows. Section 2 describes the adaptive security problem and our motivating (and running) example. Section 3 introduces our overall approach. Section 4 explains our security model that comprises asset, goal, and threat models. Section 5 deals with building the causal network based on the security model, and the reasoning capabilities it offers. Section 6 describes runtime adaptation using our causal network, and Section 7 discusses the experimental results evaluating the efficacy of our approach. Section 8 reviews key related work, and Section 9 concludes the paper.

II. Adaptive Security

Security requirements engineering is concerned with eliciting, representing and analyzing security goals and their relationships with other security concerns, such as critical assets, threats, attacks, risk, and countermeasures. However, these concerns may change dynamically as the operating environment or the requirements change. If these changes are not addressed, some existing countermeasures may become ineffective. Unfortunately, existing security requirements engineering techniques (such as [5], [6]) do not support detecting and managing runtime changes that specifically impact security. Adaptive security aims to capture and analyze such changes, and ultimately to enable countermeasures providing a satisfactory level of protection.

A. Problem Statement

Adaptive security must address different kinds of changes that may affect system security at runtime. For example, adding a valuable asset to the system may demand a higher level of protection, which in turn may require stronger countermeasures. Security goals may change or may be denied, in case some domain assumptions are no longer valid. New threats and attacks may emerge, undiscovered vulnerabilities may be appear, and existing countermeasures may become ineffective. Adaptive security must cope with the effects of these changes, which might compromise the system and harm its assets.

In this paper, we focus on a part of the adaptive security problem relating to asset variability as the main adaptation trigger. Adding a new asset or increasing the value of an existing asset can incentivize or discourage threat agents. Asset-relevant events may also impact on security goals and their criticality. When an asset is removed from a system, related security goals may need to be removed or changed as well. Finally, adaptive security must take into account risk mitigation via selection and configuration of a proper set of countermeasures. Note that, on one hand, countermeasures support the satisfaction of security goals, but, on the other hand, they can negatively impact usability, performance, or other quality attributes of the system. This makes adaptive security a multi-attribute decision-making problem that must deal with the costs and benefits of countermeasures.

This paper does not address how assets can be monitored and how adaptation actions can be applied to the system. Instead, it addresses analyzing how asset variability impacts on security and deciding how to adjust the protection level accordingly.

B. Motivating Example

Mobile smart phones are equipped with a wide range of applications that are increasingly used to perform personal and business tasks. However, their increasing popularity means that attackers and malicious users are more tempted to harm valuable assets accessible from these devices. For example, in the second quarter of 2011 the number of mobile malware incidents doubled compared to the first quarter of 2009[1].

In this paper we use mobile phone security to illustrate and evaluate our proposed approach. Our example is based on a recent survey [7] of mobile malware. Typical assets in a mobile phone include the phone itself, SIM (with monetary phone credit), banking/credit card information, email information (address, password and sent/received messages), and contact lists. Threats to these assets include stealing the phone, sending premium SMS messages, sending SMS spam, and collecting sensitive information. Users may accidentally give inappropriate permissions to applications, and installed applications may not encrypt sensitive information during transmission or storage. These vulnerabilities can facilitate potential attacks.

In this domain, increasing the value of an asset (e.g., the SIM card credit value) may increase the risk of loss. Adding new assets to the model (e.g., manipulating information such as credit card info details) may increase the threat level, e.g., by increasing the probability of breaking confidentiality.

[1] www.mcafee.com/au/resources/reports/rp-quarterly-threat-q2-2011.pdf

When this information is no longer available on a mobile phone, the corresponding security threat may diminish. In this context, adaptive security aims to select an appropriate configuration of countermeasures for the mobile phone in each situation. An example is shown in Table I: when the value of assets increases, countermeasures are adjusted.

Table I
POSSIBLE COUNTERMEASURE CONFIGURATIONS

SIM	Mobile	CC Info	Countermeasures
Low	Low	Low	PIN
Low	High	High	PIN, Encrypt Sensitive Info
High	High	High	Multi-factor authentication (PIN & finger print), Encrypt Sensitive Info

III. OVERALL APPROACH

Our approach treats assets as first-class entities during requirements modeling and runtime adaptation. As shown in Figure 1, assets and security relevant entities are integrated in the requirements model. Requirements are the starting point to *build* a causal network, which will be used at runtime to analyze the security risk and evaluate the utility of all possible configurations of countermeasures. The outcome of this analysis is used by an Adaptation Manager to select the best configuration and *(re-)configure* the running system.

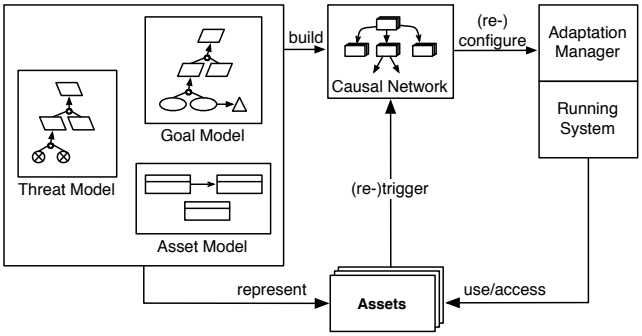

Figure 1. A framework to support adaptive security.

We use the KAOS [8] *goal model* to represent functional and non-functional requirements and its complementary *threat model* [5] (anti-model) to represent threats and attacks. The *asset model* explicitly *represents* assets in terms of KAOS entities. Note that we extend the goal model with an explicit representation of vulnerabilities, which may be brought by domain assumptions or system operations. Each vulnerability is linked to the attacks it facilitates. Security goals in the goal model may be further decomposed into concrete countermeasures. Section IV further elaborates these three models and their relationships.

The *causal network* is defined by the elements and relationships identified in the asset, goal, and threat models. Each node assigns a specific semantics to the corresponding security entity. For example, a node associated with a vulnerability represents presence (or the probability of presence) of a system weakness. The network links identify positive and negative causal relationships among security entities. For example, the positive link between a vulnerability (V1) and an attack (At1) has the following interpretation: an increased value of V1 causes an increase in the probability of success of At1. Our causal network also adds the concepts of risk and utility. The former is necessary to evaluate the security risk, while utility is fundamental to assess the advantages and disadvantages of a certain set of countermeasures when assets have specific values. Section V describes the causal network in more detail.

Assets are used or may be accessed by the running system at runtime and their value may change dynamically. For this reason, assets' values are monitored at runtime to tune the causal network and *(re-)trigger* its analysis. For each change in assets, the causal network will be updated and new utility values will be re-calculated for all applicable countermeasures. The most appropriate set of countermeasures is then selected by the Adaptation Manager, according to corresponding utility values. The asset, goal, and threat models can also change to accommodate new assets, remove existing assets, or vary security goals. These modifications are propagated onto the causal network by changing its structure.

IV. ASSET, GOAL, AND THREAT MODELS

This section describes the main features of the asset, goal and threat models with the aid of our mobile phone example.

A. Asset Model

The asset model represents assets and their relationships. In our example, we considered assets shown in Figure 2 including the mobile phone itself, SIM card, the credit card information and location data. Assets that are related to the mobile phone, such as SIM and credit card information, contribute to increase the phone value. The attacks that target the mobile phone (e.g., stealing the phone) may harm the related assets as well. Note that attacks that target credit card information may also harm the bank account. However, the boundary of our adaptive security problem does not consider protecting the bank account. This was a realistic decision, since each bank also has its own mechanisms to limit harm (e.g., daily money withdrawal cap) and to detect unusual activities (e.g., shopping from a new location).

B. Goal Model

The goal model represents the main objectives a system must achieve or maintain, and decomposes them into functional and non-functional requirements. This model includes security goals, such as confidentiality, integrity, availability and accountability (also known as CIAA). The priority (criticality) of a security goal may depend on the value of the asset that needs to be protected. Security goals have a hierarchical structure and can be ultimately decomposed into

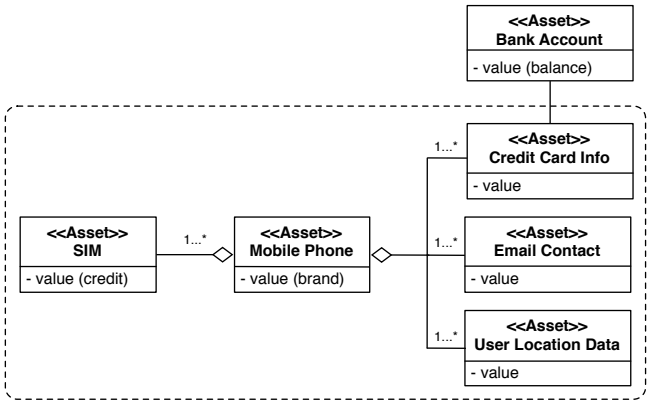

Figure 2. The asset model for the example. (Dashed line: system boundary)

concrete (operational) countermeasures. Countermeasures are functionalities to mitigate security risks (also known as security controls or safeguards [9]). In this paper we focus on preventive countermeasures, which avoid attempts to deny security goals. Note that vulnerabilities may be brought by system functionalities, including security countermeasures and domain assumptions. For this reason, we include an explicit representation of vulnerabilities in our goal model. Countermeasures are related to the vulnerabilities they try to mitigate, and a weight can be assigned to the link depending on the effectiveness of each countermeasure.

Figure 3 depicts the goal model for our example. Accountability and confidentiality are considered as security goals, and six countermeasures are designed in the system to achieve these goals. Six vulnerabilities are identified for the illustrated functional requirements, including no encryption of data in the device and root exploit (also called "jailbreak"). Some security goals cannot be satisfied without sacrificing other non-functional requirements, such as performance and usability. The countermeasures applied to enforce the satisfaction of security goals cannot be selected without considering their side effects. For example, if the system uses a stronger encryption algorithm, this countermeasure may cause degradation in performance or usability.

C. Threat Model

A threat model (or anti-model) typically includes threat agents (i.e. threat sources or counter-stakeholder), threat goals, and attacks (i.e. threat actions). Threat agents can be natural (e.g., flood), human (e.g., hacker), or environmental (e.g., power failure) [9]. In this paper we do not consider threat agents as a part of our anti-models, and we simply represent their potential goals. Assets are linked to the threat goals they motivate, while threat goals are associated with the attacks that are performed for their achievement.

Threat goals represent motivations of threat agents to attack a system. In our example, possible motivations [9] include monetary gain, blackmail, destruction and competitive advantage. Some of them can be directly associated with the assets under protection and can be decomposed into anti-goals (i.e. negation of security goals [5]) aimed at compromising the targeted assets. This paper represents anti-goals as KAOS obstacles [5]. Attacks are actions through which threat goals can be achieved [9] and assets would ultimately be harmed. Therefore, attacks can be modeled as operationalizations of threat goals.

Figure 4 depicts the threat model of our mobile phone example. The top goal is monetary gain and is decomposed into stealing phone credit and collecting sensitive information from the user device. Each of these anti-goals are refined and then linked to attacks, which include phishing and malware attacks. Note that anti-goals negate security goals. For example anti-goal *Collect Sensitive Info* negates *Confidentiality*.

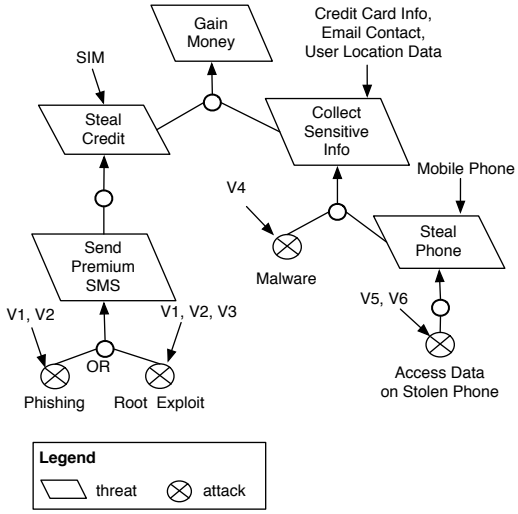

Figure 4. The threat model for the example.

V. The Fuzzy Causal Network

Causal networks are used in decision theory to support qualitative (e.g., Fuzzy Cognitive Maps (FCM) [4]) and quantitative (e.g., Bayesian decision networks [10]) analysis and decision making. A causal network built from the three discussed models enables us not only to analyze consequences of asset-relevant changes, but also to perform an impact analysis of potential decisions. However, the main challenge is that quantitative analysis may not be always feasible, due to incompleteness or imprecision of data. For instance, quantitative risk evaluation is challenging [11], since many threats might be rare or new. Entities such as risk and threat are intrinsically uncertain, but imprecision adds another level of uncertainty that prevents us assigning specific values to these entities. Therefore, we have to deal with not only the uncertainty of occurrence of an observable event, but also the imprecision and ambiguity of event assessment. For other entities, such as assets, it is not probability but imprecise evaluation that makes the value uncertain.

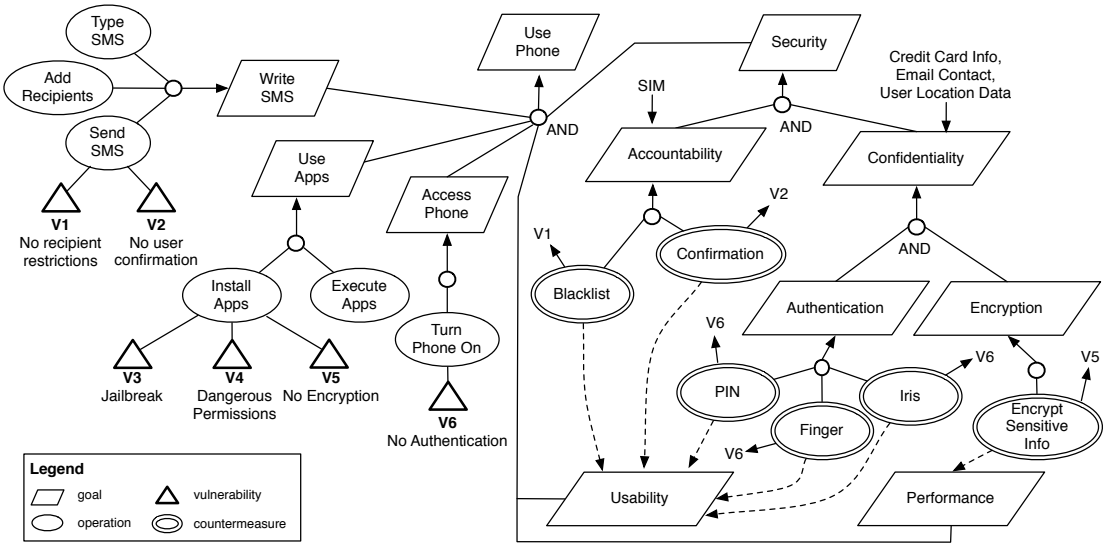

Figure 3. The goal model for the example.

To cope with such uncertainty, there are many techniques that are based on probabilistic or fuzzy values. Although, in the first category solutions like probability intervals [12] for example can be used for risk assessment, but only if the entities are probabilistic in nature. On the other hand, techniques based on fuzzy values can cover all entities in our three models, even those with uncertain probabilities. Indeed in risk assessment, linguistic terms (e.g., 'high', 'medium', and 'low') and intuitive judgements of domain experts can be better expressed by fuzzy values. Therefore, a fuzzy causal network seems a natural fit to our problem, and we still retain the possibility of benefiting from any precise quantitative evidence, if it is available. Our proposed causal network is inspired by FCMs [4], a well-known fuzzy causal network. However, we enriched the network with utility and decision nodes from the influence diagrams [3] to support decision making.

A. Building the Fuzzy Causal Network (FCN)

Our proposed Fuzzy Causal Network (FCN) is built upon the elements and links represented in the asset, goal, and threat models. Similar to an influence diagram, our causal network has three types of nodes [13]: *chance* nodes representing uncertain domain entities significant for causal reasoning (denoted by ovals), *decision* nodes indicating decisions to be made (denoted by rectangles), and *utility* nodes corresponding to the fitness value of the network configuration (denoted by a hexagon). Table II lists the nodes of our FCN. Except for the utility node, all the others are represented by fuzzy variables in the range $[0, 1]$. This allows us to initialize and analyze nodes whose values are imprecise. The asset node is associated with its value (or criticality), threat with the threat level, and attack with its probability of success. A vulnerability node shows the presence or the probability of presence of a system weakness. NFRs are nodes associated with the satisfaction level of non-functional requirements other than security (e.g., usability or performance) to represent side-effects of countermeasures. Each security goal represents its satisfaction value as a fuzzy variable. We incorporated logical AND-OR relations between security goals, which are directed from offsprings to parents.

Table II
NODES IN THE CAUSAL NETWORK

Node	Meaning	Type
Asset (As)	Value	Chance
Threat (T)	Threat level	Chance
Attack (At)	Probability of success	Chance
Security goal (SG)	Satisfaction level	Chance
Vulnerability (V)	Presence of vulnerability	Chance
Non-Functional Requirements (NFR)	Satisfaction level	Chance
Partial Risk (PR)	Partial risk of an attack	Chance
Total Risk (TR)	Risk of all attacks	Chance
Countermeasure (CM)	Strength	Decision
Utility (U)	Value	Utility

Figure 5 depicts the abstract structure of our FCN. The proposed FCN is a directed graph, and each causal link in the FCM is labeled with a signed weight, which represents the strength of causal relationship between two nodes of the network. A positive causal link, $A \xrightarrow{+} B$, means that increasing A causes increasing B, while a negative causal link, $A \xrightarrow{-} B$, means that increasing A causes decreasing B. A high/low weight for a positive or negative causal link means that an increase of A may cause a great/small increase or decrease of B. While generally weights can be fuzzy labels, as in the FCM, in this paper we used quantitative labels in the $[1, 0]$ interval. Note that the FCN is an acyclic graph, and the loops in Figure 5 are between different chance nodes; e.g., between two assets such as *Credit Card Info* and *Mobile*.

Chance nodes - Except the countermeasure, all entities taken from the three predefined models are chance nodes in the FCN. The value of an asset also depends on other related assets (see positive link between assets in Figure 5). For example, the value of a *Mobile phone* increases when the value of the enclosed information and the *SIM* card increases, which implies positive causality. The value of assets also has a positive impact on the threat level, since, in case of an attack, the loss would be higher. For example, the higher the value of the enclosed information, the higher the level of *Steal Mobile Phone* threat.

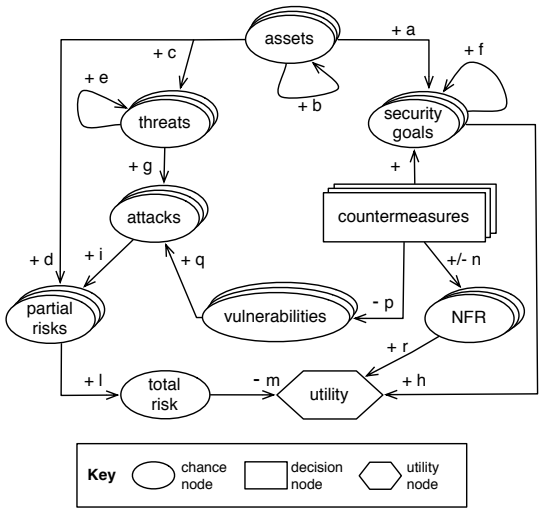

Figure 5. The abstract model of the fuzzy causal network

Threat nodes propagate their values down to their offspring, as in the threat model (shown by the positive link between threats in Figure 5) and finally to their underlying attacks (the link between threats and attacks), since threats increase the probabilities of attacks. For example, the *Steal Mobile Phone* threat propagates its value to *Access Data On Stolen Phone*. The probability of an attack is also affected by the presence of a set of vulnerabilities. For example, the success of *Access Data On Stolen Phone* depends on the presence of the *V5* vulnerability. The higher the probability of this vulnerability, the higher the probability of *Access Data On Stolen Phone* to succeed.

The value of each security goal is its satisfaction level that depends on subgoals and corresponding countermeasures. But the asset value has also influence on how the satisfaction level and criticality of associated security goals are determined (see the link between assets and security goals). The strength level of a countermeasure impacts the corresponding security goals and propagates upwards in the goal model following the semantics of AND/OR fuzzy operators (i.e. min/max). For example, the *Confidentiality* goal is affected by the *Credit Card Info*, *Location Data*, and *Email Contact* assets, and the *PIN*, *Iris*, *Finger*, *Encrypt Sensitive Info* countermeasures.

Partial risk and *total risk* are chance nodes with no counterparts in the asset, goal and threat models. A partial risk node is created for each attack to estimate the corresponding risk, and the total risk aggregates all the partial risks. Partial risk estimation follows the general definition of risk, which is the probability of an attack multiplied by the possible resulting loss. Therefore, each partial risk depends on the value of the targeted assets, i.e., the loss factor, and the probability of the related attack (see positive links from assets and attacks to partial risk in Figure 5). For example, the partial risk of *Malware* depends on the probability of the attack and the value of contained information.

Decision nodes - Each countermeasure is a decision node in the causal network. Vulnerabilities can be mitigated through the application of suitable countermeasures (see the link between countermeasures and vulnerabilities). For example, the *Encrypt Sensitive Info* countermeasure mitigates *V5*. On the other hand, a countermeasure may deteriorate or improve the satisfaction level of some non-functional requirements (see the link between countermeasures and NFRs). For example, *Encrypt Sensitive Info* has a negative impact on *Performance*. The countermeasure value is associated with its strength level in mitigating vulnerabilities. Some countermeasures, such as *confirmation*, are crisp since they can be enabled or disabled, while others, such as *encryption*, are fuzzy multi-value variables since they may have values between zero and one. For encryption, the value depends on the length of the key or the algorithm adopted.

The utility node - This node expresses the effectiveness of selected countermeasures, according to the total risk, the satisfaction of security goals and impacts on NFRs. The utility node should aggregate all these costs and benefits. Benefits depend on how much countermeasures can mitigate the risk, and costs indicate how much they hurt other NFRs. The risk and NFR nodes have a negative impact on the utility, while security goals have a positive impact. For example, the utility of applying the *PIN* countermeasure benefits the authentication, while a long pin has a negative impact on usability. By changing the value of this node, a different utility value can be determined.

Figure 6 illustrates a part of the causal network for our example. Chance nodes are defined based on entities essential for adaptation in the asset, goal and threat models. For each attack node, we add a partial risk node to the network. In Figure 6, malware and access data on stolen phone risk nodes are two samples of these partial risks. Then, the network is augmented with the total risk and utility nodes. Each of these nodes, aggregate their inputs depending on its semantics, as will be elaborated in the next section. The next step is to establish causal links between the nodes. Links in the security models are not enough for this purpose, and additional information from the domain experts and security requirements artifacts are required. For instance, risk assessment artifacts help defining links to

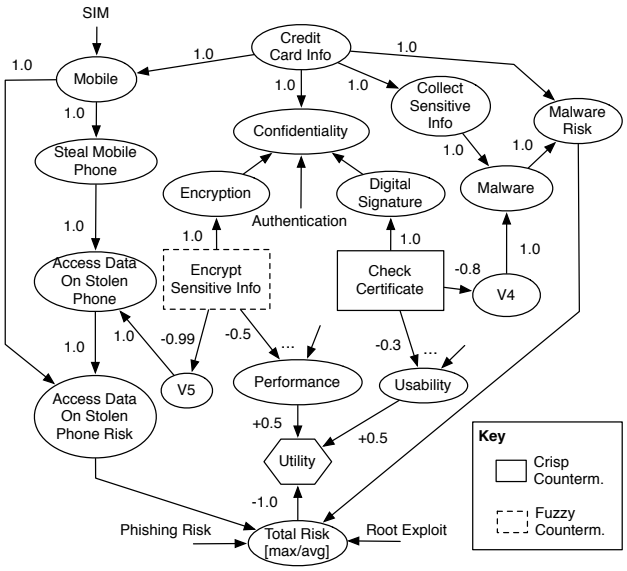

Figure 6. A part of the causal network for the mobile phone example

partial and total risk nodes. Assigning weights also needs domain knowledge, although the weights normally require adjustment.

B. Utilizing the FCN in Adaptation

The FCN is set to analyze the impact of asset variability on security, particularly risk, and select the most appropriate countermeasures to mitigate risk at runtime. The analysis and decision-making are performed using causal reasoning on the FCN. This section describes how this can be done.

Evaluating nodes - Each node in the FCN needs to aggregate causal effects from input links. This may be different for each node depending on the attributed semantics. Table III shows the aggregation functions we used in each node. The term $\{A\} \rightarrow B$ denotes the set of nodes of type A that are causally affecting B. Aggregation functions are selected from the set of these functions: Minimum, Maximum, Average and Sum. The Sum function is selected for the utility node, since it is not a fuzzy variable and accumulates risk, NFRs and security goals. For all the other nodes, selecting one of the other three functions indicates how much we want to intensify the output based on inputs. The Maximum function is more conservative, because for instance if we use Maximum to aggregate causal effects of several assets to an asset, countermeasures would have been over strengthened. We selected Average for this purpose to take the middle level between conservative and relaxing aggregations. Note that the aggregating functions are applied after applying weights to each input, which means by tuning weights we can adjust the input-output mapping.

In some cases there are more sets in the left side of the causal link; e.g., $\{As\}, \{CM\} \rightarrow SG$. In these cases, the reasoning mechanism initially aggregates the same type links and then combines different types. For example to evaluate SG, first the effects of $\{As\}$ on SG are aggregated by the Maximum function since asset protection is extremely important and associated goals should be promoted (being more critical). Then the impacts from $\{CM\}$ are combined with the Maximum function, because links between countermeasures connected to an SG is OR (as shown in Figure 3). Finally these two effects are aggregated by the Minimum function, which is more conservative for SG. This means the lower the goal value, the higher the chance of having a stronger countermeasure.

Table III
AGGREGATING CAUSAL EFFECTS

Causal Link	Aggregation
$\{As\} \rightarrow As$	Average
$\{As\} \rightarrow T$	Maximum
$\{T\} \rightarrow At$	Maximum
$\{V\} \rightarrow At$	Average
$\{T\}, \{V\} \rightarrow At$	Minimum
$\{As\} \rightarrow SG$	Maximum
$\{CM\} \rightarrow SG$	Maximum
$\{As\}, \{CM\} \rightarrow SG$	Minimum
$\{CM\} \rightarrow V$	Average
$\{CM\} \rightarrow NFR$	Average
$\{As\} \rightarrow PR$	Maximum
$At \rightarrow PR$	No aggregation (only one attack)
$\{As\}, At \rightarrow PR$	Minimum
$\{PR\} \rightarrow TR$	Maximum or Average
$TR, \{SG\}, \{NFR\} \rightarrow U$	Sum

Alternatively, Minimum and Maximum in the min/max fuzzy inference means multiplication and addition. To evaluate the partial risk PR, the loss factor (asset effect) should be multiplied by probability of attack, which is translated into the Minimum function. For aggregating partial risk to total risk, the Maximum function makes sense, because it means adding up partial risks. But we consider two options of Maximum and Average in Section VI to investigate the effect of each option on risk calculation.

Initialization and tuning - Before reasoning, the nodes and weights on causal links should be initialized. The initial values may come from stakeholders, domain experts, security requirements artifacts and existing evidence (e.g., statistical data). Initial values of countermeasures are the default security settings that may be changed later by the FCN at runtime. Note that sometimes, some countermeasures may not be available (e.g., failed) or disabled by the user. The weights of the causal links should be tuned based on existing qualitative and quantitative evidence. We used sensitivity analysis to investigate the behavior of output in different configurations and tune the weights. The next section discusses sensitivity analysis in more detail.

The asset evaluation, in particular, has a significant impact on the effectiveness of decision making in adaptation. A report published by Symantec offered values for some of the assets in our mobile phone in the underground market [14], but users may evaluate these assets differently. In our example, we considered that assets are evaluated from the user's

viewpoint. This could be rather conservative in some cases. For example, a user location may be more valuable to an attacker than to the user himself.

Runtime Reasoning - At runtime, after any changes in assets, the FCN should be re-evaluated and countermeasures should be changed if necessary. We call the set of countermeasures a security configuration, or simply a *configuration*. First, the value of the changed asset(s) is updated and then the causal effect through outward links will affect connected assets, threats and security goals, as shown in Figure 5. These effects will be propagated through other links towards the utility node. To avoid an infinite update without convergence, we specified a threshold (a convergence criterion): in case the value of a node does not change more than the specified threshold, the computation of the value at that node can terminate.

Every time a change in the assets takes place, the maximum expected utility needs to be specified for the given network state. For this purpose, different values of decision nodes, i.e., countermeasures, should be tried. While searching for the best utility would be time-consuming and can be more efficient by applying heuristics, in this paper we consider a global search to evaluate all possible countermeasures. As noted before, we used a modified version of FCM [4] in the causal network. We used the original FCM reasoning mechanism, implemented in JFCM (http://jfcm.megadix.it/), and extended it to first consider decision and utility nodes, and then to incorporate AND-OR relationships between goal nodes. Assets impact directly connected security goals, while countermeasure effects propagate bottom up through the goal structure's AND-OR links.

VI. Simulation

As a part of the evaluation of our work, we built the requirements model and the fuzzy causal network for our mobile phone example, and conducted several experiments in a simulation in order to determine the efficacy of our causal network. The network was evaluated with the three following scenarios:

Scenario 1 (S1): In the first scenario the causal network includes three assets: mobile phone, credit card information and phone credit (SIM). We assume that there is no countermeasure for malware attacks. Therefore, malware might be installed, and might harm valuable assets.

Scenario 2 (S2): The second scenario is removing an asset from the system. In our example, credit card information (*CC info*) is removed from the phone. In this case, we are interested in knowing how the FCN changes the countermeasures after removing the asset, and also when there is no *CC info* how the variability of other assets adjusts the configuration.

Scenario 3 (S3): In the third scenario, we assume that we have all the three assets in S1, and we add a countermeasure to avoid installing malware applications, which is checking the authenticity of digital signatures before installing applications.

The main objective of our simulation was to assess if the causal network efficiently suggests reasonable countermeasures, which mitigate existing vulnerabilities. For this purpose, we conducted a set of six experiments based on the above scenarios, after running a sensitivity analysis.

A. Sensitivity Analysis

We ran a sensitivity analysis before the experiments for two reasons. First, we wanted to tune the weights based on the mapping of asset changes to utility values. We generated different combinations by changing values of the assets from low to high, to analyze the effects on the other security concerns. We also varied the strength of the countermeasures to better understand the changes in the utility value. For the sensitivity analysis, we assume that the initial values of the nodes have a uniform distribution.

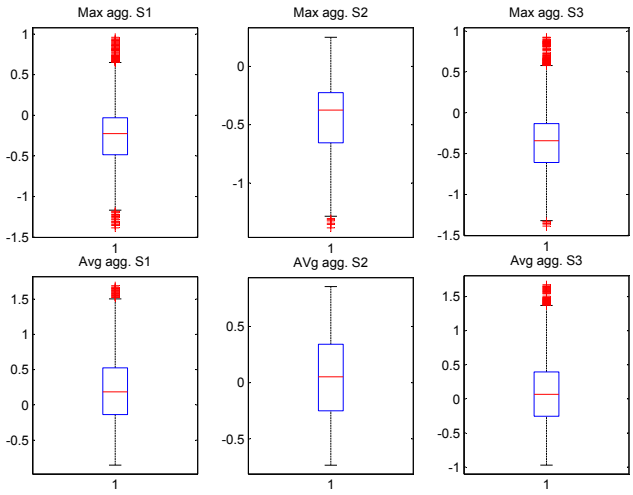

Figure 7. Utility Box plots with max and average aggregation for S1-S3

Second, we were interested in investigating the effect of selecting Maximum or Average aggregation function for total risk calculation. Figure 7 shows Box plots of utility values in the scenarios for the two aggregation functions. Utility values for Maximum are negative because of selecting stronger countermeasures. Both functions selected the same configuration for the best utility in S2, while in S1 and S3 Maximum chose the more secure one. S1 to S3 generated 27000, 5400 and 54000 different cases respectively, and the network was updated in maximum six iterations.

Generally, using the Maximum function leads to selection of stronger countermeasures that may have an adverse impact on other NFRs. On the other hand, the Average function may offer countermeasures that are less strong, but only slightly deteriorate other NFRs. If security requirements are deemed more important, the Maximum function is selected, otherwise the Average function is chosen.

Table IV
SAMPLE CONFIGURATIONS (H:HIGH, VH: VERY HIGH, M: MEDIUM, L: LOW, EN:ENABLE, DS:DISABLE, S:STRONG, N/A: NOT AVAILABLE)

	Assets		Attacks				Countermeasures					
Scenario	SIM	CC info	Malware	Phishing	Send SMS	SMS Blacklist	Encrypt	Finger	Iris	PIN	SMS Conf.	App cert.
S1	VH(1)	VH(1)	VH(1)	VH(1)	VH(1)	S(1)	S(1)	Ds(0)	En(1)	Ds(0)	Ds(0)	N/A
S1	L(0.3)	VH(1)	VH(1)	M(0.5)	L(0.3)	M(0.5)	S(1)	En(1)	Ds(0)	Ds(0)	Ds(0)	N/A
S2	VH(1)	N/A	VH(0.9)	VH(0)	VH(1)	S(1)	Ds(0)	Ds(0)	Ds(0)	Ds (0)	Ds(0)	N/A
S2	L(0.3)	N/A	VH(0.9)	M(0.5)	L(0.3)	M(0.5)	Ds(0)	Ds(0)	Ds(0)	Ds(0)	Ds(0)	N/A
S3	VH(1)	VH(1)	VH(1)	VH(1)	VH(1)	S(1)	S(1)	En(1)	Ds(0)	M(0.5)	En(1)	En(1)
S3	L(0.3)	VH(1)	VH(1)	M(0.5)	L(0.3)	M(0.5)	S(1)	En (1)	Ds(0)	Ds(0)	Ds(0)	En(1)

B. Configuration Selection

At runtime, changing the asset value may cause an increase in risk, and the system may need to modify its countermeasures accordingly. To select countermeasures with the best utility we used a global optimal strategy. Table IV shows a set of configurations selected for S1-S3 using the Maximum risk aggregation function. For each scenario, two cases are evaluated: first, very high value of phone credit (SIM) and credit card information (CC info), and, second, low value of SIM and very high value of CC info. These cases are shown in the first and second rows of each scenario.

In S1, lowering the SIM card value results in reducing the possibility of sending premium SMS and slightly lowers the possibility of success of phishing attacks. The asset change leads to decreasing the level of countermeasures related to authentication and accountability, which intuitively makes sense. The iris checking is changed to fingerprint matching, which improves the usability and performance of the system. In S2, no payment information is stored in the phone. Again, lowering the SIM card value reduces the possibility of sending premium SMS. The causal network only suggests adding a blacklist countermeasure, which again intuitively is a reasonable choice.

In S3, decreasing the asset value also shows a decrease in the required protection level. Interestingly the single factor authentication countermeasure provides higher utility than the multi-factor authentication. The fingerprint authentication, which is stronger than PIN, is still enabled. The SMS confirmation can be disabled to provide better usability, since the SIM card has a reduced value. In all these cases, usability of suggested configurations is high and the impact on the performance is medium. For each situation, the FCN generates the ranked list of configurations in less than $1ms$.

VII. RELATED WORK

Autonomic computing [2] research has considered two aspects of self-protection: defending against malicious attacks and cascading failures, and anticipating problems and avoiding them. Existing solutions mostly do not consider application security [15], while a view re-enforced by Ghosh et al. [16] argued that we still need to protect applications from "ambiguous security policies, data-driven attacks through allowed services, and insider attacks".

In security requirements engineering, the main objective is to consider security earlier in the development life cycle [17]. As noted previously, van Lamsweerde [5] proposed a dual model composing goals and anti-goals to elaborate security requirements. We have argued in this paper that an asset model should also be added and linked to the security requirements model. Haley et al. [1] noted that although knowing the goals of attackers may be useful for quantifying harm, security is not a zero-sum game. This means success of an attack will not necessarily render a security goal denied. Therefore, we assumed that anti-models consist of both motivations and anti-goals. In another work, Elahi et al. [6] enriched the i* model by adding attackers and vulnerabilities, without taking into account variability of assets and its impact on security.

Requirements at runtime determine whether software has deviated from the expected behavior. Fickas and Feather [18] highlighted the importance of runtime requirements monitoring for evolution. Souza et al. [19] introduced awareness requirements to monitor success and failure of requirements. Baresi et al. [20] proposed fuzzy live adaptive goals to deal with the denial of system goals using defined countermeasures. Salehie and Tahvildari [21] proposed a Goal-Attribute-Action Model (GAAM) to represent runtime goals in an ensemble for selecting adaptation actions. Our proposed FCN is not on directly monitoring requirements but rather on prevention as risks change.

There are few related contributions employing causal networks and decision trees in risk management. Sahinoglu [22] built a decision tree by connecting vulnerabilities, threats and countermeasures together for risk quantification. Although the approach is promising, it suffers from the limitations of decision trees (tree structure), and does not consider assets and security goals. Few efforts on risk-adaptive solutions have also been reported (e.g., [23] and [24]). Although these efforts considered risk estimation, they ignored asset variability as a source of risk change.

VIII. CONCLUSIONS AND FUTURE WORK

This paper has proposed a novel approach to support adaptive security, from requirements modeling to the enactment of a system at runtime. Our approach promotes assets as first-class entities. At design time, three models – of assets, threats, and security requirements – are assembled together. From these models, and by adding risk (partial and total) and utility nodes, a causal network is built. At runtime, changes in assets trigger the causal network, and the outcome of reasoning over this network is a list of security configurations with their associated utility values. Then the

most appropriate configuration is selected for adaptation. It is worth noting again that in this paper we have not discussed how to monitor assets and apply changes at runtime.

Linking assets to security concerns enables an adaptation manager to understand the costs and benefits of countermeasures in protecting assets, and to estimate the impact of assets on the risk of possible threats and attacks that may arise at runtime. Our simulations demonstrated that risk assessment and utility evaluation are plausible: risk increases when assets' values increase and, in those cases, stronger countermeasures are applied. We investigated the effect of removing assets and adding countermeasures in our experiments. Our proposed reasoning mechanism also updated the network quickly during simulations.

The effectiveness of FCN depends on the completeness of the three underlying models, which in turn relies on the quality of security requirements and risk assessment. Usability of the approach needs to be investigated in an empirical study, but our approach mainly adds asset modeling, linking assets to the other two models, and tuning the FCN to a common security requirements engineering process. Currently, these tasks are human-intensive, but automating building and tuning the FCN seems viable. We used a global search method to pinpoint the optimal security configuration, but for better scalability we need to find a nearly optimal solution using a search algorithm with possible heuristics.

To extend our work, we can consider other adaptation triggers, such as changing vulnerabilities. We also plan to try other fuzzy reasoning approaches and consider fuzzy labels in our FCN. For example, in risk calculation both Maximum and Average functions ignore some information from partial risks that may change the total risk. A possible solution is to add up partial risks and map the outcome to the range $[0, 1]$ using scalable monotonic chaining [25].

ACKNOWLEDGMENTS

The authors kindly thank Dr. Luca Cavallaro for his valuable feedback.This work was supported, in part, by Science Foundation Ireland grant 10/CE/I1855 to Lero. We also acknowledge the ERC and UTRC for their financial support.

REFERENCES

[1] C. Haley, R. Laney, J. D. Moffett, and B. Nuseibeh, "Security requirements engineering: A framework for representation and analysis," *IEEE Trans. on Software Eng.*, vol. 34, pp. 133–153, 2008.

[2] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[3] R. Howard and J. Matheson, "Influence diagrams," *Decision Analysis*, vol. 2, no. 3, pp. 127–143, 2005.

[4] B. Kosko, "Fuzzy cognitive maps," *Int. Journal of Man-Machine Studies*, vol. 24, no. 1, pp. 65–75, 1986.

[5] A. van Lamsweerde, "Elaborating security requirements by construction of intentional anti-models," in *Proc. of the Int. Conf. on Software Eng.*, 2004, pp. 148–157.

[6] G. Elahi, E. Yu, and N. Zannone, "A vulnerability-centric requirements engineering framework: analyzing security attacks, countermeasures, and requirements based on vulnerabilities," *Requir. Eng.*, vol. 15, pp. 41–62, March 2010.

[7] A. P. Felt, M. Finifter, E. Chin, S. Hanna, and D. Wagner, "Survey of mobile malware in the wild," in *CCS Workshop on Security and Privacy in Mobile Devices*, 2011, pp. 3–14.

[8] A. van Lamsweerde, *Requirements Engineering: From System Goals to UML Models to Software Specifications*. John Wiley and Sons, 2009.

[9] G. Stoneburner, A. Goguen, and A. Feringa, "Risk management guide for information technology systems," *NIST special publication*, vol. 800, p. 30, 2002.

[10] F. Jensen and T. Nielsen, *Bayesian networks and decision graphs*. Springer Verlag, 2007.

[11] G. Cybenko, "Why johnny can't evaluate security risk," *IEEE Security & Privacy*, vol. 4, p. 5, January 2006.

[12] G. Shafer, *A mathematical theory of evidence*. Princeton Univ. press, 1976, vol. 1.

[13] R. A. Howard and J. E. Matheson, *Readings in Decision Analysis*. Strategic Decisions Group, 1981, ch. Influence diagrams, pp. 763–771.

[14] M. Fossi *et al.*, "Symantec report on the underground economy," November 2008, Symantec.

[15] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM Trans. on Autonomous and Autonomic Systems*, vol. 4, no. 2, pp. 1–42, May 2009.

[16] A. Ghosh, J. Wanken, and F. Charron, "Detecting anomalous and unknown intrusions against programs," in *Proc. of IEEE Computer Security Applications Conf.*, 1998, pp. 259–267.

[17] R. Crook, L. L. Darrel C. Ince, and B. Nuseibeh, "Security requirements engineering: When anti-requirements hit the fan," in *Proc. of the Int. Conf. on Requirements Eng.*, 2002, pp. 203–205.

[18] S. Fickas and M. S. Feather, "Requirements monitoring in dynamic environments," in *Proc. of the Int. Symp. on Requirements Eng.*, ser. RE '95, 1995, pp. 140–147.

[19] V. E. Silva Souza, A. Lapouchnian, W. N. Robinson, and J. Mylopoulos, "Awareness requirements for adaptive systems," in *Proc. of the Int. symp. on Software Eng. for adaptive and self-managing systems*, 2011, pp. 60–69.

[20] L. Baresi, L. Pasquale, and P. Spoletini, "Fuzzy goals for requirements-driven adaptation," in *Proc. of Int. Requirements Eng. Conference (RE)*, 2010, pp. 125–134.

[21] M. Salehie and L. Tahvildari, "Towards a goal-driven approach to action selection in self-adaptive software," *Software: Practice & Experience*, vol. 42, no. 2, pp. 211–233, 2012.

[22] M. Sahinoglu, "Security meter: A practical decision-tree model to quantify risk," *Security & Privacy, IEEE*, vol. 3, no. 3, pp. 18–24, 2005.

[23] P. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger, "Fuzzy multi-level security: An experiment on quantified risk-adaptive access control," in *Proc. of the IEEE Symp. on Security and Privacy*, 2007, pp. 222–230.

[24] M. Covington, W. Long, S. Srinivasan, A. Dev, M. Ahamad, and G. Abowd, "Securing context-aware applications using environment roles," in *Proc. of the ACM symp. on Access control models and technologies*, 2001, pp. 10–20.

[25] E. Cox, *The fuzzy systems handbook: a practitioner's guide to building, using, and maintaining fuzzy systems*. Academic Press Professional, Inc., 1994.