2009 International Conference on Computational Science and Engineering

Success Likelihood of Ongoing Attacks for Intrusion Detection and Response Systems

Wael Kanoun*†, Nora Cuppens-Boulahia†, Frédéric Cuppens†, Samuel Dubus* and Antony Martin*
†Telecom Bretagne, 35576 Cesson Sévigné, France
*Bell Labs, Alcatel-Lucent, 91620 Nozay, France

***Abstract*—Intrusion Detection and Response Systems have become a core component in modern security architectures. Current researches are combining intrusion detection and response systems with risk analysis or cost-sensitive approaches to enhance the detection and the response procedure, by assessing the risk of detected attacks and candidate countermeasures. The Risk has two primary dimensions: (i) the likelihood of success of the attack(s), and (ii) the impact of the attack(s) and the countermeasure(s). In this paper, we present a model to assess the success likelihood of attack objectives. This model can be used by intrusion detection and response systems to identify candidate ongoing scenarios, calculate dynamically the likelihood of success for each of them considering the progress of the attack and the state of the target system, and finally prioritize candidate intrusion objectives and associated countermeasures.**

***Index Terms*—Success likelihood, intrusion objective, dynamic Markov model, intrusion detection and response systems.**

I. Introduction

Modern security architectures consider intrusion detection and response systems as a core component. Traditional Intrusion Detection Systems (IDSs) focus on low-level attacks or anomalies to generate alerts. In the signature based approach, IDSs can not detect zero-day attacks. In the anomaly based approach, IDSs suffer from high ratio of false positives.

On the other hand, Intrusion Prevention Systems (IPSs) and Intrusion Response Systems are highly used along with the IDSs to counter the detected threats. However, current intrusion prevention devices act only as conventional firewalls with the ability to block, terminate or redirect the traffic when the corresponding intrusion event is triggered. In other words, the intrusion response is statically associated with one (or several) intrusion event(s). As networks continue to grow in size and complexity, advanced and intelligent response systems are needed to counter the detected ongoing attacks.

Several intrusion response systems which are cost (i.e. impact) aware or cost-benefit aware, have been proposed recently. Toth and Kruegel [1] proposed a cost sensitive approach that balances between intrusion damage and response cost in order to choose a response with the least impact. Lee *et al.* [2] also discuss the need to consider the cost of intrusions damage, the cost of manual and automated response to an intrusion, and the operational cost, which measures constraints on time and computing resources. Similar approaches are also proposed in [3] and [4].

In general, we consider that security, as defined in [5], is "the protection of information systems and services against disasters, mistakes and manipulation such that the likelihood and impact of security incidents are minimized". Moreover, a general framework for advanced reaction systems based on risk analysis approach is defined in [6]; where likelihood and impact are combined to calculate the risk of detected attacks.

While previous intrusion response systems consider the cost (or impact) of the detected attacks to prioritize and launch response actions and countermeasures, we adopt a different yet complementary approach which considers the success likelihood of the detected attacks. In this paper, we propose to calculate a new metric that represents the success likelihood of ongoing attack(s), or more precisely the success likelihood of the intrusion objective(s). For simplicity, we consider in this paper that all the detected attacks have equal impacts on the target system. We define the 'Success Likelihood' as a relative logarithmic metric derived from the time needed to accomplish the ongoing attack. This metric indicates how close the attacker is to achieve his objective(s). Using this relative metric, our model is able to prioritize the attacks to handle, and thus prioritize the countermeasures to launch. A total defensive-centric view is adopted; in other words we do not aim to find the most likely intrusion objective sought by the attacker. In fact, "85% of breaches were the result of opportunist attacker" [7]. With the analogy of a king in his castle, the response system will be able to calculate the success likelihood of breach for each of the gates and prioritize the countermeasures to backup these gates, independently of the attacker's intentions that might be even unclear for the attacker himself. The success likelihood assessment is based on the analysis of an attack graph generated by the IDS framework. The attack graph is transformed to dynamic Markov Models that consider the progress of the ongoing attack(s) and the evolution of the target system state. Markov Model has been chosen because it adds to the attack graphs a 'temporal' dimension, which is needed to calculate the success likelihood. This is exactly the same principle used in cryptography: greater the time needed to decipher an encrypted message, lesser is the success likelihood to obtain the plain message.

This paper is organized as follows. Section II proposes attack and response modeling approaches. In Section III, we present a model to assess in real-time the success likelihood of ongoing attacks for intrusion detection and response systems,

978-0-7695-3823-5/09 $26.00 © 2009 IEEE
DOI 10.1109/CSE.2009.233

using dynamic Markov models. In Section IV, a VoIP use case with numerical results is presented to illustrate our model. Section V discusses existing and related work in the literature. Finally, Section VI concludes the paper, and provides a discussion for future work.

II. Attack and Response Modeling

Administrators need a fine and efficient diagnostic procedure to detect and identify the intrusions. However, due to the limitation and unreliability of the intrusion detection probes like SNORT [8], only low-level events can be detected with potentially high rates of false alarms. Therefore, to detect and recognize the current attack, an alert correlation procedure is required. The correlation procedure recognizes relationships between alerts in order to associate these alerts into a more global intrusion scenario, and the intrusion objectives that violate the predefined organization security policies. There are several approaches that can be used for this purpose: *implicit*, *explicit*, and *semi-explicit* correlations.

The *implicit* approach, aims to find relations binding the generated alerts. These relations could be statistical in nature, or be based on similarities between alerts attributes. Examples of such techniques can be found in [9] and [10]. In the *explicit* approach, whole attack scenarios are defined by an expert using explicit relations between the alerts or events. This approach is static because it requires an exhaustive definition for all the known attacks, and is not adapted to the non-automated ones ([11], [12]). The *semi-explicit* approach ([13], [14]) is based on the description of the pre and post conditions representing the prerequisites and the effects of the elementary intrusions corresponding to the alerts. This approach then finds causal relationships between these elementary alerts and connects these elementary alerts when such a relationship exists. The correlation procedure then consists in building a scenario that corresponds to an attack graph of steps corresponding to the elementary intrusions. The semi-explicit approach is generic and flexible because only the elementary steps are defined as entities and not the whole attacks scenario. Due to its flexibility, we will only consider the last approach using LAMBDA Language [15] in the remainder of this paper; even though our success likelihood assessment model can be used with any other pre/postcondition based language.

The objective of Section III is to present a model to assess the success likelihood of ongoing attack for intrusion detection and response systems, using a pre/postcondition attack language (e.g. LAMBDA). The model takes in consideration the real-time evolution of the attack and the information system. Therefore links "weights" between attack steps (i.e. nodes of the graph) have dynamic values; and are re-calculated for each evolution of the attack progress or the system state. An evolution could be the result of a new executed and detected attack step, or a modification in one of the future steps precondition of an attack step. For each intrusion objective of ongoing attacks, we propose to calculate a new metric that we call 'Success Likelihood' using dynamic Markov Models. Finally, the calculated metrics allow the response systems to prioritize the objectives of ongoing attacks, and therefore the response activation procedures.

We will first introduce briefly the LAMBDA language and the semi-explicit correlation. Then we present how candidate reactions are identified to counter the detected attacks.

A. LAMBDA Language and Semi-Explicit Correlation

We present below a short description of the LAMBDA model used to describe elementary attack steps. For a formal description, interested readers can refer to [15] and [16]:

- Pre-conditions: this field describes the information system state required so that the attacker is able to perform the step. It contains one or several logical predicates.
- Post-conditions: this field describes the information system state after the execution of the step. It contains one or several logical predicates.
- SK_Level: this field indicates the minimum level of skill and/or internal knowledge required to execute the step successfully. In this paper we consider that $0 < SK\_level < 1$, and that step *A* is "easier" than *B* if $SK\_Level_A > SK\_Level_B$. In the remainder of this paper, we denote SK_Level simply by SK.
- Detection: this field is used for the mapping of a LAMBDA model to the appropriate alert.
- Verification: this field can be used to verify if a step is successfully executed.

In this paper, we will need mostly the first three fields (i.e. pre-conditions, post-conditions and SK_Level); examples are shown in Figure 1.

Semi-Explicit Correlation: We say that two LAMBDA models *A* and *B* are correlated if the post-condition of *A* matches the pre-condition of *B*. The LAMBDA [15] language can be used to describe these elementary steps by defining their pre-conditions and post-conditions. Regarding response, it is also the most interesting because it provides a precise diagnosis of the ongoing intrusion scenario by constructing the attack graph; and predicts the potential future steps and the intrusion objectives. Using this approach, we can instantiate an attack graph representing the detected steps of the attack, and the potential future steps that may lead to several intrusion objectives. An example is shown in Figure 1: *sip_user_discovery* is correlated with *sip_malformed_packet* by matching the two predicates *is_on(H2)* and *Knows(A,useraccess(Siptext1,H1,udp,user)*.

B. Response Approaches

Intrusion Prevention Systems (IPSs) are highly used along with the IDSs to counter the detected threats. However, current intrusion prevention devices act only as conventional firewalls with the ability to block, terminate or redirect the traffic when the corresponding intrusion event is triggered. In other words, the intrusion response is statically associated with one (or several) intrusion event(s). Nevertheless, in [17] where a contextual security policy have been defined, a policy reaction formalism was introduced. This reaction is performed globally allowing a modification of the global access control

Fig. 1. Example of the semi-correlation procedure.

policy in an organization. The threat context mechanism was implemented as a set of contextual rules that are triggered when the corresponding threat contexts become active.

On the other hand, the anti-correlation approach [18] allows an easiest manner to express the response activation along with the scalability consideration. Now we introduce the anti-correlation approach with a short description:

Anti-Correlation: A countermeasure *C* is anti-correlated with an attack *A* if the post-condition of *C* matches the pre-condition negation of *A*. The anti-correlation [18] approach is based upon finding the appropriate countermeasure that turn an elementary future step of an attack unexecutable due to pre-conditions value modifications. Therefore, the response system can identify, from a predefined library, the countermeasures which are capable of blocking an ongoing attack. An example is shown in Figure 2: the countermeasure *drop_sip_traffic* is able to block the attack *sip_user_discovery* by turning the precondition predicate *network_access(A,H2)* to *false*.

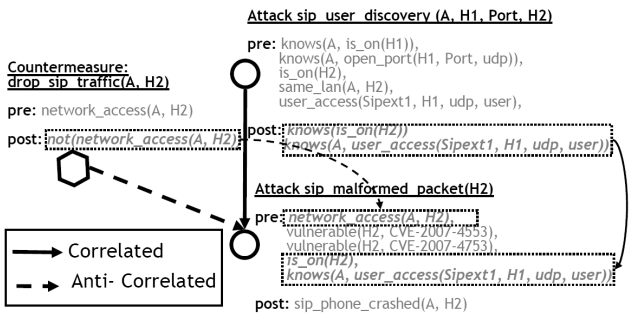

Fig. 2. Example of the correlation and anti-correlation procedures.

We note that the two response approaches (i.e. contextual security policy and anti-correlation) are compatible with our response model based on success likelihood assessment for ongoing attacks. Due to space limitation, we consider only the latter response approach in the remainder of this paper.

III. Success Likelihood Assessment Model for Ongoing Attacks

Using LAMBDA language, an attack graph can be instantiated. A node in this graph represents an elementary attack that has been executed successfully, or a potential step that could be executed in the future (i.e. not executed yet). These nodes lead to Intrusion Objectives, which constitute the terminal nodes in the attack graph. For each evolution of the attack progress or the system state, a new attack graph is instantiated. This can be due to a new executed attack step; in other words a future step in the previous graph turns to an executed step in the new instantiated graph if the appropriate alert(s) have been raised. Additionally, a new attack graph can be also instantiated if a predicate state of a future step has changed (e.g. from *true* to *false* or vice versa); which can make the concerned steps more or less executable. Therefore, the model will be re-applied for each instance of the attack graph. We can resume the procedure of the model to the following phases:

1) Decompose the attack graph into several disjoint subgraphs (i.e. one subgraph for each Intrusion objective);
2) Transform each subgraph into a dynamic Markov Model;
3) Calculate the success likelihood metric, prioritize intrusion objectives and candidate response activation procedures.

A. Decomposing the Attack Graph

The objective of the first step is to decompose the generated attack graph into several subgraphs. Each subgraph is associated to an intrusion objective. In other words, if an attack graph has n intrusion objectives (i.e. terminal nodes), the attack graph will be decomposed into n subgraphs. A given subgraph contains all the future (i.e. non executed yet) nodes that lead to the associated intrusion objective, and also contains the already executed steps adjacent to the future steps. Figure 3 is an example of an attack graph with three intrusion objectives, thus decomposed into three subgraphs. Each subgraph contains only the future (i.e. non-executed) steps, and the already executed steps adjacent to the future ones.

B. Transforming a Subgraph into a Markov Model

The objective of this phase is to transform each subgraph to a Continuous Markov Model. The Markov Model is based on the assumption that the probability to succeed executing an elementary attack in a given time before time t is described by an exponential distribution given by:

$$P(t) = 1 - e^{-\lambda t} \tag{1}$$

Therefore, provided that there is a path leading to an intrusion objective, we consider that the attacker will eventually succeed in reaching this intrusion objective if he spends enough time (that can tend to infinity).

Now, we need to calculate the transition probabilities p_{ij} between state i and state j in the subgraph. We consider that the attacker would not pass from state i to state j if state j have

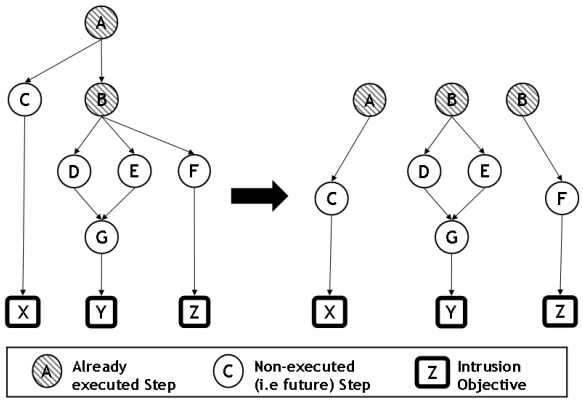

Fig. 3. Decomposition of an attack graph instance to subgraphs.

been already executed. Moreover, we consider that $p_{ij} = 0$ if state i and state j are not correlated (i.e. there are not linked in the graph). Let us denote:

- *Unified(i,j)* : number of predicates that are unified between *Post(i)* and *Pre(j)*
- *NbrPre(j)* : number of predicates in *Pre(j)*
- $W_{ij} = \frac{Unified(i,j)}{NbrPre(j)}$: correlation weight between two steps i and j
- H_{ij} Number of predicates in *Pre(j)* that are *false* and not correlated with previous step i
- *Future(i)* : Set of future steps that are correlated with the step i

Therefore we can compute p_{ij} as follows:

$$p_{ij} = (1 - \varepsilon)m_{ij} + \varepsilon.n_{ij} \tag{2}$$

The first part (i.e. m_{ij}) expresses the *weight* of the correlation link between the steps i and j, thus the coherence of executed steps with the future steps. In other words, higher m_{ij} means that the previous step i leads more significantly to step j because the number of predicates that match between these two steps is higher. Therefore, higher m_{ij} means higher probability to transit from step i to step j. The formula of m_{ij}:

$$m_{ij} = \frac{W_{ij}}{\sum_{k \in Future(i)} W_{ik}} \tag{3}$$

The second part (i.e. n_{ij}) expresses the easiness to transit from step i to step j. It is clear that the probability to transit from step i to step j (i.e. p_{ij}) is higher if future step j is easier than other future steps, considering that all the future steps in a subgraph leads to the same intrusion objective. The difficulty (or the easiness) of a future step includes the skill and knowledge level required to execute this step, and the number of precondition predicates that are *false*. In fact, the attacker has an 'extra work' to consider, because he needs to transform the *false* precondition predicates into *true* before being able to execute the future step j. The formula of n_{ij}:

$$n_{ij} = \frac{SK_j + \left(1 - \frac{H_{ij}}{NbrPre(j)}\right)}{\sum_{k \in Future(i)} \left[SK_k + \left(1 - \frac{H_{ik}}{NbrPre(k)}\right)\right]} \tag{4}$$

In equation 2, ε determines how much each part impacts the overall p_{ij} expression. In simulations we consider the two parts have the same potential impact, thus we set $\varepsilon = \frac{1}{2}$.

For a subgraph with n elementary attack steps (i.e. nodes or steps), we denote the transition probabilities matrix by P that contains p_{ij}:

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ p_{n1} & \cdots & & p_{nn} \end{bmatrix} \tag{5}$$

Every step in the attack graph presents an elementary attack. Certainly, some elementary attack (i.e. step) are more 'difficult' and time-consuming than others. It is obvious that a step with low SK_Level needs more time to be successfully executed. In addition, the attacker needs more time if some of the precondition predicates of a given step are *false*. This can be explained by the fact that the attacker has to invest more time to turn all the precondition predicates of a given step to *true*, before being able to execute this step. Therefore, we calculate the exit rates matrix Λ. Λ is a diagonal matrix where λ_{ii} (or simply λ_i) denotes the exit rate from the state i. Higher λ_i means the attacker has to spend lesser time to achieve elementary attack i. For the already executed steps, we assign a constant value for λ_i (e.g. 10 in this paper). Otherwise, we consider that for a future (i.e. non-executed) step $0 < \lambda_i < 1$, and have to be calculated:

$$\lambda_i = \frac{1}{2} \times \left[SK_j + \left(1 - \frac{H_{ij}}{NbrPre(j)}\right)\right] \tag{6}$$

For a subgraph with n elementary steps (i.e. nodes), we denote the exit rates matrix by Λ that contains λ_i:

$$\Lambda = \begin{bmatrix} \lambda_{11} & 0 & \cdots & 0 \\ 0 & \lambda_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & & \lambda_{nn} \end{bmatrix} \tag{7}$$

Once P and Λ have been calculated with equations (5) and (7) , the infinitesimal generator $n \times n$ matrix for of the Markov Model of this subgraph A can be calculated:

$$A = -\Lambda \times (I - P); \textit{ where I is the identity matrix of size n} \tag{8}$$

The Markov Model associated to a given subgraph can be totally defined with the infinitesimal generator matrix A. In

our approach, the n^{th} row of A corresponding to the intrusion objective (i.e. terminal node in the subgraph) will contain only *zeros*; which can be explained by the fact that the intrusion objective step is an absorbing state in the Markov Model. We denote by A_u the submatrix of A with the first $(n-1)$ rows and $(n-1)$ columns:

$$A_u = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1(n-1)} \\ a_{21} & a_{22} & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{(n-1)1} & \cdots & & a_{(n-1)(n-1)} \end{bmatrix} \quad (9)$$

Having only one absorbing state in each Markov Model associated with a given subgraph, the Mean Time to Intrusion Objective (MTIO) can be calculated. MTIO provides a convenient metric to assess the success likelihood of each intrusion objective in the attack graph. As we said before, our approach is totally defensive because it does not depend on the attackers goals; but dynamic because at each attack graph or system evolution, P, Λ and therefore A will be recalculated. For two intrusion objectives X and Y, $MTIO_X < MTIO_Y$ means that the success likelihood of intrusion objective X is higher than the one of Y, and thus intrusion objective X is easier to achieve and more vulnerable than Y. The MTIO of an intrusion objective X can now be calculated:

$$MTIO_X = S \times A_u \times [1\ 1\ \cdots\ 1]^t \quad (10)$$

Where S is a $1 \times (n-1)$ vector that depicts the current state of the subgraph Markov Model. The k^{th} element of vector S equals to *zero* if it corresponds to a non-executed step. Otherwise, $ts_k = \frac{1}{\#executed\ steps}$. For example if the subgraph contains one executed step, then $S = [1\ 0\ 0\ 0\ \cdots\ 0]$; if the subgraph contains two executed steps, then $S = [\frac{1}{2}\ \frac{1}{2}\ 0\ 0\ \cdots\ 0]$; if the subgraph contains three executed steps, then $S = [\frac{1}{3}\ \frac{1}{3}\ \frac{1}{3}\ 0\ \cdots\ 0]$; etc.

Considering our hypothesis that $0 < \lambda_i < 1$, it is possible to demonstrate that for any attack that has not reached yet an intrusion objective X, the $MTIO_X$ is always higher than 1 (i.e. $MTIO_X > 1$).

C. Success Likelihood Assessment and Response Procedure

Finally, for each candidate intrusion objective X, we calculate the success likelihood metric. We propose an empirical logarithmic formula, similar to the one used to express the magnitude of a physical quantity (current, voltage, power, etc.), to transform the temporal metric MTIO into a success likelihood relative logarithmic metric. The success likelihood depicts the variations of the MTIO metric, especially when MTIO tends to *zero*. Therefore, we propose to calculate the success likelihood as follows:

$$\begin{aligned} L_X &= f(MTIO_X) \\ &= -20 \times \log_{10}\left(\frac{\Delta MTIO_X}{MTIO_X}\right) \\ &= -20 \times \log_{10}\left(\frac{MTIO_X - MTIO_{min}}{MTIO_X}\right) \\ L_X &= -20 \times \log_{10}\left(\frac{MTIO_X - 1}{MTIO_X}\right) \end{aligned} \quad (11)$$

The success likelihood $L_X = f(MTIO_X)$ of an intrusion objective X grows rapidly if $MTIO_X$ decreases, and $L_X \to 0$ if $MTIO_X \to \infty$. Thus, if the attacker is closer to intrusion objective X, L_X grows faster and faster. Ultimately, if the attacker achieves the intrusion objective, we will have $L_X \to \infty$. The Figure 4 shows the plot of $L_X = f(MTIO_X)$.

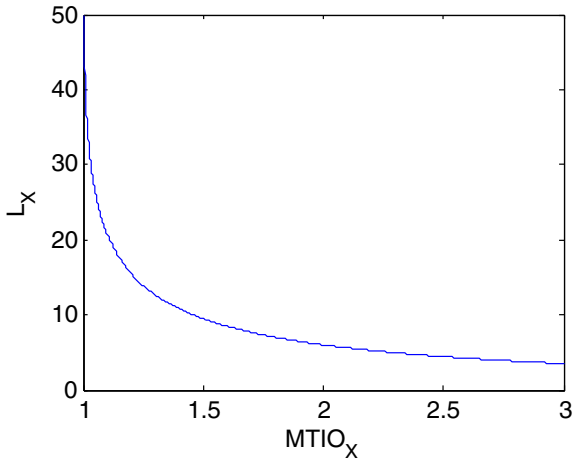

Fig. 4. Plot of $L_X = f(MTIO_X)$.

Finally, once the success likelihood for each intrusion objective has been calculated, the Response system is able to prioritize them; thus activate the response actions to block first the ongoing attacks that have the highest success likelihood. For example, if three intrusion objectives X, Y and Z have $L_X < L_Y < L_Z$, the response system must launch first the countermeasure CM_Z that blocks the intrusion objective Z (most urgent), then CM_Y, and finally CM_X (least urgent).

IV. VoIP Use Case

The case study is a SIP-based VoIP enterprise environment of an organization (See Figure 5). The basic principle of the VoIP use case is to offer to the users several access methods. In our testbed users can use both a hardphone and a softphone installed on their laptop, like typical VoIP deployments that provide flexibility and mobility. The VoIP service is composed of a SIP server on a dedicated network, which acts as a SIP registrar and a SIP router/proxy for MD5 authentication and call routing. OpenSER [19] is used as a SIP server, while the SIP MD5 authentication is delegated to a collocated RADIUS server, based on FreeRADIUS [20]. The test-bed is also

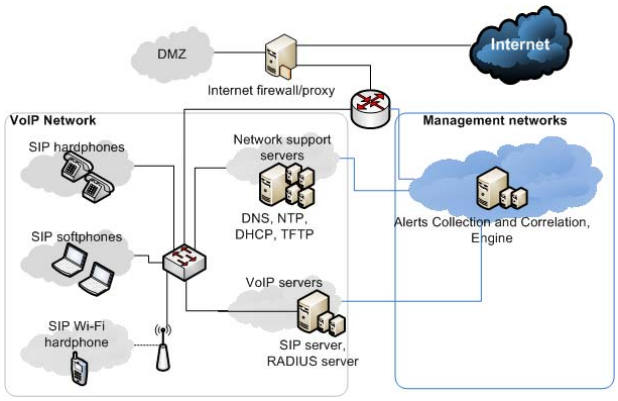

Fig. 5. The VoIP testbed.

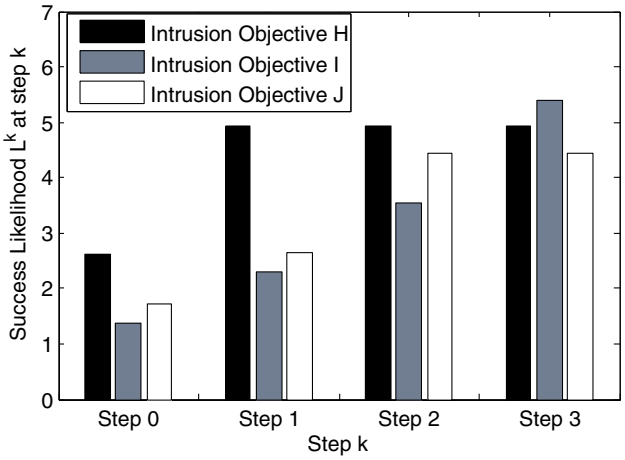

Fig. 7. Evolution of Success Likelihood L^k for the three candidate intrusion objectives.

composed of four SIP User Agents (UA) networks, first for softphones (i.e. X-Lite [21], S-JPhone [22] and Linphone [23]) and second for hardphones (i.e. Thomson, Linksys, Zyxel) which have been divided into wired or wireless networks. The testbed intrusion infrastructure relies on a cut-off Snort IDS [8]. Regarding the detection part of the test-bed, intrusion detection probes generate alerts which are collected by the *Alerts Collection and Correlation Engine*, in particular for the correlation process. We use the prototype CRIM [24] that adopts the semi-explicit correlation approach, and generates a pre/postcondition graph using the LAMBDA language. Finally a MATLAB-based module calculates the success likelihood of each intrusion objective in the attack graph, and prioritizes the candidate countermeasures.

To support demonstration of our work, we have implemented a set of elementary attacks and a complex attack scenario. Both SIP related attacks based on flaws in the protocol design [25] and flaws in software implementation have been identified and implemented on the VoIP test-bed. Considering the combination of a set of SIP hacking techniques (i.e. scanning, cracking, etc.) and SIP related attacks, we designed a complex attack scenario that enables an attacker to perform a denial of service on the SIP Server, on a legitimate user, or on a SIP phone. Moreover, a reaction for each intrusion objective has been defined. Therefore, the attack graph generated in LAMBDA language contains three intrusion objectives with three associated reactions. For simplicity, the number of LAMBDA models (i.e. elementary attack steps and intrusion objectives) used in the attack graph is restricted to seven and three countermeasures; the attack graph in its initial state is presented in Figure 6. The predicates in italic are the correlated predicates.

a) Step 0 of the ongoing attack: The attacker has not executed any attack yet. Having three intrusion objectives in the attack graph, the first phase is to decompose this attack graph into three subgraphs. The subgraph of intrusion objective H, I and J contains respectively four, seven and five nodes (including *start*). The second phase of our methodology is applied for each step to calculate the success likelihood for each intrusion objective:

$$L_H^0 = 2.6144; L_I^0 = 1.3788; L_J^0 = 1.7161$$

b) Step 1 of the ongoing attack: The attacker launches a SIP server discovery attack as a first step of his attack. Therefore the elementary attack A in the attack graph has been executed. This step affects the success likelihood of the three candidate intrusion objectives H, I and J. The subgraph of intrusion objective H, I and J contains respectively three, six and four nodes. We note that the predicate network_access(A,H1) is now *true*; and the same predicate exists in the precondition fields of the future attacks D and G. Thus, the transition probabilities from Attack C to Attacks D and E will have new values. Moreover, the exit transition rate of Attack D and Attack G will increase because their precondition predicate is *true*. At step 1 we have:

$$L_H^1 = 4.9301; L_I^1 = 2.3134; L_J^1 = 2.6603$$

Therefore, the highest priority is for countermeasure 1, then after for countermeasure 3, and countermeasure 2 has the least priority.

c) Step 2 of the ongoing attack: The attacker proceeds in his attack and launches an Active User Discovery attack. Therefore, step C in the attack graph has been executed. This step affects the success likelihood for the candidate intrusion objectives I and J, but not for H. The subgraph of intrusion objective H, I and J contains respectively three, five and three nodes. At step 2 we have:

$$L_H^2 = 4.9301; L_I^2 = 3.5417; L_J^2 = 4.4347$$

At step 2, the highest priority is for countermeasure 1, then after for countermeasure 3, and countermeasure 2 has the least priority.

d) Step 3 of the ongoing attack: At step 3, the attacker launches an Active SIP Authentication Crack attack. Therefore, step D in the attack graph has been executed. This step affects the success likelihood for the candidate intrusion

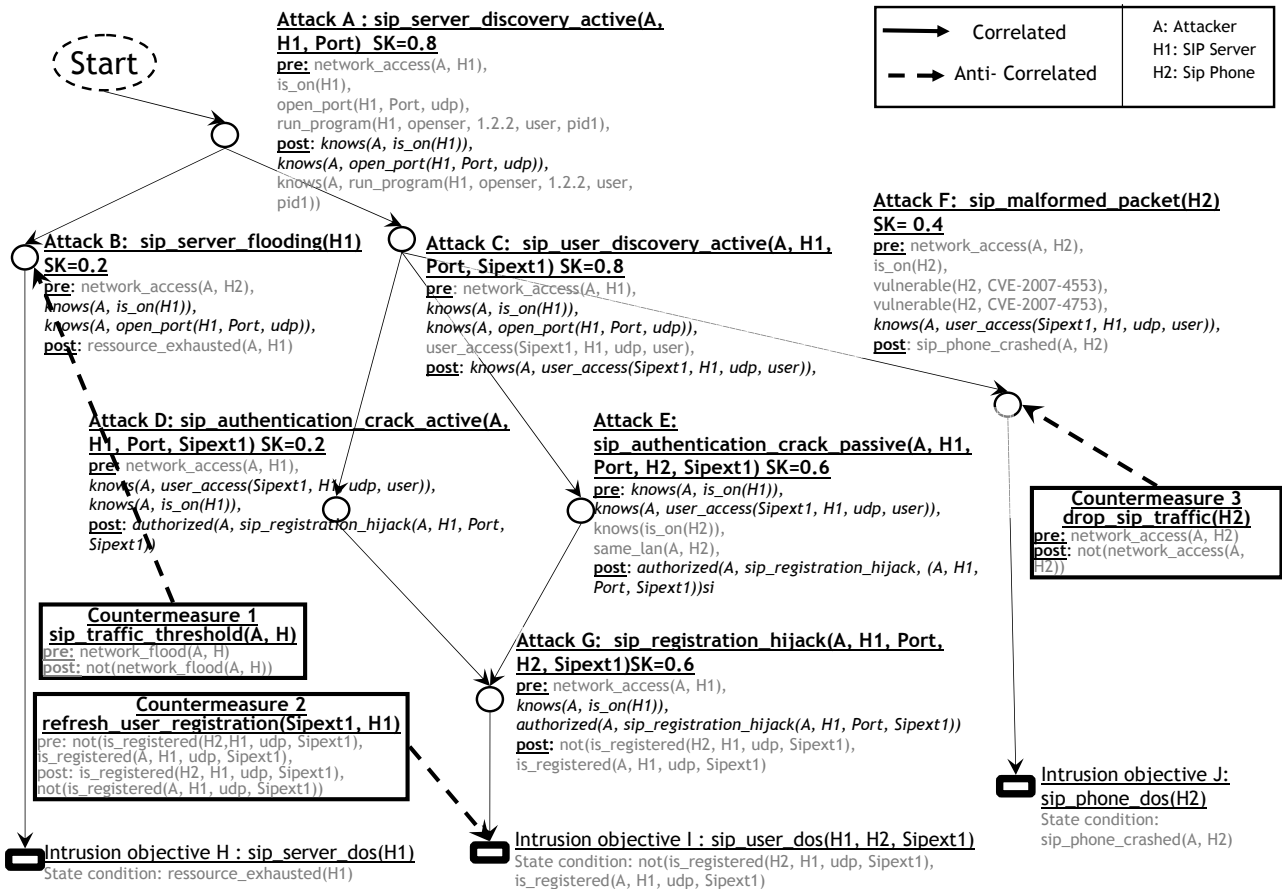

Fig. 6. The VoIP use case complex attack scenarios.

objective I, but not for H or J. The subgraph of intrusion objective H, I and J contains respectively three, five (which two of them are already executed) and three nodes. At step 3 we have:

$$L_H^3 = 4.9301; L_I^3 = 5.3996; L_J^3 = 4.4347$$

Therefore, the highest priority is for countermeasure 2, then after for countermeasure 1, and countermeasure 3 has the least priority. In fact, the success likelihood of intrusion objective I is at this stage of the attack higher than for the other two intrusion objectives (i.e. H and J).

Figure 7 shows the evolution of the success likelihood of each intrusion objective, with respect to the attack progress. We can notice that after the first step, the success likelihood of intrusion objective H increased significantly, but remained afterward stable ($L_H^k = 4.9301$ for $k \geq 1$). This is explained by the fact that only one elementary attack is needed to achieve H, but the attacker proceeded in another path. The success likelihood of the intrusion objective J stabilized after second step ($L_J^k = 4.4347$ for $k \geq 2$). As for intrusion objective H, the attacker needs only to execute one more step to achieve J. We notice also that $L_H^k > L_J^k$ for $k \geq 2$, this is explained by the fact that the remaining step (i.e. Attack F) to achieve J, is more difficult and need more preparations than of the remaining step to achieve H (i.e. Attack B). For intrusion objective I, it increases with each step executed by the attacker. This is totally normal because all the executed attacks (A, C and D) lead to the intrusion objective I.

V. Related Work

Recently, several intelligent intrusion response have been proposed. Toth and Kruegel [1] propose a cost sensitive approach that balances between intrusion damage and response cost in order to choose a response with the least impact. Lee *et al.* [2] also discuss the need to consider the cost of intrusions damage, the cost of manual and automated response to an intrusion, and the operational cost, which measures constraints on time and computing resources. Similar approaches are also proposed in [3] and [4]. While all the previous approaches consider only the cost (i.e. impact factor), we see our work complementary because our approach considers the success likelihood of ongoing attacks. In [6], a general framework for advanced reaction systems based on risk analysis approach is presented; where likelihood and impact are combined to calculate the risk of ongoing attacks. However, this paper did not propose how to calculate the success likelihood of the

attack.

On the other hand, there have been several papers in the literature that discuss the generation and the analysis of attack graph. Cuppens *et al.* presented in [13] and [14] the "semi-explicit" approach to correlate elementary attacks described using LAMBDA language [15] . Moreover, in [26] and [27], Ning *et al.* combined complementary types of alert correlation methods: (i) those based on the similarity between alert attributes; and (ii) those based on prerequisites and consequences of attacks. The work is very close to Cuppens and Miege work in the context of MIRADOR project, which has been done independently and in parallel. In [28], Sheyner *et al.* used a model of exploits (possible attacks) in terms of their preconditions and postconditions to construct possible sequences of attacks. However, our method aims to construct high-level attack scenarios from low-level intrusion alerts and reason about attacks possibly missed by the IDSs, while this kind of vulnerability analysis techniques are focused on analyzing what attacks *may happen* to a given system. In contrast, our purpose is to construct what *is happening* to a given information system according to the alerts reported by IDSs.

Dacier *et al.* suggested the use of "privilege graphs" to analyze security [29]. Privilege graphs require modeling of vulnerabilities at a very low level, and for a nontrivial sized system, would involve a graph of unmanageable size. First, Dacier observed that for some kinds of attacks, security increases as the time required for the success of the attack increases. However, it is not known how an increase in the time-to-compromise impacts an attacker; therefore this is not a precise measurement of Security Risk. However, we believe that as the time-to-compromise is increased, the likelihood of successful attack, and therefore Security Risk, tends to decrease. Second, by using Privilege Graphs to model the system, Dacier restricted his analysis to a specific family of attacks. Moreover, he adopted an "attack-centric" view of the world and considers the privileges from an attacker's point of view, which is not adapted for intrusion response systems. Finally, while his model could be useful to offline analysis, Dacier did not present a response approach for ongoing attacks.

Madan *et al.* in [30] proposed a general framework to assess the MTTSF (Mean Time To Security Failure) using a Markov Modeling approach. The main drawback of this framework is that it does not specify how to calculate transitions rates, neither how to model atomic attack actions and relation between these actions. Another point is that this framework does not take in consideration the dynamic nature of an ongoing attack, nor the real-time state of the target system.

In [31], McQueen *et al.* proposed to calculate the risk reduction due to installing/modifying Security measures (e.g. installing new updates, firewalls, IDS, etc.) based on the vulnerabilities in the system. In their approach, a graph model attack is composed of nodes that represent the attack stages, and of edges with time-to-compromise calculated in function of the number and types of vulnerabilities. We must draw attention that this paper does not discuss intrusion response systems. Furthermore, the paper did not take into consideration the real-time nature of the system and the fact that some of the vulnerabilities are not exploitable; that is totally normal because the paper aims to present an offline analysis model.

VI. Conclusion

In this paper, a real-time assessment model of the success likelihood for the ongoing attacks has been presented. This model takes in consideration the state of the attack progress and the target system state. The success likelihood metric calculated in real time can be useful to the administrators, and helps them to prioritize and handle the ongoing attacks. Our model can also offer valuable input for intelligent and automated response systems which can be risk-aware and cost-sensitive. Finally, the proposed model has been successfully validated in a VoIP use case using elementary attack LAMBDA models. In the future, we will first study the effect of launched countermeasures on the attack graph and the information system, and consequently on the success likelihood of intrusion objectives. The effectiveness of our model to chose the best countermeasure will be explored by combining the Likelihood and the Impact (thus calculating the Risk) of a given attack. Second, we will explore the use of Hidden Markov Models in the success likelihood assessment model, to take in consideration the potential uncertainty of the attack progress and the target system state.

References

[1] T. Toth and C. Kruegel, "Evaluating the impact of automated intrusion response mechanisms," in *18th Annual Computer Security Applications Conference ACSAC02*, 2002.

[2] W. Lee, W. Fan, M. Miller, S. J. Stolfo, and E. Zadok, "Toward cost-sensitive modeling for intrusion detection and response," *Journal of Computer Security*, vol. 10, no. 1/2, pp. 5–22, 2002.

[3] N. Stakhanova, S. Basu, and J. Wong, "A cost-sensitive model for preemptive intrusion response systems," May 2007, pp. 428–435.

[4] H. Wei, D. Frinke, O. Carter, and C. Ritter, "Cost-benefit analysis for network intrusion detection systems," 2001.

[5] S. Boran, "IT Security Cook Book. http://www.boran.com/security."

[6] W. Kanoun, N. Cuppens-Boulahia, and F. Cuppens, "Advanced reaction using risk assessment in intrusion detection systems," in *Second International Workshop on Critical Information Infrastructures Security (CRITIS07)*, Springer, Ed., Malaga, Spain, 2007.

[7] Verizon Risk Business Team, "2008 Data Breach Investigations Report."

[8] "Snort official website :www.snort.org."

[9] R. Lippmann, "Using key string and neural networks to reduce false alarms and detect new attacks with sniffer-based intrusion detection systems," in *Second International Workshop on the Recent Advances in Intrusion Detection (RAID'99)*, October 1999.

[10] J. Jung, V. Paxson, A. W. Berger, and H. Balakrishnan, "Fast portscan detection using sequential hypothesis testing," in *In Proceedings of the IEEE Symposium on Security and Privacy*, 2004.

[11] M. Huang, "A large-scale distributed intrusion detection framework based on attack strategy analysis," Louvain-La-Neuve, Belgium, 1998.

[12] B. Morin and H. Debar, "Correlation of intrusion symptoms: an application of chronicles," in *Proceedings of the Sixth International Symposium on the Recent Advances in Intrusion Detection (RAID'02)*, Pittsburg, USA, September 2003.

[13] F. Cuppens, F. Autrel, and A. M. et S. Benferhat, "Recognizing malicious intention in an intrusion detection process," in *Second International Conference on Hybrid Intelligent Systems*, Santiago, Chili, December 2002.

[14] F. Cuppens and A. Mige, "Alert correlation in a cooperative intrusion detection framework," *Security and Privacy, IEEE Symposium on*, vol. 0, p. 202, 2002.
[15] F. Cuppens and R. Ortalo, "Lambda: A language to model a database for detection of attacks," in *Third International Workshop on Recent Advances in Intrusion Detection (RAID'2000)*, Toulouse, France, 2000.
[16] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, and J. Araujo, "Automated reaction based on risk analysis and attackers skills in intrusion detection systems," Oct. 2008, pp. 117–124.
[17] H. Debar, Y. Thomas, F. Cuppens, and N. Cuppens-Boulahia, "Enabling automated threat response through the use of a dynamic security policy," *Journal in Computer Virology (JCV)*, vol. 3, no. 3, August 2007.
[18] *Anti-correlation as a criterion to select appropriate counter-measures in an intrusion detection framework*, January 2006, vol. 61, no. 1-2, ch. Annals of Telecommunications.
[19] "OpenSER: http://www.opensips.org/."
[20] "FreeRADIUS: http://freeradius.org/."
[21] "X-Lite: http://www.counterpath.net/X-Lite-Download.html."
[22] "SJphone: http://www.sjlabs.com/sjp.html."
[23] "Lindphone: http://www.linphone.org/."
[24] F. Autrel and F. Cuppens, *CRIM : un module de corrélation d'alertes et de réaction aux attaques*, September-October 2006, vol. 61, no. 9-10, ch. Annals of Telecommunications.
[25] D. Endler and M. Collier, *Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions (Hacking Exposed)*. McGraw-Hill Osborne Media, 2006.
[26] P. Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *ACM Conference on Computer and Communications Security*, 2002.
[27] P. Ning, D. Xu, C. G. Healey, and R. S. Amant, "Building attack scenarios through integration of complementary alert correlation methods," in *in Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS04*, 2004, pp. 97–111.
[28] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing, "Automated generation and analysis of attack graphs," 2002, pp. 273–284.
[29] M. Dacier, Y. Deswarte, and M. Kaniche, "Quantitative assessment of operational security: Models and tools," 1996.
[30] B. B. Madan, K. Vaidyanathan, and K. S. Trivedi, "A method for modeling and quantifying the security attributes of intrusion tolerant systems," *Perform. Eval*, vol. 56, pp. 167–186, 2004.
[31] M. A. McQueen, W. F. Boyer, M. A. Flynn, and G. A. Beitel, "Quantitative cyber risk reduction estimation methodology for a small scada control system," in *HICSS '06: Proceedings of the 39th Annual Hawaii International Conference on System Sciences*. Washington, DC, USA: IEEE Computer Society, 2006, p. 226.