Advanced Response System Using Success Likelihood Assessment and Analysis for Ongoing Attacks

No Author Given

No Institute Given

Abstract. Intrusion Detection and Response Systems became a core component in modern security architectures. Current researches combine 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 can be evaluated in function of the likelihood of success and the impact of the attack or the countermeasure. In this paper, a novel attack response system, based on the assessment of the likelihood of success of attack objectives, is presented. The model identifies first the candidate ongoing scenarios, then calculates dynamically the likelihood of success for each of them considering the progress of the attack and the state of the target system. Second, the effectiveness of each candidate countermeasure in reducing the pre-calculated success likelihood is assessed. Finally, the candidate countermeasures are prioritized.

1 Introduction

Modern security architectures consider intrusion detection and response systems as a core component. Traditional intrusion detection systems (IDS) focus on low-level attacks or anomalies to generate alerts. In the signature based approach, IDS can not detect zero-day attacks. In the anomaly based approach, IDS 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] 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].

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 response systems based on risk analysis approach is defined in [6]; where likelihood and impact are combined to calculate the risk of detected attacks.

While the 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 the same impact 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 will evaluate the effectiveness of each countermeasure in reducing the success likelihood for the detected attacks. Finally, the model prioritizes the countermeasures
to launch with respect to their effectiveness. For example, this can be useful in the case of several reactions that can not be activated simultaneously. Another example is where reactions have an impact on the system, and the administrator has to select among several reactions the most ‘urgent’ one(s). 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. First, 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. Second, each candidate countermeasures is simulated, and its reducing effect on the calculated success likelihood of the ongoing attacks is assessed. Finally, candidate countermeasures are prioritized to support the administrator or the response system.

This paper is organized as follows. Section 2 proposes the attack and the response modeling approaches. In Section 3, we present a response model based on a real-time assessment of the Likelihood of Success for ongoing attacks and the effectiveness of candidate countermeasures, using a dynamic Markov model. In Section 4, VoIP use case with complex attack scenarios and numerical results is presented to illustrate our model. Section 5 discusses existing and related work in the literature. Finally, Section 6 concludes the paper, and provides a discussion for future work.

2 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 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 3 is to present a Response Model based on real-time assessment of the success likelihood ongoing attacks, 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 value; 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 Model. Moreover, each candidate countermeasure will be simulated, and Success likelihood values will be re-calculated; therefore the effectiveness of each reaction countermeasure on the attack progress will be assessed and analyzed. Finally, candidate reaction countermeasures are prioritized to be activated.

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

2.1 LAMBDA Language and Semi-Explicit Correlation

We present below a short description of the LAMBDA model used to describe elementary attack step. 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.
- SKLevel: 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 < SKLevel < 1, and that step A is ‘easier’ than B if SKLevelA > SKLevelB. In the remainder of this paper, we denote SKLevel 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 SKLevel): For example in Figure 1, the elementary attack SIP_Malformed_Packet on the machine H2 can be executed successfully only if (i) the attacker A can access to H2, (ii) H2 is on and vulnerable, (iii) the attacker knows that user is registered as Siptex. Moreover, the crash of the machine H2 is the consequence of this elementary attack.

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 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 1a. If an attacker launches a sip_user_discovery, he (or she) will discover (i) that the victim has registered, and (ii) that the victim is using machine H2. Knowing that, the attacker may send malformed crafted packets to crash the victim’s machine; in other words sip_user_discovery is correlated with sip_malformed_packet by matching the two predicates is_on(H2) and Knows(A, useraccess(Siptex1, H1, udp, user)).

2.2 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 response formalism was introduced. This response is performed globally allowing a modification of the global access control 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 reaction activation along with the scalability consideration. Now we introduce the anti-correlation approach with a short description:
3 Intrusion Response System Based on Success Likelihood of 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 step 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. Phase 1: Decompose the attack graph to several sub-graphs (i.e. one subgraph for each Intrusion objective),
2. Phase 2: Transform each subgraph into a dynamic Markov Model, and calculate the Success Likelihood metrics,
3. Phase 4: Simulate each candidate countermeasure, re-calculate the Success Likelihood metric as in Phase 2,
4. Phase 4: Prioritize the candidate countermeasures.

3.1 Phase 1: Decomposing Attack Graph

The objective of the first step is to decompose the generated attack graph to 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 2 is an example of an attack graph with three intrusion objectives, thus decomposed to three subgraphs. Each subgraph contains only the future (i.e. non-executed) steps, and the already executed steps adjacent to the future ones.

![Attack Graph](image)

**Fig. 2:** Decomposition of an attack graph instance to subgraphs.

### 3.2 Phase 2: Transforming a Subgraph into a Markov Model and Assessing the Success Likelihood of Ongoing Attacks

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:

\[
P(t) = 1 - e^{-\lambda t}
\]  

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

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

Therefore we can compute \( p_{ij} \) as follows:

\[
p_{ij} = (1 - \rho)m_{ij} + \rho n_{ij}
\]  

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 \( m_{ij} \):
\[ m_{ij} = \frac{W_{ij}}{\sum_{k \in \text{Future}(i)} W_{ik}} \]  

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 \textit{false}. In fact, the attacker has an ‘extra work’ to consider, because he needs to transform the \textit{false} precondition predicates into \textit{true} before being able to execute the future step \( j \). The formula of \( n_{ij} \):

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

In equation 2, \( \rho \) 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 \( \rho = \frac{1}{2} \).

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

\[
P = \begin{bmatrix}
p_{11} & p_{12} & \cdots & p_{1n} \\
p_{21} & p_{22} & \cdots & 0 \\
0 & \ddots & \ddots & \vdots \\
p_{n1} & \cdots & 0 & p_{nn}
\end{bmatrix}
\]

Every step in the attack graph presents an elementary attack. Certainly, some elementary attack (i.e. step) are more or less ‘difficult’ and time-consuming than others. It is obvious that a step with low \( \text{SK Level} \) 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 \textit{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 \textit{true}, before being able to execute this step. Therefore, we calculate the exit rate matrix \( \Lambda \). \( \Lambda \) is a diagonal matrix where \( \lambda_{ii} \) (or simply \( \lambda_i \)) denotes the exit rate from the state \( i \). Higher \( \lambda_i \) means the attacker has to spend lesser time to achieve elementary attack \( i \). For the already executed steps, we assign a constant value for \( \lambda_i \) (e.g. 10\(^3\) in this paper). Otherwise, we consider that for a future (i.e. non-executed) step \( 0 < \lambda_i < 1 \), and have to be calculated:

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

For a subgraph with \( n \) elementary steps (i.e. nodes), we denote the exit rate matrix by \( \Lambda \) that contains \( \lambda_i \):

\[
\Lambda = \begin{bmatrix}
\lambda_{11} & 0 & \cdots & 0 \\
0 & \lambda_{22} & \cdots & \vdots \\
0 & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & \lambda_{nn}
\end{bmatrix}
\]

Once \( P \) and \( \Lambda \) 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:

\[
A = -\Lambda \times (I - P); \text{ where } I \text{ is the identity matrix of size } n
\]
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) contains 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:

$$A_u = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1(n-1)} \\ a_{21} & a_{22} & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a_{(n-1)1} & \cdots & a_{(n-1)(n-1)} \end{bmatrix}$$ (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, $P$, $A$ 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:

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

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

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:

$$L_X = f(MTIO_X)$$

$$= -20 \times \log_{10} \left( \frac{\Delta MTIO_X}{MTIO_X} \right)$$

$$= -20 \times \log_{10} \left( \frac{MTIO_X - MTIO_{min}}{MTIO_X} \right)$$

$$L_X = -20 \times \log_{10} \left( \frac{MTIO_X - 1}{MTIO_X} \right)$$ (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 3 shows the plot of $L_X = f(MTIO_X)$. 
3.3 Phase 3: Simulating Candidate Countermeasures and Re-calculating the Success Likelihood Metrics

As described in Section 2, candidate countermeasures may be identified using the anti-correlation approach to block future attack steps. As a result, these countermeasures may reduce the success likelihood of one or several intrusion objectives. Therefore, for a given instance of the attack state (i.e. attack graph), each candidate countermeasure will be simulated, and new values of the success likelihood for intrusion objectives will be calculated. Thus, the effectiveness of a given countermeasure in reducing the success likelihood of intrusion objectives can be assessed and compared to others countermeasures.

During the simulation of a given countermeasure \( C_m \), the same procedure described in Phase 2 (see Section 3.2) will be applied. The main difference is that the attack steps anti-correlated with \( C_m \) are blocked because \( C_m \) has been activated. In other words, the time needed by the attacker to execute successfully these attack steps will be very high and almost infinite. Consequently, the exit rate \( \lambda_i \) of these attack steps will not be calculated using the Equation 6. Instead, a very low value (e.g. \( 10^{-3} \)) will be assigned to the exit rate \( \lambda_i \) of attack steps blocked by countermeasure \( C_m \).

During step \( k \) in the attack progress (i.e. \( k^{th} \) instance of the attack graph), with the countermeasure \( C_m \) activated, we denote the new value of the success likelihood for the intrusion objective \( x \) by \( L_{k,v,x} \). Moreover, we denote by \( \vec{L}_{K,V} = [L_{K,V,1}, L_{K,V,2}, L_{K,V,3}, \ldots] \) the vector that contains sorted (descending order) success likelihood values of all the intrusion objectives \( x \), during the step \( k = K \) of the attack progress, while the countermeasure \( C_M \) has been activated:

\[
\vec{L}_{K,V} = [L_{K,V,1}, L_{K,V,2}, L_{K,V,3}, \ldots]
\] (12)

3.4 Phase 4: Prioritizing Candidate Countermeasures

The goal of this phase is to prioritize candidate countermeasures by their effectiveness of reducing success likelihood metrics of candidate intrusion objectives. As explained before, the model will be applied for each instance of the attack progress (i.e. attack graph). During the \( K^{th} \) attack step, and for each candidate countermeasure \( C_M \) the success likelihood vector \( \vec{L}_{K,V,x} \) that contains the success likelihood metrics of the candidate intrusion objectives is calculated.

During the \( K^{th} \) step in the attack progress, we say that countermeasure \( C_M \) has a higher priority than \( C_M' \) if \( \vec{L}_{K,V} < \vec{L}_{K,V'} \); where the \(<\) operator is a lexicographic comparison. In other words, \( C_M \) has a higher priority because it reduces more significantly the success likelihood of candidate intrusion objectives than \( C_M' \) during the \( K^{th} \) step. The descending order of \( \vec{L}_{k=K,v=V} \) ensures that the most ‘urgent’ intrusion objectives are considered first. Therefore, candidate countermeasures
can be prioritized and sent to the administrator or to the response system management module. The prioritization can be also useful to determine which countermeasure have to be launched first if there are several countermeasures that can not be activated simultaneously.

4 VoIP Use Case

The case study is a SIP-based VoIP enterprise environment of an organization (See Figure 4). 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 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.

![Fig. 4: The VoIP testbed.](image)

To support demonstration of our work, we have implemented a set of elementary attacks and a complex attack scenarios. 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 several types of intrusion objective (e.g. SIP Server DDoS, SIP Client DoS, User Highjacking, Injecting Audio Traffic, Spam over Internet Telephony, etc.). Moreover, candidate countermeasures have been implemented as scripts. Different types of countermeasures are available for the system: blocking the traffic between the attacker and the server (or the user), changing the user’s credentials, Encrypting the media traffic, etc. The attack graph generated in LAMBDA language contains six intrusion objectives with eight available countermeasures. The number of LAMBDA models (i.e. elementary attack steps and intrusion objectives) used in the attack graph is thirty one (See Figure 5).
The attack graph can be divided into two parts: During the first part, the attacker sends spam mail with a malicious link to infect potential victims in the enterprise network, in other words it is the remote-to-local part of the attack. In the following scenario, three machines in the enterprise network are infected with a bot. In the second part, the attacker being in the ‘inside’, is now able to perform several types of elementary actions to achieve one of the intrusion objectives.

**Step 0 of the ongoing attack:** The attacker has not executed any attack yet. Having six intrusion objectives in the attack graph, the first phase (See Section 3.1) is to decompose this attack graph into six subgraphs. Then, the second and the third phases (see Sections 3.2 and 3.3) of our methodology are applied for each step to calculate the success likelihood for each intrusion objective. Figure 6a shows the success likelihood of the intrusion objectives (H, I, J, K, L and M); considering the eight candidate countermeasures at step 0. The success likelihood of all the intrusion objectives have relatively low values; this can be explained by the fact that the attacker did not yet execute successfully any attack step. After the execution of phase 4 of our methodology (see Section 3.4), it is clear that the countermeasure $CM_1$ has the highest priority because it is able to stop all (future) candidate attacks. On the other hand, other candidate countermeasures can block some, but not all, the candidate intrusion objectives.

**Step 1 of the ongoing attack:** In the first step of the attack, the attacker had gained a remote shell and successfully infected three internal machines. At step 1, the four phases of the methodology are re-applied; this step affects the success likelihood of the all candidate intrusion objectives. Figure 6b shows the success likelihood of the intrusion objectives; considering the candidate three countermeasures at step 1. We notice that the success likelihood of all intrusion objectives have raised. Since the
machines are now infected with bots, the countermeasure CM_1 (Killing Remote Shells) is no more effective. As in Step 0, the highest priority is for CM_2 because it is capable of blocking four objectives with the highest success likelihood). On other hand, CM_3, CM_7 and CM_8 can block two intrusion objectives. Finally CM_4, CM_5 and CM_6 can block only one intrusion objective.

**Step 2 of the ongoing attack:** The attacker proceeds in his attack and launches an Active User Discovery attack and SIP Entities Fingerprinting attack. The four phases of the methodology are re-applied; Figure 6c shows the success likelihood of the six intrusion objectives; considering the candidate eight countermeasures, at step 2. We can notice that the success likelihood of all the intrusion objectives have risen. We can also note that the success likelihood of intrusion objective k has risen dramatically; this can be explained by the fact that the attacker has only one remaining step (i.e. sending malformed packet) to cause a Phone DoS. Therefore at this step, we can see that CM_2 has the highest priority because it is capable of stopping intrusion objective k (and H, I and J), which has the highest success likelihood (i.e. the most ‘urgent’) at this stage of the attack scenario. CM_6 has the second highest priority because it also can stop intrusion objective K.

**Step 3 of the ongoing attack:** In this step, the attacker launches MAC address Discovery and ARP Poisoning. The four phases of the methodology are re-applied. Figure 6d shows the success likelihood of intrusion objectives L and M have raised dramatically; considering that the attacker is very close to achieve these two intrusion objectives. At this step, CM_8 (i.e. Encrypting RTP Media Traffic) has the highest priority, because it is the only candidate countermeasure capable of blocking these two intrusion objectives (i.e. L and M).

Therefore, for each evolution of the attach progress (i.e. new executed step), the administrator or the response system is supported with a prioritized list of candidate countermeasures. This prioritization allows the administrator or the response system to plan and launch the most effective and ‘urgent’ countermeasures first, which is useful in case of countermeasures that can not be activated simultaneously. The prioritization considers only, as we said before, the success likelihood of the potential intrusion objectives. Therefore, we see that our model have to be combined with cost-sensitive models to take in consideration the impact of the attacks and the reactions. In other words, an ideal response system has to consider the risks (i.e. success likelihood and the impact) of ongoing attacks and the impact of candidate countermeasures, to plan and launch the most effective response plan.

On the other hand, Figure 7 shows the evolution of the Success Likelihood of each intrusion objective, with respect to the attack progress, without launching any countermeasure. After step 1, the success likelihood of all intrusion objectives have risen; this can be explained by the fact that step 1 (Remote-to-Local) leads to all the candidate intrusion objectives. Similarly, we can notice at step 2 the success likelihood of intrusion objective H, I, J and K increased significantly, but remained afterward stable. This is explained by the fact that the attacker did take another path that does leads towards intrusion objective L and M. After the successful execution of step 3, only intrusion objective L and M were concerned and have raised dramatically. The attacker is now one step away from achieving a candidate intrusion objective (i.e. injecting media RTP traffic to achieve intrusion objective L, or mixing media RTP traffic to achieve intrusion objective M). Moreover, we notice that \( L_L > L_M \); this can be explained by the fact that the remaining attack step to achieve intrusion objective L (i.e. Injecting RTP Packets) is easier than the step needed to achieve intrusion objective M (i.e. Mixing RTP Packets).

5 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
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 response system based on risk analysis approach is presented; where likelihood and impact are combined to calculate the Risk of Ongoing attacks.

On the other hand, there have been several papers in the literature that discuss the generation and the analysis of attack graph. During the MIRADOR project, Cuppens et al. presented in [13] the semi-explicit approach to correlate elementary attacks described using LAMBDA language [15]. Moreover, In [14] and [26], 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 Miège work in the context of MIRADOR project, which has been done independently and in parallel. In [27], 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 constructs high-level attack scenarios from low-level intrusion alerts, and reasons 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, our approach 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 [28]. 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 an intrusion response approach for ongoing attacks.

Madan et al. in [29] propose a general framework to assess the MMTSF (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 [30], McQueen et al. have 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. For this purpose, 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 his paper does not discuss intrusion response systems. Furthermore, the paper does not take in 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.

6 Conclusion

In this paper, real-time assessment of the Success Likelihood for the ongoing attacks has been presented. This model takes in consideration 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. Moreover, our model can help to prioritize and launch countermeasures that can not be activated simultaneously. Finally, the proposed model has been successfully validated in a VoIP use case using complex attack scenarios. In the future, the effectiveness of our model to select the best countermeasure(s) will be explored by combining the Likelihood and the Impact (thus assessing 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

