Intelligent response system to mitigate the success likelihood of ongoing attacks

Wael Kanoun, Nora Cuppens-Boulahia, Frédéric Cuppens, Samuel Dubus, Antony Martin

Intelligent Response System to Mitigate the Success Likelihood of Ongoing Attacks

Wael Kanoun*†, Nora Cuppens-Boulahia†, Frédéric Cuppens†, Samuel Dubus* and Antony Martin*
*Bell Laboratories, Alcatel-Lucent
†Telecom Bretagne, Institut Telecom

***Abstract*—Intrusion response models and systems have been recently an active field in the security research. These systems rely on a fine diagnosis to perform and optimize their response. In particular, previous papers focus on balancing the cost of the response with the impact of the attack. In this paper, we present a novel attack response system, based on the assessment of the likelihood of success of attack objectives. First, the ongoing potential attacks are identified, and their success likelihood are calculated dynamically. The success likelihood depends mainly on the progress of the attack and the state of the monitored system. Second, candidate countermeasures are identified, and their effectiveness in reducing the pre-calculated success likelihood are assessed. Finally, the candidate countermeasures are prioritized.**

***Keywords*-response, success likelihood mitigation, attack objectives, dynamic Markov models.**

I. Introduction

Intrusion Response Systems (IRS) are often used with Intrusion Detection Systems (IDS), in order to launch appropriate countermeasure(s) and stop the detected attack. A proper IRS needs an efficient diagnosis to identify ongoing attacks, and select the best countermeasure(s). For the selection and prioritization procedure, several criteria should be considered: (i) identification of potential attack objectives, (ii) success likelihood assessment of potential attack objectives, (iii) impact assessment of the attack and the countermeasure.

While previous IRS consider the cost (or impact) of the detected attacks to prioritize and launch the countermeasures, we adopt a different yet complementary approach which considers the Success Likelihood *SL* of the detected attacks. In this paper, we use the *SL* of ongoing attacks to present a novel IRS. For simplicity, we consider in this paper that all the detected attacks have the same impact on the monitored system, and that the IRS handles only known attacks. The *SL* is a relative logarithmic metric derived from the time needed to accomplish the ongoing attack [1]: it indicates how close the attacker is to achieve his objective(s). Using this metric, the proposed IRS evaluates the effectiveness of each countermeasure in reducing the success likelihood for the detected attacks. Finally, the model prioritizes candidate countermeasures w.r.t. their effectiveness. This can be useful in the case: (i) when several responses which cannot be activated simultaneously, or (ii) when responses have a cost or side effects, or even (iii) when a single response is effective against several potential attacks. Therefore, the administrator has to select among several response the most 'urgent' and effective one(s). A total defensive-centric view is adopted: 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" [2].

This paper is organized as follows. Section II shows how elementary attack are modeled, and how attack graphs are constructed. Section III presents how the *SL* of each potential attack is calculated. In Section IV, we propose a response model based on a real-time assessment of the likelihood of success for ongoing attacks, and the effectiveness of candidate countermeasures. In Section V, a VoIP use case of an enterprise environment with numerical results are presented to illustrate our model. Section VI discusses existing and related work.

II. Attack Modeling

First, an efficient IRS has to recognize the ongoing attack(s) in order to respond properly. Attack graphs depicts the attack steps that has been executed on the monitored system, and may even show potential future steps. Thus, we rely on attack graph generation techniques to monitor the attack progress. Attack graphs techniques has been extensively investigated, and several models have been proposed in the last decade. In particular, the *semi-explicit* approach (e.g. [3], [4]) relies on the description of the pre/post-conditions, which represents the prerequisites and effects of the elementary attack actions. This approach then finds causal relationships between these elementary actions and connects them when such a relationship exists. The correlation procedure then consists in building a scenario that corresponds to an attack graph. The semi-explicit approach is generic and flexible because only the elementary steps are specified, and not the whole attack scenarios. Several attack languages may be used to specify the elementary attacks (e.g. LAMBDA [5], JIGSAW [6] and CAML [7]). Since we had successfully used LAMBDA (LAnguage to Model a dataBase for Detection Attacks) with the *semi-explicit* [4] correlation during previous work, it will be retained in the remainder of this paper. However, the proposed IRS can be used with the other attack graph models.

A. LAMBDA Language

We present below a short description of LAMBDA used to describe elementary attack steps. For a formal description, interested readers can refer to [5]:

- *pre-condition*: 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-condition*: This field describes the information system state after the execution of the step. It contains one or several logical predicates.
- *sk*: This field, introduced in [8], indicates the minimum level of skill and/or internal knowledge required to execute the step successfully. In this paper we consider that $0 < sk < 1$, and that step *A* is 'easier' than *B* if $sk_A > sk_B$.
- *detection*: This field is used to map the LAMBDA attack model to the appropriate alert signature(s).

For example (see 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 *Sipext1*. Moreover, the crash of the machine *H2* is the consequence of this elementary attack.

B. Semi-Explicit Correlation

We say that two LAMBDA models *A* and *B* are correlated if the postcondition of *A* matches the precondition of *B*. Thus it can provide a precise diagnosis of the ongoing intrusion scenario by constructing the attack graph [4]; and predicts the potential future steps and the attack objectives [9]. An example is shown in Figure 1: 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. Thus, *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))*.

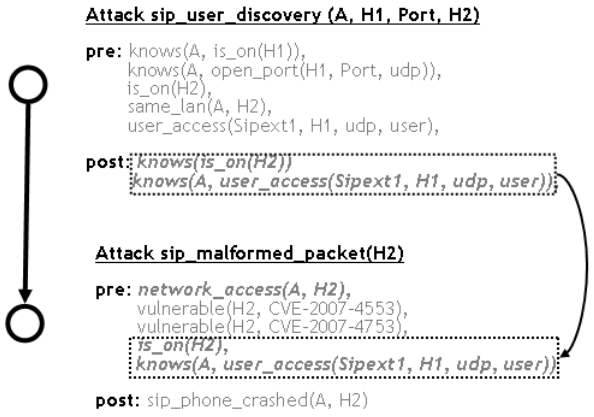


Fig. 1. Example of semi-explicit correlation

Using *semi-explicit* correlation, attack graphs are constructed from generated alerts, using LAMBDA models. The alerts generated by the IDS are first aggregated and regrouped into meta-alerts. For each meta-alert, the associated elementary attack (specified with LAMBDA language) is instantiated. Moreover, potential future steps and candidate attack objectives are identified with the *semi-explicit* correlation. Thus, the non-detection of some attack steps will not cripple the IRS. Moreover, as soon as one of the following steps is detected, the non-detection of a previous step will have no effect on the IRS effectiveness.

On the other hand, since attack graphs are constructed using pre-specified LAMBDA models, it is obvious that our IRS can handle only known attacks. *Zero-day* attacks cannot be represented in the attack graphs for one of the following reasons: (i) the 'new' attack signature is not yet associated to one of the existing LAMBDA model, or (ii) the LAMBDA model representing the 'new' attack is not yet specified. We view *zero-day* attacks as an issue related to the detection process, and not to the response process.

III. Assessing the Success Likelihood of Ongoing Attacks

A node in the attack graph represents an elementary attack that has been executed and observed successfully, or a potential step that can be executed in the future (i.e. not yet observed). These nodes lead to the *attack objectives*, which constitute the terminal nodes in the attack graph. For each evolution of the attack or system state, a new attack graph is instantiated. This can be due to a new observed attack step: a future step in the previous graph turns to be executed 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 switches the concerned step state (executable or unexecutable). Therefore, the model will be applied for each instance of the attack graph. We can summarize the procedure to the following phases (see [1] for more information):

1) decompose the attack graph to several subgraphs (i.e. one subgraph for each attack objective),
2) transform each subgraph into a dynamic Markov Model, and calculate the *SL* metric,

A. Decomposing Attack Graph

First, the generated attack graph is decomposed into several subgraphs; each subgraph is associated to an attack objective. For instance, an attack graph with n attack objectives (i.e. terminal nodes) is decomposed into n subgraphs. In result, each subgraph contains all the future (i.e. not yet observed) nodes which lead to the associated attack objective, and also contains the already observed steps adjacent to the future steps. Figure 2 is an example of an attack graph with three attack objectives, with its decomposition into three subgraphs.

B. Instantiating Markov Models and Assessing the SL

Each subgraph is transformed into a dynamic Markov Model that considers the progress of the ongoing attack(s) and the evolution of the monitored system state. The *transition probabilities* and *sojourn mean time* for each step in the subgraph are calculated. Thus, the *transition matrix* and *exit rate matrix* are

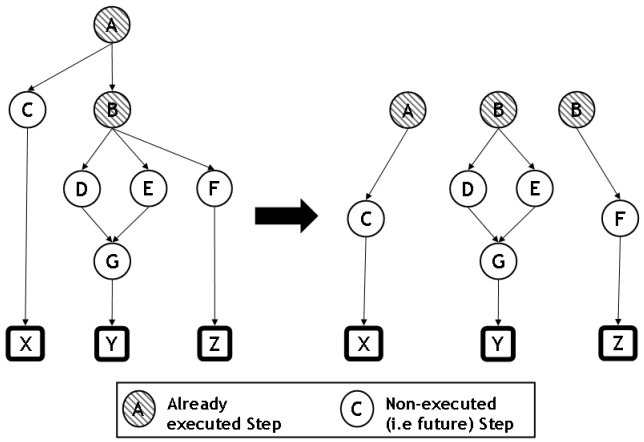


Fig. 2. Decomposition of an attack graph instance into subgraphs

instantiated for the Markov model associated to the subgraph. Interested readers may refer to [1] for more details. Markov Model has been chosen because it adds to the attack graphs a 'temporal' dimension, which is needed to calculate the *SL*. 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.

For each Markov model, the Mean Time to attack objectives *MTAO* is calculated. Finally, for each candidate attack objective X, we calculate its success likelihood SL_x. We use the logarithmic formula proposed in [1], similar to the one used to express the magnitude of a physical quantity (current, voltage, power, etc.). The success likelihood depicts the variations of the *MTAO* metric:

$$SL_x = -20 \times \log_{10}\left(\frac{MTAO_X - 1}{MTAO_X}\right) \quad (1)$$

The success likelihood $SL_X = f(MTAO_X)$ of an attack objective X grows rapidly if $MTAO_X$ decreases, and $SL_X \to 0$ if $MTAO_X \to \infty$. Thus, if the attacker is closer to attack objective X, SL_X grows exponentially. Ultimately, if the attacker achieves the attack objective, we will have $SL_X \to \infty$.

IV. Response System for Mitigating the Success Likelihood

This section presents a response model based on real-time assessment of the *SL* for the ongoing attacks. The model takes in consideration the real-time evolution of both the attack and the information system. An evolution could be the result of a new executed and detected attack step, or the modification of a precondition of a future attack step in the scenario. First, candidate countermeasures are identified. Second, each candidate countermeasure will be simulated, and *SL* values will be re-calculated. Finally, the candidate countermeasures are prioritized w.r.t. their *SL* mitigation effectiveness.

A. Identifying Candidate Countermeasures

The anti-correlation approach [10] allows to identify the candidate responses along with the scalability consideration: we do not need to statically associate each countermeasure to one or several attacks. First, all the responses are modeled with LAMBDA. Then, the association is performed dynamically using anti-correlation: A countermeasure *C* is anti-correlated with an attack *A* if the postcondition of *C* matches the precondition negation of *A*. The anti-correlation approach is based upon finding the appropriate countermeasures that turn elementary future steps unexecutable, due to precondition(s) modification. 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 3: 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*.

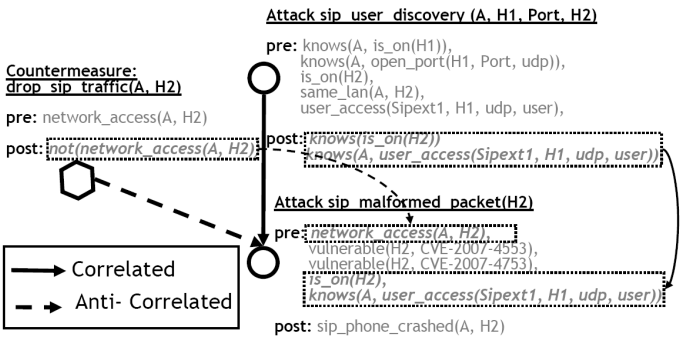


Fig. 3. Examples of semi-explicit correlation and anticorrelation procedures

B. Simulating the Countermeasures and Re-calculating the SL

Candidate countermeasures are identified using the anti-correlation approach to block future attack steps. As a result, these countermeasures reduce the *SL* of one or several attack objectives. Therefore, for a given instance of the attack graph, each candidate countermeasure will be simulated, and new values of the success likelihood for the attack objectives will be calculated. Thus, the effectiveness of a given countermeasure in reducing the *SL* of attack objectives can be assessed, and compared to other countermeasures.

During the simulation of the activation of a given countermeasure CM_V, the same procedure as described in Section III will be applied. The main difference is that the attack steps anti-correlated with CM_V are considered as blocked. In other words, the time needed by the attacker to execute successfully these attack steps will be very high and almost infinite. Consequently, a high mean sojourn time is assigned to the attack steps blocked by countermeasure CM_V. In other words, a low value (e.g. 10^{-3} in this paper) will be assigned to the Markovian parameter *exit rate* of these attack steps.

For the K^{th} instance of the attack graph, with the countermeasure CM_V activated, we denote the new value of the success likelihood for the attack objective X by $SL_{K,V,X}$.

Moreover, we denote by $\overrightarrow{SL}_{k=K,v=V}$ the vector that contains sorted (descending order) *SL* values of all the attack objectives, during the step $k = K$ of the attack progress, while the countermeasure $CM_{v=V}$ has been activated:

$$\overrightarrow{SL}_{K,V} = [SL_{K,V,1}, \quad SL_{K,V,2}, \quad SL_{K,V,3}, \quad \cdots] \qquad (2)$$

C. Prioritizing Candidate Countermeasures

The goal of this phase is to prioritize candidate countermeasures by their mitigation effectiveness of the *SL* of candidate attack objectives. During the K^{th} attack step, and for each candidate countermeasure CM_V the success likelihood vector $\overrightarrow{SL}_{K,V}$ that contains the *SL* of the candidate attack objectives is calculated. During the K^{th} step in the attack progress, we say that countermeasure CM_V has a higher priority than $CM_{V'}$ if $\overrightarrow{SL}_{K,V} < \overrightarrow{SL}_{K,V'}$; where the $<$ operator is a lexicographic comparison

Proposition 1: If $\overrightarrow{SL}_{K,V} < \overrightarrow{SL}_{K,V'}$ *then* $\mathrm{CM}_V >_{priority} \mathrm{CM}_{V'}$ *at* K^{th} *step of the ongoing attack.*

CM_V has a higher priority because it reduces more significantly the *SL* of candidate attack objectives than $CM_{V'}$. The descending order of $\overrightarrow{SL}_{k=K,v=V}$ ensures that the most 'urgent' attack objectives are considered first, and their *SL* are reduced. Therefore, candidate countermeasures are prioritized and sent to the administrator or to the response management module. The prioritization can be also useful to determine which countermeasure have to be launched first, when several countermeasures cannot be activated simultaneously.

V. VoIP Use Case

The case study is a SIP-based VoIP enterprise service. The VoIP service (see Figure 4) is composed of a SIP server on a dedicated network; which acts as a SIP registrar for the HTTP Digest authentication, and as a SIP router/proxy for call routing. OpenSER[1] is used as the SIP server, while the authentication is delegated to a collocated RADIUS server, based on FreeRADIUS[2]. There are three SIP User Agents (UA) networks: the first for softphones (i.e. X-Lite, S-JPhone and Linphone), and the second for hardphones (i.e. Thomson, Linksys, Zyxel) which has been divided into wired and wireless networks. The intrusion detection infrastructure relies on Snort[3]. For the *Alert Collection and Correlation Engine* module, we use the CRIM prototype [11] that (i) aggregates the collected alerts and (ii) generates a pre/post-condition graph adopting the semi-explicit approach. Moreover, CRIM identifies candidate countermeasures using anti-correlation. Finally a MATLAB-based module calculates the *SL* of each attack objective in the attack graph, and prioritizes the candidate countermeasures.

To demonstrate of our work, we have implemented a set of elementary attacks. Both SIP related attacks, based on flaws in the protocol design [12] and flaws in software implementation,

[1]www.openser.org
[2]www.freeradius.org
[3]www.snort.org

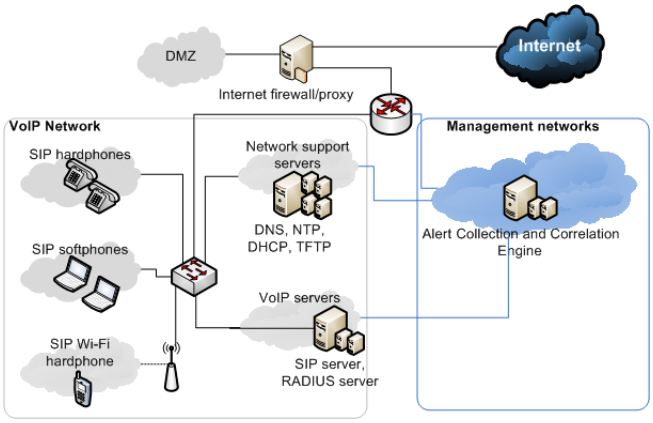


Fig. 4. The VoIP testbed

were identified and implemented in the VoIP testbed. On the other hand, six attack objectives which violates the operation and security policy have been specified (e.g. SIP server DDoS, user highjacking, injecting audio traffic, SPIT, etc.). Moreover, candidate countermeasures have been implemented using various shell script languages. Eight countermeasures are available for the system (e.g. blocking the traffic between the attacker and the server (or the user), changing the user's credentials, encrypting the media traffic, etc.). The number of LAMBDA models (i.e. elementary attack steps and objectives) used in the attack graph is thirty one. A correlation engine, using the semi-explicit approach, generates the attack graph (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: 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 'inside', is now able to perform several types of elementary actions to achieve one of the attack objectives.

a) Step 0 of the ongoing attack: The attacker has not executed any attack yet. Having six attack objectives, the attack graph is decomposed into six subgraphs. Then, the *SL* for each attack objective is calculated. Figure 6a shows the *SL* of the attack objectives (H, I, J, K, L and M), considering the eight candidate countermeasures. The *SL* of all the attack objectives have relatively low values because the attacker did not yet execute successfully any attack step. It is obvious that 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, candidate attack objectives.

b) Step 1 of the ongoing attack: The attacker gains a remote shell and successfully infects three internal machines. Obliviously, this affects the *SL* of all candidate attack objectives. Figure 6b shows the *SL* of the attack objectives, considering the eight candidate countermeasures at step 1. We notice that the *SL* of all attack objectives have raised. Since the machines are now infected with bots, the countermeasure CM_1 (kill remote shells) is no more effective. As in Step 0, the

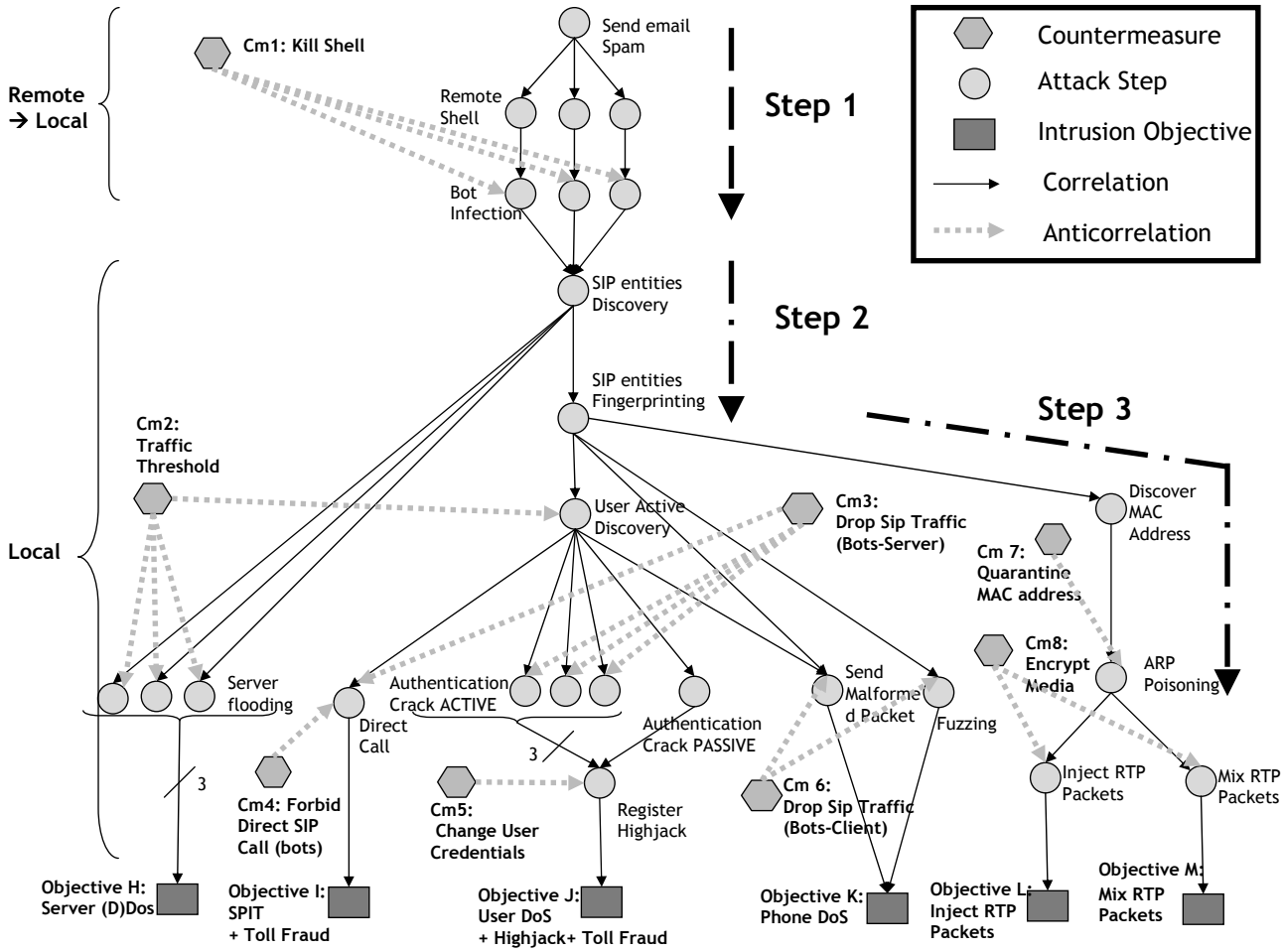


Fig. 5. The VoIP use case attack graph

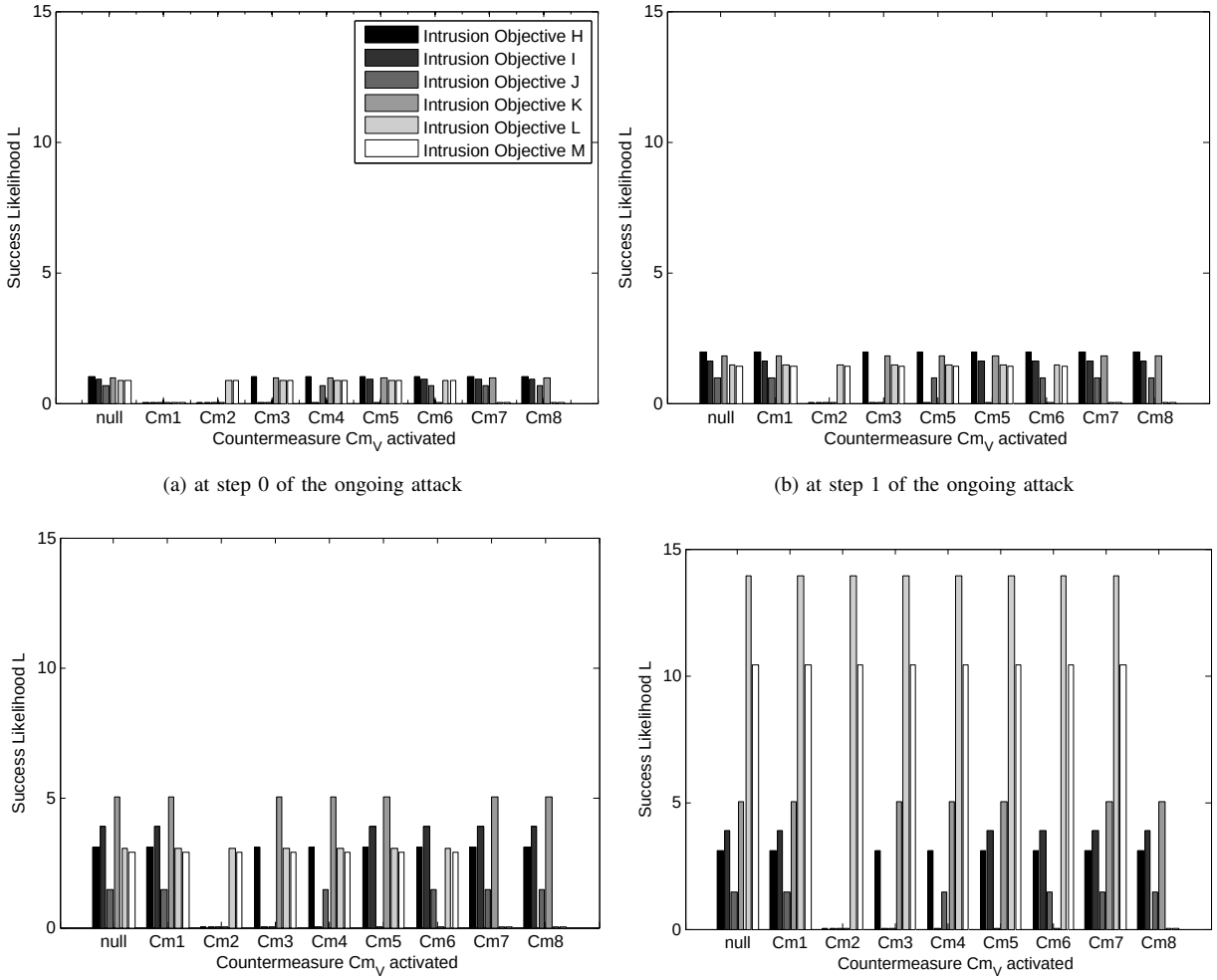
highest priority is for CM_2 because it is capable of blocking four objectives with the highest *SL*. On the other hand, CM_3, CM_7 and CM_8 can block two attack objectives. Finally CM_4, CM_5 and CM_6 can block only one attack objective.

c) Step 2 of the ongoing attack: The IRS should stop the ongoing attack as early as possible. However, the IRS might not activate a countermeasure for several reasons: the first steps of the attack were not properly detected (e.g. false positive, false negative, etc.), or the administrator has not launched the appropriate countermeasures because they cost too much, or because it is too late to launch the candidate countermeasure (e.g. killing a remote shell after the bot infection of a machine does not stop the ongoing attack). We consider in this step that the attacker proceeds and launches an *active user discovery* attack with *SIP entities fingerprinting* attack. We notice that the *SL* of all the attack objectives have risen (see Figure 6c). We can also note that the *SL* of attack 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, CM_2 has the highest priority because it is capable of stopping attack objective K (and also H, I and J), which has the highest *SL* (i.e. the most 'urgent'). CM_6 has the second highest priority because it also can stop attack objective K.

d) Step 3 of the ongoing attack: The attacker performs a *MAC address discovery* and *ARP poisoning*. After re-evaluation, Figure 6d shows that the *SL* of attack objectives L and M have raised dramatically. 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 attack objectives (i.e. L and M).

For each evolution of the attach graph, 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 launch the most effective and 'urgent' countermeasures first, which is useful in case of countermeasures that cannot be activated simultaneously.

It is obvious that the attack must be stopped as soon as possible (e.g. kill the shells or disinfect victim machines during step 1). Since IDS are not perfect, the non-detection of one or several attack steps can be missed. For instance step 1 may have been executed undetected and thus CM_1 was not activated. Hence, since the machines were successfully infected, closing the remote shells becomes an obsolete countermeasure. That is why the IRS will re-prioritize the countermeasures at each time a new attack step has been detected. Moreover,



(a) at step 0 of the ongoing attack

(b) at step 1 of the ongoing attack

(c) at step 2 of the ongoing attack

(d) at step 3 of the ongoing attack

Fig. 6. Evolution of *SL* of the attack objectives

a given countermeasure cannot be executed due to system constraints and limitations (e.g. the enforcement of the countermeasure fails, or the impact of the countermeasure is too high, or another opposite countermeasure is already active, etc.). Therefore, even if the attack was detected, a quick response may not be feasible. However, the proposed IRS can handle such cases perfectly, by re-prioritizing at each time the candidate countermeasures.

VI. Related Work

Recently, several intelligent intrusion response systems have been proposed. Toth and Kruegel [13] 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.* [14] also discuss the need to consider the cost of intrusions damage, the cost of manual and automated response to an intrusion, and the operation cost, which measures constraints on time and computing resources. Similar approaches were also proposed in [15], [16] and [17]. A general framework for advanced response systems based on risk analysis approach is defined in [18]; where likelihood of success and impact are combined to calculate the risk of detected attacks.

The generation of attack graphs generation has been an active research field in the last decade. During the MIRADOR project, Cuppens *et al.* presented in [4] the semi-explicit approach to correlate elementary attacks described using LAMBDA [5] . In [3], 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's work which has been done independently and in parallel. Similar models were presented in [6], [7]. In [19], Sheyner *et al.* used a model of exploits (possible attacks) in terms of their preconditions and postconditions to construct possible sequences of attacks. By contrast, our method constructs high-level attack scenarios from low-level intrusion alerts, and reasons about attacks possibly missed by the IDS. While the previous vulnerability analysis techniques are focused on analyzing what attacks *may happen* to a given system, our approach constructs what *is*

happening to a given information system according to the alerts reported by IDS.

[1] presents how to calculate the success likelihood of candidate attacks using generated attack graphs. Therefore, candidate attack objectives can be prioritized. However, [1] considers that each countermeasure may affect a single attack objective. Thus the prioritization of candidate attack objectives was equivalent to the response prioritization which is obviously not always true.

Madan *et al.* in [20] proposed 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. Moreover, this framework does not take into consideration neither the dynamic nature of an ongoing attack, nor the real-time state of the monitored system.

In [21], McQueen *et al.* proposed to calculate the Risk Reduction due to installing or modifying security measures (e.g. updates, firewalls, etc.). The calculation is based on the existing vulnerabilities in the system. In this approach, an attack graph is composed of nodes that represent the attack stages. The edges are associated with time to compromise, which is calculated in function of the number and types of vulnerabilities. This paper does not discuss intrusion response systems. Furthermore, since this paper presents an offline analysis model, it does not consider the real-time nature of the monitored system.

VII. Conclusion

In this paper, a novel IRS based on a 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 monitored system state. The *SL* 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 cannot be activated simultaneously. Finally, the proposed model has been successfully validated in a VoIP use case using complex attack scenarios that violate operation and security policies. However, the prioritization considered only the *SL* of the potential attack 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, the response system has to consider the risks (i.e. the success likelihood and the impact) of ongoing attacks and the impact of candidate countermeasures. 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 real-time risk) of a given attack. We will explore also the use of Hidden Markov Models in the *SL* assessment model, to take into consideration the uncertainty of the attack progress and the monitored system state.

Acknowledgment

The authors would like to thank Serge Papillon for his valuable contribution and support to this work.

References

[1] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, S. Dubus, and A. Martin, "Success likelihood of ongoing attacks for intrusion detection and response systems," in *International Conference on Computational Science and Engineering, 2009 (CSE '09)*, Vancouver, August 2009.

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

[3] P. Ning, D. Xu, C. G. Healey, and R. S. Amant, "Building attack scenarios through integration of complementary alert correlation methods," in *Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS04)*, 2004, pp. 97–111.

[4] F. Cuppens and A. Miège, "Alert correlation in a cooperative intrusion detection framework," *Security and Privacy, IEEE Symposium on*, vol. 0, p. 202, 2002.

[5] 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'00)*, Toulouse, France, 2000.

[6] S. J. Templeton and K. Levitt, "A requires/provides model for computer attacks," in *NSPW '00: Proceedings of the 2000 workshop on New security paradigms*. New York, NY, USA: ACM, 2000, pp. 31–38.

[7] S. Cheung, U. Lindqvist, and M. W. Fong, "Modeling multistep cyber attacks for scenario recognition," *DARPA Information Survivability Conference and Exposition,*, vol. 1, p. 284, 2003.

[8] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, and J. Araujo, "Automated reaction based on risk analysis and attackers skills in intrusion detection systems," in *Third International Conference on Risks and Security of Internet and Systems, 2008 (CRiSIS '08)*, Oct. 2008.

[9] 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, December 2002.

[10] F. Cuppens, F. Autrel, Y. Bouzida, J. Garcia, S. Gombault, and T. Sans, *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.

[11] 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.

[12] D. Endler and M. Collier, *Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions*. McGraw-Hill Osborne Media, 2006.

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

[14] 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.

[15] N. Stakhanova, S. Basu, and J. Wong, "A cost-sensitive model for preemptive intrusion response systems," in *21st International Conference on Advanced Information Networking and Applications (AINA'07)*, May 2007, pp. 428–435.

[16] H. Wei, D. Frinke, O. Carter, and C. Ritter, "Cost-benefit analysis for network intrusion detection systems," in *28th Annual Computer Security Conference (CSI'01)*, October 2001.

[17] N. Kheir, H. Debar, N. Cuppens-Boulahia, F. Cuppens, and J. Viinikka, "Cost evaluation for intrusion response using dependency graphs," in *IFIP International Conference on Network and Service Security (N2S'09)*, June 2009.

[18] 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.

[19] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing, "Automated generation and analysis of attack graphs," in *Proceedings of IEEE Symposium on Security and Privacy*, 2002.

[20] B. B. Madan, K. Goševa-Popstojanova, K. Vaidyanathan, and K. S. Trivedi, "A method for modeling and quantifying the security attributes of intrusion tolerant systems," *Perform. Eval.*, vol. 56, no. 1-4, 2004.

[21] 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*, 2006.