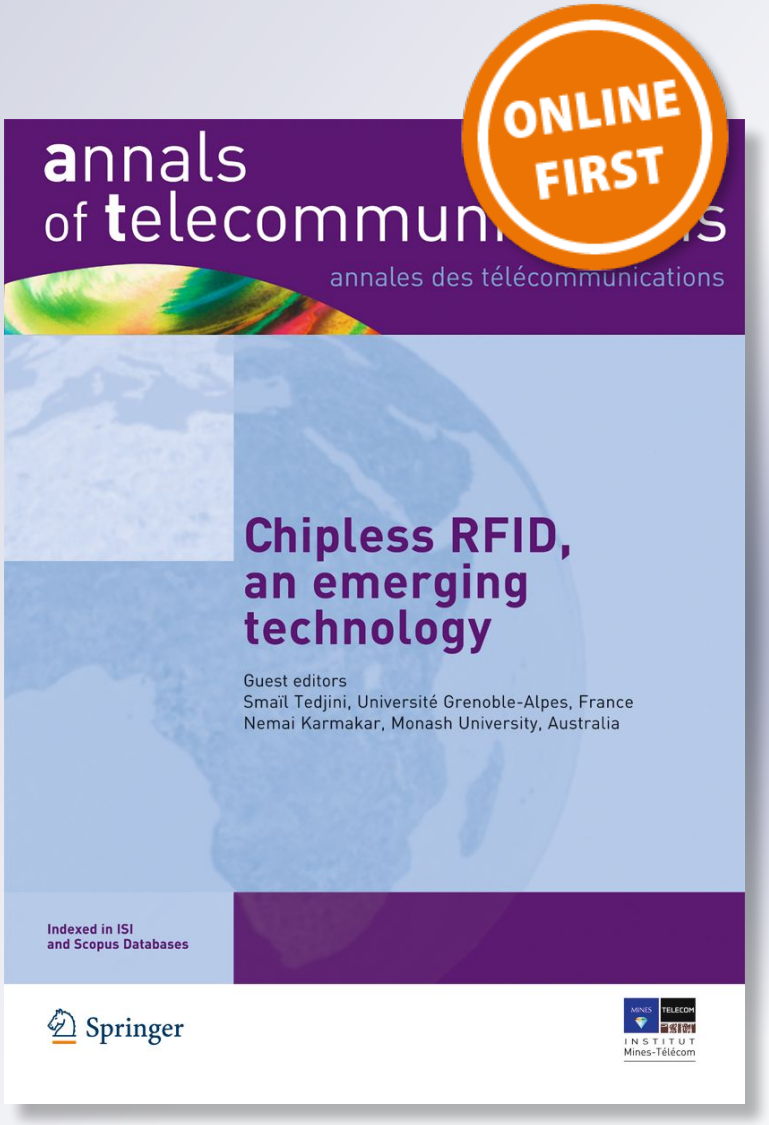
Assessing the risk of complex ICT systems

Nizar Kheir, A. Ridha Mahjoub, M. Yassine Naghmouchi, Nancy Perrot & Jean-Philippe Wary

Annals of Telecommunications

ISSN 0003-4347

Ann. Telecommun.
DOI 10.1007/s12243-017-0617-0



Springer

Annals of Telecommunications
https://doi.org/10.1007/s12243-017-0617-0

Assessing the risk of complex ICT systems

Nizar Kheir[1] · A. Ridha Mahjoub[2] · M. Yassine Naghmouchi[3,4] · Nancy Perrot[4] · Jean-Philippe Wary[4]

Received: 17 October 2016 / Accepted: 7 December 2017

Abstract
ICT systems are becoming increasingly complex and dynamic. They mostly include a large number of heterogeneous and interconnected assets (both physically and logically), which may be in turn exposed to multiple security flaws and vulnerabilities. Moreover, dynamicity is becoming paramount in modern ICT systems, since new assets and device configurations may be constantly added, updated, and removed from the system, leading to new security flaws that were not even existing at design time. From a risk assessment perspective, this adds new challenges to the defenders, as they are required to maintain risks within an acceptable range, while the system itself may be constantly evolving, sometimes in an unpredictable way. This paper introduces a new risk assessment framework that is aimed to address these specific challenges and that advances the state of the art along two distinct directions. First, we introduce the risk assessment graphs (RAGs), which provide a model and formalism that enable to characterize the system and its encountered risks. Nodes in the RAG represent each asset and its associated vulnerability, while edges represent the risk propagation between two adjacent nodes. Risk propagations in the graph are determined through two different metrics, namely the accessibility and potentiality, both formulated as a function of time and respectively capture the topology of the system and its risk exposure, as well as the way they evolve over time. Second, we introduce a quantitative risk assessment approach that leverages the RAGs in order to compute all possible attack paths in the system and to further infer their induced risks. Our approach achieves both flexibility and generality requirements and applies to a wide set of applications. In this paper, we demonstrate its usage in the context of a software-defined networking (SDN) testbed, and we conduct multiple experiments to evaluate the efficiency and scalability of our solution.

Keywords Risk assessment · Graph theory · Complex ICT systems

1 Introduction

In general, there are two main steps that compose a risk management approach [1]. The process starts first with a risk assessment step. It enables an operator to identify security flaws and vulnerabilities in a target system, and to further leverage their induced risks by evaluating the impact on the system once a vulnerability has been exploited by an attacker, and the success likelihood of such an attack when it occurs. The second step is risk treatment. It aims to propose and enforce efficient mitigation decisions in order to keep risk below an acceptable threshold. In this scope, a key challenge for a security management operator is to find the optimal balance between the deployment costs for new countermeasures and the amount of mitigated risks. To address this challenge, existing risk management solutions implement complex mathematical models that

✉ M. Yassine Naghmouchi
mohamedyassine.naghmouchi@orange.com

Nizar Kheir
nizar.kheir@thalesgroup.com

A. Ridha Mahjoub
ridha.mahjoub@lamsade.dauphine.fr

Nancy Perrot
nancy.perrot@orange.com

Jean-Philippe Wary
jeanphilippe.wary@orange.com

[1] Thales Group, Paris, France

[2] PSL Research University, CNRS, LAMSADE, Université Paris-Dauphine, 75016 Paris, France

[3] Université Paris-Dauphine, Paris, France

[4] Orange Labs, Paris, France

Published online: 27 December 2017

apply to a condensed and feature-rich representation of the target system, the latter being created during the risk assessment phase. So far, this process requires substantial expert knowledge and time, which makes it less appropriate for modern ICT systems that are increasingly dynamic and constantly evolving over time.

Until recently, information systems were almost statically designed and unlikely to evolve in a substantial way during runtime. However, the advent of modern virtualization technologies introduced a paradigm shift, enabling ICT systems to easily evolve over time, and so they became almost dynamic *by design*. Today, this manifests itself at different levels of the infrastructure, including the network (e.g., network function virtualization), the system (e.g., hypervisors), and the application layer (e.g., distributed data storage). Hence, modern ICT systems are *complex* in the sense that (1) they include a large number of heterogeneous elements; (2) these elements are connected by non-linear interactions, often of different types (e.g., physical and virtual links); (3) they are subject to external and insider inferences (e.g., intruder threats); and (4) they may constantly evolve over time (e.g., topology changes and new vulnerabilities). Faced with these evolving challenges, current risk assessment methodologies such as *scoring methods* and *graph-based models* suffer from multiple limitations.

A first limitation of existing vulnerability scoring methods such as [2] and [3] is that they only leverage intrinsic properties of a target vulnerability, while not taking into account the way it may affect other components of the system. This drastically limits the ability of scoring methods to scale well against complex ICT systems that involve multiple vulnerable assets, including also causal relationships between multiple vulnerabilities of the system.

Another limitation of scoring methods is their inability to capture the dynamic properties for a given vulnerability. So far, previous attempts to adapt the scoring methods and to incorporate time-based features have been very limited in scope, and so they have been deprecated as in the example of the Common Vulnerability Scoring System (CVSS). The latter does not recommend using its own temporal metrics due to their inability to reliably capture the dynamic aspects of the system [4].

On the other hand, several graph-based models such as attack and dependency graphs [5] offer to leverage topology information within the risk assessment process. They enable to properly address risk propagations between different components of the system. However, this is mostly being done today in a static way. In order to account for changes in the system topology, the operator is often inclined to manually generate a new instance of the graph. In this scope, current graph-based risk assessment models are unable to adapt to dynamic environments, which leaves a large gap between risks evaluated at the system design phase, and new encountered risks that may constantly appear at runtime. Furthermore, the topology description captures only the causal relationships between assets, but does not account for other system dynamics. In particular, the risk propagation between two assets of the system may be higher when these assets are frequently interacting between each other. Consequently, although the risk may propagate between two connected assets, the propagated risk depends on the access frequency between the two assets in a given period of time. The frequency of access is indeed an important factor of risk. We refer in this paper to this notion using the *accessibility* metric.

These limitations present several challenges to defenders. A first challenge consists in considering the notion of risk propagation in systems with a large number of heterogeneous elements. To do so, a model representing the system topology with its associated accessibilities and vulnerabilities is needed. This model should consider the evolution of the topology, the accessibilities, and the vulnerabilities over the time. A second challenge consists in developing a risk evaluation methodology with efficient security metrics in the context of modern ICT systems.

In this paper, we propose a new risk assessment approach that addresses the aforementioned challenges. Our contribution is twofold. First, based on graph theory ([6] and [7]), we introduce the concept of RAGs as a tool for risk assessment. These graphs capture both the topological accessibility features in the system and the security information in terms of vulnerabilities as well as their causal relationships. They take into account not only the current system state but also the way it evolves over a given period of time. In addition, all possible intruders, attack scenarios, and their target assets are explicitly considered as *paths* in the RAGs. Second, we propose a quantitative risk evaluation approach that leverages the RAGs in order to compute relevant security metrics.

This paper is organized as follows. Section 2 describes related work. Section 3 summarizes our contributions. Section 4 provides an overview of our approach. Section 5 introduces our RAG model and its appropriate formalism. Section 6 describes our risk evaluation approach. Section 7 introduces an software-defined networking (SDN) use case that we use for evaluation, and Section 8 provides the results of our experiments. Finally, Section 9 concludes.

2 Related work

The current state of the art includes multiple contributions that offer to evaluate and quantify risks in ICT systems. We discuss in this section two well-known categories that are relevant to our approach, which are the scoring methods and the graph-based methods.

International standard organizations, such as the National Vulnerability Database (NVD) [8], have provided many risk scoring methods that assign a quantitative score to each known vulnerability, based on a shared reference. In particular, the Common Vulnerability Scoring System (CVSS) [4] has become a widely accepted industry standard. While scoring methods provide a common base and reference to share information about vulnerabilities and their severity, they cannot be used as a stand-alone metric to measure risks in a real-world system. Therefore, current approaches in the literature usually compose elementary vulnerabilities that exist within a target system, and their relationships, through a graph-based model. By modeling the system as an interconnected graph, we may leverage the sequencing of elementary attack steps that enable an attacker to acquire illicit access to privileged system assets. These graphs also leverage the context through which a given vulnerability may affect the system, which is a key benefit compared to previous scoring methods. Nonetheless, current graph models are still suffering from some limitations.

To the best of our knowledge, attack graphs are used to assess the risks associated with elementary system vulnerabilities [9–21]. They put forward the cumulative effect of multiple elementary attack steps. Each path in the graph enables an attacker to acquire unintended privileges, which in turn are represented as objective nodes in the graph (e.g., gaining administrator access to a data base). Attack graphs may support multiple metrics that are relevant for risk assessment, including also the likelihood of a given attack step and the cost to the system whenever the attacker achieves an objective in the graph.

The approaches the most related to our work are in [20, 21] and [22]. In [21], authors use attack graphs and hidden Markov models. They introduce a middle-ware approach, using attack graphs, in order to represent a network assets and their vulnerabilities. The parameters used to construct these attack graphs are the network assets and vulnerabilities extracted from the National Vulnerability Database (NVD). Common vulnerabilities and exposure (CVE) scores [23] are used to analyze the potential system states. However, the system in [21] lacks the ability to automatically generate graphs and incorporate their outcome in the system. Furthermore, the network topological information is missing. The scope of our work is different, since our RAGs are automatically generated and the topological context in which the vulnerability appears is considered.

In Hong and Kim [20], both vulnerability information and the topological characteristics of the system are considered. The topological layer contains cycles depending on the network structure. The vulnerabilities layer has a directed tree structure leading to the target of the intruder. Our approach extends the work in [20] as it also accounts for accessibility changes between system assets. In our model, the topology layer includes time-based accessibility metrics. They indicate the frequency of connections between assets, and the way they may be affected by an ongoing risk.

Authors in [22] leverage simultaneous attacks by presenting a new formal description of individual, coordinated, and concurrent attacks. The generation of simultaneous attacks is based on set and graph theory. The graphs are automatically generated using a logical approach based on situation calculus [24]. This is a dialect of first-order logic with second-order logic terms for representing dynamic changes. It consists of situations, predicates, and actions. Nevertheless, in this work, the risk inferred by simultaneous attacks is not considered. Our approach is different as it also integrates the context within which each vulnerability may appear in the system, while also accounting for intruders, system assets, and the way they interact in the system.

Dependency graphs are yet another tool for risk assessment [25–27]. They capture the way the system assets may interact between each others. For example, Kheir et al. [27] introduce a dependency graph that evaluates the confidentiality, integrity, and availability (CIA) impacts for an ongoing attack. Nonetheless, this approach is mainly being used to assess the attack impacts. It is not adapted in its current form for dynamic risk management, including also the balancing between risks and featured reaction strategies.

3 Our contribution

This paper extends related work by providing a new risk assessment framework that leverages (1) the vulnerabilities, (2) the real-time system topology, (3) the accessibility between different components of the system, and (4) the way all these elements evolve over time. First, we propose the concept of RAGs as a tool for risk analysis. These graphs allow analyzing the complex systems by capturing both the topological accessibility features of the target system and security information in terms of vulnerabilities as well as their causal relationships. They take into account both the current system state and its evolution in time. In addition, all possible intruders, attack scenarios, and their target assets are explicitly considered as *paths* in the RAGs. Second, we propose a quantitative risk evaluation approach using the RAGs to compute the risk based on the aforementioned security metrics.

One may argue that the RAGs are yet another variant of existing and well-known attack graphs. Although both tools can be used to leverage the impact that multiple concurrent vulnerabilities may have on a target system, they are still theoretically very different. Attack graphs consist in modeling the system and its behavior as a nondeterministic Büchi automaton [28], where each node captures a unique

state of the system that manifests in a given level of risk. This is very different from the concept of RAGs, which are indeed based on graph theory [6, 7]. As opposed to existing attack graphs, nodes in the RAG correspond to an asset-vulnerability combination. In this scope, the different states of the system are no longer represented as distinct nodes in the RAG. Rather, every instance of the RAG captures the state of the system in a given time, and the evolution of the system state leads to a new instance of the RAG that is automatically derived from the previous one based on the encountered changes.

While a preliminary version of our work has been introduced in [29], this paper adds three main contributions compared to [29]. First, this paper elaborates more on the theoretical concepts and formalism used to build and update the RAGs. Second, it thoroughly evaluates the RAGs by instantiating and testing them using a software-defined networking (SDN) use case. These new experiments demonstrate the adaptability of our approach and its relevance to dynamic environments. Finally, it evaluates the sensitivity of our security metrics to different changes in the system, including also variations in the potentiality, the topology, and the accessibility metrics.

4 Approach overview

Our approach, which is summarized in Fig. 1, applies to discrete time intervals $t_i \in I$ that represent each a given lapse of time where the system remains in a given state. It involves two main steps, which are the risk analysis and the risk evaluation steps. The topology and the vulnerability databases are both used as input to the risk analysis phase, and their output is the RAG model. The risk evaluation phase leverages the RAG model to evaluate and aggregate risks using different security metrics that we detail in this section.

4.1 Risk analysis

We start by identifying the different factors of risk derived from the topology and the vulnerability databases. In the context of this paper, we use the NVD database to illustrate our approach. Three kinds of information are derived from the vulnerability database: the vulnerabilities, their likelihood (how easy it is to exploit a given vulnerability), and their impact (what is the expected damage for exploiting a vulnerability). The assets of the system, which are either physical (e.g., server, switch, and router) or virtual (e.g., virtual machine and network function) components, are all extracted from the topology database. From a security perspective, a subcategory of assets may constitute entry points and are possibly used by attackers to attack and infiltrate the system. These entry points are handled differently in our approach, and they are further called *access points*. In particular, these access points are the root cause for all risks that propagate through the model. The amount of propagated risk across a given edge is also determined through the accessibility between the two adjacent nodes.

The risk analysis process further includes the two following steps.

(1) Topology and vulnerability evolution Once the topology description and the vulnerabilities have been extracted from the input, the suitable models that leverage their evolution over time will be built. We introduce *the potentiality function* and *the accessibility function*; both are functions of time. The potentiality function evaluates the likelihood of each attack at different stages within the time interval I. The accessibility function characterizes the frequency of connection between the system assets within each time slot.

(2) RAGs generation For each time slot $t \in I$, one RAG is automatically generated. The RAGs represent the system states at each time slot as an oriented graph in which a node is either an asset-vulnerability pair or an access point. An arc between two nodes in the RAG represents the exploitation of the vulnerability of the target node from the source node. A path corresponds to a potential violation of a node. The accessibility and potentiality functions are used to qualify, respectively, the nodes and the arcs at each time slot.

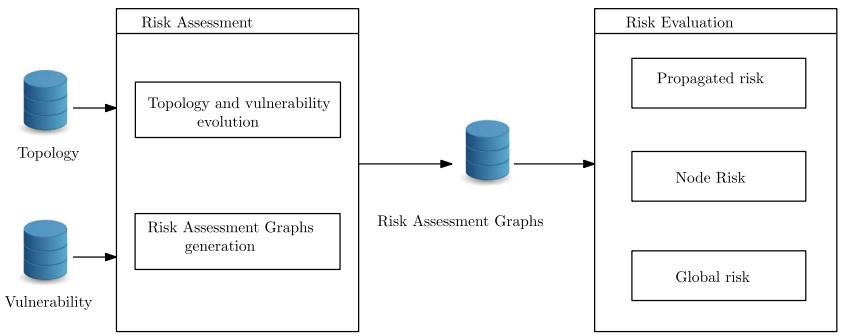


Fig. 1 Framework description

4.2 Risk evaluation

Our model leverages three risk factors, that we introduce as follows.

(1) The propagated risk When propagating through the system, an attacker may execute different attack paths that all lead to the same security objective. While the attacker is required to execute only a single path that leads to this objective, the defender is required to secure all possible paths. From a risk management perspective, the worst case scenario for the defender corresponds to the attack path that has the maximum likelihood (i.e., easiest to the attacker) and that inflicts the maximum damage. This is called the most risky path (yet also the most likely path from an attacker perspective). To identify the most risky paths in the RAG, our model introduces the concept of propagated risk. The propagated risk across a given path depends on the *length* of this path, where length is the sum of weights for all adjacent edges that compose the path. The edge weight represents the difficulty for an attacker to advance his attack scenario through this edge. It is defined as *the propagation difficulty function* that will be formally stated in Section 6.

(2) The node risk We introduce the total risk for a given node as the sum of propagated risks across all attack paths that lead from the access points in the RAG to the current node.

(3) The global risk The global risk is a global security metric that aggregates risks from all nodes in the RAG. It is the sum of the node risks for all nodes in the RAG.

5 The risk assessment graphs

In this section, we formally define the RAG model. We also define the security metrics used to evaluate the nodes, that is the potentiality function and the impact, and the one used to evaluate the edges, namely the accessibility function.

Let $I = \{1, \ldots, T\}$ be a discrete time set. The system is modeled by a set of directed graphs $(G_t = (V_t, A_t))_{t \in I}$. The set of nodes V_t is partitioned between two specified subsets U_t and W_t. A node in U_t represents an access point, and a node in W_t represents an asset-vulnerability pair, constructed and evaluated as follows.

Let Λ_t be the set of the assets of the system at time t (except the access points). Let V_a^t be a set of all vulnerabilities of an asset $a \in \Lambda_t$. To each pair $(a, v) \in \Lambda_t \times V_a^t$, we assign a node $w = (a, v) \in W_t$. Consequently, for each time slot $t \in I$ and for each asset $a \in \Lambda_t$, there is as many nodes in G_t as there exists vulnerabilities in V_a^t. Each node in W_t is assigned a potentiality and an intrinsic impact metrics.

The potentiality function is defined as follows. For each $t \in I$, this function represents the likelihood of a vulnerability being exploited, directly by intruders, at least once before time t. This should be an increasing function of time, since the more time passes, the easier it is for an intruder to exploit a vulnerability. However, until the time $t \in I$, the number of intruders that may directly exploit a given vulnerability is not deterministic. Rather, the number of potential attackers may be represented as independent random variables denoted as $(X_w^t)_{\{t \in I, w \in W_t\}}$, each of which yields to an exploitation with probability p_w at a given time $t \in I$. Hence, X_w^t follows a binomial distribution with parameters t and p_w.

Definition 1 We define the potentiality function f of a node $w = (a, v)$ at time t as the probability of the vulnerability v to be exploited on a at least once before the time slot t, and which corresponds to the following:

$$f_w^t = P(X_w^t \geq 1) = 1 - P(X_w^t = 0) = 1 - (1 - p_w)^t. \quad (1)$$

Equation 1 can be generalized by the function

$$f_w^t(\alpha_w) = 1 - (1 - p_w)^{\alpha_w t}, \quad (2)$$

where α_w is a parameter between 0 and 1 controlling how fast the potentiality of the node w converges to 1.

Now, we define the impact metric.

Definition 2 The impact I_w of a node $w = (a, v) \in W_t$ is defined as the level of damage generated by exploiting v on a.

In this paper, we assume that the impact is constant over time. We may refer to the CVSS scoring method to assign an impact I_w for each node w. We also refer to the same CVSS scoring method to determine the attack likelihood for a given node, represented with the probability of exploitation p_w.

We define the application Δ which, for each node $w = (a, v) \in W_t$, gives its associated asset $a \in \Lambda_t$

$$\Delta : \underset{w=(a,v)}{W_t} \overset{\rightarrow}{\mapsto} \underset{a}{\Lambda_t}.$$

An arc from $w_1 = (a_1, v_1)$ to $w_2 = (a_2, v_2)$ exists if the exploitation of v_1 on a_1 makes possible the exploitation of v_2 on a_2. A direct exploitation of a vulnerability v on an asset a from an access point $u \in U_t$ is represented by an arc from u to $w = (a, v)$. An indirect exploitation corresponds to a $u - w$ path in G_t. Each arc $(n_1, n_2) \in A_t$ is evaluated by the accessibility function.

Definition 3 The accessibility function, denoted by $g_{(n_1,n_2)}^t$, is defined as a frequency of access between $\Delta(n_1)$ and $\Delta(n_2)$ during the time from t to $t + 1$, $t \in I$.

An illustration of a simple RAG instance is provided in Fig. 2. The nodes u_1 and u_2 are the access points. The nodes $w_i, i = 1, \ldots 4$ are the asset-vulnerability nodes. The functions of time f and g are respectively the potentiality and the accessibility functions, and the constant function I corresponds to the impact. As illustrated in Fig. 2, the asset a_1 has two vulnerabilities v_1 and v_2, which are represented with the two nodes $w_1 = (a_1, v_1)$ and $w_2 = (a_1, v_2)$ in the RAG. The arcs in the RAGs are all assigned accessibility metrics, which determine the access frequency between every two adjacent nodes. In particular, the set of arcs A_t in the RAG is partitioned into two subsets A_f and A_u^t. The subset A_f is a fixed subset of arcs. It includes arcs that link all couples of node that belong to the same assets. Since an asset is permanently accessible from itself, we have $\forall t \in I$ $g^t_{(n_1,n_2)} = 1$ if $\Delta(n_1) = \Delta(n_2)$ (e.g., w_1 and w_2 in Fig. 2). Hence, by construction, the sub-graph induced by the nodes of a given asset $a \in \Lambda_t$ are cliques (a graph in which each pair of nodes are connected with bidirectional edge). Also, $\forall t \in I$, G_t contains at least $|\Lambda_t|$ cliques.

The subset of arcs A_u^t represents the uncertain arcs that might exist or not at each time slot in I. For example, if at a time t, we have $g^t_{(n_1,n_2)} = 0$, for a couple of nodes $n_1, n_2 \in V_t$, the arc (n_1, n_2) will be deleted.

To conclude, the RAGs give a compact representation of the target system. The potentiality, the impact, and the accessibility metrics used to label the RAGs are considered as the basic security metrics. They enable to capture the topology of the system (by using the accessibility), and the intrinsic vulnerabilities (by using the potentiality and the impact). These are all elementary security metrics that are used to calculate the elementary and global risks, as further discussed in Section 6.

6 Most likely path-based risk evaluation approach

In this section, we first introduce the most likely path notion. We then define three security metrics based on the most likely path value, that is, the propagated risk, the node risk, and the global risk. Finally, we present the risk evaluation algorithm.

6.1 Most likely paths

Let $t \in I$, $u \in U_t$, and $w \in W_t$. First, we will explain what is the most likely path between u and w at time t. Second, we show how we compute its value. Third, we point out the importance of using the most likely path metric.

6.1.1 What is a most likely path?

Let us first define how the risk propagates on an arc in the RAG. It is possible to exploit a target node only if this node is vulnerable and accessible. Formally, at a given time t, an intruder in n_i can damage an adjacent node n_{i+1} if $g^t_{(\Delta(n_i),\Delta(n_{i+1}))} \neq 0$, and $f^t_{n_{i+1}} \neq 0$. In addition, for a given source node and a given target node, the higher is the potentiality of the target node, the more likely is the propagation. The same goes for the accessibility metric.

We define the propagation function as follows.

Definition 4 Let $t \in I$, and $(n_1, n_2) \in A_t$. The value of the propagation function with respect to t and (n_1, n_2) is given as follows:

$$h^t_{(n_1,n_2)} = f^t_{n_2} \times g^t_{(n_1,n_2)}. \tag{3}$$

For all $t \in I$, and $(n_1, n_2) \in A_t$, $h^t_{(n_1,n_2)}$ is a value between 0 and 1.

Now, we generalize the concept of the propagation function for the paths between the access points and the nodes in the RAGs. Let $\pi^t_{u,w}$ define the set of paths from an access point $u \in U_t$ to an asset-vulnerability node $w \in W_t$ at time slot t. Let $\pi = (n_1, \ldots, n_k)$ be a path of length k (in number of nodes) in $\pi^t_{u,w}$ such that $n_1 = u$ and $n_k = w$. We define *the path propagated potentiality*, denoted by $P^{\pi,t}_{u,w}$, as follows.

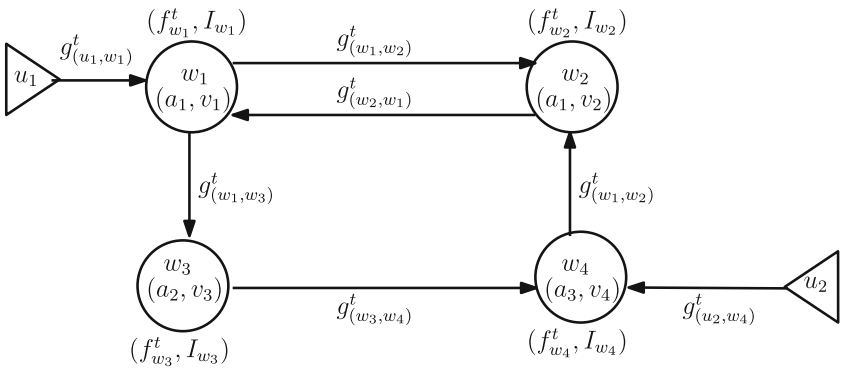


Fig. 2 The RAG model at time t

Definition 5 Let $t \in I$, $u \in U_t$, $w \in W_t$, and let $\pi \in \pi^t_{u,w}$. The propagated potentiality on path π is

$$P^{\pi,t}_{u,w} = \prod_{i=1}^{k-1} h^t_{(n_i,n_{i+1})}. \tag{4}$$

The most likely path between u and w at time t is the path allowing a maximum risk propagation, which corresponds to the path of maximum propagated potentiality. The value of the most likely path between u and w at time t is given by the following:

$$P^t_{u,w} = \max_{\pi \in \pi^t_{u,w}} \{P^{\pi,t}_{u,w}\}. \tag{5}$$

6.1.2 How to compute the value of the most likely path?

In the literature, the problem (5) corresponds to the most reliable path problem [30]. This can be reformulated as a shortest-path problem [31], by labeling the arcs of the RAG with a propagation difficulty function (6). Let us first define *the propagation difficulty function*.

Definition 6 Let $t \in I$, and let $(n_1, n_2) \in A_t$. The value of the propagation difficulty function with respect to t and (n_1, n_2) is defined as follows:

$$H^t_{(n_1,n_2)} = -\log(h^t_{(n_1,n_2)}). \tag{6}$$

Note that the metric (6) indicates how difficult it is for an attacker to exploit node n_2 from node n_1. On the other hand, the propagation function h given in (3) reflects how it is easy for an attacker to exploit node n_2 from node n_1. The metric (6) can be generalized to paths. Let $t \in I$, $u \in U_t$, $w \in W_t$, and $\pi \in \pi^t_{u,w}$, where $\pi = (n_1, \ldots, n_k)$ such that $n_1 = u$ and $n_k = w$. The value of the propagation difficulty function of the path π is given by the sum of the values of the propagation difficulty functions of the arcs composing π at time t, that is $\sum_{i=1}^{k-1} H^t_{(n_i,n_{i+1})}$. Computing the propagated potentiality for $u \in U$, $w \in W$, $t \in I$ and $\pi \in \pi^t_{u,w}$, reduces to maximizing $\prod_{i=1}^{k-1} h^t_{(n_i,n_{i+1})}$ over $\pi^t_{u,w}$. This is equivalent to minimizing $\frac{1}{\prod_{i=1}^{k-1} h^t_{(n_i,n_{i+1})}}$ over $\pi^t_{u,w}$, which is also equivalent to minimizing $\log(\frac{1}{\prod_{i=1}^{k-1} h^t_{(n_i,n_{i+1})}})$ over $\pi^t_{u,w}$. Consequently, the problem of finding the propagated potentiality $P^t_{u,w}$ is equivalent to the following:

$$sp^t_{u,w} = \min_{\pi \in \pi^t_{u,w}} \left\{ \sum_{i=1}^{k-1} H^t_{(n_i,n_{i+1})} \right\}. \tag{7}$$

Let $u \in U$, $w \in W$, and $t \in I$. In order to compute the value of the most likely path between u and w at time $t \in I$, we simply label the arcs of G_t by the values of the propagation difficulty function (6). Consequently, by running a shortest-path algorithm on G_t, the length of the shortest path between u and w at time t is $sp^t_{u,w}$. Therefore,

$$P^t_{u,w} = \frac{1}{\exp(sp^t_{w,u})}. \tag{8}$$

6.1.3 What is the role of the most likely path?

For $u \in U_t$, $w \in W_t$, $t \in I$, let d^w_u be a propagation difficulty threshold, which means that the value of the propagation difficulty function of any path between u and w at time t must not exceed d^w_u. Given a path π between $u \in U_t$ and $w \in W_t$ at time t, π is considered as *secured* if its propagation difficulty function value is greater than or equal to d^w_u.

As stated in Section 4, from a risk management perspective, eliminating the propagated risks requires the securing of all the paths between access points and each asset-vulnerability node at each time slot. The importance of the most likely path is that it is possible to reduce the security of all the paths in the RAG to that of the most likely paths. In other words, the most likely paths are sufficient to conclude about the security of all the system paths. The most likely path between $u \in U_t$ and $w \in W_t$ at time $t \in I$ is secured when its propagation difficulty function value is greater than or equal to d^w_u. On the other hand, as previously shown, the most likely path between u and w at time t is nothing but the shortest path between u and w at time t. Therefore, if the most likely path is secured, all other paths are also secured.

Consequently, the most likely paths are sufficient to conclude about the security of all the system paths. In addition, as the weights on the graph G_t are nonnegative, we can compute the most likely paths in polynomial time (using a shortest-path algorithm).

6.2 Risk evaluation algorithm

The risk of having a successful attack on $w = (a, v)$ is a potential exploitation of the vulnerability v. This exploitation has an impact on the affected assets. The propagated risk $R^t_{u,w}$ from an access point u to a node w is the combination of the following two factors: the *propagated potentiality* $P^t_{u,w}$ and the *impact* I_w.

Definition 7 Let $t \in I$, $u \in U_t$, and $w \in W_t$. The propagated risk from u to w, at the time slot t, is given by

$$R^t_{u,w} = P^t_{u,w} I_w. \tag{9}$$

For each asset-vulnerability node, the summation of the propagated risk from all access points gives the node risk. This is defined as follows.

Definition 8 Let $t \in I$, and $w \in W_t$

$$R_w^t = \sum_{u \in U_t} R_{u,w}^t. \tag{10}$$

Finally,

Definition 9 Let $t \in T$. The global risk at time t is given by the summation of the nodes risks, that is as follows:

$$R^t = \sum_{w \in W_t} R_w^t. \tag{11}$$

Our risk evaluation algorithm is presented in Algorithm 1.

Algorithm 1 Risk evaluation

Data: $G_t \; \forall t \in I$
Result: $R_{u,w}^t$, R_w^t, $R^t \; \forall t \in I, u \in U_t, w \in W_t$

$R_{u,w}^t = 0$
$R_w^t = 0$
$R^t = 0$
/* Propagated risk */
for $t \in T$ **do**
  for $u \in U_t$ **do**
    for $w \in W_t$ **do**
      $sp_{u,w}^t = Dijkstra(u, w)$
      if $sp_{u,w}^t != \infty$; // an $u - w$ path exists
      then
        $P_{u,w}^t = \frac{1}{\exp(sp_{u,w}^t)}$
      end
      else
        $P_{u,w}^t = 0$
      end
      $R_{u,w}^t = P_{u,w}^t I_w$
    end
  end
end
/* Node risk */
for $t \in T$ **do**
  for $u \in U_t$ **do**
    for $w \in W_t$ **do**
      $R_w^t += R_{u,w}^t$
    end
  end
end
/* Global risk */
for $t \in T$ **do**
  for $w \in W_t$ **do**
    $R^t += R_w^t$
  end
end

7 SDN case study

In this section, we illustrate our methodology through its application to a software-defined networking (SDN) [32] use case. Conventional networks unify the control and data planes on a physical device, which typically consists of proprietary hardware and software. SDN decouples the control plane from the data forwarding. An SDN controller uses a protocol such as OpenFlow to control switches, which are now only responsible for handling the data plane. From a security point of view, the separation of the control and data planes brings new security challenges, as it adds a new attack surface. Since the controller is responsible for managing the entire network, existing security flaws in the control plane may have drastic impacts on the underlying forwarding plane. Some of these flaws may be indeed directly exploited from the data plane, which brings more security challenges compared to conventional networks.

Figure 3 illustrates the topology of the SDN architecture that we use as a case study. We examine *the dynamicity in time* induced by the evolution of the potentialities and the accessibilities, as well as *the dynamicity in space* which is induced by adding a new device and cutting some accessibilities.

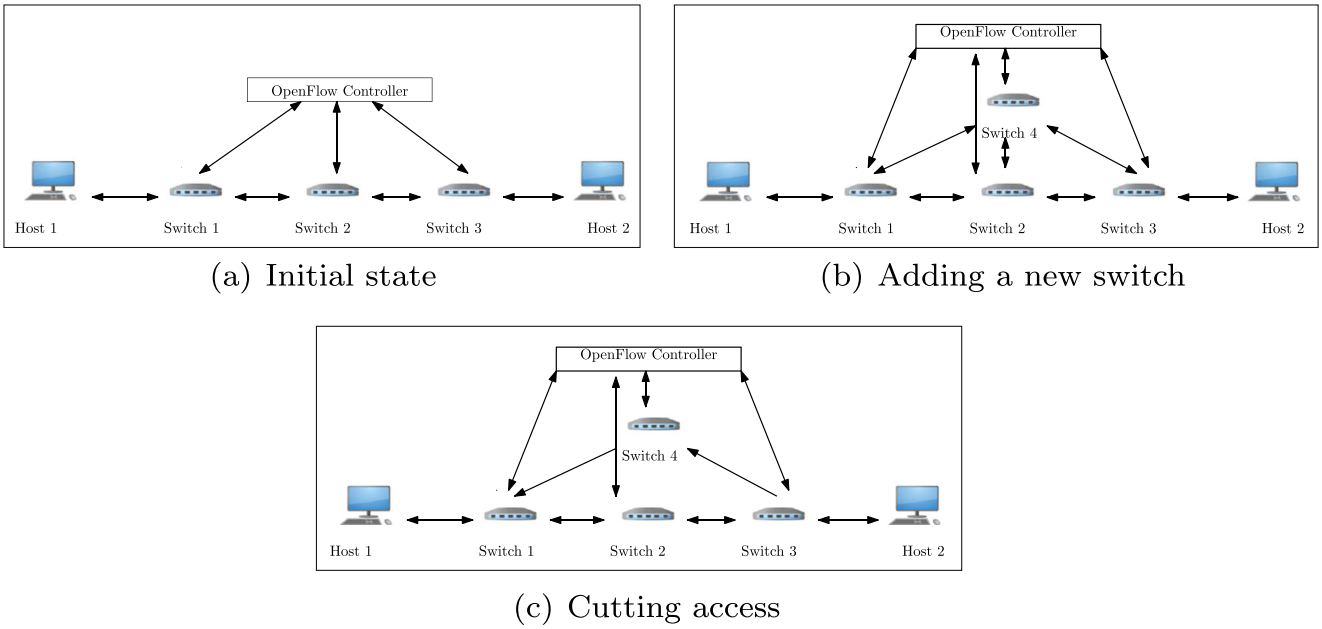
Figure 3a corresponds to the initial state of the system. Two hosts are connected to the network and may communicate through sharing network flows between them. Host 1 is connected to switch 1, host 2 is connected to switch 3, and switch 2 acts as a default gateway that connects switches 1 and 3. The flow transfer is supposed to be bidirectional inside the SDN data plane, as well as between the control plane and the data plane. The assets of the system (the controller and the switches) are $CISCO$ products, named using the standard Common Platform Enumeration (CPE). The CPE is used as a standardized method of describing and identifying classes of applications, operating systems, and hardware devices [33]:

- Controller (denoted by C): $cpe : /h : cisco : 2106\_wireless\_lan\_controller$;
- Switches 1 and 2 (denoted by s_1, s_2, and s_3): $cpe : 2.3 : h : cisco : nexus\_5548up$.

In Fig. 3b, a new switch s_4 is added to the SDN architecture. It is connected to the controller and the three other switches, in a mesh-like topology, through bidirectional links. The links between (s_4, s_3), (s_1, s_4), (s_4, s_2), and (s_2, s_4) are discarded in Fig. 3c. The construction of the RAGs associated with this system and the impact of the dynamicity of the system on the security metrics are examined in the following.

7.1 The risk assessment graphs

We show the construction and the visualization of the RAGs for the SDN case study introduced in Fig. 3. We study the



(a) Initial state

(b) Adding a new switch

(c) Cutting access

Fig. 3 SDN use case

system in the discrete time set $I = \{1, \ldots, 4\}$. Table 1 contains a detailed description of the vulnerabilities and their associated assets at the initial state of the system ($t = 1$). The exploitation and the impact of vulnerabilities are derived from the NVD database, as accessed in October 2016.

The potentiality function (1) is used to label the nodes of the RAGs. For this example, we assume that the accessibilities are previously determined. As seen in Fig. 4, they are equal to 1 for all the arcs, at $t = 1, 2$. At $t = 3$, the accessibilities between s_2 and all the other assets, as well as those between all the other assets and s_2 become 0.5. At $t = 4$, the accessibilities between (s_4, s_3), (s_1, s_4), (s_4, s_2), and (s_2, s_4) are equal to zero, and the corresponding arcs are dropped: $((s_4, v_3), (s_2, v_3))$, $((s_2, v_3), (s_4, v_3))$, $((s_1, v_3), (s_4, v_3))$, and $((s_4, v_3), (s_3, v_3))$.

The RAG instances that correspond to this use case are illustrated in Fig. 4, where each asset-vulnerability node is labeled with its appropriate potentiality. The arcs are labeled with the accessibility. An arc (n_1, n_2) is drawn if $g^t_{(n_1,n_2)} \neq 0$. The nodes (c, v_1) and (c, v_2) correspond to the same asset, and so the accessibility between them is always equal to 1. The red potentialities and links correspond to a change compared to the previous time slot. The nodes u_1 and u_2 correspond to the hosts 1 and 2. These are access points according to the RAG formalism, and they are represented with triangles in Fig. 4.

As in Fig. 4a, at $t = 1$, there are five asset-vulnerability nodes drawn as circles, and they are referred to as (c, v_1), (c, v_2), (s_1, v_3), (s_2, v_3), and (s_3, v_3). The corresponding initial potentialities p_w are derived from Table 1 which describes the vulnerabilities of the assets and their associated exploitation likelihood and impact.

At $t = 2$, and as illustrated in Fig. 4b, the switch s_4 whose associated vulnerability is v_3 and the arcs connecting

Table 1 Topology and vulnerability data basis mapping

Assets	Vul.	Name: summary	p_t	I_t
$Controller(C)$	v_1	CVE-2012-0368 : The administrative management interface on Cisco Wireless LAN Controller (WLC) devices allows remote attackers to cause a denial of service (device crash) via a malformed URL in an HTTP request	0.5	6.9
	v_2	CVE-2013-1235 : Cisco Wireless LAN Controller (WLC) devices do not properly address the resource consumption of terminated TELNET sessions, which allows remote attackers to cause a denial of service (TELNET outage) by making many TELNET connections and improperly ending these connections	0.5	2.9
$Switches1, 2(s_1, s_2, s_3)$	v_3	CVE-2013-5556 : The license-installation module on the Cisco Nexus 1000V switch allows local users to gain privileges and execute arbitrary commands	0.155	10

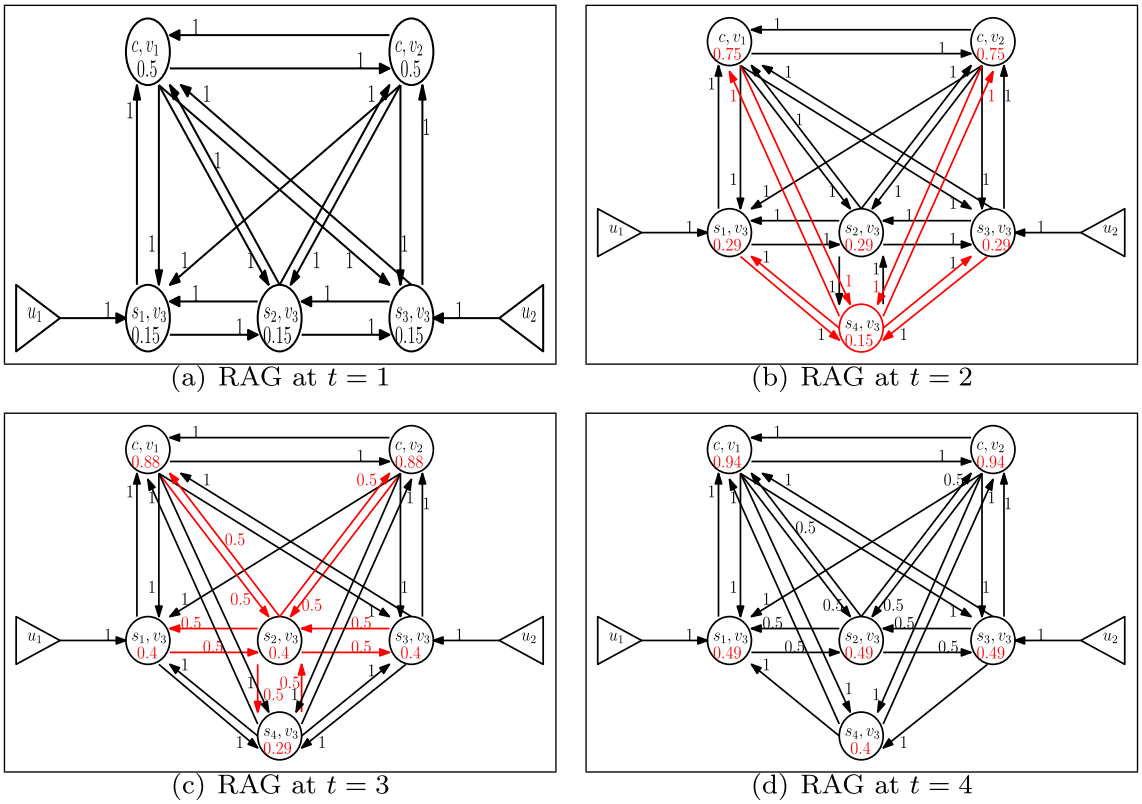


Fig. 4 SDN risk assessment graphs ($I = 4$)

(s_4, v_3) to the other nodes in the graph are added to the RAG. The potentiality of the nodes (c, v_1), (c, v_2), (s_1, v_3), (s_2, v_3), and (s_3, v_3) increases according to function (1). Since (s_4, v_3) appears only at $t = 2$, at this time slot, the node is labeled by the initial potentiality of v_3 which is 0.155 (see Table 1).

At $t = 3$, the accessibility between (s_2, v_3) and the other assets decreases from 1 to 0.5 according to Fig. 4c. Finally, at $t = 4$, the arcs $((s_4, v_3),(s_2, v_3)),((s_2, v_3),(s_4, v_3)),((s_1, v_3), (s_4, v_3))$, and $((s_4, v_3), (s_3, v_3))$ are deleted (see Fig. 4d).

7.2 Risk evaluation

The RAGs constructed in Section 7.1 are used by the Algorithm 1 in order to evaluate our security metrics. The results are presented in Figs. 5 and 6.

Figure 5 shows the variation of the nodes risk as a function of time. We discuss the risks for each node in the graph. We see that (c, v_1) node risk is higher than the one for (c, v_2) at all time slots. In fact, for the access points u_1 and u_2, the propagated potentiality to (c, v_1) or (c, v_2) has the same value. The factor making the (c, v_1) risk higher than the (c, v_2) risk is actually the impact (6.9 for (c, v_1) compared to 2.9 for (c, v_2)).

Having the same values of propagated potentiality and the same values of impact during all the time slots, the nodes (s_1, v_3) and (s_3, v_3) consequently have the same value of risk, as seen in Fig. 5.

The node (s_2, v_3) has a smaller risk than (s_1, v_3) and (s_3, v_3) for all time slots, even though they have the same values of exploitation p_w and impact I_w. This is explained by the fact that the intruder should pass by (s_1, v_3) (if it is u_1) or by (s_3, v_3) (if it is u_2) in order to reach (s_2, v_3). Therefore, the difficulty of propagation increases for the intruder. Consequently, the risk of the node (s_2, v_3) decreases.

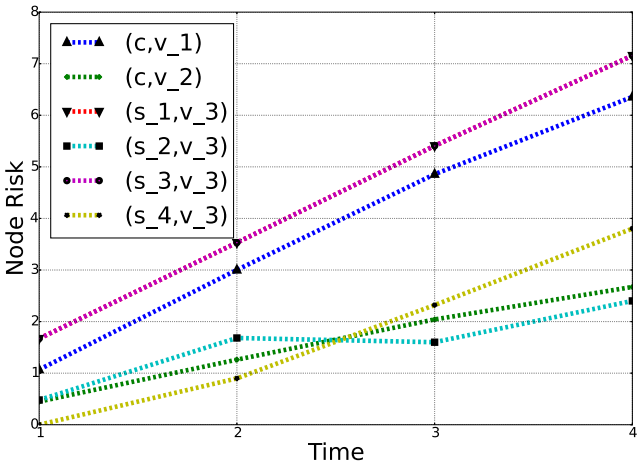


Fig. 5 Node risk as function of time

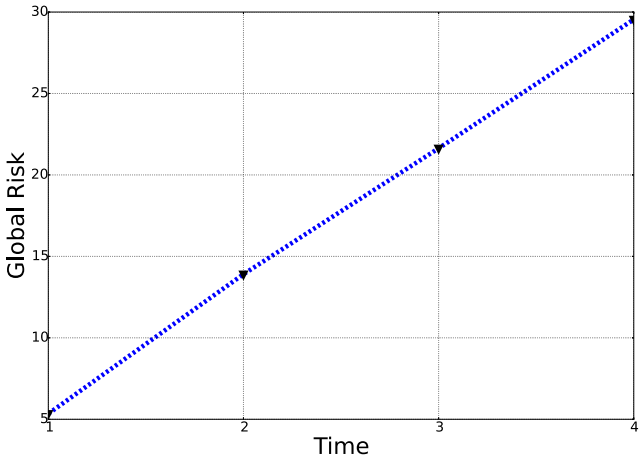


Fig. 6 Global risk as function of time

The risk of the node (s_4, v_3) at $t = 1$ is equal to 0 because the switch s_4 does not yet exist at this time slot and appears only at $t = 2$. At $t = 3$, the risk of the node (s_4, v_3) becomes higher than the risk of (s_2, v_3) even if the potentiality of (s_2, v_3) is higher than the one of (s_4, v_3) at this time slot $(0.4 > 0.29)$. In fact, the accessibilities between s_2 and all the other assets, as well as those between all the other assets and s_2, are dropped to 0.5 at $t = 3$. This implies a higher propagated risk to (s_2, v_3).

Figure 6 shows that the global risk of the system is increasing over time. This is mainly explained by the fact that all individual risks for all nodes in the RAG are also increasing over time.

8 Simulations

In this section, we present the simulation results. We randomly generated systems with a large number of nodes. The aim is to show, for random systems, the sensitivity of the mean global risk ($\sum_{t\in I} R^t$) to the number of nodes, the convergence speed of the potentialities, the topology, and the accessibilities. The experiments have been conducted on a computer equipped with an 2x Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60-GHz machine with 128 GB of RAM, running under Linux. We used python 2.7 as a programming language and Networkx [34] as a graph library.

8.1 Random systems generation

Our random systems are generated as follows. We first set $I = \{1, \ldots, 12\}$. The nodes, the arcs, and the parameters of the RAGs for each $t \in I$ are configured as follows.

(1) The sets U_t and W_t: the labels of the nodes in W_t are computed using Eq. 2. The parameter p_w is randomly generated using a continuous uniform distribution $U(0, 1)$, and the value of α_w varies in the set $\{0.1, 0.2, \ldots, 1\}$. We also set $\alpha_w = \alpha = cst$ for all $w \in W_t$.

(2) For each time slot in I, two specific subsets of arcs are randomly generated; the arcs induced by the nodes of W_t, denoted by $A_t(W_t)$, and those connecting the nodes of U_t with those of W_t, denoted by $A_t(U_t, W_t)$.

   (a) The set $A_t(W_t)$ is randomly generated using Erdös-Renyi graphs [35], in such a way that the sub-graph induced by the nodes of W_t is an Erdös-Renyi random graph of parameters W_t and p. This means that the graph is constructed by randomly connecting $|W_t|$ nodes, while each arc is included with probability p independent from every other arc.
   We set $p \in \{0.1, 0.2, \ldots, 0.9\}$.

   (b) The arcs $A_t(U_t, W_t)$ are added by connecting each $u \in U_t$ to one node in W_t, starting by the one having the biggest *out-degree*. The out-degree of a node is the number of its output arcs.

(3) The labels of the arcs are calculated based on Eq. 6. This requires the accessibilities as a parameter, which is simulated as an increasing function of time for these experiments and computed using the following equation:

$$g^t_{(n_1,n_2)} = a_{(n_1,n_2)} + (1 - a_{(n_1,n_2)})\frac{\beta(t-1)}{t}. \qquad (12)$$

Here, $a_{(n_1,n_2)}$ is the accessibility on (n_1, n_2) at the initial state of the system $(t = 1)$. This is randomly generated using a continuous uniform distribution $U(0, 1)$. The parameter β controls how fast the accessibility tends to 1. We vary β in the set $\{0.1, 0.2, \ldots, 1\}$, and we take the same value for all the arcs.

In the following, we will focus on the sensitivity of the mean global risk to the parameters $|V_t|$, p, β, and α. Recall that $|V_t|$ is the number of nodes in the RAG at time t. The parameter p gives an indication of the density of the links in the system top0ology. The speed of convergence of the accessibility is given by β, and the one of the accessibility is given by α.

8.2 Impact of the number of nodes

Let us now investigate the sensitivity of the mean global risk to the number of nodes. We set $\alpha = \beta = p = 0.5$, and $|U_t| = \frac{1}{2}|W_t|$. We vary $|V_t|$ in $[150, \ldots, 1500]$. The results plotted in Fig. 7 show a quasi-exponential growth of the mean global risk with respect to the number of nodes.

8.3 Impact of the topology and the accessibility changes *p* and *β*

We vary the parameters p and β as illustrated in Fig. 8. We observe that, for a fixed value of β, a variation of p from 0.1

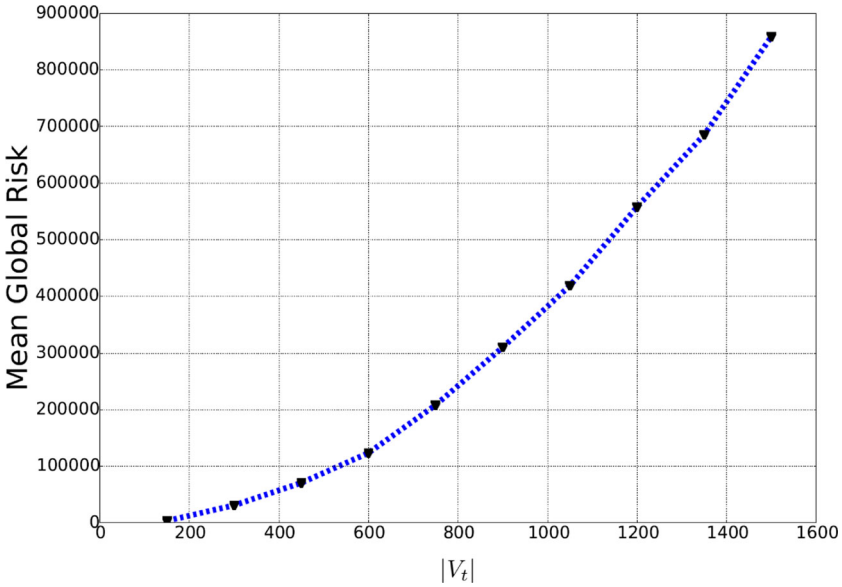


Fig. 7 Mean global risk as function of the number of nodes

to 0.9 implies an increase of the mean global risk by nearly 1000. On the other hand, for a fixed value of p, a variation of β from 0.1 to 0.9 yields an increase of the mean global risk by nearly 4000. This indicates that the parameter β has more impact on the mean global risk than parameter p. In other words, a sudden change in the accessibilities may have more impact on the global risk than a sudden change in the topology itself, for this set of random systems.

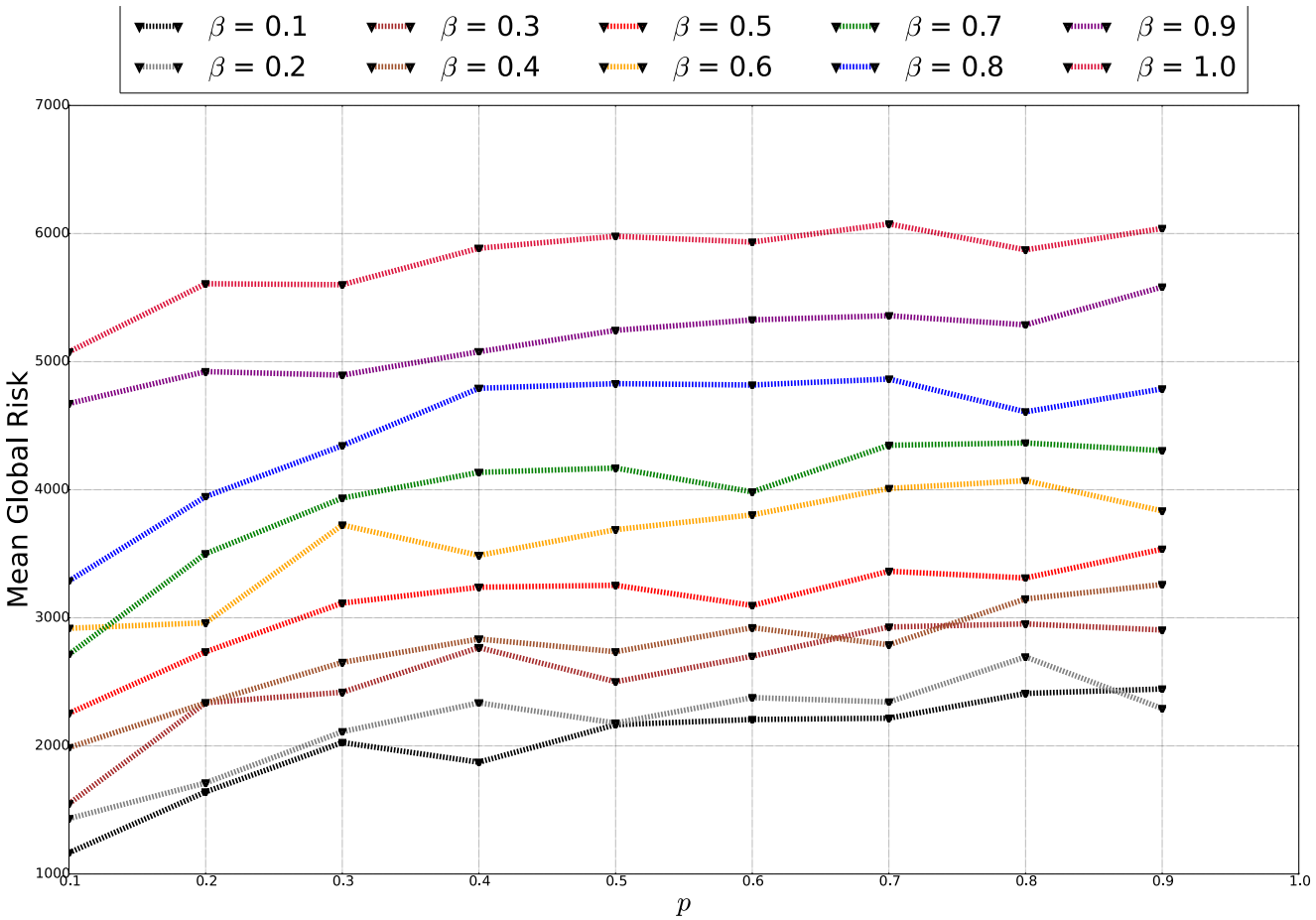


Fig. 8 Impact of the topology and the accessibility convergence speed (p and β)

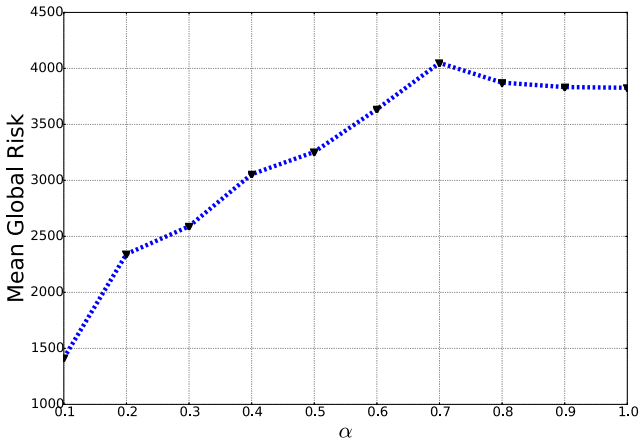


Fig. 9 Impact of the potentiality convergence speed α

8.4 Impact of the potentiality convergence speed α

We leverage the mean global risk evolution with respect to the parameter α. We set $|W_t| = 200, |U_t| = 20$, and $\beta = p = 0.5$. The variation of the mean global risk is illustrated in Fig. 9. This risk increases with the increase of the parameter α until $\alpha \leq 0.7$. When α has exceeded the threshold 0.8, the value of the mean global risk has slowly decreased. This is mainly explained by the fact that the topology also changes when varying the parameter α. While for this case, the probability of existence of a topological link p remains constant ($p = 0.5$), the links are less certain, and the realization of the random Erdös-Renyi sub-graph could generate a topology which prevents intruders to have higher propagation in the system.

9 Conclusion and future work

In this paper, we introduced a new risk assessment framework. In particular, we introduced the risk assessment graphs (RAGs), which include a model and formalism that capture a given system topology, including the assets, their accessibilities, the vulnerabilities, and the way all these elements evolve over time. We have also described a risk evaluation approach based on the propagation of the intruder threats across different vantage points in the system. We have defined three security metrics, namely the propagated risk, the node risk, and the global risk. Finally, we have demonstrated the use of our approach using an SDN testbed, and we conducted multiple experiments to evaluate the sensitivity of our metrics and the way they are affected by the size of the system, the vulnerability convergence properties, the topology, and the accessibilities between the system assets.

Our approach identifies the appropriate time where the global risk exceeds an acceptable threshold, and may also alert the operator in order to trigger relevant countermeasures. Nonetheless, while a countermeasure may contribute to reduce risks, it may also require substantial deployment and configuration costs. This motivates us to investigate more the use of combinatorial optimization techniques [36–38] in future work in order to elaborate intelligent risk mitigation actions, using the RAGs, which minimize risks and optimize costs.

Acknowledgements We would like to thank the anonymous referees for their valuable comments which permitted to improve the presentation of the paper.

Appendix A: Table of notations

In Table 2, we describe the different notations used in this paper.

Table 2 Table of notations

G_t	The RAG at time t
U_t	The set of access points
W	The set of asset-vulnerability nodes
Λ_t	The set of assets
V_a^t	The set of vulnerabilities on an asset $a \in \Lambda_t$
A_t	The set of arcs at the time t
A_f	The set of fixed arcs at the time t
A_u^t	The set of uncertain arcs at time t
p_w	The exploitation of a node w
α_w	The convergence speed of the potentiality function
I_w	The impact a node w
$f_w^t(\alpha_w)$	The potentiality function of a node w at the time t
β	The convergence speed of the accessibility increasing function
$g_{(n_1,n_2)}^t$	The accessibility function between n_1 and n_2 at the time t
$h_{(n_1,n_2)}^t$	The propagation function between the nodes n_1 and n_2 at the time t
$P_{u,w}^{\pi,t}$	The propagated potentiality in the $u-w$ path π at the time t
$P_{u,w}^t$	The most likely $u-w$ path value at the time t
F_w^t	The exploitation difficulty function of the node w at the time t
$H_{(n_1,n_2)}^t$	The propagation difficulty function between the nodes n_1 and n_2 at the time t
$R_{u,w}^t$	The propagated risk from an access point u to a node w at time t
R_w^t	the risk of a node w at time t
R^t	The global risk at time t

Appendix B: Future work

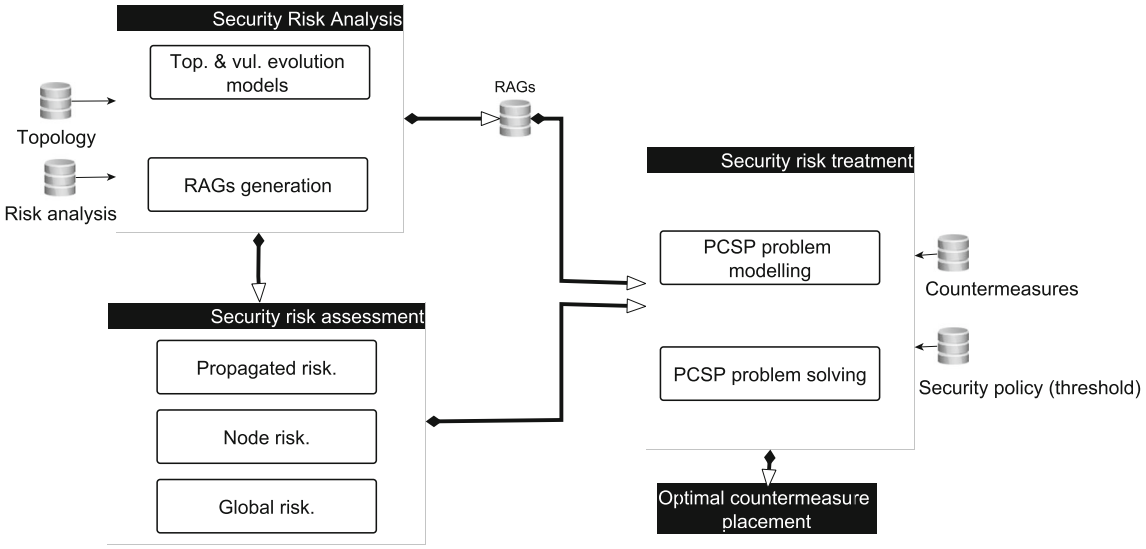


Fig. 10 Complete risk management framework

Future work will expand our approach described in this paper through integrating a risk treatment step. A possible illustration of the entire process is provided in Fig. 10. The risk treatment process deals with the following Proactive Countermeasure Selection Problem (PCSP): Given the RAGs, the countermeasures and the security policies (thresholds), find an assignment of countermeasures to the asset-vulnerability nodes that both respects the security policies and minimizes the cost of its deployment. The solution of the problem may be conducted in two steps.

PCSP problem modeling A mathematical programming formulation will be given to model the PCSP.

PCSP problem solving Based on the formulation, efficient optimization algorithms will be developed to solve the problem. The solver Cplex [39] will be used.

A preliminary work related to this problem is published in [40].

References

1. Purdy G (2010) ISO 31000: 2009—setting a new standard for risk management. Risk Anal 30(6):881–886
2. EBIOS, Central directorate for information systems security, version, http://www.ssi.gouv.fr
3. Alberts CJ, Behrens SG, Pethia RD, Wilson WR (1999) Operationally critical threat, asset, and vulnerability evaluation (OCTAVE) framework. Version 1.0
4. Mell P, Scarfone K, Romanosky S (2007) A complete guide to the common vulnerability scoring system version 2.0. Published by FIRST-forum of incident response and security teams, 1–23
5. Sheyner OM (2004) Scenario graphs and attack graphs (Doctoral dissertation, US Air Force Research Laboratory)
6. Bondy JA, Murty USR (1976) Graph theory with applications, vol 290. London: Macmillan
7. West DB (2001) Introduction to graph theory, vol 2. Upper Saddle River: Prentice hall
8. NIST, National institute of science and technology, http://nvd.nist.gov/download.cfm
9. Phillips C, Swiler LP (1998) A graph-based system for network-vulnerability analysis. In: Proceedings of the 1998 workshop on new security paradigms, pp 71–79
10. Ou X, Boyer WF, McQueen MA (2006) A scalable approach to attack graph generation. In: Proceedings of the 13th ACM conference on computer and communications security, pp 336–345
11. Ammann P, Wijesekera D, Kaushik S (2002) Scalable, graph-based network vulnerability analysis. In: Proceedings of the 9th ACM conference on computer and communications security, pp 217–224
12. Huang H, Zhang S, Ou X, Prakash A, Sakallah K (2011) Distilling critical attack graph surface iteratively through minimum-cost sat solving. In: Proceedings of the 27th annual computer security applications conference, pp 31–40
13. Viduto V, Huang W, Maple C (2011) Toward optimal multi-objective models of network security: survey. In: Automation and computing, ICAC, pp 6–11
14. Xie P, Li JH, Ou X, Liu P, Levy R (2010) Using Bayesian networks for cyber security analysis. In: IEEE/IFIP international conference on dependable systems and networks, 2010, pp 211–220
15. Mehta V, Bartzis C, Zhu H, Clarke E, Wing J (2006) Ranking attack graphs. In: Recent advances in intrusion detection, pp 127–144
16. Kijsanayothin P, Hewett R (2010) Analytical approach to attack graph analysis for network security. In: ARES'10 international conference on availability, reliability, and security, pp 25–32
17. Wing JM et al. (2008) Scenario graphs applied to network security. In: Information assurance: survivability and security in networked systems, pp 247–277
18. Homer J, Zhang S, Ou X, Schmidt D, Du Y, Rajagopalan SR, Singhal A (2013) Aggregating vulnerability metrics in enterprise networks using attack graphs. J Comput Secur 21(4):561–597

19. Lippmann RP, Ingols KW (2005) An annotated review of past papers on attack graphs (No. PR-IA-1). Massachusetts Inst Of Tech Lexington Lincoln Lab
20. Hong J, Kim D-S (2012) HARMs: hierarchical attack representation models for network security analysis. Security Research Institute, Edith Cowan University, Perth, Western Australia
21. Wang S, Zhang Z, Kadobayashi Y (2013) Exploring attack graph for cost-benefit security hardening: a probabilistic approach. Comput Secur 32:158–169
22. Samarji L, Cuppens F, Cuppens-Boulahia N, Kanoun W, Dubus S (2013) Situation calculus and graph based defensive modeling of simultaneous attacks. In: Cyberspace safety and security, pp 132–150
23. Common vulnerabilities and exposures, CVE, http://cve.mitre.org/
24. Van Benthem J (2011) Logical dynamics of information and interaction. Cambridge University Press
25. Noel S, Jajodia S, O'Berry B, Jacobs M (2003) Efficient minimum-cost network hardening via exploit dependency graphs. In: 19th annual computer security applications conference proceedings, pp 86–95
26. Jakobson G (2011) Mission cyber security situation assessment using impact dependency graphs. In: Proceedings of the 14th international conference on information fusion (FUSION), pp 1–8
27. Kheir N, Cuppens-Boulahia N, Cuppens F, Debar H (2010) A service dependency model for cost-sensitive intrusion response. In: Computer security–ESORICS, pp 626–642
28. Shandilya V, Simmons CB, Shiva S (2014) Use of attack graphs in security systems. Journal of Computer Networks and Communications, 2014
29. Yassine NM, Nancy P, Nizar K, Mahjoub AR, Wary JP (2016) A new risk assessment framework using graph theory for complex ICT systems. In: Proceedings of the 2016 international workshop on managing insider security threats. ACM, pp 97–100
30. Baras JS, Theodorakopoulos G (2010) Path problems in networks. Synthesis Lectures on Communication Networks 3(1):1–77
31. Floyd RW (1962) Algorithm 97: shortest path. Commun ACM 5(6):345
32. Ahmad I, Namal S, Ylianttila M et al. (2015) Security in software defined networks: a survey. IEEE Commun Surv Tutorials 17(4):2317–2346
33. Common platform enumeration, CPE, https://cpe.mitre.org/
34. Networkx documentation, https://networkx.github.io/documentation/networkx-1.9.1/
35. Erdös P, Rényi A (1959) On random graphs, I. Publicationes Mathematicae (Debrecen) 6:290–297
36. Ben-Tal A, El Ghaoui L, Nemirovski A (2009) Robust optimization. Princeton University Press
37. Schrijver A (2002) Combinatorial optimization: polyhedra and efficiency, vol 24. Springer Science & Business Media
38. Dantzig GB (1998) Linear programming and extensions. Princeton University Press
39. IBM ILOG CPLEX Optimizer, http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/
40. Mahjoub AR, Naghmouchi MY, Perrot N (2017) A bi-level programming model for proactive countermeasure selection in complex ICT systems, INOC. Lisbonne, Portugal